Bus-Type Motion Controller

DVP15MC11T

DVP50MC11T

CANopen

EtherCAT

--- NC

DELTA

Contents

Introduction to NC

- NC
- G Codes
- How to Use NC Function
- How to Use Axes Group Function
- Robot Function

Introduction to NC and G codes

Introduction to NC

NC technology, i.e. **numerical control (NC)**, is the technology of realizing the automation of controlling a working process through numeric control. Usually what is controlled by NC includes mechanical parameters such as **position, angle** and **speed** as well as switch parameters which are related to the mechanical energy flow. The realization of NC relies on the data carrier and binary-data operation.

CNC High-speed Tapping machine

CNC Engraving Machine,
CNC machine center

CNC Engraving Machine

CNC Woodworking Machinery

G/M Codes

G/M code	Function	Num. of axes	Format
G0	Rapid Positioning	8	Format 1: G0 X_Y_Z_A_B_C_P_Q_
G1	Linear Interpolation	8	Format 1: G1 X_Y_Z_A_B_C_P_Q_ E_ F_
G2	Clockwise Circular/ Helical Interpolation	8	Format 1: G2 X_Y_Z_A_B_C_P_Q_I_J_(I_K_/J_K_)T_E_F_ Format 2: G2 X_Y_Z_A_B_C_P_Q_R_T_E_F_
G3	Anticlockwise Circular /Helical Interpolation	8	Format 1: G3 X_Y_Z_A_B_C_P_Q_I_J_(I_K_/J_K_)T_E_F_ Format 2: G3 X_Y_Z_A_B_C_P_Q_R_T_E_F_
G4	Dwell Instruction	--	Format 1: G4 K__
G17	Specify XY Plane	--	Format 1: G17
G18	Specify ZX Plane	--	Format 1: G18
G19	Specify YZ Plane	--	Format 1: G19
G50	Precise Stop	--	Format 1: G50
G51	Round path transition	--	Format 1: G51 D_
G52	Smooth path transition	--	Format 1: G52
G90	Absolute mode	--	Format 1: G90
G91	Relative mode	--	Format 1: G91
M0~M99	M code	--	Format 1: M_ D_

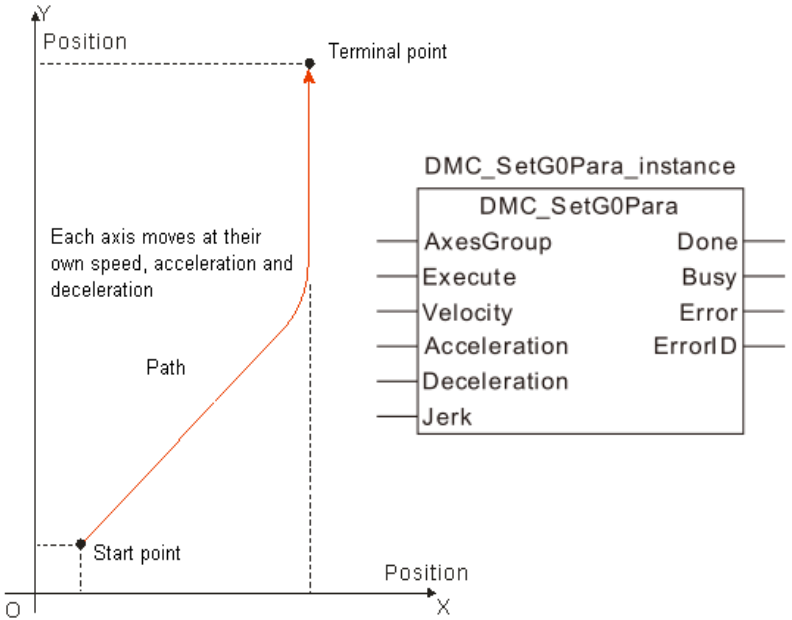
G0: Rapid Positioning

Feature

- The axes perform a rapid positioning at their preset **accelerations/decelerations** and **max speeds** to move to the specified target position.
- Only makes the **rapid positioning** from one point to another.
- Used for the positioning without any requirement on the motion path.

Use of G0

- **Format:**
  - Terminal point position：X、Y、Z、A、B、C、P、Q
- **Example:** N0 G0 X10 Y20 Z50 A20 B20 C20 P20 Q20
- **Explanation:** When G0 execution starts, all axes **speed up at their preset accelerations** to move toward the target position. When their **max. velocities** are reached, they will move constantly at the max. velocity . While the target position is being approached, the axes will **speed down at their preset decelerations**. When the velocity is 0, the axes will just reach the target position.



