ADTECH 众为兴

Leave the work to the machine, and keep the thought to yourself

AMC1600 User's Guide (Motion Control)

Adtech (Shenzhen) Technology Co., Ltd.

Add: F/5, Bldg/27-29, Tianxia IC Industrial Park, Yiyuan Rd, Nanshan District, Shenzhen

Postal code：518052

Tel：0755-26722719　Fax：0755-26722718

email：tech@adtechcn.com　http：//www.adtechcn.com

Copyright Notice

ADTECH (SHENZHEN) TECHNOLOGY CO., LTD

Basic information of the manual

Instruction manual version number：V1.0

Precautions and Explanations

※ Transport and storage

- Do not stack product package more than six layers;
- Do not climb, stand on or place heavy stuff on the product package
- Do not pull the cable still in connection with the machine or move product;
- Forbid hitting and scratching; the panel and display;
- Prevent the product package from humidity, sun exposure, and rainwater.

※ Unpacking inspection

- Unpack to confirm if the product is purchased by you;
- Check if the product is damaged during transportation
- Confirm the integrity of parts comparing with the parts list and damages;
- Contact us promptly for discrepant models, shortage accessories, or transport damages.

※ Wiring

- Ensure the person involved in wiring and inspecting is a specialized staff;
- Guarantee the product is grounded with less than 4Ω grounding resistance. Do not use the neutral line (N) to substitute earth wire;
- Ensure the wiring be correct and solid to avoid product failures or unexpected consequences;
- Connect the surge absorption diodes connected with the product in the required direction, or the product would be damaged;
- Ensure the power switch is OFF before plugging, unplugging, or opening the product case.

※ Overhauling

- Ensure the power is OFF before overhauling or components replacement;
- Make sure to check failures after short circuit or overloading, and then restart the machine after troubleshooting
- Do not frequently disconnect the power, and do reconnect after disconnection. The disconnection - reconnection interval should be at least one minute.

※ Miscellaneous

- Do not open housing without permission;
- Keep power OFF if not in use for a long time;
- Pay close attention to keep dust and ferrous powder from entering the controller;
- Parallel freewheel diode on the relay coil in parallel if a non-solid state relay is used as the output relay. Check if the power supply meets the requirement to ensure no burning of the controller.
- Install cooling fan if the processing field is too hot as the service life of the controller is closely related to the environmental temperature. Keep proper operative temperature range for the controller: -20℃ - 60℃.

- Avoid using the product in the overheating, humid, dusty, or corrosive environments;
- Add rubber cushion where vibration is strong.

※ Maintenance

- Please implement routine inspection and regular check upon the following items, under the general usage conditions (i.e. environmental condition: daily average 30℃, load rate: 80%, and operating rate: 12 hours/ day)

Routine Inspection	Routine	Confirm the environmental temperature, humidity, dust, or foreign materials; Confirm abnormal vibration and noise; Check whether vents are blocked by yarn, etc.
Regular check	One year	Check whether solid components are loose; Confirm whether the terminal block is damaged.

Directory

Chapter I Introduction to PLCopen

Standard

PLCopen International is a worldwide organization independent of manufacturers and products. Its mission is to solve those topics related to control programming to support the use of international standards in this field.

PLCopen International was established in 1992 and is headquartered in the Netherlands. It also has branches in Ontario, Canada and Tokyo, Japan. The purpose of PLCopen International is to promote the development and use of PLC-compatible software.

A major activity of PLCopen is to work on the global standardization of IEC61131-3 (the international standard specification for PLC programming). It standardizes the programming interface and harmonizes the way people design and engage in industrial control. The standard programming interface allows people of different backgrounds and skills to create programs with different elements at different stages of the software life cycle: technical specifications, design, implementation, testing, installation and maintenance. However, they all follow a common structure and work together in harmony. The standard defines the SFC (Sequential Function Chart) language and four interoperable programming languages used to construct the internal structure of the program: IL (instruction list), LD (ladder diagram), FBD (function block diagram) and ST (structured text) . By decomposing each program into logical elements, modularization and modern software technology, it improves its reusability, reduces errors, and improves programming and user efficiency. The standard also standardizes the function blocks used in motion control, which include functions such as single-axis positioning, electronic cam functions, and multi-axis interpolation control. At the same time, it defines the basic steps of starting instructions. With PLCopen motion control function blocks, the dependence on hardware can be reduced and the reuse rate of user programs can be improved. It can also reduce the cost of training, support, etc.

SAS is a programming software developed by Shanghai STEP Electric Corporation , which based on the CODESYS platform of the German 3S company and fully complies with PLCopen programming specifications.

Chapter II Components of a Typical

AMC1600 Control System

The AMC1600 is a high-performance industrial bus controller developed based on SAS software programming. It features powerful SoftMotion motion control, and supports functions single-axis positioning, CAM, CNC, multi-axis interpolation control and so on. AMC1600 has abundant local resources (6-ch standard pulse interfaces, 32-ch inputs, 24-ch outputs, etc.), and supports also Ethercat bus communication. The bus servo and bus I/O devices supporting Ethercat communication can be connected through the Ethercat interface of the controller.

The components of typical control system are shown in the figure below:

The above configuration allows AMC1600 to connect to multiple QXE servos and ET1616A bus I/O modules of our standard EtherCAT bus communication.

The AMC1600 control system based on CODESYS is provided with powerful motion controls.

For Ethercat bus communication, the system takes the clock of the first slave connected to the Ethercat master as the reference synchronous clock of the network. Therefore, if there're servo devices and IO devices connected to the bus, the first slave device must be mounted with a servo device, so as to ensure the stability of the reference synchronization clock.

Chapter III Components of AMC1600

Platform User Program

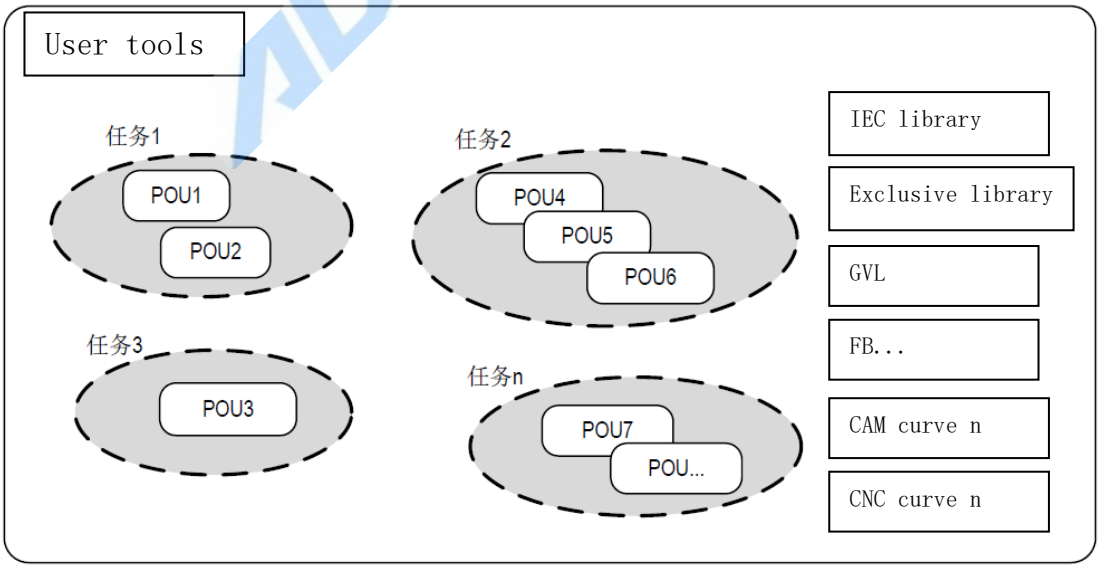
3.1 Structure of user program

AMC1600 is a multi-task mechanism motion control system based on CODESYS platform. There are many types and priorities of system tasks. The user program can be divided into multiple task groups to be executed separately according to the task type and priority set by the user.

In the actual programming, the user can create a variety of tasks and assign the POU programs to the corresponding tasks based on the emergency and applications of the program execution.

3.1.1 Components of user program

AMC1600 uses a multi-task execution scheme, under which each user program may be assigned with multiple tasks, and each task may have several program organization units (POU). The typical structure model diagram is as follows:



IEC library

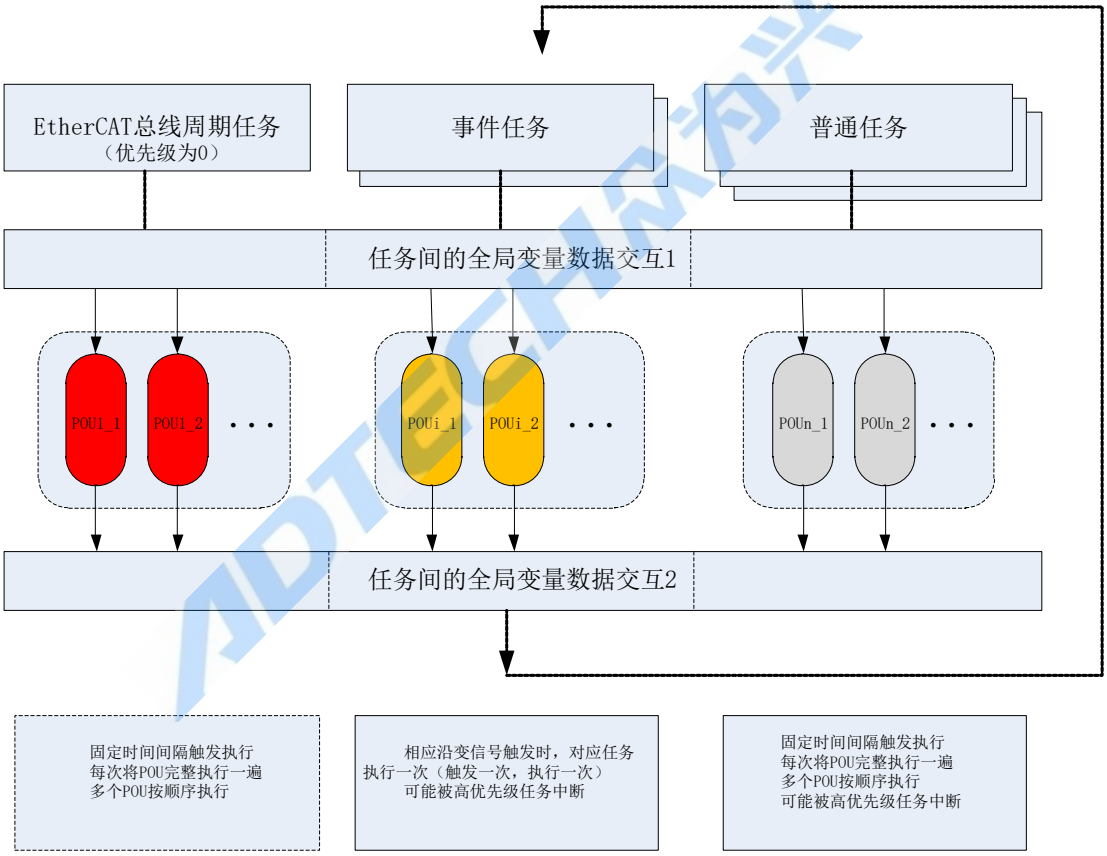
The user engineering is composed of several POUs, which can be divided into several task groups based on characteristics of the POU. POUs that are

not assigned to tasks will not be executed. In addition, there're some objects used to support the user program in the engineering, such as library functions, global variables, CAM table (defining cam table), CNC program (defining CNC track), etc., which are also the integral part of the user engineering.

3.1.2 Types of tasks

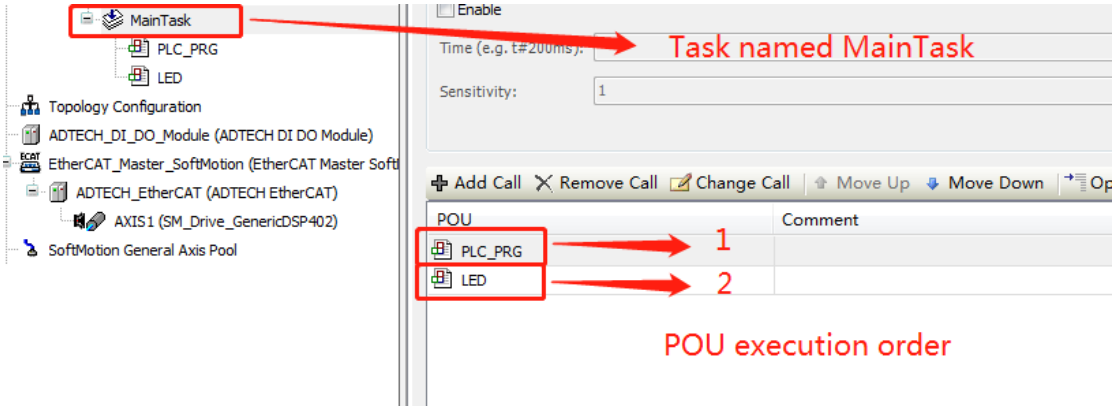
Task configuration is to divide the user program into several task groups based on the execution requirements. Different execution trigger conditions, execution time intervals, and priority can be set for each task group.

Common tasks in user programs may include bus cycle task of EtherCATs, ordinary cyclic tasks, and event tasks. The motion-related subject is arranged to be executed under the bus cycle task of EtherCAT.



The bus cycle task of EtherCATs is the most important task in the program. The real-time processing of motion control is performed in this task. It is a task with a short execution time interval and the highest priority. Once the conditions are satisfied, it will interrupt other tasks and start execution, and will not exit until all POUs under this task are executed.

Under each task, multiple POUs can be specified, and these POUs are executed one by one in order. The following figure shows the task configuration of a project:



In the above figure, two POUs under the MainTask task can be executed in order as follows, PLC_PRG, LED. When there's a global update and judgment, you need to arrange the order appropriately.

3.1.3 Multiple POUs programming in user program

Programs processed in different cycles of execution should be written in different POUs. For example: the POU executed under the bus cycle task of EtherCAT, the POU executed under the event task, and the POU executed under the ordinary task with a cycle of 20ms must be programmed using a separate POU.

To improve the readability of the project, different POUs will be used for different control process sections and different operation objects. Each POU is named to allow easy understanding.

When multiple programmers are working together in programming, each programmer writes and debugs the POU of the process section that they are responsible for, and finally to form a user program.

SAS software supports 6 programming languages. As the required processing logics differ, a certain language may be more convenient. Generally, each POU can only be written in one programming language. When multiple languages are required for a project, it is recommended to program by dividing into multiple POUs.

3.1.4 Motion control and logic control of a user program

Users should pay attention to the priority of the task when programming. They should arrange the motion instructions and the logic that requires quick motion in the bus cycle task of EtherCAT, and assign other logic controls to the

ordinary cyclic task as much as possible, To enhance the execution efficiency of bus cycle task of EtherCAT.

In actual motion control, high real-time requirements are imposed to synchronous control and trajectory control, while the response speed of logic control is relatively low. In the user program, the motion control program should be assigned to the bus cycle task, while the logic control program should be assigned to a normal task. The logic control program performs interaction with the motion control program through specific global variables, to allow the synergic motion of logic control and motion control.

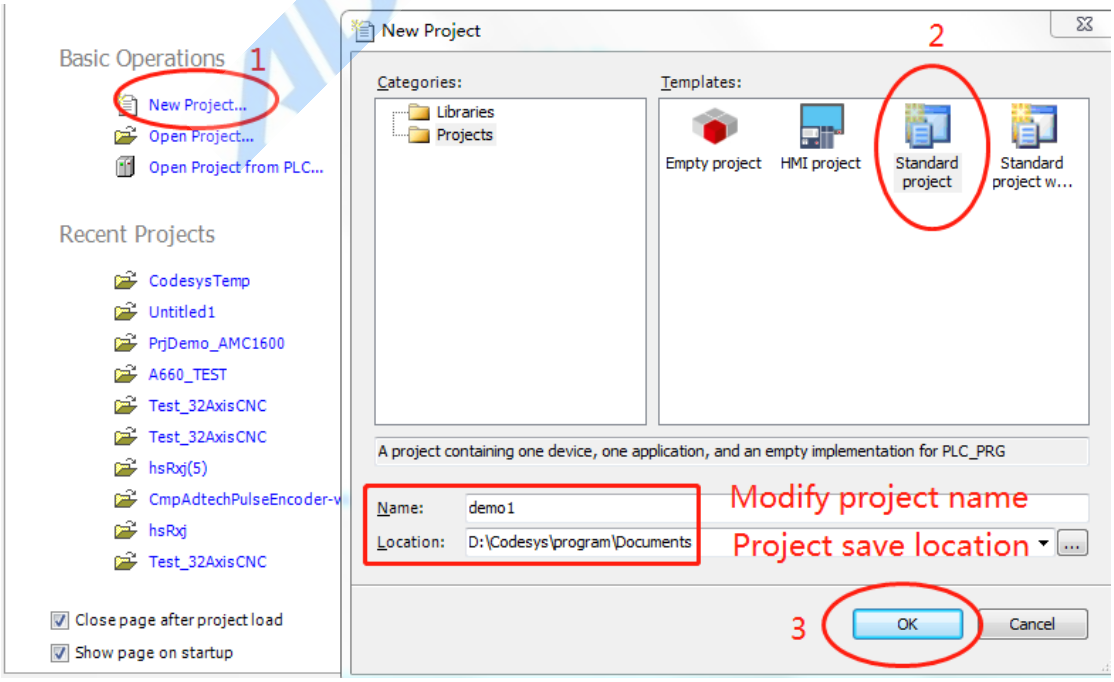
3.2 Simple user engineering example

Requirement: Axis 1 QXE servo forward and reverse; Axis 2 QXE servo JOG operation; the output in the local IO executes the marquee motion in a cycle of 500ms; the even numbers (0, 2, 4, 6, 8, 10, 12, 14) and odd numbers (1, 3, 5, 7, 9, 11, 13, 15) of 16-ch outputs of ET1616A alternately light up in a cycle of 500ms.

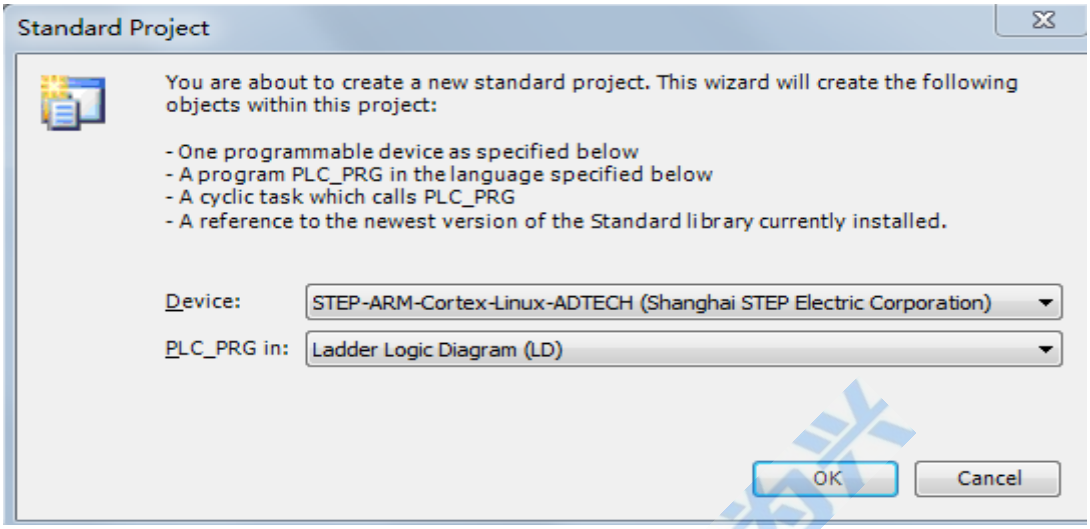
3.2.1 Build a new project named Demo1 and the involved equipment of AMC1600

✧ Create a standard project

1. Create a new standard project.

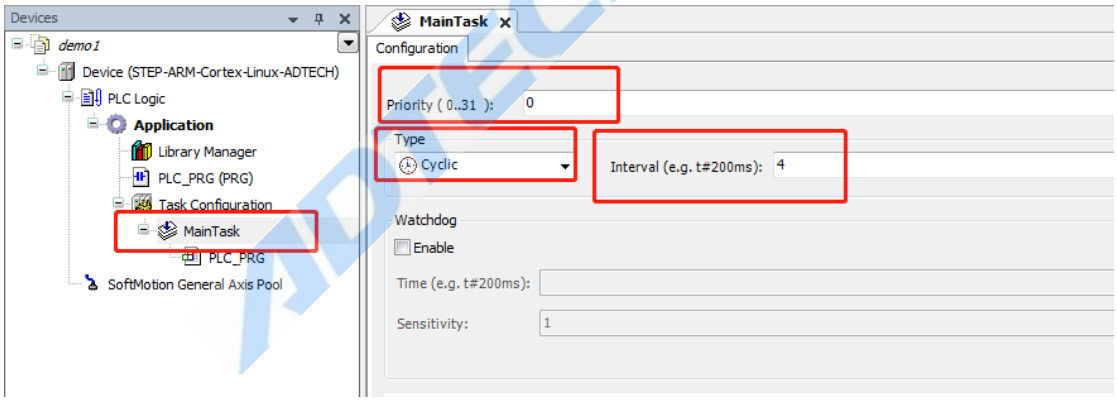


2. Specify the equipment of the project as "STEP-ARM-Cortex-Linux-ADTECH (Shanghai STEP Electric Corporation)", and designate the first POU as "Ladder Logic Diagram (LD)" (the form of POU can be determined as per the programming requirements, of which six types are optional).



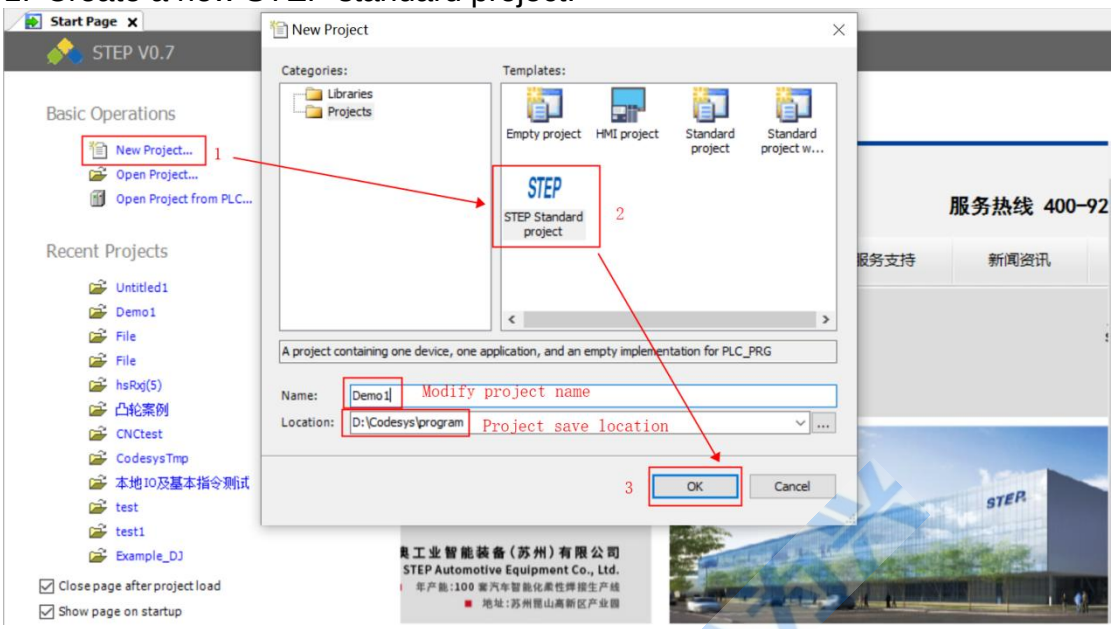
3. Task attribute settings of MainTask

In this project, "MainTask" is regarded as the "bus cycle task of EtherCAT". (The MainTask priority is set to 0 here, and when adding devices, its "bus cycle task" is designated as "MainTask")

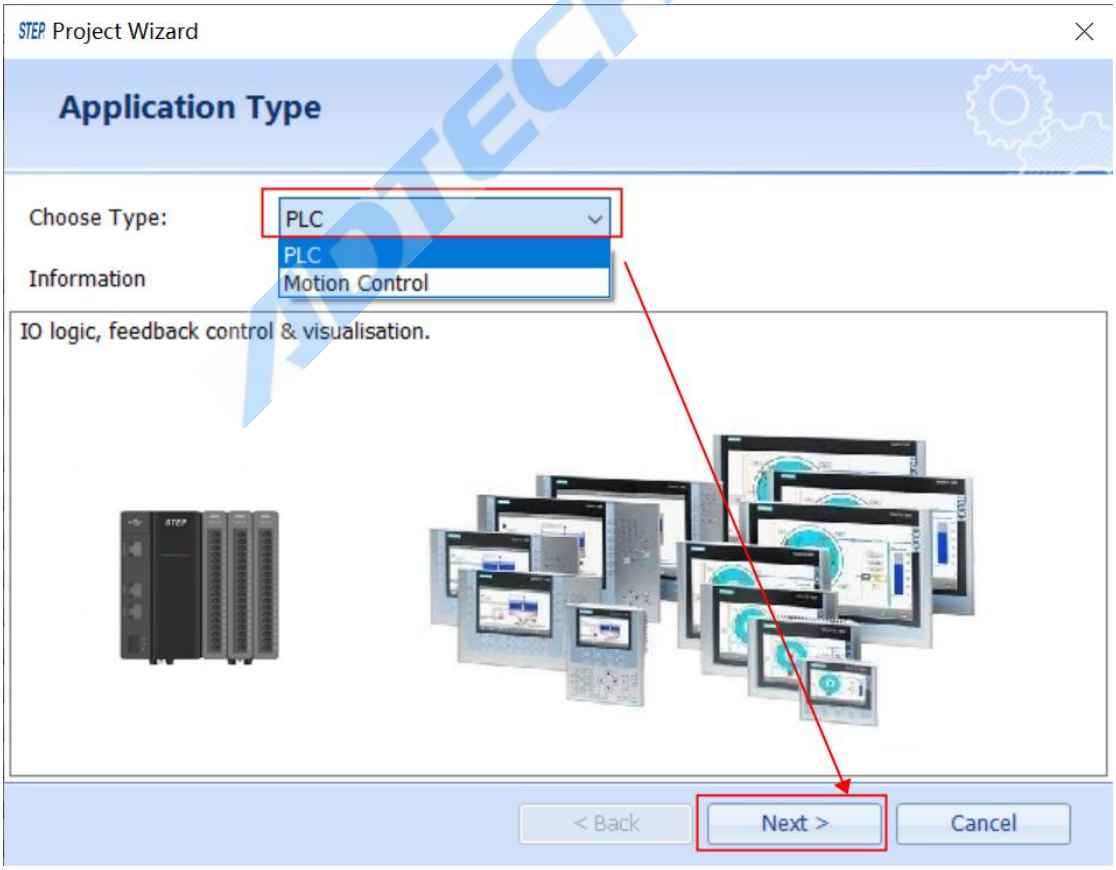


✧ Create a STEP standard project

1. Create a new STEP standard project.

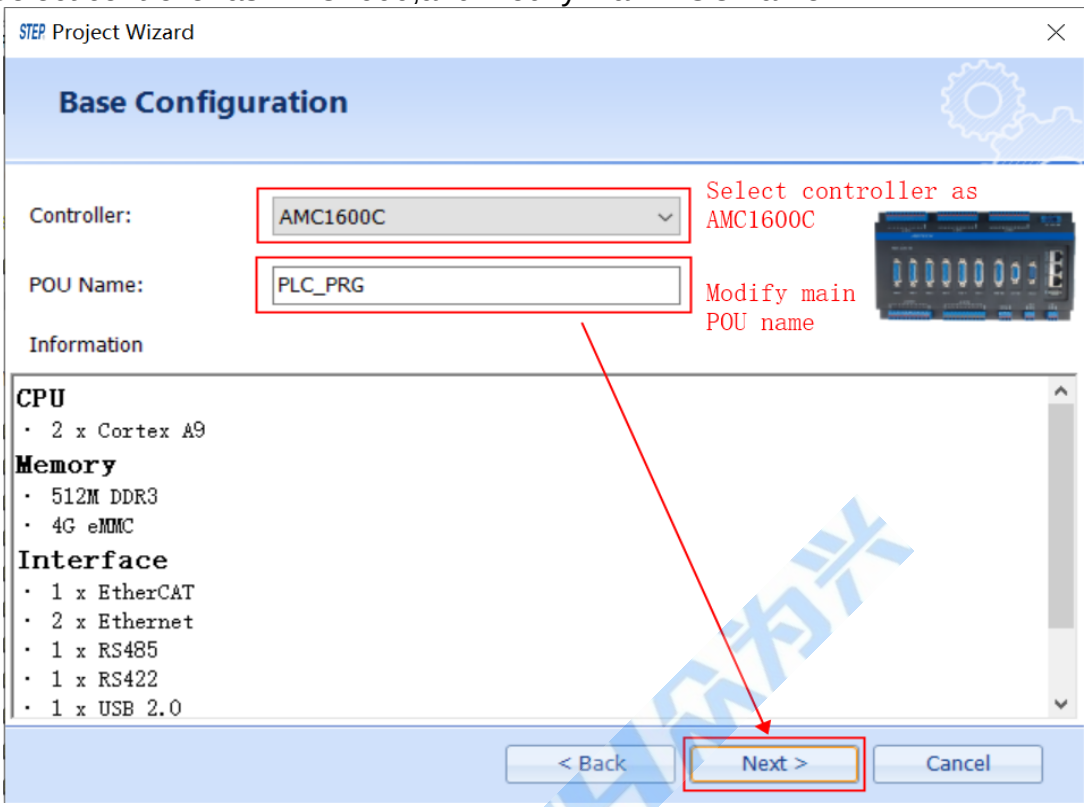


2. Application type
Select the application type as PLC.



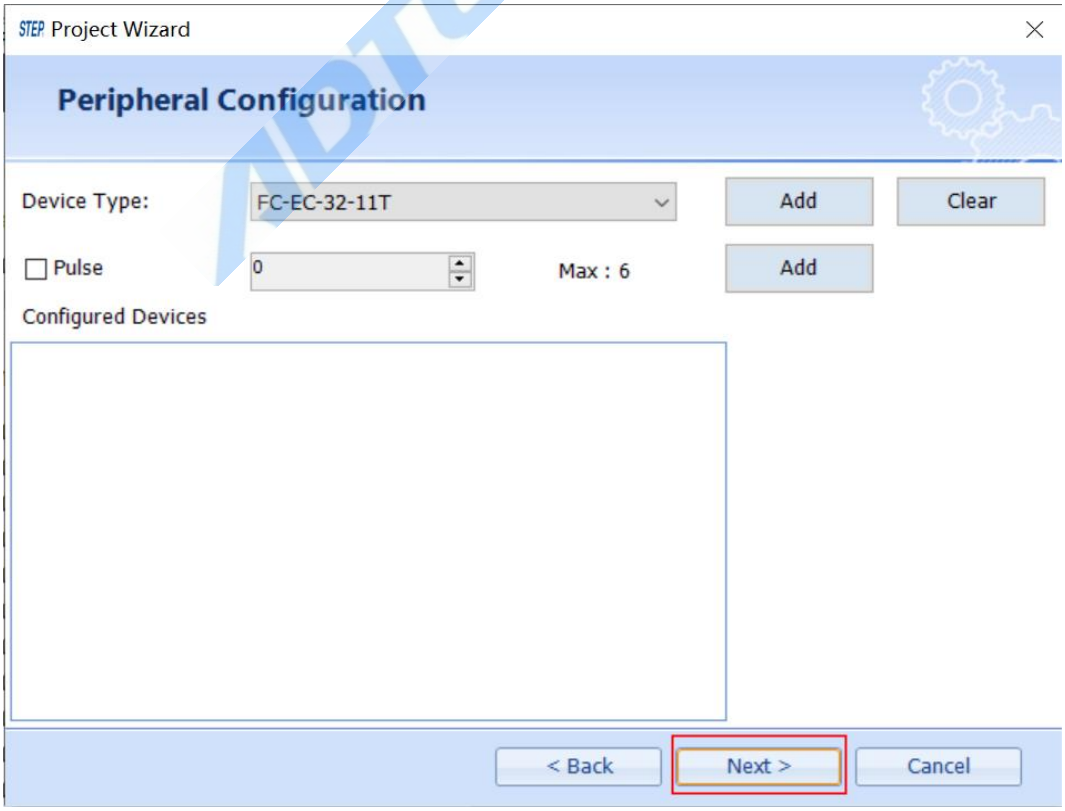
3. Base Configuration

Select controller as AMC1600,and Modify main POU name.



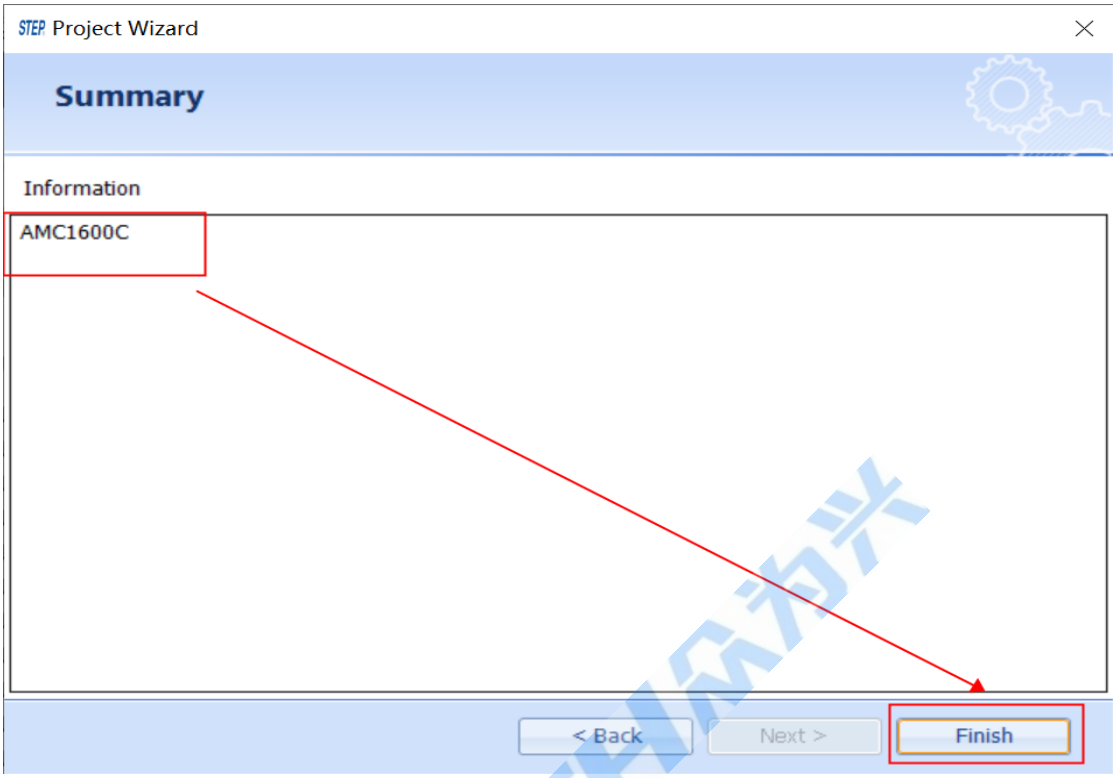
4. Peripheral configuration

Klick Next.



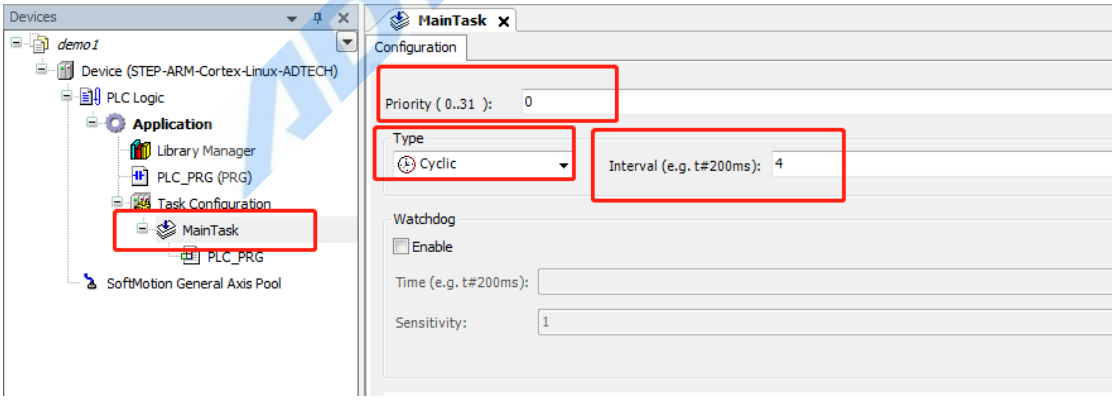
5. Summary

Confirm that the information box is correct and click finish.



6.Task attribute settings of MainTask

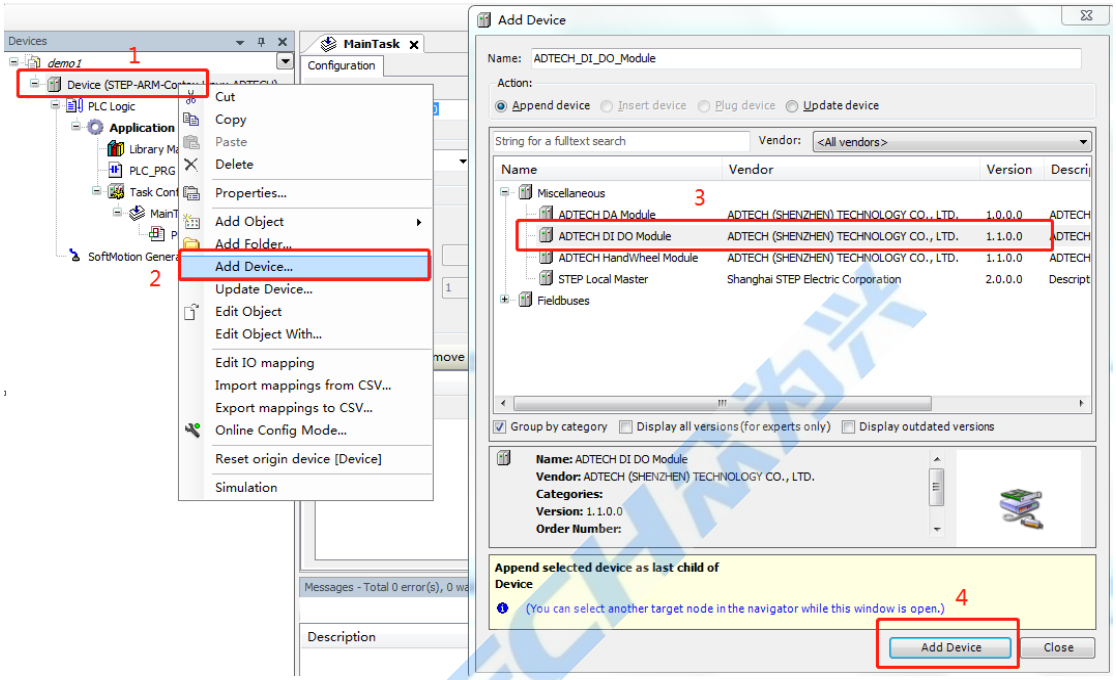
In this project, "MainTask" is regarded as the "bus cycle task of EtherCAT". (The MainTask priority is set to 0 here, and when adding devices, its "bus cycle task" is designated as "MainTask")



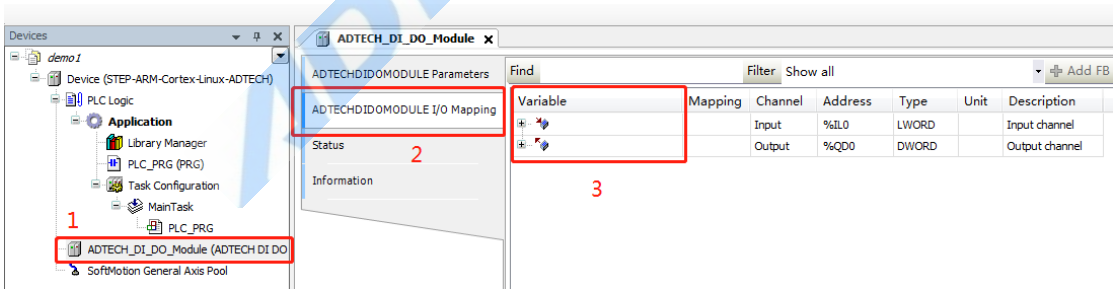
3.2.2 Adding equipment

3.2.2.1 Local I/O device addition

1. Add local I/O devices.



2. Perform variable mapping to local I/O addresses.

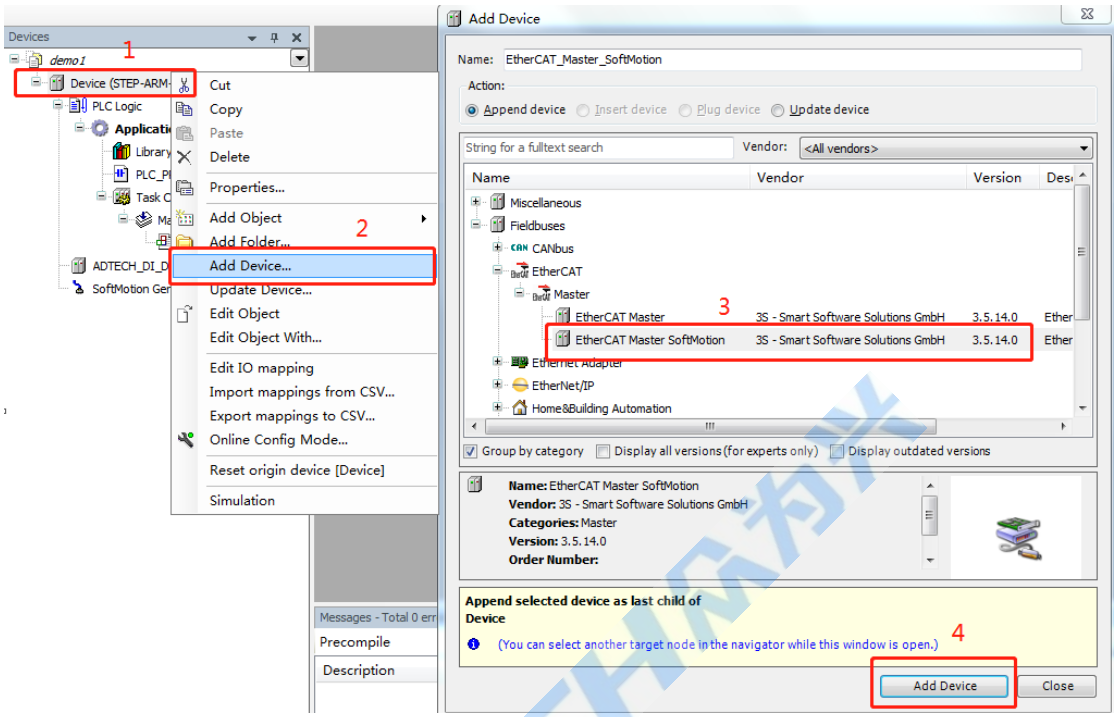


3. Local I/O task settings

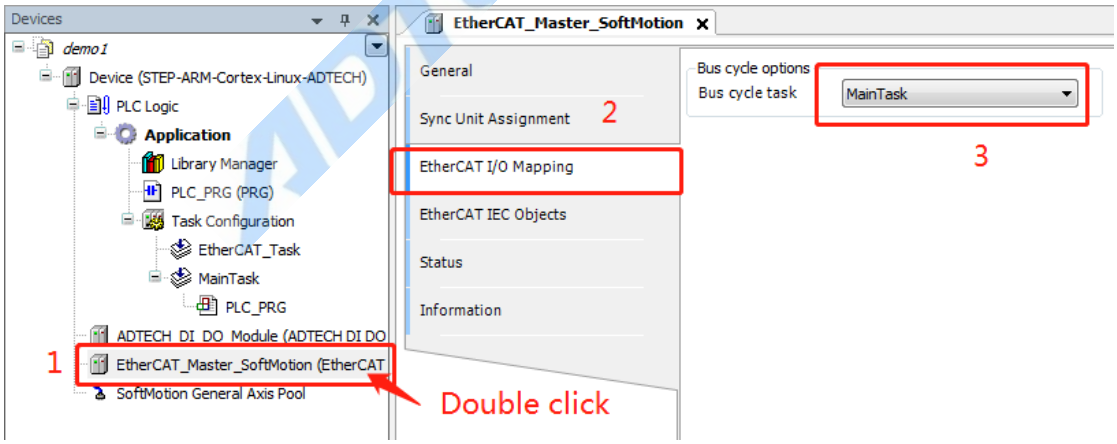
Always update variables: enable 2 (always in the bus cycle task);
Bus cycle task: MainTask; (local I/O will be updated under this task)

3.2.2.2 Add EtherCAT device

1. Create a new EtherCAT master device

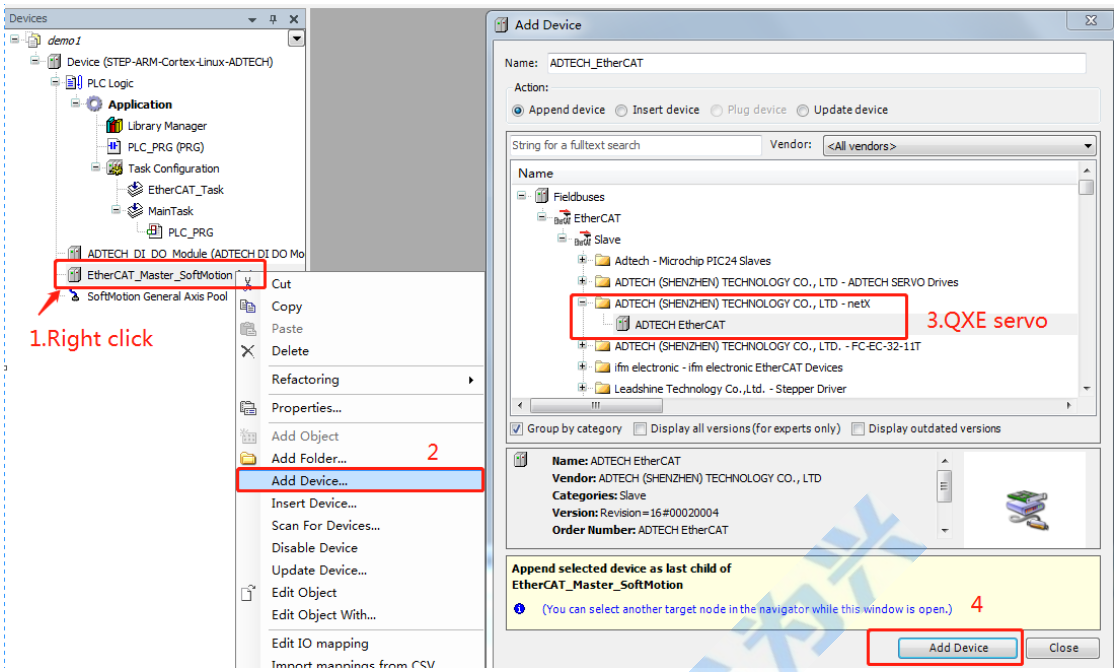


2. EtherCAT master I/O mapping setting

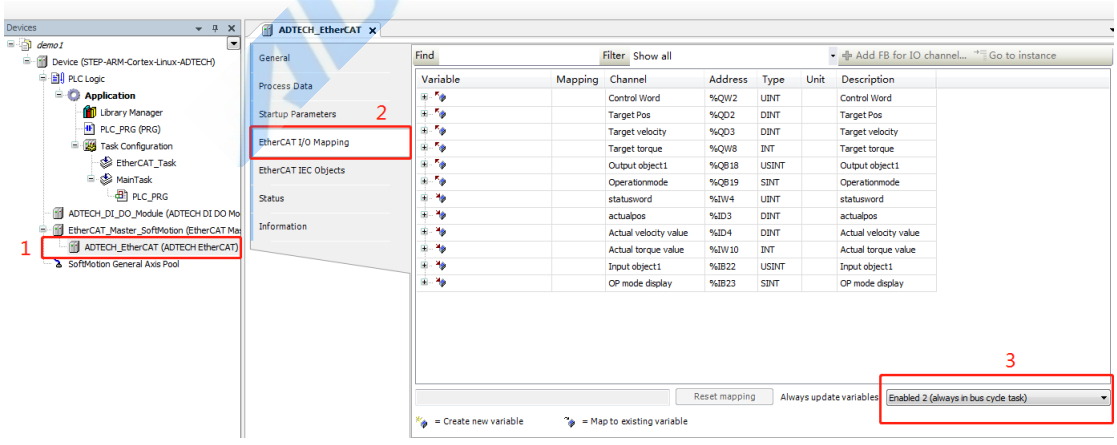


EtherCAT I/O mapping-bus cycle task: MainTask (in this project, MainTask is regarded as bus cycle task of EtherCAT);

3. Add QXE servo slave device

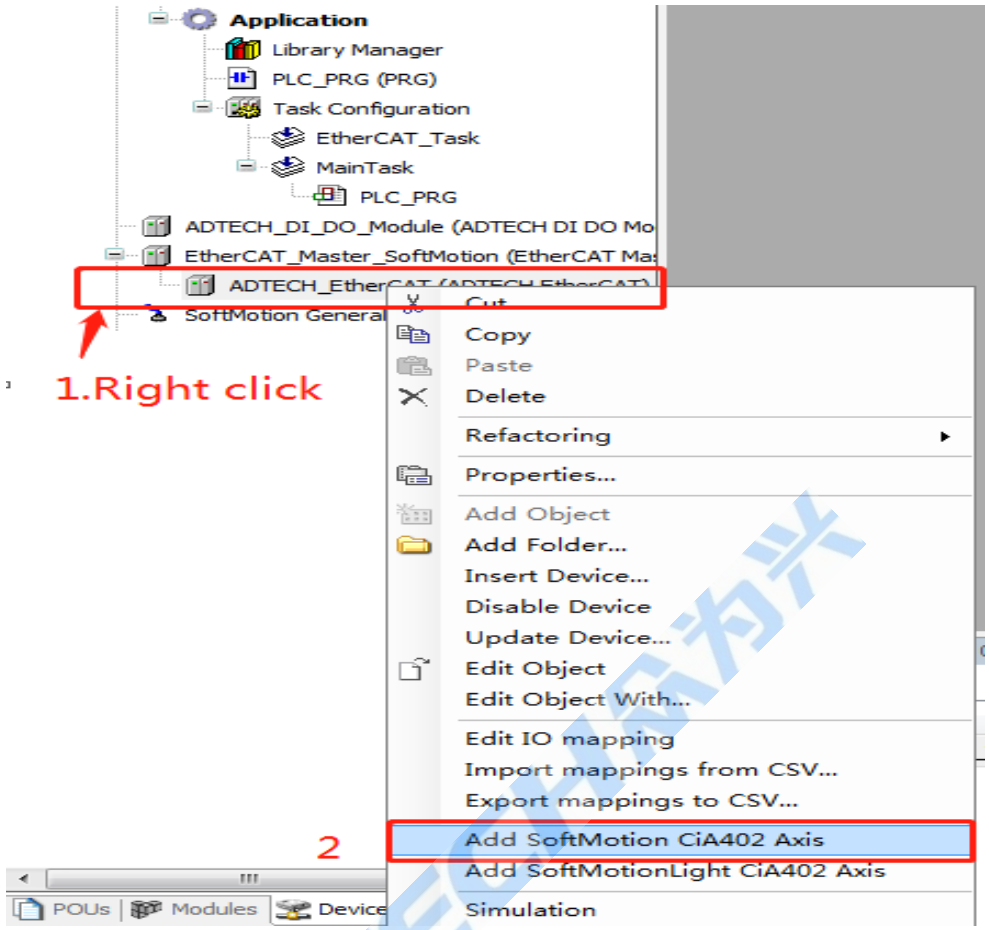


4. QXE servo slave I/O mapping settings

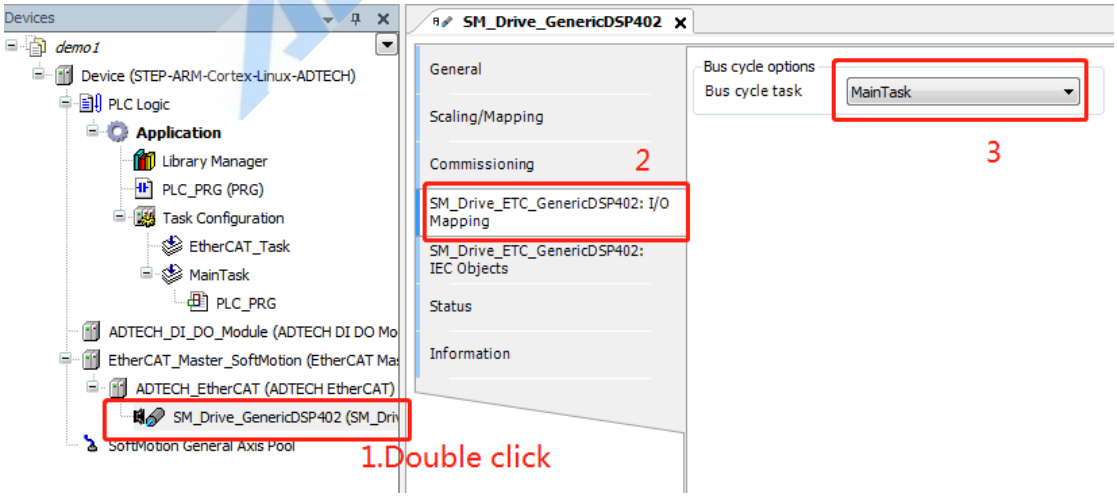


EtherCAT I/O mapping-always update variables: Enable 2 (always in bus cycle tasks);

5. Add SoftMotion CiA402 Axis to QXE servo slave



6. Related parameter settings of SoftMotion CiA402 Axis object

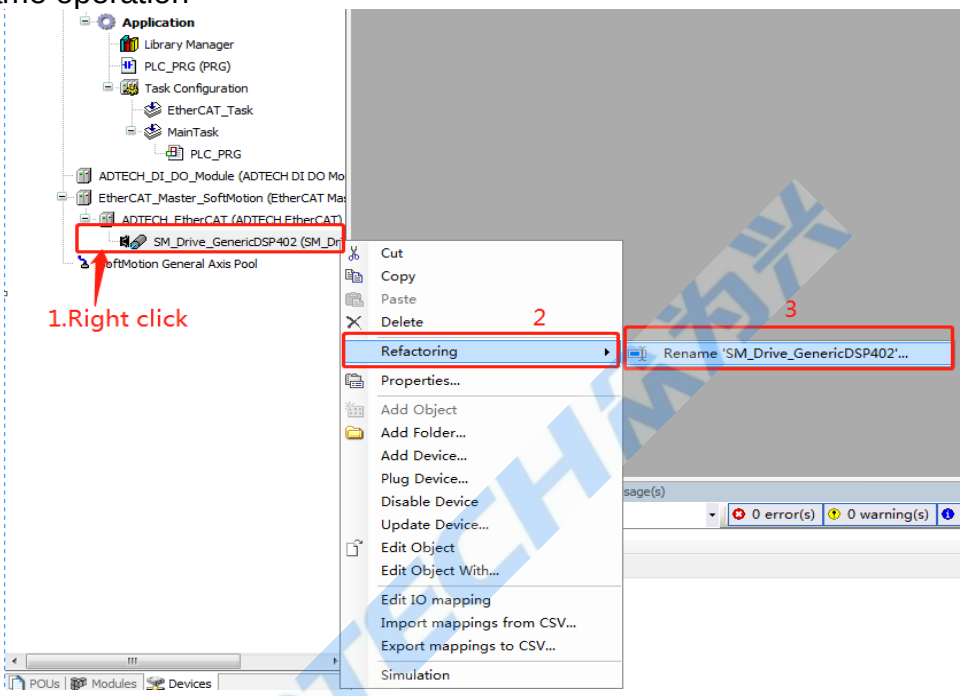


SM_Drive_ETC_GenericDSP402: I/O mapping-bus cycle task: MainTask (assign the I/O mapping of axis variables and data update to MainTask);

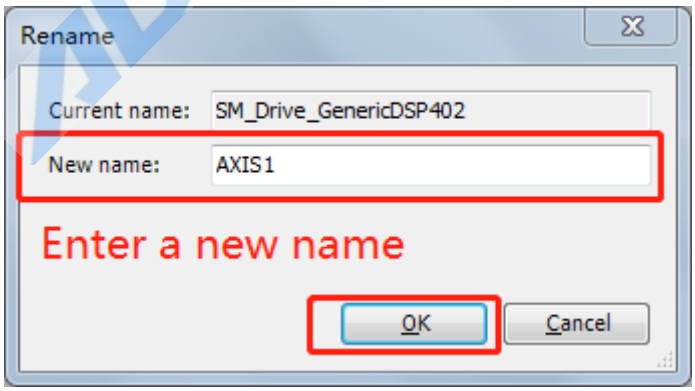
In the above engineering, the MainTask is selected for the I/O mapping-bus cycle tasks of the device relating to EtherCAT communication (here the priority of the MainTask is set to 0 as the bus cycle task of EtherCAT). Where there're other axis devices in the project (i.e. virtual axis, external encoder, etc.), bus cycle task of EtherCAT shall be selected for the I/O mapping-bus cycle task. All the motion control instructions should run in bus cycle task of EtherCATs.

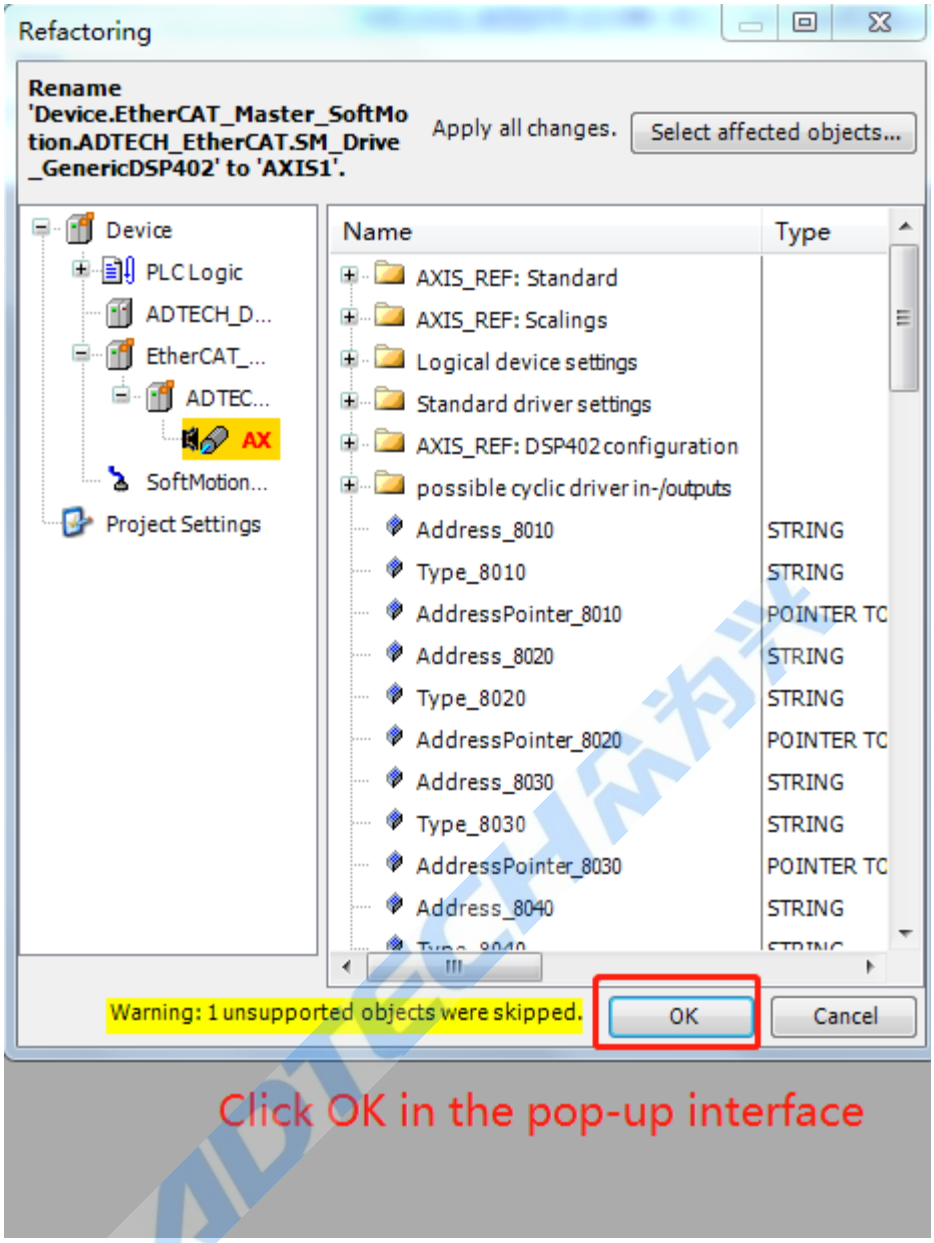
7. Change the name of the axis to a name you are familiar with

Rename operation



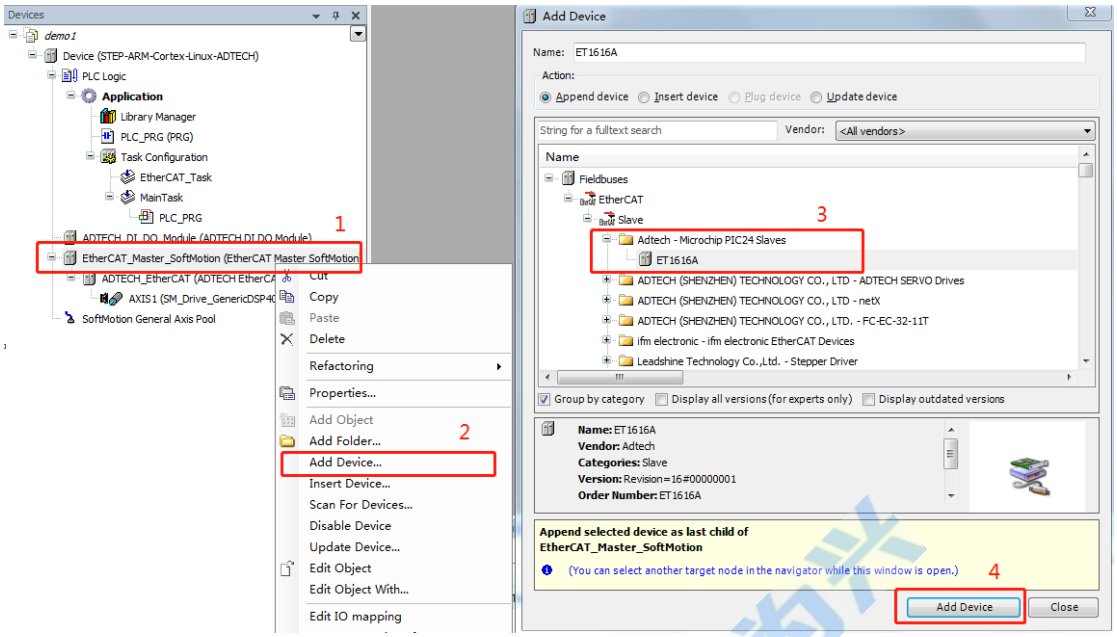
Modify axis name



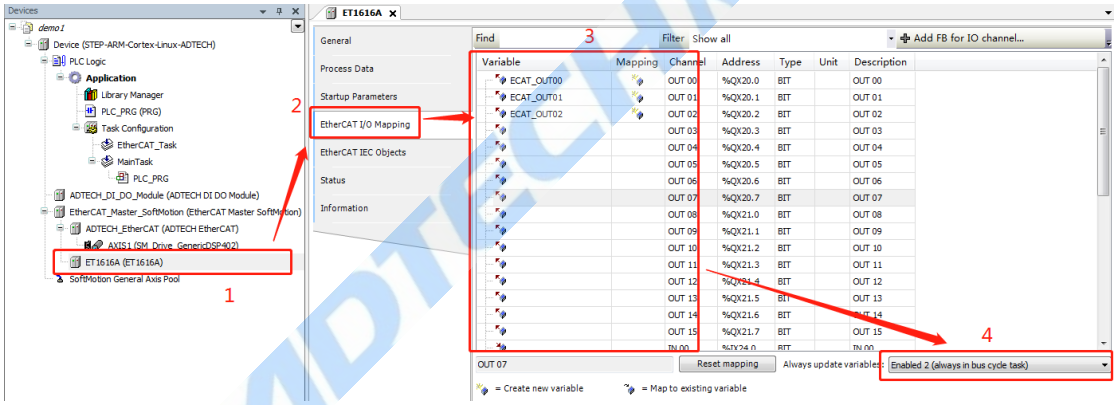


8. Repeat step 6 to add the second QXE servo and so on.

9. Add Zhongweixing ET1616A module.



10. Map variables and update settings for ET1616A module I/O



After mapping the variables to the corresponding I/O in the "EtherCAT I/O Mapping", you can obtain the input status of the ET1616A and control its output through the appropriate variables.

Always update variables: Enable 2 (always in the bus cycle task)

3.2.3 Programming POU for axis motion

1. Motion function in the project

   1). AXIS1 and AXIS2 enable.

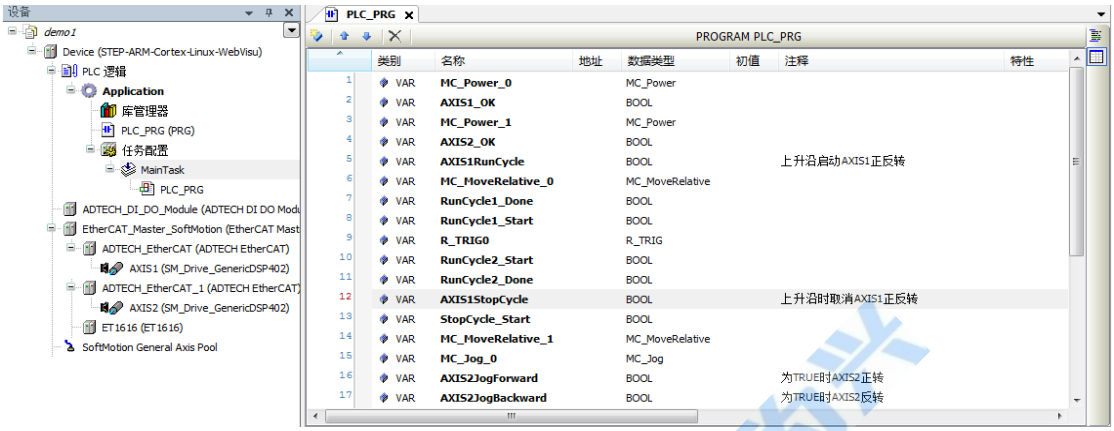
   2). AXIS1 forwards and reverses.

   3). AXIS2 forward/reverse jogs.

2. Programming motion POUs

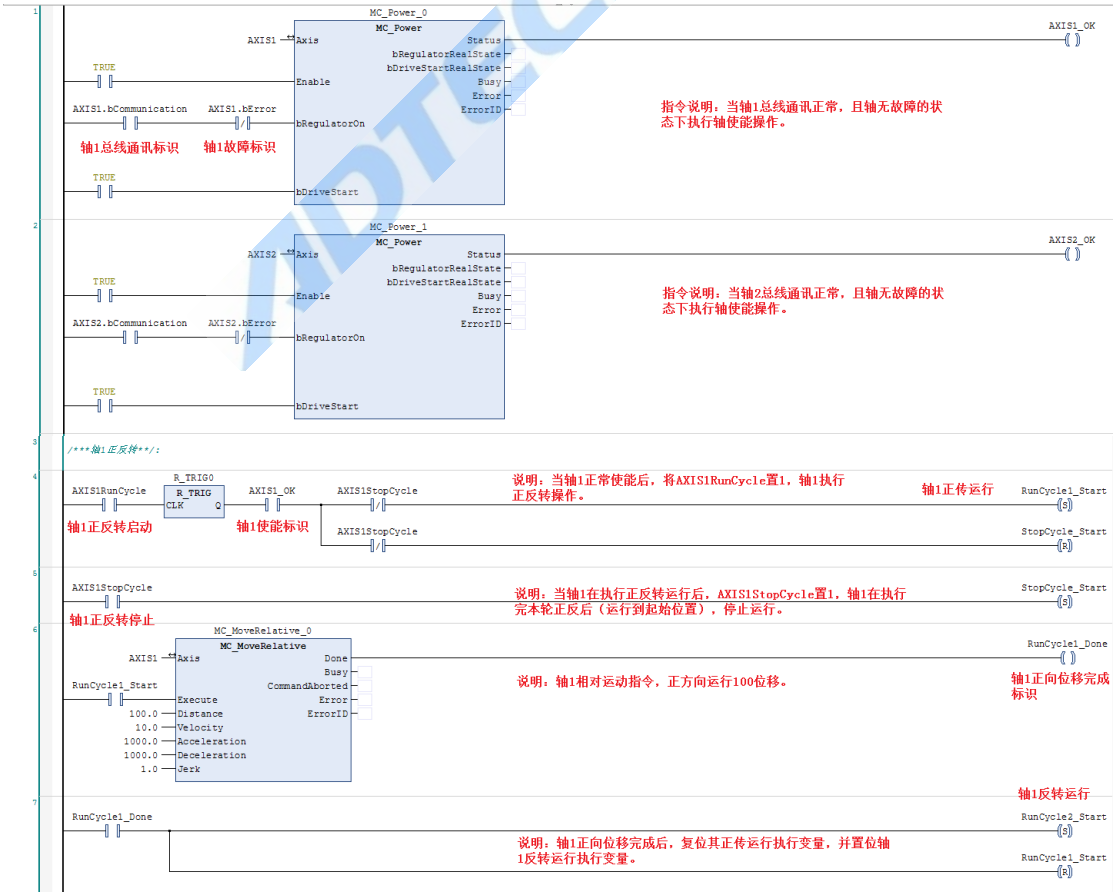
The MianTask is a bus cycle task of EtherCAT in this project. The axis motion instructions are programmed in the POU named PLC_PRG, and PLC_PRG is under the MainTask.

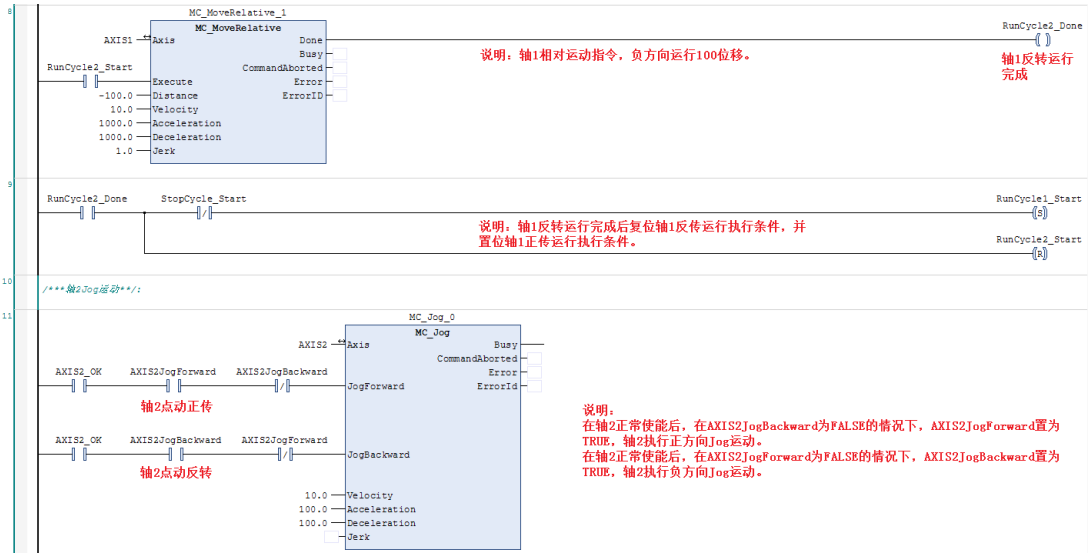
1）. The variables used in the motion function are as follows:



Note: Axis variables are not reflected in this variable list.