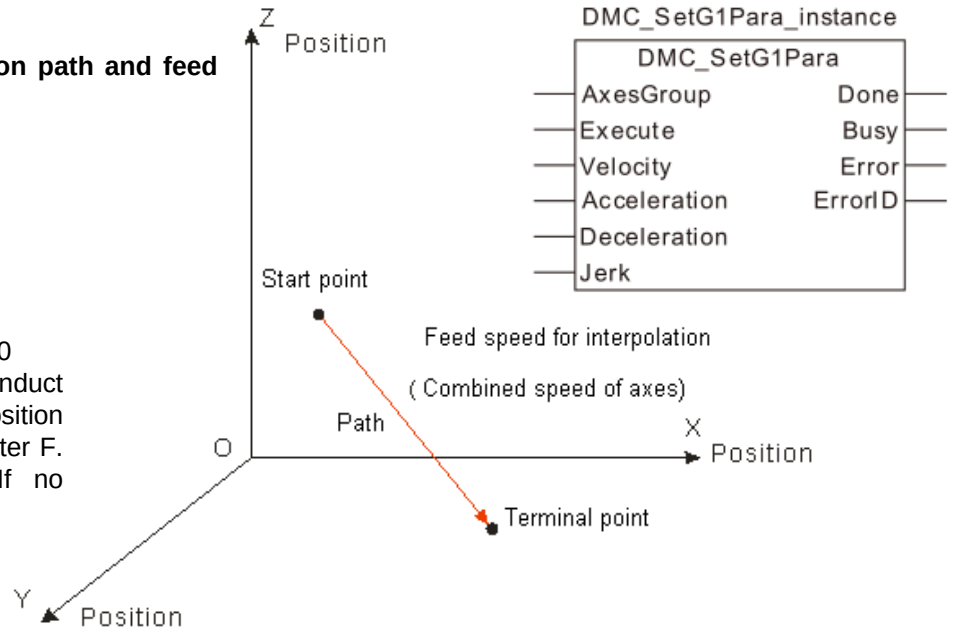
G1: Linear Interpolation

Feature

- The terminal actuator makes a linear interpolation with the **preset velocity** and **acceleration/deceleration** to move toward the specified target position.
- To achieve the precise positioning from one point to another.
- **Used for the interpolation with strict requirements for the motion path and feed velocity.**

Use of G1

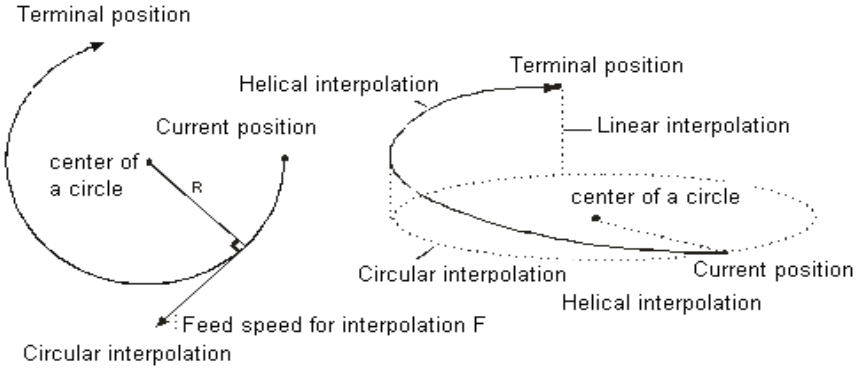
- **Format**:
  - Terminal point position: X、Y、Z、A、B、C、P、Q
  - Acceleration/deceleration: E
  - Velocity: F
- **Example**: N0 G1 X50 Y50 Z50 A20 B20 C20 P20 Q20 E20 E-20 F100
- **Explanation**: When G1 execution starts, the terminal actuator will conduct the linear interpolation toward the end point position with current position as the start point. The interpolation velocity is determined by parameter F. The acceleration/deceleration is determined by parameter E. If no parameters are specified for them, the default values are used.



G2/G3 :Circular/ Helical Interpolation

Feature

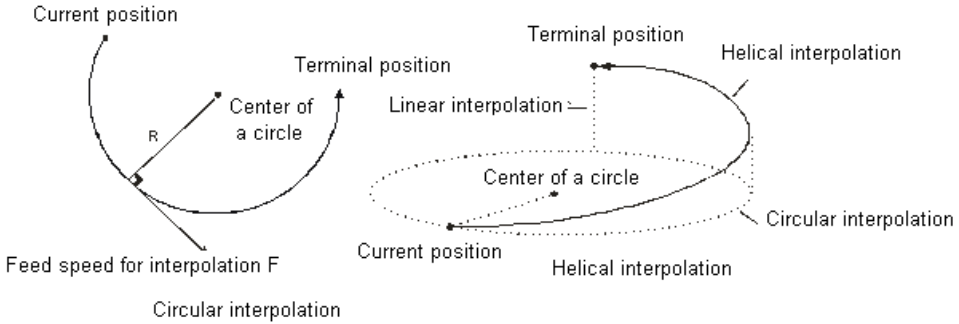
- The terminal mechanism makes a helical interpolation with **the preset velocity and acceleration/deceleration** to move to the specified target position.
- G2 for **clockwise** circular /helical interpolation; G3 for the **anticlockwise** circular/helical interpolation.
- Use the **center** method and **radius** method for the circular interpolation.



G02

Use of G2/G3

- **Format** :
  - Terminal point position: X、Y、Z、A、B、C、P、Q
  - Acceleration: E
  - Velocity: F
  - Number of circles: T
  - Radius :R
  - Coordinates of circle centers: I J (XY plane), I K (ZX plane), J K (YZ plane) are all relative coordinates **based on the start point of a circular arc.**
- **Example (specify a circle center)** :
- N0 G2 X50 Y50 Z50 A20 B20 C20 P20 Q20 I25 J25 T2 E10 E-10 F10
- **Explanation (specify a circle radius)** :
- N0 G2 X50 Y50 Z50 A20 B20 C20 P20 Q20 R25 T2 E10 E-10 F10



G03

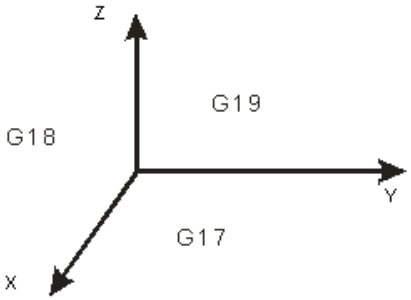
G17/G18/G19 : Specify Circular Interpolation Plane