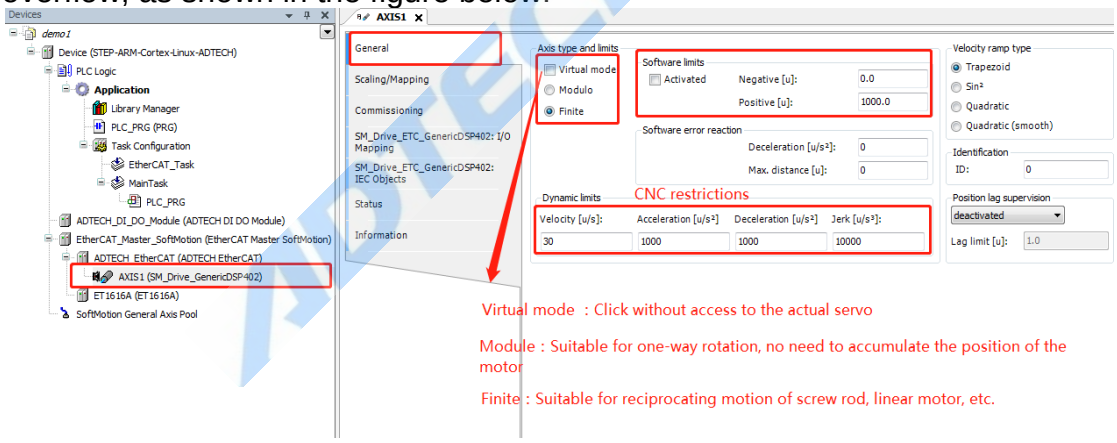
2）. The motion function program is as follows：





3.2.4 Motor parameter settings

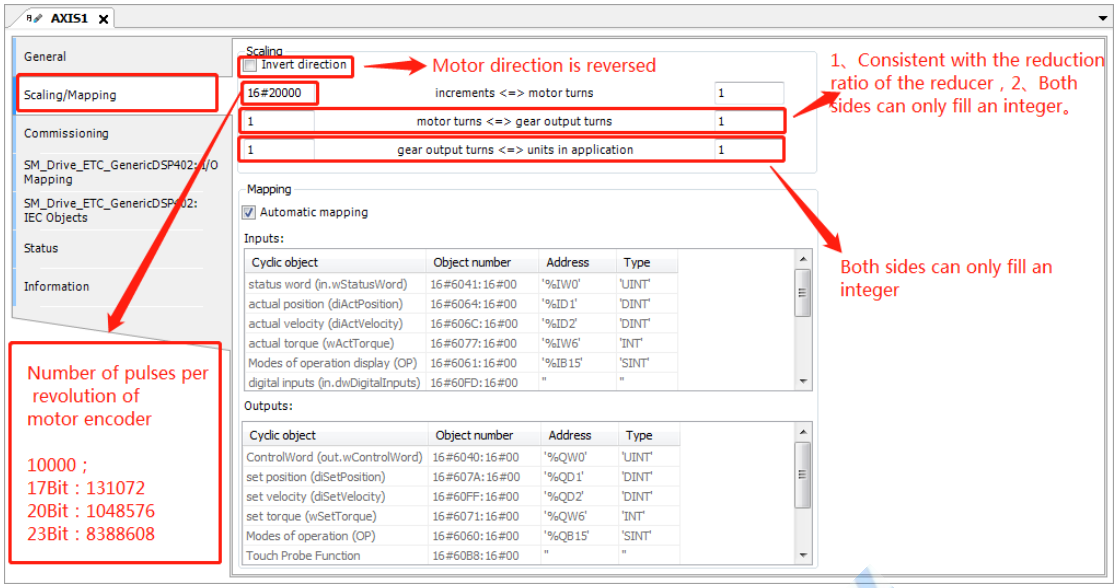
To control the motion location accurately, you need to calculate the location of the servo motor accurately. Select the "axis type and limit" according to the running and stroke characteristics of the application system, so that the controller can calculate the encoder feedback information to get the accurate location, and avoid the error caused by encoder pulse accumulation overflow, as shown in the figure below:



Since the stroke is limited for the reciprocating motion mechanism of the screw type, we need to know its absolute position within the range of the screw. The "Limit" mode is recommended at this moment.

If it is an axis of unilateral operation type, when running in the "Limit" mode, the location counting overflow is likely to occur, resulting in a location calculation error. The "Modulus" mode is recommended at this moment.

The resolution of the motor encoder and the mechanical reduction ratio of the application system are varied, which should be set during programming according to the actual situation. The setting interface is as follows:



In this example, the motor encoder is 17bit, the motor and the screw are directly connected (no reducer is connected), and the screw lead is 10mm.

3.2.5 Programming POUs for local and remote I/O controls

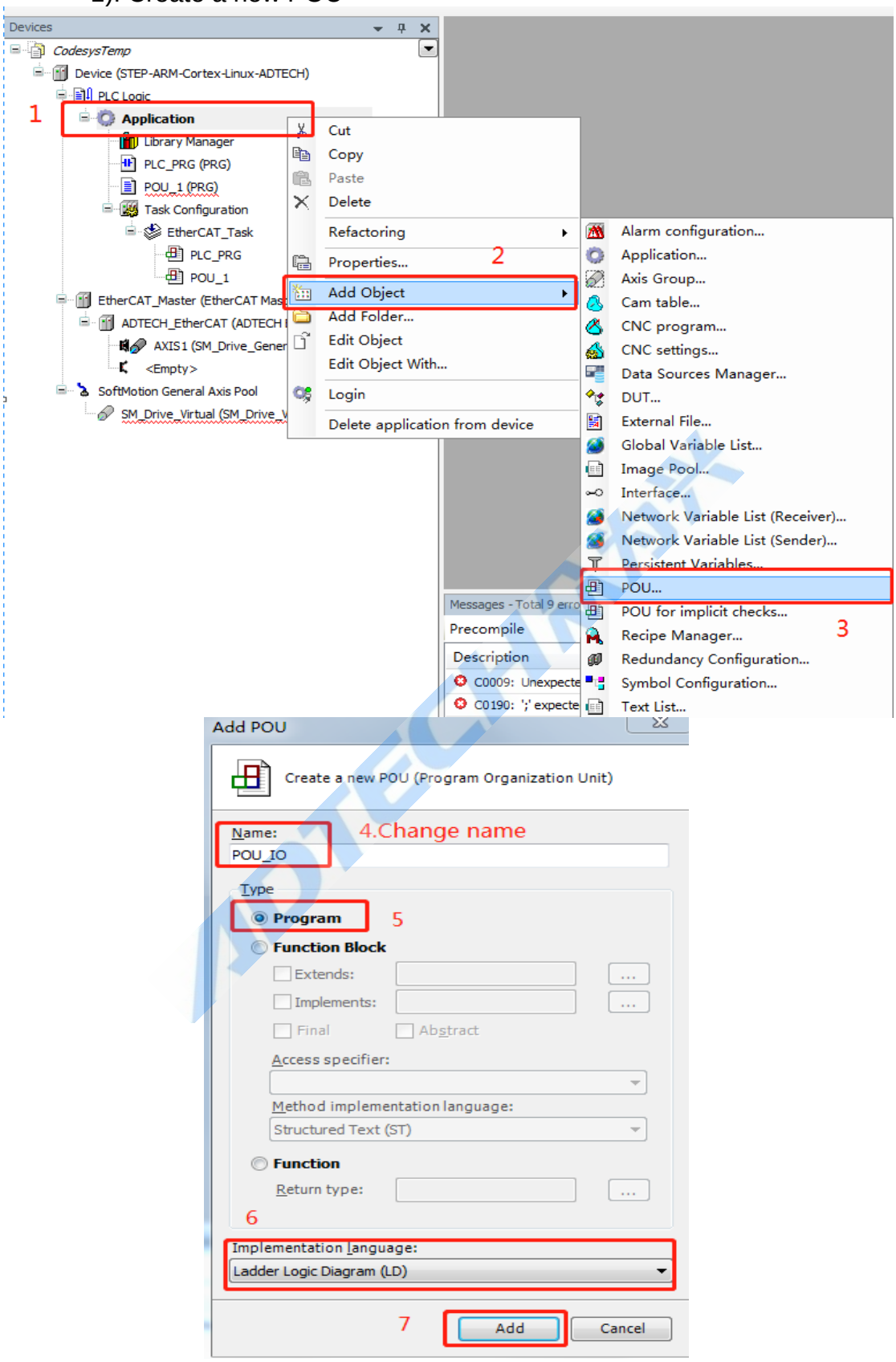
1. I/O control technology in the project

1). The local 24-ch outputs will light up the LED in sequence at a fixed interval of 500ms, and output them cyclically.

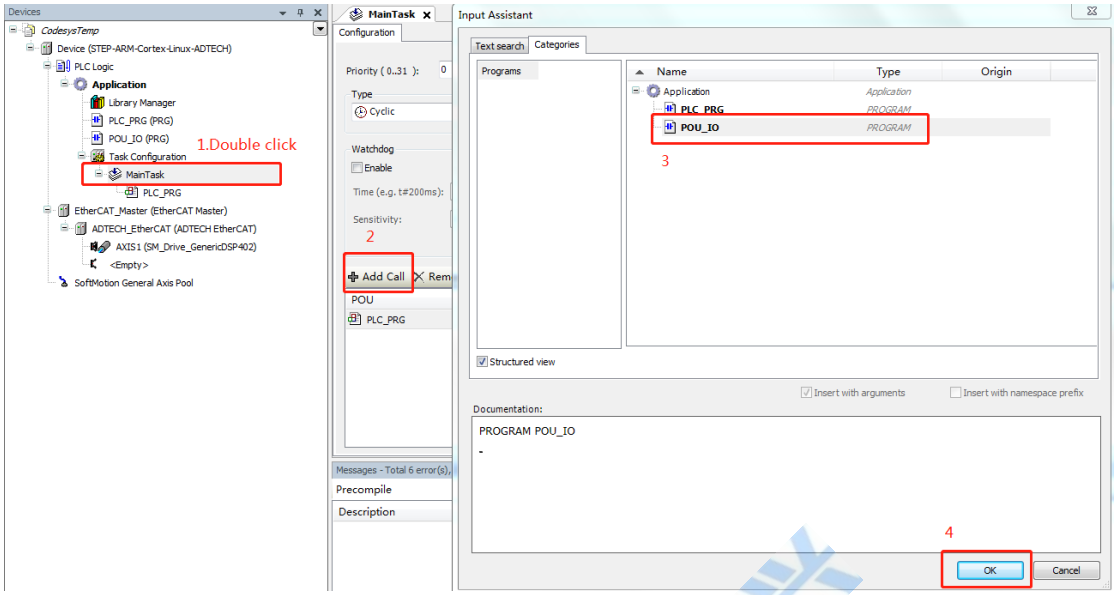
2). 16-ch output ports (0, 2, 4, 6, 8, 10, 12, 14) of EtherCAT bus I/O ET1616A and 1, 3, 5, 7, 9, 11, 13, 15 alternatively light up in a cycle of 500ms.

2. Programming and implementing POUs of the I/O process in the project

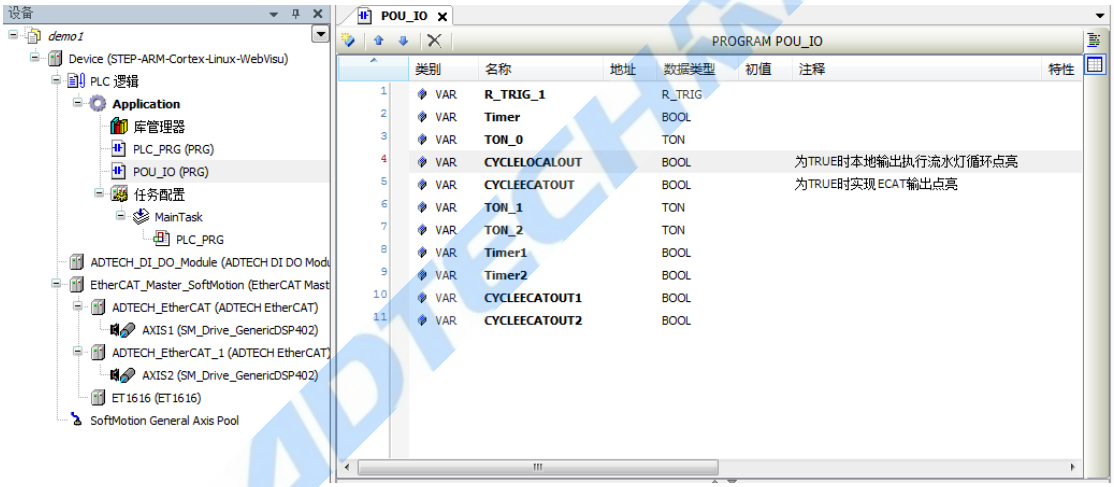
1). Create a new POU



2）Assign POU_IO to the MainTask task

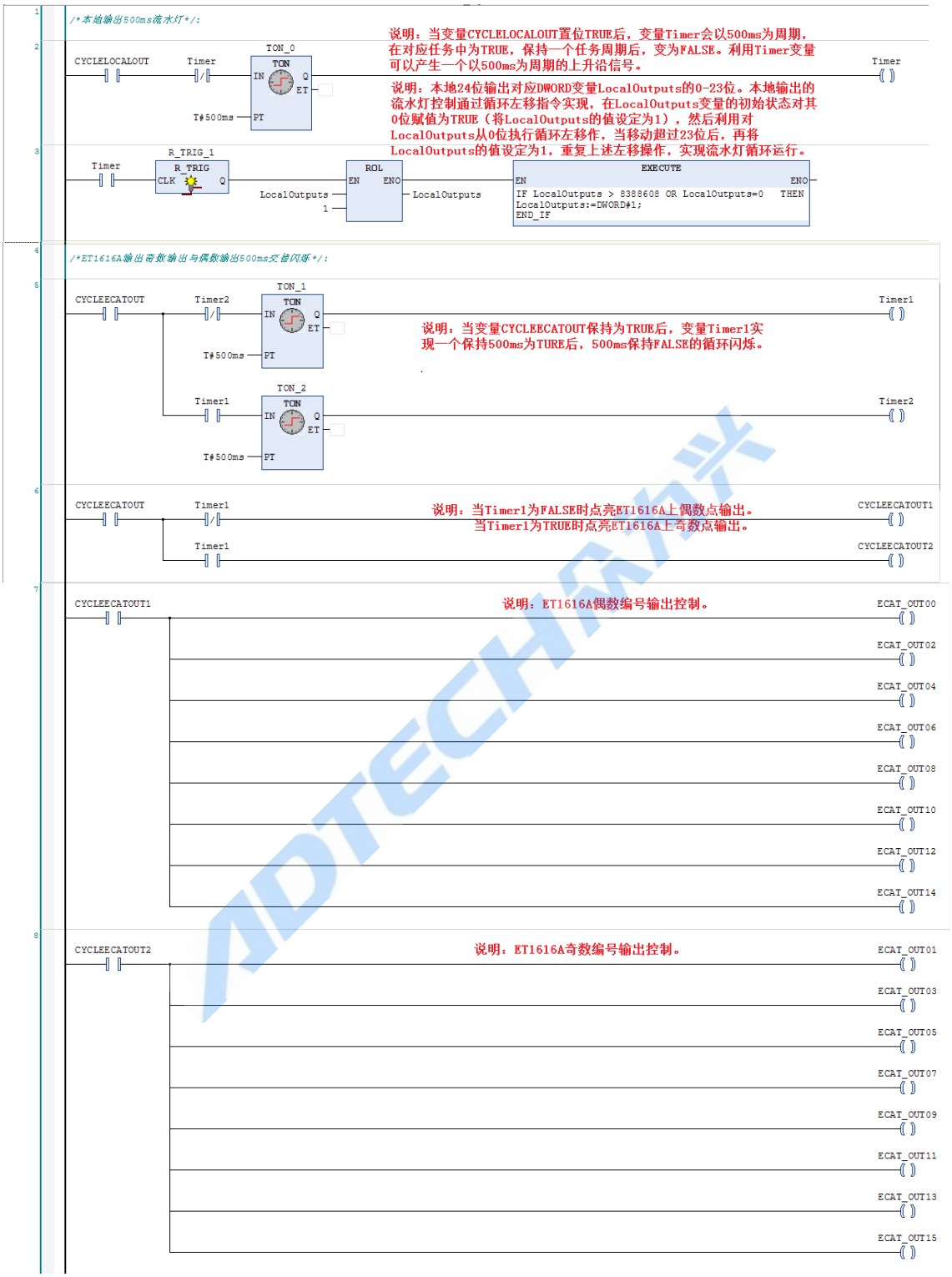


3）. The variables used in POU are as follows

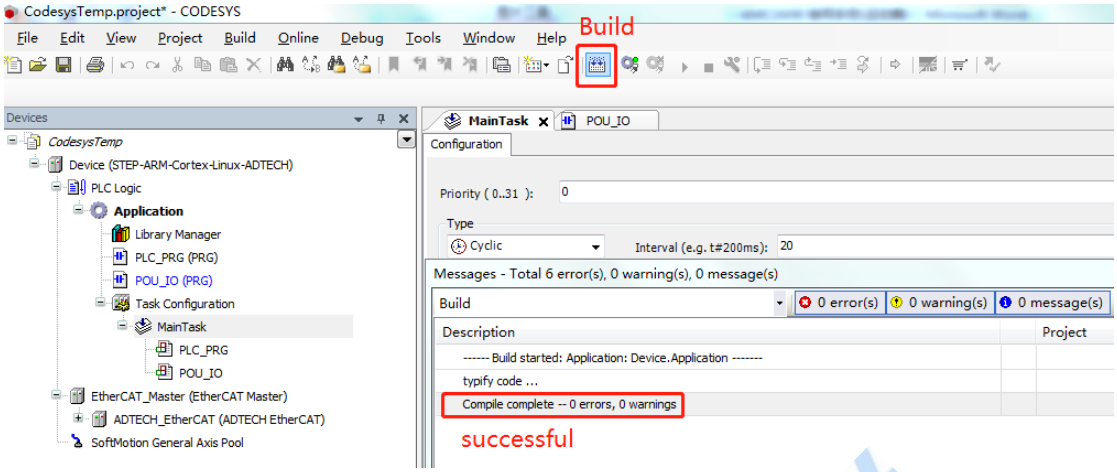


Note: I/O mapping variables are not reflected in this variable list.

4）. The program to realize I/O function is as follows：



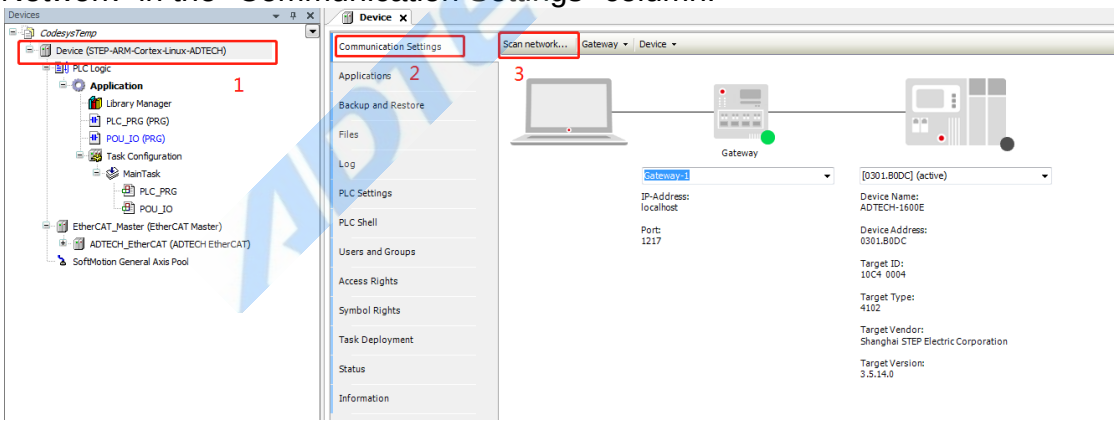
3.2.6 User program compilation



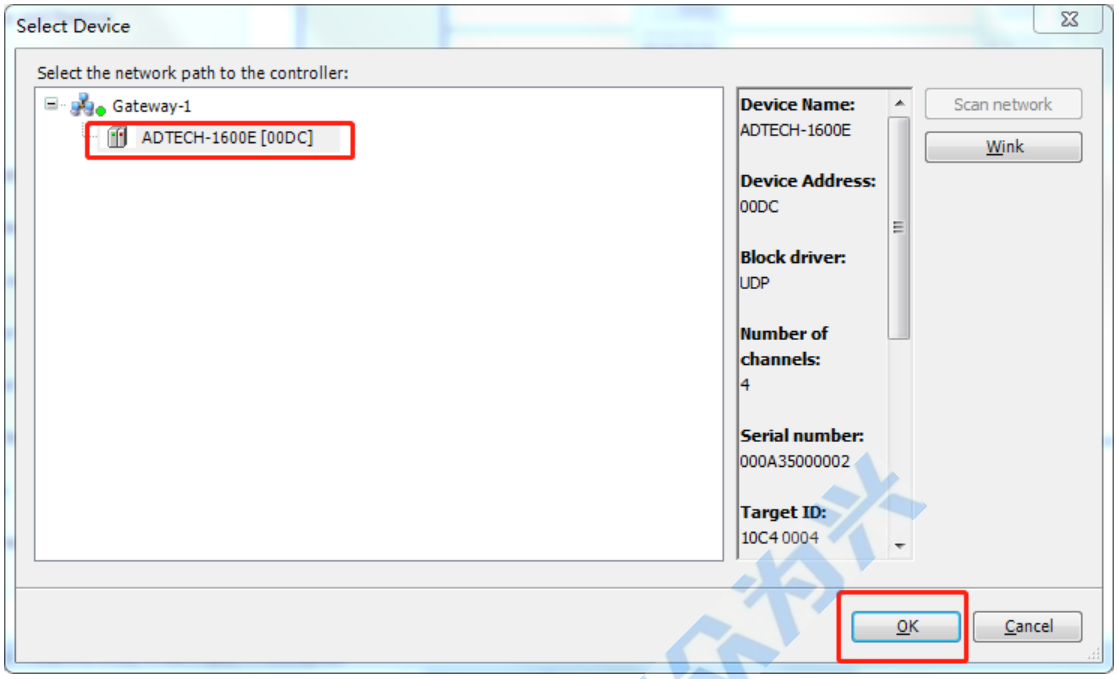
As shown in the figure above: Compile the program by pressing the compile icon or the shortcut key F11. The compilation result will be displayed in the datagram window. In case of any error, there will be a prompt in the datagram bar. Double click the error description, and the cursor will jump to the corresponding program editing window to facilitate revision. Compile upon processing, until all compilation problems are eliminated.

Download the user program to the AMC1600 controller.

1. Find "Device" in the device column, double-click it and enter, and click "Scan Network" in the "Communication Settings" column.

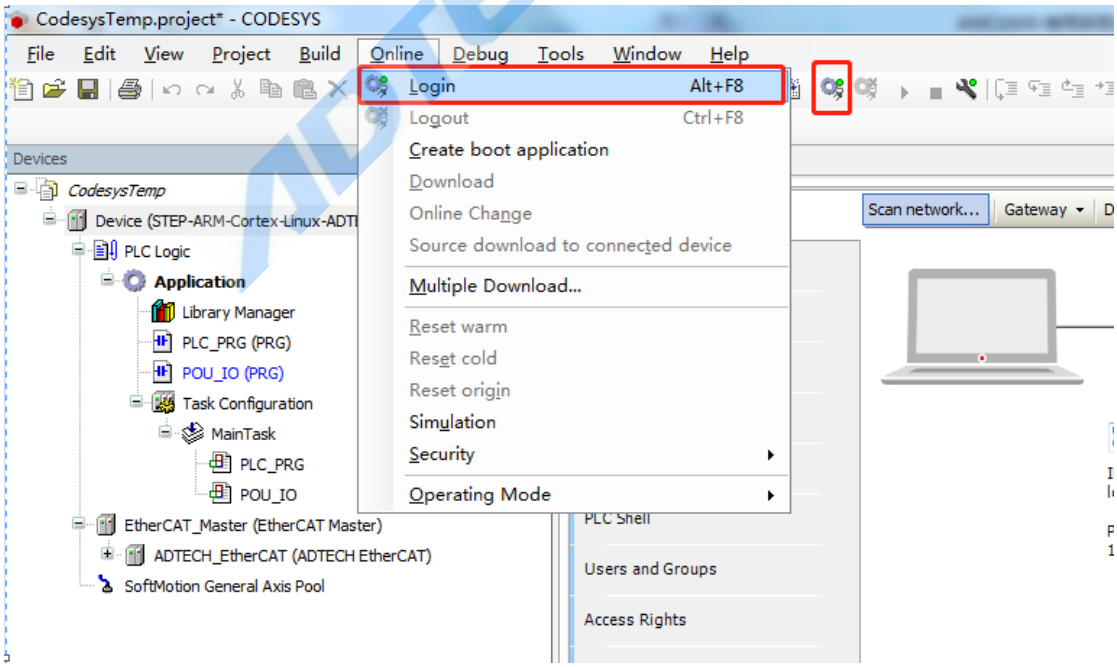


2. The following device window will appear after the line is properly connected and the computer network segment is correctly set.



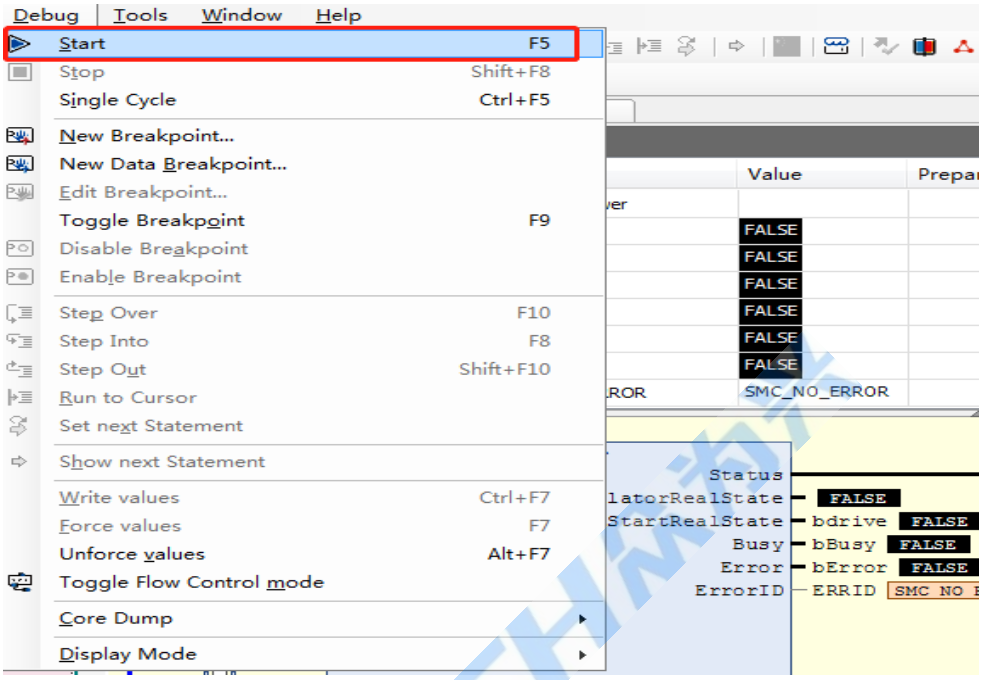
Select the "1600C" device and click "OK", to allow the normal connection between the computer and the controller.

3. Select "Login to" in the drop-down list of the "Online" column, to download the chart completion program to the controller.

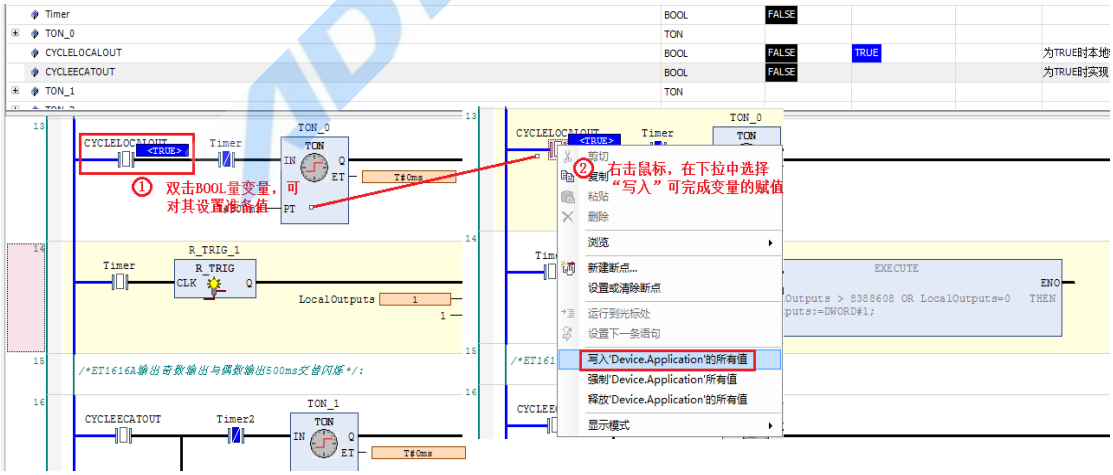


3.2.7 Online debugging and running of a user program

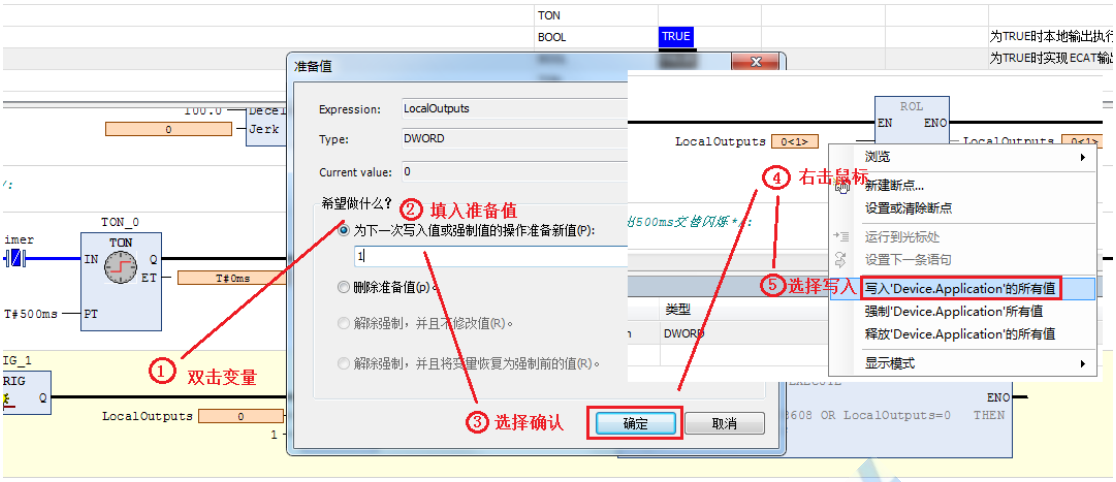
After downloading the program to the AMC1600 controller, click "Start" in the drop-down menu of the "Debug" to run the program.



Double-click the BOOL variable in the program interface to set the preparation value, then right-click and select "Write" to write the preparation values.



A setting box for preparation values will pop up when double clicking the other variables. For example, double-clicking a DWORD variable, the preparation value will be wrote as follows.



3.2.8 Typical steps for programming a user engineering

According to the above example, to write a user program with MC motion control, you need to take the following steps:

Configuration of application system: Configure the system hardware of the project with reference to the used and so on.

Programming of a user program: According to the functions to be implemented, write the motion control using a POU, and write the common logic control also using a POU.

Configuration of servo drive parameters: Configure SDO and PDO objects based on the running mode of the servo to ensure that the communication objects required between the MC function block and the servo of the user program are in the same configuration table.

Configuration of servo motor parameters: complete the encoder resolution of the servo motor, the transmission ratio of the mechanical structure, the characteristics of the axis range of motion correctly, to make the command displacement of the control object correspond to the actual displacement correctly.

Task arrangement: According to the real-time requirements of control, execute the motion control PLC_PRG under the bus cycle task of EtherCAT, set the period to 4ms, and the priority to 0; execute the common logic control POU2 in common tasks and set the cycle to 20ms, and the priority to 16.

Online debugging: Connect the AMC1600 controller to the PC via the LAN

network, power on upon correct wiring, download and debug the user program, and eliminate bugs of the user program; if an external hardware provided to connect the external bus device to the controller, make direct debugging; if there is no servo system, the servo can be set to virtual axis debugging; if there is no controller, the debugging program can be simulated on PC, to eliminate the possible errors in the program.

Chapter IV AMC1600 Execution Mechanism

4.1 Tasks and configuration in user engineering

4.1.1 Types of tasks in the user program

Task type	Characteristic	Task example
Cycle	Periodic task, every set time, the task is run once.	EtherCAT Bus cycle task Normal task cycle
Event	When the bool amount specified by the trigger event corresponding to the normal task cycle is set to true from false, it is executed once (triggered by rising edge)	Soft interrupt handling pou
External events	Triggered by external interrupt.	
Inertial sliding	After the task is started, it will be executed in a circle without a fixed cycle. As long as other tasks are not executed, it will be executed	Normal task cycle
State	When the bool value is set to true, the program is executed repeatedly	Conditional execution pou

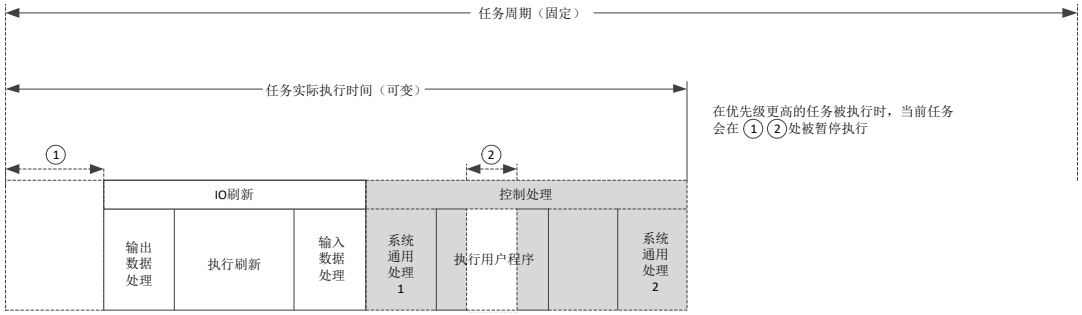
4.1.1.1 Cyclic tasks

1. When I/O refresh and motion control commands appear in cyclic tasks, such as bus cycle task of EtherCATs. The task execution flow is as follows (the figure shows the execution flow of bus cycle task of EtherCAT; set the task priority to the highest priority as 0).

任务周期　　：循环任务中设定的任务执行的间隔时间
任务实际执行时间：任务的执行条件成立，到实际执行完成的时间

Process		Processing content
I/O Refresh	Output data processing	1.Generate out refresh data for I / O refresh. 2.When forced refresh is set, the forced output is reflected in the out refresh data.
	Perform refresh	1.Perform data exchange between I / O.
	Input data processing	1.Import in refresh data of I / O refresh. 2.When forced refresh is set, the forced input is reflected in the in refresh data.
Control processing	General processing 1 of system	1. Mutual exclusion between tasks of variables. 2.Motion input processing (if the bus task cycle specified by I / O mapping item of motion axis, encoder and other devices is this cycle task, the feedback value or status of corresponding axis variable will be updated). 3.Data tracking processing (data sampling, trigger decision).
	Execute user program	1.Execute the program in sequence according to the order of POU allocation in the task.
	Motion control	1.Execute the user's motion control instruction and issue the motion control instruction.
	General processing 2of system	1.Mutual exclusion between tasks of variables.

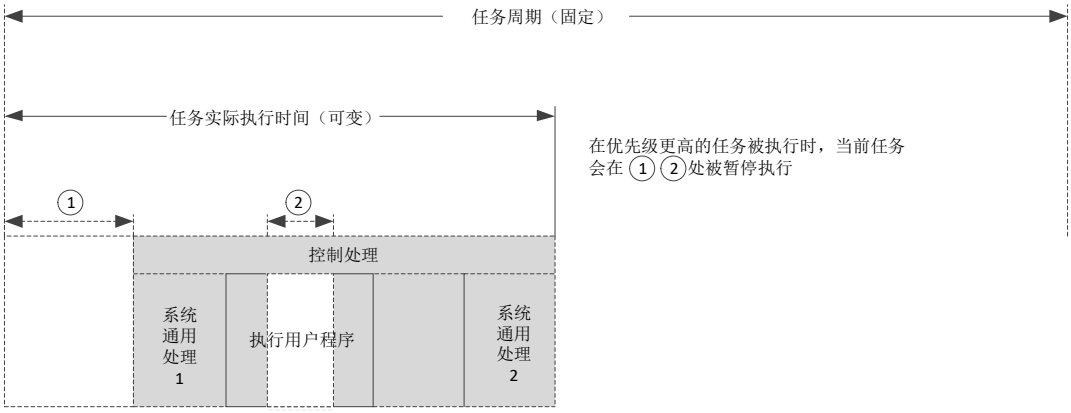
2. When there's I/O refresh but no motion control instruction in the cyclic task.



任务周期　　：循环任务中设定的任务执行的间隔时间
任务实际执行时间：任务的执行条件成立，到实际执行完成的时间

Process		Processing content
I/O Refresh	Output data processing	1. Generate the out refresh data of I / O refresh. 2. When the forced refresh is set, the forced output will be reflected in the out refresh data.
	Perform refresh	1. Perform data exchange between I / O.
	Input data processing	1. Import in refresh data of I / O refresh. 2. When the forced refresh is set, the forced input is reflected in the in refresh data.
Control processing	General processing 1 of system	1. Mutual exclusion of variables between tasks. 2. Data tracking processing (data sampling, trigger decision).
	Execute user program	1. Execute the program in sequence according to the order of POU allocation in the task.
	General processing 2 of system	1. Mutual exclusion of variables between tasks.

3. When there're no I/O refresh and motion control instructions in the cyclic task, for example, a common logical cyclic task. The task execution flow is as follows.

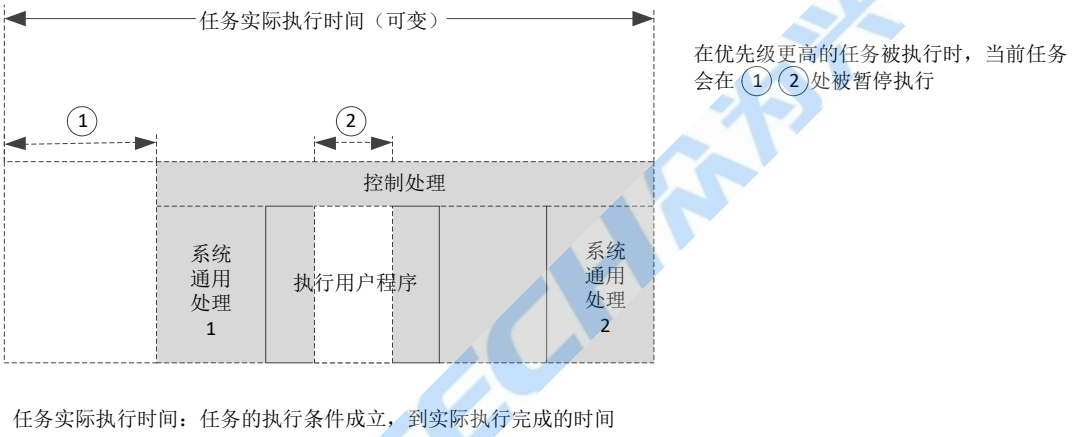


任务周期　　：循环任务中设定的任务执行的间隔时间
任务实际执行时间：任务的执行条件成立，到实际执行完成的时间

Handling		Processing content
Control processing	General processing 1 of system	1. Mutually exclusive processing of variables 2. Data tracking processing (data sampling, trigger judgment)
	Executing a user program	1. According to the assignment order of POU in the task, execute the program in order.
	General processing 2 of system	1. Mutually exclusive processing of variables between tasks

4.1.1.2 Event tasks

The BOOL value specified by a trigger event is executed once when changing FALSE to TRUE (rising edge is triggered). The task execution flow is as follows.



Process		Processing content
Control processing	General processing 1 of system	1. Mutual exclusion of variables between tasks. 2. Data tracking processing (data sampling, trigger decision)
	Executing a user program	1. Execute the program in sequence according to the order of POU allocation in the task.
	General processing 2 of system	1. Mutual exclusion of variables between tasks.

4.1.2 Description of task configuration

Common tasks in user programs may include bus cycle task of EtherCATs, ordinary cyclic tasks, and event tasks.

In the "cyclic" task, the "interval" shall mean the space of time during which the task is executed. For general logic in the program, such as common IO status and fault alarm display, you can set a larger task "interval", for example,

10-20ms. For logic tasks that need to be processed promptly, you may set the "interval" smaller.

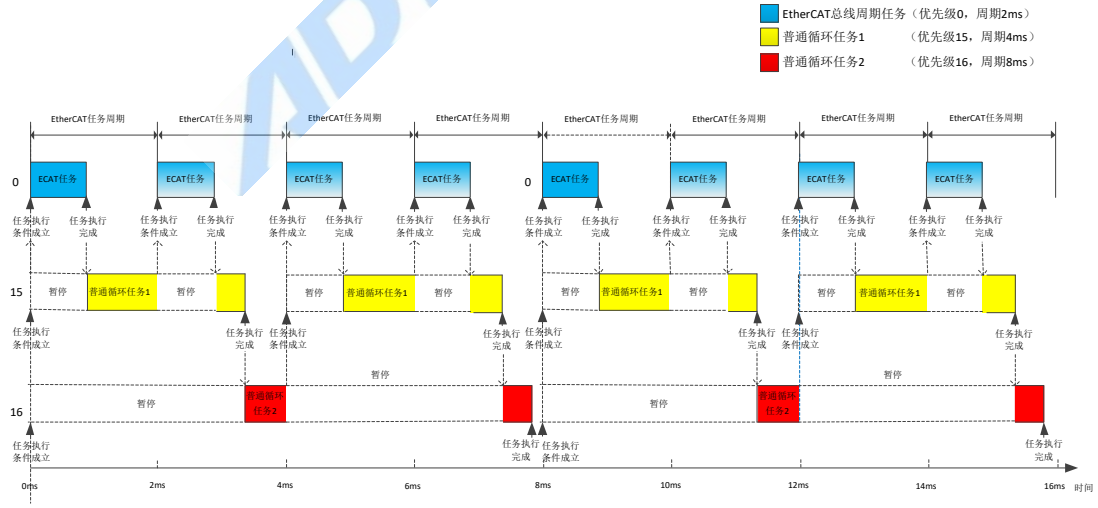
Since the bus cycle task of EtherCAT is special, set its priority set to the highest among all tasks, so as to ensure the stability of EtherCAT communication. In general, the priority of the EtherCAT bus task is set to 0 (the priority can be 0…31, the smaller the value, the higher the priority). Setting of the "interval" of the bus cycle task of EtherCAT to 1-4ms is recommended. The smaller the "interval" time, the faster the servo response and the higher the control precision. Once the number of axes controlled in the program increases, the "interval" time needs to be increased accordingly, otherwise the CPU computation may be overloaded.

Precautions for CODESYS platform in actual programming: all instances of motion function blocks in the program must be specified to a same task. To ensure real-time capability of motion control and not be interrupted by other tasks, the task needs to have the highest priority, that is the bus cycle task of EtherCAT. If the EtherCAT bus devices (bus axis, and bus I/O module) is provided in the system, the "I/O mapping" parameter of the EtherCAT bus device-"bus cycle task" must also be assigned to "bus cycle task of EtherCAT". The POU corresponding to the motion instruction involving the bus axis in the program must also be assigned to the bus cycle task of EtherCAT, otherwise the motion function block involving the bus axis will report an error during execution.

Only one type is available to a task configuration. You need to specify a priority and set a time interval. To get multiple different execution characteristics, you can make multiple task configurations by adding tasks.

Multiple POUs can be added to a task, and the POUs are executed from the top to the bottom based on the order under the task.

The smaller the value of the priority of the task, the higher the priority of the task. The execution of low-priority tasks will be interrupted by the execution of high-priority tasks. As shown in the following picture:



According to the figure above:

When the task is executed in the controller, there is a time alignment point, as shown on the leftmost side in the figure. At this time point, the task begins with the highest-priority to the low one.

When being executed, the low-priority tasks may be interrupted by high-priority tasks. After the high-priority tasks are executed, it will return to the interrupted task to continue to execute such task.

The bus cycle task of EtherCAT is endowed with the highest priority. It starts the task according to the set cycle, and completes all the POUs of the task before returning to the lower priority task.

AMC1600 uses a multi-task execution plan when running. Different execution cycles are used for different tasks. Some global variables need to be accessed and modified between different POUs. Therefore, the global variables need to be interactively synchronized, which is also performed at the time alignment point of the task. When setting the cycle of a cyclic task, the relationship as an integer multiple is required. For example, if the bus cycle task of EtherCAT is set to 1ms, the ordinary task needs to be an integer multiple of the EtherCAT as 1ms, 2ms, 3ms, 4ms, 5ms, 8ms, 10ms,...; set the bus cycle task of EtherCAT to 2ms, then the ordinary task to 2ms, 4ms, 8ms, 10ms, 20ms...; set the bus cycle task of EtherCAT to 4ms, and the ordinary task to 4ms, 8ms, 16ms, 32ms... Setting the bus cycle tasks of EtherCAT to 3ms, 6ms, 7ms, and 9ms will lead to the non-integer multiple relationships between task cycles.

4.1.3 Prioritization of tasks

For the CODESYS platform, the priority and type of different tasks are to be defined by the user. In user engineering, a reasonable prioritization of the task would maximize the performance of the controller. Task priority will be arranged roughly according to the following rules in actual application:

1. Bus cycle task of EtherCAT

It is a cyclic task, which has the highest priority in the task (set its priority to 0). Assign the "Bus Task Cycle" of the "EtherCAT I/O Mapping" in the EtherCAT master device to this task; assign the "Always Variable Update" of the "EtherCAT I/O Mapping" in the EtherCAT slave to this task; assign the "I/O mapping" item of the axis parameter to this task (including bus axis, virtual axis, and encoder); while assign the POU corresponded to the MC motion instructions to this task. Set the priority to 0, and the interval to 2ms, 4ms, and 8ms.

2. Interrupt processing task

This includes the events and external events. At present, AMC1600 supports only event interrupt processing, and event interrupts are of soft interrupt processing. Considering the priority as well as the processing mechanism of variables in tasks, the soft interrupts features low real-time performance. To achieve higher real-time performance, try to set the priority of the soft interrupt task as high as possible, but lower than the bus cycle task of EtherCAT. For example, set the priority to 3. If lower real-time requirement of the interrupt is required, the task priority nay also be set lower, for example, set it to 30.

3. TCP and UDP communication tasks

It is a cyclic task. It is recommended to set the priority higher than the ordinary cyclic task but lower than the bus cycle task of EtherCAT. For example: Set the priority to 4, the cycle interval to 8ms (the smaller the cycle interval is, the faster the communication rate; however, the other low-priority tasks may accordingly be interrupted multiple times during execution, leading the delay in the execution duration of low-priority tasks, which affects the execution efficiency of low-priority tasks. Therefore, it is not recommended to set the cycle interval of TCP and UDP communication tasks too small).

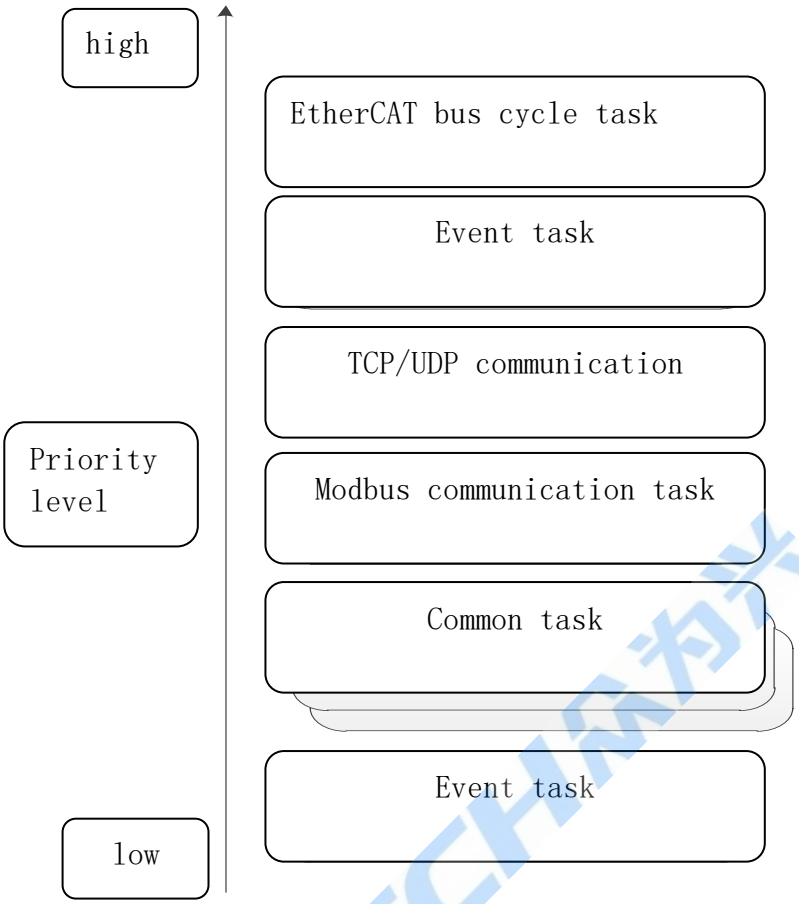
4. Modbus communication tasks (ModbusTCP and serial communication)

It is a cyclic task type. It is recommended to set the priority higher than ordinary cyclic task and lower than TCP and UDP communication tasks. For example, set the priority to 4, and the cycle interval to 8ms. It is the same as the interval setting instructions of TCP and UDP communication task cycles. It is not recommended to set the interval of such task cycle too small.

5. Ordinary cyclic task

It is a cyclic task type. Its priority is recommended to be lower than Modbus communication tasks. The content of the task is ordinary logic control with low real-time requirements. It interacts with the I/O and MC motion control in the bus cycle task of EtherCAT through global variables. For example: set the priority to 16, and the cycle interval to 10ms.

The priority of the above tasks in the user engineering is as follows:



4.2 Principle of EtherCAT bus communication

EtherCAT is a real-time industrial Ethernet technology proposed by BECKHOFF Automation GmbH in 2003. It is characterized by high speed and high data efficiency. It supports multiple device connection topologies. Its slave node uses a professional control chip, while the master station uses a standard Ethernet controller.

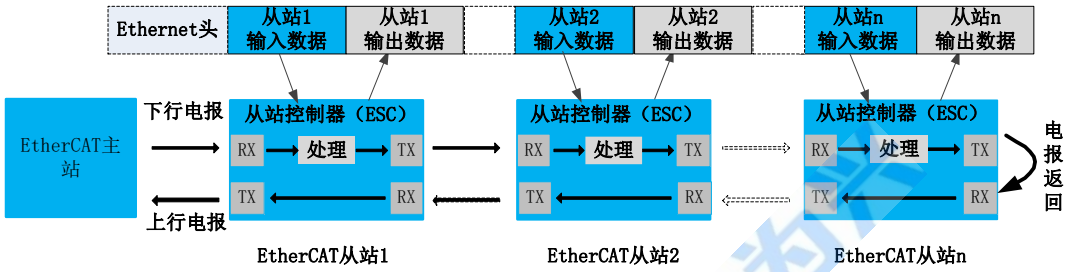
The transmission medium of EtherCAT uses a Cat 5 UTP cable of 100BASE-TX standard Ethernet, which supports a maximum transmission rate of 100Mpbs, with the transmission distance between two nodes of up to 100m.

4.2.1 System components of EtherCAT

EtherCAT is a real-time industrial Ethernet technology, which makes full use of the full duplex of Ethernet. It accesses control (MAC) using the master-slave mode media, of which the master station uses a standard Ethernet interface card, while the slave station uses a special ESC (EtherCAT

Slave Controller). In the physical layer of EtherCAT, the standard Ethernet physical layer devices are used.

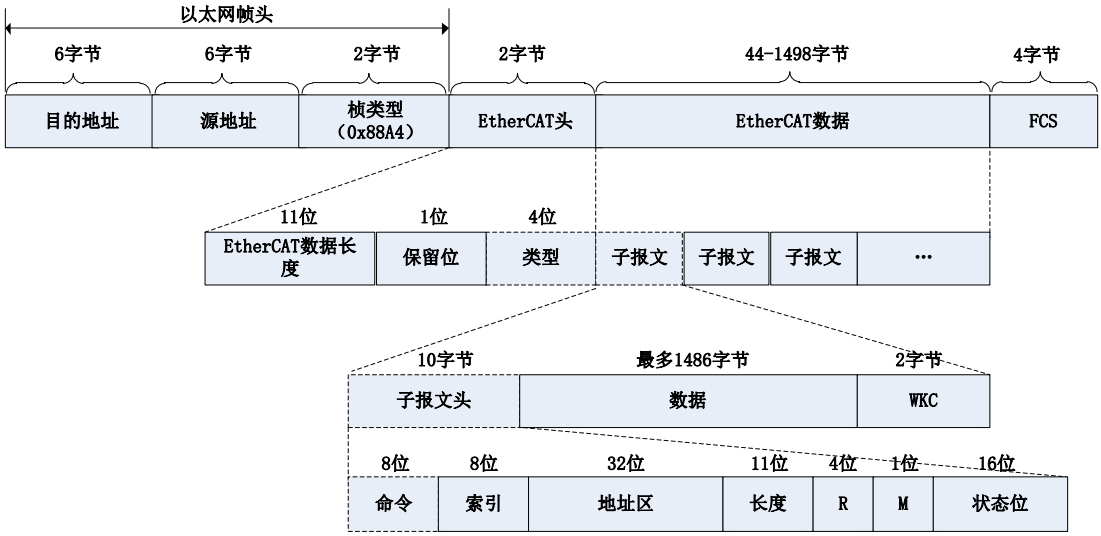
The master station sends an Ethernet frame to each slave station, and the slave station extracts data from or inserts data into the data frame. The master station connects multiple slave stations, as shown in the figure below. The slave station processes directly the received datagram, extracts from and inserts relevant user data into the datagram, and then transmits the datagram to the next EtherCAT slave station. When being transmitted to the last EtherCAT slave station, the datagram will pass through all slaves and finally be transmitted to the EtherCAT master. The EtherCAT network communication model is detailed as follows:



EtherCAT Communication principle

4.2.2 Data frame structure of EtherCAT

EtherCAT data is directly transmitted using Ethernet data frame, which uses frame type 0x88A4. EtherCAT data consists of a 2-byte header and 44-1498 bytes of data. The data field is composed of one or more EtherCAT sub-datagrams, and each sub-datagram corresponds to an independent device or slave storage area. The data structure frame of EtherCAT is defined in figure below.



4.2.3 Clock synchronization concept of EtherCAT communication

To ensure the control accuracy and stability of the master on some devices, it is necessary to provide a synchronized clock to each slave device and the master device in the EtherCAT communication network.

4.2.3.1 Distributed clocks

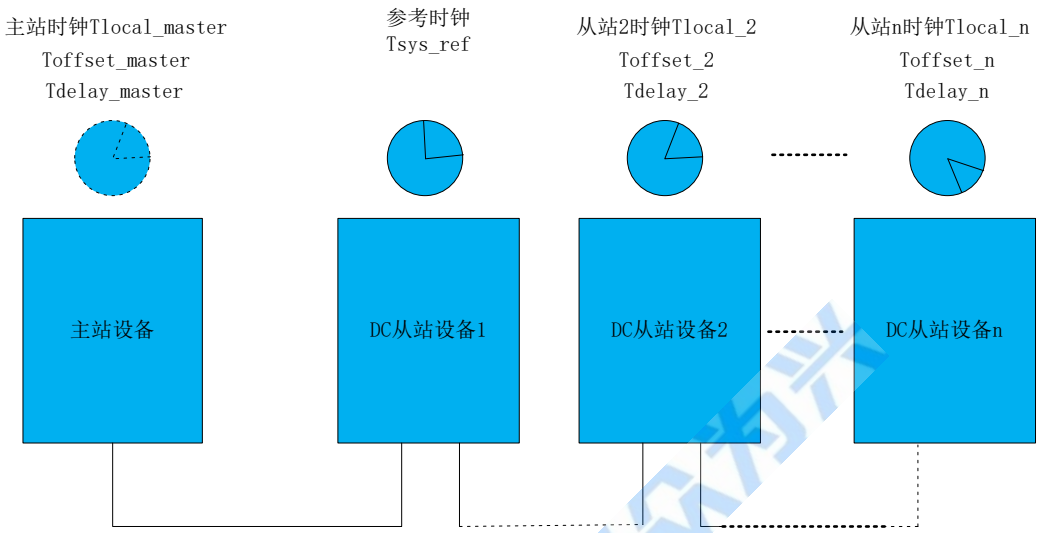
By using the distributed clock (DC), all the EtherCAT devices can use the same system time, so that the tasks of each device can be executed synchronously. The slave devices can produce a synchronization signal according to the synchronized system time, which is used for interrupting control or triggering digital input and output. The slave station that supports the distributed clock is called the DC slave. The distributed clock mechanism functions to enable all the slave stations to synchronize to a reference clock.

4.2.3.2 Clock synchronization of EtherCAT communication

According to the EtherCAT protocol, the clock of the first slave station connected to the master station with the distributed clock function is used as the reference clock. The reference clock provides the EtherCAT system time. It is used to synchronize the slave clocks of other devices and the master station. Each DC slave has a local clock, which runs independently and is timed using the local clock signal. When the system starts, there is a certain difference between the local clock and the reference clock which is called the initial clock offset. During operation, due to the fact that the reference clock and the DC

slave clock use their own clock sources, their timing cycles drift to a certain extent, which will lead to the clock running out of sync. It is called the lock clock drift. To give precise clock synchronization control, the system must calibrate the initial clock offset, and make compensation for the local clock drift.

The clock synchronization control of EtherCAT network is described as follows:



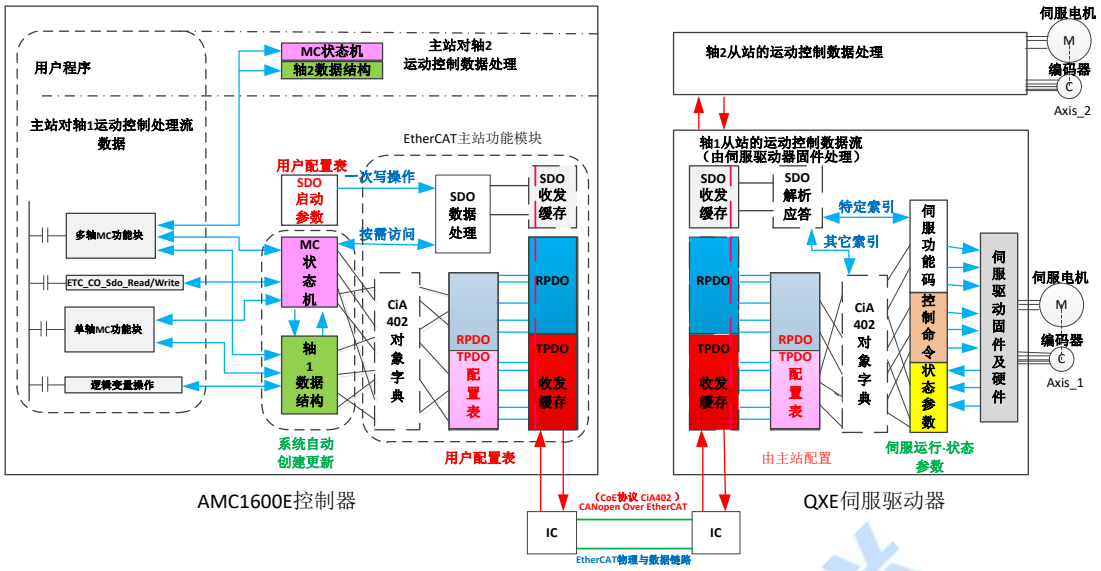
In the EtherCAT network, no DC configuration (such as IO slaves, etc.) is required for slaves that do not support distributed clock (DC). When performing clock synchronization, the EtherCAT master will ignore its clock calibration. The processing and forwarding delays are processed as physical connection delays by surrounding slave stations that support DCs.

During EtherCAT network initialization, the system will initialize the DCs. During the distributed initialization, the system will measure the transmission delay Tdalay between the reference clock and all slave clocks, as well as the initial offset Toffset between the slave clock and the reference clock, and write the above 2 values to each slave station. In this process, the preliminary synchronization of the slave clock is done.

The clock synchronization of EtherCAT network communication is controlled and completed by the system itself. When configuring EtherCAT slaves, users should enable the DC mode of the device that supports DC functions (i.e. servo), and try to arrange it at the front end of the bus network.

4.3 Communication process between AMC1600 and servo slave

The COE protocol is used by the application layer during the EtherCAT communication of AMC1600. When the controller executes the MC (Motion Control) user program, the data of communication between the control system and the EtherCAT servo is processed by multi-stage functional units. The flow is shown in the figure below:



The left side of the above figure shows the AMC1600 controller, and the right side shows the QXE servo drive and its supporting servo motor system. The network interface and protocol of the controller and the drive conform to the PLCopen specification. Both of them are compatible and interchangeable with third-party device. The internal communication process is also applicable to the third-party device using the CiA402 protocol. After understanding the usage and characteristics of each function block in the figure, you're regarded as mastering the motion control principle and method based on EtherCAT network control.

The source of the motion control instruction is the MC function block in the user program, and the object of the control operation is the servo axis. The state of the motion control is stored in the content of the main controller in the form of "axis data structure" to facilitate the user program access.

4.3.1 Operating steps of control information

Step1:Execute the MC motion function block of the user program,and process the command data to be sent

When the controller is processing the user program, it executes the MC motion control function block instance, for example, MC MoveRelative (Axis_1), for which the controller checks the current state of the slave axis based on the state machine and data structure of the slave (Axis_1) in the memory:

1). MC execution error will be reported if the slave axis is not enabled, or is running in torque mode, or running in synchronous mode, or is running in Homing, or is warning;

2). If the slave axis is in the non-synchronous running state under which it stops or in the location mode, send the command ControlWord to allow the slave axis running;

3). Analyze the fSetPositio, fSetVelocity and the constraint conditions such as the target position, allowable maximum speed, acceleration/deceleration of the slave axis, and calculate the motion location command TargetPosition required for the next running cycle;

4). Also, the controller needs to wait for the data sent from the slave of the next cycle, analyzes and judges the execution of this MC function block instruction, to allow the user to know the execution status of the instruction: Busy (executing), Done (finished), Error (execution error) or CommandAborted (this command was interrupted by other commands) and so on.

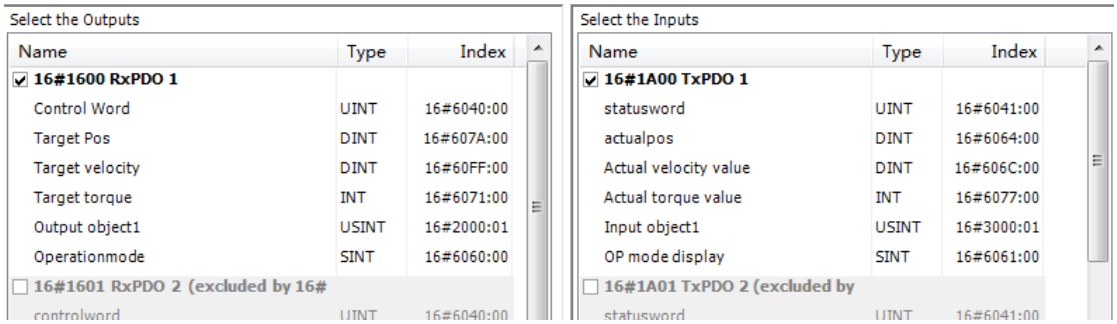
Step 2: Send the control command data to be sent to the sending cache unit of EtherCAT

Store the command data ControlWord and TargetPosition that need to be sent to the slave Axis_1 in the PDO cache unit. To complete this operation, there must be 2 parameters (called "objects" in CiA402) options in the PDO configuration table;

The "PDO configuration table" stores the "index numbers" (main index and sub-index numbers) of the control parameters (objects) that need to be sent and read by the master station;

	Use description
TPDO configuration table	When the table needs to be programmed by the user, the corresponding object and attribute list table will be configured according to the contents that need to be circularly sent in the slave station control; This table is sent by the controller to the slave ESC in the network initialization phase; The master station of the controller arranges the size of the transmit buffer according to the table, and stores the command data to be sent into the transmit buffer at runtime; The slave station parses the received data frame according to the table; Each slave can have a different TPDO configuration table;
RPDO configuration table	When the table needs user programming, configure the corresponding object and attribute list table according to the object content to be answered by the slave station; This table is sent by the controller to the slave ESC in the network initialization phase; When the master station visits the slave station, it inserts the data into the time slot of EtherCAT data frame in time and returns it to the master station; When running, the master station analyzes the data frame returned by the slave station according to this table (data responded by the slave station); Each slave can have different rpdo configuration tables;

The figure below shows the PDO configuration table in CODESYS software. The index number and data type of each control parameter are regulated by CiA402 profile. We can also know the Parameters in the "object dictionary OD", the width and type of parameters and so on through the "index number".



In the initial network stage, the master station will send this PDO configuration table containing TPDO, RPDO, object width, and type to the slave station, which will be used as the basis for the data analysis carried out by the slave station.

The index number and data width of each object are stored in the TPDO configuration table. The order of each object in the table will be the basis for sending the cache unit of the data to be sent by the system MC command. According to the figure above, put ControlWord in the first sending unit, targetPosition in the second unit, and so on;

The slave station, according to the RPDO configuration table, stores the operation status data of the servo in the response cache unit based upon the index number and sequence of each object. When the communication frame of the master station accesses the slave station, the ESC inserts the data of the cache unit into the appropriate time slot of the data frame automatically, and returns it to the master station;

The RPDO table is also the basis on which the master station analyzes the response data from the slave station.

Step 3: The control chip of master station sends the data of the sending cache unit to the slave station ESC at a fixed time, and the slave station responds to the data simultaneously

As the master station, the controller initiates EtherCAT communication as per the EtherCAT clock cycle set by the user, and sends the data of the PDO cache unit to multiple slave stations, via one or several frames. Meanwhile, it retrieves the response data of each slave station is sent to the same communication frame.

From the view of time sequence, the data of the cache unit sent by the controller is the command data produced by the POU during the execution of the previous EtherCAT cycle;

The response data of the slave station is not for the response upon receipt of the inquiry of the master station, but the current value of the cyclic response "object" that is based on the RPDO configuration requirements.

Step 4: The slave station receives and parses the data sent from the master station

After the EtherCAT network enters the normal running state, the slave ESC will receive the communication data frame sent from the master station at the fixed time, and store the received data in the communication data frame into the local caching automatically.

After receiving a string of PDO data from the slave processor, intercept the received data string by the specified object data type (width) according to the TPDO table, and store it to the corresponding control command unit by the parameter attribute represented by the "object index to facilitate servo operation control;

The slave processor will refresh the response cache unit in the local ESC cyclically based on the object attributes and sequence required by the RPDO configuration table. In the right communication frame slot, ESC inserts the cache data into the EtherCAT communication frame by high-speed hardware operation and "sends" it to the master station.

Step 5: The master station receives and parses the response data from the slave station, and updates the parameters of axis status

As the EtherCAT network master station, the controller will receive the communication frame returned by the closed loop of the slave station network while sending data frames, from which it can extract the data string responded by each slave station, judge the network communication status and analyze if the communication operation is successful or not.

According to the data (e.g. StatusWord, ActualPosition, and ErrorCode) responded from the slave station, the controller system can judge whether the servo has reached the running location as required by the instance of MC function block, and refresh the output variable status of the MC function block instance; meanwhile, the controller system software will update the data structure of the axis state parameters promptly to allow user program access.

The above explains the sending, receiving and parsing principle and flow of EtherCAT packet in the AMC1600 controller, so that users can understand its internal mechanism. Many of the links are done automatically by the system. Users need to understand the concept of CiA402 objects, know the commonly used "object" type of the servo axis, and make object selection of the TPDO and RPDO configuration tables.

4.3.2 CiA402 protocol and common servo object dictionary

The EtherCAT protocol layer of AMC1600 adopts the CoE (CANopen Over EtherCAT) protocol. The application layer adopts the CiA402 protocol (a standard protocol for servo and motion control in the CANopen protocol). CiA402 protocol is applied widely in the EtherCAT bus-based network control.

The core of CiA402 is as follows:

Define the functional attributes of "Object Dictionary (OD)" and "Object", and unify the parsing way of communication data.

Periodic process data communication adopts the communication mode of configuring the object first, followed by the periodic sending of object parameters according to the structure of configuration frame;

The additional communication fields are used for non-periodical data communication, for purpose of question-and-answer operations.

CiA402 protocol summarizes the representative setup parameters, control parameters and status parameters in the servo motion control into "objects" with fixed numbers (index number + sub-index number). The complete object definition table can be the "objects dictionary".

4.3.2.1 Common Object Dictionary

In the CODESYS software, the following objects are available for the "SoftMotion CiA402 axis"-"SoftMotion drive: zoom/mapping"-"mapping" project of the EtherCAT bus axis device:

output			
Periodic objects	Number of objects	address	type
ContrlWord(out.wContrlWord)	16#6040:16#00	‘’	
set postion(diSetPosition)	16#607A:16#00	‘’	
set velocity(diSetVelocity)	16#60FF:16#00	‘’	
set torque(wSetTorque)	16#6071:16#00	‘’	
Modes of operation(OP)	16#6060:16#00	‘’	
Touch Probe Function	16#60B8:16#00	‘’	
Add velocity value	16#60B1:16#00	‘’	
Add torque value	16#60B2:16#00	‘’	
Digital outputs(A637)	16#60FE:16#01	‘’	

Input			
Periodic objects	Number of objects	address	type
status word(in.wStatusWord)	16#6041:16#00	‘’	
actual postion(diActPosition)	16#6064:16#00	‘’	
actual velocity(diActVelocity)	16#606C:16#00	‘’	
actual torque(wActTorque)	16#6077:16#00	‘’	
Modes of operation Display(OP)	16#6061:16#00	‘’	

digital inputs (in.dwDigitalInputs)	16#60FD:16#00	‘’	
Touch Probe Status	16#60B9:16#00	‘’	
Touch Probe 1 rising edge	16#60BA:16#00	‘’	
Touch Probe 1 falling edge	16#60BB:16#00	‘’	
Touch Probe 2 rising edge	16#60BC:16#00	‘’	
Touch Probe 2 falling edge	16#60BD:16#00	‘’	
Following error (A632)	16#60F4:16#00	‘’	

The periodic objects of the I/O on the left correspond directly to the parameters in the axis object or some function blocks of CODESYS. When PDO mapping is performed on the bus axis device, a mapping relationship with it will automatically be established by the related PDO object in the left column. When the motion function blocks as well as the related instructions are executed, the PDO object with which the mapping relationship is established can be changed by modifying the output value of the object on the left. The introduction to commonly used ODs focuses on the above objects.

Controlword 6040H

This object is the control word given by the master to the servo. It is used to control the Enable, Start, Stop, Reset, etc. of the servo. This object will be used in many servo control modes. It is an object required to be configured in PDO configuration.

Index number	6040H
Name	Controlword
Data structure	VAR
Data type	UNSIGNED16
Access properties	RW（Readable and writable）
PDO configuration	YES
Value range	0-65535
Default value	No

This object is the control word through which the main controller unit can control the servo. It is defined as follows:

Bit	Name	Instruction
0	Servo ready	
1	Power on the main circuit	
2	fast stop	
3	Servo operation	
4-6	Control mode dependent	
7	Fault reset	Reset the executable fault. The rising edge is valid. When held at 1, other control commands are invalid.

8	Pause	
9-10	Reserve	Reserve
11-15	Factory defined	Reserve

Target position 607Ah

The target position command sent to the servo from the main controller. The data type is of a 32-bit signed integer (DINT). It is required to configure this object in RxPDO when the servo is running in cycle synchronous position (CSP) and profile position (PP) modes. Since the servo runs primarily in CSP mode in practical applications, it is the object that must be configured in the configuration of PDO.

Index number	607AH
Name	Target position
Data structure	VAR
Data type	INTEGER32
Access properties	RW（Readable and writable）
PDO configuration	Yes
Value range	-2147483648-2147483647
Default value	No

Servo pre-operation mode 6060H (Modes of operation)

The main controller can set and change the running mode of the servo via the 6060H object.

Index number	6060H
Name	Modes of operation
Data structure	VAR
Data type	INTEGER8
Access properties	RW（Readable and writable）
PDO configuration	Yes
Value range	-
Default value	No

The control mode corresponded to the control word 6060H is as follows:

6060h	Operation mode
0	Reserve
1	PP profile position mode
2	Reserve
3	PV profile velocity mode
4	PT profile moment

	mode
5	Reserve
6	HM home mode
7	-
8	CSP synchronous periodic position mode
9	CSV synchronous periodic velocity mode
10	CST synchronous periodic moment mode

Target velocity 60FFh

When the servo is running in cyclic synchronous velocity (CSV) and profile velocity modes (PV), set the velocity by using this object. The data is a 32-bit signed integer (DINT), and its dimension is the physical dimension set by the user, which is generally plus/s.

Index number	60FFh
Name	Target velocity
Data structure	VAR
Data type	INTEGER32
Access properties	RW（Readable and writable）
PDO configuration	Yes
Value range	-2147483648-2147483647
Default value	No

Target Torque 6071h

When the servo is running in cyclic synchronous torque (CST) or profile torque modes (PT), set the torque using this object. The dimension is one thousandth of the torque, that is, when the given value is 1, it indicates 0.1% of the rated torque, and when the given value is 1000, it indicates 100% of the rated torque.

Index number	6071h
Name	Target Torque
Data structure	VAR
Data type	INTEGER32
Access properties	RW（Readable and writable）
PDO configuration	Yes
Value range	-2147483648-2147483647
Default value	No

The Max profile velocity 607Fh

This object is used to set the maximum velocity limit in profile mode. It is the maximum velocity limit in CSV or PV mode. In the CST and PT modes, use the set value of this parameter as the maximum velocity limit, so as to avoid excessive velocity and impact on the device. This maximum velocity limit is invalid for the CSP mode.

Index number	607Fh
Name	Max profile velocity
Data structure	VAR
Data type	INTEGER32
Access properties	RW（Readable and writable）
PDO configuration	Yes
Value range	-2147483648-2147483647
Default value	No

Touch Probe Function 60B8h

This object is used by the main controller to control the start and stop of the touch probe function of servo. The input signal (DI) on the servo is set as the probe trigger signal. After the servo touch probe function is enabled, and when the probe signal DI is triggered, the servo grabs the current position information of the motor by calling the interrupt and stores it in the appropriate variable to allow the reading and use by the main controller. When bus servo touch probe function is used, configure this item to PDO.

Index	60B8h
Name	Touch Probe Function
Data structure	VAR
Data type	INTEGER16
Access properties	RW（Readable and writable）
PDO configuration	Yes
Value range	-
Default value	0

The 60B8 touch probe function has the following meanings:

Bit	Instruction
0	0- Closing probe 1； 1- Enable probe 1；
1	Probe 1 trigger mode 0- Single trigger is only valid when the

	signal is triggered for the first time; 1- Continuous triggering;
2	Probe 1 detection signal selection 0- Driver input terminal; 1- Motor Z pulse;
3	Reserve
4	0-The rising edge of probe 1 does not latch; 1-Probe 1 rising edge latch;
5	0-Probe 1 falling edge does not latch; 1-Probe 1 falling edge latch;
6-7	Reserve
8	0-Closing probe 2; 1-Enable probe 2;
9	Probe 2 trigger mode 0-Single trigger, only valid when the signal is triggered for the first time; 1-Continuous triggering;
10	Probe 2 detection signal selection 0-Driver input terminal; 1-Motor Z pulse;
11	Reserve
12	0-The rising edge of probe 2 does not latch; 1-Probe 2 rising edge latch;
13	0-Probe 2 falling edge does not latch; 1-Probe 2 falling edge latch;
14-15	Reserve

NOTE:

When using the function of servo Touch Probe 1:

60B8h (Touch Probe Function) needs to be configured in the output RxPDO;

60B9h, as well as the 60BAh (Touch Probe 1 rising edge) or 60BBh (Touch Probe 1 falling edge) needs to be configured in the input TxPDO (Touch Probe Status). The configuration of 60BAh or 60BBh during use is determined by the trigger condition of Touch Probe 1. Select 60BAh if the Touch Probe 1 is triggered by a rising edge, and select 60BBh if the Touch Probe 1 is triggered by a falling edge.

When using the function of servo Touch Probe 2:

60B8h (Touch Probe Function) needs to be configured in the output RxPDO;

60B9h, as well as the 60BAh (Touch Probe 1 rising edge) or 60BBh (Touch Probe 1 falling edge) needs to be configured as well as the 60BCh (Touch Probe 2 rising edge) or 60BDh (Touch Probe 2 falling edge). The configuration

of 60BCh or 60BDh during use is determined by the trigger conditions of Touch Probe 2. Select 60BCh if the Touch Probe 2 is triggered by a rising edge, and select 60BDh if the Touch Probe 2 is triggered by a falling edge.

Statusword 6041H

This object feedbacks the current running status of the servo. It is an object that must be configured for TxPDO input.

Index	6041H
Name	Statusword
Data structure	VAR
Data type	UNSIGNED16
Access properties	RO（Readable ）
PDO configuration	Yes
Value range	0 65535
Default value	No

6064h Position actual value

It indicates the dimension of the actual position the motor. Its dimension is the physical dimension set by the user. This object is a mandatory option in the PDO configuration table, as the bus axis is mostly used for position control.

Index	6064h
Name	Position actual value
Data structure	VAR
Data type	INTEGER32
Access properties	RO（Readable）
PDO configuration	Yes
Value range	-2147483648-2147483647
Default value	No

606Ch Velocity actual value

It indicates the current real-time velocity value of the motor. Its dimension is the physical dimension set by the user.

Index	606Ch
Name	Velocity actual value
Data structure	VAR
Data type	INTEGER32
Access properties	RO（Readable）
PDO configuration	Yse

Value range	-2147483648-2147483647
Default value	No

6077h Torque actual value

It indicates the current real-time torque value of the motor. Its dimension is one thousandth of the rated torque. The symbol represents the direction of the torque, and its absolute value represents the magnitude of the torque. When the feedback value is 1, it indicates 0.1% of the rated torque, and when the feedback value is 1,000, it indicates 100% of the rated torque.

Index	6077h
Name	Torque actual value
Data structure	VAR
Data type	INTEGER16
Access properties	RO（Readable）
PDO configuration	Yes
Value range	-32768-32767
Default value	No

6061h modes of operation display

This object dictionary shows the current operating mode of the servo. Its value is required to be consistent with the 6060h object dictionary (servo pre-operational mode).

Index	6061h
Name	Modes of operation display
Data structure	VAR
Data type	INTEGER8
Access properties	RO（Readable）
PDO configuration	Yes
Value range	-
Default value	No

You can view the current running mode of the servo through the 6061h.

6061h	Operation mode
0	Reserve
1	PP profile position mode
2	Reserve
3	PV profile velocity mode
4	PT profile moment mode
5	Reserve
6	HM home mode

7	-
8	CSP synchronous periodic position mode
9	CSV synchronous periodic velocity mode
10	CST synchronous periodic moment mode

Note: 6061h is the current running mode of the servo. 6060h cannot be regarded as the current running mode of the servo.

60B9h Touch Probe Status

This object displays the setting status of the touch probe port and the trigger status of the probe. It allows the main controller to judge the effectiveness of the touch probe, and accurately acquire the position of the touch probe when it is triggered. This object must be configured into the PDO when it is necessary to use the touch probe function.

Index	60B9h
Name	Touch Probe Status
Data structure	VAR
Data type	INTEGER16
Access properties	RO（Readable）
PDO configuration	Yes
Value range	-
Default value	No

The 60B9h object is defined as follows:

Bit	Instruction
0	2- Closing probe 1； 0- Enable probe 1；
1	0- 0-probe 1 rising edge latch not executed 1- Probe 1 rising edge latch execution completed
2	0-Probe 1 falling edge latch not executed 1-Probe 1 falling edge latch execution completed
3-5	保留
6	0-The driver input terminal is used as the trigger signal of probe 1; 1-The motor Z pulse is used as the trigger signal of probe 1;
7	-
8	0-Probe 2 not enabled

	1-Probe 2 enabled
9	0-Probe 2 rising edge latch not executed 1-Probe 2 rising edge latch execution completed
10	0-Probe 2 falling edge latch not executed 1-Probe 2 falling edge latch execution complete
11-13	Reserve
14	0-The driver input terminal is used as the trigger signal of probe 2; 1-The motor Z pulse is used as the trigger signal of probe 2;
15	-

Touch Probe 1 position feedback 60BAh and 60BBh

1. Touch Probe 1 rising edge trigger position feedback 60BAh (Touch Probe 1 rising edge)

2. Touch Probe 1 falling edge trigger position feedback 60BBh (Touch Probe 1 falling edge)

When using the Touch Probe 1 function, the object in relation to the position feedback of Touch Probe 1 must be configured in TxPDO. To select 60BAh or 60BBh, you need to determine according to the trigger conditions of Touch Probe 1. If Touch Probe 1 is latched at the rising edge, configure 60BAh in TxPDO; if Touch Probe 1 is latched at the falling edge, configure 60BBh in TxPDO.

Index	60BAh	60BBh
Name	Touch Probe 1 rising edge	Touch Probe 1 falling edge
Data structure	VAR	VAR
Data type	INTEGER32	INTEGER32
Access properties	RO（Readable）	RO（Readable）
PDO configuration	Yes	Yes
Value range	-	-
Default value	No	No

Touch Probe 2 position feedback 60BCh and 60BDh

1. Touch Probe 2 rising edge trigger position feedback 60BCh (Touch Probe 2 rising edge)

2. Touch Probe 2 falling edge trigger position feedback 60BDh (Touch Probe 2 falling edge)

When using the Touch Probe 2 function, configure the object in relation to the Touch Probe 2 position feedback in TxPDO. To select 60BCh or 60BDh, you need to determine according to the trigger conditions of Touch Probe 2. If Touch Probe 2 is latched at the rising edge, configure 60BCh in TxPDO; if Touch Probe 2 is latched at the falling edge, configure 60BDh in TxPDO.

Index	60BCh	60BDh
Name	Touch Probe 2 rising edge	Touch Probe 2 falling edge
Data structure	VAR	VAR
Data type	INTEGER32	INTEGER32
Access properties	RO（Readable）	RO（Readable）
PDO configuration	Support	Support
Value range	-	-
Default value	No	No

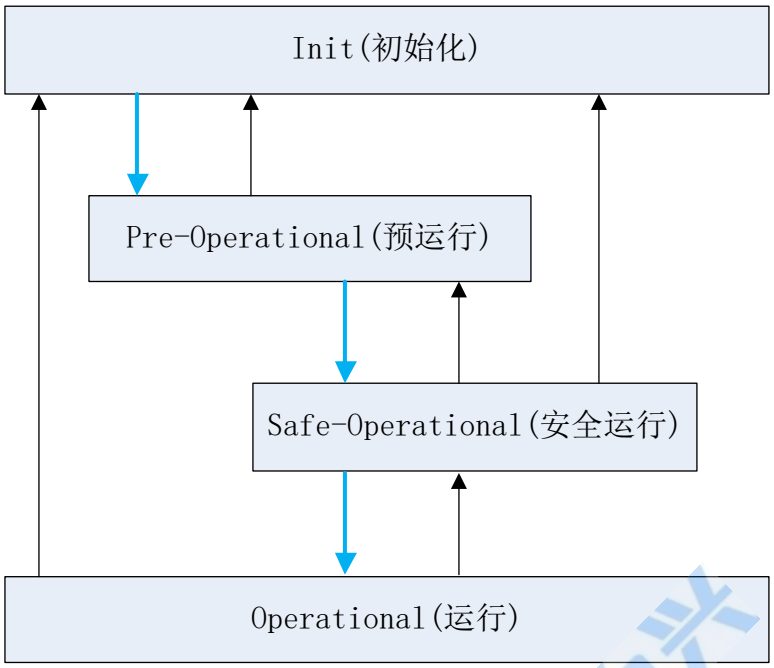
603Fh Error code

Record the error code output when the drive fails (in case of multiple errors, only the last error code will be recorded). By reading the value of this object on the main controller, you can locate the servo fault accurately (the analysis method of the servo error code as well as the code meaning are varied for different manufacturers).

Index	603Fh
Name	Error code
Data structure	VAR
Data type	UNSIGNED16
Access properties	RO（Readable）
PDO configuration	Support
Value range	-2147483648-2147483647
Default value	No

4.3.3 State transition of EtherCAT communication

After the 1600E is powered on, the operating system and user programs will be loaded completely in 15 sec. When the system is provided with EtherCAT bus device, AMC1600 will, as the EtherCAT master, initialize the EtherCAT network. The initialization process of EtherCAT network is operated according to the conversion relationship of the EtherCAT state machine, which is shown as follows:



The EtherCAT state machine coordinates the state relationship between the master and the slave application programs during initialization and running.

To transition from the initial state to the running state, you need to perform the transition in the order of "Initialization -> pre-operational -> Safe run -> Run" You may skip the certain transition process when returning from the running state. All the state changes are initiated by the master station. The master station sends a control command to the slave to request a new state. The slave station responds to this command, performs the requested state transition, and writes the results to slave state variables. If the requested state transition fails, the slave will give an error flag.

(1) Initialization (Init):

The initialization state defines the initial communication relationship between the master and the slave at the application layer. At this moment, the direct communication between the application layers of the master and the slave is unavailable. The master station initializes some configuration registers of the ESC using the initialization state. If the slave station supports communication via mailbox, configure the parameters of mailbox channel.

(2) Pre-Operational:

The mailbox data is activated in the Pre-operational state. Both the master and the slave can use mailbox data communication to exchange initialization operations and parameters of the application.

(3) Safe-Operational

In the Safe-Operational state, the slave application program reads input data, but does not produce output signals. The device makes no output and is in a "safe state". Mailbox communication is still available at this moment.

(4) Operational:

In the Operational state, the slave application reads input data, the master application sends out output data, while the slave device generates output signals. Mailbox data communication is still available at this moment.

The EtherCAT status and its conversion process are as listed in table below:

State and state transformation	operation
initialization（Init）	The application layer has no communication, so the master station can only read and write ESC register;
Initialization to pre-run conversion （Init to Pre-Op）	Master station configures slave station address register; If mailbox communication is supported, configure mailbox channel parameters; If distributed clock is supported, DC related registers are configured; Master station writes status control register to request "Pre-OP" status;
Pre-run （Pre-Op）	Application layer mailbox data communication;
Conversion from pre-operation to safe operation （Pre-Op to Safe-Op）	The master station uses mailbox initialization process data mapping; The master station configures the SM channel for data communication; The master station is equipped with FMMU; Master station writes status control register to request "Safe-OP" status;
safe operation （Safe-Op）	The application layer supports mailbox data communication; There is process data communication, but only the input data can be read without output signal;
Safe operation to operation （Safe-Op to Op）	The master station sends valid output data; The master station writes the status control register to request the "Op" status
run （Op）	Input and output are all valid; Can use mailbox communication;

When the EtherCAT is in OP state, the slave device is abnormally disconnected and then connected again (for example, the network cable is disconnected, or the power failure of slave device), the master device can no

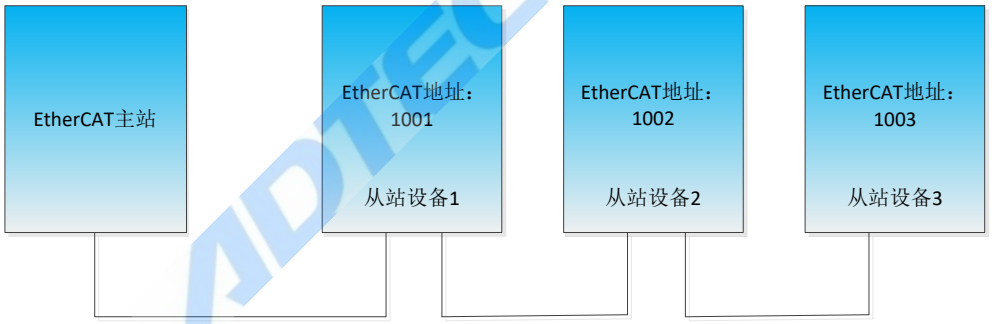
longer initialize and configure its parameters, and the slave device may not run again.

4.3.4 Slave address setting

4.3.4.1 Sequential addressing

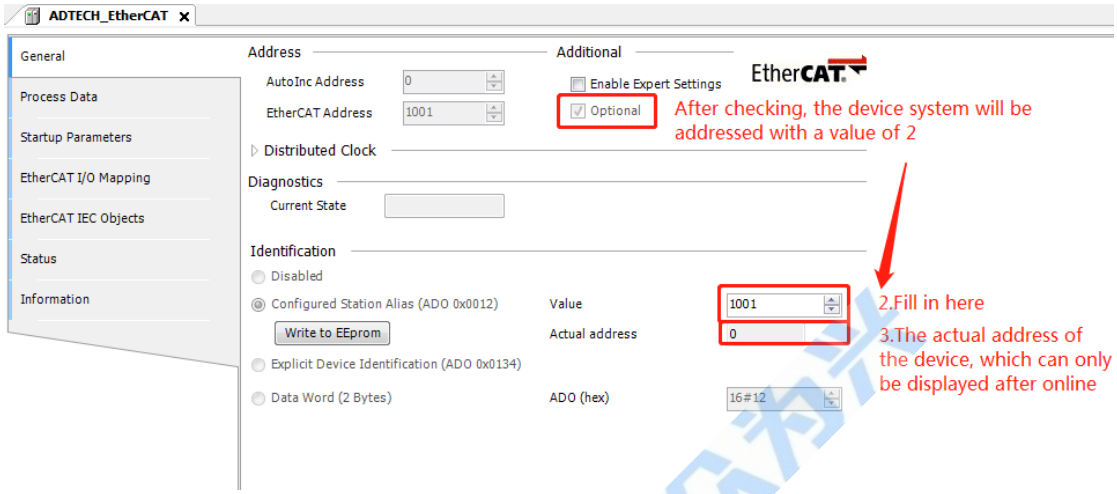
When making EtherCAT network configuration in a user program, the system will, by default, automatically configure the EtherCAT addresses of the slave as per the order in which the slaves in the network are connected. The first slave has an EtherCAT address of 1001, and the second slave has an EtherCAT address of 1002. The EtherCAT addresses of the subsequent slaves in series are allocated on the basis of this rule until the last slave device.

By using such address configuration, users do not need to set the station number of the slave and consider the problem of the same station number, but only need to ensure that the connection sequence of the external device is consistent with the device sequence in the EtherCAT network configuration in the program to allow the proper connection between the external device and the program. Meanwhile, main controller under this configuration is allowed to detect the network configuration and identify the hardware connection error. Its network configuration is as shown below:



4.3.4.2 Slave alias configuration function

In some devices, some of the axis functions have been clearly defined by giving a fixed name. The master controller is required to address using such fixed name, and the addressing mode of the slave can be changed to the addressing mode of "slave alias". The way of setting can be as follows:



1. After the slave uses the "alias", the slave address has no connection with its connection sequence in the network.

2. In a user program, one or more devices can be configured by aliases, but they cannot use the same aliases.

3. If some slaves in the system have "alias", some slave addresses can automatically be assigned. Then, the system determines the slaves with "alias", and the remaining slaves are to be handled according to the rules of automatic allocation.

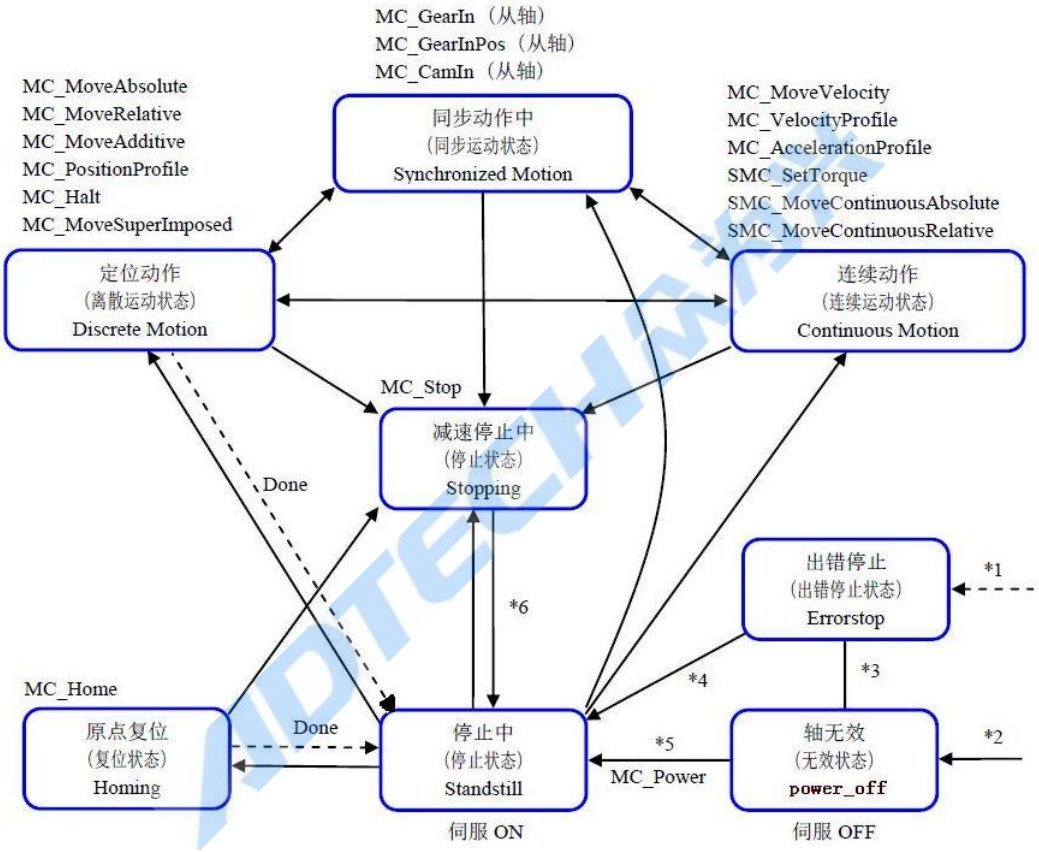
4.3.5 Transition of axis motion states

The axis is classified as several states in motion control. Those states are converted under certain conditions. In the axis data structure, the element AXIS_REF_SM3.nAxisState (Parameter number: 1000) is used to indicate the current state of the axis. The type of nAxisState is of SMC_AXIS_STATE enumeration type. The elements are defined as follows:

Type	Instruction
0: power_off	Axis not enabled, axis invalid state
1: errorstop	Error stop
2: stopping	Deceleration stop
3: standstill	Stopping
4: discrete_motion	Positioning movement

5: continuous_motion	Continuous motion
6: synchronized_motion	Synchronous action
7: homing	Origin Return

The figure below shows the state transition of the axis. The axis is always in one of states as shown in figure below. In the state transition diagram, arrows indicate that transitions between states are available. The state transitions executed by the start function block are indicated by solid arrows. The dashed arrow is used for the ending of the command of the axis or the state transition in case of a system switching (e.g. an error). The function blocks described in states then convert the axis to the corresponding state.



Notes:
*1 When an axis error occurs in any state;
*2 MC_Power's Enable=TRUE, bRegulator=FALSE, and there' no error in the axis in any state;
*3 Status=FALSE of MC_Reset and MC_Power
*4 For the MC_Reset and MC_Power, Enable=TRUE, bRegulator=TRUE, and Status=TRUE
*5 For the MC_Power, Enable=TRUE of , and bRegulator=TRUE, and Status=TRUE
*6 For the MC_Stop, Done=TRUE and Execute=FALSE

When the MC instructions are used, control the running of the axis using the mode shown in the figure above, otherwise the state of the axis will not respond, and the MC function block will get the error message.

In the user program, the subsequent logic can be started based on the state of the axis. Compared with the Done signal of the MC instruction, it will be more accurate to judge by the axis state variable AXIS_REF_SM3.nAxisState. For the actual debugging of the user engineering, in case of abnormal motion control, it is a great way to proceed with the monitoring of the axis state variables so as to identify and analyze the cause.

Understanding the state transition of the axis allows users to know the use of MC instructions correctly. You need to pay attention to the use logic of MC instructions in programming, as such the conflict between motion instructions during program execution will be minimized, and the quality and efficiency of program development will be improved.

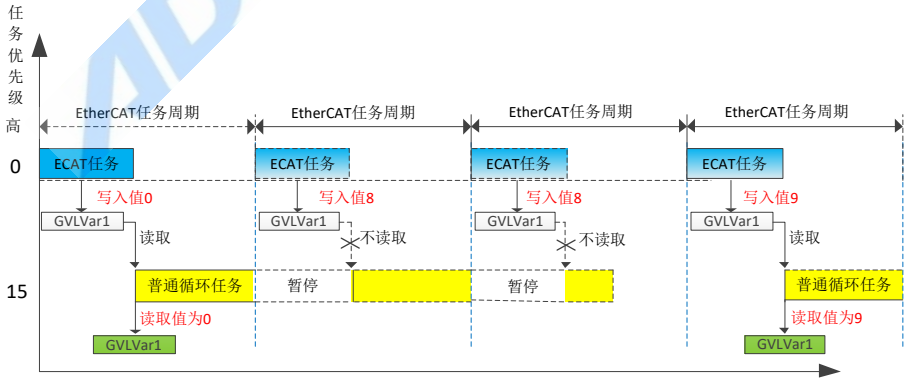
4.3.6 Interaction way of global variables (GVL) between different tasks

Global variables can be used to interact between different POUs. When global variables are used in different tasks, it can be known from the task execution mechanism that:

1. When the task is to be executed, the system reads the value of the buffer cache of the global variable once in the general system processing stage 1.

2. The general system processing stage 2 in the control processing stage will write the changes of the global variables into the global variable buffer cache.

The axis variable is of a global variable defined by the system. Every time the EtherCAT task is executed, the system will automatically refresh this variable. As for other tasks, the data interaction is as follows when reading the value of this variable element.

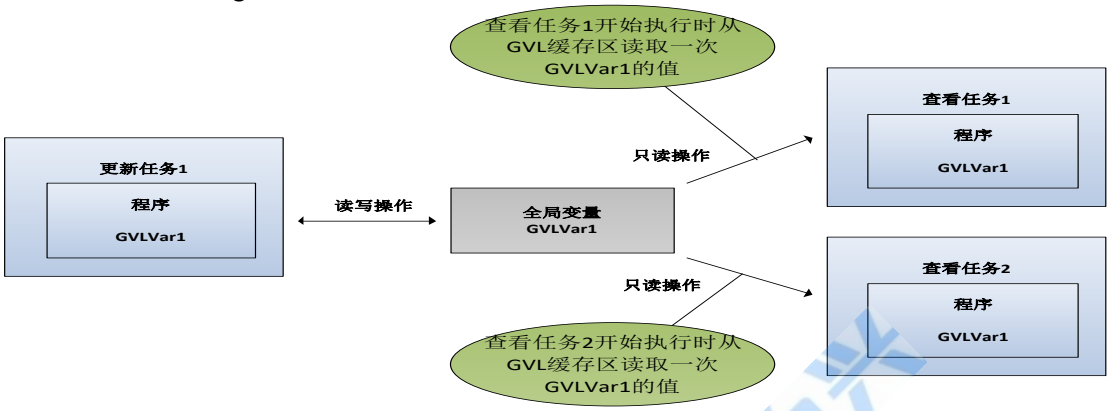


According to the characteristics of GVL data interaction, when a task is executed, the GVL value used by the POU in the task is read from the GVL buffer cache, and it will not be read from the GVL buffer cache in the current round of task execution. Upon the execution of this round of tasks, the value of the written GVL will be assigned to the corresponding GVL buffer cache, and the cache will then be updated. Due to the GVL variable interaction, when multiple tasks read and write global variables, it is necessary to ensure the

simultaneity of global variables between tasks. It is recommended to note the mutual exclusion of global variables during use.

The mutual exclusion way of global variables 1: Read and write global variables in one task, while read the variables only for other tasks.

Only one task can read and write a global variable. This task is defined as the update task. Other tasks that can only read this variable are defined as the view task. The figure below can be used in this manner:



Chapter V Application Programming

5.1 application points of single axis positioning motion

5.1.1 Key points of AMC1600 application programming

Amc1600 controller and servo axis cooperate, and the movement is based on pulse + EtherCAT bus network. The pulse axis is controlled by virtual axis mapping, and the bus axis controls the servo by calculating and issuing a control command in each EtherCAT bus cycle.

For the operation of the bus, users need to pay attention to:
1) The user's MC control program is executed in EtherCAT task cycle. The POU related to MC should be configured under EtherCAT task. If most MC function blocks are placed in the POU of main task with low priority, they cannot run.

2) The execution of MC function block needs the communication data object in the communication to transfer. Therefore, the PDO configuration table must have the necessary configuration items. If the configuration related data object is omitted, the servo may not work normally and there will be no error alarm.

3) The controller can initialize the function code of the servo through the configuration of SDO, so that the operation mode of the servo (generally CSP mode), the encoder mode of the servo motor, the electronic gear ratio, etc. can ensure the correspondence between the control command and the physical operation position; the initialization of the servo can also improve the debugging efficiency of the equipment, and will not make mistakes after the replacement of parts.

4) For the control of servo axis, the rules and logic of axis state should be followed. According to the current state of the axis and the movement of the target, the appropriate MC function block should be used to control.

5) The example of MC function block is used in the user program. One MC example can only be used for the control of one servo axis. If it is used for the control of several servo axes at the same time, it will lead to control confusion.

6) A running servo shaft must have an MC function block to monitor its operation, even MC_ Stop is also a kind of monitoring, to avoid the state without MC function block monitoring due to the jump of program logic, the system will stop and report an error, which is not easy to check.

7) Pay attention to the safety of debugging. If the incremental encoder is used in the servo system, it is necessary to return to zero operation before normal operation. The di signal input port of the servo driver can be connected with the origin position signal. For the movement within a limited range (such as screw rod), there should be limit and safety protection signal before debugging.

5.1.2 Common MC function blocks for single axis control

MC function block (FB) is also called MC instruction. To be exact, the user program uses the object instance of MC function block. The servo axis is controlled by the MC object instance, and the servo axis is controlled by the MC object instance,for example:

```
MC_Power1 :MC_Power; //Declaration instance MC_Power1
MC_Power_1(Axis:= Axis1, );
```

The single axis control is generally used for positioning control, that is, the servo motor drives the external mechanism to move to the specified position; sometimes it also needs the servo to move at the specified speed or torque, etc. in the single axis control, the following MC function blocks are commonly used:

Control operation	**MC function block**	Description
Servo enable	MC_Power	Run the command to enable the servo axis, then the subsequent operation control can be carried out
Servo inching operation	MC_Jog	The inching operation of servo motor is often used in low-speed test run to test equipment or adjust the position of servo motor
Relative positioning	MC_MoveRelative	Take the current position as the reference, run the specified distance
Absolute positioning	MC_MoveAbsolute	Command the servo to run to the specified coordinate point
Relative superposition positioning	MC_MoveAdditive	On the basis of the current operation command, the relative operation command distance is superimposed
Speed control	MC_MoveVelocity	Command the servo to run at the specified speed
Torque control	MC_MoveTorque	Command the servo to run at the specified torque
Servo pause	MC_Halt	Command the servo to pause: If MC_ The movexxx command is triggered again and the servo can continue to run
Emergency stop	MC_Stop	Command servo emergency stop: MC is triggered only after stop command is reset_ Move XXX command, the servo can run
Warning reset	MC_Reset	When the servo is stopped by warning, run the command to reset
Servo operation mode	MC_ControlMode	With this command, the servo can select "position", "speed" or "torque" mode
Servo return to zero	MC_Home	The servo starts the origin regression operation, and the origin signal of the application system and the limit signal on both sides are connected to the DI Port of the servo.

Controller return to zero	MC_Homing	The control system starts the origin regression operation, and the origin signal of the application system and the limit signal on both sides are connected to the DI Port of the controller.

5.1.3 MC instruction and PDO / SDO configuration

When AMC1600 executes the MC control command of servo axis in user program, it needs to add the information items needed to interact with servo when executing MC command to the PDO / SDO configuration table of communication, so as to complete the required function control.

MC instruction	Required TPDO objects	Required RPDO objects
MC_Power	ControlWord	StatusWord Errorcode
MC_Halt		
MC_Stop		
MC_Home		
MC_Homing		
MC_Reset		
MC_Jog	TargetPosition	Position actual value Following error actual value
MC_MoveRelative		
MC_MoveAbsolute		
MC_MoveAdditive		
MC_MoveVelocity	Target Velocity Max Profile velocity	
MC_MoveTorque	Target troque	Torque actual value
MC_ControlMode	Modes of operation	16#6060=8：Periodic synchronization position CSP 16#6060=9：Speed mode 16#6060=10：Torque mode

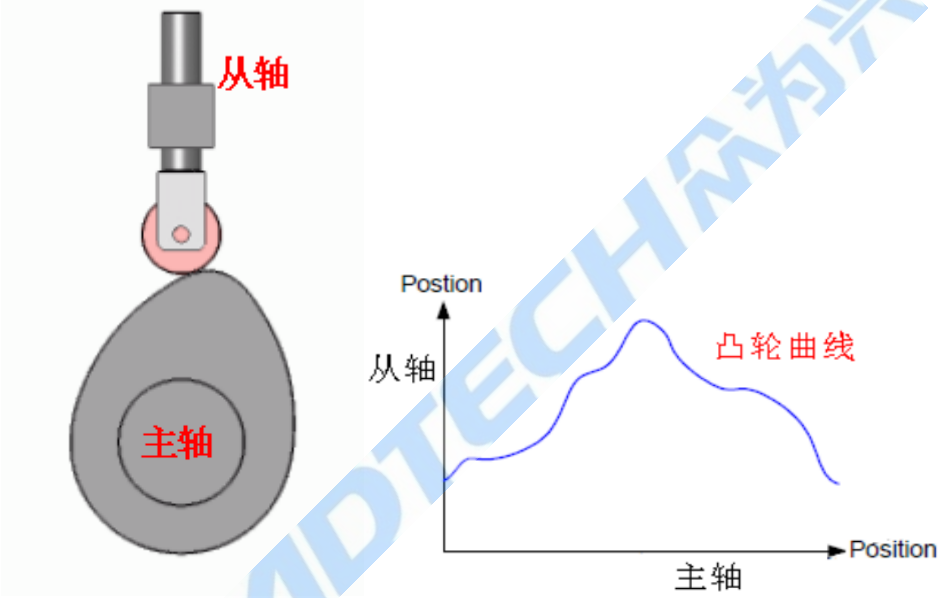
The above TPDO / RPDO is the basic configuration item for single axis control; In MC control, servo is mostly in position control mode, especially in the application system based on EtherCAT bus, which is "periodic synchronous position mode". Therefore, in SDO configuration, servo is usually set to this operation mode. Taking QXE as an example, the following items are usually initialized in SDO:

Initialization of servo	SDO objects to use	Description
Set to periodic synchronous position mode	Modea or operation(16#6060)	Set 8
Set the maximum allowable speed	Max profile velocity (16#2049)(There may be differences between different manufacturers)	It is suitable for speed mode and torque mode
Set the maximum allowable moment	Max troque （16#6072）	It is suitable for speed mode and torque mode
Home method	Homing method（16#6098）	
Home speed	Homing speeds（16#6099）	
Probe function	Touchprobe function（16#60B8）	

5.2 Motion control programming of multiaxis cam synchronization

Cam motion is a kind of motion which borrows the concept of relative motion characteristics of mechanical cam and tappet. It means that the controller makes the servo slave follow the main shaft in a continuous synchronous motion according to the specific nonlinear relationship of relative position, so as to meet the motion characteristics required by the equipment. It is widely used in fixed length cutting, chasing shear control, flying shear control, multi-color overprint and other applications without communication requirements .

The master-slave position relationship of the electronic cam is shown in the figure below. The horizontal axis is the spindle position, while the vertical axis is the slave position:



AMC1600 adopts the software mode to realize the cam motion control characteristics, that is, using the software digital "Cam table" to replace the mechanical cam, so it is also called electronic cam control. Compared with the mechanical cam, it has the following characteristics:

1) Cam shape is easy to make: cam table, cam curve or array are used to describe cam;
2) The cam shape is easy to be diverse: support multiple cam table selection, dynamic switching in operation;
3) It is easy to modify the cam shape: the key points of cam table can be modified dynamically during operation;
4) Cam owns Multiple slave axis : Cams are allowed multiple slave axis；
5) Cam tappet: multiple cam tappets and multiple setting intervals are allowed;
6) Cam clutch: during cam operation, the user program can enter or exit cam operation;
7) Special functions of electronic cam: support virtual spindle, phase offset, output superposition;

The cam operation of amc1600 is carried out by pure software.If in cam running state, every time entering EtherCAT task, the next target point of slave shaft will be calculated, so it has better functional flexibility than hardware Cam operation.

There are three elements in the control of electronic cam
① Spindle: reference axis used for synchronous control;
② Slave axis: according to the spindle position, according to the required nonlinear characteristics to follow the motion of the servo axis;
③ Cam table: a data table or cam curve that describes the relative position and range, periodicity, etc. of the spindle and slave shafts.

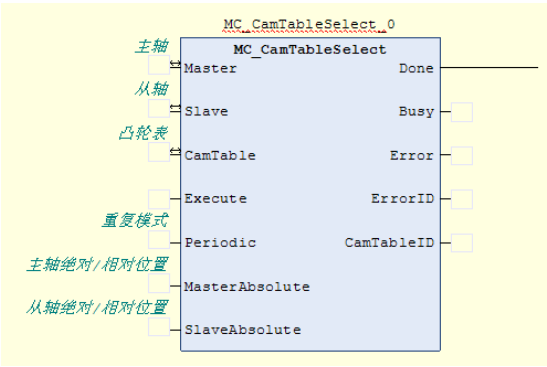
The user needs to design the cam table, specify the spindle and slave shaft, and trigger the cam to run at the right time, then the slave shaft can enter the cam to run.
Basic command function block of electronic cam control:

Control operation	MC instructions to be used	Description
Cam table selection	MC_CamTableSelect	Run the command to associate the relationship between the spindle and the slave cam table
Enter cam operation	MC_CamIn	Let the slave shaft run into the cam
Exit cam operation	MC_CamOut	Let the slave shaft exit the cam operation
Correct cam phase	MC_Phasing	Spindle phase modification

5.2.1 Main function block description of cam operation

1. The cam table function block Selects MC_CamTableSelect

This function block is used to correlate the relationship between the master and slave cam tables, and set the cycle of cam operation, the position mode (absolute position/phase position) of the master and slave axes, etc. The instruction is a management instruction, that is, the instruction is triggered After only one execution, the relevant master-slave axis can continue to run according to this characteristic; if you need to change the cam table or change the master-slave axis, you need to trigger the execution of the function block again.



```
MC_CamTableSelect(
    Master:= , //Cam spindle
    Slave:= ,  //Cam slave
    CamTable:= , //Cam table
    Execute:= , //The instruction triggers the variable, the rising edge
is valid
    Periodic:= , //Cam periodic
    MasterAbsolute:= , //Spindle position mode
    SlaveAbsolute:= , //Slave position mode
    Done=> ,
    Busy=> ,
    Error=> ,
    ErrorID=> ,
    CamTableID=> );
```

Function block description:

- Input and output variables

Input and output variables	type of data	Initial value	Description
Master	AXIS_REF_SM3	—	Spindle mapping
Slave	AXIS_REF_SM3	—	Slave axis mapping
CamTable	MC_CAM_REF	—	Map to CAM table description

- Input variables

Input and output variables	type of data	Initial value	Description
Execute	BOOL	FALSE	Trigger on rising edge, start execution condition when TRUE
Periodic	BOOL	TRUE	Specify whether to execute the cam table repeatedly or only once TRUE:repeat FALSE:Not repeating
MasterAbsolute	BOOL	TRUE	Specify whether the spindle is absolute or relative TRUE:Absolute position FALSE:Relative position
SlaveAbsolute	BOOL	TRUE	The current position of the slave axis is the absolute value output by the cam table or relative. TRUE:Absolute position FALSE:Relative position

- Output variables

Output variable	type of data	Initial value	Description
Done	BOOL	FALSE	TRUE:Phase shift completed
Busy	BOOL	FALSE	TRUE:Command execution
Error	BOOL	FALSE	TRUE:Error signal occurred

ErrorID	SMC_ERROR	0	Output fault code when an error occurs
CamTableID	MC_CAM_ID		Definition of CAM table, select valid CAMID

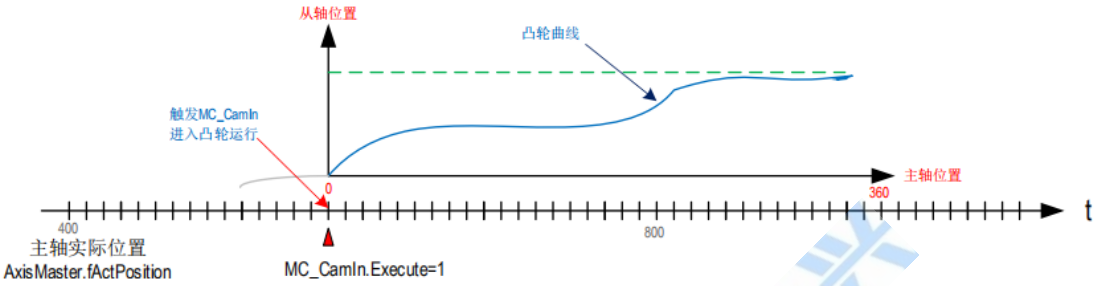
2. MC_CamIn function block for cam operation

After the MC_CamTableSelect function block is run, the Slave can be made to perform the cam operation status (Synchronized_Motion, ie Axis.nAxisStatus=6) in the function block, and the system will execute the function block once every time it enters the EtherCAT task. The current position of the master axis and the cam table are used to calculate the next target point of the slave axis. If the MC_CamTableSelect function block is not run beforehand, an error will be reported when the function block is triggered.


```
MC_CamIn(
    Master:= ,//Spindle
    Slave:= , //Slave axis
    Execute:= , //Execution trigger
    MasterOffset:= , //Spindle position offset
    SlaveOffset:= , // Axis position offset
    MasterScaling:= , //Spindle magnification ratio
    SlaveScaling:= , //Slave zoom ratio
    StartMode:= , //Slave axis trigger position mode
    CamTableID:= , //Cam table pointer
    VelocityDiff:= , //Speed deviation
    Acceleration:= , //Acceleration
    Deceleration:= , //Decrease speed
    Jerk:= , //Jerk
    TappetHysteresis:= , //Tappet backlash
    InSync=> , //Synchronization instructions
    Busy=> , //running
    CommandAborted=> , //Instruction interrupted
    Error=> , //error occurred
```

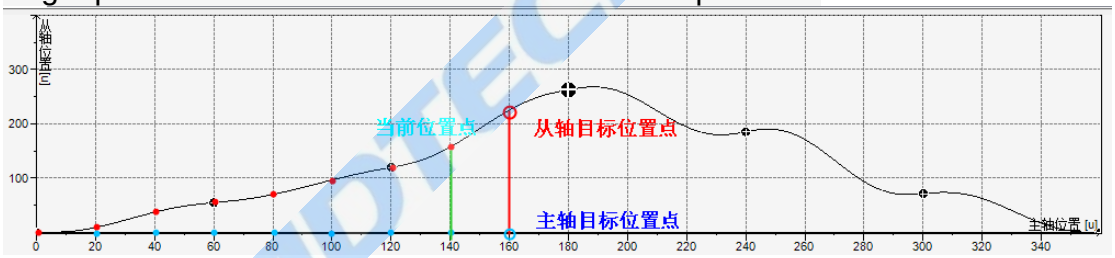
```
ErrorID=> , //error code
EndOfProfile=> , //Executed to the end of the cam
Tappets=> );//Effective tappet
```

The function of this instruction is to make the cam slave axis move in the drawing and the cam spindle synchronously, and control the cam slave axis to adjust to the corresponding target point according to the current position of the spindle and the position relationship of the cam table. The execution of this instruction has no effect on the spindle.

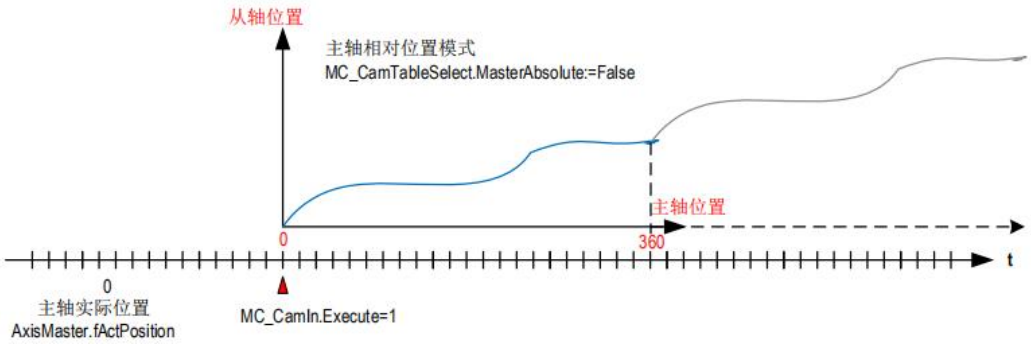


Once MC_CamIn is triggered, the slave axis follows the position of the master according to the position correspondence of the cam table. Note that it is the position correspondence, not the speed correspondence.

After entering the cam operation, the system parses the CAM cam table every EtherCAT interruption, calculates the next target point of the slave axis according to the current position of the master axis, and then sends the next target position to the slave axis to command its operation:



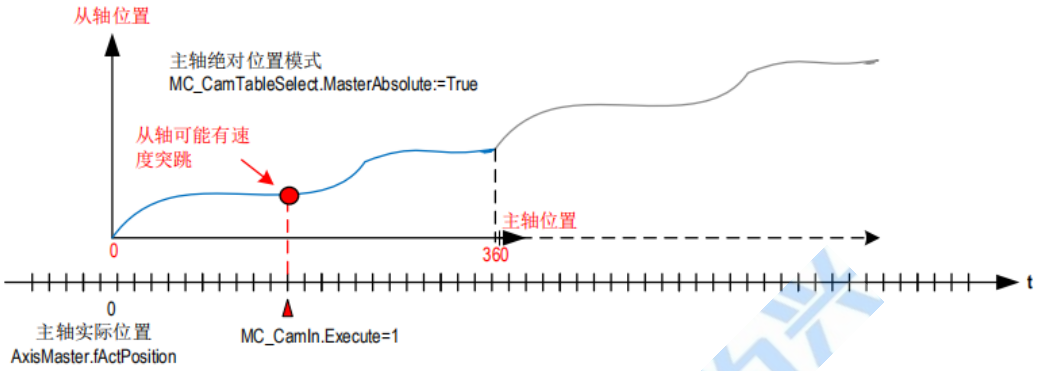
5.2.2 Operating characteristics when both the master axis and slave axis are in relative position mode



When the spindle is in the relative position mode, when entering CAM, the cam calculation module will use the current position as the starting point of the spindle X=0 for calculation;

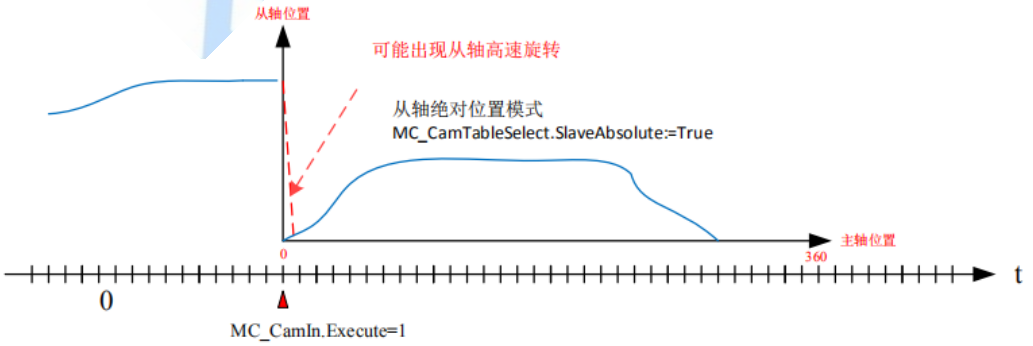
When the slave axis is in the relative position mode, when entering CAM, the cam calculation module will use the current position as the starting point of the slave axis Y=0 for calculation, and the subsequent CAM output results will be superimposed on this basis.

5.2.3 The current master axis is in absolute position mode, and the slave axis is in relative position mode



When the master axis is in absolute position mode, when entering CAM, the cam operation module will calculate the slave axis position based on the current master axis position, so:
The high-speed rotation of the slave axis position adjustment when entering CAM operation, resulting in equipment impact or damage;
If the current position exceeds the valid range of the CAM table, the slave axis will not move and a warning will be issued;
If the CAM table is in periodic mode, when the current cycle is completed, the continuous operation of the next CAM cycle will start.

5.2.4 The current master axis is in relative position mode, and the slave axis is in absolute position mode



When entering CAM operation from axis absolute position mode, it will be adjusted to the position required by CAM. If the deviation is large, automatic adjustment of high-speed motion will occur.

Take countermeasures according to application characteristics:

For equipment that must be aligned, for example, the fixed-length cutting rotary blade must adopt an absolute position as a cam slave axis. When

programming, pay attention to the zero-return operation of the rotary knife before the first rotary cutting action of the rotary knife;

Properly set the spindle position range of the cam table to avoid the position adjustment of the cam in the opposite direction at the beginning of the next cycle;

Run SMC_GetCamSlaveSetPosition to set the slave axis position of the cam entry point to the current coordinate of the slave axis.

For applications that can use the relative position mode, use the relative position mode:

```
MC_CamTableSelect.SlaveAbsolute:=Flase;
```

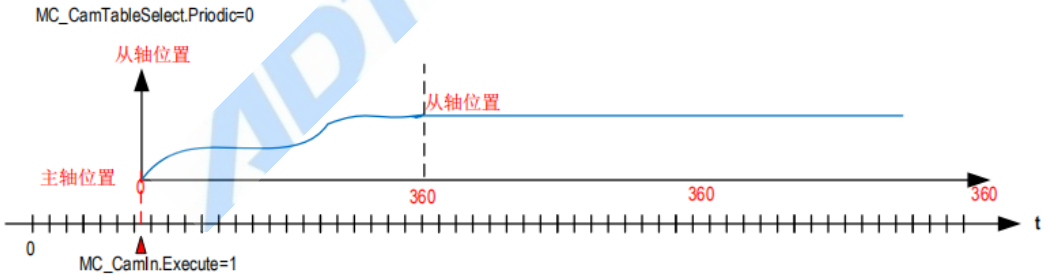
Or set `MC_CamIn.Startmode=1;(Relative position mode)`

note:

When the slave axis is set to the absolute mode of "finite length", when making the zero return adjustment, if it can turn left to return to zero, the controller will choose a relatively close direction to return to zero. When designing the range of the cam table, pay special attention not to allow the range of the cam table to exceed the actual range that needs to be operated, otherwise the servo slave axis may instantaneously rotate and adjust at a high speed, causing mechanical shock.

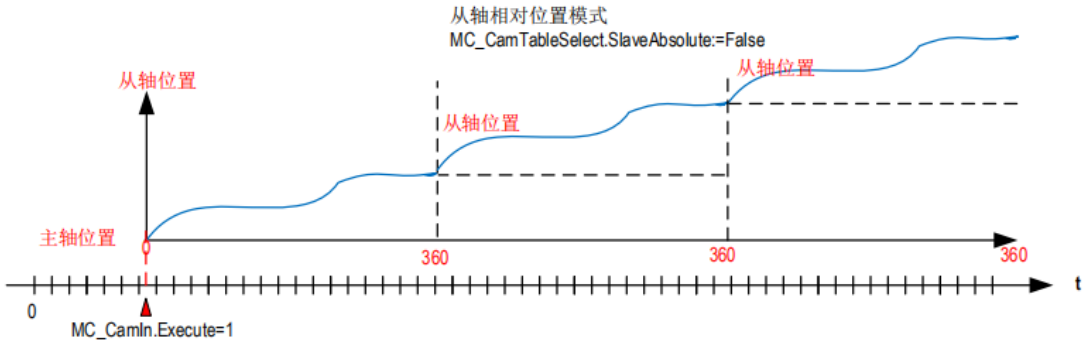
5.3 Periodic mode characteristics of the cam table

The following figure shows the effect of single-cycle cam operation. When the cam table is selected as the single-cycle mode (Periodic:=0), after running a cam table cycle, the slave axis will leave the cam running state.



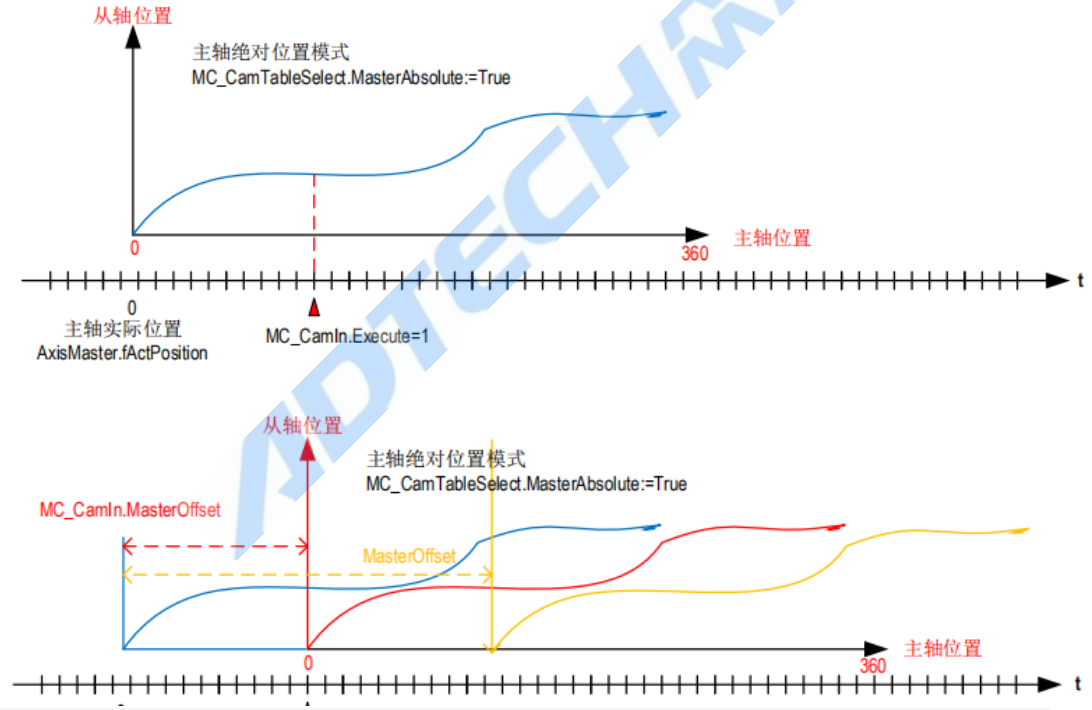
When the cam table is selected as the periodic mode (Periodic:=1), after running a cam table cycle, the slave axis will start the next cam and cycle operation until the user program commands it to exit the cam operation state,

as shown in the following figure:



The above figure shows the effect of periodic cam operation. Once the position range of the spindle in the cam table is completed, the movement of the next cam cycle is automatically started.

5.3.1 Offset function of CamIn operation

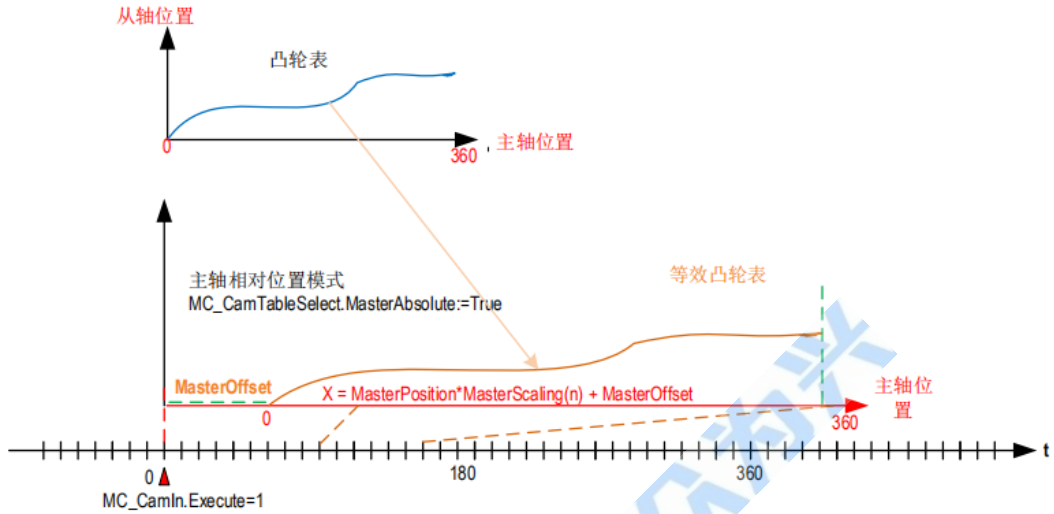


By setting an appropriate offset value for the cam master axis, the starting point of the cam slave axis can be modified. According to the current actual position of the spindle, the offset value can be calculated, which can start from the 0 point of the cam table:

```
MasterOffset := 0-AxisMaster.fActPosition;
```

5.3.2 MasterScaling calculation of spindle during cam operation

By default, the system sets MasterScaling=1. If the user program modifies this variable, then:



Setting the proportional MasterScaling value to the cam spindle can linearly scale the position of the spindle to make the corresponding positional relationship with the cam table meet the desired requirements.

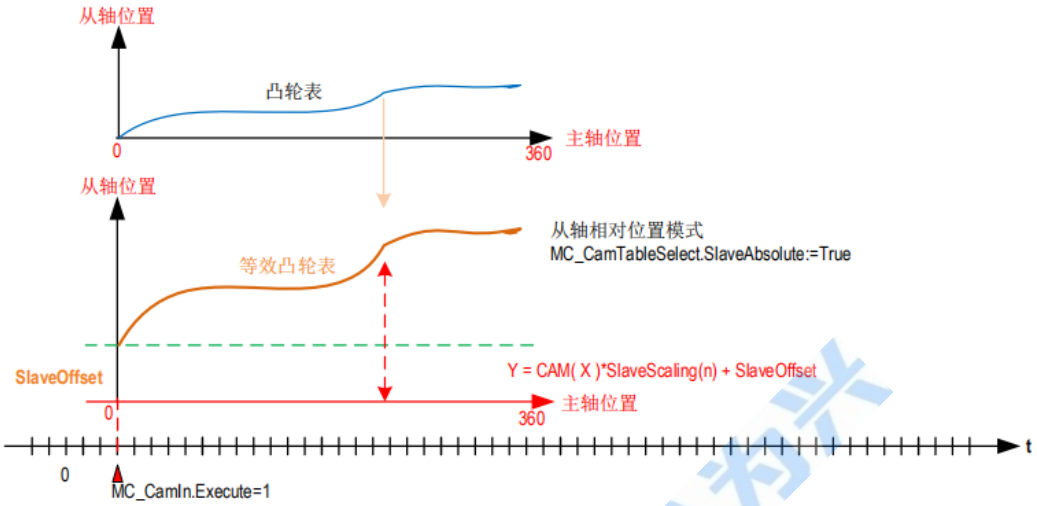
If the offset setting value of the spindle is considered, the calculated position of the spindle (X) in the cam table will be:

```
X = MasterPosition * MasterScaling + MasterOffset;
```

This parameter can be used to fine-tune the size of the machined workpiece.

5.3.3 Calculation of spindle SlaveScaling during cam operation

By default, the system sets SlaveScaling=1. If the user program modifies this variable, then:



Setting the proportional SlaveScaling value for the cam slave axis can linearly scale the position of the slave axis so that the output of the cam control meets the desired position of the slave axis.
If the offset setting value of the slave axis is considered, the output position of the cam slave axis (Y) will be:

```
Y = CAM(X)*SlaveScaling(n)+SlaveOffset;
```

This parameter can be used to fine-tune the size of the machined workpiece.

5.3.4 Offset and Scale usage characteristics and precautions in cam operation

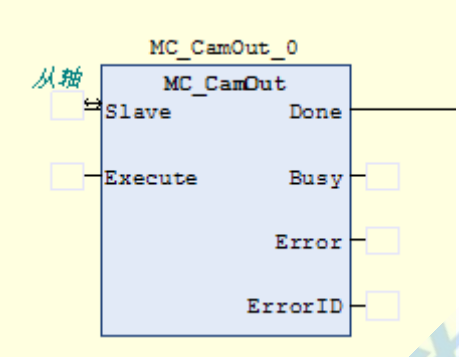
In addition to the special requirements of the application system, it is recommended to use the relative mode as much as possible for the spindle position mode and slave axis position mode, so that the programming is simple and the possibility of mechanical system impact is relatively small;

The starting and ending range of the cam table spindle, Offset, Scale and other settings can compensate for the design deviation of the CAM table. It is recommended to refer to the default setting as much as possible, which is convenient for debugging and maintenance, and the chance of running errors can also be reduced;

When the CAM cam table cycle is executed/or exited/or switched to the CAM table, and MC_CamIn is executed again to re-enter, the system will clear the Offset, Scale and other set values in the memory and restore to the default values. Pay attention.

5.3.5 MC_CamOut function block to exit the cam running state

When the slave axis is cam running, triggering the execution of this function block can make the slave axis exit the cam running state and enter the continuous running state (Continuios_Motion, ie Axis.nAxisStastus=5). The execution of this instruction has no effect on the spindle.



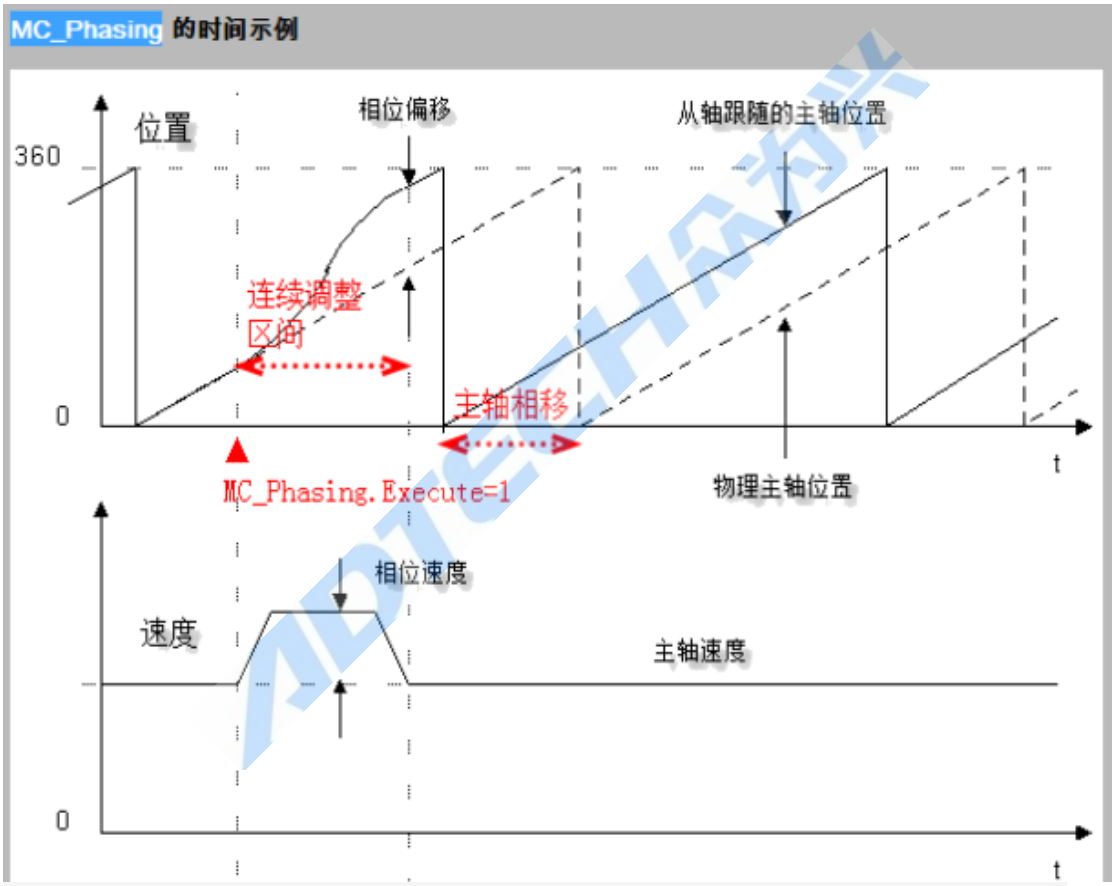

```
MC_CamOut(
    Slave:= , //Cam slave
    Execute:= , //Execute trigger variable, rising edge trigger is valid
    Done=> , //Execution complete
    Busy=> , //In progress
    Error=> , //Error
    ErrorID=> );//Error number
```

Note: By executing this command, the slave axis will leave the cam running state, but its running speed will continue to run at the instant it exits the cam state, just as the device driven wheel will continue to run by inertia after the mechanical clutch is disconnected. At this time, there must be other MC function blocks to take over the motion control of the slave axis, such as MC_Movexxxx, MC_Halt, MC_Stop and so on.

5.4 Cam spindle phase adjustment MC_Phasing function block

In the CAM synchronous operation of some equipment, it is sometimes necessary to correct the relative phase of the cam slave axis and the main axis, which can be achieved by using the MC_Phasing instruction.

When this instruction is executed, the calculation result of the CAM slave axis position is modified, and the speed and acceleration constraints caused by phase adjustment can be set; the running speed and position of the slave axis remain continuous during adjustment; after the adjustment is completed, the phase difference is kept running continuously



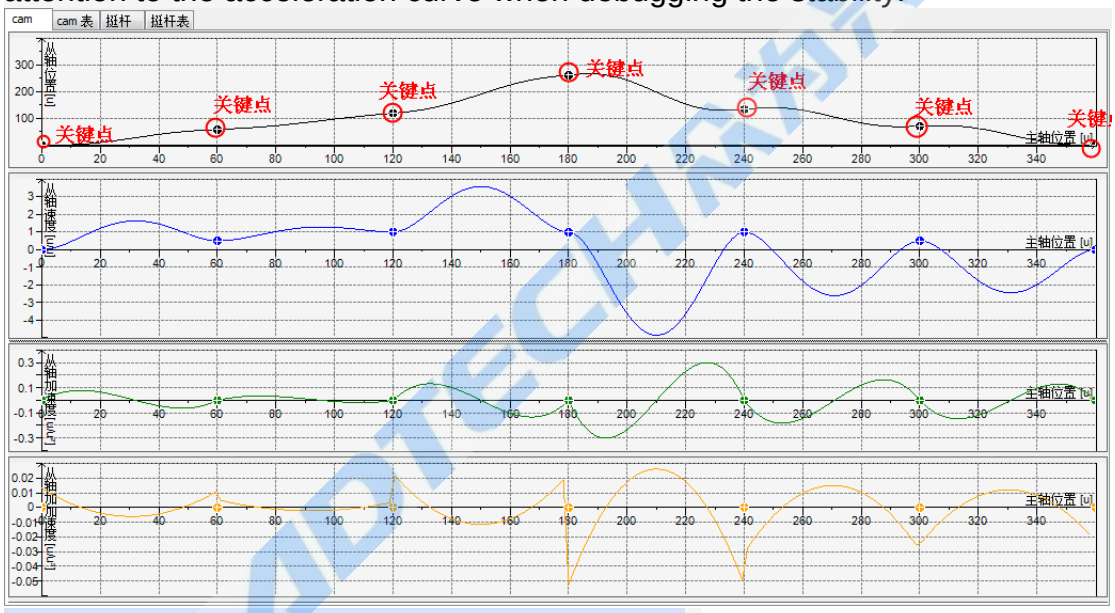
This command is used in some synchronous control with color code, if you need to adjust the position of the color code in the segment to be cut, you need to use this command.

5.5 Cam table design and its data structure

When writing a user program for cam operation, the cam table is one of the programming items, which determines the characteristics of the cam operation and can be entered in the form of graphics and tables.

5.5.1 Features of cam table

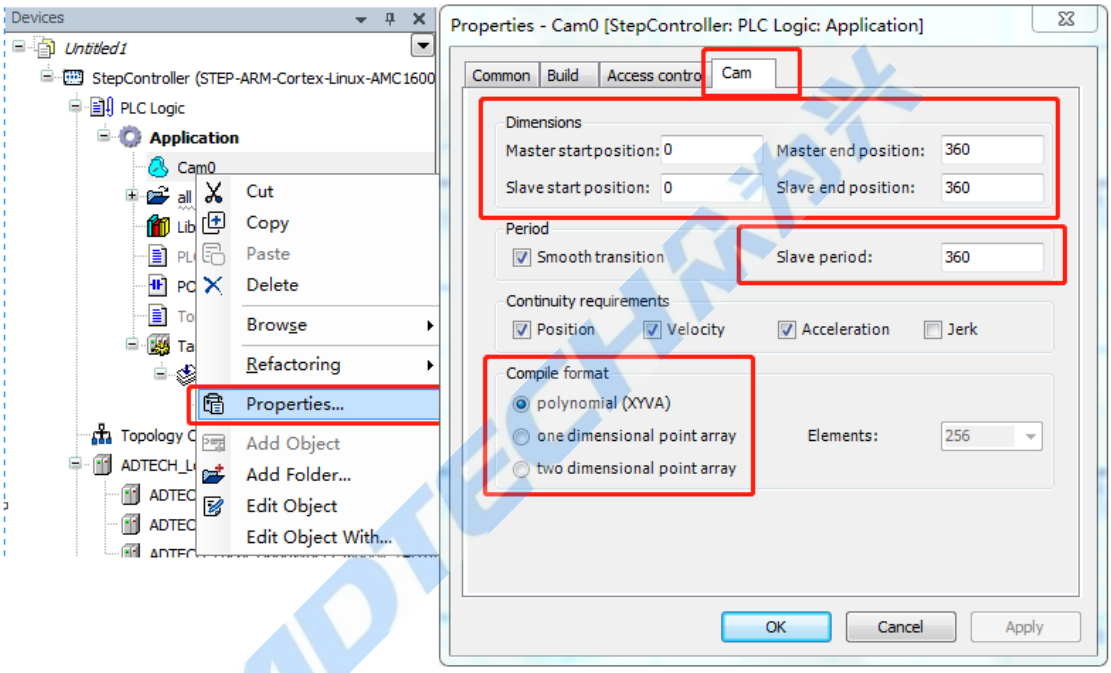
The figure below is a graphical CAM cam table. The abscissa is the position of the master axis, and the length of the axis is the stroke of the cam operation. There are four coordinate curves. The vertical axis is the position of the slave axis, the speed of the slave axis, the acceleration of the slave axis, and the speed of the slave axis. Jerk curve. When programming and debugging, often pay more attention to the position curve and speed curve, and also pay attention to the acceleration curve when debugging the stability.



The cam curve has the following characteristics:
In the coordinate of the master-slave position curve, the vertical axis is the range where the slave axis can travel; the vertical axis of the other three curves is the ratio of the speed of the slave axis to the master axis, and the ratio of the acceleration of the slave axis to the master axis;
The cam curve is a monotonic curve in the vertical direction, that is, each coordinate of the master axis can only correspond to the unique coordinate value of the slave axis;
when the cam is executed, the master axis coordinates move from small to large; The cam curve can have several key points. The line type between the two key points can be set as a straight line or a quintic curve. The system will optimize each quintic curve to minimize the sudden changes in speed and acceleration. ;
By default, the start and end coordinates of the horizontal axis (main axis) start at 0 and end at 360. The user can modify it according to the actual physical stroke.