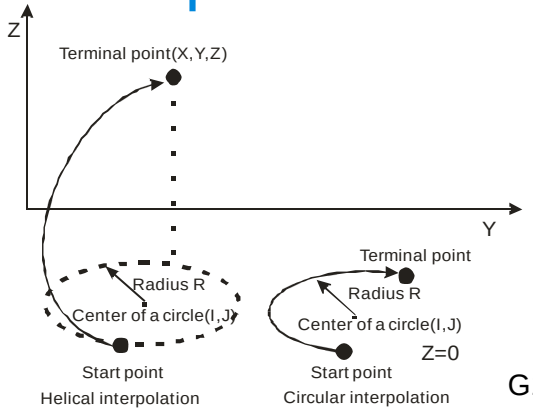
Feature

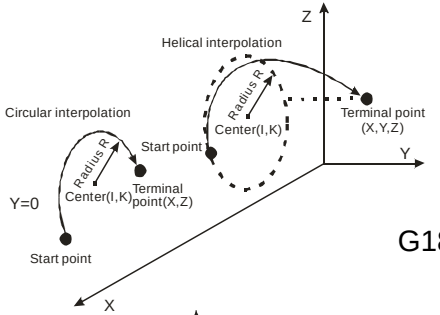
- G17/G18/G19 are used for setting the planes for the **circular/helical** interpolation and have nothing to do with the linear interpolation.
- During the program execution, the three planes for interpolation can be switched to each other. If there is no plane for option in the program, the initial status is the XY plane (G17).

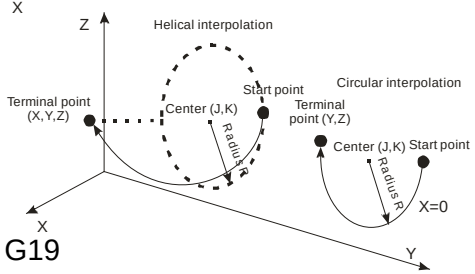
Use of G17, G18, G19

- **Explanation**: After these instructions are used, G2/G3 instructions following them will move according to the planned planes.
- **Example**:
  - N0 G17
  - N0 G18
  - N0 G19









G Codes

G4 (Dwell Instruction)

- **Example:** N0 G4 K10
- **Explanation:** The motion is **paused for a while** and then the motion is going on after the pause is over. The pause time is specified by the value after K. The unit: seconds; Range:0.001s ~ 100,000s.

G90 (Absolute Mode)

- **Example:** N0 G90
- **Explanation:** When G90 is written in the program, all subsequently written coordinate values are **based on the zero point of G code**.

G91 (Relative Mode)

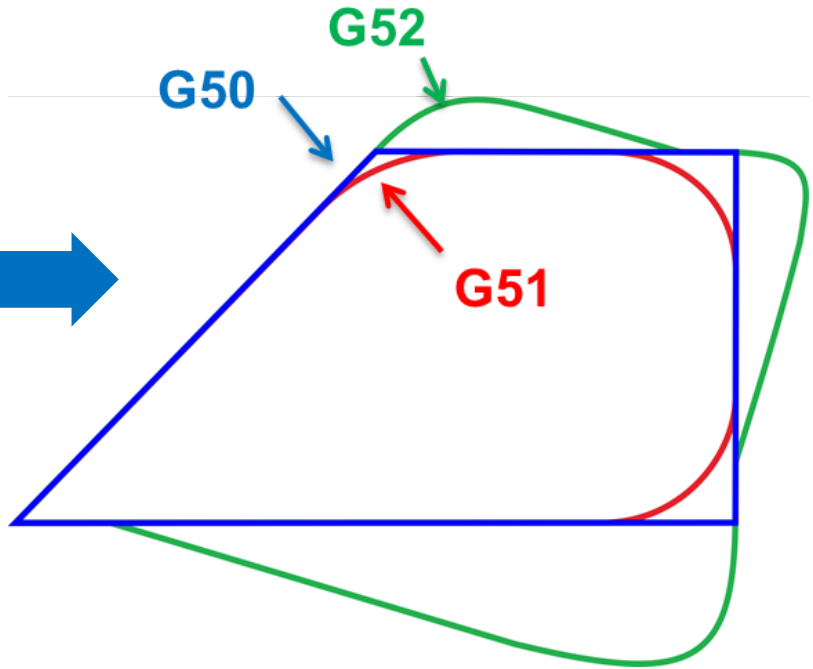
- **Example**: N0 G91
- **Explanation**: When G91 is written in the program, all subsequently written coordinate values will be calculated by **taking the previous coordinate position as the start point**.

Transition Mode

While moving objects, **rapidly passing a position without stopping** is often required.

In the processing of parts, non-circular arc curve segments are often encountered. Using G1 for the small line segment interpolation is **low-speed and low-efficiency.**

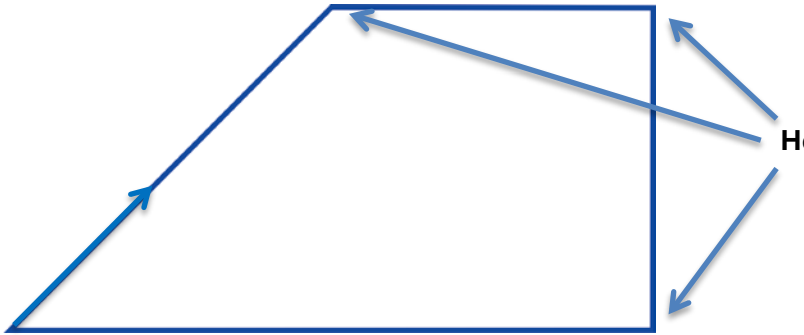
To solve such problems , new G codes: G50, G51,G52 are added