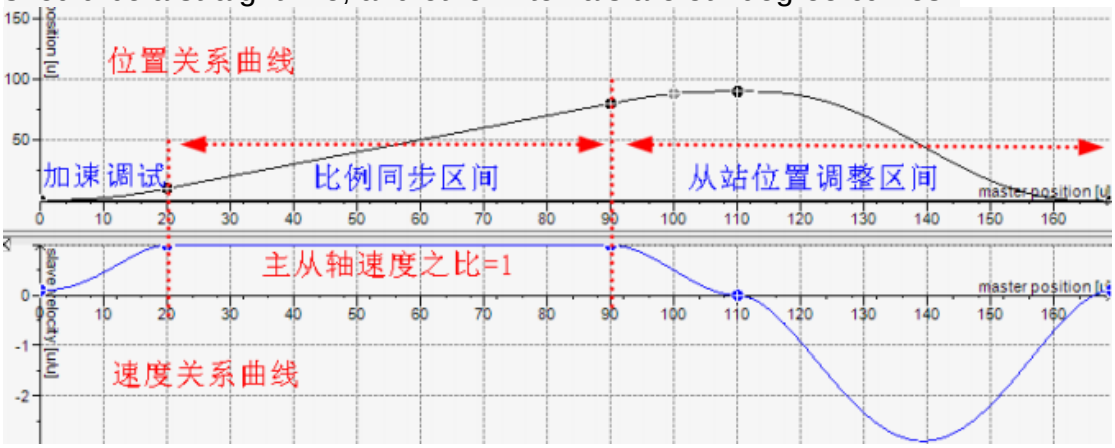
5.5.2 Input method of CAM table

1. When creating a new cam table, the system will automatically set the simplest cam curve, and the user can modify it on this basis to form the CAM curve table he needs;
2. The user can increase or decrease the number of key points of the cam curve and modify the coordinates of the key points;
3. The user can modify the line type between any two adjacent key points, or the quintic curve or straight line;
4. Between the key points of the cam curve, the system will connect according to the 5th curve by default, which can ensure the continuity of the speed during operation and reduce the mechanical impact;



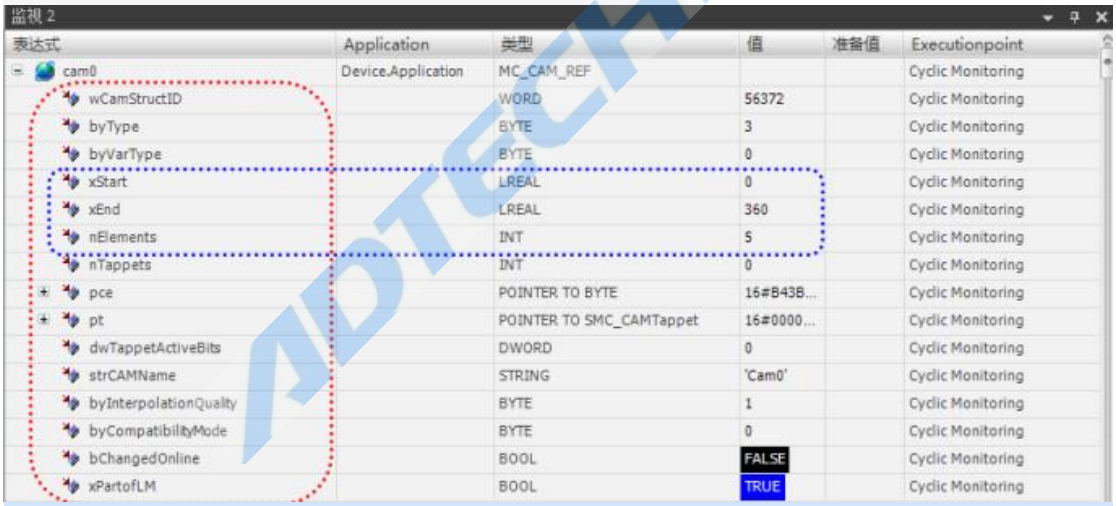
The key points in the cam curve are often related to the mechanical motion requirements of the controlled object, such as:
1. For chase shear applications, the coordinate range of the spindle is recommended to correspond to the physical stroke of the operating interval, which is convenient for analysis;
2. The starting and ending points of the reciprocating travel of the axis, the starting position point of the synchronous operation interval, the out-of-synchronous position point, etc., are important key points;

3. Proportional synchronization interval, the line segment of the cam curve should be a straight line, and other intervals are 5th degree curves



5.5.3 Internal data structure and array of CAM table

In CODESYS, for each CAM table, there is a data structure describing the CAM table and the characteristic data of the CAM table. The following figure shows the description data structure of the "CAM00" cam table. Please pay attention to the variable names of the structure:



CODESYS has an internal data structure to describe the characteristics of the CAM table: it is also possible if we manually write a CAM table, as follows:

```
VAR i: INT;
CAM: MC_CAM_REF :=(
        byType:=2,(*non-equidistant *)
        byVarType:=2,(*UINT*)
        nElements:=128,
        xStart:=0,
        xEnd:=360);
Table: SMC_CAMTable_UINT_128_2:=(
        fEditorMasterMin:=0,fEditorMasterMax:=360,
        fTableMasterMin:=0,fTableMasterMax:=6000,
```

```
        fEditorSlaveMin:=0,fEditorSlaveMax:=360,
        fTableSlaveMin:=0,fTableSlaveMax:=6000,);
END_VAR
```

Although we do not need to manually write the CAM table, we can modify the required CAM table feature data through the access operation of the data structure.
note:
When we declare the CAM0 cam table, the system automatically declares the CAM0 data structure of the global variable type by default, and at the same time declares the CAM0_A[i] array.
For example, in the user program, modify the number or coordinates of key points in the CAM0 cam table:

```
    CAM0.xEnd:=500; //Change the end point of the spindle to 500.//For
                      example, in the user program, modify the
                      coordinates of two key points;
    CAM0_A[3].dx:=30;
    CAM0_A[3].dy:=45;
    CAM0_A[3].dv:=1;
    CAM0_A[3].da:=0;

    CAM0_A[4].dx:=60;
    CAM0_A[4].dy:=75;
    CAM0_A[4].dv:=1;
    CAM0_A[4].da:=0;
```

Method to modify CAM cam table online;

The so-called "online modification of CAM curve" refers to the modification of the key point coordinates of the CAM curve according to the needs of control characteristics during the execution of the program written by the user.
The content of the modification is generally to modify the coordinates of the key points, but the number of key points and the distance range of the main axis can also be modified.
Reminder: Before entering the cam operation, modify the cam table. It is not advisable to modify it during operation to avoid unexpected motion results.
Need to modify CAM cam table applications:
1. Under normal circumstances, OEM customers use cam meters that have been successfully debugged and verified;
2. If there are several processing objects or modes, you can consider pre-setting multiple cam tables, and automatically switch according to the needs of users;
3. Some equipment requires a wider range of adaptation, such as packaging equipment. If the applicable packaging length is required to be in the range of 10cm~25cm, and the corresponding operating speed is automatically adapted to change, it may be necessary to modify the CAM cam table online

5.5.4 Reference and dynamic switching of CAM table

The CAM cam table is stored in an array inside the controller, which can be pointed to by a specific MC_CAM_REF variable type, such as the declaration:
Cam table p: MC_CAM_REF;
You can assign a value to the variable, or it can be thought of as pointing it to a specific cam table;
Cam table p:=Cam0; //Point to the required cam table

```
Cam table p:  MC_CAM_REF;  //Cam pointer;

TableID:  uint; //Cam table selection command can be set by HMI;

Case TableID of

0: Cam table p: = Cam table A;
1: Cam table p: = Cam table B;
2: Cam table p: = Cam table C;
End_case

MC_CamTableSelect_0(
        Master:=                //Cam spindle
        Slave:=                 //Cam Slave
        CamTable:=
        Execute:= ReSelect,
        Periodic:=TRUE,
        MasterAbsolute:=FALSE,
        SlaveAbsolute:=FALSE);
```

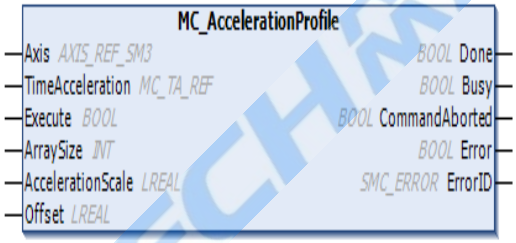
The above routine, using the MC_CAM_REF variable assignment operation, can realize the switching operation of multiple cam tables.

Chapter VI Detailed explanation of MC single-axis motion related instructions

MC_AccelerationProfile

This function block is a contour motion model of time period and acceleration/deceleration. The axis moves according to the specified "time-acceleration" contour data.

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Accele-rationProfile	Accelerati-on profile movement		MC_AccelerationProfile(Axis:= , TimeAcceleration:= , Execute:= , ArraySize:= , AccelerationScale:= , Offset:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis
Time Acceleration	Time and acceleratio n description	MC_TA_REF	—	—	Axis acceleration time and acceleration data description, acceleration data consists of multiple sets of data

- Input variables

Input variables	Name	Type of data	Effective range	Initial value	Description
Execute	Execution condition	BOOL	TRUE/FALSE	FALSE	Rising edge trigger: the function block is executed on the rising edge
ArraySize	Dynamic array	INT	-	0	Number of arrays used in running contour
Acceleration Scale	Acceleration ratio factor	LREAL	Positive number	1	The scale factor of acceleration or deceleration in MC_TA_REF
Offset	Offset	LREAL		0	Offset value of the whole acceleration and deceleration

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Done	Instruction execution completed	BOOL	TRUE/FALSE	FALSE	TRUE：Instruction execution completed
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE：The current instruction is executing
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE：Interrupted by another function block while the current instruction is being executed
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：Error occurred
ErrorID	Error code	SMC_ERROR	Refer to SMC_ERROR		Output error code when an error occurs

3）Function Description

✧ This function block is a contour motion model of time period and acceleration/deceleration. The operation mode is Discrete Motion, and it runs according to the data set by the user in the TimeAcceleration variable.

✧ The running state of this function block is in Standstill, the state when the instruction is running is Discrete Motion, and other states cannot run.

✧ The instruction is started at the rising edge of Execute. The repeated operation speed of this instruction in Discrete Motion is superimposed on the previous one, which may easily cause system failure.

✧ TimeAcceleration is MC_TA_REF data type;

MC_TA_REF is described as follows:

member	type	Initial value	content
Number_of_pairs	INT	0	Reserved: not used
IsAbsolute	BOOL	TRUE	Absolute movement （TRUE），Relative motion （FALSE）
MC_TA_Array	ARRAY [1..100] OF SMC_TA	-	Time-acceleration array

The specific description of SMC_TA is as follows:

member	type	Initial value	content
delta_time	TIME	TIME#0ms	Time difference between last time and this time
acceleration	LREAL	0	Acceleration

```
note:
The set acceleration is reflected in the change of speed. All
acceleration changes are changed in the way of S-curve, from
the final result to [the initial acceleration is A, the final
acceleration is B] (A+B)/2. At the final speed;
```

4）Timing diagram

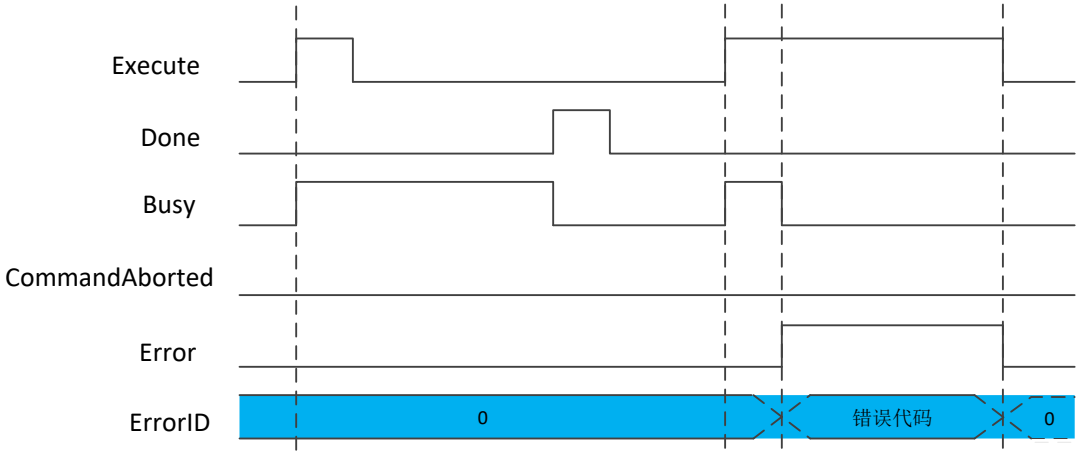
✧ The condition MC_TA_Array has been set in other ways;

✧ The command must be in the Standstill state to run;

✧ The Execute of the function block must have a rising edge condition;

✧ Done of the function block means the normal execution of the instruction is completed;

✧ Busy of the function block indicates that the current function block is being executed;



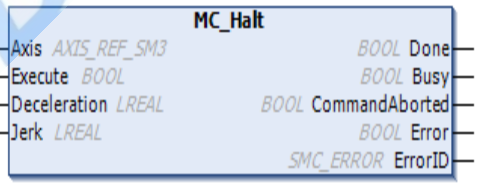
5）Error description

✧ The occurrence of the error means that the axis state is not the start command in Standstill or the parameter error in the command system. If an axis error occurs, the operation can only be started after the error is cleared.

MC_Halt

Decelerate the specified axis to a stop. Other motion function commands can also be executed during deceleration and after stopping.

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Halt	Axis normal stop command		MC_Halt(Axis:= , Execute:= , Deceleration:= , Jerk:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

2）Related variables

● Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

● Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Execute	Execution condition	BOOL	TRUE/FALSE	FALSE	Rising edge trigger: the function block is executed on the rising edge
Deceleration	decrease speed	LREAL	Positive number ,0	0	Function block deceleration [u/s²].
Jerk	Jerk	LREAL	Positive	0	Specify jerk (jerk) [u/s³]

			number ,0		

- Output variables

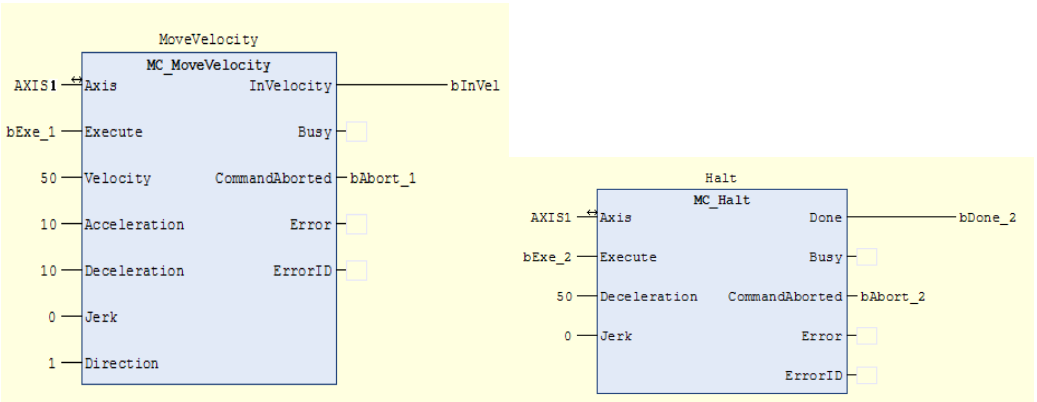
Output variables	Name	Type of data	Effective range	Initial value	Description
Done	Instruction execution completed	BOOL	TRUE/FALSE	FALSE	TRUE：The command is executed and the axis speed is 0
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE：The current instruction is executing
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE：The current instruction was interrupted by another executing function block.
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	Error code	SMC_ERROR	Refer to SMC_ERROR	0	Output error code when an error occurs

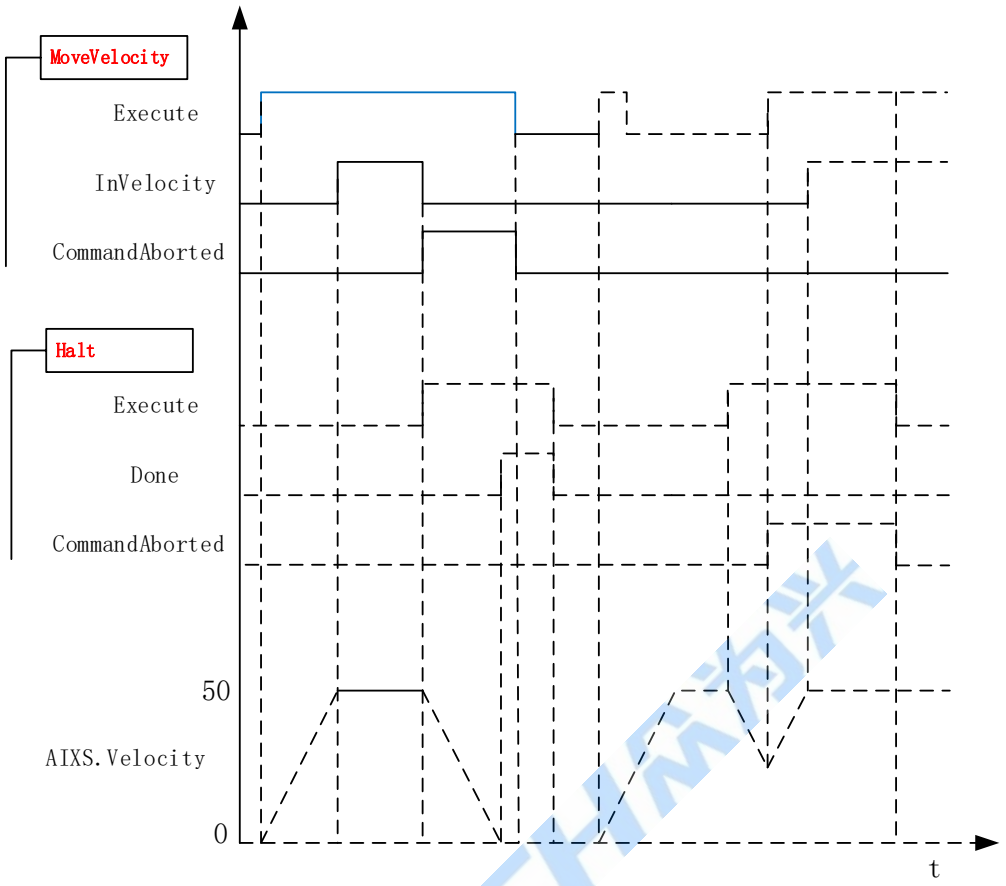
3）Function Description

- ✧ This function block stops the referenced axis in a control mode. When the instruction is executed normally, the axis enters the discrete motion state (Discrete Motion), until it reaches the speed 0, the axis becomes the stop state (Standstill).
- ✧ If this function is being executed and the operation of other function blocks is running, this function will be interrupted. As long as MC_Halt is active, a new action command can be issued to interrupt the execution of MC_Halt.
- ✧ The function block axis state can only run when it is in the running state (Motion) or Standstill, and cannot run in other states.
- ✧ The instruction is started at the rising edge of Execute, the state of the instruction is Discrete Motion during execution, and the state of the axis after the execution is completed is Standstill.

4）Timing diagram

Routine: The corresponding flag bit changes in different timing operations during the execution of MC_MoveVelocity and MC_Halt instructions; the processing of CommandAborted is described as the timing description in the following figure.





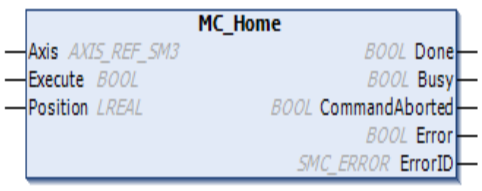
5) Error description

✧ The occurrence of an error means that the axis state is not in the running (Motion) or Standstill state, the start command or the parameter error in the command system. When an axis error occurs, operation can only be started after the error is cleared.

MC_Home

This function block is the zero return operation on the EtherCAT bus servo side.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Home	Axis home command		MC_Home(Axis:= , Execute:= , Position:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Execute	Execution condition	BOOL	TRUE/FALSE	FALSE	Rising edge: start to execute the function block
Position	Axis reach position	LREAL	data range	0	The absolute position set when the axis returns to the origin signal position[u]

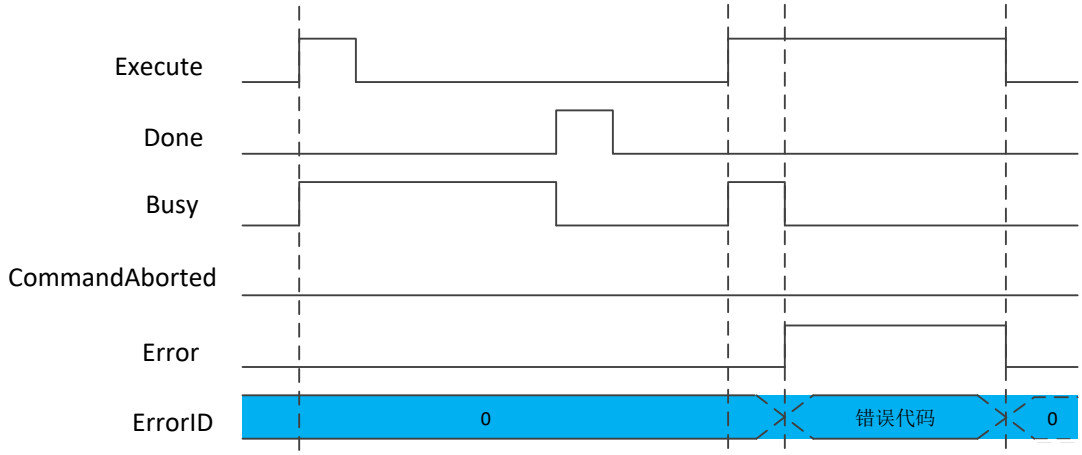
- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Done	Instruction execution completed	BOOL	TRUE/FALSE	FALSE	TRUE：The command is executed and the axis reaches the home position.
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE：The current instruction is executing
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE：The current instruction was interrupted by another executing function block .
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	Error code	SMC_ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs

3) Function description

- This function block is the zero return operation, and the Position data is the absolute position set after the axis returns to the zero point. When the origin reference signal is detected, the Position input is used to set the absolute position. The zero return ends when the function block outputs Done.
- The axis state machine is in Standstill before the function block is running, the axis state is homing when the command is running, and other states cannot run.
- The start instruction is the rising edge of Execute.

4) Timing diagram



MC_Jog

This function block is jog operation.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Jog	Axis jog command		MC_Jog(Axis:= , JogForward:= , JogBackward:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , Busy=> , CommandAborted= > , Error=> , ErrorId=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Jog Forward	Positive effect	BOOL	TRUE/FALSE	FALSE	TRUE: The axis moves forward at the specified speed, acceleration, and deceleration. FALSE:Stop moving forward. If JogBackward is TRUE at the same time, no action is performed.
Jog Backward	Negative effective	BOOL	TRUE/FALSE	FALSE	TRUE: The axis moves in the negative direction at the specified speed, acceleration, and deceleration. FALSE:Stop moving in the negative direction. If JogForward is TRUE at the

					same time, no action is performed.
Velocity	Target speed	LREAL	Positive number ,0	0	Specify target speed(u/s)
Acceleration	Acceleration	LREAL	Positive number ,0	0	Specify acceleration(u/s²)
Deceleration	decrease speed	LREAL	Positive number ,0	0	Designated deceleration (u/s²)
Jerk	Jerk	LREAL	Positive number ,0	0	Jerk(u/s³)

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is being executed.
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block.
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred.
ErrorID	Error code	SMC_ERROR	Refer to SMC_ERROR	0	Output error code when an error occurs

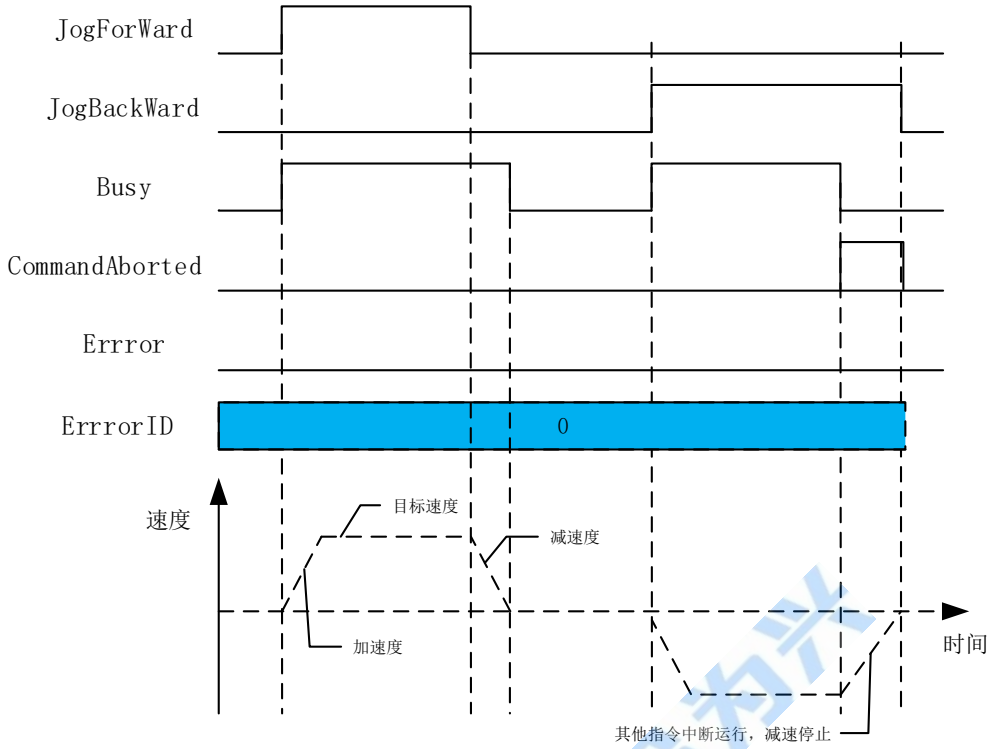
3) Function description

Execute jog operation according to the specified target velocity (Velocity).

- ✧ JogForward = TRUE ,Forward movement ；
  JogForward = FALSE ,Stop forward movement ；
- ✧ JogBackward = TRUE ,Negative movement ；
  JogBackward = TRUE , Stop negative movement ；
- ✧ JogForward = TRUE，At the same time JogBackward = TRUE, do not perform any movement ；

4) Timing diagram

- ✧ While starting JogForward (valid for forward operation) or JogBackward (valid for negative operation), Busy (in execution) becomes TRUE.
- ✧ Start decelerating and stopping the axis at the falling edge of JogForward (valid for forward operation) or JogBackward (valid for negative operation), and Busy (in execution) becomes FALSE.
- ✧ When this command is aborted by other commands, CommandAborted (execution interrupt) becomes TRUE and Busy (execution) becomes FALSE.



MC_MoveAbsolute

This function block moves the axis to a specified absolute position and uses the values of speed, deceleration, acceleration, and jump. If there are no other operations waiting to be executed, then execution will end at speed 0.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Move Absolute	Absolute position motion command	MC_MoveAbsolute Axis AXIS_REF_SM3 — Done BOOL Execute BOOL — Busy BOOL Position LREAL — CommandAborted BOOL Velocity LREAL — Error BOOL Acceleration LREAL — ErrorID SMC_ERROR Deceleration LREAL Jerk LREAL Direction MC_Direction	MC_MoveAbsolute(Axis:= , Execute:= , Position:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , Direction:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Execute	Execution condition	BOOL	TRUE/FALSE	FALSE	Rising edge trigger: the function block starts to execute on the rising edge
Position	Target location	LREAL	Positive, negative ,0	0	The target (absolute) position of the movement [u]
Velocity	Target speed	LREAL	{Positive,0}	0	Specify target speed [u/s]
Acceleration	Acceleration	LREAL	{Positive,0}	0	Specify acceleration[u/s²]
Deceleration	decrease speed	LREAL	{Positive,0}	0	Designated deceleration[u/s²]
Jerk	Jerk	LREAL	{Positive,0}	0	Jerk[u/s³]
Direction	Direction designation	MC_Direction	fastest, current, positive, shortest, negative	shortest	Specify the direction of movement, allowed values: negative（-1）：Move in the negative direction; shortest（0）：According to the trajectory of the shortest path. The choice of direction is based on the position when the command is called (only the rotation axis); Positive（1）：Move in the positive direction; Current（2）：Keep the current direction (only the rotation axis); Fastest（3）：The faster one is automatically selected (only the rotation axis). If there is only one mathematical solution to reach the target position, for example in a linear system, the value of the direction input is ignored.

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Done	Instruction execution completed	BOOL	TRUE/FALSE	FALSE	TRUE：The instruction execution is complete and the target position has been reached.
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is being executed .
CommandAborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block.
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	Error code	SMC_ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs

3) Function description

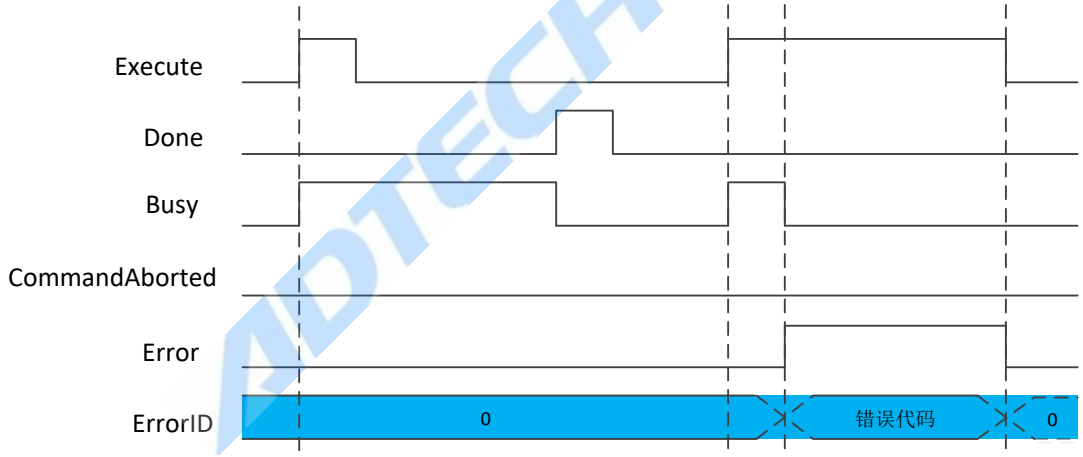
✧ This function block is an axis absolute positioning command, which makes the axis move to an absolute position (Position) and uses the values of

speed, deceleration, acceleration, and jump. If there are no other operations waiting to be executed, then the execution will end with velocity 0.

- ✧ Position: For modulo axis, the effective absolute position value is in the range of [0,360]. Therefore, 360 is not included. The range can be adjusted. The application program can move the setting position of MC_MoveAbsolute to the appropriate mode range. Die 360 can be used for relative positions.
- ✧ The axis state machine is in Standstill before the function block is running, and the axis state is Discrete Motion when the command is running.
- ✧ Start command is Execute rising edge start, this command can be valid on Discrete Motion repeatedly rising edge, and the latest Position position can be refreshed each time.
- ✧ Acceleration and Deceleration are 0, the instruction is running in an abnormal state, but the state of the axis is in Discrete Motion.

4) Timing diagram

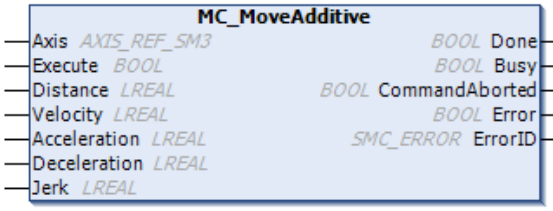
- ✧ The axis must be in the Standstill state command to run;
- ✧ The Execute of the function block must have a rising edge condition;
- ✧ Done of the function block means the normal execution of the instruction is completed;
- ✧ Busy of the function block indicates that the current function block is being executed;



MC_MoveAdditive

This function block makes the axis superimpose the data specified by Distance on the original command position, which is used for the online position superposition of the motion axis control process.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Move Additive	Axis superimposed motion command		MC_MoveAdditive(Axis:= , Execute:= , Distance:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Execute	Execution condition	BOOL	TRUE/FALSE	FALSE	Rising edge trigger: the function block starts to execute on the rising edge
Distance	superimposed distance	LREAL	Value range	0	Relative distance of superimposed movement
Velocity	Target speed	LREAL	Positive	0	Maximum speed when the axis is running [u/s]
Acceleration	Acceleration	LREAL	Positive	0	Specify acceleration [u/s²]
Deceleration	decrease speed	LREAL	Positive	0	Designated deceleration [u/s²]
Jerk	Jerk	LREAL	Positive	0	Jerk[u/s³]

- Output variables

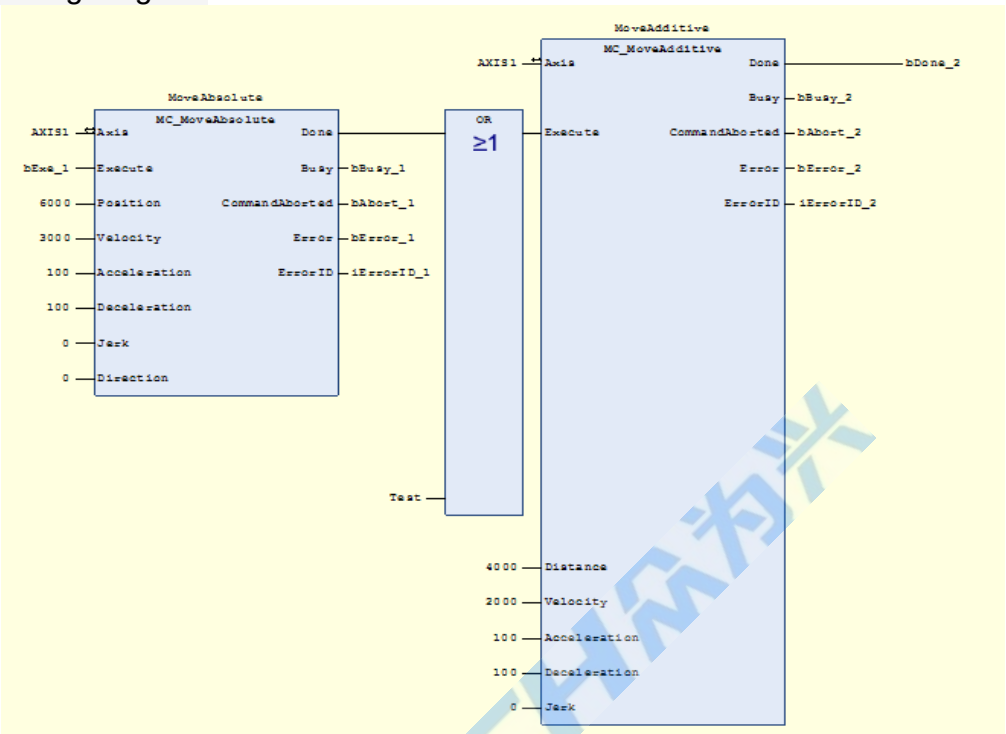
Output variables	Name	Type of data	Effective range	Initial value	Description
Done	Instruction execution completed	BOOL	TRUE/FALSE	FALSE	TRUE：The instruction is executed, and the distance has been superimposed.
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is being executed.
CommandAborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE:The current instruction was interrupted by another executing function block.
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	Error code	SMC_ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs

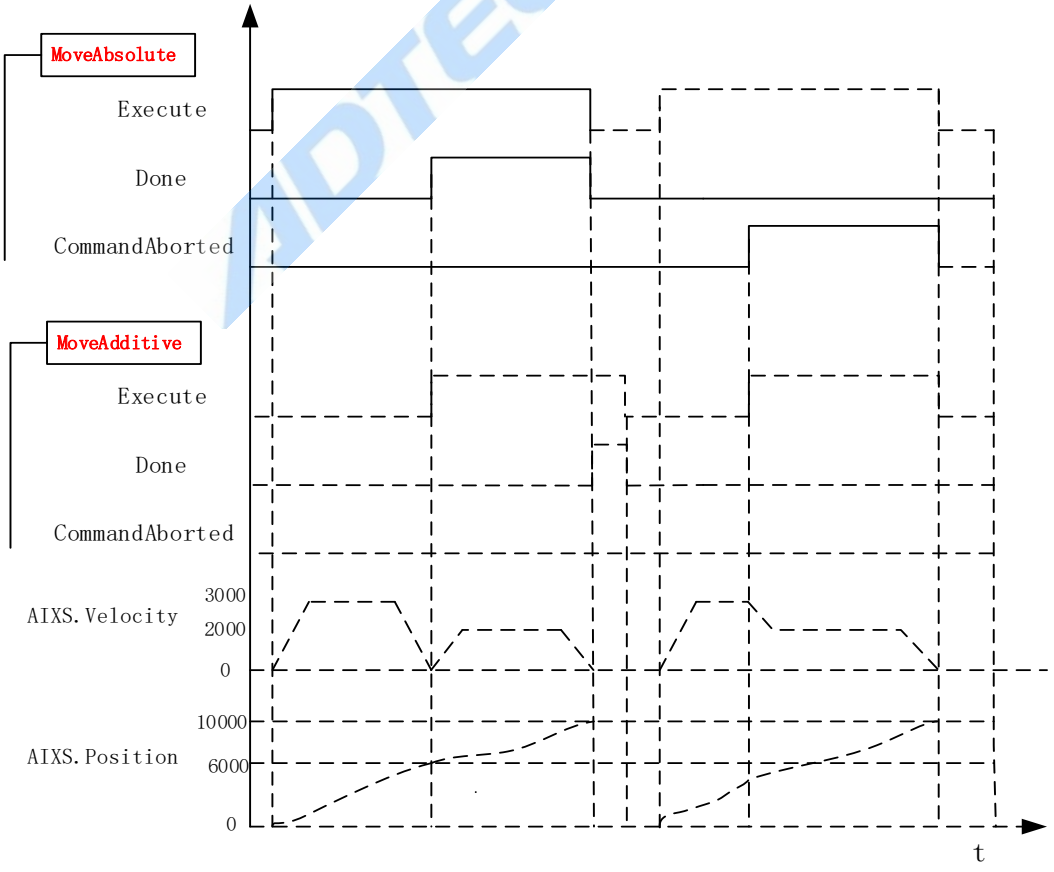
3) Function description

✧ The parameters Velocity, Acceleration, Deceleration are zero, and the instruction is running in an abnormal state.

- ✧ In the Discrete Motion state, the relevant execution process of MC_MoveAdditive can be added at any time.
- ✧ MC_MoveAdditive is equivalent to MC_MoveRelative in the Standstill state.

4) Timing diagram

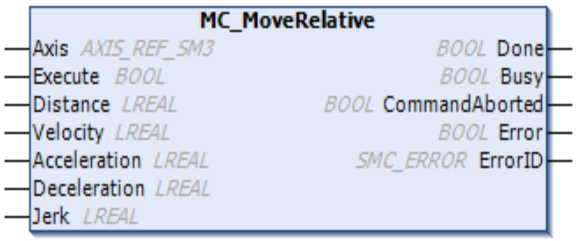




MC_MoveRelative

This function block makes the axis run in a relative position, and the position is specified by Distance.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_MoveRelative	Axis relative motion command		MC_MoveRelative(Axis:= , Execute:= , Distance:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Execute	Execution condition	BOOL	TRUE/FALSE	FALSE	Rising edge trigger: the function block starts to execute on the rising edge
Distance	Relative movement distance	LREAL	Value range	0	Relative distance of movement
Velocity	Target speed	LREAL	Positive	0	Maximum speed when the axis is running [u/s]
Acceleration	Acceleration	LREAL	Positive	0	Specify acceleration[u/s²]
Deceleration	decrease speed	LREAL	Positive	0	Designated deceleration [u/s²]
Jerk	Jerk	LREAL	Positive	0	Jerk(u / s³)

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Done	Instruction execution completed	BOOL	TRUE/FALSE	FALSE	TRUE：The instruction execution is complete, and the relative movement distance has been achieved .
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is being executed.
CommandAborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block.
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred

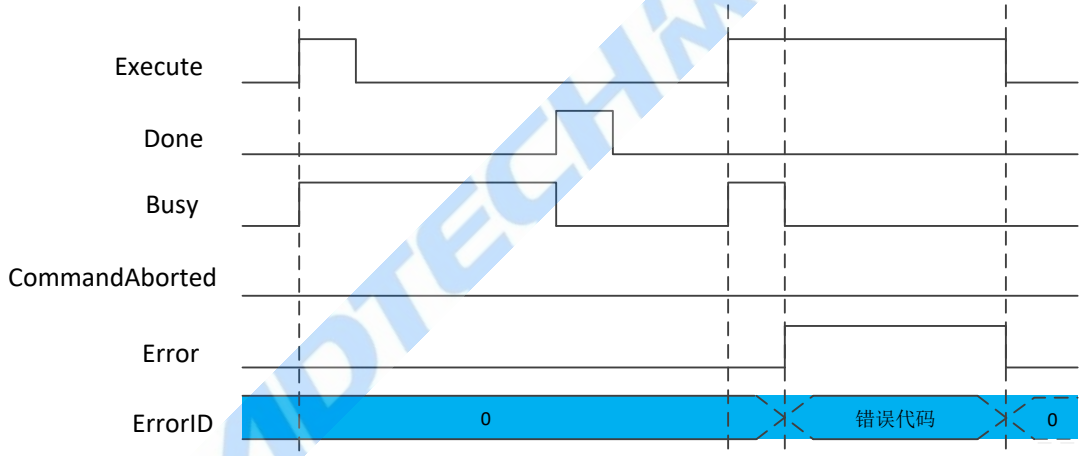
ErrorID	Error code	SMC_ERROR	参阅 SMC_ERROR	0	Output error code when an error occurs

3) Function description

- ✧ This function block commands the controlled movement of the specified distance relative to the set position during execution. The movement ends at a speed of 0.
- ✧ The parameters Velocity, Acceleration, Deceleration are zero, the command operation is abnormal, but the state of the axis is in Discrete Motion.
- ✧ The running status of this function block is in Standstill, and the status when the command is running is Discrete Motion. Please pay attention to the running status of this axis during command execution, and avoid interrupting other commands of this axis or being interrupted by other commands.

4) Timing diagram

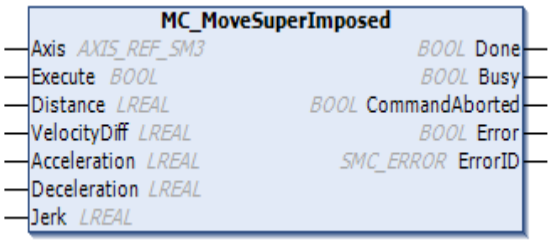
- ✧ The Execute of the function block must have a rising edge condition;
- ✧ Done of the function block means the normal execution of the instruction is completed;
- ✧ Busy of the function block indicates that the current function block is being executed;



MC_MoveSuperImposed

On the basis of the original command speed and position of the axis, the speed and position data are superimposed on the running command, and there is no change in the execution time model of the entire original command.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_MoveSuperImposed	Axis superimposed relative motion command		MC_MoveSuperImposed(Axis:= , Execute:= , Distance:= , VelocityDiff:= , Acceleration:= , Deceleration:= , Jerk:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Execute	Execution condition	BOOL	TRUE/FALSE	FALSE	Rising edge trigger: the function block starts to execute on the rising edge
Distance	superimposed distance	LREAL	Data range	0	Additional stacking distance[u]
VelocityDiff	Target speed	LREAL	positive	0	Maximum speed when the axis is running [u/s]
Acceleration	Acceleration	LREAL	positive	0	Specify acceleration [u/s²]
Deceleration	decrease speed	LREAL	positive	0	Designated deceleration [u/s²]
Jerk	Jerk	LREAL	positive	0	Jerk[u/s³]

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Done	Instruction execution completed	BOOL	TRUE/FALSE	FALSE	TRUE：The execution of the instruction is complete, and the relative distance of the stack has been achieved.
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is being executed.
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block.
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	Error code	SMC_ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs

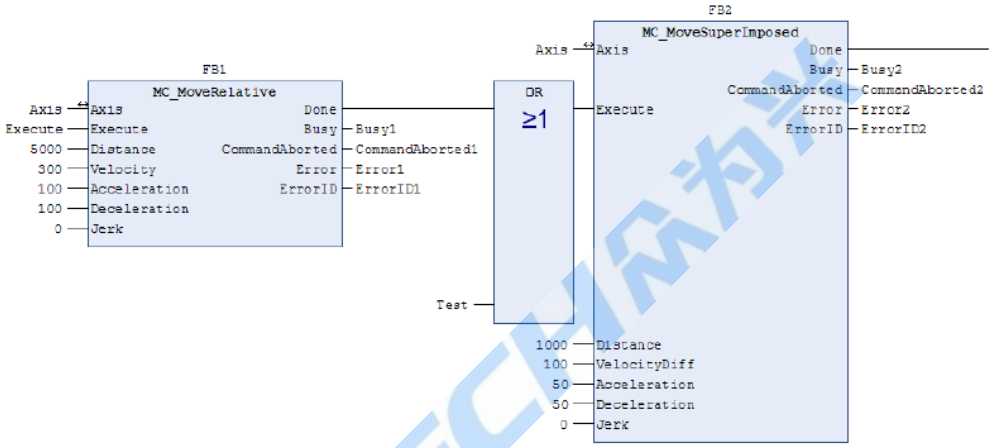
3) Function description

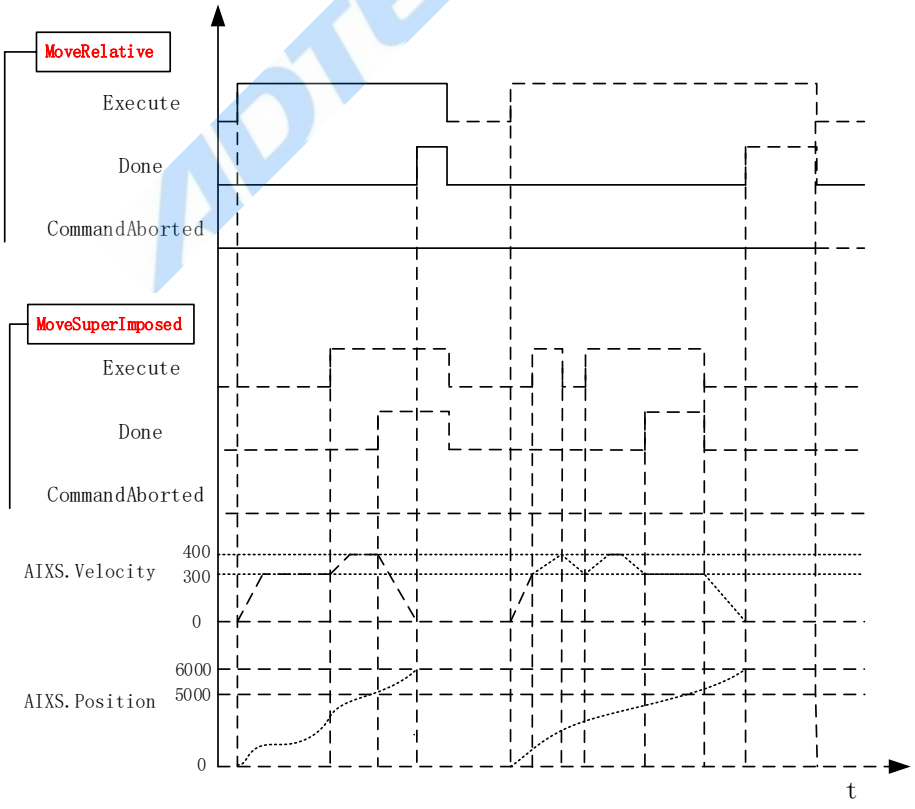
- ✧ This function block is the superimposed position and speed command, VelocityDiff and Distance are the speed and position superimposed on other commands, respectively.
- ✧ MC_MoveSuperImposed can be superimposed on any other instruction in motion mode. MC_MoveSuperImposed can also be aborted by MC_MoveSuperImposed.
- ✧ In the Standstill state, the action of the function block MC_MoveSuperImposed is similar to MC_MoveRelative.

- ✧ The function block causes additional movement on the basis of the continuous (original) movement of the axis. The original movement is not stopped, but the movement of the given distance is superimposed.
- ✧ Through this instruction, we can solve the belt and gear gap error compensation during our time running, and can ensure the consistency of the movement; when the command is running, the parameter acceleration (Acceleration) and deceleration (Deceleration) are zero, and the command is running in an abnormal state.

4) Timing diagram

- ✧ The Execute of the function block must have a rising edge condition;
- ✧ Done of the function block means the normal execution of the instruction is completed;
- ✧ Busy of the function block indicates that the current function block is being executed.

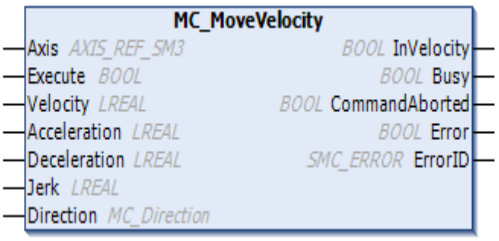




MC_MoveVelocity

In the analog speed control in the axis position control mode, when the axis is enabled and the command is valid, the assignment of Velocity can control the speed of the axis.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Move Velocity	Speed control command		MC_MoveVelocity(Axis:= , Execute:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , Direction:= , InVelocity=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>););

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Rising edge trigger: the function block starts to execute on the rising edge
Velocity	Target speed	LREAL	positive	0	Target speed [u/s]
Acceleration	Acceleration	LREAL	positive	0	Specify acceleration [u/s²]
Deceleration	decrease speed	LREAL	positive	0	Designated deceleration [u/s²]
Jerk	Jerk	LREAL	positive	0	Jerk[u/s³]
Direction	Command polarity	MC_Direction	Fastest, current, positive, shortest, negative	current	Specify the direction of movement, allowed values: negative（-1）：Move in the negative direction; shortest（0）：According to the trajectory of the shortest path. The choice of direction is based on the position when the command is called (only the rotation axis); Positive（1）：Move in the positive direction; Current（2）：Keep the current direction (only the rotation axis); Fastest（3）：The faster one is automatically

					selected (only the rotation axis). If there is only one mathematical solution to reach the target position, for example in a linear system, the value of the direction input is ignored.

- Output variables

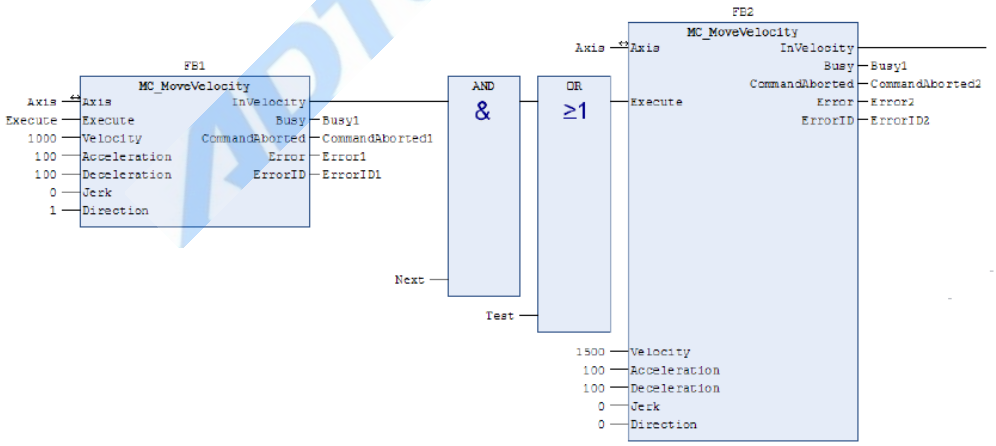
Output variables	Name	Type of data	Effective range	Initial value	Description
InVelocity	Reach set speed	BOOL	TRUE/FALSE	FALSE	TRUE: The instruction execution is complete, and the set running speed is reached for the first time.
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is being executed
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is executed by another function block
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE: An error occurred
ErrorID	Error code	SMC_ERROR	参阅 SMC_ERROR	0	Output fault code when an error occurs

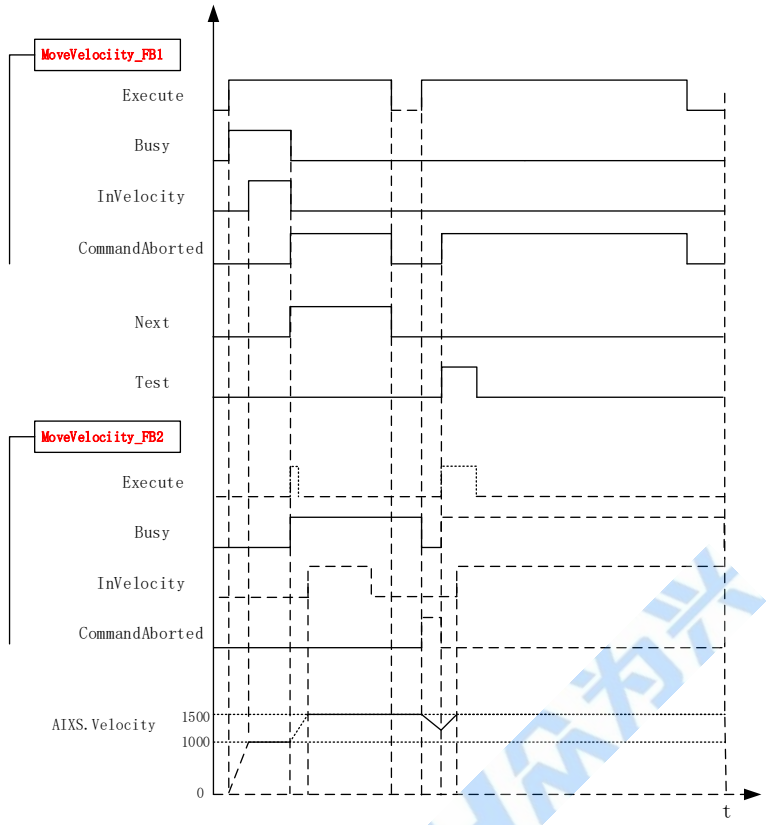
3) Function description

✧ Change Velocity parameters to control the analog speed of the drive.

4) Timing diagram

for example:

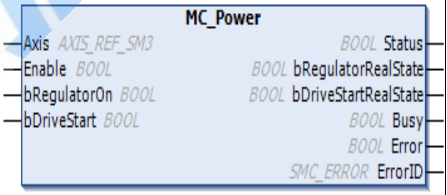




MC_Power

This function block is an axis enable command.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_ Power	Axis enable command		MC_Power(Axis:= , Enable:= , bRegulatorOn:= , bDriveStart:= , Status=> , bRegulatorRealState=> , bDriveStartRealState=> , Busy=> , Error=> , ErrorID=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Enable	Enable effective	BOOL	TRUE/FALSE	FALSE	TRUE：Start execution function block
bRegulatorOn	Enable state	BOOL	TRUE/FALSE	FALSE	TRUE：Start enable state

bDriveStart	Allow drive	BOOL	TRUE/FALSE	FALSE	TRUE：Prohibit fast stop mechanism FALSE：Allow quick stop

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Status	Operational state	BOOL	TRUE/FALSE	FALSE	TRUE：轴已经准备好运动
bRegulator RealState	Axis enable signal status	BOOL	TRUE/FALSE	FALSE	TRUE:主电源接通，轴使能处于有效状态
bDriveStart RealState	Allow drive status	BOOL	TRUE/FALSE	FALSE	TRUE: 轴没有被快速停止机制中断
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE：功能块正在执行
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：发生错误
ErrorID	Error code	SMC_ERROR	参阅 SMC_ERROR	0	发生错误时输出故障代码

3) Function description

✧ Only when the input Enable is TRUE, other inputs will be processed by the function block.

✧ If the function block MC_Power has been called and bRegulatorOn=FALSE, the function block will set the axis state (nAxisState) of the relevant axis to the Power_off state, indicating that the drive is not ready for movement.

✧ If the function block MC_Power has been called and bRegulatorOn=TRUE, if there is no error in the axis at this time, the function block will set the related axis state (nAxisState) to the Standstill state; if an error occurs, the corresponding error state will be output.

✧ If Enable, bRegulatorOn and bDriveStart are all TRUE, but the output Status is still FALSE after a certain period of time, then the output Error will be set. This situation may occur when certain hardware problems occur in the enabled state.

✧ If the enable signal is lost (usually in operation mode), the nAxisState of the relevant axis will be set to the ErrorStop state.

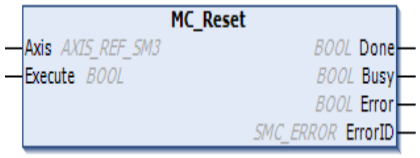
5) Error description

✧ Do not write a program for starting the MC_Power instruction of other instances in the axis that is executing the MC_Power instruction. In principle, only one MC_Power instruction can be set for one axis. If in the axis where the MC_Power instruction is being executed, the MC_Power instruction of another instance is started to operate this axis, the MC_Power instruction that is executed afterwards will be executed first.

MC_Reset

Reset the internal error of the specified axis and make it change from ErrorStop state to Standstill state.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_ Reset	Axis error state reset command		MC_Reset(Axis:= , Execute:= , Done=> , Busy=> , Error=> , ErrorID=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Rising edge trigger: the function block starts to execute on the rising edge

- Output variables

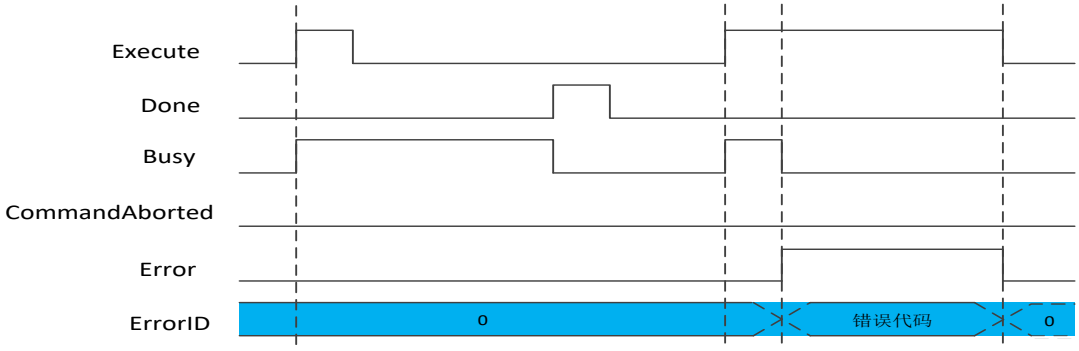
Output variables	Name	Type of data	Effective range	Initial value	Description
Done	Instruction execution completed	BOOL	TRUE/FALSE	FALSE	TRUE：Instruction execution completed
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE：Function block is executing
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	ErrorID	SMC_ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs

3) Function description

✧ When this function block is in normal axis communication, this function block resets all internal errors related to the axis, changes the axis state from errorstop to Standstill, and changes the abnormal state of the axis to a normal and operable state;

✧ When the axis errorstop cannot be reset, and Axis.bCommunication is in the FALSE state, the communication between the master and slave axes must be re-established. The Busy flag in the instruction is very short when it is connected. Please pay attention when using it;

4) Timing diagram



MC_SetPosition

Set the position data in the command as the position data of the current axis, and the operation of setting the position data will not produce any displacement movement, which is used to generate the displacement of the coordinate system.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_ SetPosition	Axis position setting command		MC_SetPosition(Axis:= , Execute:= , Position:= , Mode:= , Done=> , Busy=> , Error=> , ErrorID=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Rising edge trigger: the function block starts to execute on the rising edge
Position	Axis position data	LREAL	Data range	0	Location data
Mode	mode	BOOL	TRUE/FALSE	FALSE	Location mode: TRUE：relative position（RELATIVE）； FALSE：Absolute position（ABSOLUTE）；

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Done	Setting operation is successful	BOOL	TRUE/FALSE	FALSE	TRUE：Axis position setting operation succeeded
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE：Function block is executing
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	ErrorID	SMC_ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs

3) Function description

✧ This function block is designed to change the coordinate system of an axis. Therefore, the function block will operate the set position and actual position of the axis with the same value without causing any movement. This recalibration can also be used in the case of sports situations. Without changing the command position, the function block can be called during the movement, and then the function block will be located in the shifted

coordinate system.

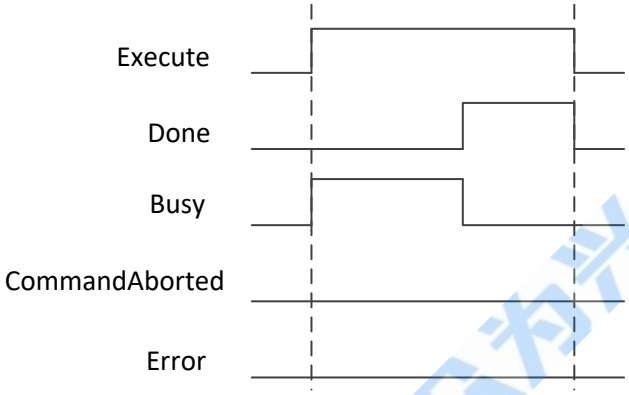
✧ The instruction is triggered by the rising edge of Execute; the instruction can be used repeatedly without affecting each other.

4) Timing diagram

✧ The Execute of the function block must have a rising edge condition;

✧ Done of the function block indicates the completion of the normal execution of the instruction;

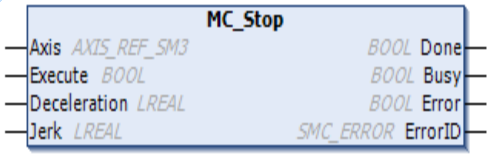
✧ Busy of the function block indicates that the current function block is being executed;



MC_Stop

Decelerate the specified axis to a stop. This function block is to stop the movement of an axis under normal operation. When the axis is in the stopping state, any command to this axis is invalid.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Stop	Emergency stop command		MC_Stop(Axis:= , Execute:= , Deceleration:= , Jerk:= , Done=> , Busy=> , Error=> , ErrorID=>);

2) Related variables

● Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

● Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	TRUE：Start to execute the function block
Deceleration	decrease	LREAL	positive	0	Designated deceleration

	speed				[u/s²]
Jerk	Jerk	LREAL	positive	0	Jerk[u/s³]

- Output variables

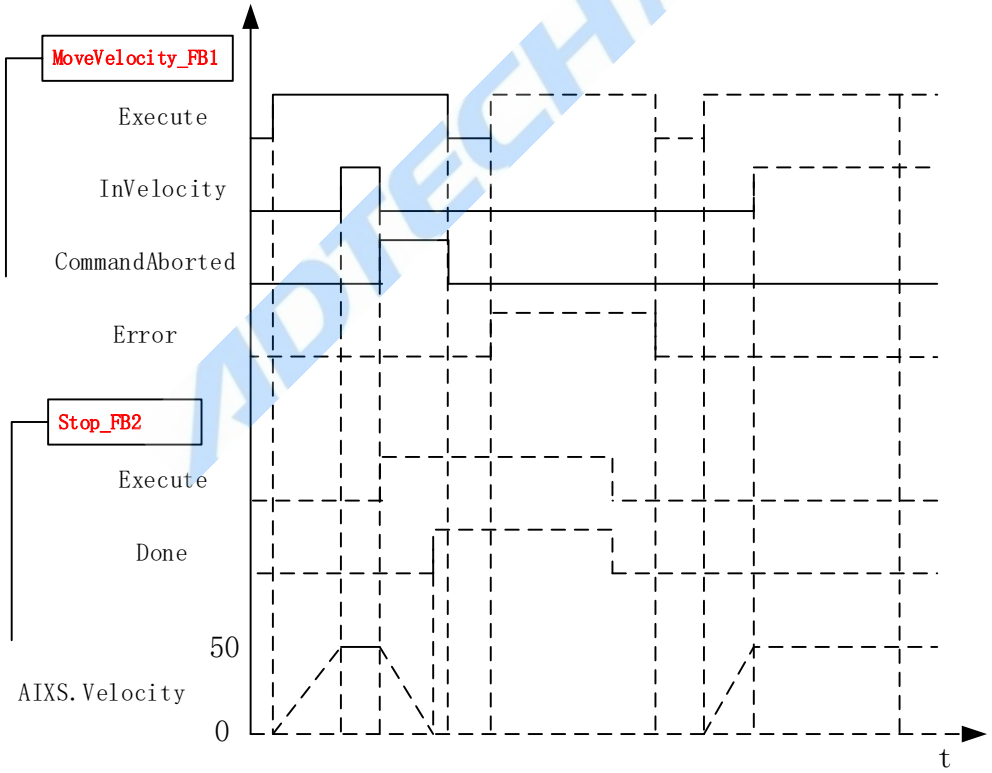
Output variables	Name	Type of data	Effective range	Initial value	Description
Done	Execution complete	BOOL	TRUE/FALSE	FALSE	TRUE：The execution of the function block is completed, and the axis speed is 0
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE：Function block is executing
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	ErrorID	SMC_ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs

3) Function description

✧ MC_Stop puts the axis in the stopping state, so the action of the currently running function block instance will be stopped. As long as the axis is in the stopping state, no other instance can perform actions on it. If the axis reaches a speed value of 0, the completion output Done is set to TRUE. As long as the Execute input is TRUE, the axis remains in the stopping state. If Execute is FALSE and the Done output is TRUE, the axis enters the Standstill state.

✧ The instruction is triggered by the rising edge of Execute.

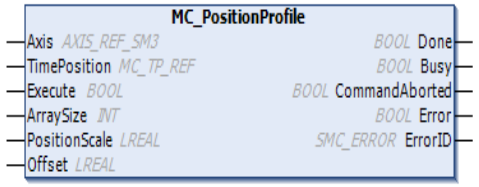
4) Timing diagram



MC_PositionProfile

This function block is a contour motion model of time period and position, and the axis moves according to the specified "time-position" contour data.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Position Profile	Position contour command		MC_PositionProfile(Axis:= , TimePosition:= , Execute:= , ArraySize:= , PositionScale:= , Offset:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis
TimePosition	Axis position running time and position description	MC_TP_REF			Description of the time and position data of the axis position operation, the data consists of multiple sets of data

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Rising edge: start to execute the function block
ArraySize	Dynamic array	INT	Data range	0	Number of arrays used in running contour
PositionScale	Comprehensive factor	LREAL	positive	1	Scale factor of position in MC_TP_REF
Offset	Offset	LREAL		0	The overall offset value of the location

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Done	Execution complete	BOOL	TRUE/FALSE	FALSE	TRUE：The configuration file has been completed
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is being executed
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	ErrorID	SMC_ERRO	Refer to	0	Output fault code

		R	SMC_ERROR		when an error occurs

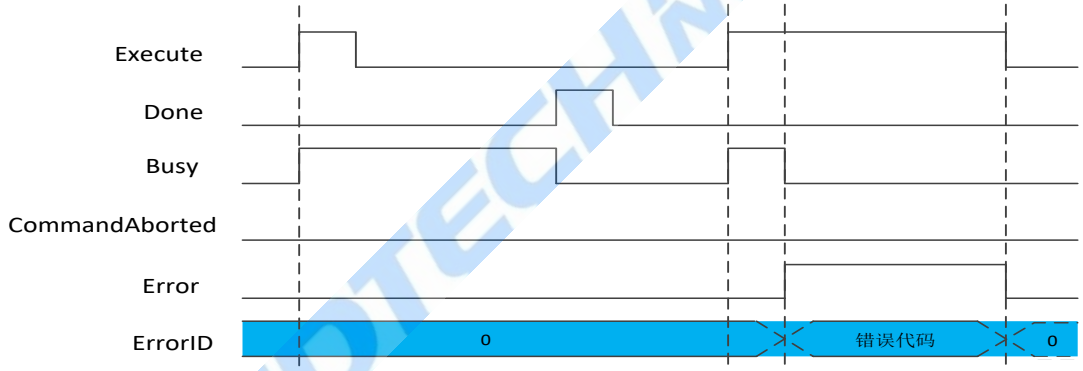
3) Function description

- ✧ This function block is the contour motion model of time period and position, the running mode is Discrete Motion, and it runs according to the data set by the user in the TimePositon variable.
- ✧ The running state of this function block is in Standstill, and the state when the instruction is running is Discrete Motion.
- ✧ The start instruction is the rising edge of Execute, and this instruction runs repeatedly in Discrete Motion.
- ✧ MC_MoveSuperImposed function block is not supported on MC_PositionProfile.

Note: When the speed corresponding to the set position data changes, it will be adjusted according to the S curve.

4) Timing diagram

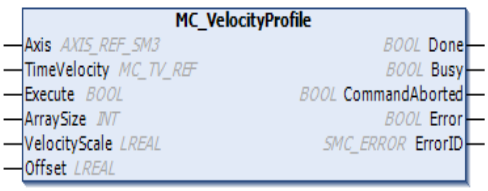
- ✧ The axis must be in the Standstill state before the command can run;
- ✧ The Execute of the function block must have a rising edge condition;
- ✧ Done of the function block indicates the completion of the normal execution of the instruction;
- ✧ Busy of the function block indicates that the current function block is being executed;



MC_VelocityProfile

This function block is a contour motion model of time period and speed, and the axis moves according to the specified "time-speed" contour data.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Velocity Profile	Speed profile command		MC_VelocityProfile(Axis:= , TimeVelocity:= , Execute:= , ArraySize:= , VelocityScale:= , Offset:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis
TimeVelocity	Axis speed running actual And speed description	MC_TV_REF			The axis speed running time and speed description data are composed of multiple sets of data.

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Rising edge: start to execute the function block
ArraySize	Dynamic array	INT	Data range	0	Number of arrays used in running contour
VelocityScale	Comprehens ive factor	LREAL	positive	1	Speed scale factor
Offset	Offset	LREAL		0	The overall offset value of the speed value

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Done	Execution complete	BOOL	TRUE/FALSE	FALSE	TRUE：Function block execution completed
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE：Function block is executing
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	ErrorID	SMC_ERRO R	Refer to SMC_ERROR	0	Output fault code when an error occurs

3) Function description

- ✧ This function block is a contour motion model of time period and speed. The running mode is Continuous Motion. It runs according to the data set by the user in the TimeVelocity variable. After the instruction is executed, the running mode becomes Discrete Motion.
- ✧ The start instruction is the rising edge of Execute.
- ✧ MC_MoveSuperImposed function block is not supported on MC_ VelocityProfile.
- ✧ TimeVelocity is MC_TV_REF data type.

The specific description of MC_TV_REF is as follows:

member	type	Initial value	content
Number_of_pairs	INT	0	Reserved: unused
IsAbsolute	BOOL	TRUE	Absolute movement（TRUE）, Relative motion（FALSE）
MC_TV_Array	ARRAY [1..100] OF SMC_TV	-	Time-speed array

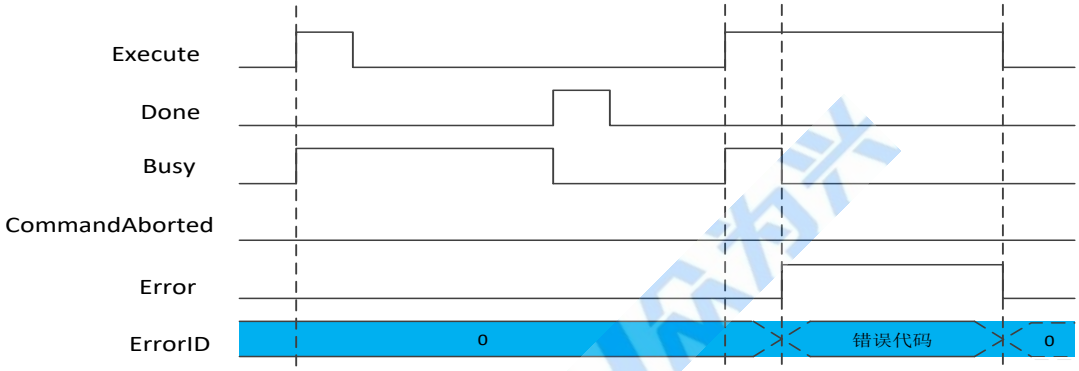
The specific description of SMC_TV is as follows:

member	type	Initial value	content
delta_time	TIME	TIME#0ms	The time difference between this time and the last time
Velocity	LREAL	0	speed

Note: The entire speed process is S-curve acceleration and deceleration, and each contour speed is a superimposed calculation method; when the command is repeated, the speed is also superimposed to avoid the occurrence of speed exceeding the limit when the command is used.

4) Timing diagram

- ✧ The condition MC_TV_Array has been set in other ways to run the position contour curve command.
- ✧ The Execute of the function block must have a rising edge condition;
- ✧ Done of the function block indicates the completion of the normal execution of the instruction;
- ✧ Busy of the function block indicates that the current function block is being executed;

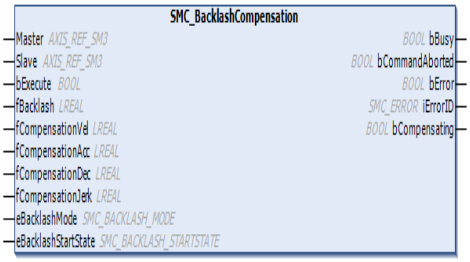


SMC_BacklashCompensation

This function block can be used to compensate for the gap generated in the belt drive or gear box. For example, the virtual axis in the belt transmission is the main axis, and the slave axis is the virtual axis synchronous mirroring. Due to external reasons, there is a gap between the position of the slave axis and the main axis. Command to compensate for this gap.

This command is similar to the phase shift command (MC_Phasing), and its phase depends on the direction of the spindle movement.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_Backlash Compensation	gap Compensation instruction		SMC_BacklashCompensation(Master:= , Slave:= , bExecute:= , fBacklash:= , fCompensationVel:= , fCompensationAcc:= , fCompensationDec:= , fCompensationJerk:= , eBacklashMode:= , eBacklashStartState:= , bBusy=> , bCommandAborted=> , bError=> , iErrorID=> , bCompensating=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Master	Spindle	AXIS_REF_SM3	—	—	Map to spindle
Slave	Slave axis	AXIS_REF_SM3	—	—	Map to slave axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
bExecute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Rising edge: start to execute the function block, set the offset
fBacklash	Compensation gap	LREAL		0	Distance/gap to be compensated
fCompensationVel	Compensation speed	LREAL			Additional speed, must be compensated with additional speed
fCompensationAcc	Compensated acceleration	LREAL			Acceleration during compensation
fCompensationDec	Compensation deceleration	LREAL			Deceleration during compensation
fCompensationJerk	Jerk	LREAL			Jump during compensation
eBacklashMode	Compensation mode	SMC_BACKLASH_MODE		SMC_BL_AUTO	间隙补偿模式 SMC_BL_AUTO: The direction of spindle movement determines the compensation direction. SMC_BL_POSITIVE: Positive compensation, has nothing to do with the movement of the spindle. SMC_BL_NEGATIVE: Compensation in the negative direction, which has nothing to do with the movement of the spindle. SMC_BL_OFF: No compensation
eBacklashStartState		SMC_BACKLASH_STARTSTATE		SMC_BL_START_NONE	Describe the working status of the axis when the instruction is working: SMC_BL_START_NEGATIVE: When pulling the movement from the negative direction of the axis, that is, the negative movement does not need compensation; once it moves in the positive direction, it is compensated with double the flashlash gap. SMC_BL_START_POSITIVE: Traction movement from the positive direction of the axis, that is, movement in the positive direction does not require compensation; once it moves in the opposite

					direction, it will be compensated with double the flashlash gap. SMC_BL_START_NONE movement in the positive or negative direction will cause the compensation of the flashlash value

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
bBusy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE：Function block is executing
bCommand Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block
bError	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
iErrorID	ErrorID	SMC_ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs
bCompensating	Compensating	BOOL	TRUE/FALSE	FALSE	TRUE: Compensation occurs

3) Function description

✧ This function block can be used to compensate the gap generated in the belt drive or gear box.

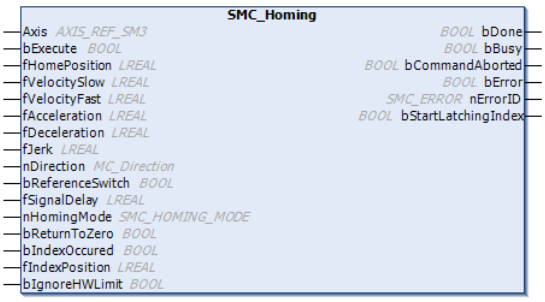
✧ The rising edge of bExecute, bBusy outputs TRUE if there is no error, and bCompensating outputs TRUE. After the compensation is completed, the bCompensating output is FALSE;

Note: When the module starts to work, the master axis and the slave axis are at the same position, otherwise the slave axis will be positioned on the master axis.