G50: Precise Stop

- G50 is used for the precise stop **without transition** and **with velocity decelerated to 0**. It is a default mode.
  - **Example:** N0 G50
  - **Explanation**: When G50 is written in the program, the terminal mechanism will always stop precisely with the velocity decelerated to 0 as the execution of every instruction after G50 is over.
  - **See the G codes for the right-side path:**

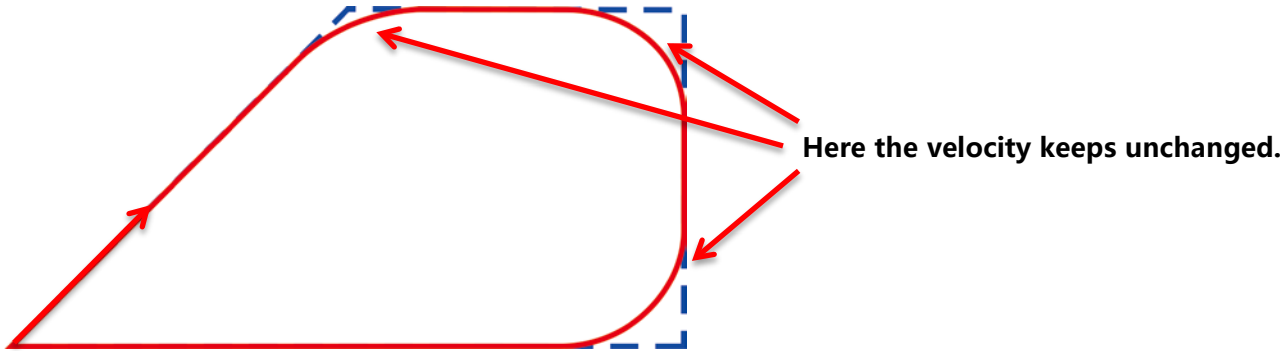
```
N0 G50
N1 G1 X10 Y10
N2 G1 X20 Y10
N3 G1 X20 Y0
N4 G1 X0 Y0
```



G51 : Round Path Transition

- G51 is used for the round path transition with the velocity unchanged.
  - **Example:** N0 G51 D20
  - **Explanation:** When G51 is written in the program, the transition mode will be the round path transition for all subsequent linear/ circular interpolation instructions. In G51 mode, only the first one among F velocity parameters of interpolation instructions is valid. To change the velocity during the operation, change the **VelOverride** value of the called instruction.
  - **See the G codes for the right-side path:**

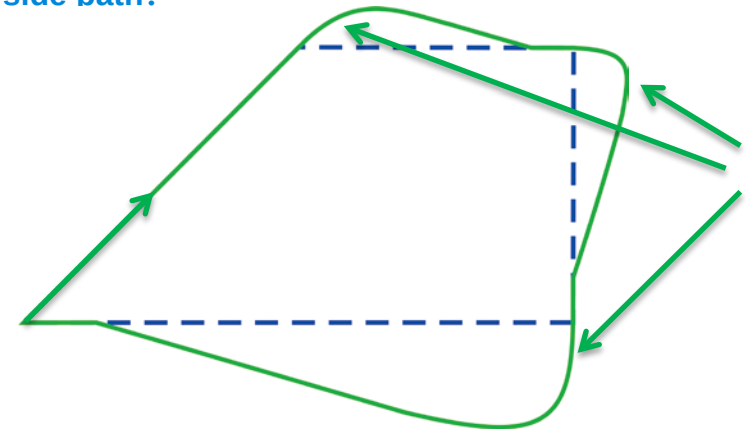
```
N0 G51 D2
N1 G1 X10 Y10
N2 G1 X20 Y10
N3 G1 X20 Y0
N4 G1 X0 Y0
```



G52 : Smooth Path Transition

- G52 is used for the smooth path transition and is suitable for continual interpolation of small segments.
  - **Example**： G52
  - **Explanation**： When G52 is written in the program, the transition mode will be the smooth path transition for all subsequent linear/circular interpolation instructions. In G52 mode, only the first one of F velocity parameters of interpolation instructions is valid. To change the velocity during the operation, change the **VelOverride** value of the called instruction.
  - **See the G codes for the right-side path：**

```
N0 G52
N1 G1 X10 Y10
N2 G1 X20 Y10
N3 G1 X20 Y0
N4 G1 X0 Y0
```



Here are the curves formed through the linear fitting. If a line segment is small enough, the linear - fitting curve will get closer to the actual curve.

Transition Mode

While moving objects**, rapidly passing a position without stopping** is often required. —— **G51**

In the processing of parts, non-circular arc curve segments are often encountered. So using G1 for the small line segment interpolation is **slow-speed and low-efficiency**. —— **G52**

	G50 (Precise stop)	G51 (Round path transition)	G52 (Smooth path transition)
G0	Available	The transition mode does not work. The operation effect is the same as G50.	The transition mode does not work. The operation effect is the same as G50
G1	Available	Available	Available
G2	Available	Available	Used when the line/ circular arc and circular arc are (almost) tangent.
G3	Available	Available	Used when the line/ circular arc and circular arc are (almost) tangent.

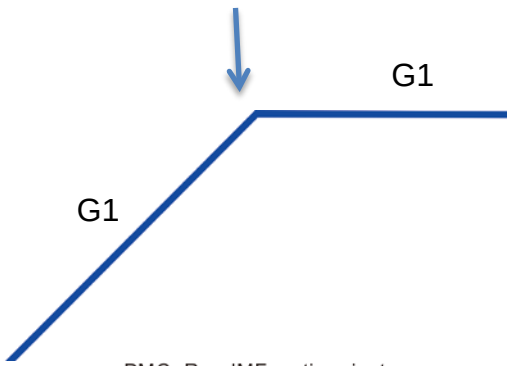
M Codes

Features

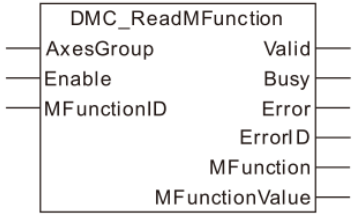
- Can be used for the interaction with other control program.
- M0~M99 include 100 M codes and meanwhile transmit one value for exchange.

Use of M Codes

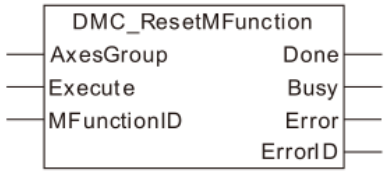
- **Example**：
  - **A row without any code but M code**：M0 D100.01
  - **A row with M code and other instruction**：G1 X100 M0 D100.01
- **Explanation:**
  - **A row without any code but M code**：When the program execution reaches this row, CNC execution will stop and M code will change to TRUE and meanwhile send a value. Resetting M code, the CNC execution will continue.
  - **A row with M code and other instruction**：When the program execution reaches this row, G code execution will **not stop**. At the beginning of the execution, M code changes to TRUE and sends a value. No need to reset M code.



DMC_ReadMFunction_instance



DMC_ResetMFunction_instance



Using Variables by G Code

How to use variables by G code

- G codes can represent key values via variables in the program.
  - All of X、Y、Z、A、B、C、P、Q、I、J、K、R、T、F、E can use device variables.
  - The data type of device variables of T is ULINT. And the data type of device variables of others is LREAL.
  - N0 G1 X$ML100$ Y$ML101$. It means that %ML100 and %ML101 are used and the data type is LREAL.

```
1   N0 G1 X$ML100$ Y$ML101$
```

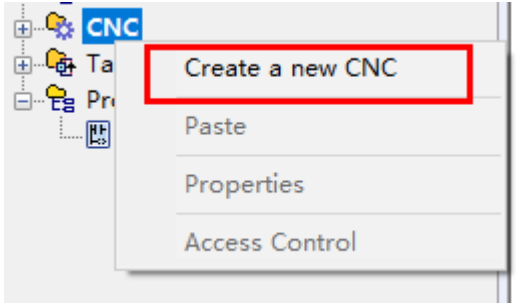
Index	Scope	Name	Address	Data Type
1	VAR_GLOBAL	X axis position	%ML100	LREAL
2	VAR_GLOBAL	Y axis position	%ML101	LREAL

How to Use NC Function

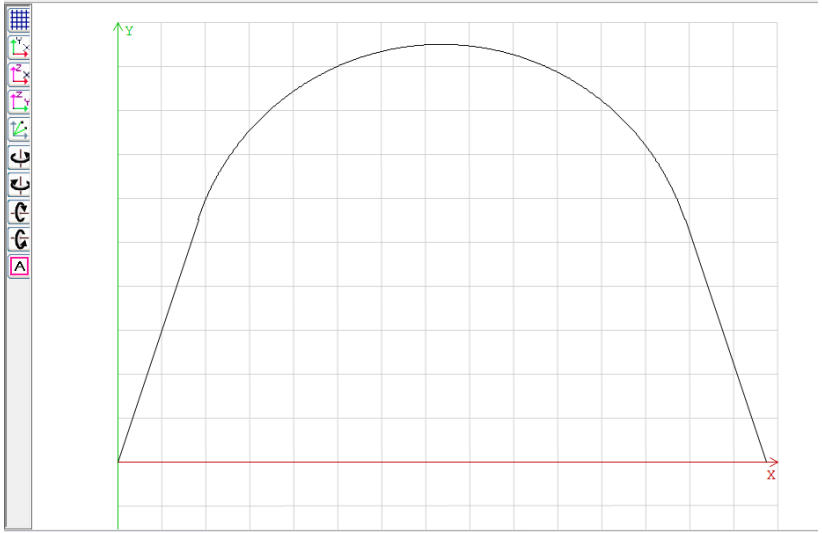
How to Use NC Function

1. Generate G codes

- Generate G codes for product processing by using CAD or other software or edit G codes by manual based on actual needs.
- Copy G codes generated or import them to the software.
- **Write only one G code for each row.**