SMC_Homing

The axis homing command is different from MC_Home. MC_Home is the controller sending a homing command to the driver, and the driver completes the homing operation; SMC_Homing is the controller controlling the execution of the homing operation.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_ Homing	Axis home command		SMC_Homing(Axis:= , bExecute:= , fHomePosition:= , fVelocitySlow:= , fVelocityFast:= , fAcceleration:= , fDeceleration:= , fJerk:= , nDirection:= , bReferenceSwitch:= , fSignalDelay:= , nHomingMode:= , bReturnToZero:= , bIndexOccured:= , fIndexPosition:= , bIgnoreHWLimit:= , bDone=> ,

			bBusy=> , bCommandAborted=> , bError=> , nErrorID=> , bStartLatchingIndex=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
bExecute	Enable effective	BOOL	TRUE/ FALSE	FALSE	TRUE：Start to execute the function block
fHomePosition	Origin setting position	LREAL		0	The origin is set to the position after returning to zero. [u]
fVelocitySlow	Slow	LREAL		0	Slow speed, used to find the set speed used after the reference switch [u/s]
fVelocityFast	fast	LREAL		0	Speed; the set speed used before the reference switch is found [u/s]
fAcceleration	Acceleration	LREAL			Acceleration setting value
fDeceleration	decrease speed	LREAL			Deceleration setting value
fJerk	Jerk	LREAL			Jerk[u/s^3]
nDirection	Home direction	MC_Direction		negative	Return to the starting direction, move quickly to find the direction of the reference switch (positive or negative)
bReferenceSwitch	Reference switch	BOOL	TRUE/ FALSE	FALSE	Connect reference switch TRUE: The reference switch is turned on (the drive is not in the switch) FALSE: The reference switch is closed (the drive is in the switch)
fSignalDelay	delay	LREAL		0	The transmission time of the reference switch is used to compensate for the dead time. Unit: Second (S)
nHomingMode	Return to zero mode	SMC_HOMING_MODE		FAST_BSLOW_S_STOP	Return to zero mode
bReturnToZero	Return to zero	BOOL	TRUE/ FALSE	FALSE	TRUE: The axis moves to position zero after the home return is completed (Note: If fHomePosition=10, the axis position is set to 10 after the home return is completed, and the axis will continue to move to position 0.

					bReturnToZero=TRUE）
bIndexOccured		BOOL	TRUE/ FALSE	FALSE	TRUE: The mark pulse record has been detected, and the index position is stored in the input fIndexPosition. The zero return mode is FAST_BSLOW_S_STOP, which takes effect when FAST_SLOW_S_STOP.
fIndexPosition		LREAL		0	Position recorded at the mark pulse
bIgnoreHWLimit	Ignore hard limit	BOOL	TRUE/ FALSE	FALSE	TRUE: Set the hardware limit enable switch to FALSE during the zero return process. If the same physical switch is used for the hardware limit switch and the reference switch, then the hardware control will be set to FALSE.

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
bDone	Execution complete	BOOL	TRUE/ FALSE	FALSE	TRUE：The execution of the function block is completed, and the zero return is completed
Busy	Executing	BOOL	TRUE/ FALSE	FALSE	TRUE：Function block is executing
Command Aborted	Execution interrupt	BOOL	TRUE/ FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block
Error	Error	BOOL	TRUE/ FALSE	FALSE	TRUE：An error occurred
ErrorID	ErrorID	SMC_ ERROR	Refer to SMC_ ERROR	0	Output fault code when an error occurs
bStart LatchingIndex	Latched state	BOOL	TRUE/ FALSE	FALSE	For some search modes, the indicator pulse is calculated. TRUE: The storage of the index pulse has been started and must be completed by the application. The result is in the input bIndexOccured and fIndexPosition.

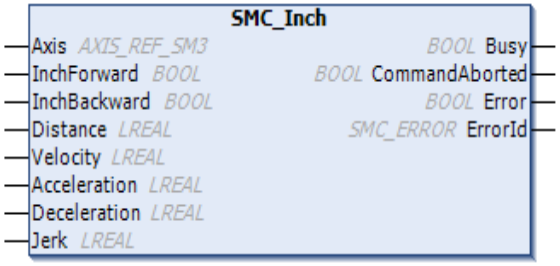
3) Function description

After SMC_Homing is started by the rising edge of bExecute, the axis will start to move in the direction defined by nDirection at the speed fVelocityFast until bReferenceSwitch=FALSE. Then the axis will stop slowly and leave the reference switch bReferenceSwitch=TRUE at the speed fVelocitySlow in the opposite direction and the zero return is completed.

SMC_Inch

This function block is for inching operation, single-step motion control, and can realize step-by-step single-step control through the program.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_Inch	Axis inching operation command		SMC_Inch(Axis:= , InchForward:= , InchBackward:= , Distance:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , Busy=> , CommandAborted=> , Error=> , ErrorId=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
InchForward	Positive effect	BOOL	TRUE/ FALSE	FALSE	TRUE: The given parameter values (speed, acceleration, deceleration and jerk) of the axis move in the positive direction until the distance has been moved. InchForward must be set to FALSE and set to TRUE again before the movement can be triggered again. If InchForward is set to FALSE before moving the distance, the axis will immediately decelerate to a speed value of 0. Then set Busy to FALSE. If InchForward and InchBackward are TRUE at the same time, no action is performed.
InchBackward	Negative effective	BOOL	TRUE/ FALSE	FALSE	TRUE: The given parameter values (speed, acceleration, deceleration and jerk) of the axis move in the negative direction until the distance has been moved. InchBackward must be set to FALSE and then set to TRUE again before the movement can trigger the start again. If InchBackward is set to FALSE before moving the distance, the axis will immediately decelerate to a speed value of 0. Then set Busy to FALSE. If InchBackward and InchForward are TRUE at the same time, no action is performed.
Distance	Moving distance	LREAL	Data range		The movement distance of one axis movement
Velocity	Target speed	LREAL	positive	0	Specify the maximum speed of the target (u/s)
Acceleration	Acceleration	LREAL	positive	0	Specify acceleration (u/s²)

Deceleration	decrease speed	LREAL	positive	0	Designated deceleration (u/s²)
Jerk	Jerk	LREAL	positive	0	jerk(u/s³)

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is being executed
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	ErrorID	SMC_ ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs

3) Function description

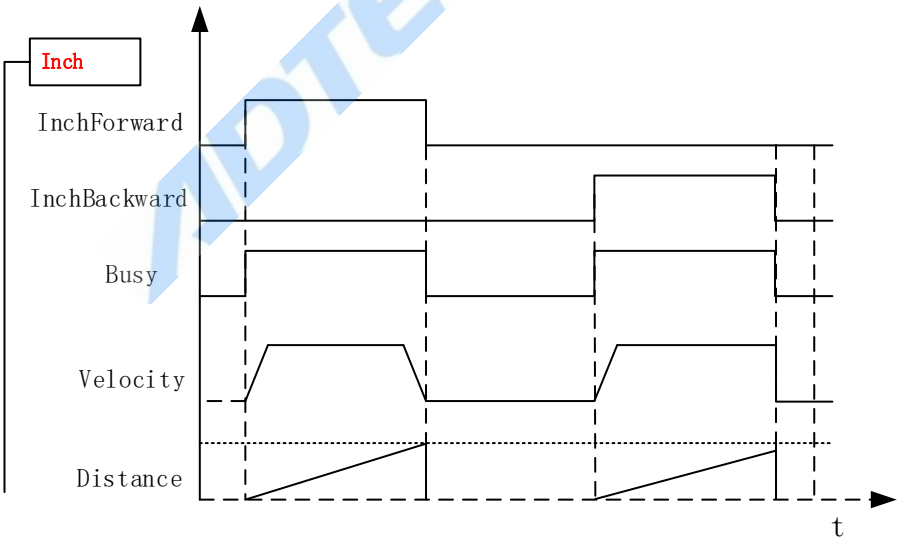
✧ The running state of this function block is in Standstill, and the state when the instruction is running is Discrete Motion. Pay attention to the running state of this axis during the execution of the instruction, and avoid interrupting other instructions of this axis or being interrupted by other instructions.

✧ The parameters acceleration (Acceleration) and deceleration (Deceleration) are zero when the command is running, and the command is running in an abnormal state, but the state of the axis is Discrete Motion;

4) Timing diagram

✧ InchForward/InchBackward of the function block must have TRUE/FALSE conditions.

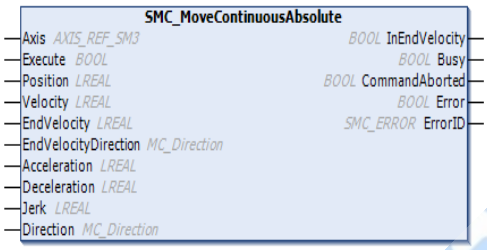
✧ Busy of the function block indicates that the current function block is being executed.



SMC_MoveContinuousAbsolute

This function block makes the axis run continuously at the absolute position, the absolute position is specified by Position, and after reaching the position, it runs continuously at the speed of EndVelocity.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_Move Continuous Absolute	Axis absolute position continuous motion command		SMC_MoveContinuousAbsolute(Axis:= , Execute:= , Velocity:= , EndVelocity:= , EndVelocityDirection:= , Acceleration:= , Deceleration:= , Jerk:= , Direction:= , InEndVelocity=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	TRUE: On the rising edge, the function block starts to execute.
Position	position	LREAL	Data range	0	Target position of the movement
Velocity	Running speed	LREAL	positive	0	Maximum speed (u / s)
EndVelocity	End of operation speed	LREAL	positive	0	Running speed after instruction execution is completed [u/s]
EndVelocityDirection	End speed direction	MC_Direction		current	allowance：positive, negative, current; Value not allowed: shortest, fastest
Acceleration	Acceleration	LREAL	positive	0	Specify acceleration (u / s^2)
Deceleration	decrease speed	LREAL	positive	0	Designated deceleration (u / s^2)
Jerk	jerk	LREAL	positive	0	jerk(u / s^3)
Direction	direction	MC_Direction	positive, negative, current, shortest, fastest	shortest	Linear axis: positive, negative Axis of rotation: positive, negative, current, shortest, fastest

- Output variables

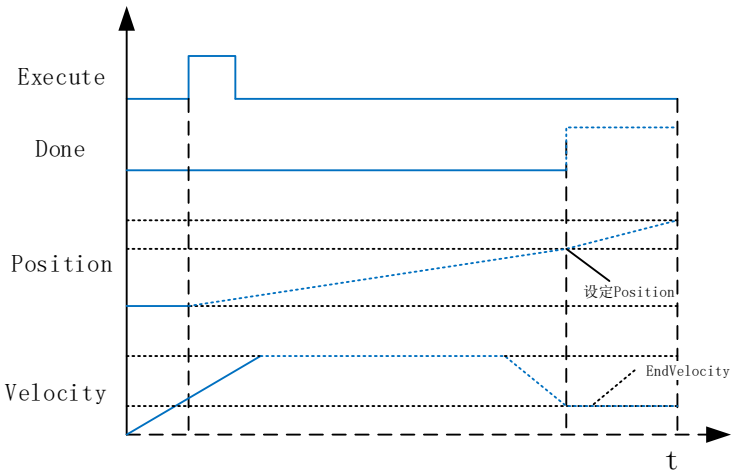
Output variables	Name	Type of data	Effective range	Initial value	Description
InEndVelocity	Command position reached	BOOL	TRUE/FALSE	FALSE	TRUE: Command position reached
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is being executed
CommandAborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	ErrorID	SMC_ERROR	参阅 SMC_ERROR	0	Output fault code when an error occurs

3) Function description

- ✧ The function block executes an absolute move. Contrary to MC_MoveAbsolute, it will not reach the target position at a speed value of 0, but at a specified speed. When reaching the target position, the function block sets the InEndVelocity output and maintains this speed.
- ✧ The start instruction is the rising edge of Execute. This instruction can be valid at the rising edge of Discrete Motion repeatedly, and the latest Position position can be refreshed each time.
- ✧ The running state of this function block is in Standstill, and the state when the instruction is running is Discrete Motion. Pay attention to the running state of this axis during the execution of the instruction, and avoid interrupting other instructions of this axis or being interrupted by other instructions.
- ✧ When the command is running, the parameters Acceleration and Deceleration are zero, and the command is running in an abnormal state, but the state of the axis is Discrete Motion;

4) Timing diagram

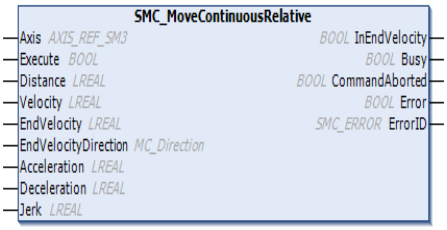
- ✧ The Execute of the function block must have a rising edge condition;
- ✧ Done of the function block indicates the completion of the normal execution of the instruction;
- ✧ Busy of the function block indicates that the current function block is being executed;



SMC_MoveContinuousRelative

This function block causes the axis to run continuously at the relative position, the relative position is specified by Distance, and it runs continuously at the speed of EndVelocity after reaching the position.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_Move Continuous Relative	Axis relative position continuous motion command		SMC_MoveContinuousRelative(Axis:= , Execute:= , Distance:= , Velocity:= , EndVelocity:= , EndVelocityDirection:= , Acceleration:= , Deceleration:= , Jerk:= , InEndVelocity=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	TRUE: Rising edge: Start to execute the function block.
Distance	relative position	LREAL	Data range	0	Relative position of movement
Velocity	Running speed	LREAL	positive	0	Maximum speed (u/s)
EndVelocity	End of run speed	LREAL	positive	0	Running speed after instruction execution is completed [u/s]
EndVelocity Direction	End speed direction	MC_Direction		current	allowance：positive, negative, current; Value not allowed: shortest, fastest
Acceleration	Acceleration	LREAL	positive	0	Specify acceleration (u/s^2)
Deceleration	decrease speed	LREAL	positive	0	Designated deceleration (u/s^2)
Jerk	jerk	LREAL	positive	0	jerk(u/s^3)
Direction		MC_Direction	positive, negative, current, shortest, fastest	shortest	Linear axis: positive, negative Axis of rotation: positive, negative, current, shortest, fastest

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
InEndVelocity	Command position	BOOL	TRUE/FALSE	FALSE	TRUE: Command position reached

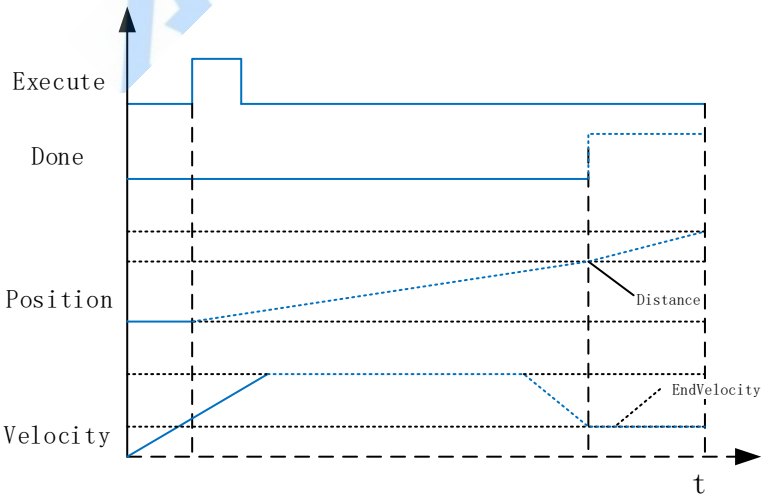
	reached				
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is in execution
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred.
ErrorID	ErrorID	SMC_ ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs

3) Function description

- ✧ The function block executes an absolute move. Contrary to MC_MoveRelative, it will not reach the target position at a speed value of 0, but at a specified speed. When reaching the target position, the function block sets the InEndVelocity output and maintains this speed.
- ✧ The start instruction is the rising edge of Execute. This instruction can be valid at the rising edge of Discrete Motion repeatedly, and the latest Distance position can be refreshed each time.
- ✧ The running state of this function block is in Standstill, and the state when the instruction is running is Discrete Motion. Pay attention to the running state of this axis during the execution of the instruction, and avoid interrupting other instructions of this axis or being interrupted by other instructions.
- ✧ The parameters acceleration (Acceleration) and deceleration (Deceleration) are zero when the command is running, and the command is running in an abnormal state, but the state of the axis is Discrete Motion;

4) Timing diagram

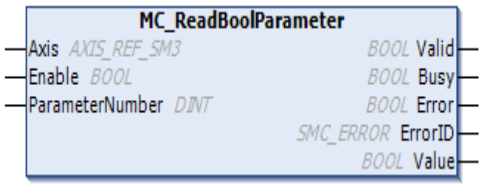
- ✧ The Execute of the function block must have a rising edge condition;
- ✧ Done of the function block indicates the completion of the normal execution of the instruction;
- ✧ Busy of the function block indicates that the current function block is being executed;



MC_ReadBoolParameter

Read the bit parameter of the drive shaft and save it in the variable unit defined by yourself.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_ReadBool Parameter	Read axis parameter command		MC_ReadBoolParameter(Axis:= , Enable:= , ParameterNumber:= , Valid=> , Busy=> , Error=> , ErrorID=> , Value=>);

2) Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	TRUE: Rising edge: start to execute the function block, read the servo parameters
Parameter Number	Axis parameter number	DINT	Data range		Access index, sub-index number and serial number of axis parameters

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Valid	Read successfully	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is executed and the parameters can be read normally
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is being executed
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	ErrorID	SMC_ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs
Value	Axis position	BOOL			Bit state read by instruction

3) Function description

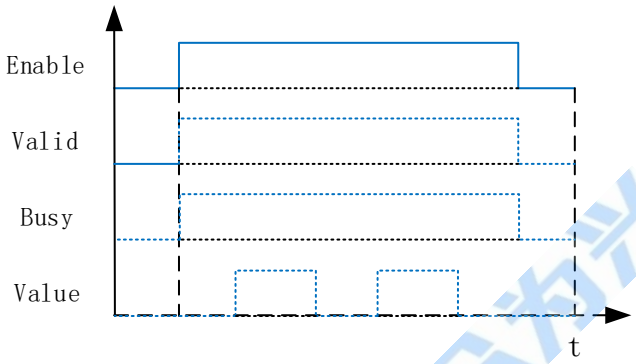
✧ diParameterNumber(DINT):=
-DWORD_TO_DINT(SHL(TO_DOWRD(usiDataLength), 24)(Data length in the object dictionary)
+ **SHL**(TO_DWORD(uiIndex), 8)（Index in the object dictionary -16Bit）
+ usisubIndex);（Sub-index in the object dictionary -8Bit）

usiDataLength: Fill in according to the number of bytes; 1 byte is 26#01; 2 bytes are 26#02; 4 bytes are 26#04, etc.

- ✧ Read the bit data status in the drive through MC_ReadBoolParameter, the command is Enable high-level enable effect. The instructions can be used repeatedly without affecting each other.

4) Timing diagram

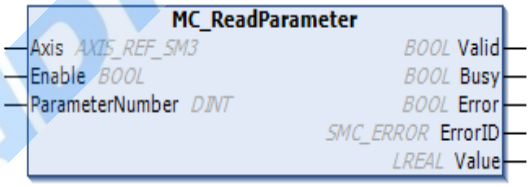
- ✧ The Enable of the function block must have a rising edge condition;
- ✧ Valid of the function block means that the read Valid is valid bit status data;
- ✧ Busy of the function block indicates that the current function block is being executed;



MC_ReadParameter

Read the parameters of the drive shaft, the return value is LREAL type.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Read Parameter	Read axis parameter command		MC_ReadParameter(Axis:= , Enable:= , ParameterNumber:= , Valid=> , Busy=> , Error=> , ErrorID=> , Value=>);

2) Related variables

- ● Input and output variables

Input variables	Name	Type of data	Value range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- ● Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	TRUE: Rising edge: start to execute the function block, read the servo parameters
Parameter Number	Axis parameter number	DINT	Data range		Access index, sub-index number and serial number of axis parameters

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Valid	Read successfully	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is executed and the parameters can be read normally
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is being executed.
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block.
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	ErrorID	SMC_ERROR	参阅 SMC_ERROR	0	Output fault code when an error occurs
Value	Axis parameters	LREAL			Command to read the parameters of the axis

3) Function description

✧ diParameterNumber(DINT):= -DWORD_TO_DINT(SHL(TO_DOWRD(usiDataLength), 24) (data length in the object dictionary)

✧ + SHL(TO_DWORD(uiIndex), 8) (index in object dictionary-16Bit)

✧ + usisubIndex); (Sub index in the object dictionary-8Bit)

✧ usiDataLength: Fill in the number of bytes; 1 byte is 26#01; 2 bytes are 26#02; 4 bytes are 26#04, etc.

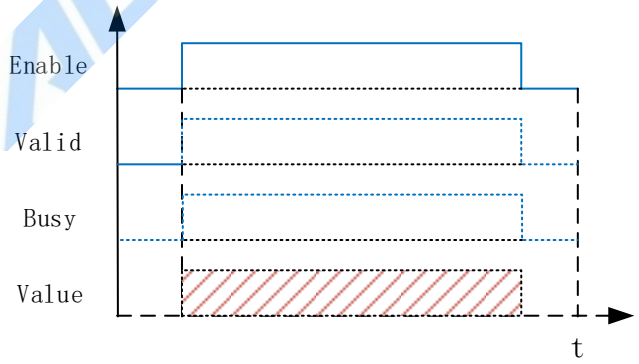
✧ Read the data in the drive through MC_ReadBoolParameter, the command is Enable high-level enable effect. The instructions can be used repeatedly without affecting each other.

4) Timing diagram

✧ The Enable of the function block must have a rising edge condition;

✧ The Valid of the function block indicates that the read Valid is valid bit state data;

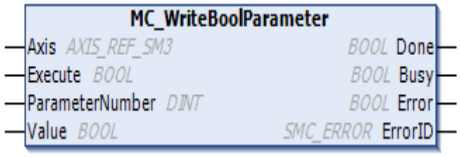
✧ Busy of the function block indicates that the current function block is being executed;



MC_WriteBoolParameter

Set the bit parameter of the drive axis.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_WriteBoolParameter	Set axis parameter command		MC_WriteBoolParameter(Axis:= , Execute:= , ParameterNumber:= , Value:= , Done=> , Busy=> , Error=> , ErrorID=>);

2) Related variables

- Input and output variables

Input variables	Name	Type of data	Value range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Rising edge: start to execute the function block, set the servo parameters once
Parameter Number	Axis parameter number	DINT	Data range		Access index, sub-index number and serial number of axis parameters
Value	Set value	BOOL		FALSE	Set parameter status

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Valid	Read successfully	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is executed and the parameter can be set successfully.
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is in execution
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	ErrorID	SMC_ ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs

3) Function description

✧ diParameterNumber(DINT):=
-DWORD_TO_DINT(SHL(TO_DOWRD(usiDataLength), 24) (data length in the object dictionary)
+ SHL(TO_DWORD(uiIndex), 8) (index in object dictionary-16Bit)
+ usisubIndex); (Sub index in the object dictionary-8Bit)

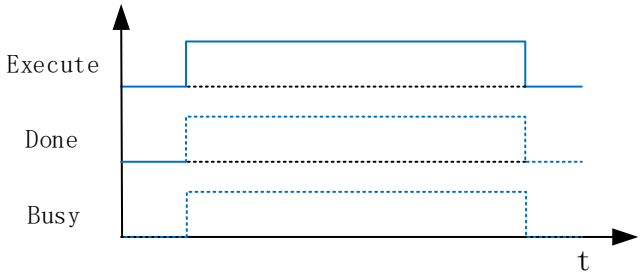
usiDataLength: Fill in the number of bytes; 1 byte is 26#01; 2 bytes are 26#02; 4 bytes are 26#04, etc.

✧ Set the bit parameter data in the drive through MC_WriteBoolParameter, the instruction is the Enable rising edge trigger. The instructions can be used repeatedly without affecting each other.

4) Timing diagram

✧ The Execute of the function block must have a rising edge condition;

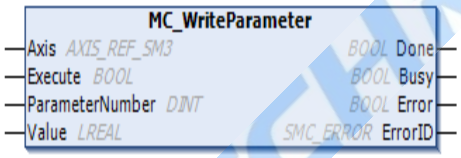
- ✧ Done of the function block indicates that the setting operation is successful;
- ✧ Busy of the function block indicates that the current function block is being executed;



MC_WriteParameter

Set the parameters of the drive shaft.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Write Parameter	Set axis parameter command		MC_WriteParameter(Axis:= , Execute:= , ParameterNumber:= , Value:= , Done=> , Busy=> , Error=> , ErrorID=>);

2) Related variables

- Input and output variables

Input variables	Name	Type of data	Value range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Rising edge: start to execute the function block, set the servo parameters once
Parameter Number	Axis parameter number	DINT	Data range		Access index, sub-index number and serial number of axis parameters
Value	Set value	LREAL		0	Set parameter value

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Valid	Read successfully	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete, and the parameter can be set. The operation is successful.
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is being executed.

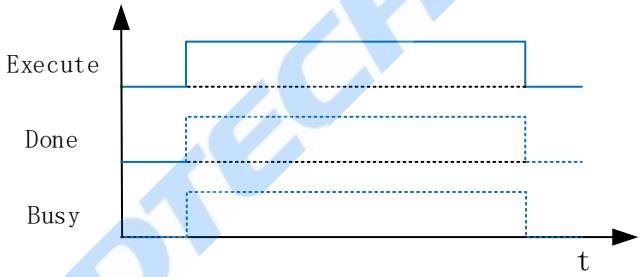
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block.
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred.
ErrorID	ErrorID	SMC_ERROR	参阅 SMC_ERROR	0	Output fault code when an error occurs

3) Function description

- diParameterNumber(DINT):= -DWORD_TO_DINT(SHL(TO_DOWRD(usiDataLength), 24) (data length in the object dictionary)
  + SHL(TO_DWORD(uiIndex), 8) (index in object dictionary-16Bit)
  + usisubIndex); (Sub index in the object dictionary-8Bit)
  usiDataLength: Fill in the number of bytes; 1 byte is 26#01; 2 bytes are 26#02; 4 bytes are 26#04, etc.
- Set the parameter data in the drive through MC_WriteParameter, and the instruction is the Enable rising edge trigger. The instructions can be used repeatedly without affecting each other.

4) Timing diagram

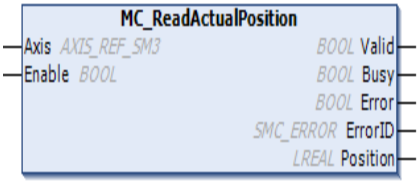
- The Execute of the function block must have a rising edge condition;
- Done of the function block indicates that the setting operation is successful;
- Busy of the function block indicates that the current function block is being executed;



MC_ReadActualPosition

Read the actual running position of the drive shaft and save it in the variable unit defined by yourself.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Read Actual Position	Read axis actual position command		MC_ReadActualPosition(Axis:= , Enable:= , Valid=> , Busy=> , Error=> , ErrorID=> , Position=>);

2) Related variables

- Input and output variables

Input variables	Name	Type of data	Value range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	When TRUE, read the current actual position of the axis

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Valid	Read successfully	BOOL	TRUE/FALSE	FALSE	When TRUE, it means that the parameters can be read normally
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is being executed
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred
ErrorID	ErrorID	SMC_ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs
Position	Obtained location	LREAL	-	0	The current position read by the command

3) Function description

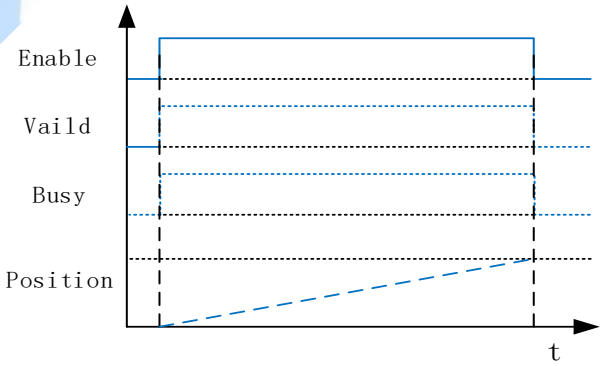
✧ Read the actual position command in the drive through this command, the command is the Enable level enable effect. The instructions can be used repeatedly without affecting each other.

4) Timing diagram

✧ The Enable of the function block must be TRUE;

✧ Valid of the function block indicates that the read Position is a valid value;

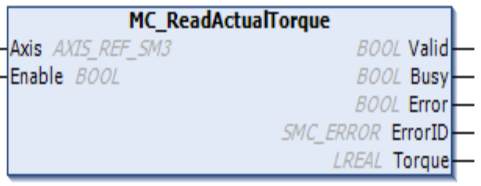
✧ Busy of the function block indicates that the current function block is being executed;



MC_ReadActualTorque

Read the current torque value of the drive running axis and save it in the variable unit defined by yourself.

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_ReadActualTorque	Read the actual torque command of the axis		MC_ReadActualTorque(Axis:= , Enable:= , Valid=> , Busy=> , Error=> , ErrorID=> , Torque=>);

2) Related variables

- Input and output variables

Input variables	Name	Type of data	Value range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variables	Name	Type of data	Value range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Rising edge: start to execute the function block, read the current actual torque of the axis

- Output variables

Output variables	Name	Type of data	Effective range	Initial value	Description
Valid	Read successfully	BOOL	TRUE/FALSE	FALSE	TRUE means that the parameters can be read normally
Busy	Executing	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction is being executed
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE：An error occurred.
ErrorID	ErrorID	SMC_ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs
Torque	Torque obtained	LREAL		0	The current torque read by the command

3) Function description

✧ Use this command to read the current torque value in the drive. The command is the Enable level enabling effect. The instructions can be used repeatedly without affecting each other.

Note: As long as Enable is TRUE, the function block will return the current torque or force value. Once the data output torque is valid, Valid will be set to TRUE. If the enable is reset, the data will lose its validity, and the valid will be reset regardless of whether new data is available.

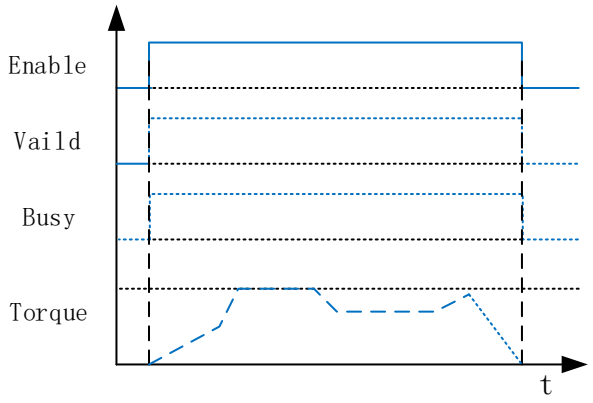
4）Timing diagram

✧ Enable of function block must be TRUE condition;

✧ Valid of the function block indicates that the Torque read is a valid value;

✧ Busy of the function block indicates that the current

function block is being executed;



MC_ReadActualVelocity

Read the current speed of the drive shaft and save it in the variable unit defined by yourself。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Read Actual Velocity	Read the current speed command of the axis		MC_ReadActualVelocity(Axis:= , Enable:= , Valid=> , Busy=> , Error=> , ErrorID=> , Velocity=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Rising edge: start to execute the function block, read the current actual speed of the axis

- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
Valid	Read successfully	BOOL	TRUE/FALSE	FALSE	TRUE means that the parameters can be read normally
Busy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The current command is being executed
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: An error occurred
ErrorID	error	SMC_	Refer to	0	Output fault code when an

	code	ERROR	SMC_ERROR		error occurs
Velocity	Acquired speed	LREAL		0	The current speed value read by the instruction

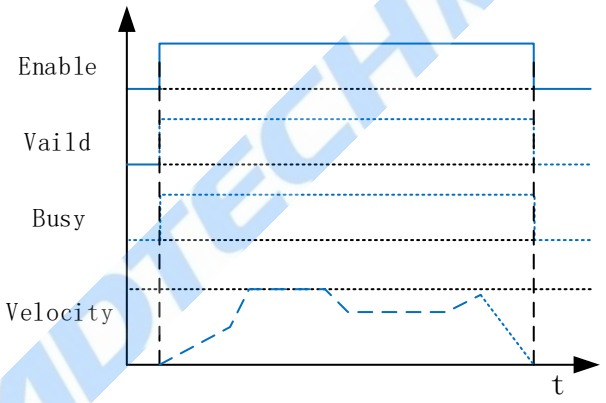
3）Function Description

- Read the actual speed in the drive through this instruction, the instruction is the Enable level enable effect. The instructions can be used repeatedly without affecting each other.

Note: As long as Enable remains TRUE, the function block will return the current speed value. Once the data output speed becomes valid, Valid will be set to TRUE. If the enable is reset, the data will lose its validity, and the Valid will also be reset regardless of whether new data is available.

4）Timing diagram

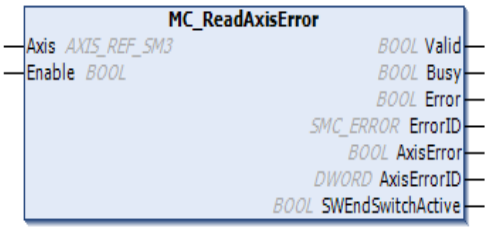
- The Enable of the function block must be TRUE condition;
- The Valid of the function block indicates that the read Velocity is a valid value;
- Busy of the function block indicates that the current function block is being executed;



MC_ReadAxisError

Read the current error condition of the drive shaft and save it in the variable unit defined by yourself。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Read AxisError	Read the current error command of the axis		MC_ReadAxisError(Axis:= , Enable:= , Valid=> , Busy=> , Error=> , ErrorID=> , AxisError=> , AxisErrorID=> , SWEndSwitchActive=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Rising edge: start to execute the function block, read the current error of the axis

- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
Valid	Read successfully	BOOL	TRUE/ FALSE	FALSE	TRUE means that the parameters can be read normally.
Busy	In execution	BOOL	TRUE/ FALSE	FALSE	TRUE: The current command is being executed.
Error	error	BOOL	TRUE/ FALSE	FALSE	TRUE: An error occurred.
ErrorID	error code	SMC_ ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs
AxisError	Get error status	BOOL		0	TRUE: An axis error has occurred
AxisErrorID	Axis error code	DWORD			Axis error identification code
SWEndSwitch Active	Soft limit is valid	BOOL			TRUE: The software limit switch is turned on.

3）Function Description

✧ Read the error code in the drive through this instruction, the instruction is the Enable level enable effect. The instructions can be used repeatedly without affecting each other.

4）Timing diagram

✧ Enable of function block must be TRUE condition;

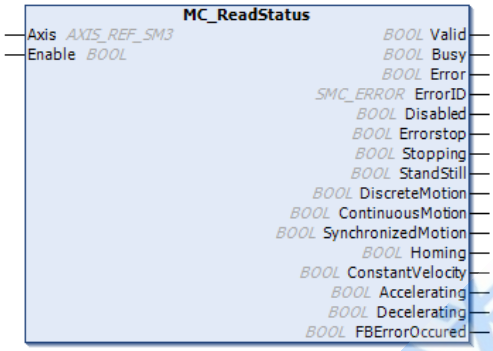
✧ Valid of the function block indicates that the read AxisError and AxisErrorID are valid data;

✧ Busy of the function block indicates that the current function block is being executed;

MC_ReadStatus

Read the status data of the drive shaft and save it in the variable unit defined by yourself。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Read Status	Read the current state command of the axis		MC_ReadStatus(Axis:= , Enable:= , Valid=> , Busy=> , Error=> , ErrorID=> , Disabled=> , Errorstop=> , Stopping=> , StandStill=> , DiscreteMotion=> , ContinuousMotion=> , SynchronizedMotion=> , Homing=> , ConstantVelocity=> , Accelerating=> , Decelerating=> , FBErrorOccured=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Rising edge: start to execute the function block, read the current error of the axis

- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
Valid	Read successfully	BOOL	TRUE/FALSE	FALSE	TRUE means that the parameters can be read normally
Busy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The current command is being executed
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: An error occurred.
ErrorID	error code	SMC_ ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs
Disabled	The axis is not enabled	BOOL	TRUE/FALSE	FALSE	TRUE: The axis is not enabled
Errorstop	Axis error status	BOOL	TRUE/FALSE	FALSE	TRUE: The axis is in the wrong operating state
Stopping	Axis stop process status	BOOL	TRUE/FALSE	FALSE	TRUE: The axis is in the process of stopping
StandStill	Axis standard	BOOL	TRUE/FALSE	FALSE	TRUE: The axis is in the standard (runnable) state

	state				
Discrete Motion	Axis discrete motion state	BOOL	TRUE/FALSE	FALSE	TRUE: The axis is in discrete motion
Continuous Motion	Axis continuous motion state	BOOL	TRUE/FALSE	FALSE	TRUE: The axis is in continuous motion
SynchronizedMotion	Axis synchronous operation status	BOOL	TRUE/FALSE	FALSE	TRUE: The axis is in synchronous motion
Homing	Axis home state	BOOL	TRUE/FALSE	FALSE	TRUE: The axis is in the home state
Constant Velocity	Axis running speed reached	BOOL	TRUE/FALSE	FALSE	TRUE: The axis reaches the running speed
Accelerating	Axis acceleration process status	BOOL	TRUE/FALSE	FALSE	TRUE: The axis is in the accelerating process state
Decelerating	Shaft deceleration process status	BOOL	TRUE/FALSE	FALSE	TRUE: The axis is in the deceleration process state
FBError Occured	Axis function block error status	BOOL	TRUE/FALSE	FALSE	TRUE: Axis function block error flag

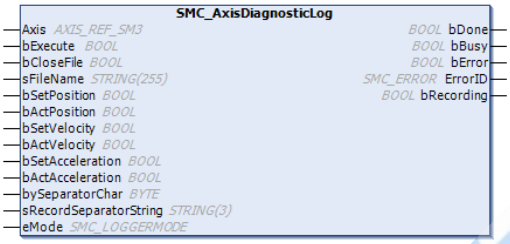
3）Function Description

- Read the various states of the axis through this instruction, the instruction is the Enable level enable effect. The instructions can be used repeatedly without affecting each other.
- The enablement of the function block must be TRUE;;
- Valid of the function block indicates that the read status data is valid;
- Busy of the function block indicates that the current function block is being executed;

SMC_AxisDiagnosticLog

Periodically write a parameter of the axis to the file。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Axis Diagnostic Log			SMC_AxisDiagnosticLog(Axis:= , bExecute:= , bCloseFile:= , sFileName:= , bSetPosition:= , bActPosition:= , bSetVelocity:= , bActVelocity:= , bSetAcceleration:= , bActAcceleration:= , bySeparatorChar:= , sRecordSeparatorString:= , eMode:= , bDone=> , bBusy=> , bError=> , ErrorID=> , bRecording=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Rising edge: start execution of the function block
bCloseFile	Close file	BOOL	TRUE/FALSE		TRUE: close the file immediately
sFileName	file name	STRING (255)			Storage file name (path)
bSetPosition	Record setting location	BOOL	TRUE/FALSE		TRUE: Record the set position when executing the command
bActPosition	Record the actual location	BOOL	TRUE/FALSE		TRUE: Record the actual position when the instruction is executed
bSetVelocity	Record setting speed	BOOL	TRUE/FALSE		TRUE: Record the set speed when executing the command
bActVelocity	Record actual speed	BOOL	TRUE/FALSE		TRUE: Record the actual speed when the instruction is executed
bSetAcceleration	Record set acceleration	BOOL	TRUE/FALSE		TRUE: Record the set acceleration when executing the command
bActAcceleration	Record actual	BOOL	TRUE/FALSE		TRUE: Record the actual acceleration when the

	acceleration				instruction is executed
bySeparatorChar		BYTE		9	ASCII code value, written between two different values on the same date
sRecordSeparatorString		STRING (3)			The string written at the end of the date
eMode	mode	SMC_LOGGERMODE		LOG_CONTINUOUS	LOG_CONTINUOUS: Log in to the file continuously. LOG_AT_CLOSE: Continuous login buffer (10kByte). When bCloseFile is TRUE, write the contents of the buffer to the file

- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
bDone	Read successfully	BOOL	TRUE/FALSE	FALSE	TRUE: The save is complete, the log is terminated, and the file is closed.
bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The current command is being executed.
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: An error occurred.
ErrorID	error code	SMC_ERROR	参阅 SMC_ERROR	0	Output fault code when an error occurs
bRecording	Shaft brake status	BOOL	TRUE/FALSE	FALSE	TRUE: The parameter is being saved and recorded

3）Function Description

✧ This function block is used to cyclically write a set of parameter values belonging to an axis into a file. This output file is very suitable for diagnostic purposes. Since it usually takes some time to write data on the data medium, this module stores the collected data in a buffer of 10 kByte, and does not write data until the module action "WriteToFile" is called. This action call should be placed in a slower (about 50 milliseconds) task with a lower priority to prevent interference with the actual motion task and the motion itself.Once the buffer is exceeded, the module will create an error output.

4）Timing diagram

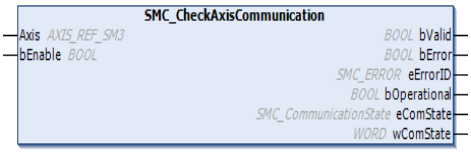
5）Error description

bExecute rising edge, axis error report, Error output; invalid axis input, Error output.

SMC_CheckAxisCommunication

Check the current communication status of the drive。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_CheckAxisCommunication	Test drive communication command		SMC_CheckAxisCommunication(Axis:= , bEnable:= , bValid=> , bError=> , eErrorID=> , bOperational=> , eComState=> , wComState=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
bEnable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Rising edge: start to execute the function block, start to detect

- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
bValid	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The instruction execution output is valid
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: An error occurred.
eErrorID	error code	SMC_ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs
bOperational	Communication is normal	BOOL	TRUE/FALSE	FALSE	TRUE: Communication is normal (code 100) and operable
eComState	Communication status	SMC_CommunicationState			**Communication status (enumeration)**: SMC_COMSTATE_NOT_STARTED Communication did not start SMC_COMSTATE_VARIABLE_INITIALIZATION Communication variable initialization SMC_COMSTATE_BASE_COM_INITIALIZATION Basic port initialization SMC_COMSTATE_DRIVE_INITIALIZATION Communication driver initialization SMC_COMSTATE_DRIVE_WAITING_FOR_SYNC Sync warning

					SMC_COMSTATE_INITIALIZATION_DONE loading finished **SMC_COMSTATE_OPERATIONAL** Communication can be used normally **SMC_COMSTATE_REINITIALIZATION** Communication reinitialization **SMC_COMSTATE_ERROR** Communication error **SMC_COMSTATE_UNKNOWN** I don't know the communication status
wComState	Communication code	DWORD			The value is the same as the input and output **AXIS.wCommunicationState**, which represents the code of the current communication state, refer to the value of **AXIS_REF_SM3** parameter 1013

3) Function Description

- bEnable is TRUE, no error, bValid outputs TRUE. Perform axis communication status check. Check the axis communication status when bValid outputs TRUE. When eComState output is SMC_COMSTATE_OPERATIONAL, bOperational output is TRUE.。

4) Timing diagram

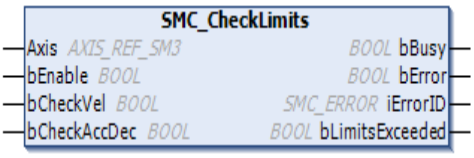
5) Error description

bExecute rising edge, axis error report, Error output; invalid axis input, Error output.

SMC_CheckLimits

Check whether the current axis setting value exceeds the maximum value of the controller configuration。

1) Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Check Limits	Detection limit instruction		SMC_CheckLimits(Axis:= , bEnable:= , bCheckVel:= , bCheckAccDec:= , bBusy=> , bError=> , iErrorID=> , bLimitsExceeded=>);

2) Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bEnable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Execute function block when TRUE
bCheckVel	Speed check	BOOL	TRUE/FALSE	FALSE	TRUE: Perform speed check FALSE: Do not perform speed check
bCheckAccDec	Acceleration and deceleration check	BOOL	TRUE/FALSE	FALSE	TRUE: Perform acceleration and deceleration check FALSE: Do not perform acceleration and deceleration checks

- Output variable

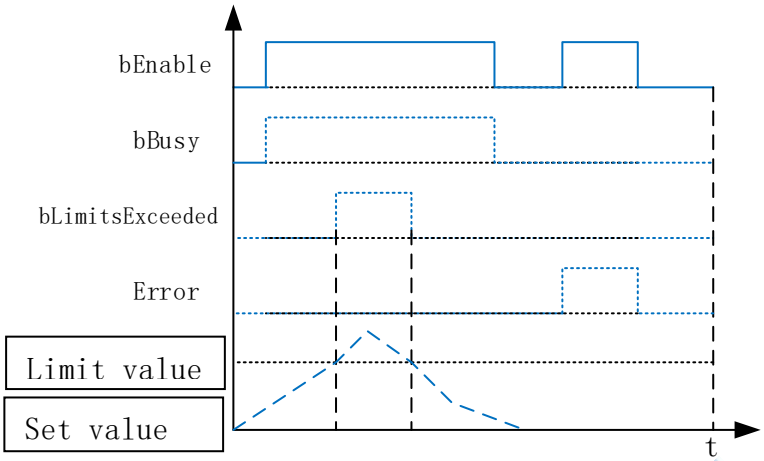
Output variable	Name	Type of data	Effective range	Initial value	Description
bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: Instruction execution check FALSE: No check is performed
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: An error occurred.
iErrorID	error code	SMC_ ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs
bLimits Exceeded	Check limit	BOOL	TRUE/FALSE	FALSE	TRUE: The current point set speed or acceleration/deceleration exceeds the value configured in the controller (Axis.fSWMaxVelocity, Axis.fSWMaxAcceleration, Axis.fSWMaxDeceleration)

2）Function Description

✧ bEnable is TRUE, bBusy outputs TRUE. Perform axis speed and acceleration check. The set speed or acceleration/deceleration of the current axis exceeds the set value of Axis.fSWMaxVelocity, Axis.fSWMaxAccleration, Axis.fSWMaxDeceleration, and the bLimitsExceeded signal output is TRUE.

Note: This function only checks that the current command speed or acceleration/deceleration exceeds the set limit, and cannot stop the axis.

4）Timing diagram



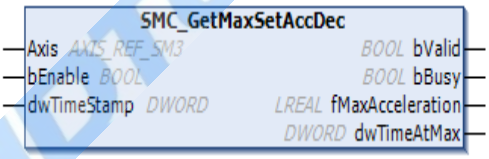
5）Error description

✧ bExecute rising edge, axis error report, Error output; invalid axis input, Error output.

SMC_GetMaxSetAccDec

Maximum acceleration/deceleration of reading axis。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GetMaxSetAccDec	Maximum axis acceleration/deceleration command		SMC_GetMaxSetAccDec(Axis:= , bEnable:= , dwTimeStamp:= , bValid=> , bBusy=> , fMaxAcceleration=> , dwTimeAtMax=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
bEnable	Execution conditions	BOOL	TRUE/FALSE	FALSE	TRUE: Start to execute the function block, check
dwTimeStamp	Speed check	DWORD			Optional timestamp input; can be used to find what happens when the maximum is

- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
bValid	Command is valid	BOOL	TRUE/FALSE	FALSE	TRUE: instruction execution is valid
bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The command is being executed
fMax Acceleration	Maximum addition and subtraction	LREAL			Maximum acceleration/deceleration value (positive is acceleration, negative is deceleration, the maximum absolute value of acceleration/deceleration is the final value)
dwTimeAtMax	Speed value Maximum value corresponds to timestamp	DWORD			The value of dwTimeStamp corresponding to the maximum acceleration and deceleration (for example, when the acceleration continues to increase, the value follows dwTimeStamp, and the value of fMaxAcceleration is also updated. Once the acceleration reaches the maximum value, fMaxAcceleration records the maximum value and the maximum value of dwTimeStamp is also recorded)

3）Function Description

✧ bEnable is TRUE, no error, bValid outputs TRUE. Perform the maximum acceleration/deceleration detection of the axis.
When the absolute value of acceleration and deceleration is greater than the previously recorded value, fSWMaxAccleration and dwTimeAtMax will be refreshed

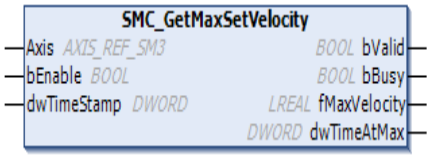
✧ dwTimeAtMax value corresponds to the dwTimeStamp value during maximum acceleration and deceleration, so dwTimeStamp should be set to be variable, such as a count value that follows the task period or a fixed time period.

✧ This function block can be used to measure the maximum absolute value of the axis acceleration (or deceleration). If bEnable is true, the measurement is taken, and as long as bResetMax is true, it will be reset to 0. Using dwTimeStamp, you can read any DWORD (such as call counter), and it will take over and output with the new maximum value.

SMC_GetMaxSetVelocity

Maximum speed of reading axis。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_ GetMaxSetVelocity	Read axis maximum speed command		SMC_GetMaxSetVelocity(Axis:= , bEnable:= , dwTimeStamp:= , bValid=> , bBusy=> , fMaxVelocity=> , dwTimeAtMax=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bEnable	Execution conditions	BOOL	TRUE/FALSE	FALSE	TRUE: start to execute the function block, read
dwTimeStamp	Speed check	DWORD			Optional timestamp input; can be used to find what happens when the maximum is

- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
bValid	Command is valid	BOOL	TRUE/FALSE	FALSE	TRUE: instruction execution is valid
bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The command is being executed
fMaxVelocity	Maximum speed value	LREAL			Maximum speed value (positive is positive, negative is reverse, the maximum absolute value is the final value)
dwTimeAtMax	The maximum value corresponds to the timestamp	DWORD			Maximum value of dwTimeStamp

3）Function Description

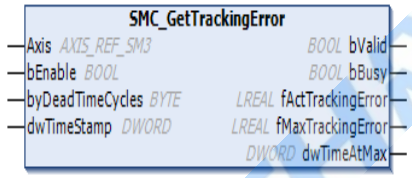
✧ bEnable is TRUE, no error, bValid outputs TRUE. Perform axis maximum speed detection. When the absolute value of acceleration and deceleration is greater than the previously recorded value, fSWMaxVelocity and dwTimeAtMax will be refreshed.

- dwTimeAtMax value corresponds to the dwTimeStamp value during maximum acceleration and deceleration, so dwTimeStamp should be set to be variable, such as a count value that follows the task period or a fixed time period.
- This function block can be used to measure the maximum speed of the axis. If bEnable is true, then the measurement is taken, and as long as bResetMax is true, it will be reset to 0. Using dwTimeStamp, you can read any DWORD (for example: call counter), it receives and outputs a new maximum value.

SMC_GetTrackingError

Measure the current or maximum hysteresis error (to the command position and actual position error) to compensate for the dead time.

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_ GetTrackingError	Axis lag deviation reading		SMC_GetTrackingError(Axis:= , bEnable:= , byDeadTimeCycles:= , dwTimeStamp:= , bValid=> , bBusy=> , fActTrackingError=> , fMaxTrackingError=> , dwTimeAtMax=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
bEnable	Execution conditions	BOOL	TRUE/FALSE	FALSE	TRUE: start to execute the function block, read
byDeadTimeCycles	Speed check	BYTE			The number of dead zone cycles, how many dwTimeStamp values are delayed by bEnable trigger to start hysteresis detection
dwTimeStamp					Optional timestamp input; can be used to find what happens when the maximum is

- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
bValid	Command effect	BOOL	TRUE/FALSE	FALSE	TRUE: instruction execution is valid
bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The command is being executed
fActTrackingError	Maximum speed	LREAL			Current deviation detection related to

	value				byDeadTimeCycles
fMaxTrackingError					Current deviation value (deviation between command position and feedback position)
dwTimeAtMax	Maximum value corresponds to timestamp	DWORD			Maximum deviation (positive is lagging, negative is leading, the maximum absolute value is the final value) Note: byDeadTimeCycles will affect this value.

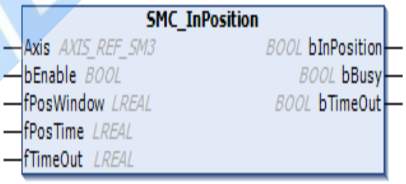
3）Function Description

- ✧ bEnable is TRUE, bValid outputs TRUE. Perform shaft lag deviation detection. When the absolute value of acceleration and deceleration is greater than the previously recorded value, fSWMaxTrackingError and dwTimeAtMax will be refreshed.
- ✧ dwTimeAtMax value corresponds to the dwTimeStamp value during maximum acceleration and deceleration, so dwTimeStamp should be set to be variable, such as a count value that follows the task period or a fixed time period.

SMC_InPosition

Monitor the deviation between the current axis set position and the actual value, and determine whether the axis is within the required deviation range through the set deviation window。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_ InPosition	Axis deviation monitoring		SMC_InPosition(Axis:= , bEnable:= , fPosWindow:= , fPosTime:= , fTimeOut:= , bInPosition=> , bBusy=> , bTimeOut=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
bEnable	Execution conditions	BOOL	TRUE/FALSE	FALSE	TRUE: start to execute the function block, read
fPosWindow	Deviation window	LREAL		0	Set the window for deviation monitoring, fPosWindow> Distance (the deviation between command position and

					feedback position), output bInPosition according to fPosTime
fPosTime	Trigger time	LREAL		0	The time when the deviation is within the window, used to trigger bInPosition, unit: second (S)
fTimeOut	overtime time	LREAL		0	Deviation timeout, unit: second (S)

- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
bInPosition	Deviation is normal	BOOL	TRUE/FALSE	FALSE	TRUE: The deviation value is within the setting window
bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The command is being executed
bTimeOut	Deviation timeout	BOOL	TRUE/FALSE	FALSE	?

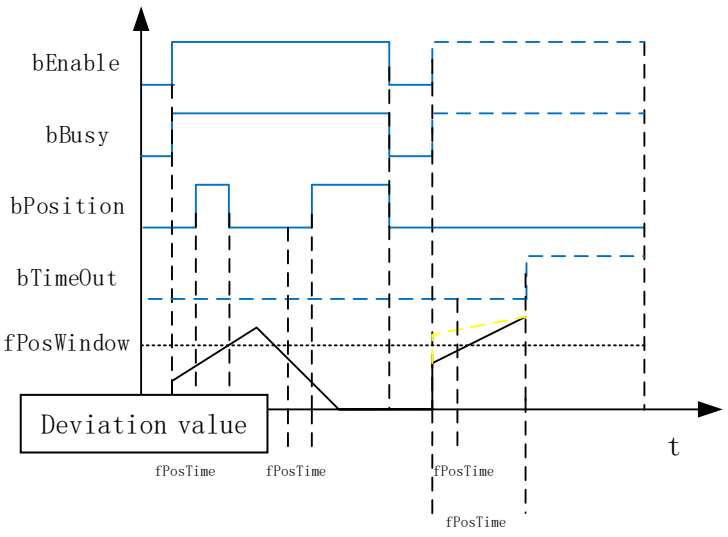
2) Function description

bEenable is TRUE, once the detected deviation is less than the set window **fPosWindow** for **fPosTime** seconds, **bInPosition** triggers to be TRUE. Once the detected deviation is greater than the set window, **bInPosition** immediately outputs FLASE.

Note: fPosTime must be set reasonably, otherwise it will cause fPosOut to trigger (for example, the cam period of the song cam is 2 seconds, the continuous deviation does not exceed the set window for 1.5 seconds, if fPosTime is set to be greater than 1.5 seconds, it will cause bInPosition Will not trigger).

- ✧ bEenable is TRUE, bBusy output is TRUE.
- ✧ The deviation value can be monitored. The data fCurrentDistance.bEenable in the SMC_bInPosition structure is TRUE. If bInPosition is not triggered for TRUE after the set time of fPosTime, then bTimeOut will be triggered as TRUE.

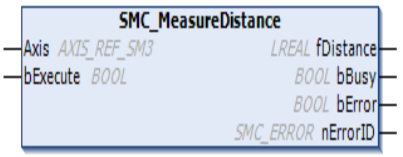
4) Timing diagram



SMC_MeasureDistance

This function block can be used on a rotating shaft to measure the distance covered by winding.

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_ MeasureDistance			SMC_MeasureDistance(Axis:= , bExecute:= , fDistance=> , bBusy=> , bError=> , nErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Execution conditions	BOOL	TRUE/FALSE	FALSE	TRUE: start to execute the function block, read

- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
fDistance	distance	LREAL			Travel distance from the start of the measurement
bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The command is being executed
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: An error occurred.
nErrorID	error code	SMC_ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs

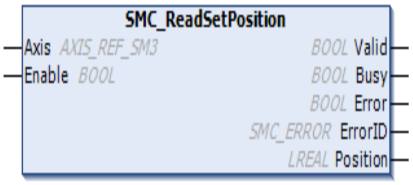
3）Function Description

✧ This function block can be used on a rotating shaft to measure the distance covered by winding.

SMC_ReadSetPosition

This function block can be used to read the current command position of the axis。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_ ReadSetPosition	Read axis command position		SMC_ReadSetPosition(Axis:= , Enable:= , Valid=> , Busy=> , Error=> , ErrorID=> , Position=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	TRUE: Start to execute the function block

- Input variable

Output variable	Name	Type of data	Effective range	Initial value	Description
Valid	Effective state	LREAL			TRUE: The function block output parameter is valid
Busy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The command is being executed
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: An error occurred.
ErrorID	error code	SMC_ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs
Position	position	LREAL			Current setting position

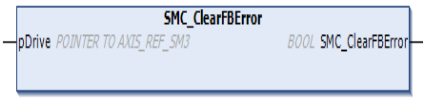
3）Function Description

- Eenable is TRUE, Valid if there is no error, bBusy output is TRUE. The value of Position output is the value of Axis.fSetPosition
- Eenable becomes FALSE, then Valid, bBusy output is FALSE. Position stays at the value before FALSE.

SMC_ClearFBError

This function is used to clear the error information of the function block。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_ ClearFBError	Clear function block error		SMC_ClearFBError(pDrive:=)

2）Related variables

- Return

Return	Name	Type of data	Effective range	Initial value	Description
SMC_ClearFBError	Function return value	BOOL	—	—	Cleared successfully

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
pDrive	Axis variable pointer	POINTER TO AXIS_REF_SM3	—	—	Enter the axis address

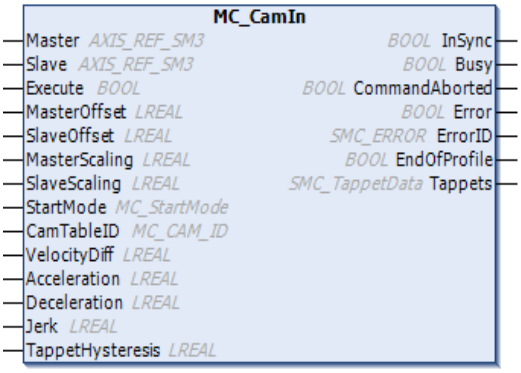
3）Function description

Example:TEST:=SMC_ClearFBError(pDrive:=ADR(AXIS1));

MC_CamIn

Execute cam movement according to the specified cam table. The phase offset, scaling ratio and working mode of the master and slave axes can be specified according to the needs。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_CamIn	Electronic cam function block		MC_CamIn(Master:= , Slave:= , Execute:= , MasterOffset:= , SlaveOffset:= , MasterScaling:= , SlaveScaling:= , StartMode:= , CamTableID:= , VelocityDiff:= , Acceleration:= , Deceleration:= , Jerk:= , TappetHysteresis:= , InSync=> , Busy=> , CommandAborted=> , Error=> ,

			ErrorID=> , EndOfProfile=> , Tappets=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Master	Spindle	AXIS_REF_SM3	—	—	Spindle mapping
Slave	Slave axis	AXIS_REF_SM3	—	—	Slave axis mapping

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start the function block when TRUE
Master Offset	Spindle offset	LREAL	Positive, negative, 0	0	Offset the spindle phase with the specified offset value
Slave Offset	Slave axis offset	LREAL	Positive, negative, 0	0	Offset the slave axis phase by the specified offset value
Master Scaling	Spindle factor	LREAL	Positive number	1	Spindle phase scaling factor
Slave Scaling	Slave axis coefficient	LREAL	Positive number	1	Slave axis phase scaling factor
StartMode		MC_Start Mode	0：absolute 1：relative 2：ramp_in 3:ramp_in _pos 4:ramp_in _neg	absolute	0: **absolute** absolute position 1: **relative** relative position 2：**ramp_in** Slope cut 3：**ramp_in_pos** Positive slope cut 4：**ramp_in_neg** Reverse ramp cut
CamTableID	Cam table ID	MC_CAM_ID	_	_	Used to identify the referenced cam table. It needs to be connected with the output parameter CamTableID of the corresponding MC_CamTableSelect function block instance.
VelocityDiff	speed	LREAL			Maximum speed in Ramp_in related mode
Acceleration	Acceleration	LREAL			Acceleration in Ramp_in related mode
Deceleration	decrease speed	LREAL			Deceleration in Ramp_in related mode
Jerk		LREAL			Ramp_in related mode time jump
Tappet Hysteresis	Lagging behind	LREAL			Damping coefficient of tappet

- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
InSync	Coupling completed	BOOL	TRUE/FALSE	FALSE	TRUE: Master and slave complete cam coupling
Busy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed and the cam is being coupled.
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The function block was interrupted by other instructions
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: An error occurred
ErrorID	error code	SMC_ERROR			Output fault code when an error occurs
EndOfProfile	Cam cycle completed	BOOL	TRUE/FALSE	FALSE	TRUE: The cycle is completed once, in aperiodic mode
Tappets	Tappet	SMC_ TappetData			Associated with a cam tappet, which can be read by MC_GetTappetValue instruction

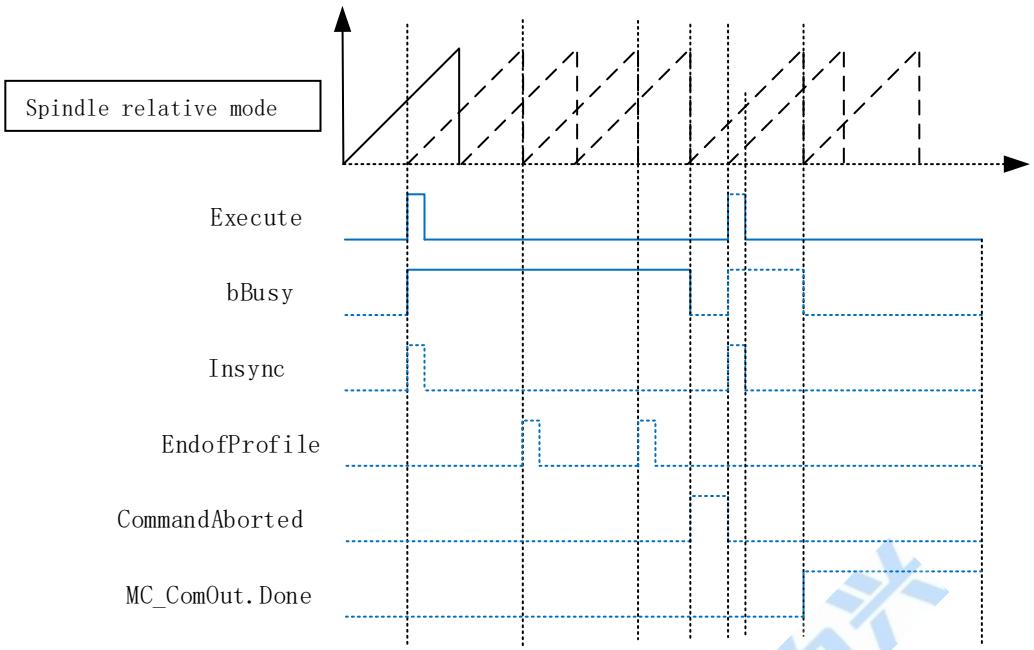
3）Function Description

- ✧ Execute rising edge, the axis does not report an error. If the cam table is selected correctly, this instruction will start.
- ✧ In a cam system, to call a cam curve, first call the MC_CamTzbleSelect instruction to select the corresponding cam table, and then execute MC_CamIn. If you want to change the cam curve, then call the MC_CamTzbleSelect instruction to reselect the cam table.
- ✧ The MC_CamOut instruction needs to be used to disconnect the cam coupling relationship between the master and slave axes.
- ✧ When the instruction is executed, when the instruction's slave axis executes other operation instructions, the cam relationship between the slave axis and the master axis will be cancelled, and the CommandAborted output will be TRUE.

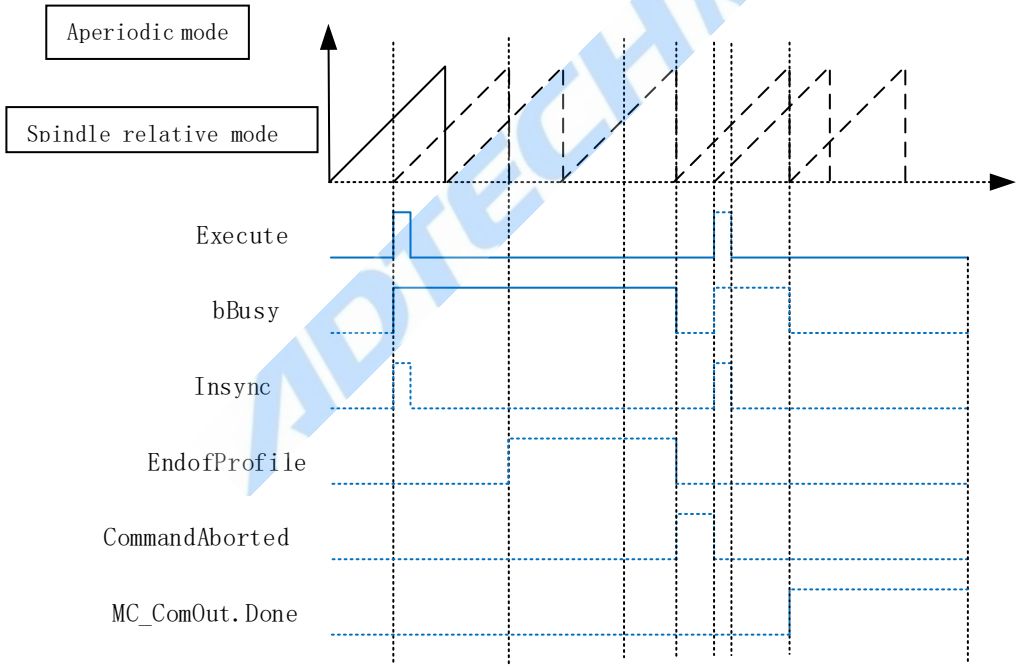
4）Timing diagram

Periodic mode (MC_CamTableSelect.Periodic is set to TRUE) as shown below:

Note: The MC_Camout instruction only cuts off the cam coupling relationship between the master and slave axes. If the speed of the slave axis is not 0 when disconnected, the slave axis will not automatically decelerate to 0, and the MC_STOP instruction must be used in conjunction.



Aperiodic mode (MC_CamTableSelect.Periodic is set to FALSE) as shown below:



Restart of the electronic cam:
Basically, the two electronic cams can be switched at any time, but some situations need to be considered. In the electronic cam editor, the position of the slave axis is defined as the calculation output of the electronic cam function. The electronic cam function is based on the range of the main axis. A spindle position is a calculation condition, which can be expressed by the following simple company: SlavePosition = CAM (MasterPosition) Because the actual cycle of the spindle drive is generally different from the spindle range defined by the electronic cam, so in order to satisfy the electronic cam For the correct

input of the function, the spindle position must be adjusted proportionally to the function domain: SlavePosition = CAM (MasterScale * MasterPosition + MasterOffset). Similarly, if an electronic cam is started in the absolute value mode, an upward jump occurs. The output of the function (that is, the position of the virtual slave axis) will also be corrected proportionally; SlavePosition=SlaveScale * CAM(MasterScale*Position)+SlaveOffset In the worst case, both of these scale corrections must be used. Therefore, in fact, the slave axis position (SlavePosition) is calculated by a more complicated formula;

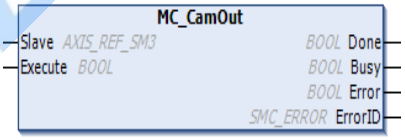
SlacePosition=SlaveScale * CAM(MasterScale * MasterPosition + MasterOffsset) +SlaveOffset

At the end of each electronic cam cycle, the ratio and offset can be changed to obtain more suitable parameters. Unfortunately, restarting the MC_CamIn module of the electronic cam will delete its memory and include the scale value and offset value. Therefore, the electronic cam function defined by the lock will adapt to various values of the slave axis. For this reason, it is recommended to restart the MC_CamIn function block only when a different electronic cam needs to be processed.

MC_CamOut

Disconnect the cam of the slave shaft from the coupling relationship. Note: After executing this instruction, the slave axis will continue to run at the speed before separation, so it needs to be used in conjunction with instructions such as MC_Stop。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_CamOut	Disengage cam coupling		MC_CamOut(Slave:= , Execute:= , Done=> , Busy=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Slave	Slave axis	AXIS_REF_SM3	—	—	Slave axis mapping

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start the function block when TRUE

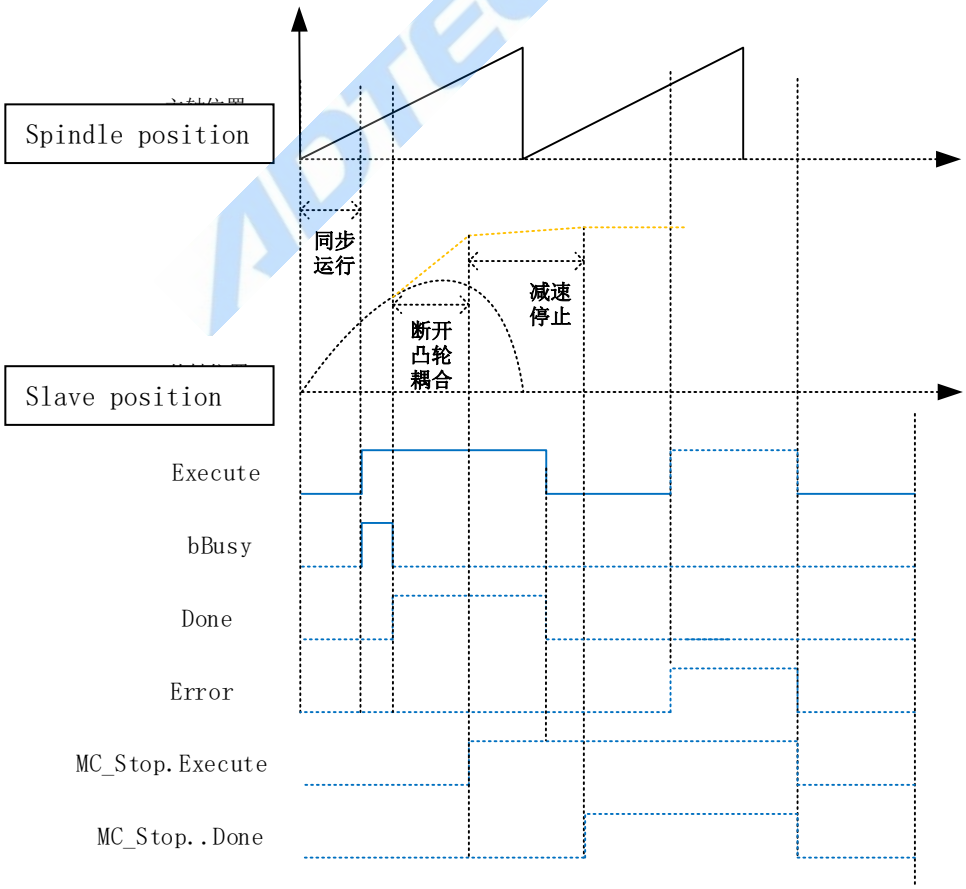
- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
Done		BOOL	TRUE/FALSE	FALSE	TRUE after the master and slave shafts release the gear coupling relationship
Busy		BOOL	TRUE/FALSE	FALSE	TRUE during command execution
Error		BOOL	TRUE/FALSE	FALSE	TRUE when an error occurs
ErrorID		SMC_ERROR			Output fault code when an error occurs

3）Function Description

- Execute this instruction to cancel the cam coupling relationship of the slave axis.
- The cam coupling relationship of the slave axis is disconnected at the rising edge of Execute.
- After the cam relationship is disconnected, the slave axis is not necessarily stopped.
- If the speed of the slave axis is not 0 before the instruction is executed again, the cam coupling relationship will be disconnected after the command DONE signal is completed, but the slave axis will still run at the speed before cutting out. If the slave axis does not execute the cam coupling relationship, ERROR will be output.

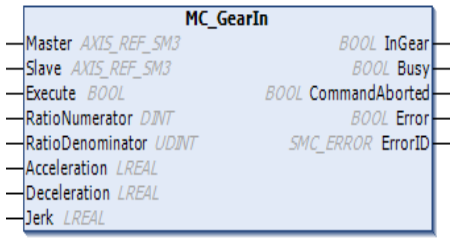
4）Timing diagram



MC_GearIn

Set the gear ratio between the slave axis and the master axis, and perform electronic gear operation.

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GearIn	Electronic gear function block		MC_GearIn(Master:= , Slave:= , Execute:= , RatioNumerator:= , RatioDenominator:= , Acceleration:= , Deceleration:= , Jerk:= , InGear=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Master	Spindle	AXIS_REF_SM3	—	—	Spindle mapping
Slave	Slave axis	AXIS_REF_SM3	—	—	Slave axis mapping

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
Ratio Numerator	Gear ratio numerator	DINT	Positive or negative	1	Gear ratio of master-slave gear movement
Ratio Denominator	Gear ratio denominator	UDINT	Positive number	1	molecular
Acceleration	Acceleration	LREAL	Positive number	0	Gear ratio of master-slave gear movement
Deceleration	decrease speed	LREAL	Positive number	0	Denominator
Jerk	Jump	LREAL	Positive number or 0	0	Specify the acceleration, the unit is

- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
InGear	Gear ratio reaches	BOOL	TRUE/FALSE	FALSE	TRUE: The slave axis reaches the target speed
Busy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The current command is being executed
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: When the current instruction is interrupted by another executing function block
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs

ErrorID	error code	SMC_ERRO R			Error code output when an error occurs

3）Function Description

After Execute detects the rising edge, the instruction will be executed and the electronic gear will start。

After executing the electronic gear, you must pass the GearOut command to decouple。

This command is a speed electronic gear function, and the loss of synchronization distance caused during acceleration will not be automatically compensated.