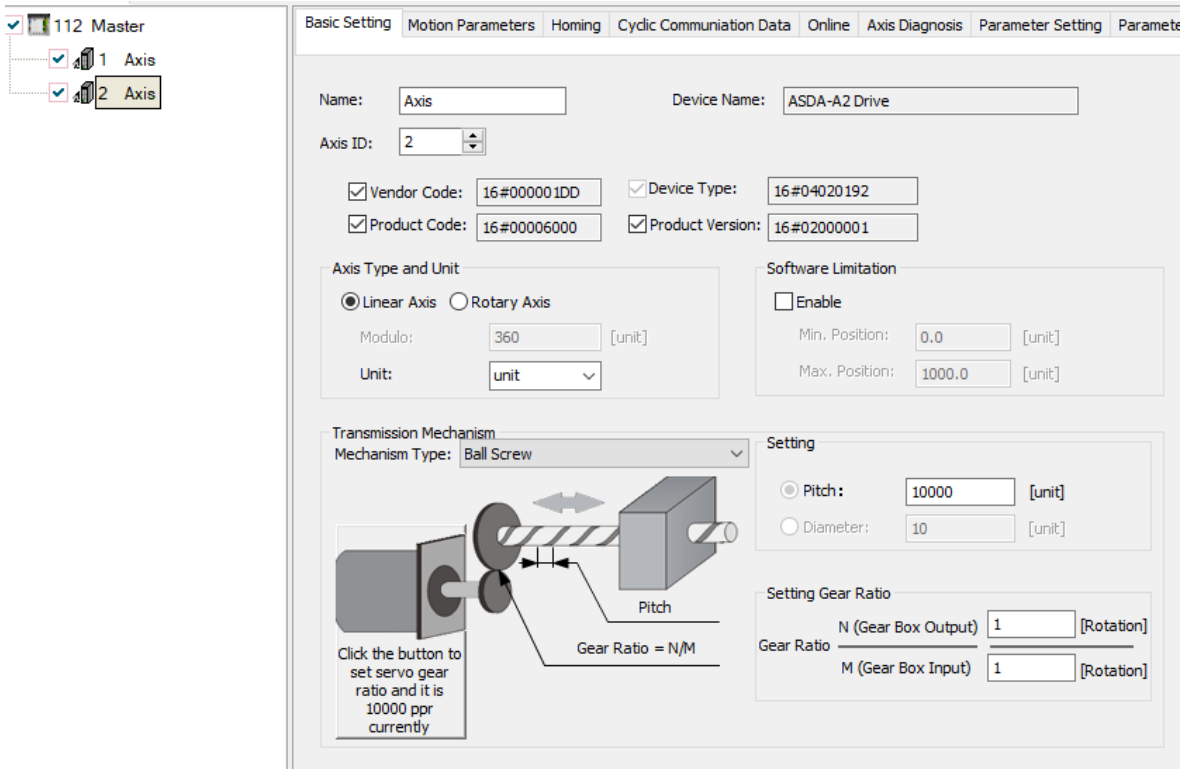



```
1  N0 G1 X10000 Y30000 E10000 E-10000 F10000
2  N1 G2 X70000 Y30000 I30000 J-10000
3  N2 G1 X80000 Y0
4
```



How to Use NC Function

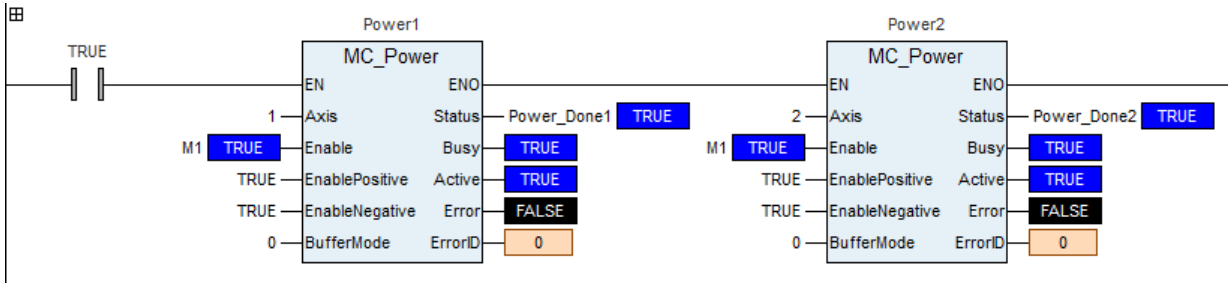
2. Configure axis parameters



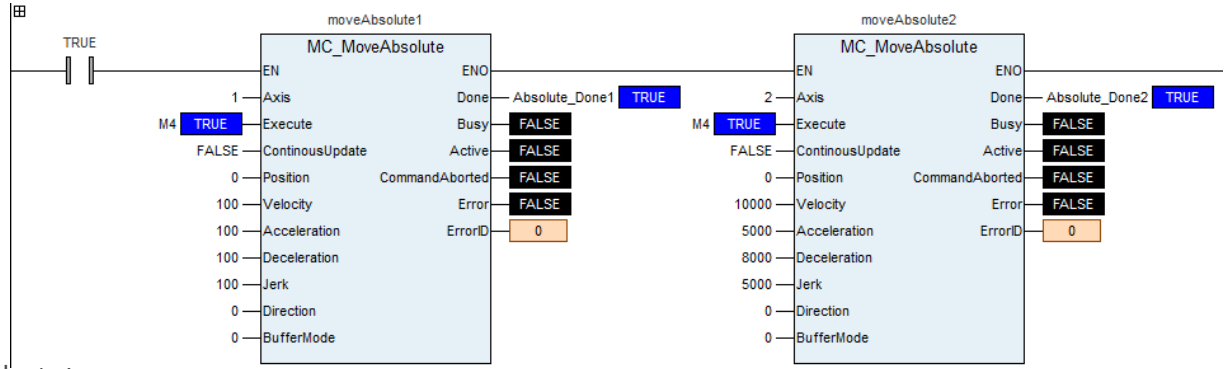
How to Use NC Function

3. Edit the control program

(1) The axes are Servo ON.

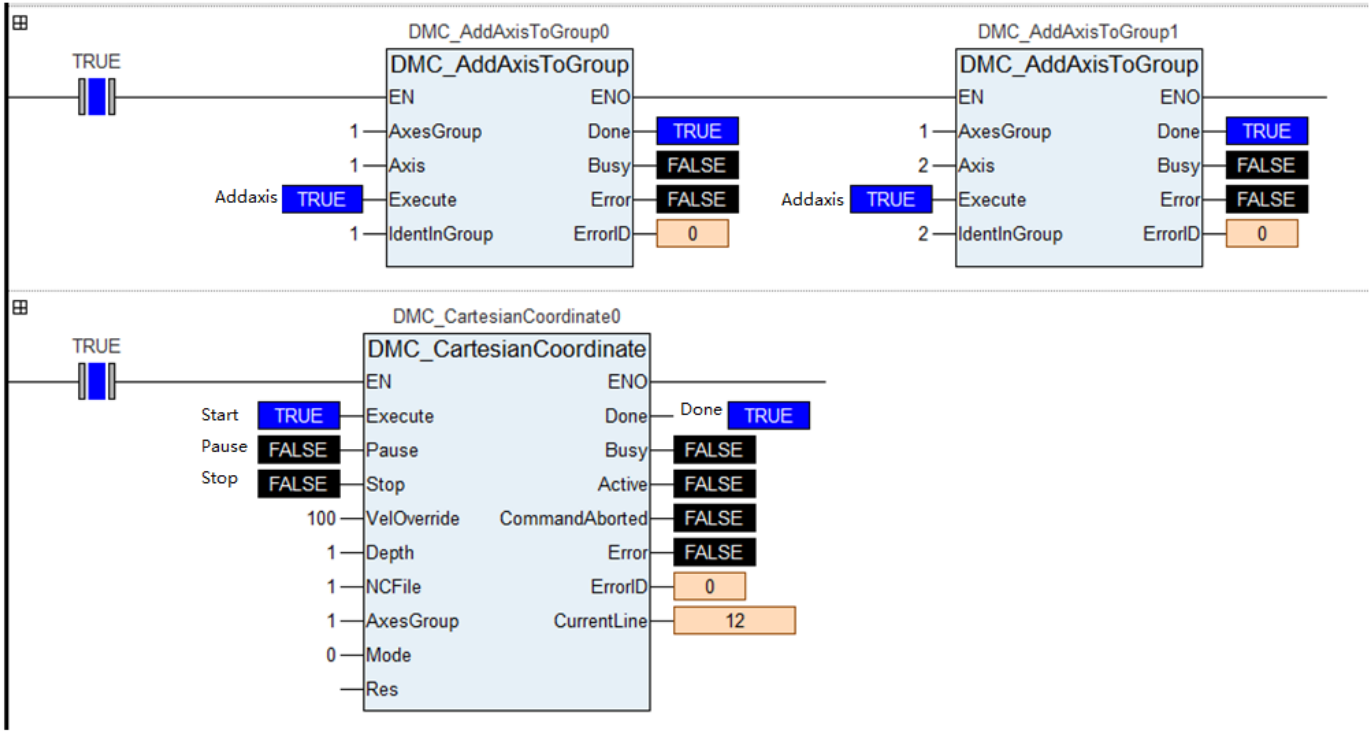


(2) The axes return to the origin.



How to Use NC Function

(3) After adding axes to an axes group via DMC_AddAxisToGroup, execute DMC_CartesianCoordinate instruction to start the motion.



How to Use NC Function

DMC_CartesianCoordinate

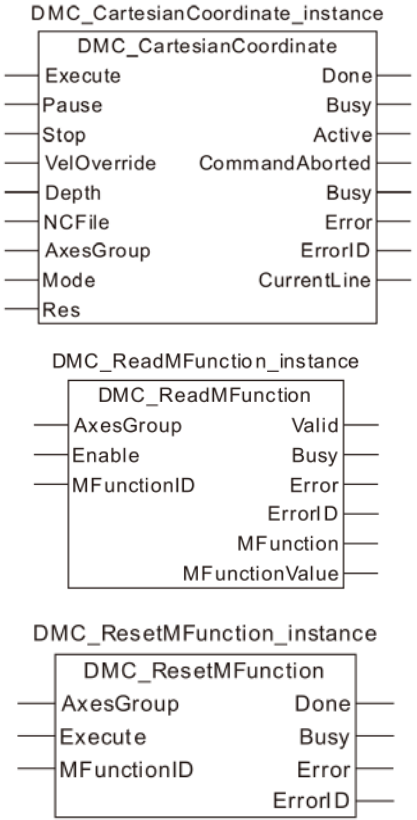
- Controls **Cartesian coordinate robotic arm** to make the interpolation according to G code.
- Performs the pause via the pin **Pause** and stop via the pin **Stop**.
- Adjusts the execution velocity in real time via the pin **VelOverride**.
- Calls different NC files via the pin **NCFile**.
- Gets the row number of G code which is being executed currently via the output pin **CurrentLine**.

DMC_ReadMFunction

- **Reads** the state of M code and the value from it.

DMC_ResetMFunction

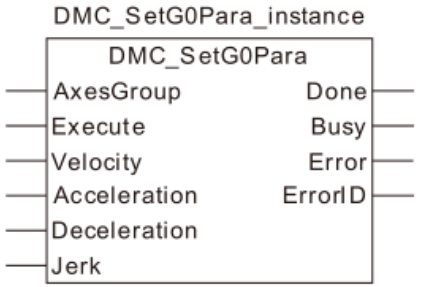
- **Resets** the state of M code.



How to Use NC Function

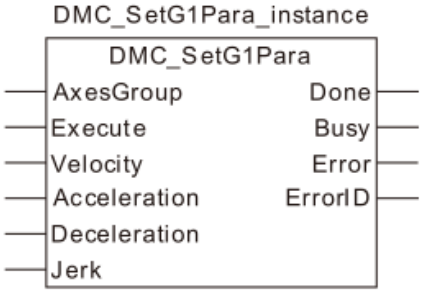
DMC_SetG0Para

- Sets the **velocity, acceleration, deceleration and jerk** of G0.
- When G0 of G codes is executed, the velocity, acceleration, deceleration and jerk will be performed according to the parameters set by DMC_SetG0Para instruction.