When the Busy signal is TRUE during the execution of the instruction, if the target speed of the slave axis is not reached, the new rising edge will not affect it.。

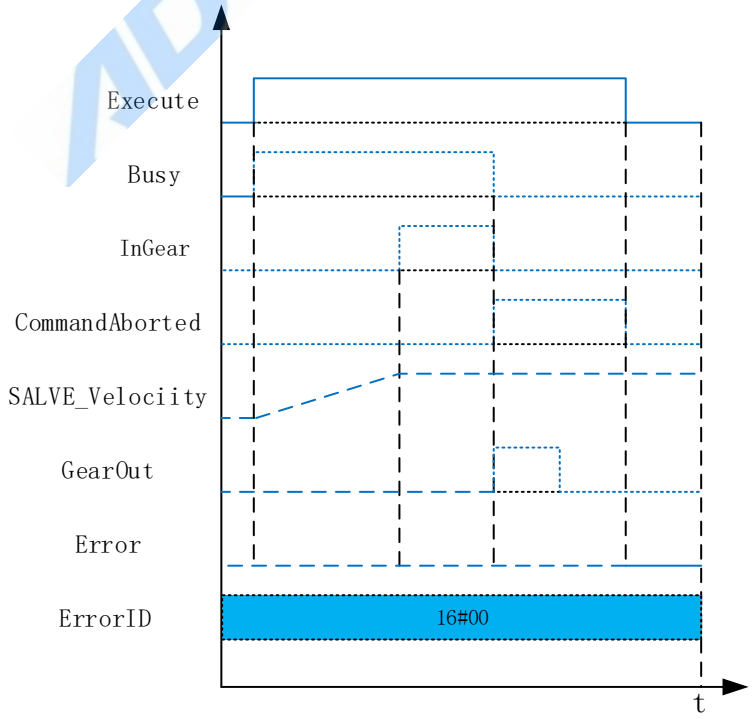
During the speed synchronization before execution, the lost position will not be compensated. After the speed synchronization reaches the InGear output TRUE, the master axis and the slave axis are satisfied：

Slave axis movement distance = master axis movement distance * RatioNumerator / RatioDenominator。

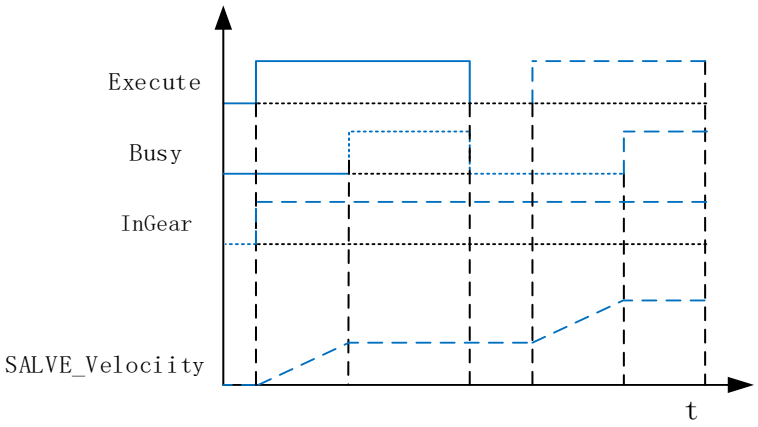
After the electronic gear coupling is executed, it must be decoupled by the MC_GearOut instruction before other instructions can obtain the control right of the slave axis.

The slave axis is prohibited from changing the position of the master master axis through the MC_Setposition command during the coupling process and after the coupling is completed. Because calling MC_Setposition to change the position of the Master spindle may cause the Slaver slave axis to track rapidly, and the motor speed will lose control. If you must change the position of the spindle, you should first call MC_GearOut to release the gear coupling motion, and then call the MC_Setposition instruction。

4）Timing diagram



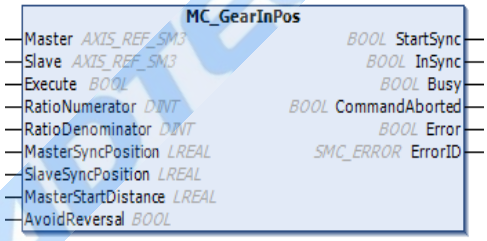
The sequence diagram of restart command after changing the gear ratio parameter is as follows:



MC_GearInPos

Set the electronic gear ratio between the slave shaft and the main shaft, and perform gear coupling motion。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GearInPos	Specify the position to cut into the electronics Gear coupling function		MC_GearInPos(Master:= , Slave:= , Execute:= , RatioNumerator:= , RatioDenominator:= , MasterSyncPosition:= , SlaveSyncPosition:= , MasterStartDistance:= , AvoidReversal:= , StartSync=> , InSync=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Master	Spindle	AXIS_REF_SM3	—	—	Spindle mapping
Slave	Slave axis	AXIS_REF_SM3	—	—	Slave axis mapping

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
Ratio Numerator	Gear ratio numerator	DINT	Positive or negative	1	Numerator of master-slave gear movement gear ratio
Ratio	Gear ratio	DINT	Positive or	1	Denominator of gear ratio

Denominator	denominator		negative		of master-slave gear movement
Master SyncPosition	Spindle synchronization position	LREAL		0	Spindle position when coupling is complete
Slave SyncPosition	Slave axis synchronization position	LREAL		0	Slave axis position when coupling is complete
MasterStart Distance	Spindle start synchronization distance	LREAL		0	
Avoid Reversal	Reversal is prohibited	BOOL	TRUE/FALSE	FALSE	When set to FALSE: the slave axis is allowed to reverse when the slave axis is leading. When set to TRUE: In the case where the physical structure does not allow, or cause danger, the slave station is not allowed to flip. (Only applicable to modal axis) If reverse rotation cannot be prohibited, the axis will stop by error

- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
StartSync	Start the coupling process	BOOL	TRUE/FALSE	FALSE	TRUE: start coupling processing
InSync	Coupling completed	BOOL	TRUE/FALSE	FALSE	TRUE: Coupling completed
Busy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The current command is being executed
Command Aborted	Execution interrupt	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: An error occurred
ErrorID	error code	SMC_ERROR			Error code output when an error occurs

3）Function Description

Specify the spindle position to start synchronization, start the synchronization distance from the axis position and the spindle, and use this to complete the cutting-in electronic cam action.

Execute rising edge signal, start to execute instructions。

After starting the operation, the Slave (slave axis) uses the Master (the speed obtained by multiplying the spindle speed by the gear ratio as the target speed, and performs acceleration and deceleration operations.)

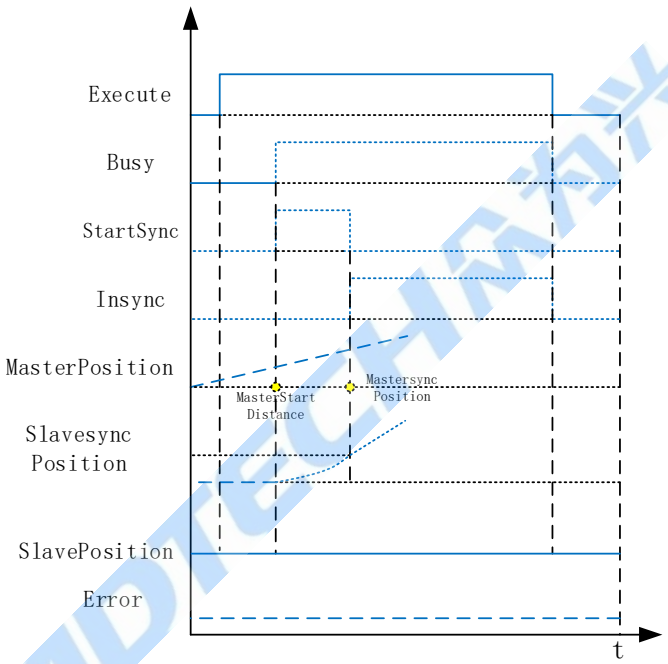
The process of synchronization from the start to the end of the function block is essentially an electronic cam that the slave axis follows the master axis in the synchronization interval. At this time, according to the master axis range (MasterSyncPostion-MasterStartDistance, MasterSyncPostion), the slave axis range (current position, SlaveSyncPostion), the command will be set according

to the setting The gear ratio and the above three parameters will automatically shoot a cam curve, and the slave axis will complete the cam action with which main axis during synchronization.。

Note: If the master and slave axis work in linear mode, the above parameters must be set reasonably, otherwise the gear action cannot be performed correctly. Therefore, it is recommended that the master and slave axis work in periodic mode when using this command。

For example: both the master and slave axis linear working mode want to move forward, if the master axis position>, MasterSyncPostion-MasterSyncPostion, or the slave axis position> SlaveSyncPostion when the instruction is executed, the electronic gear movement cannot be cut into。

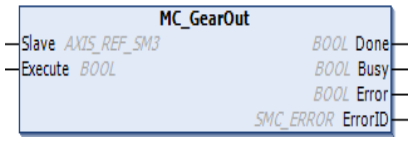
4）Timing diagram



MC_GearOut

Release the electronic gear coupling relationship between the slave axis and the master axis, and terminate the execution of MC_GearIn, MC_GearInPos instructions。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GearOut	Decoupling		MC_GearOut(Slave:= , Execute:= , Done=> , Busy=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Slave	Slave axis	AXIS_REF_SM3	—	—	Slave axis mapping

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE

- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Completion signal	BOOL	TRUE/FALSE	FALSE	TRUE: The master-slave shaft has released the gear coupling relationship
Busy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The current command is being executed
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: An error occurred
ErrorID	error code	SMC_ERROR			Output fault code when an error occurs

3）Function Description

Execute rising edge, execute cut-out electronic cam action。

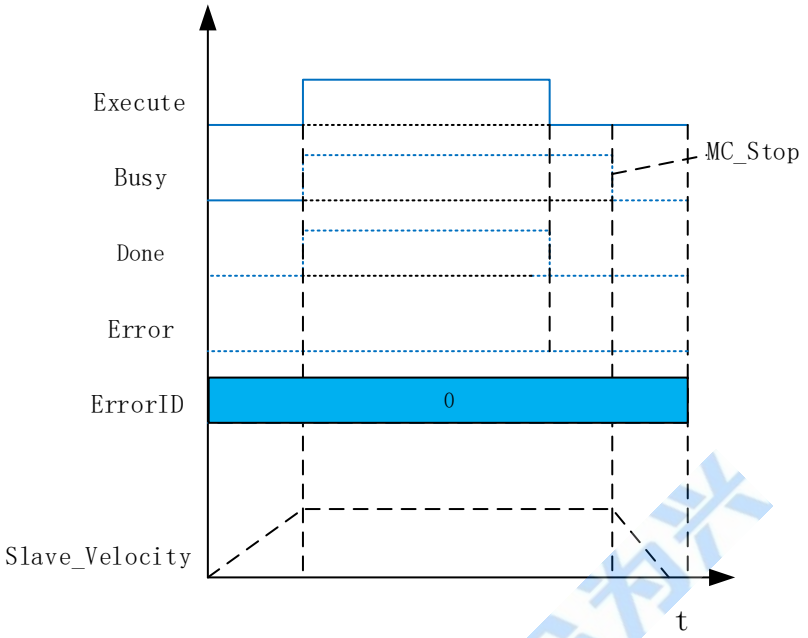
Eenable is TRUE, Error is FALSE, bBusy output is TRUE, Done output is TRUE。

After cutting off the electronic gear, the speed of the slave axis at this time is the familiar reading before cutting out, so it is necessary to stop the slave axis with the MC_Stop instruction。

Execute falling edge, Done output is FALSE。

There is MC_Stop instruction to execute reset Busy signal。

4）Timing diagram



5）Error description

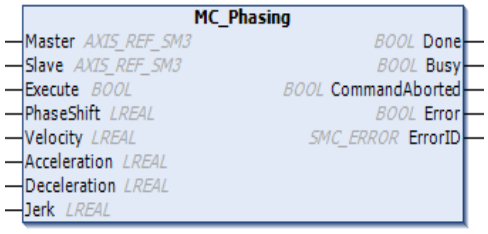
Wrong setting of related parameters will cause command alarm。

If the axis is not enabled, it will cause a command alarm。

MC_Phasing

Specify the phase deviation between the master and slave axis。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Phasing			MC_Phasing(Master:= , Slave:= , Execute:= , PhaseShift:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Master	Spindle	AXIS_REF_SM3	—	—	Spindle mapping
Slave	Slave axis	AXIS_REF_SM3	—	—	Slave axis mapping

- Input variable

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
PhaseShift	Phase deviation value	LREAL			The phase deviation value of the master and slave axis, a positive number indicates that the slave axis is lagging.
Velocity	speed	LREAL			Maximum speed value when performing phase shift
Acceleration	Acceleration	LREAL			Maximum acceleration value when performing phase shift
Deceleration	decrease speed	LREAL			Maximum deceleration value when performing phase shift
Jerk	Jump	LREAL			Maximum jump value when performing phase shift

- Output variable

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Completion signal	BOOL	TRUE/FALSE	FALSE	TRUE: Phase shift completed
Busy	In execution	BOOL			TRUE: The instruction is executing
CommandAborted		BOOL			TRUE: The current instruction was interrupted by another executing function block
Error	error	BOOL			TRUE: When an error occurs
ErrorID	error code	SMC_ERROR			Output fault code when an error occurs

3）Function Description

The phase shift is performed on the rising edge of Execute, and the slave axis automatically calculates a smooth curve to complete the phase shift from the slave axis to the master axis. The phase difference between the master and slave axis is the PhaseShift value of the output signal, and the positive value is after the slave axis and the master axis。

Done signal output is True after the offset is completed。

Compensate the phase difference between the master and slave axes according to the set PhaseShift, Velocity, Acceleration, and Deceleration. When the phase difference between the master and slave axis reaches PhaseShift, the Done signal is output。

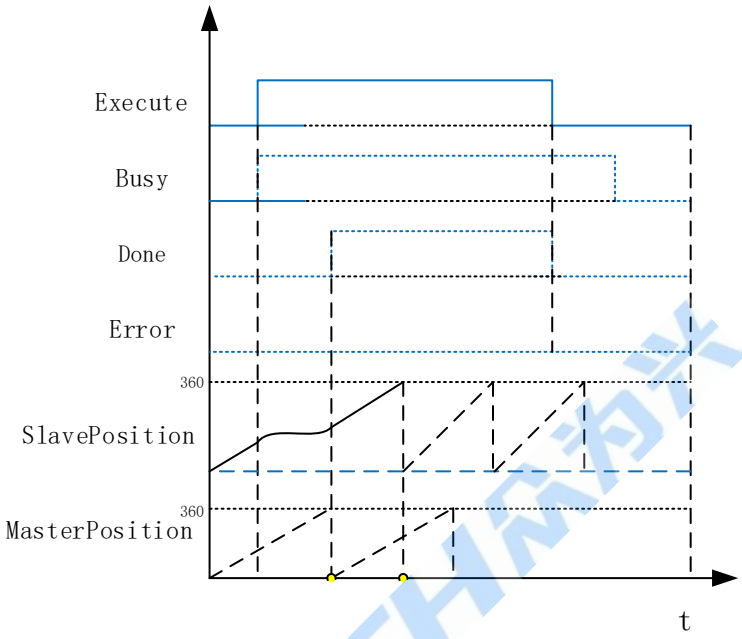
When the command is executed, the command position and feedback position of the master axis will not change, and the slave axis will be adjusted. After completion, the limit difference between the slave axis and the master axis will be PhaseShift。

The final result of this instruction is the phase offset between the given values of the axis, so the actual feedback value of the real axis may not be consistent with the final offset。

This instruction is used in conjunction with the MC_GearIn instruction, as follows: the master axis is Virtual_X, the slave axis is Virtual_Y, and the rising

edge of the Executive executes the spindle speed control and the electronic gear action of the master and slave axes, and then executes the phase shift. In addition, it can be used in conjunction with electronic cams. At this time, the slave axis is used as "electronic cam spindle" to achieve the effect of electronic cam spindle phase shift。

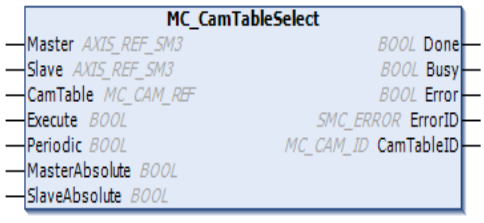
4）Timing diagrkam



MC_CamTableSelect

This function block is used to select the cam table by setting the connection to the related table. (Specify the cam table, used in conjunction with the MC_CamIn instruction)

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_CamTable Select			MC_CamTableSelect(Master:= , Slave:= , CamTable:= , Execute:= , Periodic:= , MasterAbsolute:= , SlaveAbsolute:= , Done=> , Busy=> , Error=> , ErrorID=> , CamTableID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Master	Spindle	AXIS_REF_SM3	—	—	Spindle mapping
Slave	Slave axis	AXIS_REF_SM3	—	—	Slave axis mapping

CamTable	Cam table	MC_CAM_REF	—	—	Map to CAM table description

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
Periodic	Repeat pattern	BOOL	TRUE/FALSE		Specify whether to execute the cam table repeatedly or only once TRUE: Repeat FALSE: no repetition
MasterAbsolute	Spindle absolute mode	BOOL	TRUE/FALSE		Specify whether the spindle tracking distance coordinate system is based on absolute position or relative position TRUE: absolute position FALSE: relative position
SlaveAbsolute	Slave absolute mode	BOOL	TRUE/FALSE		Synthesize with StartMode in the MC_CamIn instruction to specify whether the current position of the slave axis is the absolute value of the cam table output or relative. TRUE: absolute position FALSE: relative position
Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Completion signal	BOOL	TRUE/FALSE	FALSE	TRUE: Phase shift completed
Busy		BOOL			TRUE: The instruction is executing
Error	In execution	BOOL			TRUE: When an error occurs
ErrorID	error	SMC_ERROR			Output fault code when an error occurs
CamTableID	Effective CAMID	MC_CAM_ID			Select the effective CAMID

3）Function Description

This instruction specifies the cam table required for electronic cam operation, so you must edit the cam table (edit with cam editor or online) before using this instruction only.

Execute rising edge, execute the command cam table, or refresh the command cam table after the cam table is updated。

When the Done signal output is TRUE, the output variable "CamTableID" is generated and effective.

During instruction execution, Busy signal output TRUE, Done signal TRUE, Busy signal FLASE。

For the specific functions of the MasterAbsolute, SlaveAbsolute, and Periodic parameters, please refer to the MC_CamIn instruction details.

Precautions for use:

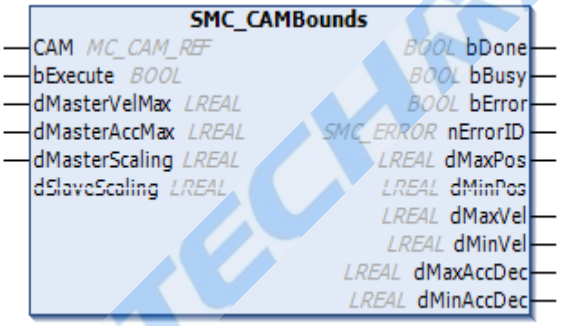
The master axis and slave axis cannot be designated as the same axis, otherwise there will be an error output, and the cam table corresponding to

CamTable must be edited correctly, otherwise it will also cause an instruction error. The master axis and slave axis can be real or imaginary。
Improper selection of MasterAbsolute and SlaveAbsolute may cause the electronic cam output to jump, so please make sure to set the working mode of the cam curve before setting。

SMC_CAMBounds

When the slave axis is coupled with the master cam, the maximum position, speed, and acceleration of the slave axis can be calculated through this function block. The spindle moves under the input maximum speed and acceleration/deceleration limit. This instruction can check whether the cam table is correct when designing the cam table, provided that the maximum acceleration and deceleration, speed, etc. of the spindle are known。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_ CAMBounds	Cam upper and lower limit		SMC_CAMBounds(CAM:= , bExecute:= , dMasterVelMax:= , dMasterAccMax:= , dMasterScaling:= , dSlaveScaling:= , bDone=> , bBusy=> , bError=> , nErrorID=> , dMaxPos=> , dMinPos=> , dMaxVel=> , dMinVel=> , dMaxAccDec=> , dMinAccDec=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
CAM	Cam	MC_CAM_REF	—	—	Map to cam

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
dMasterVelMax	Maximum speed	LREAL		1	Maximum spindle speed in absolute mode
dMasterAccMax	Maximum acceleration	LREAL		0	Maximum spindle acceleration in absolute mode
dMasterScaling	Scale factor	LREAL			Scale factor in spindle cam application
dSlaveScaling	Scale factor	LREAL			Scale factor in slave cam application

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Completion signal	BOOL	TRUE/ FALSE	FALSE	TRUE: Phase shift completed
Busy	Command processing	BOOL			TRUE: The instruction is executing
Error	error	BOOL			TRUE: When an error occurs
ErrorID	error code	SMC_ERROR			Output fault code when an error occurs
dMaxPos	Maximum position	LREAL			Calculate the maximum position of the slave axis according to the cam table
dMinPos	Minimum position	LREAL			Calculate the minimum position of the slave axis according to the cam table
dMaxVel	Maximum speed	LREAL			Calculate the maximum speed of the slave axis
dMinVel	Minimum speed	LREAL			Calculate the minimum speed of the slave axis
dMaxAccDec	Maximum acceleration	LREAL			Calculate the maximum acceleration of the slave axis [u/s²]
dMinAccDec	Minimum acceleration	LREAL			Calculate the minimum acceleration of the slave axis [u/s²]

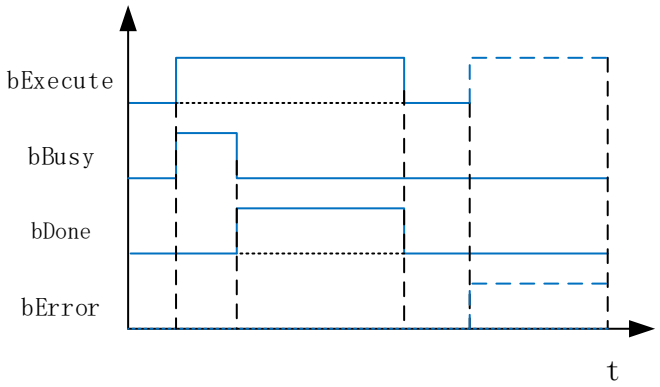
3）Function Description

The rising edge of bExecute integrates the "dMasterVelMax", "dMasterAccMax", "dMasterScaling", "dSlaveScaling" values of the input variables and the cam table data to calculate the slave axis "maximum position", "minimum position" and other values。

This function block calculates the maximum position, speed and acceleration value of the slave axis coupled with the master axis in absolute mode, and the master axis moves according to the specified maximum speed and acceleration/deceleration。

This module is especially useful when creating and modifying cam discs in online mode and the compliance with the maximum value must be checked in advance。

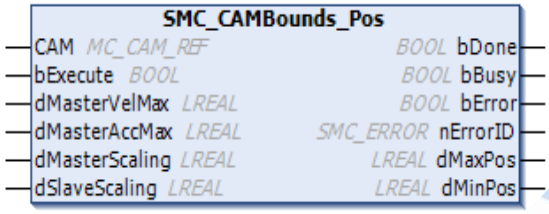
4）Timing diagram



SMC_CAMBounds_Pos

When the slave axis is coupled with the master cam, the maximum and minimum positions of the slave axis can be calculated through this function block. Compared with SMC_CAMBounds, this function block has less calculation of maximum acceleration and other functions.

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_CAMBounds_Pos	Upper and lower limit of cam position		SMC_CAMBounds_Pos(CAM:= , bExecute:= , dMasterVelMax:= , dMasterAccMax:= , dMasterScaling:= , dSlaveScaling:= , bDone=> , bBusy=> , bError=> , nErrorID=> , dMaxPos=> , dMinPos=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
CAM	Cam	MC_CAM_REF	—	—	Map to cam

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
dMasterVelMax	Maximum speed	LREAL		1	Maximum spindle speed in absolute mode
dMasterAccMax	Maximum acceleration	LREAL		0	Maximum spindle acceleration in absolute mode
dMasterScaling	Scale factor	LREAL			Scale factor in spindle cam application
dSlaveScaling	Scale factor	LREAL			Scale factor in slave cam application

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Completion signal	BOOL	TRUE/FALSE	FALSE	TRUE: Phase shift completed
Busy	Command processing	BOOL			TRUE: The instruction is executing
Error	error	BOOL			TRUE: When an error occurs
ErrorID	error code	SMC_ERROR			Output fault code when an error occurs
dMaxPos	Maximum position	LREAL			Calculate the maximum position of the slave axis according to the cam table
dMinPos	Minimum position	LREAL			Calculate the minimum position of the slave axis according to the cam table

3）Function Description

The rising edge of bExecute integrates the "dMasterVelMax", "dMasterAccMax", "dMasterScaling", "dSlaveScaling" values of the input variables and the cam table data to calculate the slave axis "maximum position", "minimum position" and other values。

The spindle runs in absolute mode or the spindle is set to cycle mode, and the modulus is set to spindle cycle can be calculated by this instruction。

Cam table is XYVA (valid in polynomial mode), one-dimensional array, two-dimensional array, etc. are invalid

Contrary to SMC_CAMBounds, this function block only calculates the maximum and minimum position values of the slave axis coupled with the master axis in absolute mode, and the master axis moves according to the specified maximum speed and acceleration/deceleration.

This module is especially useful when the cam table is created and modified in online mode and the compliance with the maximum value must be checked in advance。

4）Timing diagram

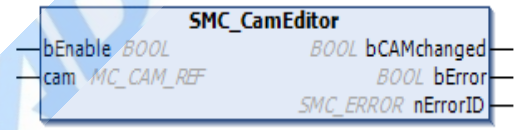
5）Error description

The cam table format is not a polynomial mode; the set value of the cam table MC_CAN_REF does not match the actual cam table。

SMC_CamEditor

This function block is used to display the cam table in the visualization。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_ CamEditor			SMC_CamEditor(bEnable:= , bCAMchanged=> , bError=> , nErrorID=> , cam:=);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
CAM	Cam	MC_CAM_REF	—	—	Map to cam

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bEnable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE

- Output variables

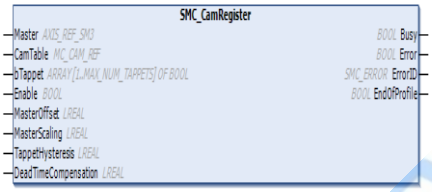
Output variable	Name	Type of data	Effective range	Initial value	Description
bCAMchanged	Completion signal	BOOL	TRUE/FALSE	FALSE	TRUE: Phase shift completed

Error	error	BOOL			TRUE: When an error occurs
ErrorID	error code	SMC_ERROR			Output fault code when an error occurs

SMC_CamRegister

This realizes the cam lifter control (cam switch). It is not necessary to edit the master and slave axis curve when editing the cam, just configure the tappet table to realize tappet control through this function block。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_ CamRegister			SMC_CamRegister(Master:= , CamTable:= , bTappet:= , Enable:= , MasterOffset:= , MasterScaling:= , TappetHysteresis:= , DeadTimeCompensation:= , Busy=> , Error=> , ErrorID=> , EndOfProfile=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Master	Spindle	AXIS_REF_SM3	—	—	Map to axis
CamTable	Cam table	MC_CAM_REF			Map to cam
bTappet	Tappet output	ARRAY [1..MAX_NUM_TAPPETS] OF BOOL			Tappet output

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
MasterOffset	Spindle offset	LREAL			Spindle offset
MasterScaling	Spindle ruler	LREAL			Spindle linear scaling factor
TappetHysteresis	Tappet damping	LREAL			Tappet control damping coefficient
DeadTimeCompensation	Dead time compensation	LREAL			The dead zone compensation time is: second (S), the tappet output is linearly compensated according to the current speed of the spindle, which can be a positive value or a negative value.

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Busy	In execution	BOOL	TRUE/FALSE		TRUE: The instruction is executing
Error	error	BOOL	TRUE/FALSE		TRUE: When an error occurs
ErrorID	error code	SMC_ERROR			Output fault code when an error occurs
EndOfProfile	Curve cycle completed	BOOL	TRUE/FALSE		TRUE: The spindle position is greater than or equal to the set period

3）Function Description

The Enable signal is TRUE. If there is no error output, the Busy output is TRUE, and the tappet control is executed.

The control function block has nothing to do with the slave axis in the electronic cam, just configure the spindle cycle and tappet table。

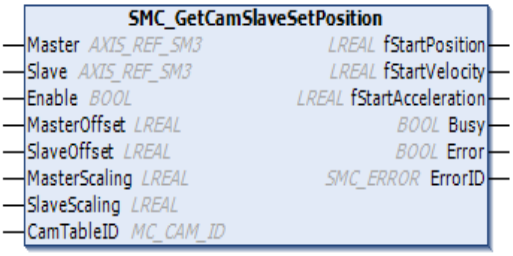
"BTappet" is a one-dimensional Boolean structure (MAX_NUM_TAPPETS=512), and the output DeadTimeCompensation unit of bTappet[i] to the i-th tappet point is S/sec. When it is set to a negative value, the tappet signal will be delayed. For example, if the Ethcat duty cycle is set to 0.02 seconds and 4ms, the tappet output position is P according to the spindle linear speed v, and the tappet output value is at the spindle setting position=P-V*0.02. On the contrary, if it is set to -0.02 seconds, the spindle setting position is greater than or equal to P and the tappet signal output is delayed by five cycles。

The function block represents a tappet control unit, and works on MC_CAM_REFnegating original path information and only read tappet information (such as MC_CamIn)。

SMC_GetCamSlaveSetPosition

Read cam table slave axis position, speed, acceleration information。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_GetCamSlaveSetPosition	Get cam slave position		SMC_GetCamSlaveSetPosition(Master:= , Slave:= , Enable:= , MasterOffset:= , SlaveOffset:= , MasterScaling:= , SlaveScaling:= , CamTableID:= , fStartPosition=> , fStartVelocity=> , fStartAcceleration=> , Busy=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Master	Spindle	AXIS_REF_SM3	—	—	Map to main axis
Slave	Slave axis	AXIS_REF_SM3	—	—	Map to slave axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
MasterOffset	Spindle offset	LREAL			Spindle offset
SlaveOffset	Offset from the axis	LREAL			Offset from the axis
MasterScaling	Spindle zoom	LREAL		1	Spindle zoom factor
SlaveScaling	Zoom from axis	LREAL		1	Slave scaling factor
CamTableID	Cam ID	MC_CAM_ID			Cam table ID

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
fStart Position	Slave position	LREAL			The slave axis position obtained according to the cam table and the current master axis information
fStart Velocity	Slave speed	LREAL			The slave axis speed obtained according to the cam table and the current master axis information
fStart Acceleration	Slave acceleration	LREAL			The acceleration of the slave axis obtained according to the cam table and the current master axis information
Busy	In execution	BOOL	TRUE/FALSE		TRUE: The command is being executed
Error	error	BOOL	TRUE/FALSE		TRUE: When an error occurs
ErrorID	error code	SMC_ERROR			Output fault code when an error occurs

3）Function Description

This function block calculates the current target position of one axis (slave axis) when one axis is coupled by the movement of another axis (master axis). Will not move the slave axis or affect it in other ways。

This module can be used to determine the starting position, speed and acceleration of the follower coupled to the main shaft through the cam table. Since the corresponding value is calculated in one cycle, there is no need to output。

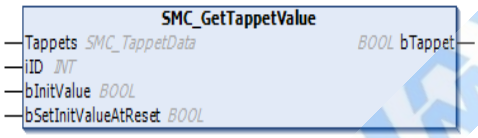
Change the output value calculated by the instruction to：Y=（cam((Cam start spindle look-up table position +Masteroffsrt)*MasterScaling)+slaveoffset）*SlaveScaling，Cam 为 Cam

table function。For example: the starting spindle position of the cam is 0, the scaling of the master and slave axes is 1, the Masteroffset is 100, and the slaveiffset is 0, then the output of the function block is the position of the slave corresponding to the cam at 100。
This function block reads the position of the slave axis, only the cam table is constructed successfully, and there is no requirement for the operation of the master and slave axes。

SMC_GetTappetValue

Used in conjunction with MC_CamIn instruction to get the current tappet output value。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_ GetTappetV alue	Get tappet output value		SMC_GetTappetValue(Tappets:= , iID:= , bInitValue:= , bSetInitValueAtReset:= , bTappet=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Tappets	Tapp et	SMC_TappetData	—	—	Mapped to tappet

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
iID	Tappet group number	INT	TRUE/FALSE	0	ID of the tappet group
bInitValue	Initial value	BOOL			The tappet initialization value when the function block is called for the first time
bSetInitValueAtR eset		BOOL			**TRUE**: Set the value of **tappet** to **bInitValue** when restarting **MC_CamIn**' function block. **FALSE**: Keep the **tappet** value when restarting the **MC_CamIn** function block.

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
bTappet	Tappet output	BOOL		FALSE	Tappet value

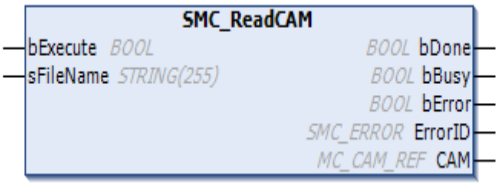
3）Function Description

This function block calculates the output tappet of the function block MC_CamIn and contains the current tappet state.

This function block is the same as the SMC_CamRegister function to read the tappet output, but there is a conflict between the two, so only one type can be used in the same cam tappet table。

SMC_ReadCAM

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_ReadCAM			SMC_ReadCAM(bExecute:= , sFileName:= , bDone=> , bBusy=> , bError=> , ErrorID=> , CAM=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Tappets	Tappet	SMC_TappetData	—	—	Mapped to tappet

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Tappet group number	INT	TRUE/FALSE	0	Trigger on rising edge, start execution condition when TRUE
sFileName	file name	STRING(255)			File name, which contains the defined cam description in ASCII format.

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
bDone	Tappet output	BOOL		FALSE	TRUE: The instruction was executed successfully
bBusy	In execution	BOOL		FALSE	TRUE: The command is being executed
bError	error	BOOL		FALSE	TRUE: When an error occurs
ErrorID	error code	SMC_ERROR			Output fault code when an error occurs
CAM		MC_CAM_REF		STRUCT(nElements := 0, nTappets := 0)	

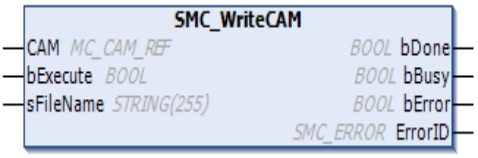
3）Function Description

This function block is designed to load cam at runtime and make it available to MC_CamTableSelect and MC_CamIn modules. The loaded cam must be created in the cam editor and saved in "*". CAM file (see SMC_WriteCAM). The size of the loadable cam is limited by the global constants gc_SMC_FILE_MAXCAMEL (number of elements) and gc_SMC_FILE_MAXCAMTAP (number of cam switching operations).

SMC_WriteCAM

It is used to store the edited cam table as a file when the program is running. It can be used by instructions such as MC_CamIn. The generated file contains content refer to "Cam Format"。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_ WriteCAM			SMC_WriteCAM(CAM:= , bExecute:= , sFileName:= , bDone=> , bBusy=> , bError=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
CAM	Cam	MC_CAM_REF	—	—	Map to cam

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Instruction execution	BOOL	TRUE/FALSE	0	Trigger on rising edge, start execution condition when TRUE
sFileName	file name	STRING(255)			File name, which contains the defined cam description in ASCII format.

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
bDone	Tappet output	BOOL	TRUE/FALSE	FALSE	TRUE: CAM table has been written
bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The command is being executed
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	error code	SMC_ERROR			Output fault code when an error occurs

3）Function Description

This function block is used to store the cam table created in the cam editor into a file at runtime. This command can be used in conjunction with SMC_ReadCAM。

bExecute rising edge, the instruction is executed. Store the cam information connected with "CAM" to the file connected with the file name "sFileName"。

The storage is successful and the bDone signal output is TRUE。

Note: When this function is executed during program operation, the cam table information can also be manually stored in offline information。

4）Error description

This command can only complete the cam table of XYVA multiple modes, one-dimensional, two-dimensional, etc. will cause error output sFileName The connected file name does not exist or the information is wrong。

SMC_FollowPosition

The command function is to directly set the position of the axis without any inspection. After executing the rising edge signal of bExecute, the axis position command will be given every task cycle regardless of the status of the axis。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_ FollowPosition	Axis position setting		SMC_FollowPosition(Axis:= , bExecute:= , fSetPosition:= , bBusy=> , bCommandAborted=> , bError=> , iErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Axis mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
fSetPosition	Set location	LREAL			Axis setting position

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: During the execution of the instruction, the axis is in the synchronization state at this time, which is the same as the axis state when the cam MC_CamIn is running. Use MC_CamOut to clear bBusy status
bCommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
iErrorID	error code	SMC_ERROR			Output fault code when an error occurs

3）Function Description

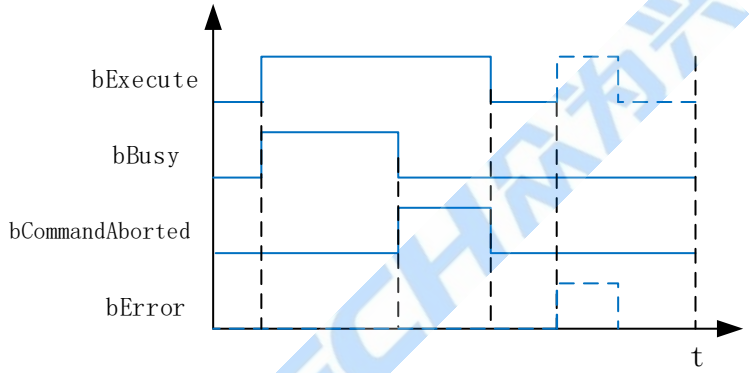
After SMC_FollowPosition is started by the rising edge of bExecute, the axis will send a position command to the axis every task cycle。

When the bBusy signal output is TRUE, the state of the axis is synchronous operation and the state of the slave axis when the MC_CamIn instruction is effective, and can be cleared by the MC_CamOut instruction。

Speed of the axis: calculated by the position increment between the two task periods of the axis, the speed ΔL/Δt (ΔL: the difference between fSetPosition in this task period and fSetPosition in the previous task period; Δt is the scan time)。

When the bExecute signal is TRUE, bBusy will change from TRUE to FALSE when other control commands interrupt the command。

4）Timing diagram



5）Error description

At the rising edge of bExecute:

Axis variable is connected to non-AIS_REF_SM3 type structure variable, Error output;

The axis is not enabled, Error output。

During command operation, axis error, Error output。

SMC_FollowPositionVelocity

The command function is the same as the SMC_FollowPosition use and function, but the speed setting is added。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_FollowPositionVelocity	Axis position setting		SMC_FollowPositionVelocity(Axis:= , bExecute:= , fSetPosition:= , fSetVelocity:= , bBusy=> , bCommandAborted=> , bError=> , iErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
fSetPosition	Set location	LREAL			Axis setting position
fSetVelocity	Set speed	LREAL			Axis set speed

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
bBusy	In execution	BOOL	TRUE/ FALSE	FALSE	TRUE: During the execution of the instruction, the axis is in the synchronization state at this time, which is the same as the axis state when the cam MC_CamIn is running. Use MC_CamOut to clear bBusy status
bCommandAborted	Instruction interrupted	BOOL	TRUE/ FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block
bError	Error	BOOL	TRUE/ FALSE	FALSE	TRUE: When an error occurs
iErrorID	Error code	SMC_ERROR			Output fault code when an error occurs

3）Function Description

After SMC_FollowPositonVelocity is started by the rising edge of bExecute, the axis will send a set position and set speed command to the axis every task cycle。

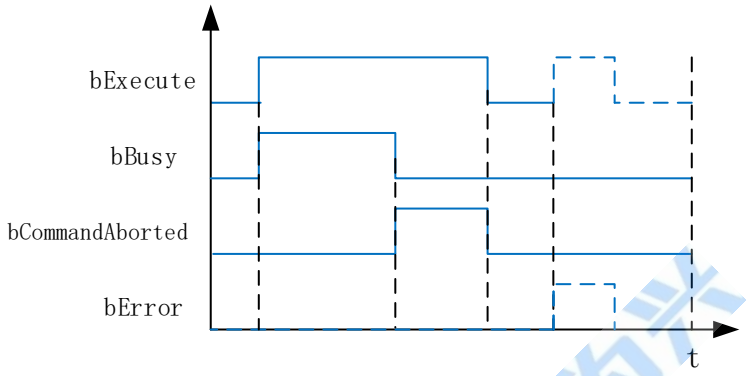
When the bBusy signal output is TRUE, the state of the axis is synchronous operation and the state of the slave axis when the MC_CamIn instruction is effective, and can be cleared by the MC_CamOut instruction。

When the bExecute signal is TRUE, bBusy will change from TRUE to FALSE when other control commands interrupt the command.。

Note:

The speed setting must meet the position setting change, that is: speed setting = the first derivative of the position setting difference with respect to time in the task period. For example: if the position settings are the same for two periods apart, the speed should be set to 0, otherwise it will cause severe vibration of the motor。

4）Timing diagram



5）Error description

At the rising edge of bExecute:

Axis variable is connected to non-AIS_REF_SM3 type structure variable, Error output;

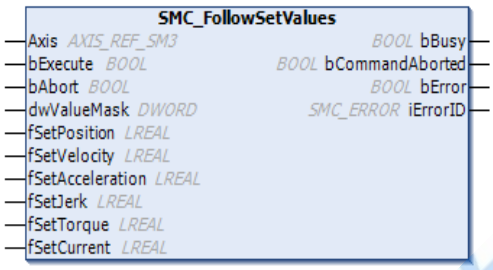
If the axis is not enabled, Error will be output.

When the instruction is running, an axis error occurs and Error is output.

SMC_FollowSetValues

The command function is the same as the use and function of SMC_FollowPosition, but the speed, acceleration, jump, current, and torque settings are added. Use dwValueMask to select the required settings。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_FollowSetValues	Axis position setting		SMC_FollowSetValues(Axis:= , bExecute:= , bAbort:= , dwValueMask:= , fSetPosition:= , fSetVelocity:= , fSetAcceleration:= , fSetJerk:= , fSetTorque:= , fSetCurrent:= , bBusy=> , bCommandAborted=> , bError=> , iErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
bAbort	Abort execution	BOOL	TRUE/FALSE	FALSE	Abort the execution of the function block (For example, to avoid errors when restarting a different axis)
dwValueMask	Control management	DWORD			Setting options: TRUE is activated, FALSE is closed. Bit0:fSetPosition (TRUE is activefSetPosition FALSE is closedfSetPosition The same as below） Bit1:fSetVelocity Bit2:fSetAcceleration Bit3:fSetJerk Bit4:fSetTorque Bit5:fSetCurrent
fSetPosition	Set location	LREAL			Position value of axis setting [u]
fSetVelocity	Set speed	LREAL			Speed value of axis setting [u/s]
fSetAcceleration	Set acceleration	LREAL			Acceleration value set by axis [u/s²]
fSetJerk	Set hop	LREAL			Jump value of axis setting [u/s³]

fSetTorque	Set torque	LREAL			Torque value of axis setting [Nm]/[N]
fSetCurrent	Set current	LREAL			Current value of axis setting [A]

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The command is being executed At this time, the axis is in a synchronized state, which is the same as the axis state when the cam MC_CamIn is running. Use MC_CamOut to clear bBusy status
bCommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
iErrorID	error code	SMC_ERROR			Output fault code when an error occurs

3）Function Description

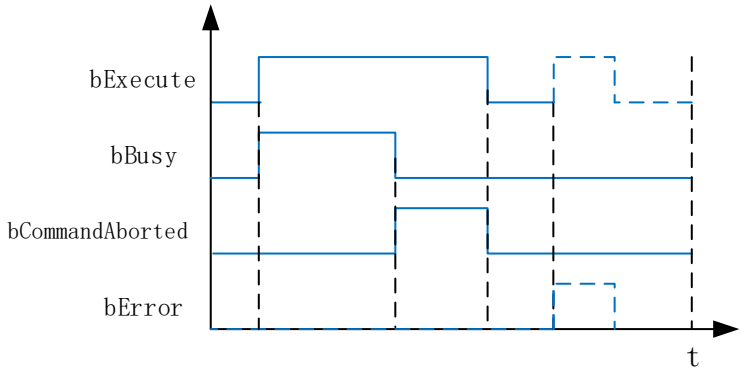
After SMC_FollowSetValues is started by the rise of bExecute for one year, the controller will send the selected parameter commands to the axis every task cycle。

When the bBusy signal output is TRUE, the state of the axis is synchronous operation and the state of the slave axis when the MC_CamIn instruction is effective, and can be cleared by the MC_CamOut instruction。

When the bExecute signal is TRUE, bBusy will change from TRUE to FALSE when other control commands interrupt the command。

Select the control parameter through the dwValueMask value. For example, if dwValueMask is 1, the position parameter sent to the axis in each task cycle has the same function as the SMC_FollowPosition instruction. If dwValueMask is 2, it is a single speed command output. When dwValueMask is 3, it is the position speed command output. When dwValueMask is 7, it is the position, velocity, acceleration command output, etc。

4）Timing diagram



5）Error description

At the rising edge of bExecute:

Axis variable is connected to non-AIS_REF_SM3 type structure variable, Error output;

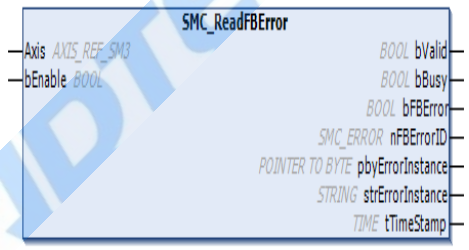
If the axis is not enabled, Error will be output.

When the instruction is running, an axis error occurs and Error is output.

SMC_ReadFBError

This function block can be used to read function block errors。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_ReadFBError	Read function block error		SMC_ReadFBError(Axis:= , bEnable:= , bValid=> , bBusy=> , bFBError=> , nFBErrorID=> , pbyErrorInstance=> , strErrorInstance=> , tTimeStamp=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	TRUE: start to execute the function block, read

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Valid	Effective state	LREAL			TRUE: The function block output parameter is valid

bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The command is being executed
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: An error occurred in this function
nErrorID	error code	SMC_ERROR	Refer to SMC_ERROR	0	Output fault code when an error occurs
pbyErrorInstance		POINTER TO BYTE			Point to the instance of the function block reporting the error
strErrorInstance		STRING			
tTimeStamp		TIME			Timestamp when the error occurred

3）Function Description

✧ Eenable is TRUE, Valid if there is no error, bBusy output is TRUE. If there is a function block alarm, bError output is TRUE。

✧ Eenable becomes FALSE, then Valid, bBusy output is FALSE。

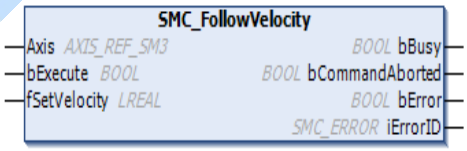
4）Error description

bExecute rising edge, axis error, Error output; invalid axis input, Error output。

SMC_FollowVelocity

This function block writes the set speed to the axis without any check. After executing the rising edge signal of bExecute, the axis speed command will be given every task cycle。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_FollowVelocity	Shaft speed reference		SMC_FollowVelocity(Axis:= , bExecute:= , fSetVelocity:= , bBusy=> , bCommandAborted=> , bError=> , iErrorID=>);

2）Related variables

● Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

● Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE

fSetVelocity	Set speed	LREAL			Speed value of axis setting [u/s]

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The command is being executed At this time, the axis is in a synchronized state, which is the same as the axis state when the cam MC_CamIn is running. Use MC_CamOut to clear bBusy status
bCommand Aborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
iErrorID	Error code	SMC_ERROR			Output fault code when an error occurs

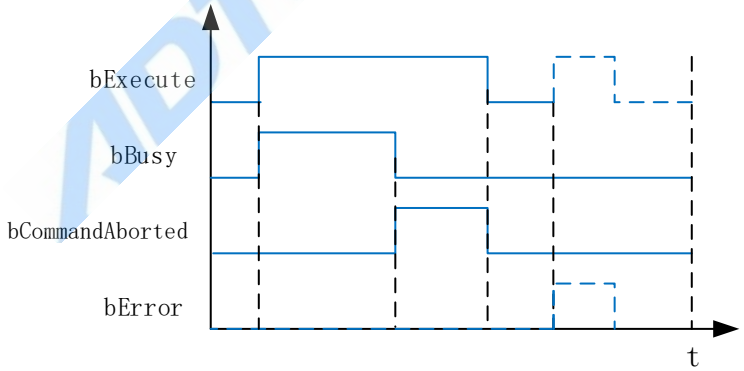
3）Function Description

After SMC_FollowVelocity is started by the rising edge of bExecute, the controller will send a speed command to the axis every task cycle. (MC_MoveVelocity command speed must be re-triggered to take effect after the speed is changed).

When the bBusy signal output is TRUE, the state of the axis is synchronous operation and the state of the slave axis when the MC_CamIn instruction is effective, and can be cleared by the MC_CamOut instruction.

When the bExecute signal is TRUE, bBusy will change from TRUE to FALSE when other control commands interrupt the command.

4）Timing diagram



5）Error description

At the rising edge of bExecute:

Axis variable is connected to non-AIS_REF_SM3 type structure variable, Error output;

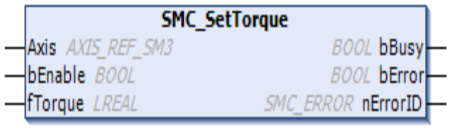
If the axis is not enabled, Error will be output.

When the instruction is running, an axis error occurs and Error is output.

SMC_SetTorque

Set the shaft torque. (Valid in torque control mode)

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_ SetTorque	Shaft torque setting		SMC_SetTorque(Axis:= , bEnable:= , fTorque:= , bBusy=> , bError=> , nErrorID=>);

2）Related variables

● Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

● Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bEnable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
fTorque	Set torque	LREAL			Unit: 0.1% (Nm)

● Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The command is being executed
bCommand Aborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	TRUE: The current instruction was interrupted by another executing function block
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
iErrorID	error code	SMC_ERRO R			Output fault code when an error occurs

3）Function Description

bEnable rising edge, if there is no error, bBusy output is TRUE。
This command only sets the torque value for the week, not the feat control function, the axis control mode is valid in the torque control mode。

4）Error description

5）Error description

At the rising edge of bExecute;
Axis error, Error output; invalid axis input, Error output;
Axis control mode error, Error output, error code SMC_ST_WRONG_CONTROLLER_MODE。

MC_AbortTrigger

The function block interrupts the correlation characteristics of input latch related events, and is used in conjunction with MC_TouchProbe。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_ AbortTrigger	Function block terminati on event correlatio n		MC_AbortTrigger(Axis:= , TriggerInput:= , Execute:= , Done=> , Busy=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis
TriggerInput	Trigger signal	TRIGGER_REF	—	—	Description of trigger signal, trigger attribute, etc.

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Setting operation is successful	BOOL	TRUE/FALSE	FALSE	TRUE: The setting operation is successful
Busy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The command is being executed
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERRO R		0	Output fault code when an error occurs

3）Function Description

The MC_AbortTrigger function block is to terminate the associated operation of the trigger signal or attribute and the related trigger instruction.

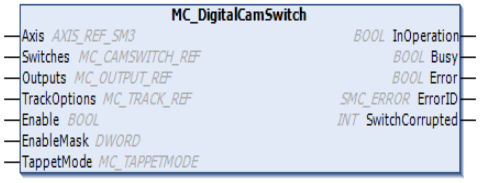
The Execute of the function block must be a rising edge trigger condition;

The Done of the function block indicates that the setting operation is successful;

The function block Busy indicates that the current function is being executed.

MC_DigitalCamSwitch

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_DigitalCamSwitch			MC_DigitalCamSwitch(Axis:= , Switches:= , Outputs:= , TrackOptions:= , Enable:= , EnableMask:= , TappetMode:= , InOperation=> , Busy=> , Error=> , ErrorID=> , SwitchCorrupted=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis
Switches		MC_CAMSWITCH_REF			Switch action reference
Outputs		MC_OUTPUT_REF			Reference signal output directly related to the reference trajectory
TrackOptions		MC_TRACK_REF			Reference to structures that contain track-related attributes, for example. On/off compensation for each output/track

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
EnableMask					
TappetMode					

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
InOperation	Setting operation is successful	BOOL	TRUE/FALSE	FALSE	TRUE: Command track is enabled
Busy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The command is being executed
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs
SwitchCorrupted		INT			Signal processing for potential problems caused by compensation time:

					Compensation time may cause the extrapolated position to pass through one switching point several times before passing another switching point. Therefore, a non-inverted switch may become an inverted switch (and vice versa). In this example, this variable indicates the switch involved. There are two possible reasons: Noisy speed value, The "huge" difference between switch compensation. -1 means that this behavior did not happen

3）Function Description

This function block is similar to opening a motor shaft: the function block commands a set of discrete output bits to switch, similar to a set of mechanical cam control switches connected to the shaft. Allow forward and backward movement。

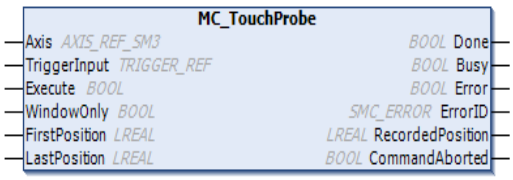
Changing the var_in_out parameter requires enabling a new rising edge of the input。

This function is sometimes called partial least squares phase or position or programmable limit switch。

MC_TouchProbe

The function block is triggered by an external signal to save the position data of the current axis。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_ TouchProbe			MC_TouchProbe(Axis:= , TriggerInput:= , Execute:= , WindowOnly:= , FirstPosition:= , LastPosition:= , Done=> , Busy=> , Error=> , ErrorID=> , RecordedPosition=> , CommandAborted=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis
TriggerInput	Trigger				Switch action reference

	signal				

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
WindowOnly	Trigger window	BOOL			
FirstPosition	Trigger start position	LREAL			Specify the start position of the receive trigger
LastPosition	Trigger end position	LREAL			Specify the end position of the receive trigger

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Setting operation is successful	BOOL	TRUE/FALSE	FALSE	TRUE: Command track is enabled
Busy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The command is being executed
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs
RecordedPosition	Trigger recording position	LREAL			
CommandAborted	Instruction interrupted	BOOL	TRUE/FALSE	FALSE	

3）Function Description

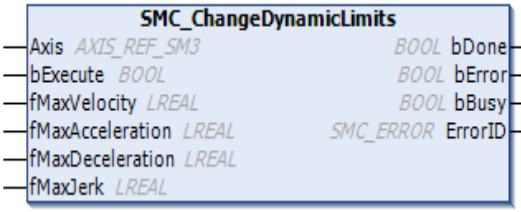
Record the current position of the running axis when triggered by the signal of the SMC_TouchProbe function block TriggerInput. Enable rising edge execution。

When the driver is latched: the driver will record the position when the latched signal is collected。

SMC_ChangeDynamicLimits

Set the dynamic limits of the axis (speed, acceleration, jolt). It can be called only when the axis is in power_off or stopped state。

1） Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_ ChangeDyn amicLimits			SMC_ChangeDynamicLimits (Axis:= , bExecute:= , fMaxVelocity:= , fMaxAcceleration:= , fMaxDeceleration:= , fMaxJerk:= , bDone=> , bError=> , bBusy=> , ErrorID=>);

2） Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
fMaxVelocity	speed	LREAL	Positive number		Maximum speed [u/s]
fMaxAcceleration	Acceleration	LREAL	Positive number		Maximum acceleration [u/s²]
fMaxDeceleration	deceleration	LREAL	Positive number		Maximum deceleration [u/s²]
fMaxJerk	Jerk	LREAL	Positive number		Maximum jump [u/s³]

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
bDone	Location recovery	BOOL	TRUE/FALSE	FALSE	TRUE: 当前位置新的限制已经设定
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE:发生错误时
bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE:指令执行中
nErrorID	error code	SMC_ERROR		0	发生错误时输出故障代码

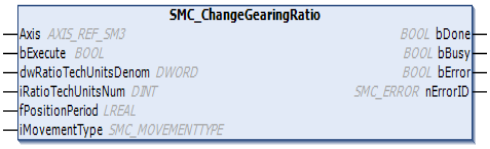
3） Function Description

Set the dynamic limits of the axis (speed, acceleration, jerk). It can be called only when the axis is in power_off or stopped state。

SMC_ChangeGearingRatio

This function block is used to change the user-set electronic gear ratio (pulse to user unit ratio) and drive type。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_ ChangeDynamicLimits			SMC_ChangeGearingRatio(Axis:= , bExecute:= , dwRatioTechUnitsDenom:= , iRatioTechUnitsNum:= , fPositionPeriod:= , iMovementType:= , bDone=> , bBusy=> , bError=> , nErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
dwRatioTechUnitsDenom		DWORD			Conversion of pulse units to applied units
iRatioTechUnitsNum		DINT			The dwRatioTechUnitsDenom value corresponds to the required application unit
fPositionPeriod		LREAL			The position cycle period (modulus value) is only valid for rotating motors.
iMovementType		SMC_MOVEMENTTYPE			0: Modulo axis 1: Finite axis

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
bDone	Location recovery	BOOL	TRUE/FALSE	FALSE	TRUE: The execution setting is completed
bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The command is being executed
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
nErrorID	error code	SMC_ERROR		0	Output fault code when an error occurs

3）Function Description

With the help of this function block, the electronic gear transmission ratio and transmission type can be modified.

On the rising edge of bExecute, if there is no error, the bBusy output is TRUE, the completed bDone output is TRUE, and the bBusy output is FALSE.

For example, a 17-bit encoder servo motor adds a 10:1 reduction ratio and drives the lead screw (10mm pitch), then the motor rotates 1 revolution

and the lead screw moving distance is 10mm. Set dwRatioTechUnitsDenom to 131072*10 and iRatioTechUnitsNum to 10.

4）Timing diagram

5）Error description

At the rising edge of bExecute;

Axis report error, Error output.

Invalid input value, Error output, error code SMC_CGR_ZERO_VALUES

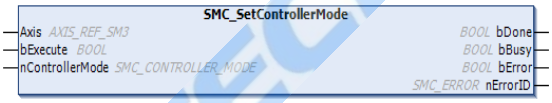
Axis is in command control operation, Error output, error code SMC_CGR_DRIVE_POWERED

The input modulus value is invalid (eg:<0), Error output, error code SMC_CGR_INVAALID_POSPERIOD

SMC_SetControllerMode

Set the current running mode of the servo, the default is the synchronous cycle position control mode. This function block can use the driver to switch to another control mode (the driver needs to support this function block)

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_SetControllerMode			SMC_SetControllerMode(Axis:= , bExecute:= , nControllerMode:= , bDone=> , bBusy=> , bError=> , nErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
nControllerMode	Control mode	SMC_CONTROLLER_MODE	SMC_position		The mode in which the motor is controlled SMC_nocontrol (SoftMotion does not send cyclic data to the drive, so the drive can be controlled in other ways.) SMC_torque（Torque mode） SMC_velocity（Speed mode） SMC_position（Position Mode） SMC_current（Current mode）

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
bDone	Recovery position	BOOL	TRUE/FALSE	FALSE	TRUE: The execution setting is completed
bBusy	In execution	BOOL	TRUE/FALSE	FALSE	TRUE: The execution setting is completed
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
nErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs

3）Function Description

This function block, if supported by the driver, can be used to switch to another controller mode。

SMC_SetControllerMode, activated by the rising edge of bExecute, gives the servo drive control mode command, or set the control mode through the value of Axis.out.byModesofOpration after the axis is configured (add object dictionary 6060h to the required value process data)。

The function block meets the conditions

1. The axis must support the required control mode, for example, the virtual axis cannot use this function block. In order to check this, please refer to the feature text PDFinside of the corresponding SoftMotion driver library。

2. The required cyclic I/O data must be mapped (for example, for torque mode: set torque object), and the synchronization cycle supported by each mode must be consistent。

3. When calling this function block, the axis cannot be in errorstop, stop or homing state. Otherwise, an error will be reported。

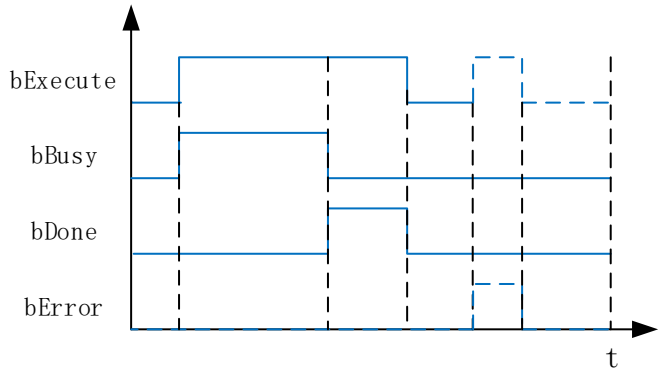
If the axis still does not change to the set control mode after the instruction is executed for 1000 task cycles, the instruction will report an error, and bError will change from FALSE to TRUE。

When the axis control mode changes from low-level to high-level: （torque->velocity,torque->position,velocity->position）, The function block will calculate the set value of the advanced mode. For example, when the torque mode is changed to the position mode, the function block will superimpose an expected position distance based on the actual position of the current axis (through the current actual speed and the time offset in the task period) To calculate) to compensate for the time lag between actual and set value。

After the instruction is executed, when the actual control mode of the axis is changed to the set control mode, the bDone signal is triggered, the actual internal axis will still run between the command trigger and the bDone signal trigger, and the function block will follow the setting during this time The control method calculates the appropriate setting value, but once the bDone signal is triggered and no other control commands continue to set the value of the axis, the axis will immediately stop and report an error. Therefore, the rising edge of the bDone signal needs to

be used to trigger MC_Halt, MC_MoveVelocity, MC_MoveAbsolute Wait for instructions to smoothly control the axis。

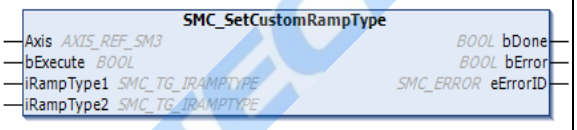
4）Timing diagram



SMC_SetCustomRampType

This function block sets the slope type of the axis。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_ SetCustom RampType			SMC_SetCustomRampType(Axis:= , bExecute:= , iRampType1:= , iRampType2:= , bDone=> , bError=> , eErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
iRampType1	First slope trajectory	SMC_TG_IRAMPTYPE			The trajectory of the first slope of the new slope type is usually the acceleration phase.
iRampType2	Second slope trajectory	SMC_TG_IRAMPTYPE			The new ramp type is the trajectory of the second ramp, usually in the deceleration phase.

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
bDone	carry out	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
nErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs

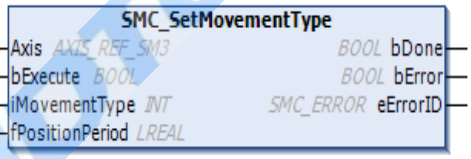
3）Function Description

Set the slope type of the axis to the new value. Unlike SMC_SetRampType, user-defined ramptypes (implementing SMC_TG_IRAMPTYPE) are possible. Generally, iRampType1 is equal to iRampType2. It can only be used when the axis is in the stopped state or power_off state. FB is executed synchronously on the rising edge of bExecute. After the call, bDone or bError is set。

SMC_SetMovementType

This function block changes the movement type of the virtual axis to modulo or linear mode。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_SetMovementType			SMC_SetMovementType(Axis:= , bExecute:= , iMovementType:= , fPositionPeriod:= , bDone=> , bError=> , eErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	Axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
iMovement Type	Drive type	INT		0	New drive type: 0 is the modulus value, 1 is linear.
fPosition Period		LREAL	Positive number	1	The new position period of the modular movement, must be a positive number

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
bDone	Complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
bError	Error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
nErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs

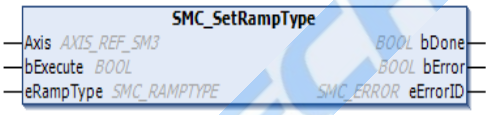
3）Function Description

Set the movement type of the virtual axis to linear or modulo type. It can only be used when the axis is in the stopped state or power_off state. FB is executed synchronously on the rising edge of bExecute. After the call, bDone or bError is set。

SMC_SetRampType

This function block sets the speed ramp type of the axis to a new value。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
SMC_SetRampType			SMC_SetRampType (Axis:= , bExecute:= , eRampType:= , bDone=> , bError=> , eErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
eRampType	Type of slope	SMC_RAMPTYPE		trapez	Define a new slope type. trapez: trapezoid sinsquare: Sin² quadratic_ramp: two times quadratic_smooth_ramp: quadratic smoothing

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
bDone	complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error

					occurs
nErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs

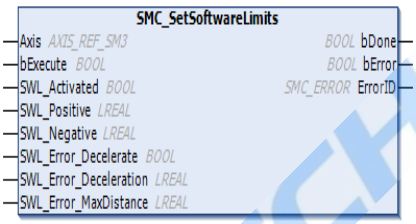
3）Function Description

Set the slope type of the axis to the new value. It can only be used when the axis is in the stopped state or power_off state. FB is executed synchronously on the rising edge of bExecute. After the call, bDone or bError is set。

SMC_SetSoftwareLimits

This function block sets the position limit of the axis and the corresponding error response

1）Instruction format

Instruction	Name	Graphic performance	ST perforceman
SMC_SetSoftwareLimits			SMC_SetSoftwareLimits(Axis:= , bExecute:= , SWL_Activated:= , SWL_Positive:= , SWL_Negative:= , SWL_Error_Decelerate:= , SWL_Error_Deceleration:= , SWL_Error_MaxDistance:= , bDone=> , bError=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
Axis	axis	AXIS_REF_SM3	—	—	Map to axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
bExecute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
SWL_Activated		BOOL		trapez	TRUE: activate location restriction
SWL_Positive		LREAL			Positive limit
SWL_Negative					Negative limit
SWL_Error_Decelerate					
SWL_Error_Deceleration					Error deceleration
SWL_Error_MaxDistance					Maximum braking distance

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
bDone	complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
bError	Error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
nErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs

3）Function Description

Set the position limit of the axis and the corresponding error response. If an axis is used in the context of coordinated motion, the corresponding axis group must be re-enabled in order to take into account the new limitations of the coordinated motion calculation。

Axis group function block description

MC_GroupEnable

This function block changes the state of the axis group from GroupDisabled to GroupStandby。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GroupEnable			MC_GroupEnable(AxisGroup:= , Execute:= , CompatibilityOptions:= , Done=> , Busy=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	Axis group	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
CompatibilityOptions		SMC_AXIS_GROUP_COMPATIBILITY_OPTIONS			To get the behavior of the previous version. Enter MC_GroupEnable to enable the user to make the behavior of the SoftMotion robot compatible with the previous version

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs

3）Function Description

This function block changes the status of the group from GroupDisabled to GroupStandby. This is a management function block because there is no movement. This command does not affect the power state of any single axis in the group. The configuration of the axis (such as dynamic limits) is readable. If the configuration of the group axis is changed, these changes will only be considered when the axis group is disabled and enabled again。

MC_GroupDisable

This function block changes the status of a group from groupstandby to GroupDisabled。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GroupDisable			MC_GroupDisable(AxisGroup:= , Execute:= , Done=> , Busy=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	Axis Group	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs

3) Function Description

This function block changes the status of a group from groupstandby to GroupDisabled. This is a management function block because there is no movement。

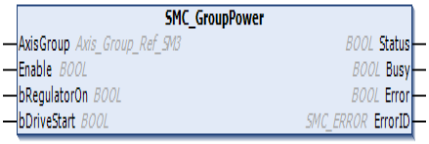
caution:

This command does not affect the power state of any single axis in the group If the group is not in the GroupStandby state when the command is issued, the function block returns an error and does not affect the group in any way.

SMC_GroupPower

Enable axis group (enable all axes of an axis group)。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GroupPower			SMC_GroupPower(AxisGroup:= , Enable:= , bRegulatorOn:= , bDriveStart:= , Status=> , Busy=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	AxisGroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
bRegulatorOn	Enable enable	BOOL	TRUE/FALSE		Set bRegulatorOn to TRUE to enable the power phase
bDriveStart	Disable stop	BOOL	TRUE/FALSE		Set bDriveStart to TRUE to disable quicktopmechanism

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Status	status	BOOL	TRUE/FALSE	FALSE	TRUE: The axis enable is completed and it is in the ready state.
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs

3) Function Description

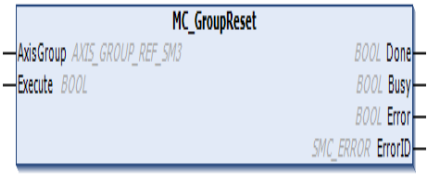
Enable all axes of an axis group。

Calling this function block will simply call the **MC_Power** of all axes in the axis group。This function block is not bound to the state of the axis group, nor does it affect the state of the axis group. For example, it can be called while the axis group is still disabled. After the output status becomes TRUE, the group status remains **GroupDisabled**.

MC_GroupReset

Axis group reset

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GroupReset			MC_GroupReset(AxisGroup:= , Execute:= , Done=> , Busy=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	AxisGroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs

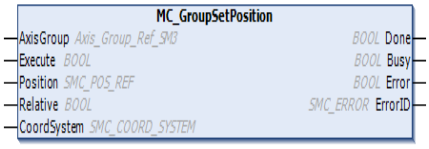
3) Function Description

This function block completes the transition from GroupErrorStop to GroupStandby state by resetting all internal errors related to the group.
This function block will also reset all axes in this group, such as MC_Reset。

MC_GroupSetPosition

Set the position of all axes of an axis group

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_Group SetPosition			MC_GroupSetPosition(AxisGroup:= , Execute:= , Position:= , Relative:= , CoordSystem:= , Done=> , Busy=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	AxisGroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
Position	Set location	SMC_POS_REF			The new position of the axis
Relative	Relative OR Absolute	BOOL	TRUE/FALSE	FALSE	Relative=TRUE, Absolute=FALSE (default)
CoordSystem	Coordinate system mode	SMC_COORD_SYSTEM		-	Refer to applicable coordinate system

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs

3）Function Description

Allows to set the positions of all axes of an axis group.
This function block calls MC_SetPosition for all axes of the axis group.
This function block must be called in the state groupstandby, and does not follow the dynamic coordinate system. After the command mc_groupcontinue, the command cannot be executed in the same loop.

MC_GroupContinue

This function block transfers the program back to the situation when MC_GroupInterrupt was issued。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GroupContinue			MC_GroupContinue(AxisGroup:= , continueData:= , Execute:= , Done=> , CommandAborted=> , Busy=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	AxisGroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
continueData		SMC_AXIS_GROUP_CONTINUE_DATA			Continuation data stored by MC_GroupInterrupt
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
CommandAborted	Terminated	BOOL	TRUE/FALSE	FALSE	TRUE: The current command was terminated by another command
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs

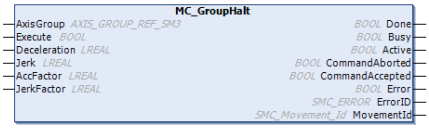
3）Function Description

This function block transfers the program back to the situation when MC_GroupInterrupt was issued. If MC_GroupStop is active, it immediately returns CommandAborted. If the axis group is not in the standby state, an error is returned. If you want to continue CP-tracking-movement, the axis group needs to have followed the corresponding position in PCS-coordinates. Otherwise, all axes must be at rest at the continuously recorded position. (See SMC_GroupGetContinuePosition).

MC_GroupHalt

This function block commands the axis group to control the movement to stop

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GroupHalt			MC_GroupHalt(AxisGroup:= , Execute:= , Deceleration:= , Jerk:= , AccFactor:= , JerkFactor:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	AxisGroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
Deceleration		LREAL		-	Deceleration [u/s²]. Only used when the continuous path movement stops.
Jerk		LREAL		-	Jump value [u/s³]. Only used when the continuous path movement stops.
AccFactor		LREAL		1	The maximum deceleration of each axis is multiplied by this coefficient, and it must be 1 in the range of [0,1]. It is only used when the point-to-point movement is stopped.
JerkFactor		LREAL			The maximum jump value of each axis is multiplied by this factor, and it must be in the range of [0,1]. It is only used when the point-to-point movement is stopped.

- Output variables

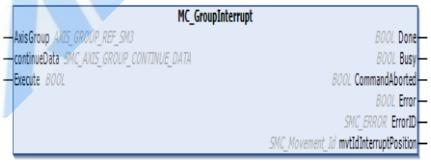
Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Active	In control	BOOL	TRUE/FALSE	FALSE	TRUE: The function block controls the axis
CommandAborted	Terminated	BOOL	TRUE/FALSE	FALSE	TRUE: The current command was terminated by another command
CommandAccepted		BOOL	TRUE/FALSE	FALSE	TRUE: The command has been accepted by the axis group
bError	Error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
nErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs
MovementId		SMC_Movement_Id			The unique identifier of the move is valid if CommandAccepted or Done is TRUE.

MC_GroupInterrupt

This function block interrupts the ongoing action and stops the movement of the axis group, but does not stop the interrupted movement (this means that in the interrupted function block, the output of CommandAborted will not be set and it is still busy). This function block is coupled with MC_GroupContinue. When MC_GroupContinue is executed, the function block will transfer the program back to the situation when MC_GroupInterrupt is executed. After completion, the axis group will be on standby。

This function block commands the axis group to control the movement to stop

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GroupInterrupt			MC_GroupInterrupt(AxisGroup:= , continueData:= , Execute:= , Done=> , Busy=> , CommandAborted=> , Error=> , ErrorID=> , mvtIdInterruptPosition=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	AxisGroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping
continueData		SMC_AXIS_GROUP_CONTINUE_DATA			

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
CommandAborted	Terminated	BOOL	TRUE/FALSE	FALSE	TRUE: The current command was terminated by another command
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
nErrorID	error code	SMC_ERROR		0	Output fault code when an error occurs
mvtIdInterruptPosition		SMC_Movement_Id			Movemnet Id corresponding to the interrupt position

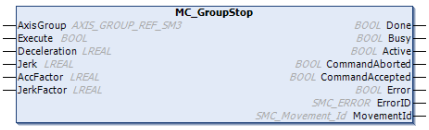
3）Function Description

If a cp -track -movement is active, we will stop on the path instead of relative to the MCS.
Does not support interrupting ptp tracking motion.
The interrupt can only be terminated by MC_GroupStop.
If another MC_GroupInterrupt or MC_GroupStop is active, the FB will immediately return CommandAborted.

MC_GroupStop

This function block commands a controlled motion stop and converts the axis group to the state group stop。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GroupStop			MC_GroupStop(AxisGroup:= , Execute:= , Deceleration:= , Jerk:= , AccFactor:= , JerkFactor:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	Axisgroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
Deceleration		LREAL		-	Deceleration [u/s²]. Only used when the continuous path movement stops.。
Jerk		LREAL		-	Jump value [u/s³]. Only used when the continuous path movement stops.。
AccFactor		LREAL		1	The maximum deceleration of each axis is multiplied by this coefficient, and it must be 1 in the range of [0,1]. It is only used when the point-to-point movement is stopped.
JerkFafctor		LREAL			The maximum jump value of each axis is multiplied by this factor, and it must be in the range of [0,1]. It is only used when the point-to-point movement is stopped.

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is completed and the pause state has been reached
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Active	In control	BOOL	TRUE/FALSE	FALSE	TRUE: The function block controls the axis
CommandAborted	Terminated	BOOL	TRUE/FALSE	FALSE	TRUE: The current command was terminated by another command
CommandAccepted		BOOL	TRUE/FALSE	FALSE	TRUE: The command has been accepted by the axis group
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
nErrorID	error code	SMC_ERROR		0	Output fault code when an error occurs
MovementId		SMC_Movement_Id			The unique identifier of the move is valid if CommandAccepted or Done is TRUE.

3）Function Description

This function block commands a controlled motion stop and converts the axis group to a state group stop.

It aborts any other movement commands. When the axis group is in the groupstop state, other FBs cannot perform any actions on the same axis group. When the axes grouphas reaches the pause state, the Done output is immediately set to TRUE. As long as the execution is still true or the axis has not stopped, the axis group remains in the state groupstopped. Once the setting is completed and the execution is FALSE, the axis group enters the state GroupStandBy. This command can only be terminated by MC_GroupDisableor by MC_Power. The given dynamic is not affected by MC_GroupSetOverride, neither in advance nor after。

Stop point-to-point movement: The speed/acc-/deceleration/jerk of each axis is the attribute of each axis and is not specified in this function block. See AccFactor and JerkFactor input.

MC_MoveCircularAbsolute

This block command moves a group cycle to the absolute position in the specified coordinate system. According to the value of the input circle modulus, several methods can be used to describe the arc segment。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_MoveCircularAbsolute			MC_MoveCircularAbsolute(AxisGroup:= , Execute:= , CircMode:= , AuxPoint:= , EndPoint:= , PathChoice:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , CoordSystem:= , BufferMode:= , TransitionMode:= , TransitionParameter:= , OrientationMode:= , VelFactor:= , AccFactor:= , JerkFactor:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	axisgroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
CircMode	Arc mode	SMC_CIRC_MODE			Specify the meaning of input auxiliary points and circle direction
AuxPoint	Auxiliary point coordinates	SMC_POS_REF			The auxiliary point specifies the coordinate system. According to CircMode
EndPoint	End point coordinates	SMC_POS_REF			End point designated coordinate system
PathChoice	Path direction	MC_CIRC_PATHCHOICE			Path selection: clockwise or counterclockwise。
Velocity	speed	LREAL			
Acceleration	Acceleration	LREAL			
Deceleration	deceleration	LREAL			Deceleration [u/s²]. Only used when the continuous path movement stops.
Jerk	Jump	LREAL			Jump value [u/s³]. Only used when the continuous path movement stops.
CoordSystem	Coordinate system mode	SMC_COORD_SYSTEM		-	Refer to applicable coordinate system
BufferMode	Buffer mode	MC_BUFFER_MODE		-	The buffer mode describes how to insert motion commands into the commanded motion queue.。
TransitionMode	Mixed mode	MC_TRANSITION_MODE			Define the mixing method in mixed buffer mode
TransitionParameter	Parameter mix	ARRAY [0..(SMC_RCNST.MAX_TRANS_PARAMS - 1)] OF LREAL			Parameter mix
OrientationMode	Interpolation direction	SMC_ORIENTATION_MODE			Determine how to interpolate the direction. The direction mode describes how the direction of CP movement is interpolated.
VelFactor		LREAL		1	The maximum speed of each axis is multiplied by this factor, which must be in the range of

					[0,1].
AccFactor		LREAL		1	The maximum deceleration of each axis is multiplied by this coefficient, and it must be 1 in the range of [0,1]. It is only used when the point-to-point movement is stopped.
JerkFafctor		LREAL		1	The maximum jump value of each axis is multiplied by this factor, and it must be in the range of [0,1]. It is only used when the point-to-point movement is stopped.

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Active	In control	BOOL	TRUE/FALSE	FALSE	TRUE: The function block controls the axis
CommandAborted	Terminated	BOOL	TRUE/FALSE	FALSE	TRUE: The current command was terminated by another command
CommandAccepted		BOOL	TRUE/FALSE	FALSE	TRUE: The command has been accepted by the axis group
bError	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
nErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs
MovementId		SMC_Movement_Id			The unique identifier of the move is valid if CommandAccepted or Done is TRUE.

MC_MoveCircularRelative

This function block commands a relative position circular motion in the specified coordinate system of an axis group。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_MoveCircularRelative			MC_MoveCircularRelative(AxisGroup:= , Execute:= , CircMode:= , AuxPoint:= , EndPoint:= , PathChoice:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , CoordSystem:= , BufferMode:= , TransitionMode:= , TransitionParameter:= , OrientationMode:= , VelFactor:= , AccFactor:= , JerkFactor:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	axisgroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
CircMode	Arc mode	SMC_CIRC_MODE			Specify the meaning of the input auxiliary point and the direction of the circle. . These points are relatively designated, that is, the vertical line starts from this point and ends at the spearhead point designated by the auxiliary point of the input signal.
AuxPoint	Auxiliary	SMC_POS_			The auxiliary point

	point coordinates	REF			specifies the coordinate system. According to CircMode. This point is relative to the starting point
EndPoint	End point coordinates	SMC_POS_REF			The end point specifies the coordinate system. This point is relative to the starting point
PathChoice	Path direction	MC_CIRC_PATHCHOICE			Path selection: clockwise or counterclockwise.
Velocity	speed	LREAL			In the coordinate system that defines the path, the maximum path speed [u/s]. Must be a positive value.
Acceleration	Acceleration	LREAL			Maximum acceleration (u / s²)
Deceleration	deceleration	LREAL			Maximum acceleration (u / s²)
Jerk	Jump	LREAL			Jump value [u/s³].
CoordSystem	Coordinate system mode	SMC_COORD_SYSTEM		-	Refer to applicable coordinate system
BufferMode	Buffer mode	MC_BUFFER_MODE		-	The buffer mode describes how to insert motion commands into the commanded motion queue。
TransitionMode	Mixed mode	MC_TRANSITION_MODE			Define the mixing method in mixed buffer mode
TransitionParameter	Parameter mix	ARRAY [0..(SMC_RCNST.MAX_TRANS_PARAMS - 1)] OF LREAL			Parameter mix
OrientationMode	Interpolation direction	SMC_ORIENTATION_MODE			Determine how to interpolate the direction. The direction mode describes how the direction of CP movement is interpolated.
VelFactor		LREAL		1	The maximum speed of each axis is multiplied by this factor, which must be in the range of [0,1].
AccFactor		LREAL		1	The maximum deceleration of each axis is multiplied by this coefficient, and it must be 1 in the range of [0,1]. It is only used when the point-to-point movement is stopped.
JerkFafctor		LREAL		1	The maximum jump value of each axis is multiplied by this factor, and it must be in the

					range of [0,1]. It is only used when the point-to-point movement is stopped.

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete,
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Active	In control	BOOL	TRUE/FALSE	FALSE	TRUE: The function block controls the axis
CommandAborted	Terminated	BOOL	TRUE/FALSE	FALSE	TRUE: The current command was terminated by another command
CommandAccepted		BOOL	TRUE/FALSE	FALSE	TRUE: The command has been accepted by the axis group
bError	Error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
nErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs
MovementId		SMC_Movement_Id			The unique identifier of the move is valid if CommandAccepted or Done is TRUE.

3)Function Description

This function block commands a circular motion of an axis group to the relative position in the specified coordinate system. According to the value of the input circle modulus, several methods can be used to describe the arc.

In the case of SMC_CIRC_MODE. Boundary or SMC_CIRC_MODE. The center, auxiliary point is also interpreted relative to the start position.

MC_MoveDirectAbsolute

This function block commands to move an axis group to the absolute position specified in the specified coordinate system. (Non-imputation)

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_MoveDirectAbsolute			MC_MoveDirectAbsolute(AxisGroup:= , Execute:= , Position:= , MovementType:= , CoordSystem:= , BufferMode:= , TransitionMode:= , TransitionParameter:= , VelFactor:= , AccFactor:= , JerkFactor:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	axisgroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
Position	Absolute position	SMC_POS_REF			The position of the endpoint in the specified coordinate system
MovementType		SMC_PTP_MOVEMENT_TYPE			Types of PTP movement
CoordSystem	Coordinate system mode	SMC_COORD_SYSTEM		-	Refer to applicable coordinate system
BufferMode	Buffer mode	MC_BUFFER_MODE		-	The buffer mode describes how to insert motion commands into the commanded motion queue.
TransitionMode	Buffer mode	MC_TRANSITION_MODE			Define the mixing method in mixed buffer mode
TransitionParameter	Parameter mix	ARRAY [0..(SMC_RCNST.MAX_			Parameter mix

		TRANS_PARAMS - 1)] OF LREAL			
VelFactor		LREAL		1	The maximum speed of each axis is multiplied by this factor, which must be in the range of [0,1].
AccFactor		LREAL		1	The maximum deceleration of each axis is multiplied by this coefficient, and it must be 1 in the range of [0,1]. It is only used when the point-to-point movement is stopped.
JerkFafctor		LREAL		1	The maximum jump value of each axis is multiplied by this factor, and it must be in the range of [0,1]. It is only used when the point-to-point movement is stopped.

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Active	In control	BOOL	TRUE/FALSE	FALSE	TRUE: The function block controls the axis
CommandAborted	Terminated	BOOL	TRUE/FALSE	FALSE	TRUE: The current command was terminated by another command
CommandAccepted		BOOL	TRUE/FALSE	FALSE	TRUE: The command has been accepted by the axis group
bError	Error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
nErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs
MovementId		SMC_Movement_Id			The unique identifier of the move is valid if CommandAccepted or Done is TRUE.

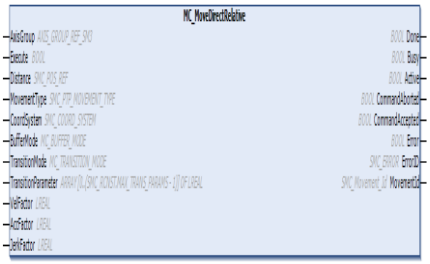
3）Function Description

Each axis is moved to its target position independently, and the movement is only synchronized, so that all axes reach the target at the same time. This means that the path of TCP transmission depends on the kinematic transformation used. Generally not a straight line。

MC_MoveDirectRelative

This function block command moves an axis group through the specified relative distance in the specified coordinate system。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_MoveDirectRelative			MC_MoveDirectRelative(AxisGroup:= , Execute:= , Distance:= , MovementType:= , CoordSystem:= , BufferMode:= , TransitionMode:= , TransitionParameter:= , VelFactor:= , AccFactor:= , JerkFactor:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	axisgroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
Distance	relative position	SMC_CIRC_MODE			The distance in the specified coordinate system. The relative orientation in a /. b /. c is interpreted as rotation. Then determine the final direction of the end position as Rot_End(u) = Rot_Rel(Rot_Start(u)), where Rot_Rel is rotation, and Rot_Start is the rotation belonging to the movement start position. The rotation belonging to the azimuth is a function that converts the direction vector u from the TCP

					coordinate system to the base coordinate system. The corresponding rotation matrix can be calculated by A/B/C asR = Rz(A) * Ry(B) * Rz(C).
MovementType		SMC_PTP_MOVEMENT_TYPE			Types of PTP movement
CoordSystem	Coordinate system mode	SMC_COORD_SYSTEM		-	Refer to applicable coordinate system
BufferMode	Buffer mode	MC_BUFFER_MODE		-	The buffer mode describes how to insert motion commands into the commanded motion queue.
TransitionMode	Mixed mode	MC_TRANSITION_MODE			Define the mixing method in mixed buffer mode
TransitionParameter	Parameter mix	ARRAY [0..(SMC_RCNST.MAX_TRANS_PARAMS - 1)] OF LREAL			Parameter mix
VelFactor		LREAL		1	The maximum speed of each axis is multiplied by this factor, which must be in the range of [0,1].
AccFactor		LREAL		1	The maximum deceleration of each axis is multiplied by this coefficient, and it must be 1 in the range of [0,1]. It is only used when the point-to-point movement is stopped.
JerkFafctor		LREAL		1	The maximum jump value of each axis is multiplied by this factor, and it must be in the range of [0,1]. It is only used when the point-to-point movement is stopped.

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Active	In control	BOOL	TRUE/FALSE	FALSE	TRUE: The function block controls the axis
CommandAborted	Terminated	BOOL	TRUE/FALSE	FALSE	TRUE: The current command was terminated by another command
CommandAccepted		BOOL	TRUE/FALSE	FALSE	TRUE: The command has been accepted by the axis group
bError	Error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
nErrorID	Error	SMC_ERRO		0	Output fault code when an

	code	R			error occurs
MovementId		SMC_Movement_Id			The unique identifier of the move is valid if CommandAccepted or Done is TRUE.

MC_MoveLinearAbsolute

This function block command linearly move an axis group to the absolute position in the specified coordinate system。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_MoveLinearAbsolute			MC_MoveLinearAbsolute(AxisGroup:= , Execute:= , Position:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , CoordSystem:= , BufferMode:= , TransitionMode:= , TransitionParameter:= , OrientationMode:= , VelFactor:= , AccFactor:= , JerkFactor:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	axisgroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
Position	Absolute position	SMC_POS_REF			The position of the endpoint in the specified coordinate system
Velocity	velocity	LREAL			The maximum path speed [u/s] of the coordinate system that defines the path. It is always positive and does not necessarily

					reach this speed.
Acceleration	Acceleration	LREAL			Maximum acceleration (u / s²). Always positive.
Deceleration	deceleration	LREAL			Maximum deceleration (u / s²). Always positive.
Jerk	Jump	LREAL			Maximum jump (u / s³) Always positive.
CoordSystem	Coordinate system mode	SMC_COORD_SYSTEM		-	Refer to applicable coordinate system
BufferMode	Buffer mode	MC_BUFFER_MODE		-	The buffer mode describes how to insert motion commands into the commanded motion queue.
TransitionMode	Mixed mode	MC_TRANSITION_MODE			Define the mixing method in mixed buffer mode
TransitionParameter	Parameter mix	ARRAY [0..(SMC_RCNST.MAX_TRANS_PARAMS - 1)] OF LREAL			Parameter mix
OrientationMode	Interpolation direction	SMC_ORIENTATION_MODE			Determine how to interpolate the direction. The direction mode describes how the direction of CP movement is interpolated.
VelFactor	Speed factor	LREAL		1	The maximum speed of each axis is multiplied by this factor, which must be in the range of [0,1].
AccFactor	Acceleration factor	LREAL		1	The maximum deceleration of each axis is multiplied by this coefficient, and it must be 1 in the range of [0,1]. It is only used when the point-to-point movement is stopped.
JerkFafctor	Jump factor	LREAL		1	The maximum jump value of each axis is multiplied by this factor, and it must be in the range of [0,1]. It is only used when the point-to-point movement is stopped.

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Active	In control	BOOL	TRUE/FALSE	FALSE	TRUE: The function block controls the axis
CommandAborted	Terminated	BOOL	TRUE/FALSE	FALSE	TRUE: The current command was terminated by another command
CommandAccepted		BOOL	TRUE/FALSE	FALSE	TRUE: The command has been accepted by the axis

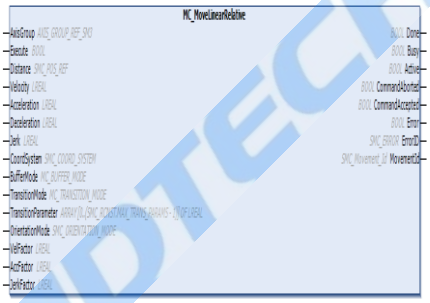
					group
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs
MovementId		SMC_Movement_Id			The unique identifier of the move is valid if CommandAccepted or Done is TRUE.

3）Function Description

The vel/acc-/dec/jerk of each axis is the attribute of each axis and is not specified in this function block. See input speed factor, additional factor and jitter factor。

MC_MoveLinearRelative

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_MoveLinearRelative			MC_MoveLinearRelative(AxisGroup:= , Execute:= , Distance:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , CoordSystem:= , BufferMode:= , TransitionMode:= , TransitionParameter:= , OrientationMode:= , VelFactor:= , AccFactor:= , JerkFactor:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	Axisgroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
Distance	Absolute position	SMC_POS_REF			Relative distance in the specified coordinate system

Velocity	velocity	LREAL			The maximum path speed [u/s] of the coordinate system that defines the path. It is always positive and does not necessarily reach this speed.
Acceleration	Acceleration	LREAL			Maximum acceleration (u / s²). Always positive.
Deceleration	deceleration	LREAL			Maximum deceleration (u / s²). Always positive.
Jerk	Jump	LREAL			Maximum jump (u / s³) Always positive.
CoordSystem	Coordinate system mode	SMC_COORD_SYSTEM		-	Refer to applicable coordinate system
BufferMode	Buffer mode	MC_BUFFER_MODE		-	The buffer mode describes how to insert motion commands into the commanded motion queue.
TransitionMode	Mixed mode	MC_TRANSITION_MODE			Define the mixing method in mixed buffer mode
TransitionParameter	Parameter mix	ARRAY [0..(SMC_RCNST.MAX_TRANS_PARAMS - 1)] OF LREAL			Parameter mix
OrientationMode	Interpolation direction	SMC_ORIENTATION_MODE			Determine how to interpolate the direction. The direction mode describes how the direction of CP movement is interpolated.
VelFactor	Speed factor	LREAL		1	The maximum speed of each axis is multiplied by this factor, which must be in the range of [0,1].
AccFactor	Acceleration factor	LREAL		1	The maximum deceleration of each axis is multiplied by this coefficient, and it must be 1 in the range of [0,1]. It is only used when the point-to-point movement is stopped.
JerkFafctor	Jump factor	LREAL		1	The maximum jump value of each axis is multiplied by this factor, and it must be in the range of [0,1]. It is only used when the point-to-point movement is stopped.

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Active	In control	BOOL	TRUE/FALSE	FALSE	TRUE: The function block

					controls the axis
CommandAborted	Terminated	BOOL	TRUE/FALSE	FALSE	TRUE: The current command was terminated by another command
CommandAccepted		BOOL	TRUE/FALSE	FALSE	TRUE: The command has been accepted by the axis group
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs
MovementId		SMC_Movement_Id			The unique identifier of the move is valid if CommandAccepted or Done is TRUE.

3) Function Description

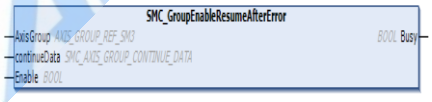
The vel/acc-/dec/jerk of each axis is the attribute of each axis and is not specified in this function block. See input speed factor, additional factor and jitter factor。

SMC_GroupEnableResumeAfterError

This function block allows the programmed movement to be restored after an axis group error. The continueData passed to the functionblock is used to store the necessary information。

To recover after an error occurs, you first need to use the mc_groupreset_axis group error. Then, you need to move the axis group to a continuous position, you can use smc_groupgetcontinueposition to query the position from the continuous data. Finally, you can use MC_GroupContinue to resume the movement that was interrupted by the error。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GroupEnableResumeAfterError			SMC_GroupEnableResumeAfterError(AxisGroup:= , continueData:= , Enable:= , Busy=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	axisgroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping
continueData		SMC_AXIS_GROUP_CONTINUE_DATA			Write on completion

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Enable	Execute	BOOL		—	TRUE: execute function block

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Busy	Being executed	BOOL		—	TRUE: The function block is being executed

3）Function Description

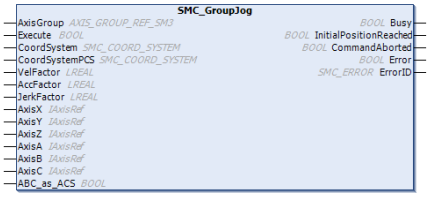
If MC_GroupInterrupt is active when an error occurs, the resumeafter error behaves as if there is no command interruption. After MC_GroupContinue, the movement will continue without interruption. The MC_GroupInterrupt that was active when the error occurred will return an error。

If MC_GroupStop is active when an error occurs, it will recover after the error occurs and re-enable group stop. But the axis groupwill has been stopped, and MC_GroupStop has cleared all the swing movements, so no movement will occur。

If an axis group error occurs, the function block instance of the active motion will display an error (output active and busy become false, and output error becomes TRUE). An error is also displayed when the function block instance for buffering or mixed action is not yet activated. After recovery, all these function block instances will become busy/active again and their errors/commandabortedoutput will be reset。

SMC_GroupJog

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GroupJog			SMC_GroupJog(AxisGroup:= , Execute:= , CoordSystem:= , CoordSystemPCS:= , VelFactor:= , AccFactor:= , JerkFactor:= , AxisX:= , AxisY:= , AxisZ:= , AxisA:= , AxisB:= , AxisC:= , ABC_as_ACS:= , Busy=> , InitialPositionReached=> , CommandAborted=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	Axisgroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
CoordSystem	Coordinate system mode	SMC_COORD_SYSTEM		-	The coordinate system used to explain the position of the coordinate axis
CoordSystemPCS		SMC_COORD_SYSTEM		-	The coordinate system used internally for JOG movement.
VelFactor	Speed factor	LREAL		1	The maximum speed of each axis is multiplied by this factor, which must be in the range of [0,1].
AccFactor	Acceleration factor	LREAL		1	The maximum deceleration of each axis is multiplied by this coefficient, and it must be 1 in the range of [0,1]. It is only used when the point-to-point movement is stopped.
JerkFafctor	Jump factor	LREAL		1	The maximum jump value of each axis is multiplied by this factor, and it must be in the range of [0,1]. It is only used when the point-to-point movement is stopped.
AxisX	the axis of the X coordinate	IAxisRef			Control the axis of the x coordinate. If not applied, set to 0.
AxisY	the axis of the Y coordinate	IAxisRef			Control the axis of the y coordinate. If not applied, set to 0.
AxisZ	the axis of the Z coordinate	IAxisRef			Control the axis of the z coordinate. If not applied, set to 0.
AxisA	the axis of the A coordinate	IAxisRef			The axis that controls the a coordinate (rotation around the z axis), or the first tool axis. If not applied, set to 0.
AxisB	the axis of the B coordinate	IAxisRef			The axis that controls the b coordinate (rotation around the y axis), or the second tool axis. If not applied, set to 0.
AxisC	the axis of the C coordinate	IAxisRef			The axis that controls the c coordinate (rotation around the z

					axis), or the third tool axis.
ABC_as_ACS		BOOL			If TRUE, the position of AxisA, AxisB and AxisC will be interpreted as the target position of the tool kinematics axis, otherwise it will be interpreted as the zyz representation of the direction. If CoordSystem is set to ACS, it is ignored. If TRUE, the kinematic conversion of the axis group must be type-kin_coupled and support SMC_ORIENTATION_MODE.Axis.

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Active	In control	BOOL	TRUE/FALSE	FALSE	TRUE: The function block controls the axis
InitialPosition Reached		BOOL			TRUE: The position given by the coordinate axis has been reached
CommandAborted	Terminated	BOOL	TRUE/FALSE	FALSE	TRUE: The current command was terminated by another command
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs

3）Function Description

Activate the coordinated jogging axis group. When this function block is active, the axis group will track the position (and direction) given by the configured axis.

Initially, moving to the current position of the axis is commanded.

The position of the axis is interpreted in the coordinate system defined by the input CoordSystem. For example, to jog in machine coordinates, use MCS here. By setting ABC_as_ACS, you can use the mixed interpretation of the reference axis, X/Y/Z as the Cartesian coordinate position, and a /B/C as the tool kinematics axis position.

Since the location is tracked by tracking, one of the product coordinate systems is used for this purpose. This can be reflected in the input CoordSystemPCS. Note that this coordinate system has changed in the axis group.

In order to actually perform Cartesian coordinate jogging, the function block MC_Jogor SMC_Inch can be used on the axis that is the input of this function block.

Before restarting the FB without changing CoordSystemPCS, you must order an additional action, such as MC_GroupHalt. Otherwise, it will return SMC_AXIS_GROUP_PCS_STILL_IN_USE.

SMC_GroupJog2

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GroupJog2			SMC_GroupJog2(AxisGroup:= , Enable:= , Forward:= , Backward:= , MaxLinearDistance:= , MaxAngularDistance:= , Velocity:= , Acceleration:= , Deceleration:= , Jerk:= , VelFactor:= , AccFactor:= , JerkFactor:= , CoordSystem:= , OrientationMode:= , ABC_as_ACS:= , Active=> , Busy=> , Error=> , ErrorID=> , CurrentPosition=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	axisgroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
Forward		ARRAY [0..(SMC_RCNST.MAX_AXES - 1)] OF BOOL			Enter the respective coordinates to jog forward
Backward		ARRAY [0..(SMC_RCNST.MAX_AXES - 1)] OF BOOL			Enter the respective coordinates to jog backward.
MaxLinearDistance					When a non-new rising edge is detected at the forward/backward input, the maximum

					straight line distance of the coordinate is jog.
MaxAngularDistance					When a non-new rising edge is detected at the forward/backward input, the maximum angular distance of the coordinate is jog.
Velocity	Velocity	LREAL			The maximum path speed [u/s] of the coordinate system that defines the path. It is always positive and does not necessarily reach this speed.
Acceleration	Acceleration	LREAL			Maximum acceleration (u / s²). Always positive.
Deceleration	Deceleration	LREAL			Maximum deceleration (u / s²). Always positive.
Jerk	Jump	LREAL			The maximum path speed [u/s] of the coordinate system that defines the path. It is always positive and does not necessarily reach this speed.
VelFactor	Speed factor	LREAL		1	The maximum speed of each axis is multiplied by this factor, which must be in the range of [0,1].
AccFactor	Acceleration factor	LREAL		1	The maximum deceleration of each axis is multiplied by this coefficient, and it must be 1 in the range of [0,1]. It is only used when the point-to-point movement is stopped.
JerkFafctor	Jump factor	LREAL		1	The maximum jump value of each axis is multiplied by this factor, and it must be in the range of [0,1]. It is only used when the point-to-point movement is stopped.
CoordSystem	Coordinate system mode	SMC_COORD_SYSTEM		-	The coordinate system used to explain the position of the coordinate axis
OrientationMode	Interpolation direction	SMC_ORIENTATION_MODE		-	The direction mode describes how the direction of CP movement is interpolated.
ABC_as_ACS		BOOL			If TRUE forward/backward (0. .2] control tcp (X, Y, Z) position and forward/backward

					[3..]5] control tool axis position. If CoordSystem is set to ACS, It is ignored. If it is TRUE, the kinematic conversion of the axis group must be type-kin_coupled and support SMC_ORIENTATION_MODE.Axis.

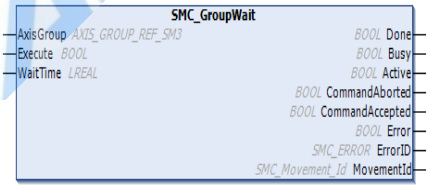
- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Active	In control	BOOL	TRUE/FALSE	FALSE	TRUE: The function block controls the axis
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	error code	SMC_ERROR		0	Output fault code when an error occurs
CurrentPosition	current location	SMC_POS_REF			The current position of the axis group

SMC_GroupWait

This function block allows the program to wait on the path, that is, between the other two movements. You cannot blend in or not wait. The waiting time starts immediately after the previous command is completed。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GroupWait			SMC_GroupWait(AxisGroup:= , Execute:= , WaitTime:= , Done=> , Busy=> , Active=> , CommandAborted=> , CommandAccepted=> , Error=> , ErrorID=> , MovementId=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	Axisgroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE
Distance	Absolute position	SMC_POS_REF			Relative distance in the specified coordinate system
Velocity	velocity	LREAL			The maximum path speed [u/s] of the coordinate system that defines the path. It is always positive and does not necessarily reach this speed.
Acceleration	Acceleration	LREAL			Maximum acceleration (u / s²). Always a positive number.
Deceleration	deceleration	LREAL			Maximum deceleration (u / s²). Always a positive number.
Jerk	Jump	LREAL			Maximum jump (u / s³) Always a positive number.
CoordSystem	Coordinate system mode	SMC_COORD_SYSTEM		-	Refer to applicable coordinate system
BufferMode	Buffer mode	MC_BUFFER_MODE		-	The buffer mode describes how to insert motion commands into the commanded motion queue.
TransitionMode	Mixed mode	MC_TRANSITION_MODE			Define the mixing method in mixed buffer mode
TransitionParameter	Parameter mix	ARRAY [0..(SMC_RCNST.MAX_TRANS_PARAMS - 1)] OF LREAL			Parameter mix
OrientationMode	Interpolation direction	SMC_ORIENTATION_MODE			Determine how to interpolate the direction. The direction mode describes how the direction of CP movement is interpolated.
VelFactor	Speed factor	LREAL		1	The maximum speed of each axis is multiplied by this factor, which must be in the range of [0,1].
AccFactor	Acceleration factor	LREAL		1	The maximum deceleration of each axis is multiplied by this coefficient, and it must be 1 in the range of [0,1]. It is only used when the point-to-point movement is stopped.
JerkFafctor	Jump factor	LREAL		1	The maximum jump value of each axis is multiplied by this factor, and it must be in the

					range of [0,1]. It is only used when the point-to-point movement is stopped.

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Complete	BOOL	TRUE/FALSE	FALSE	TRUE: The execution of the function block is complete
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Active	In control	BOOL	TRUE/FALSE	FALSE	TRUE: The function block controls the axis
CommandAborted	Terminated	BOOL	TRUE/FALSE	FALSE	TRUE: The current command was terminated by another command
CommandAccepted		BOOL	TRUE/FALSE	FALSE	TRUE: The command has been accepted by the axis group
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERROR		0	Output fault code when an error occurs
MovementId		SMC_Movement_Id			The unique identifier of the move is valid if CommandAccepted or Done is TRUE.

3) Function Description

This function block allows the program to wait on the path, that is, between the other two movements. You cannot blend in or not wait. The waiting time starts immediately after the previous command is completed。The waiting time between the two movements around will be exactly the configured time. This means that the effective waiting time between two actions can even be shortened by SMC_GroupWait. An example is a buffer movement: usually, a buffer movement always starts at the beginning of a bus cycle. However, if the bufferedmovement follows SMC_GroupWait, then it will start immediately after the wait time, not at the beginning of the next cycle

If a waiting command follows a tracking movement, the axis group will track the previously moved endpoint within the specified time.

If the axis is not in a static state, but is not controlled by the axis group, and the command is waiting, the axis group will report errorSMC_AXIS_GROUP_IDLE_WAIT_AXES_MOVING.

When the set value is updated once every bus cycle, the waiting time that is different from the multiple of the bus cycle time will serve as a smoother start of the subsequent movement.

Due to technical reasons, in the following situations, the waiting time can be increased by one cycle at most:-not tracking -> waiting -> tracking-tracking -> waiting -> not tracking-tracking -> waiting -> ptp tracking

MC_AddAxisToGroup

This function block adds an axis to an axis group

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_AddAxis ToGroup			MC_AddAxisToGroup(AxisGroup:= , Axis:= , Execute:= , Done=> , Busy=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	Axisgroup	AXIS_GROUP_ REF_SM3	—	—	Axis group mapping
Axis	Axis	AXIS_REF_SM3	-	-	Map to axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Add complete	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is completed and the axis is added
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Error	Error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERRO R			An error has occurred fault code

3) Function Description

This function block adds an axis to an axis group

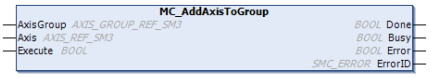
If the state of the axis group is not disabled, an error will be created and the axes will not be deleted from the group.

This is a management function block because there is no movement. The command could not be buffered.

MC_RemoveAxisFromGroup

This function block deletes an axis from an axis group。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_RemoveAxisFromGroup			MC_RemoveAxisFromGroup(AxisGroup:= , Axis:= , Execute:= , Done=> , Busy=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	Axisgroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping
Axis	axis	AXIS_REF_SM3	-	-	Map to axis

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Removal complete	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is completed and the axis has been removed
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERROR			An error has occurred fault code

3) Function Description

This function block deletes an axis from an axis group。

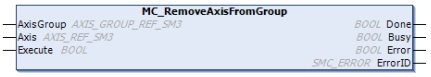
If the state of the axis group is not disabled, an error will be created and the axes will not be deleted from the group.

This is a management function block because there is no movement. Unable to buffer the command。

MC_UngroupAllAxes

This function block deletes all axes from an axis group.

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_UngroupAllAxes			MC_UngroupAllAxes(AxisGroup:= , Execute:= , Done=> , Busy=> , Error=> , ErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	axisgroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Execute	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE

- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Done	Removal complete	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is completed and the axis has been removed
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	error code	SMC_ERROR			An error has occurred fault code

3) Function Description

This function block deletes all axes from an axis group。

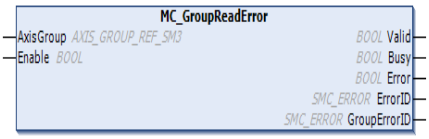
If the state of the axis group is not disabled, an error will be created and the axes will not be deleted from the group.

This is a management function block because there is no movement. The command could not be buffered.

MC_GroupReadError

This function block reads general axis group errors that have nothing to do with the function block.
For example: (software) limit switch overrun or single axis error

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GroupReadError			MC_GroupReadError(AxisGroup:= , Enable:= , Valid=> , Busy=> , Error=> , ErrorID=> , GroupErrorID=>);

2）Related variables

- Input and output variables

Input and output variables	Name	Type of data	Effective range	Initial value	Description
AxisGroup	axisgroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	Type of data	Effective range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE

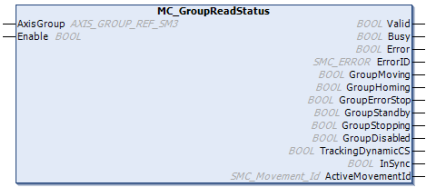
- Output variables

Output variable	Name	Type of data	Effective range	Initial value	Description
Valid	The output is valid	BOOL	TRUE/FALSE	FALSE	TRUE: The effective output of the function block is available
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERROR			An error has occurred fault code
GroupErrorID	Axis group error code	SMC_ERROR			Axis group wrong value

MC_GroupReadStatus

This function block returns the status of an axis group。

1）Instruction format

Instruction	Name	Graphic performance	ST performance
MC_GroupReadStatus			MC_GroupReadStatus(AxisGroup:= , Enable:= , Valid=> , Busy=> , Error=> , ErrorID=> , GroupMoving=> , GroupHoming=> , GroupErrorStop=> , GroupStandby=> , GroupStopping=> , GroupDisabled=> , TrackingDynamicCS=> , InSync=> , ActiveMovementId=>);

2）Related variables

- Input and output variables

Input and output variables	Name	type of data	Effective range	Initial value	Description
AxisGroup	Axisgroup	AXIS_GROUP_REF_SM3	—	—	Axis group mapping

- Input variables

Input variable	Name	type of data	Effective range	Initial value	Description
Enable	Execution conditions	BOOL	TRUE/FALSE	FALSE	Trigger on rising edge, start execution condition when TRUE

- Output variables

Output variable	Name	type of data	Effective range	Initial value	Description
Valid	The output is valid	BOOL	TRUE/FALSE	FALSE	TRUE: The valid output of the function block is available
Busy	Being executed	BOOL	TRUE/FALSE	FALSE	TRUE: The function block is being executed
Error	error	BOOL	TRUE/FALSE	FALSE	TRUE: When an error occurs
ErrorID	Error code	SMC_ERROR			An error has occurred fault code
GroupMoving		BOOL	TRUE/FALSE		TRUE: The axis group is in this state. Specific reference: SMC_AXIS_GROUP_STATE
GroupHoming		BOOL	TRUE/FALSE		TRUE: The axis group is in this state. Specific reference: SMC_AXIS_GROUP_STATE
GroupErrorStop		BOOL	TRUE/FALSE		TRUE: The axis group is in this state. Specific reference: SMC_AXIS_GROUP_STATE
GroupStandby		BOOL	TRUE/FALSE		TRUE: The axis group is in this state. Specific reference: SMC_AXIS_GROUP_STATE
GroupStopping		BOOL	TRUE/FALSE		TRUE: The axis group is in this state. Specific reference:

					SMC_AXIS_GROUP_STATE
GroupDisabled		BOOL	TRUE/FALSE		TRUE: The axis group is in this state. Specific reference: SMC_AXIS_GROUP_STATE
TrackingDynamicCS		BOOL	TRUE/FALSE		Active movement is commanded by a dynamic coordination system
InSync		BOOL	TRUE/FALSE		Only used for dynamic tracking: For continuous path movement, this mark indicates that the current position belongs to the specified path. For PTP movement, it means that the current position is equal to the target position of the movement
ActiveMovementId		SMC_Movement_Id			The identifier of the active action. Id of 0 means no activity. The identifier of the action. Mobile will never receive the 0 identifier. Each movement of the axis group will receive a unique identifier. (Unless overflow occurs, at least 2^64 moves have passed.)

Chapter VI I Simulation and Debugging

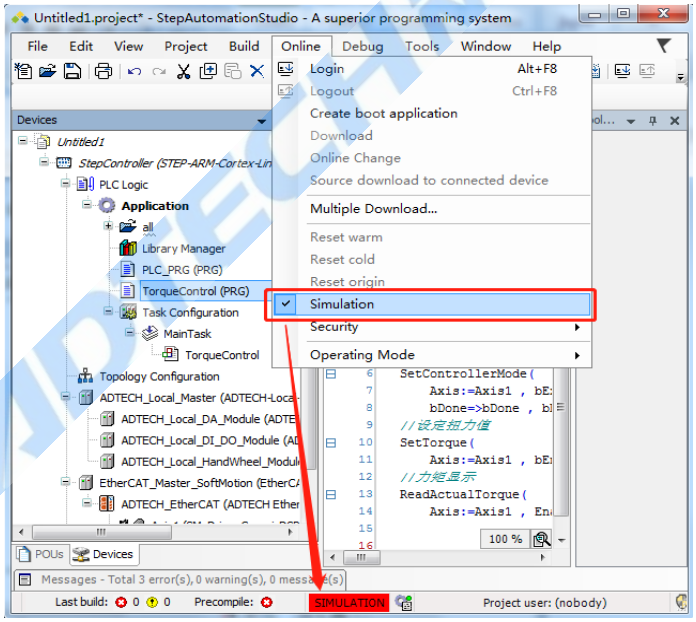
一、Simulation and debugging of AMC1600 controller

1. Turn on offline simulation

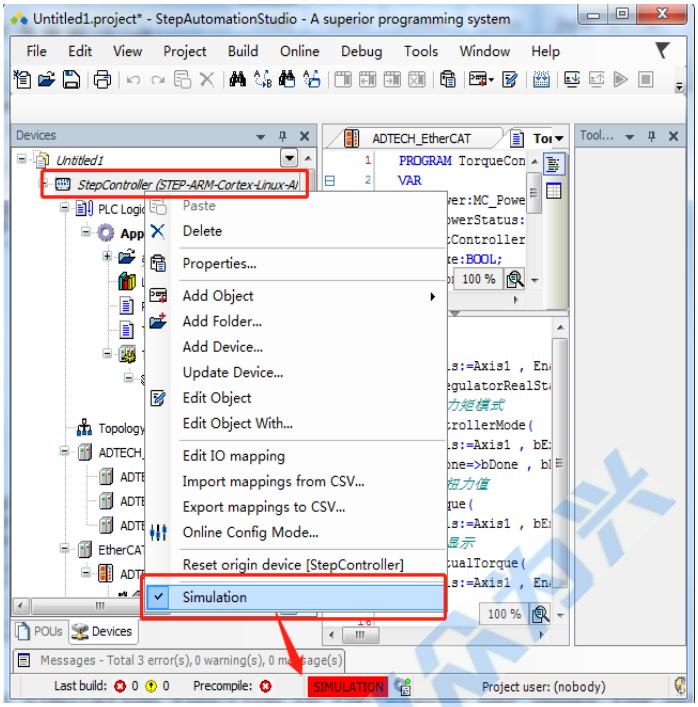
When the user is programming and debugging, if there is no AMC1600 controller hardware, the CODESYS programming software can be used to simulate and debug the user's program. The method to start the simulation is as follows:

- There are two ways to turn on the simulation function:

  1. In the menu bar Online->Simulation, turn on the simulation function.



2. Right-click Device in the project to turn on the simulation function.



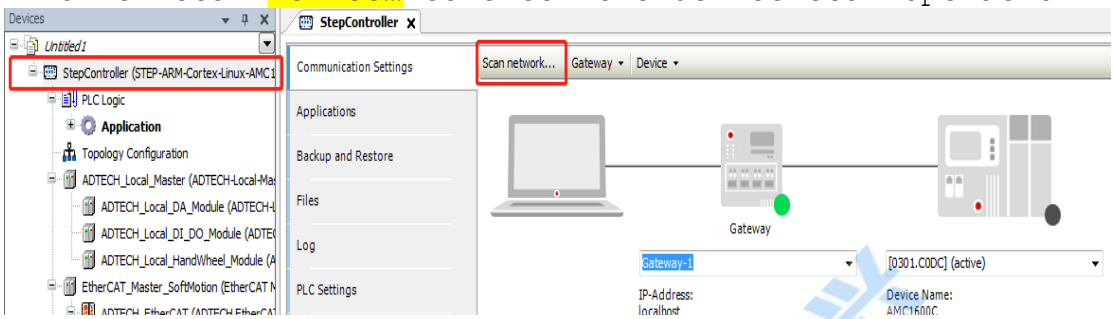
Turn on the simulation function in the status bar under the programming software, it will display the current simulation state

➢ Device option status, in simulation mode.

In the simulation state, the user program can also be compiled and "logged in" to the controller, but the user loads the program into the PC simulator to run. The operation of user program monitoring, modification and parameter change can be carried out as if the controller is actually connected. The network bus cannot be simulated.

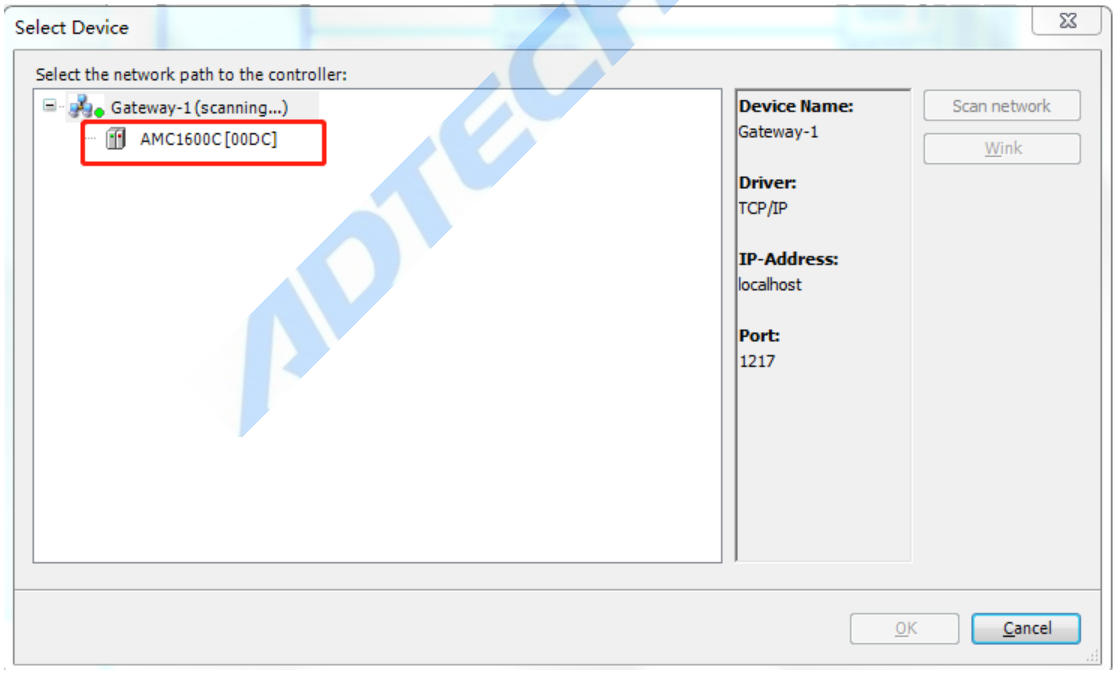
2.Open online debugging function (connect CODESYS to the device)

- Double-click in the project, Device enters the device setting page。
- Click Scan Device… to enter the device scan operation。

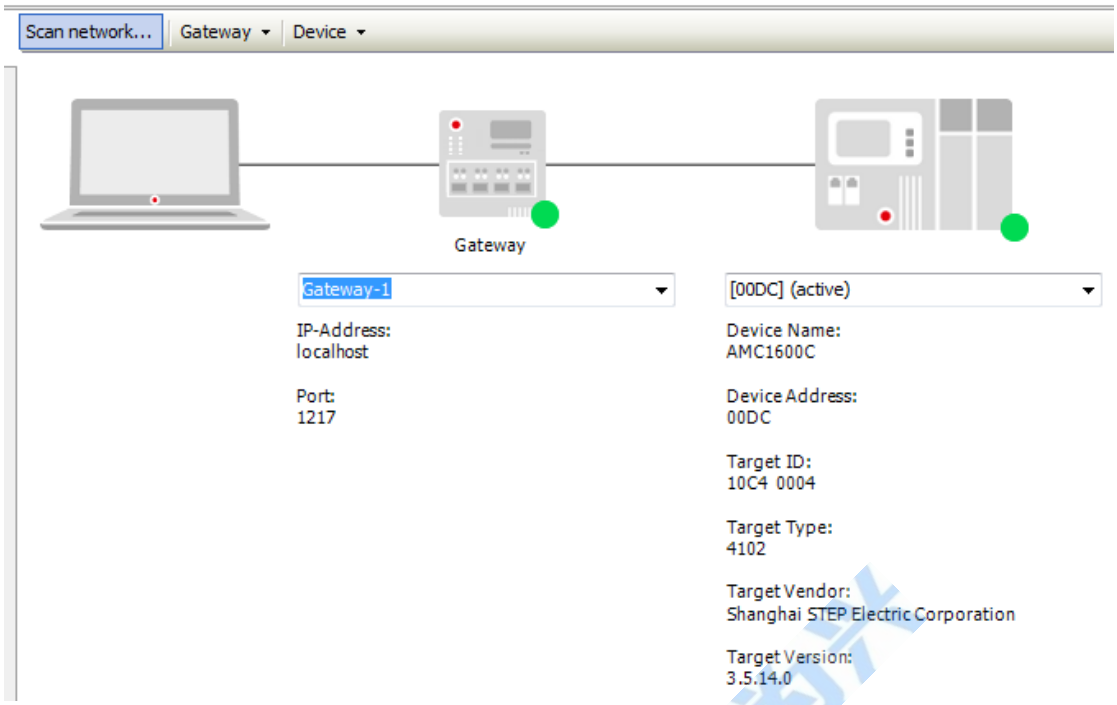


Note: To connect to the real device, the simulation mode must be closed before it can be used normally.

Click on the appropriate device, press the OK button, and enter the connection state



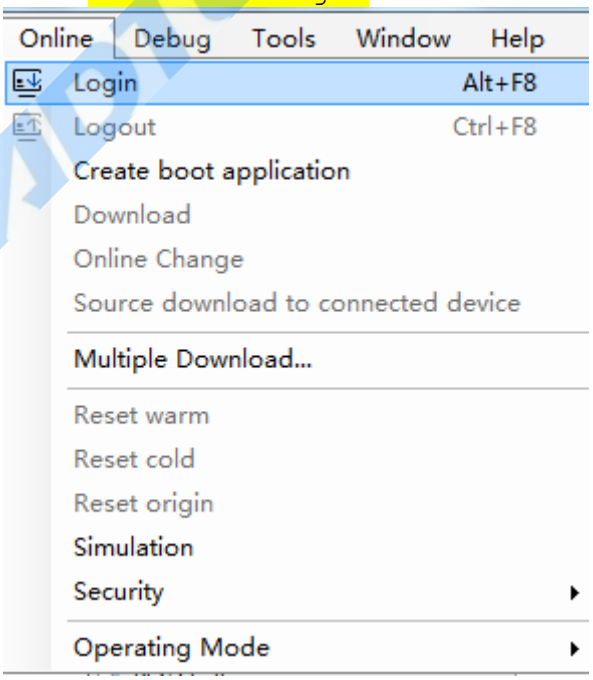
Note: If there is device access in the local area network, the appropriate device options will appear under this window。



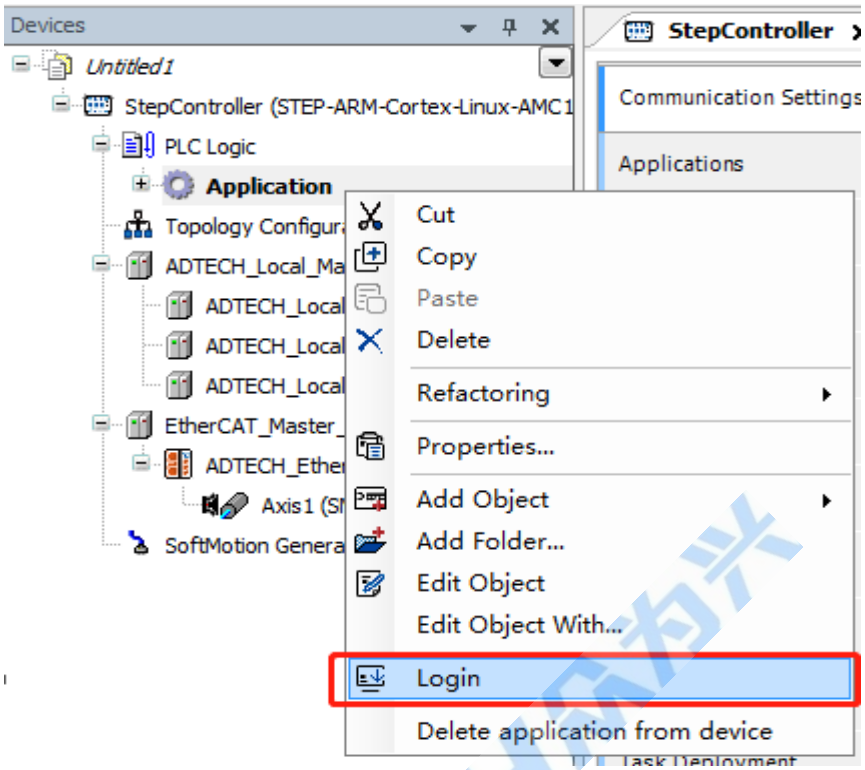
3.How to debug

3.1. Log in to the device

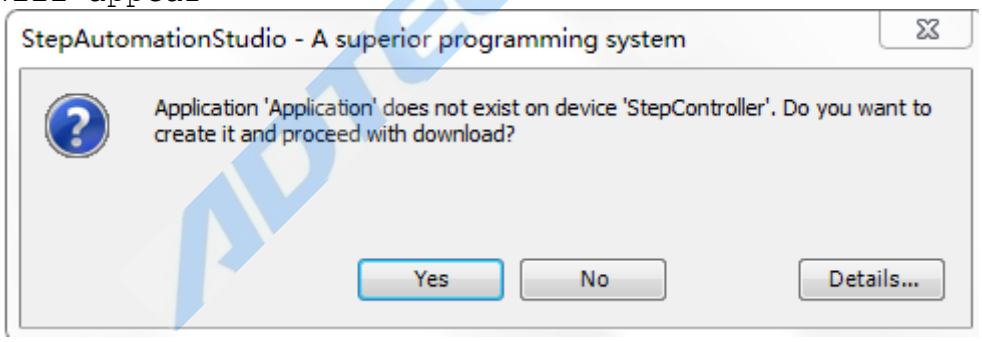
- In the menu bar Online -> Login to connect to the device.



➢ Right-click Application in the project and select Login to connect to the device.

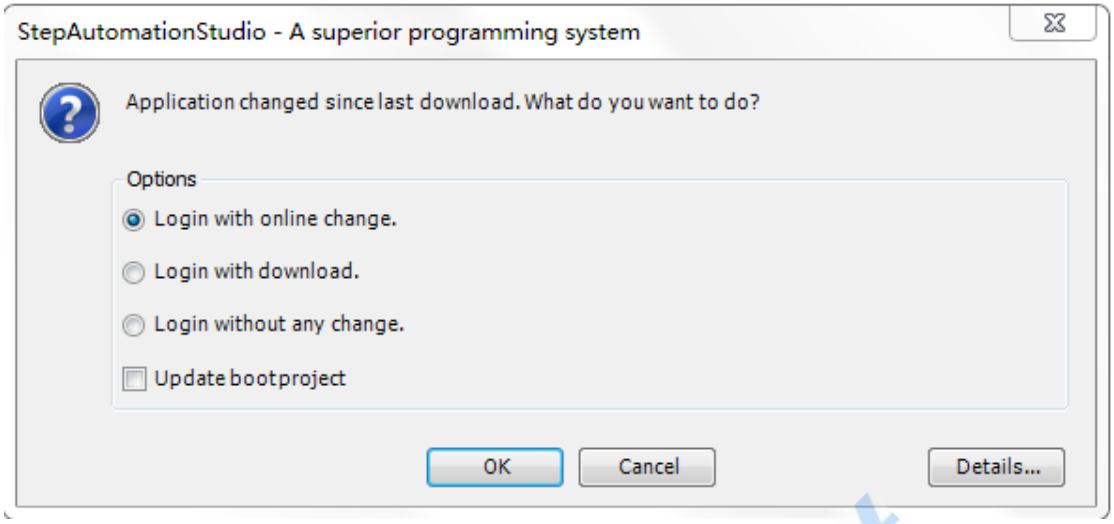


After logging in to is selected,
If there is no program in the device, the following options will appear



Choose Yes, download the program to the controller。

If there is a program in the device and the current program has been modified, the following options will appear



Option description:

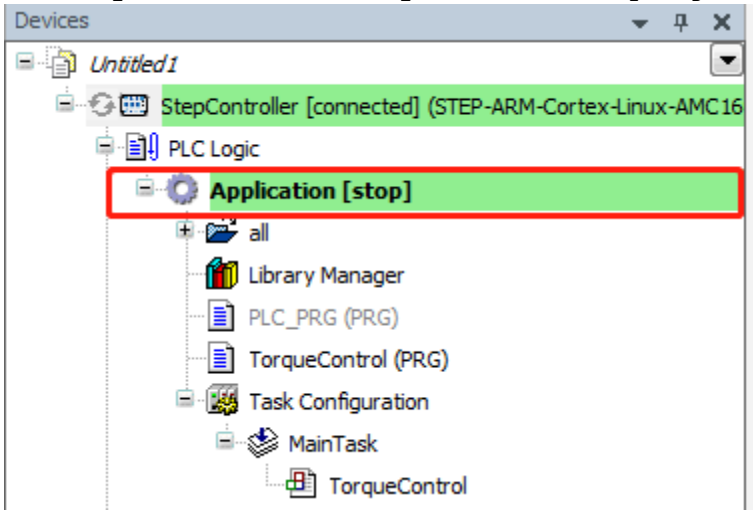
Login-online modification, when the current controller program is running, download the current program without stopping。

Log in and download, it will stop the program running in the controller first, after downloading the current program。

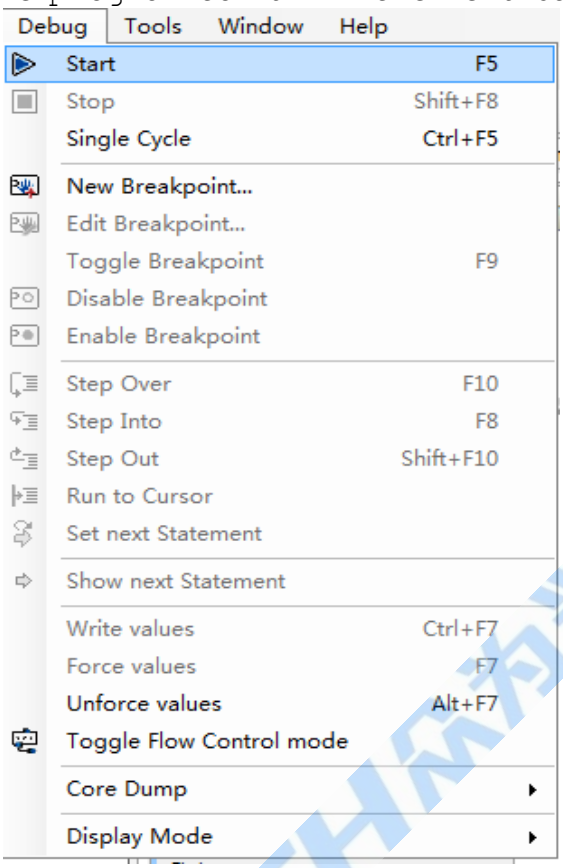
Login-do not make any modification, do not download the currently modified program。

3.2. Start the program to run

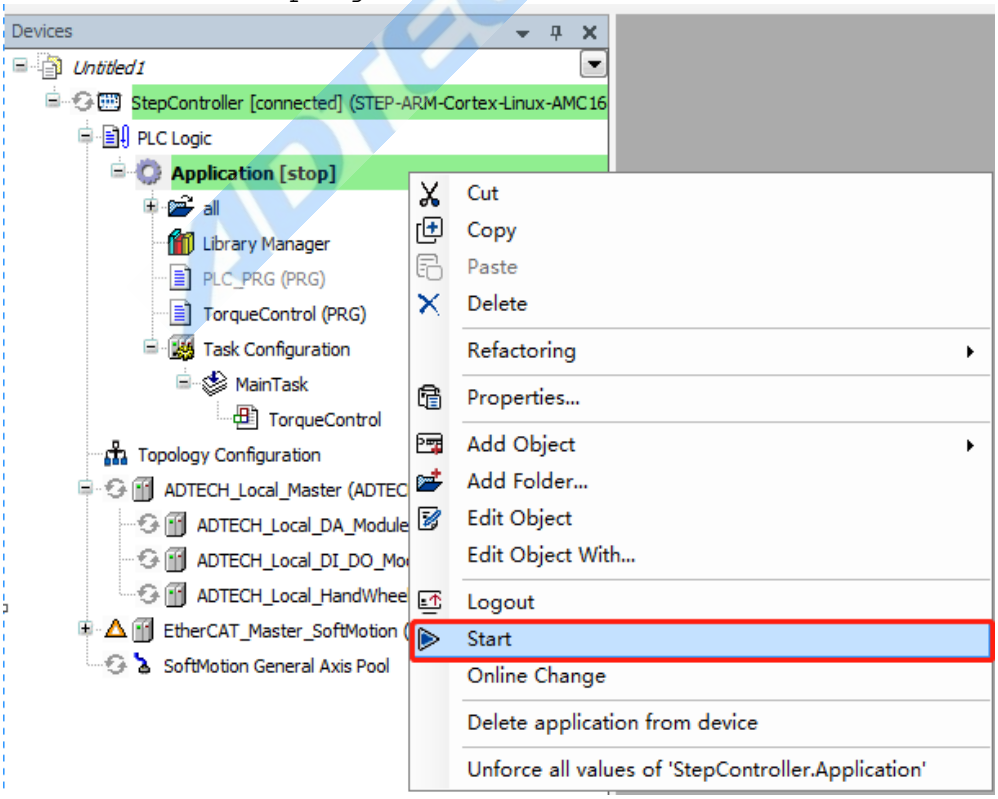
After logging in to the device, if the program is in a stopped state, you can manually start the program to run。



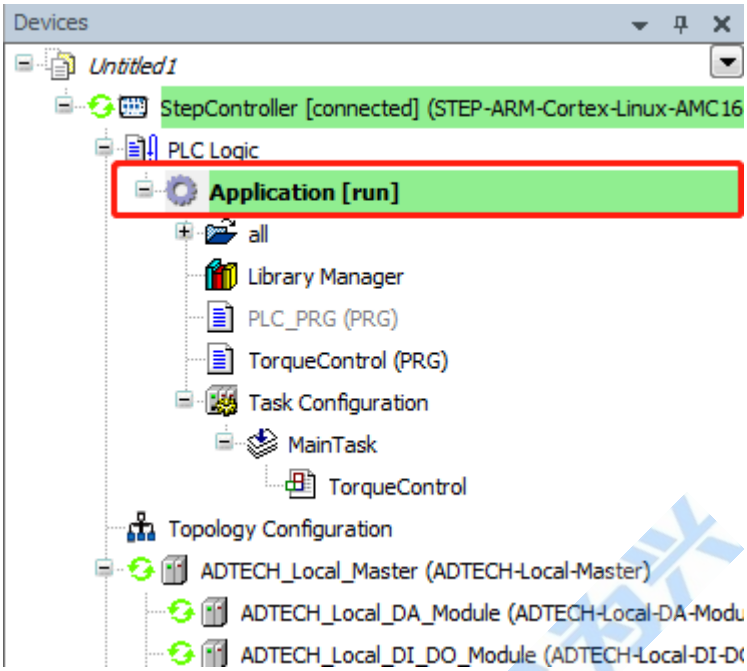
You can start the program to run in the menu bar Debug->Start



In the project, right-click Application and select Start, you can start the program to run



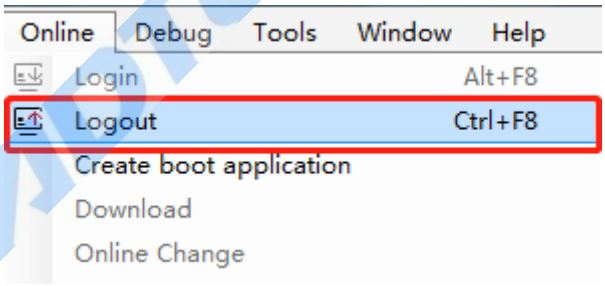
After the program is started, there will be a status display as shown below.



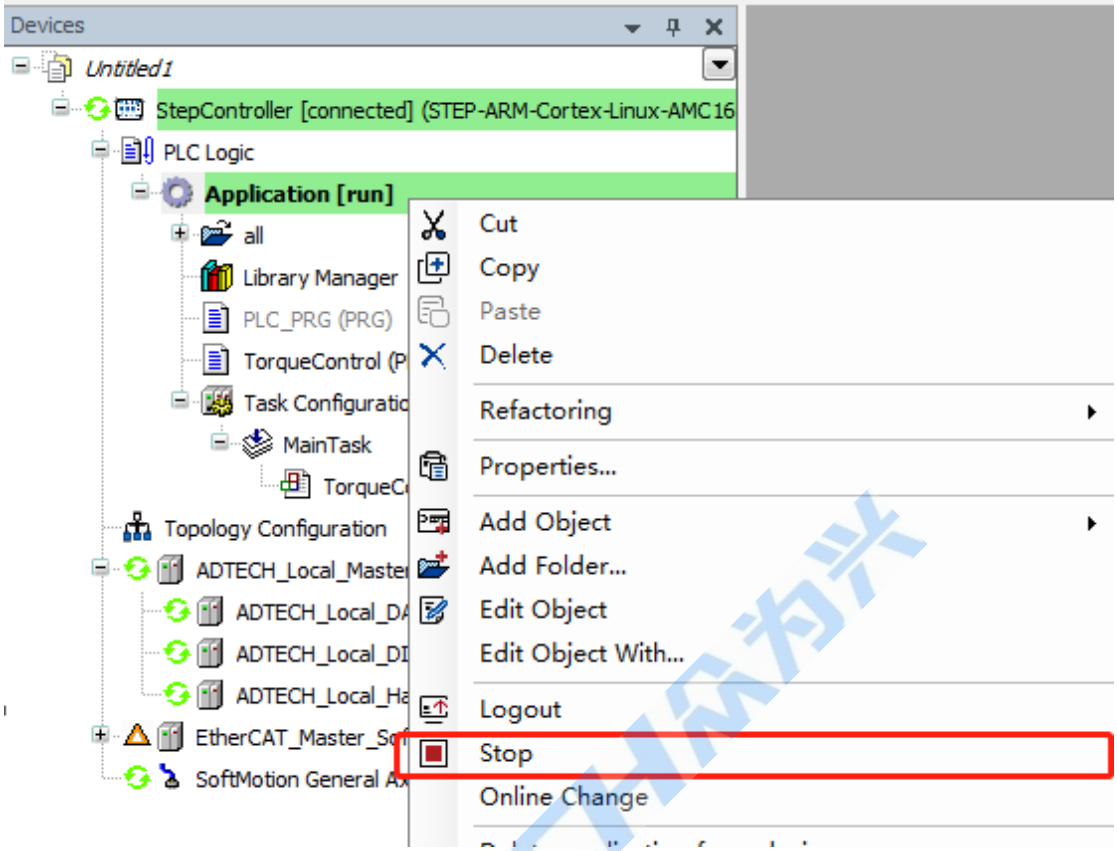
3.3. Stop the program

During the running of the program, you can manually stop the running of the program。

You can stop the program running in the menu bar Debug->Stop。

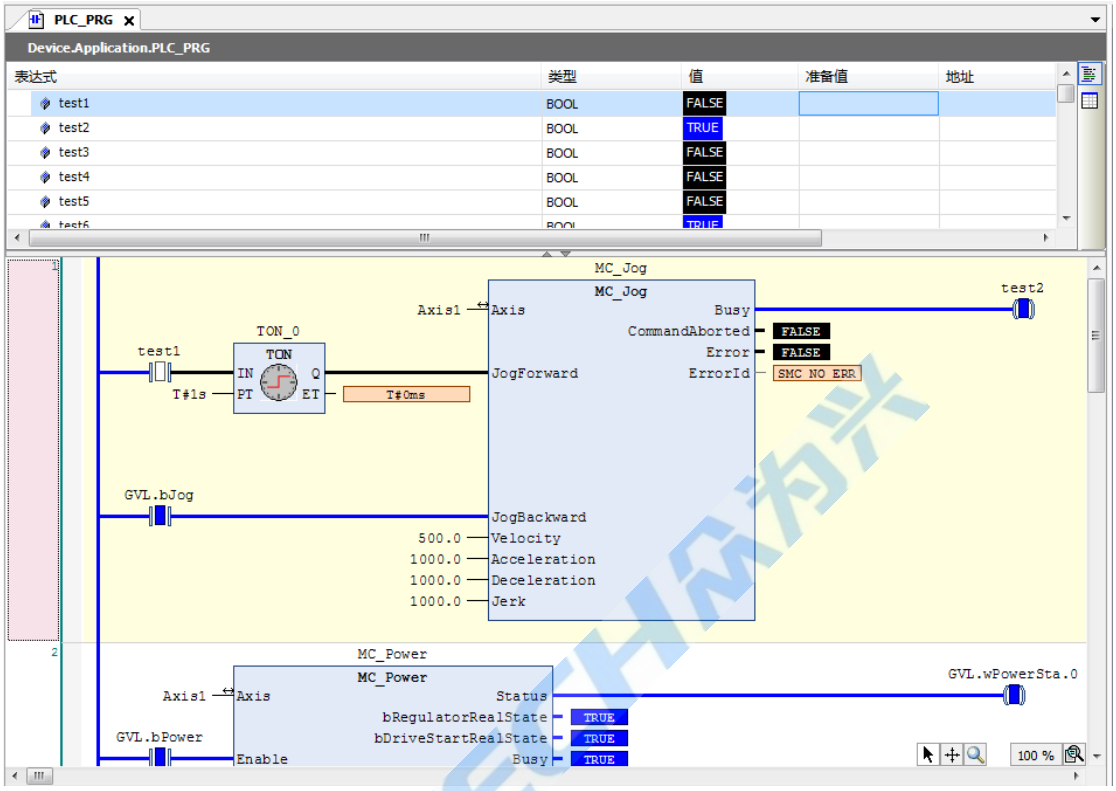


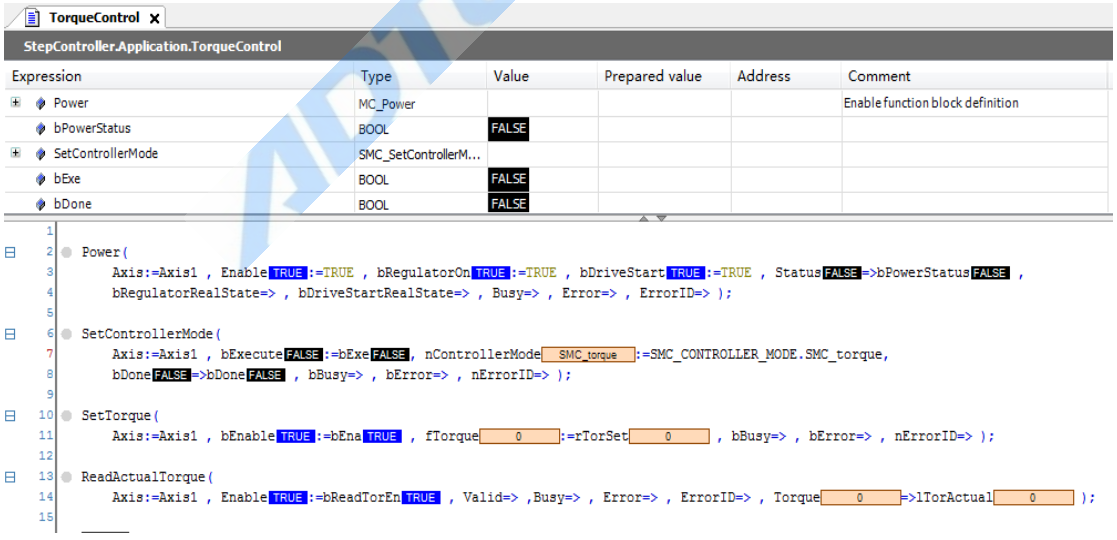
In the project, right-click Application and select Stop to stop the program.



3.4. Program (POU) debugging

3.4.1. Program in debug mode





3.4.2.How to modify variable values online

There are two ways to debug the program (POU)。

➢ Modify through the variable table。

Expression	Type	Value	Prepared value	Address	Comment
test1	BOOL	FALSE	TRUE		
test2	BOOL	TRUE	FALSE		
test3	BOOL	FALSE	TRUE		
test4	BOOL	FALSE			
test5	BOOL	TRUE			
test6	BOOL	FALSE			
test7	BOOL	FALSE			

which Value —— List as the value of the current variable。

Prepared value —— Listed as the current value to be modified。

How to set the preparation value:

BOOL Boolean data only need to click the corresponding row of the table, the system will automatically change the preparation value. There are 3 kinds of BOOL preparation values, TRUE-FALSE-empty. When the mouse is clicked, the preparation values will appear in these 3 states in cycles。

For other types of data, you can prepare values by clicking the corresponding row of the table. When the cursor appears in the table, you can use the keyboard to enter the value that needs to be modified。

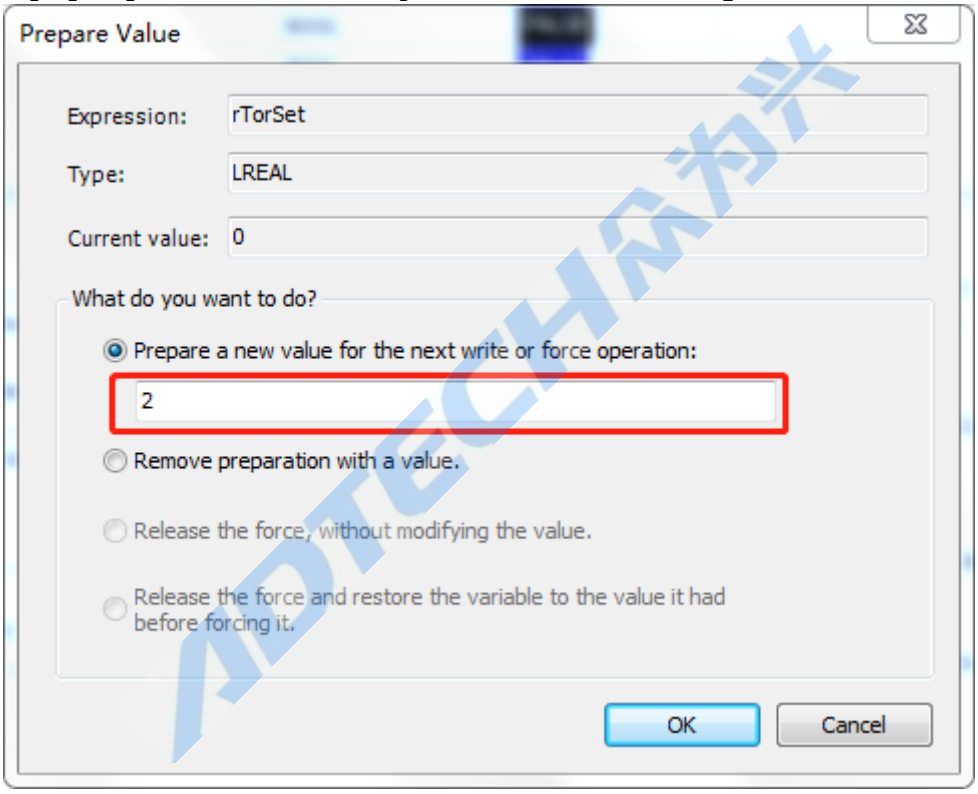
➢ Modify through the program。

Can be directly in the program, double-click the variable that needs to be modified。

At this time, if the data is of BOOL type, the following display will appear。

```
1  GVL.bPower TRUE := TRUE;
2  IF GVL.wPowerSta 7 = 2#00000111 THEN
3      GVL.bJog TRUE <FALSE> := TRUE;
4
5      TON(IN TRUE :=TRUE, PT T#500ms :=T#500MS);
6      IF TON.Q FALSE THEN
7          TON(IN TRUE :=FALSE);
```

If it is of non-BOOL type, double-clicking the variable will pop up the following window to modify the value。

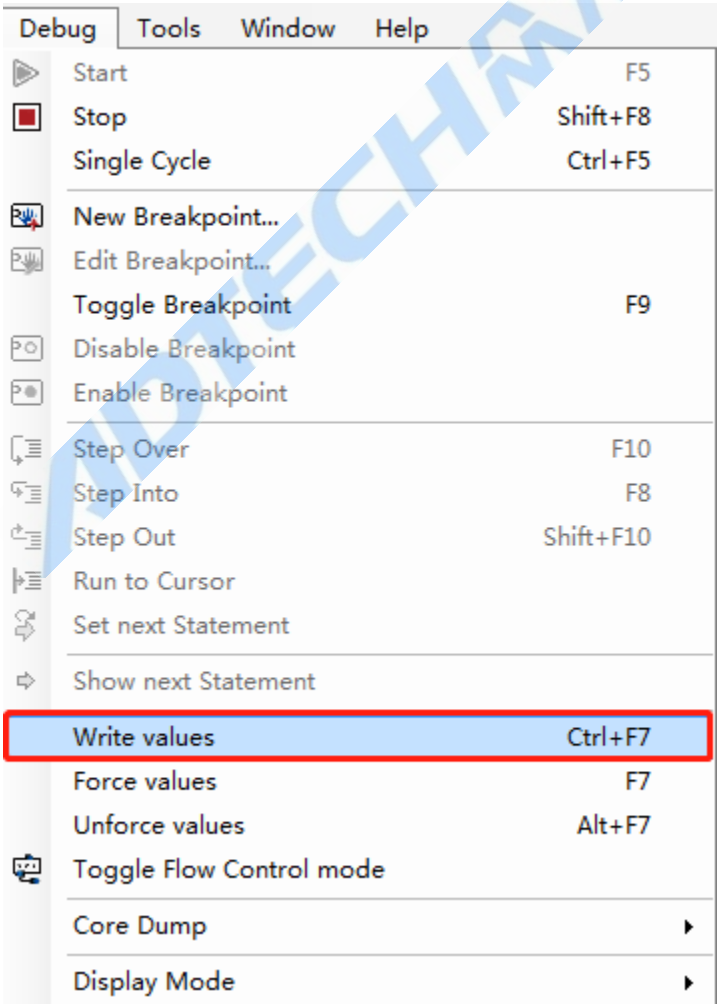


After setting, the program status is modified as follows。

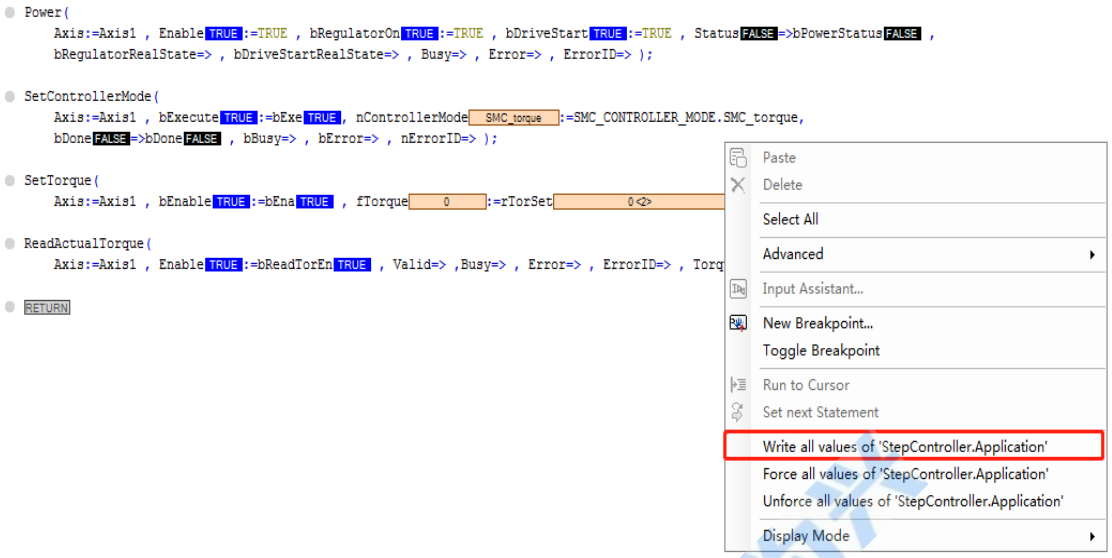
```
Power(
    Axis:=Axis1 , Enable TRUE :=TRUE , bRegulatorOn TRUE :=TRUE , bDriveStart TRUE :=TRUE , Status FALSE =>b
    bRegulatorRealState=> , bDriveStartRealState=> , Busy=> , Error=> , ErrorID=> );

SetControllerMode(
    Axis:=Axis1 , bExecute TRUE :=bExe TRUE , nControllerMode SMC_torque :=SMC_CONTROLLER_MODE.SMC_torque,
    bDone FALSE =>bDone FALSE , bBusy=> , bError=> , nErrorID=> );

SetTorque(
    Axis:=Axis1 , bEnable TRUE :=bEna TRUE   fTorque 0 :=rTorSet 0<2> , bBusy=>

ReadActualTorque(
    Axis:=Axis1 , Enable TRUE :=bReadTorEn TRUE , Valid=> ,Busy=> , Error=> , ErrorID=> , Torque 0

RETURN
```

- The final completion is completed by writing the prepared value, and the variable assignment。

  In the menu bar Debug->Write Value (shortcut key Ctrl + F7), write the prepared value into the current value。



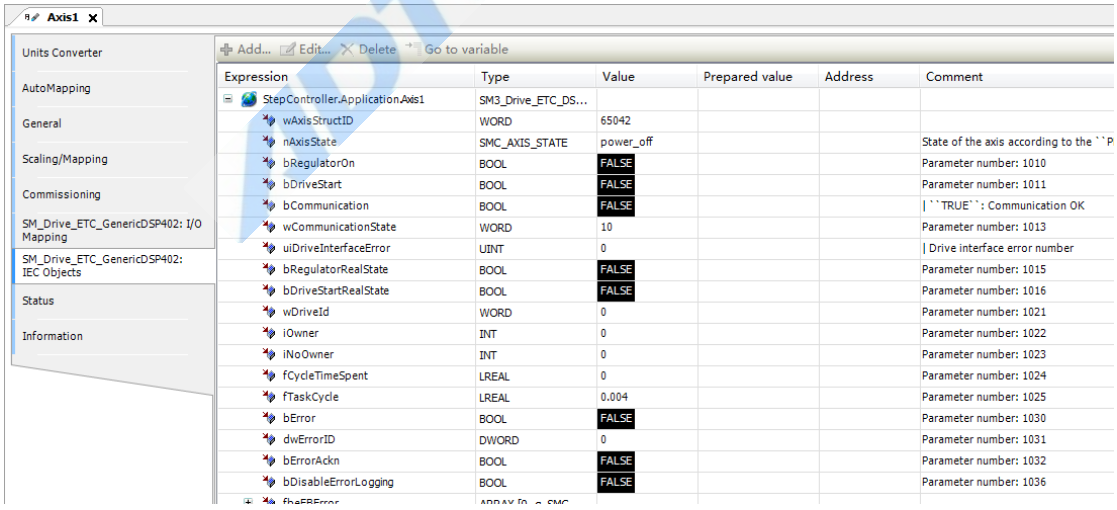
Directly in the variable table or program area, right click and write all values。



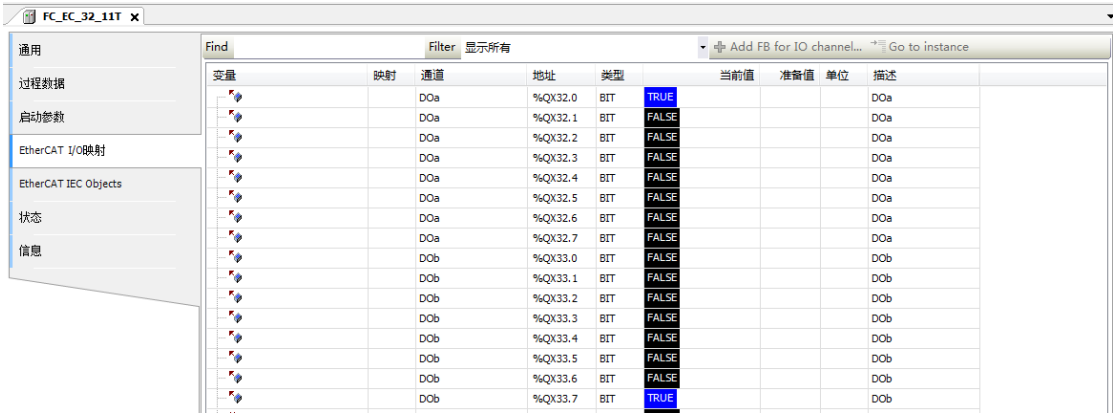
3.5. Debugging of other equipment

In online mode, you can open the device window to modify part of the data in the device that can be used for online debugging。

- Axis equipment, debugging



➢ IO device debugging



➢ EtherCat Bus data

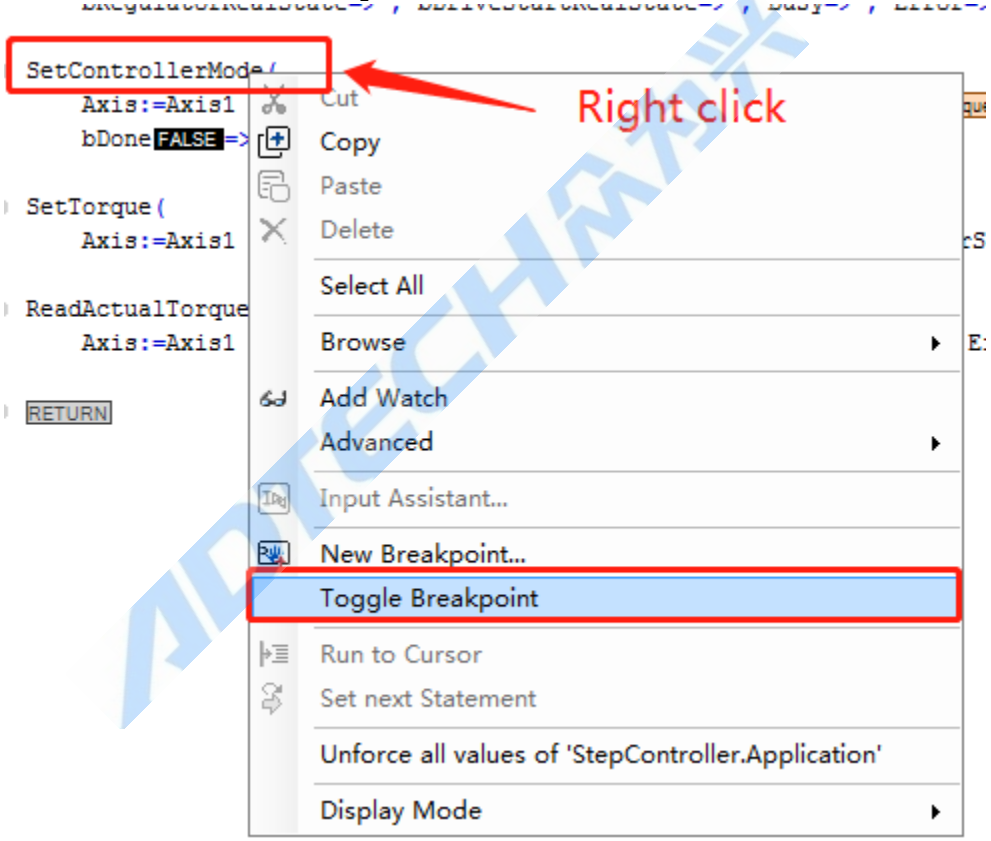
4.Breakpoint debugging

In online mode, you can set breakpoints on the program to debug the program.。

4.1. How to create a breakpoint

- In online mode, click the mouse on the program where the breakpoint needs to be set. And right click, select 'set or clear breakpoint'。

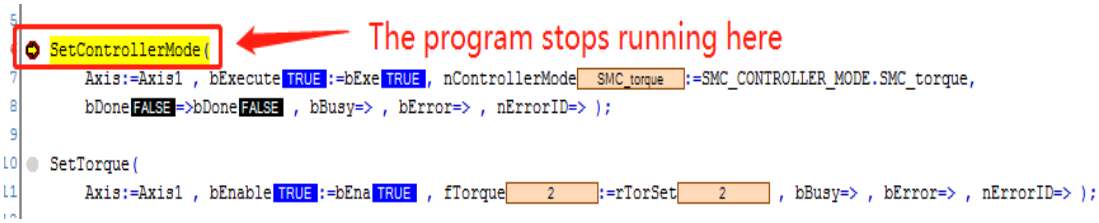
  ST: Right-click to select the number of lines where you want to set a breakpoint。

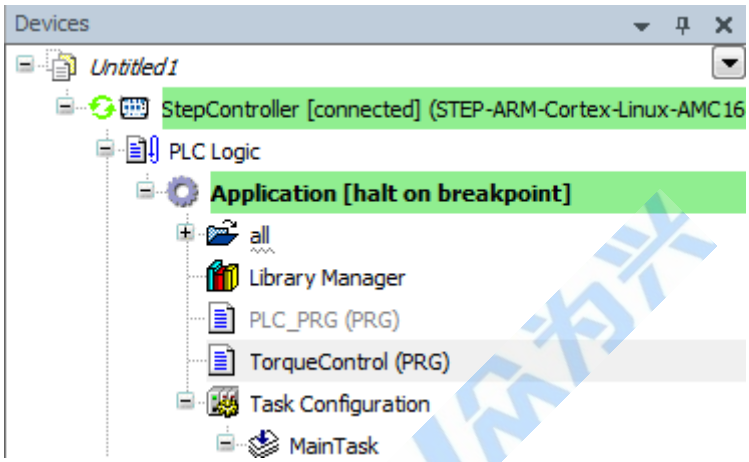


Note:

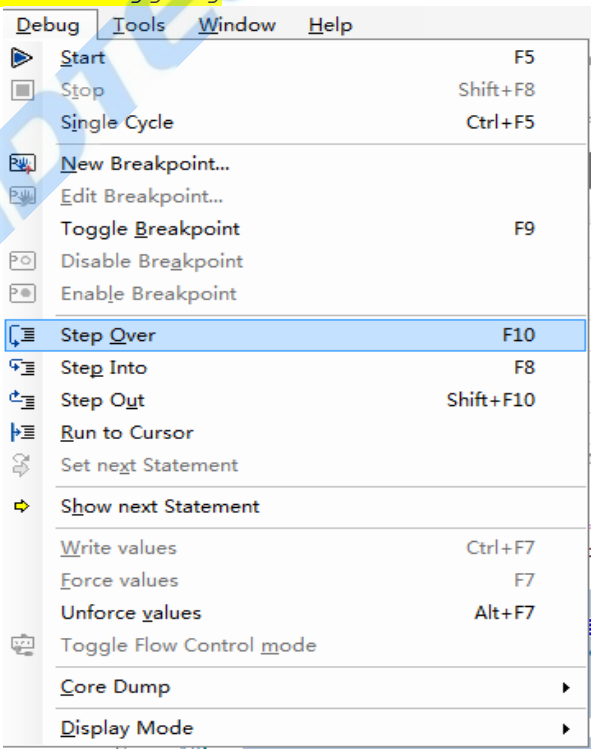
1. If it is at the breakpoint, select 'Set or Clear Breakpoint' again to clear the current breakpoint. at this time, If the program has stopped at the breakpoint, you can restart the program to run.
2. Set breakpoints can also be set in the menu bar -> debugging。

➢ After the breakpoint is set successfully, the program will stop running when it reaches the breakpoint。





➢ At this time, you can execute debugging commands through the menu bar -> debugging window。



Start: End the current cycle breakpoint command, enter the next cycle to run。

Disable breakpoint: Do not use this breakpoint temporarily.

Skip (shortcut key F10): Continue to execute in the current program without jumping into the function block or function block.

Jump into (shortcut key F8): jump into the function block or function block of the current line.

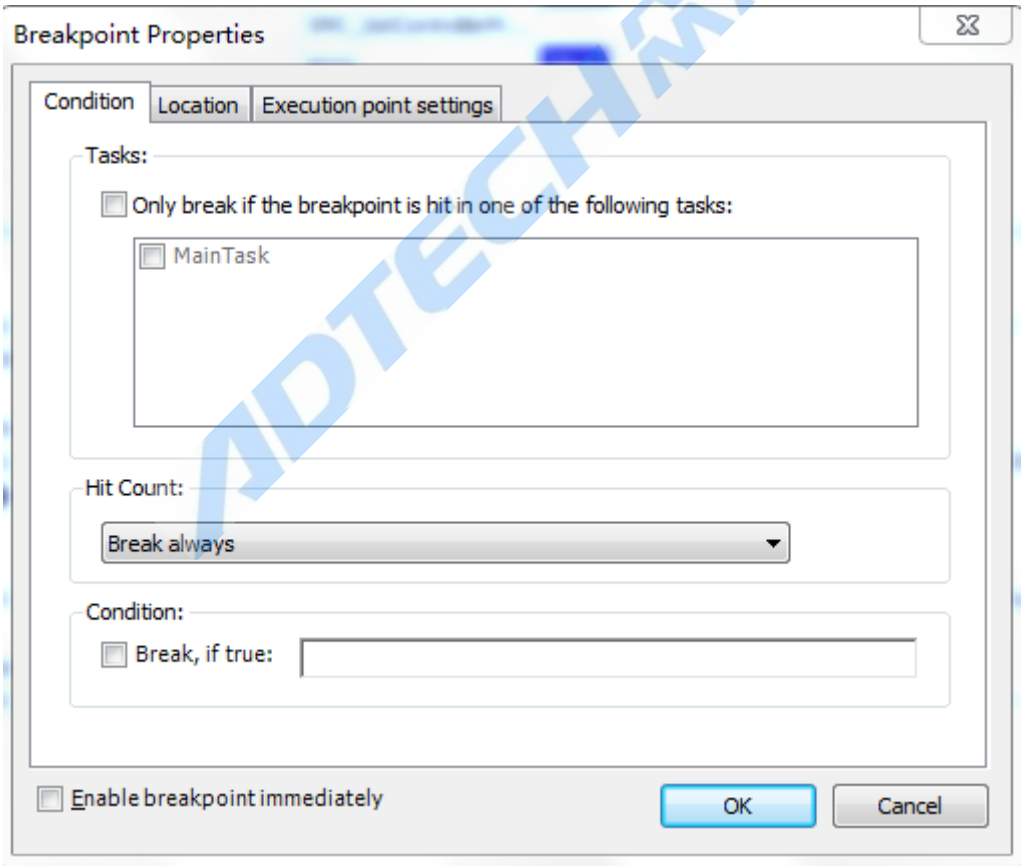
Jump out (shortcut key Shift+F10): Jump out of the current function block or function block。

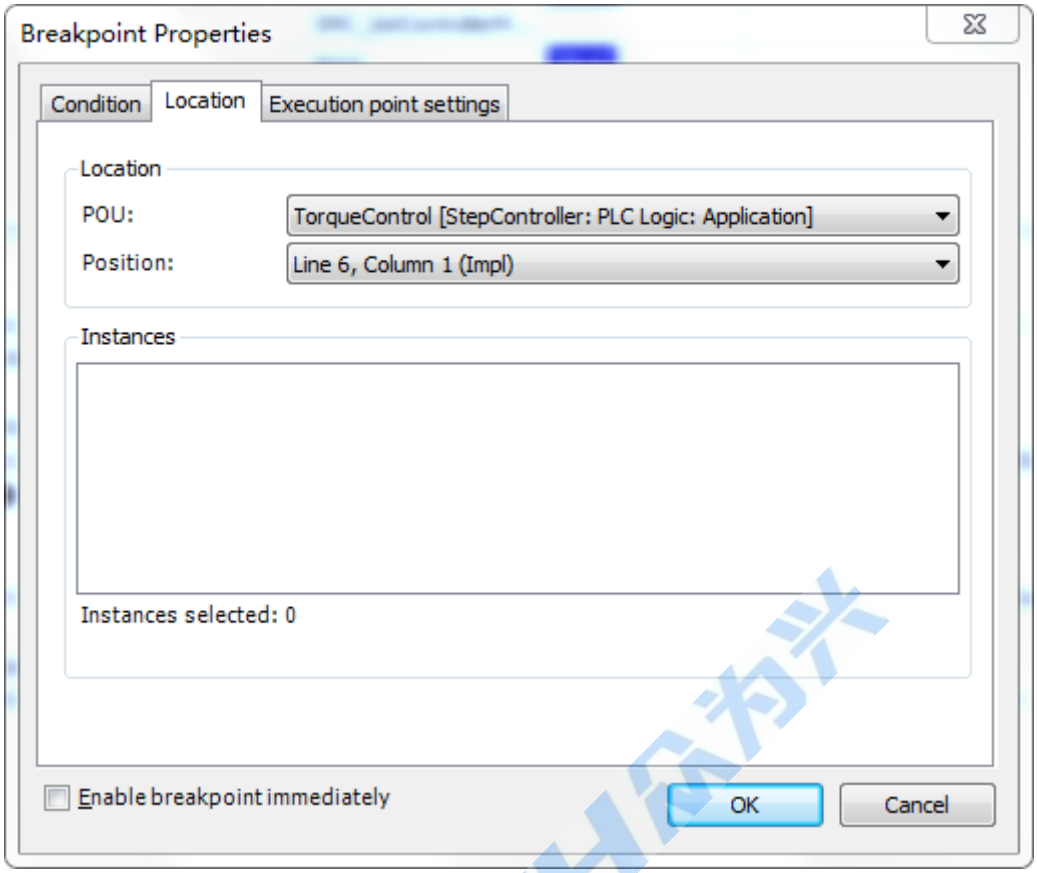
Run to the cursor: The program will run to the current mouse cursor (if there is a breakpoint at the cursor, it will

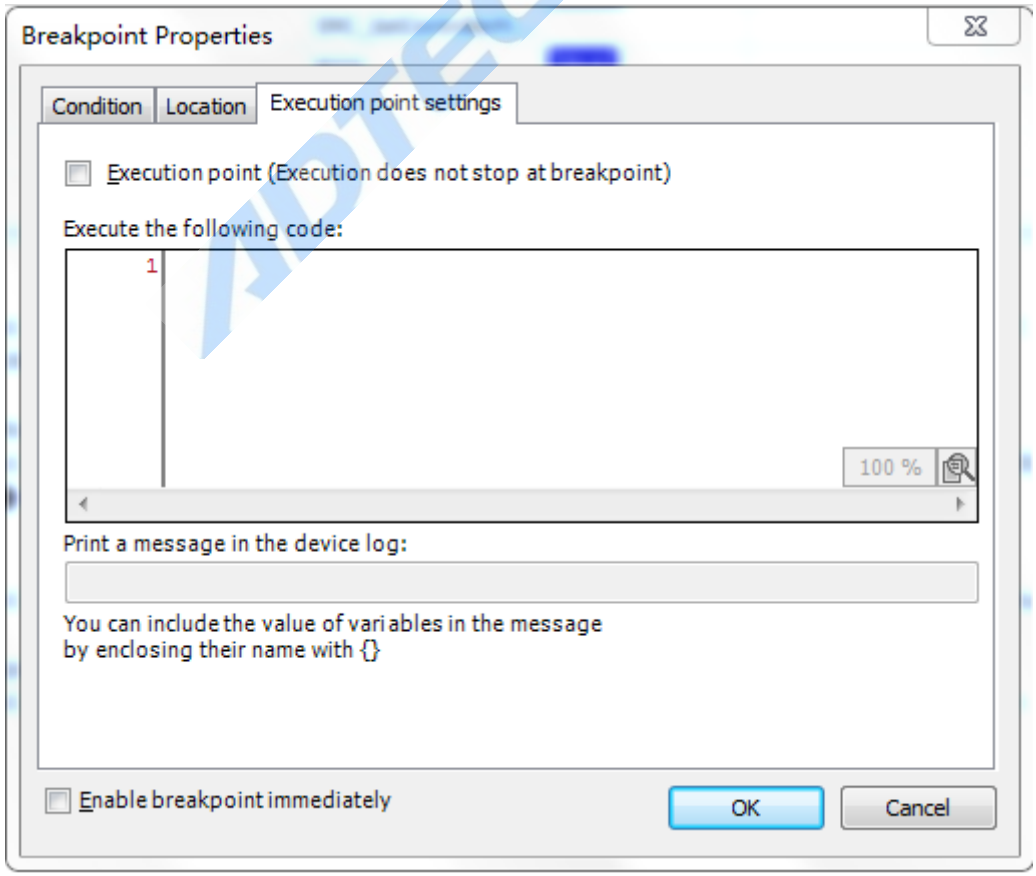
Execute, the breakpoint before the cursor will not be executed).4.2.

New/Edit Breakpoint

When using the new/edit breakpoint, an operation window as shown in the figure below will appear, where the user can further set the relevant properties of the breakpoint.




Breakpoint Properties
Condition
Location
Execution point settings
Location
POU:
TorqueControl [StepController: PLC Logic: Application]
Position:
Line 6, Column 1 (Impl)
Instances
Instances selected: 0
Enable breakpoint immediately
OK
Cancel


Breakpoint Properties
Condition
Location
Execution point settings
Execution point (Execution does not stop at breakpoint)
Execute the following code:
1
100 %
Print a message in the device log:
You can include the value of variables in the message
by enclosing their name with {}
Enable breakpoint immediately
OK
Cancel

Appendix

1. Type of data

1.1 MC_TA_REF

This structure describes a trajectory，Can be used later by MC_AccelerationProfile carried out。

Name	Type	Initial value	Description
Number_of_pairs	INT	0	The number of segments of the contour path. This variable is not used, it only appears for compatibility. Please use **MC_AccelerationProfile. "ArraySize"** instead.
IsAbsolute	BOOL	TRUE	TRUE: The acceleration value is interpreted as an absolute value (not a relative value). FALSE: Relative
MC_TA_Array	ARRAY [1..100] OF **SMC_TA**		Array of time/acceleration values

1.2 SMC_TA

Structure
This structure defines a point on the trajectory defined by **MC_TA_REF** and consists of a pair of time and acceleration values。

Name	Type	Initial value	Description
delta_time	TIME	TIME#0ms	Acceleration time: the time difference between the current point and the last point
acceleration	LREAL	0	(Absolute/relative) current acceleration value at this point

1.3 MC_TP_REF

Structure
This structure describes a trajectory，Can be used later by MC_PositionProfile carried out。

- Input and output variables

Name	Type	Initial value	Description
Number_of_pairs	INT	0	This variable is not used, it only appears for compatibility. Please use MC_PositionProfile. "ArraySize" instead.

IsAbsolute	BOOL	TRUE	TRUE: The position is absolute (not relative). FALSE: position is relative
MC_TA_Array	ARRAY [1..100] OF **SMC_TP**		Time/location array

1.4 SMC_TP

Structure
This structure defines a point on the trajectory defined by MC_TP_REF and consists of a pair of time and position values.

Name	Type	Initial value	Description
delta_time	TIME	TIME#0ms	Time difference between current point and last point
position	LREAL	0	(Absolute/relative) position at this point

1.5 MC_TV_REF

Structure
This structure describes a trajectory，Can be done later by MC_VelocityProfile carried out。

Name	Type	Initial value	Description
Number_of_pairs	INT	0	This variable is not used, it only appears for compatibility. Please use MC_VelocityProfile. "ArraySize '' instead.
IsAbsolute	BOOL	TRUE	TRUE: The speed value is absolute, not relative. FALSE: The speed value is relative
MC_TA_Array	ARRAY [1..100] OF **SMC_TV**		Time/speed array

1.6 SMC_TV

Structure
This structure defines a point on the trajectory defined by MC_TV_REF and consists of a pair of time and speed values.

Name	Type	Initial value	Decsription
delta_time	TIME	TIME#0ms	Time difference between current point and last point
velocity	LREAL	0	Speed at this point

1.7 TRIGGER_REF

This structure describes a trigger input. The axis driver used will determine which hardware trigger corresponds to the number of triggers.

Name	Type	Initial value	Description
iTriggerNumber	INT	-1	-1 is used when bFastLatching=FALSE; Trigger channel number (only used when bFastLatching=TRUE) In the drive mode, which one of the functions is locked 0: The rising edge of probe 1 is latched 1: Probe 1 falling edge latch 2: The rising edge of probe 2 is latched 3: Probe 2 falling edge latch
bFastLatching	BOOL	TRUE	TRUE: The latching is done in the drive (precise). (Drive mode) FALSE: The latching is done using bInput in the loop of the motion task (inaccurate) (controller mode)
bInput	BOOL		When bFastLatching = FALSE, the signal is triggered by the controller Input
bActive	BOOL	FALSE	Triggered effective signal

1.8 MC_CAMSWITCH_REF

名称	类型	初值	说明
NoOfSwitches	BYTE	0	开关数
CamSwitchPtr	POINTER TO MC_CAMSWITCH_TR	0	MC_CAMSWITCH_TR 地址

1.9 MC_CAMSWITCH_TR

Name	Type	Initial value	Description
TrackNumber	INT		Track number (1-32)
FirstOnPosition	LREAL		Switch ON position
LastOnPosition	LREAL		Switch OFF position
AxisDirection	INT		-1: Only the reverse transmission is ON/OFF; 0: 2 directions output ON/OFF; 1: Only positive transmission is ON/OFF
CamSwitchMode	INT		0: Both output ON and OFF are controlled by position; 1: Only when the output is ON, it is controlled by the position, when it is OFF, it depends on the setting of Duration.
Duration	TIME		When CamSwitchMode=1, ON hold time

1.10 MC_CAM_ID

Name	Type	Initial value	Description
pCT	POINTER TO BYTE		Define the structure of the CAM table to use, connect the MC_CamTableSelect output and MC_CamIn input interface structure
Periodic	BOOL	FALSE	
MasterAbsolute	BOOL	FALSE	
SlaveAbsolute	BOOL	FALSE	
StartMaster	LREAL	0	
EndMaster	LREAL	0	
StartSlave	LREAL	0	
EndSlave	LREAL	0	
byCompatibilityMode	BYTE		

1.11 SMC_HOMING_MODE

Enumerated type
Define the homing sequence。For function blocks SMC_Homing。

- Input and output variables

Name	Initial value	Description
FAST_BSLOW_S_STOP	0	Move to the origin switch with rapid speed according to the set direction, and move to the origin switch at a slow speed in the reverse direction after hitting the origin. After leaving, execute the "set position" (fHomePosition) first, and then execute the stop。
FAST_BSLOW_STOP_S	1	Move to the origin switch with rapid speed according to the set direction, and move to the origin switch at a slow speed in the reverse direction after hitting the origin. After leaving, execute the stop first, and then execute the "set position" (fHomePosition).
FAST_BSLOW_I_S_STOP	2	Move to the origin switch at a rapid speed according to the set direction, and move to the origin switch at a slow speed in the reverse direction after hitting the origin. Wait for the index pulse (bIndexOccured), and then execute the "set position" first; then execute the stop。
FAST_SLOW_S_STOP	4	Move to the origin switch at a rapid speed according to the set direction, and move to the origin switch at a slow speed in the original direction after hitting the origin. First execute the "set position", and then stop。
FAST_SLOW_STOP_S	5	Move to the origin switch at a rapid speed according to the set direction, and move to the origin switch at a slow speed according to the original direction after it hits the origin. The stop is executed first, and then the "set position" is executed。
FAST_SLOW_I_S_STOP	6	Move to the origin switch at a rapid speed according to the set direction, and move to the origin switch at a slow speed in the original direction after it hits the origin. Wait for the index pulse (bIndexOccured); execute "set position" first, and then execute stop。

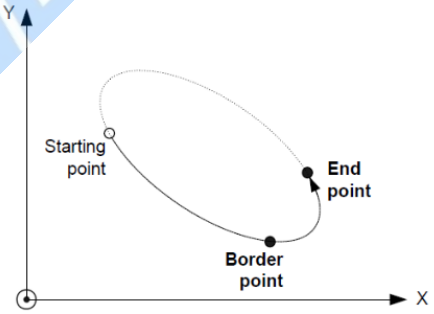
1.12 MC_Direction

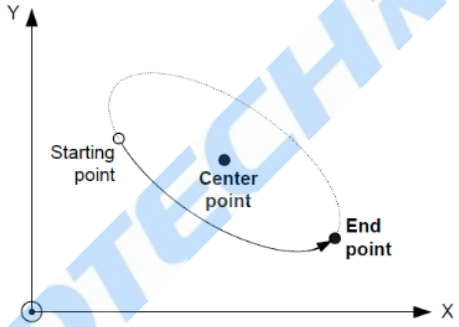
This global variable is part of the library **SM3_Basic**. It is used as input for several function blocks and specifies the direction of movement. Please note that not all modes are applicable to all function blocks and axis types (modular/limited)。

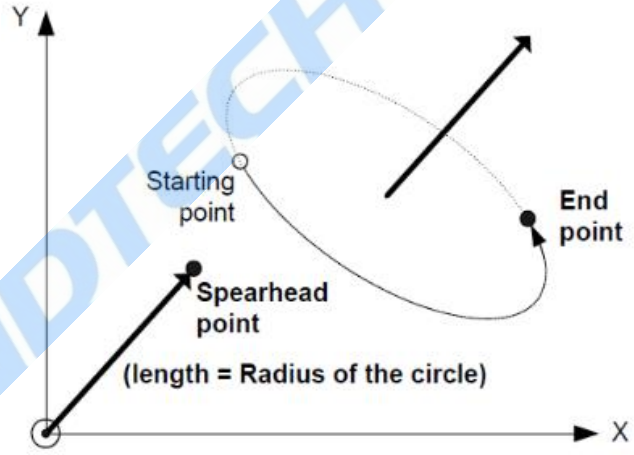
Name	Initial value	Description
fastest	3	The direction is automatically selected to reach the target position as quickly as possible (rotation axis only).
current	2	Keep the current direction to reach the target (rotation axis only).
positive	1	Move in the positive direction.
shortest	0	Select the direction based on the shortest distance (rotation axis only).
negative	-1	Move in the negative direction.

1.13 SMC_CIRC_MODE

Defines how to describe the arc。

Name	Mode	Description
BORDER	Boundary point mode	The user defines an end point and a boundary point (input auxiliary point) on the sector of the circle, which will be patrolled by the machine. (This mode does not consider input path selection) The advantage of this model is that the machine can usually reach the boundary point, that is, the machine can be taught. Mode: A single command is limited to an angle <2π, These three points (start point, end point and auxiliary point) cannot be collinear. 
CENTER	Center point mode	The user defines the end and center point of the circle (input auxiliary points). When using this mode, the input path selection defines which two possible arcs are used. The plane normal is the unit vector in the direction of (start-center) x (dest-center). The direction of the arc is selected according to the rules of the right thumb: the counterclockwise direction is the direction of bending along the axis of the thumb, which represents the normal vector.

		 limit: Limit angle <2π and ≠π in a single command; Due to conflicts with obstacles, the center point is usually not indicated. If the distance between the center point and the start point and the end point is not exactly the same (due to the limited accuracy of setting the center point, this is often the case), then the center point is projected onto the vertical bisector of the start point and the end point. Once the center point is too far (more than 1% of the radius), an error will be returned. 
RADIUS	Radius mode	The user defines the end point and vertical vector of the circle plane according to the rules of the right thumb: the counterclockwise direction is the finger bent along the thumb axis。  The normal vector is the input auxiliary point. In other words, the auxiliary verb is not a position but a direction vector. The radius of the circle is the length of the vector。 If the distance between the start and end points

		is not equal to the diameter, there may be two circles within a given radius。 The input path selection defines the direction of the arc. Then the system selects a circle with a shorter distance from the start point to the end point. This means that the total anglewill is at most π. If the movement is carried out in the ac-coordinate system, the auxiliary points will be converted into MCS usi。 If the start and end points are not on the plane defined by the normal vector, a spiral movement from the start point to the end point will occur. This means circular motion on the plane plus linear motion orthogonal to the plane. Inconvenient mode: limit angle <=π in a single command; The vertical vector must be calculated. Example: AuxPoint = (50, 0, 0) -> The circle is on a plane parallel to the yzplane with a radius of 50, and rotates around an axis parallel to the x-axis according to the rules of the right thumb (CoordSystem = MCS) 

1.14 SMC_POS_REF

Indicates the location of the TCP.
The position can be defined in the Cartesian coordinate system (X, Y, Z, A, B, C) or in the axis coordinate system (A0,...A5).

Name	Type	Description
a	TRAFO.AXISPOS_REF	Axis coordinates
c	MC_COORD_REF	Cartesian position
v	ARRAY [0..(SMC_RCNST.MAX_AXES - 1)] OF LREAL	The value of the array, the interpretation depends on the use of the coordinate system

1.15 SMC_COORD_SYSTEM

Coordinate system mode that represents the position

Name	Description
ACS	Axis coordinate system. This is not a Cartesian coordinate system. Instead, each axis of the axis group spans one dimension. In this coordinate system, a point is just the assignment of a value on each axis of the coordinate axis group.
MCS	Machine coordinate system. This coordinate system is directly related to the kinematics of the single-axis group. The position and direction of the MCS are determined by the kinematic transformation.
WCS	World coordinate system. This coordinate system is the common coordinate system of all axis groups that work together to complete the task
PCS_1	Product coordinate system 1. This coordinate system is connected to a product. It may be static or dynamic.
PCS_2	Product coordinate system 2. This coordinate system is connected to a product. It may be static or dynamic.
TCS	Tool coordinate system. When used with MoveDirect-, MoveLinear- or MoveCircular-FBs, the position data is interpreted according to the position and direction of the axis group when the movement starts, so relative movement and absolute movement are treated the same. In all other possible applications (such as plesmc_groupconvertposition), TCS means

1.16 MC_BUFFER_MODE

Buffer mode

Name	Description
Aborting	Start function block immediately (default mode)
Buffered	Start the function block after the last action in the current queue is completed
BlendingLow	The path mixes the minimum dynamic limit provided by the two functional blocks
BlendingPrevious	This path mixes the dynamic limits of the first functional block
BlendingNext	This path is mixed with the dynamic limits of the second functional block
BlendingHigh	This path mixes the maximum dynamic limit provided by the two functional blocks

1.17 MC_TRANSITION_MODE

Transition mode describes how to mix continuous motion commands。

Nmae	Description
TMNone	No mixing。
TMStartVelocity	Speed-based mixing. In the case of PTP motion, the cutoff point is determined by the two original motion trajectories previously calculated. The first transition parameter (selected between 0 and 1) specifies the portion of time during which the deceleration-/acceleration- ramp is cut off. In the case of CP movement, we simulated a deceleration ramp on the initial path

	pointing to the vertex and an acceleration ramp pointing to the vertex. TransitionParameter[0] is again used as a factor, but not in time, but in relation to the length of the path: a value of 1 indicates the beginning of the deceleration ramp
TMCornerDistance	Hybrid based on distance. The first conversion parameter specifies the length of the path to be cut before and outside the corner. In the case of CP motion, the second transition parameter specifies the minimum radius of curvature that the mixing element may have.

1.18 SMC_ORIENTATION_MODE

Determine how to interpolate the direction.
The direction mode describes how the direction of CP movement is interpolated.

Name	Description
GreatCircle	Interpolate from the starting direction to the target direction along the shortest path. Even if the start and target directions are in the work space, this interpolation mode may leave the work space.
Axis	The direction axes are inserted from their starting value to their target value in the axis space. This mode can be used to move through the direction of the singularity. Not all kinematic transformations support this model.

1.19 SMC_PTP_MOVEMENT_TYPE

Name	Description
Fast	Time optimal PTP movement
Path_Invariant	For this type of PTP movement, in addition to mixing with TransitionMode TMStartVelocity, the path in space is independent of the dynamic limits of all axes (speed, acceleration, deceleration, and jerk auxiliary limits and global limits). In TransitionMode, the tmcorner distance is independent of the type of blending (BlendingHigh / Low / Previous / Next). Stop or stop when not left (MC_GroupHalt / MC_GroupStop). Did not leave when interrupted and continued (MC_GroupInterrupt / MC_GroupContinue).

2.CiA402 common data object quick reference table

index (hex)	Sub-index (hex)	Name	Visit	Size	Unit	Setting range	Defaults	PDO mapping
603F	00	error code	RO	UINT16	-			TPDO
This object gives the latest fault code or warning code of the drive.								
6040	00	Control word	RW	UINT16	-	0~65535	0	TPDO
State guidance after the servo is powered on, command control in each servo mode Control Word Bit0- Servo ready 1-valid, 0-invalid Bit1- Connect the main circuit power 1-valid, 0-invalid Bit2- Quick stop 1-valid, 0-invalid Bit3- Servo operation 1-valid, 0-invalid Bit4~6- Related to each servo operation mode Bit7- Fault reset For resettable faults and warnings, perform a negative reset function. The rising edge is valid; keep at 1, other control commands are invalid Bit8- Pause Please refer to the object dictionary 605Dh for the pause mode in each mode Bit9~10-NA Reserved Bit11~15- Manufacturer custom reserved								
6041	00	Status word	RO	UINT16	-			TPDO
Reflect the running status of the servo drive Bit0- Servo without failure Bit1- Waiting to turn on the servo enable Bit2- Servo operation Bit3- fault Bit4- Connect the main circuit power Bit5- Quick shutdown Bit6- Servo ready Bit7- Warning Bit8- Manufacturer customized Reserved, undefined Bit9- Remote control 0: Non-remote control mode 1: Remote control mode Bit10- Target reached 0: Target position or speed is not reached 1: Target position or speed is reached Bit11- The internal position of the software exceeds the limit 0: Position command or feedback does not reach the software internal position limit 1: The position command reaches the internal position limit of the software. After the absolute position limit of the software takes effect (refer to the object dictionary 607Dh and 200A-02h), the servo will run with the position limit value as the target position and stop at the limit value. Bit12~13- Related to each servo mode Bit14-NA Reserved Bit15- Origin return completed 0: Origin return has not been performed or not completed 1: Origin return has been completed, and the reference point has been found								
6060	00	Servo mode selection	RW	INT8		0~10	0	RPDO
Bit0-NA Bit1- Contour position mode (pp) Bit2-NA Bit3- Contour speed mode (pv) Bit4- contour torque mode (pt) Bit5-NA Bit6- Return to zero mode (hm) Bit7- interpolation mode (ip) Bit8- Periodic synchronous position mode (csp) Bit9- Periodic synchronous speed mode (csv) Bit10- Periodic synchronous torque mode (cst)								

6061	00	Servo operation mode display	RO	INT32	-			TPDO
Actual operating mode Bit0-NA Bit1- Contour position mode (pp) Bit2-NA Bit3- Contour speed mode (pv) Bit4- contour torque mode (pt) Bit5-NA Bit6- Return to zero mode (hm) Bit7- interpolation mode (ip) Bit8- Periodic synchronous position mode (csp) Bit9- Periodic synchronous speed mode (csv) Bit10-Periodic synchronous torque mode (cst)								
6062	00	Position command	RO	INT32	Instruction unit			TPDO
Position command value within each position loop cycle time, command unit								
6063	00	Position feedback	RO	INT32	Encoder unit			TPDO
The current position of the motor fed back by the motor encoder.								
6064	00	Position feedback	RO	INT32	Instruction unit			TPDO
The position feedback value after the inverse operation of the gear ratio. 6063=6064×gear ratio								
6065	00	Excessive position deviation threshold	RW	UINT32	Instruction unit	0~232-1	3145728	RPDO
When the position deviation 60F4 is greater than ±6065, the drive reports a fault of excessive position deviation. At the same time in contour position mode, bit13 of 6041=1, this fault can be reset								
6067	00	Position reached threshold	RW	UINT16	Instruction unit	0~65535	7	RPDO
When the position deviation 60F4 is less than this value, and the time reaches 6068, the DO signal is valid after the positioning is completed, and the bit10 of 6041=1. If either of the two conditions is not met, the position arrival is invalid.								
6068	00	Position arrival window time	RW	UINT16	ms	0~65535	0	RPDO
When the position deviation 60F4 is less than this value and the time reaches 6068, the DO signal is valid when the positioning is completed, and at the same time, the bit of 6041 is 1. If either of the two conditions is not met, the position arrival is invalid								
606C	00	Actual speed	RO	INT32	Command unit/s			TPDO
This object displays position feedback per second (instruction unit)								
606D	00	Speed reaching threshold	RW	UINT32	rpm	0~65535	100	RPDO
When the difference between the motor speed feedback and the speed command is within ±606D, and the time reaches 606E, the DO signal of the speed arrival is valid, and bit10 of 6041=1, if any of the two conditions is not met, the speed arrival is invalid								
606E	00	Speed reaching window time	RW	UINT16	ms	0~65535	0	RPDO
When the difference between the current speed feedback and the speed command is ±606D, and the time reaches 606E, the DO signal of speed arrival is valid, and bit10 of 6041=1, if either of the two conditions is not met, the speed arrival is invalid								
6071	00	Target torque	RW	INT16	0.1%	-5000~5000	0	RPDO

In torque mode, target torque setting								
6072	00	Maximum torque command	RW	UINT16	0.1%	0~5000	0	RPDO
Maximum torque limit value.								
6074	00	Torque command	RO	INT16	0.1%	-5000~5000	0	TPDO
Torque output command calculated inside the drive								
6077	00	Actual torque	RO	INT16	0.1%	-5000~5000	0	TPDO
The feedback torque value obtained by the drive								
607A	00	target location	RW	INT32	Instruction unit	-231-(231-1)	0	RPDO
The target position given by the host computer is the displacement increment that the servo motor responds to when controlled according to the position factor.								
607C	00	Origin offset	RW	INT32	Instruction unit	-231-(231-1)	0	RPDO
The position of the mechanical origin offset from the mechanical zero								
607D	Software absolute position limit							
	00	Number of sub-indexes	RO	UINT8	-	2	2	-
	01	Minimum position limit	RW	INT32	User location unit	-231-(231-1)	-231	RPDO
	02	Maximum position limit	RW	INT32	User location unit	-231-(231-1)	231-1	RPDO
After the origin return is completed, by combining with 607C, set the allowable minimum and maximum position limit values. The position command exceeding this value will stop after reaching the limit.								
607E	00	Command polarity	RW	UINT8	-	0-255	0	RPDO
Bit7-Position command polarity: 0- keep the original polarity, 1- polarity inversion Bit6-Speed command polarity: 0- keep the original polarity, 1- polarity inversion Bit5-Torque command polarity: 0- keep the original polarity, 1- polarity reversal								
607F	00	Maximum speed	RW	UINT32	Command unit/s	0~232-1	104857600	RPDO
Maximum allowable speed limit value. Setting method: 607F = Maximum allowable motor speed (rmp) * encoder resolution / 60								
6081	00	Contour running speed	RW	UINT32	User speed unit	0~232-1	0	RPDO
In the contour position mode, the constant-speed running speed of the motor within the displacement of the segment is set.								
6083	00	Contour acceleration	RW	UINT32	Command unit/s2	0~232-1	1747626667	RPDO
Acceleration in pp, csv, pv mode The default value is 1747626667 The instruction unit/s2 means that it takes 10ms to accelerate from 0rmp to 1000rmp.								
6084	00	Contour deceleration	RW	UINT32	Command unit/s2	0~232-1	1747626667	RPDO
Acceleration in pp, csv, pv mode The default value is 1747626667 The instruction unit/s2 means that it takes 10ms to accelerate from 0rmp to 1000rmp.								
6085	00	Rapid stop deceleration	RW	UINT32	User acceleration	0~232-1	1747626667	RPDO

					unit			
When the host computer issues a quick stop command (bit2=0 of 6040), the acceleration of the deceleration section when 605A=2. The default value is 1747626667 The instruction unit/s2 means that it takes 10ms to accelerate from 0rmp to 1000rmp.								
6086	00	Rapid stop deceleration	RW	INT16	-	0	-	RPDO
Set the motor running curve in contour position mode. Currently only supports linear motion								
6087	00	Torque ramp	RW	UINT32	0.1%/s	0	0xFFFFFFFF	RPDO
Set the torque command increment per second in the profile torque mode								
6091	Gear ratio							
	00	Number of sub-indexes	RO	UINT8		2	2	
	01	Motor resolution	RW	UINT32	-	0~232-1	1	RPDO
	02	Load shaft resolution	RW	UINT32	-	1~232-1	1	RPDO
Establish the proportional relationship between the encoder unit and the command unit.								
6098	00	Return to origin method	RW	INT8	-	0~35	0	RPDO
Support 35 kinds of zero return methods stipulated by DS402 protocol								
6099	01	High speed search origin deceleration point	RW	UINT32	Command unit/s	0~232-1	1747626	RPDO
	02	Search origin low speed	RW	UINT32	Command unit/s	0~232-1	174762	RPDO
609A	00	Zero acceleration	RW	UINT32	Command unit/s2	1~232-1	174	RPDO
The acceleration of the variable speed stage in the zero return mode. The default value of 1747 instruction unit/s2 means: it takes 10ms to accelerate from 0rmp to 1000rmp								
60B0h	00	Position offset	RW	INT32	Instruction unit	-231-(231-1)	0	RPDO
60B1h	00	Speed offset	RW	INT32	Command unit/s	-231-(231-1)	0	RPDO
60B2h	00	Torque bias	RW	INT32	0.1%	-5000-5000	0	RPDO
60B8h	00	Probe mode	RW	UINT16	-	0~65535	0	RPDO
Bit0- Probe 1 enable 0: Disable; 1: Enable Bit1- Probe 1 trigger mode 0: Single trigger, trigger only when the trigger signal is valid for the first time; 1: Continuous trigger Bit2- Probe 1 trigger signal selection 0: Input signal; 1: Z signal Bit3-NA Bit4-Probe 1 rising edge enable 0: No latch on rising edge; 1: Latch on rising edge Bit5-Probe 1 falling edge enable 0: No latch on rising edge; 1: Latch on rising edge Bit6-NA Bit7-NA Bit8- Probe 2 enable 0: Disable; 1: Enable Bit9- Probe 2 trigger mode 0: Single trigger, trigger only when the trigger signal is valid for the first time; 1: Continuous trigger Bit10- Probe 2 trigger signal selection 0: Input signal; 1: Z signal Bit11-NA Bit12-Probe 2 rising edge enable 0: No latch on rising edge; 1: Latch on rising edge Bit13-Probe 2 falling edge enable 0: No latch on rising edge; 1: Latch on rising edge Bit14-NA								

Bit15-NA								
60B9h	00	Probe status	RW	UINT16	-	0~65535	0	RPDO
Bit0- Probe 1 enable 0: Disable; 1: Enable Bit1- Probe 1 rising edge latch execution 0: rising edge latch not executed; 1: rising edge latch executed Bit2- Probe 1 falling edge latch execution 0: falling edge latch not executed; 1: falling edge latch executed Bit3- NA Bit4- NA Bit5- NA Bit6-Probe 1 trigger signal selection 0: Input signal; 1: Z signal Bit7- Probe 1 trigger signal monitoring 0: Low level; 1: High level Bit8- Probe 2 enable 0: Disable; 1: Enable Bit9- Probe 2 rising edge latch execution 0: rising edge latch not executed; 1: rising edge latch executed Bit10- Probe 2 falling edge latch execution 0: falling edge latch not executed; 1: falling edge latch executed Bit11- NA Bit12- NA Bit13- NA Bit14-Probe 2 trigger signal selection 0: input signal; 1: Z signal Bit15-Probe 2 trigger signal monitoring 0: low level; 1: high level								
60BAh	00	Probe 1 rising edge position value	RW	INT32	Instruction unit	-231-(231-1)	0	RPDO
60BBh	00	Probe 1 falling edge position value	RW	INT32	Instruction unit	-231-(231-1)	0	RPDO
60BCh	00	Probe 2 rising edge position value	RW	INT32	Instruction unit	-231-(231-1)	0	RPDO
60BDh	00	Probe 2 falling edge position value	RW	INT32	Instruction unit	-231-(231-1)	0	RPDO
60E0h	00	Sinusoidal torque limit	RW	UINT16	0.1%	0-5000	2000	RPDO
60E1h	00	Reverse torque limit	RW	UINT16	0.1%	0-5000	2000	RPDO
60E3h	00	Supported return to zero method	RW	UINT16	-	-	-	-
60E6h	00	Position calculation method	RW	UINT16	-	0-1	0	-
60F4h	00	Position deviation	RO	INT32	Instruction unit	-231-(231-1)	0	TPDO
Position deviation, command unit								
60FC	00	Position command	RO	INT32	Encoder unit		0	TPDO
Position command, encoder unit								
60FDh	00	DI status	RO	UINT32	-	0-231-1	0	RPDO
60FEh	00	DO status	RO	UINT32	-	0-231-1	0	RPDO
60FFh	00	Target speed	RW	INT32	Command unit/s2	-231-(231-1)	0	RPDO
In the synchronous cycle speed mode, the set speed command								
6502	00	Support drive mode	RO	UINT32			00003AD hex	TPDO

3. Error code

SMC_ERROR: Record the error serial number returned by the motion control function block。

Error code	Source of production	Variable name	Description of the cause of the error
0	all	SMC_NO_ERROR	No error
1	Drive interface	SMC_DI_GENERAL_COMMUNICATION_ERROR	Communication error (such as broken Sercos ring)
2	Drive interface	SMC_DI_AXIS_ERROR	Axis error
10	Drive interface	SMC_DI_SWLIMITS_EXCEEDED	The software limit is activated, and the position exceeds the allowable range **bSWLimitEnable** is enabled, the current position of the axis is not **fSWLimitPosition** and **fSWLimitNegative** range
11	Drive interface	SMC_DI_HWLIMITS_EXCEEDED	Hardware limit switch is activated
13	Drive interface	SMC_DI_HALT_OR_QUICKSTOP_NOT_SUPPOTED	Drive status stopped or does not support quick stop
14	Drive interface	SMC_DI_VOLTAGE_DISABLED	Drive is not enabled
15	Drive interface	SMC_DI_IRREGULAR_ACTPOSITION	The format of the position currently given by the drive is incorrect. Check communication.
16	Drive interface	SMC_DI_POSITIONLAGERROR	Position lag error. Exceeding the limit value in the setting and current position.
20	All modules created by run control	SMC_REGULATOR_OR_START_NOT_SET	The controller is not enabled or the brake is not opened
21	The axis is in the wrong control mode	SMC_WRONG_CONTROLLER_MODE	Axis is not a correct way to control
30	Drive interface	SMC_FB_WASNT_CALLED_DURING_MOTION	The module created by motion control is not called before the end of the motion
31	All modules	SMC_AXIS_IS_ON_AXIS_REF	The AXIS_REF variable given is not of type AXIS_REF
32	The axis is in error control mode	SMC_AXIS_REF_CHANGED_DURING_OPERATION	The return value of the SMC_REF-variable is processed before the module is activated
33	Drive	SMC_FB_ACTIVE_AIXS_DIABLED	The axis is not activated when

	interface		it is moving (MC_Power.bRegulatorOn)
34	All modules created by motion control	SMC_AXIS_NOT_READY_FOR_MOTION	The axis cannot process the current command in the current state
40	Virtual drive	SMC_VD_MAX_VELOCITY_EXCEEDED	Reach the maximum velocity (fMaxVelocity)
41	Virtual drive	SMC_VD_MAX_ACCELERATION_EXCEEDED	Reach the maximum acceleration (fMaxAcceleration)
42	Virtual drive	SMC_VD_MAX_DECELERATION_EXCEEDED	Reach the maximum deceleration (fMaxDeceleration)
50	SMC_Homing	SMC_3SH_INVALID_VELACC_VALUES	Invalid speed or acceleration value
51	SMC_Homing	SMC_3SH_MODE_NEEDS_HWLIMIT	The module needs to use the end limit switch (safety use)
70	SMC_SetControllerMode	SMC_SMC_NOT_SUPPORTED	Mode does not support
71	SMC_SetControllerMode	SMC_SMC_AXIS_IN_WRONG_STATE	The control mode used in the current mode does not support
75	SMC_SetTorque	SMC_ST_WRONG_CONTROLLER_MODE	The axis is not a correct control mode, and this function block needs to be enabled in the torque mode
80	SMC_ResetAxisGroup	SMC_RAG_ERROR_DURING_STARTUP	An error occurred when the axis group was started
90	SMC_ChangeGearingRatio	SMC_CGR_ZERO_VALUES	Incorrect variable
91	SMC_ChangeGearingRatio	SMC_CGR_DRIVE_POWERED	The transmission ratio cannot be changed in the drive control mode
92	SMC_ChangeGearingRatio	SMC_CGR_INCALID_POSPERIOD	Inappropriate position period (<=0)
110	MC_Power	SMC_P_FTASKCYCLE_EMPTY	The axis does not contain any messages during the scan cycle (fTaskCycle = 0)
120	MC_Reset	SMC_R_NO_ERROR_TO_RESET	Axis reset without error
121	MC_Reset	SMC_R_DRIVE_DOESNT_ANSWER	The axis did not perform an error reset
122	MC_Reset	SMC_R_ERROR_NOT_RESETIABLE	The error cannot be reset
123	MC_Reset	SMC_R_DRIVE_DOESNT_ANSWER_IN_TIME	No response to communication with the axis
130	MC_ReadParameter	SMC_RP_PARAM_UNKNOWN	Parameter number position
131	MC_ReadParameter, MC_ReadBoolParameter	SMC_RP_REQUSTING_ERROR	An error occurred during transmission of parameters to the drive. Refer to the error of the function block instance ReadDriveParameter (SM_DriveBasec.lid)
140	MC_WriteParame	SMC_WP_PARAM_INVALID	Parameter number position or

	ter, MC_WriteBoolParameter		write operation is not allowed
141	MC_WriteParameter, MC_WriteBoolParameter	SMC_WP_SENDING_ERROR	Refer to the error of the module instance WriteDriveParameter (Drive_Basic.lib)
170	MC_Home	SMC_H_AXIS_WASNT_STANDSTILL	The axis is not in a standard (stationary) state
171	MC_Home	SMC_H_AXIS_DIDNT_START_HOMING	An error occurred while executing the return to zero
172	MC_Home	SMC_H_AXIS_DIDNT_ANSWER	Communication error
173	MC_Home	SMC_H_ERROR_WHEN_STOPPING	Perform zero return error to stop, check whether the deceleration is set.
180	MC_Stop	SMC_MS_UNKNOWN_STOPPING_ERROR	An unknown error occurred while stopping
181	MC_Stop	SMC_MS_INVALID_ACCDEC_VALUES	Inappropriate speed or acceleration value
182	MC_Stop	SMC_MS_DIRECTION_NOT_APPLICABLE	Unable to use the shortest setting of sports video to be used Direction = shortest is not available
183	MC_Stop	SMC_MS_AXIS_IN_ERRORSTOP	The axis is in an error stop state. The stop cannot be processed.
184	MC_Stop	SMC_BLOCKING_MC_STOP_WASNT_CALLED	An instance of MC_Stop, locked axis (Execute = TRUE), cannot be adjusted. Please call MC_Stop(Execute=FALSE)
201	MC_MoveAbsolute	SMC_MA_INVALID_VELACC_VALUES	Inappropriate speed or acceleration value
202	MC_MoveAbsolute	SMC_MA_INVALID_DIRECTION	Wrong direction
226	MC_MoveRelative	SMC_MA_INVALID_VELACC_VALUES	Inappropriate speed or acceleration value
227	MC_MoveRelative	SMC_MA_INVALID_DIRECTION	Wrong direction
251	MC_MoveAdditive	SMC_MAD_INVALID_VELACC_VALUES	Inappropriate speed or acceleration value
252	MC_MoveAdditive	SMC_MAD_INVALID_DIRECTION	Wrong direction
276	MC_MoveSuperImposed	SMC_MSI_INVALID_VELACC_VALUES	Inappropriate speed or acceleration value
277	MC_MoveSuperImposedF	SMC_MSI_INVALID_DIRECTION	Wrong direction
301	MC_MoveVelocity	SMC_MV_INVALID_ACCEC_VALUES	Inappropriate speed or acceleration value
302	MC_MoveVelocit	SMC_MV_DIRECTION_NOT_APPLIC	Direction=shortest/fastest

	y	ABLE	Not support
325	MC_PositionProfile	SMC_PP_ARRAYSIZE	Wrong size
326	MC_PositionProfile	SMC_PP_STEPOMS	Step time = t#0s
350	MC_VelocityProfile	SMC_VP_ARRAYSIZE	Wrong size
351	MC_VelocityProfile	SMC_VP_STEPOMS	Step time = t#0s
375	MC_AccelerationProfile	SMC_AP_ARRAYSIZE	Wrong size
376	MC_AccelerationProfile	SMC_AP_STEPOMS	Step time = t#0s
400	MC_TouchProbe	SMC_TP_TRIGGEROCCUPIED	Trigger conditions and be activated
401	MC_TouchProbe	SMC_TP_COULDNT_SET_WINDOW	Drive interface does not support window function
402	MC_TouchProbe	SMC_TP_COMM_ERROR	Communication error
410	MC_AbortTrigger	SMC_AT_TRIGGERNOTOCCUPIED	The trigger condition has been terminated
426	MC_MoveContinuousRelative	SMC_MCR_INVALID_VELACC_VALUES	Inappropriate speed or acceleration value
427	MC_MoveContinuousRelative	SMC_MCR_INVALID_DIRECTION	Wrong direction
451	MC_MoveContinuousAbsolute	SMC_MCA_INVALID_VELACC_VALUES	Inappropriate speed or acceleration value
452	MC_MoveContinuousAbsolute	SMC_MCA_INVALID_DIRECTION	Wrong direction
453	MC_MoveContinuousAbsolute	SMC_MCA_DIRECTION_NOT_APPLICABLE	Direction = fastest unavailable
500	SMC_ControlAixsByPos	SMC_CAP_GAP_VELACCDEC_INVALID	Invalid speed or acceleration
510	SMC_ControlAixsByPosvel	SMC_CAPV_GAP_VELACCDEC_INVALID	Invalid speed or acceleration
520	SMC_ControlAixsByvel	SMC_CAV_GAP_VELACCDEC_INVALID	Invalid speed or acceleration
600	SMC_CamRegister	SMC_CR_NO_TAPPETS_IN_CAM	No tappet is included in CAM
601	SMC_CamRegister	SMC_CR_TOO_MANY_TAPPETS	Tappet group ID reaches MAX_NUM_TAPPETS
602	SMC_CamRegister	SMC_CR_MODE_THAN_32_ACCESSES	More than 32 interfaces in one CAM_REF
625	MC_CamIN	SMC_CI_NO_CAM_SELECTED	No CAM is selected
626	MC_CamIN	SMC_CI_MASTER_OUT_OF_SCALE	Spindle out of range
627	MC_CamIN	SMC_CI_RAMPIN_NEEDS_VELACC_VALUES	The speed and acceleration of the ramp_in function block must be precisely specified
628	MC_CamIN	SMC_CI_SCALING_INCORRECT	The scale variable fEditor/TableMasterMin/Max is incorrect

640	SMC_CAMBounds, SMC_CAMBounds_Pos,	SMC_CB_NOT_IMPLEMENTED	The function block of the given CAM format does not support
675	MC_GearIn	SMC_GI_RATIO_DENOM	RatioDenominator = 0
676	MC_GearIn	SMC_GI_INVALID_ACC	Inappropriate acceleration
677	MC_GearIn	SMC_GI_INVALID_DEC	Inappropriate acceleration
725	MC_Phase	SMC_PH_INVALID_VALACCDEC	Speed, acceleration and deceleration are inappropriate
726	MC_Phase	SMC_PH_ROTARYLID_PERIOD0	Rotation axis fPositionPeriod = 0
750	All modules using MC_CAM_REF as input	SMC_NO_CAM_REF_TYPE	The given CAM is not of type MC_CAM_REF
751	MC_CamTableSelect	SMC_CAM_TABLE_DOES_NOT_COVER_MASTER_SCALE	If the data obtained from CamTable is not the main axis area obtained by data conversion (xStart and xEnd)
775	MC_GearInPos	SMC_GIP_MASTER_DIRECTION_CHANGN	The master axis changes the direction of rotation during the slave axis coupling process
800	SMC_BacklashCompensation	SMC_BC_BL_TOO_BIG	Gear return ratio (fBacklash) is too large (>positionperiode/2)
1000	CNC need to authorize the function block	SMC_NO_LICENSE	The target is not authorized by CNC.
1001	SMC_Interpolator	SMC_INT_BEL_ZERO	The path cannot be processed because speed=0
1002	SMC_Interpolator	SMC_INT_NO_STOP_AT_END	Last path object Vel_End> 0
1003	SMC_Interpolator	SMC_INT_DATA_UNDERRUN	Warning: GEOINFO-The list is processed in DataIn, but the list is not set at the end. Reason: Forgetting to set EandOfList or SMC_Interpolator in DataIn is faster than the path compilation module
1004	SMC_Interpolator	SMC_INT_VEL_NONZERO_AT_STOP	Stop speed>0
1005	SMC_Interpolator	SMC_INT_TOO_MANY_RECURSIONS	Using too many SMC_Interpolator to call SoftMotion-error.
1006	SMC_Interpolator	SMC_INT_NO_CHECKVELOCITIES	Input-OutQueue DataIn is not used as the final processing module of SMC_CheckVelocities

1007	SMC_Interpolator	SMC_INT_PATH_EXCEEDED	Internal/numerical error
1008	SMC_Interpolator	SMC_INT_VEL_ACC_DEC_ZERO	Speed, acceleration or deceleration is empty or too low
1009	SMC_Interpolator	SMC_INT_DWIPOTIME_ZERO	FB call dwIpoTime=0
1050	SMC_Interpolator2Dir	SMC_INT2DIR_BUFFER_TOO_SMALL	Data buffer is too small
1051	SMC_Interpolator2Dir	SMC_INT2DIR_PATH_FITS_NOT_IN_QUEUE	The path is not fully contained in the queue
1080	SMC_Interpolator	SMC_WAR_INT_OUTQUEUE_TOO_SMALL	Warning: The size of OutQueue DataIn is too small. It is not guaranteed to meet the requirements for stopping.
1081	SMC_Interpolator	SMC_WAR_END_VELOCITIES_INCORRECT	Warning: End point speed is inconsistent
1100	SMC_Interpolator2Dir	SMC_CV_ACC_DEC_VEL_NONPOSITIVE	Speed, deceleration or acceleration value is not in the positive direction
1120	SMC_Controlaxisbypos	SMC_CA_INVALID_ACCDEC_VALUES	The variable of fGapVelocity / fGapAcceleration / fGapDeceleration is not a positive value
1200	SMC_NCDecoder	SMC_DEC_ACC_TOO_LITTLE	Acceleration value not allowed
1201	SMC_NCDecoder	SMC_DEC_RET_TOO_LITTLE	Deceleration value not allowed
1202	SMC_NCDecoder	SMC_DEC_OUTQUEUE_RAN_EMPTY	Data below Queue is read and is empty
1203	SMC_NCDecoder	SMC_DEC_JUMP_TO_UNKNOWN_LINE	Because the line number is unknown, the jumped line number cannot be executed
1204	SMC_NCDecoder	SMC_DEC_INVALID_SYNTAX	Grammatical errors
1205	SMC_NCDecoder	SMC_DEC_3DMODE_OBJECT_NOT_SUPPORTED	These objects do not support 3D mode
1300	SMC_GCodeViewer	SMC_GCV_BUFFER_TOO_SMALL	The buffer is too small
1301	SMC_GCodeViewer	SMC_GCV_BUFFER_WRONG_TYPE	Buffer element type error
1302	SMC_GCodeViewer	SMC_GCV_UNKNOWN_IPO_LINE	The current imputation cannot be found
1500	All function blocks using SMC_CNC_REF	SMC_NO_CNC_REF_TYPE	The given CNC program is not of type SMC_CNCN_REF
1501	Use all function blocks of SMC_OUTQUEUE	SMC_NO_OUTQUEUE_TYPE	The given OutQueue is not of type SMC_OUTQUEUE
1600	CNC function block	SMC_3D_MODE_NOT_SUPPORTED	This function block is only available in the 2D path
2000	SMC_ReadNCFile	SMC_RNCF_FILE_DOESNT_EXIST	file does not exist

2001	SMC_ReadNCFile	SMC_RNCF_NO_BUFFER	No buffer allocation
2002	SMC_ReadNCFile	SMC_RNCF_BUFFER_TOO_SMALL	The buffer is too small
2003	SMC_ReadNCFile	SMC_RNCF_DATA_UNDERRUN	The low-buffer data in the buffer is read and is empty
2004	SMC_ReadNCFile	SMC_RNCF_VAR_COULDNT_BE_REPLACED	Placeholder variables cannot be replaced
2005	SMC_ReadNCFile	SMC_RNCF_NOT_VARLIST	The entered pvl cannot point to the SMC_VARLIST object
2050	SMC_ReadNCQueue	SMC_RNCQ_FILE_DOESNT_EXIST	File cannot be opened
2051	SMC_ReadNCQueue	SMC_RNCQ_NO_BUFFER	No buffer definition
2052	SMC_ReadNCQueue	SMC_RNCQ_BUFFER_TOO_SMALL	The buffer is too small
2053	SMC_ReadNCQueue	SMC_RNCQ_UNEXPECTED_EOF	Unknown end of file
2100	SMC_AxisDiagnostiLog	SMC_ADL_FILE_CANNOT_BE_OPENED	File cannot be opened
2101	SMC_AxisDiagnostiLog	SMC_ADL_BUFFER_OVERRUN	Over-range buffering: **WriteToFile** must be called more brilliantly
2200	SMC_ReadCAM	SMC_RCAM_FILE_DOESNT_EXIST	File cannot be opened
2201	SMC_ReadCAM	SMC_RCAM_TOO_MUCH_DATA	Too much data saved to CAM
2202	SMC_ReadCAM	SMC_RCAM_WRONG_COMPILE_TYPE	Error compilation mode
2203	SMC_ReadCAM	SMC_RCAM_WRONG_VERSION	File version error
2204	SMC_ReadCAM	SMC_RCAM_UNEXPECTED_EOF	Unknown end of file
3001	SMC_WriteDriveParamsToFile	SMC_WDOF_CHANNEL_OCCUPIED	SMC_WDPF_TIMEOUT_PREPARING_LIST
3002	SMC_WriteDriveParamsToFile	SMC_WDOF_CANNOT_CREATE_FILE	File cannot be created
3003	SMC_WriteDriveParamsToFile	SMC_WDOF_ERROR_WHEN_READING_PARAMS	Error when reading file parameters
3004	SMC_WriteDriveParamsToFile	SMC_WDOF_TIMEOUT_PREPARING_LIST	Error when preparing parameters
5000	SMC_Encoder	SMC_ENC_DENOM_ZERO	Time error in the decoder parameter list
5001	SMC_Encoder	SMC_ENC_AXISUSEDBYOTHERFB	Other modules are processing the decoding axis
5002	Drive interface	SMC_ENC_FILTER_DEPTH_INVALID	Inappropriate filter rotation