DMC_SetG1Para

- Sets the **default velocity, acceleration, deceleration and jerk** of G1/G2/G3.
- When G codes execution reaches G1/G2/G3, G1/G2/G3 will be performed according to the velocity, acceleration and deceleration set by the instruction if E and F values are not specified. Otherwise, they will run based on the filled E and F values in G codes.



Exercise on NC function

How to Use Axes Group Function

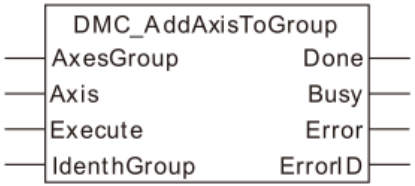
How to Use Axes Group Function

Axes Group Instructions

DMC_ AddAxisToGroup

- Used to **add** an axis to an axes group. The axis must be added to the group before the axes group is used.
- **IdentInGroup** indicates the number of the axis in the axes group. Range: 1~8. 1 represents X axis, 2 is Y axis and so on.

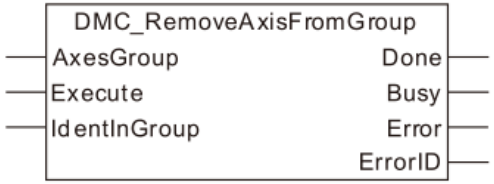
DMC_AddAxisToGroup_instance



DMC_RemoveAxisFromGroup

- Used to **remove** an axis from an axis group.
- The axis can be removed only when the axes group is disabled.

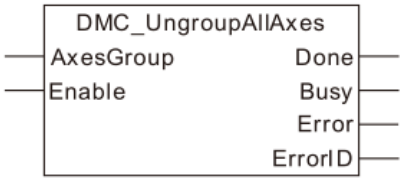
DMC_RemoveAxisFromGroup_instance



DMC_UngroupAllAxes

- Used to **remove all axes in an axes group**.
- The axes can be removed only when the axes group is disabled.

DMC_UngroupAllAxes_instance



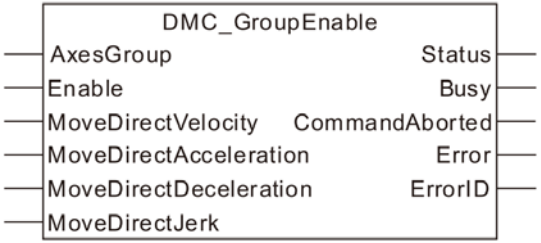
How to Use Axes Group Function

Axes Group Instructions

DMC_ GroupEnable

- Used to **enable** an axes group.
- The axes group **must be enabled** before controlling an axes group to move accordingly. When the axes group is not enabled, the instructions of rapid positioning, linear interpolation and round path interpolation can not be executed.

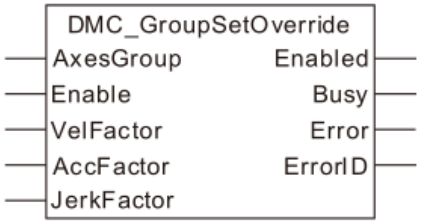
DMC_GroupEnable_instance



DMC_GroupSetOverride

- Used to set **the value of override** for the coordinated motion of an axes group.
- The unit of **VelFactor** is %. “100” means “100%”. The valid range of **VelFactor** value is 0~500.
- **AccFactor** and **JerkFactor** are reserved.

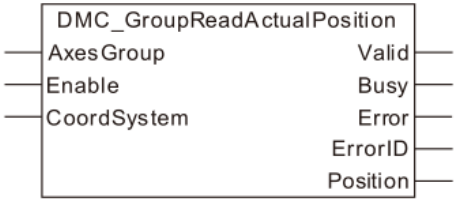
DMC_GroupSetOverride_instance



DMC_GroupReadActualPosition

- Used to read the position of axes in the axes group.
- The output parameter **Position** of the instruction is an array. Every member of an array corresponds to an axis position.

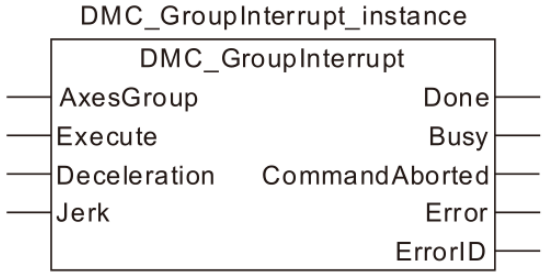
DMC_GroupReadActualPosition_instance



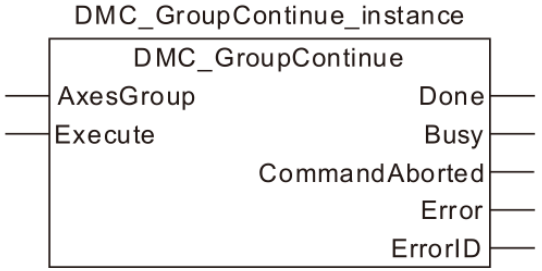
How to Use Axes Group Function

Axes Group Instructions

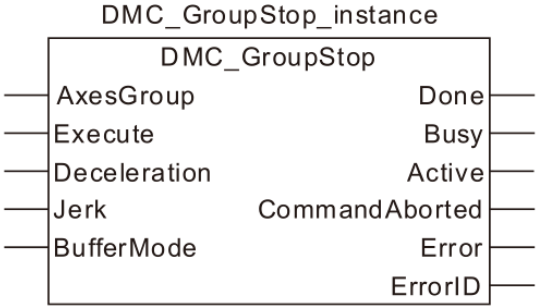
- **DMC_GroupInterrupt**
  - Used to interrupt the motion of the current axes .



- **DMC_GroupContinue**
  - Used to make the interrupted axes group continue to run.



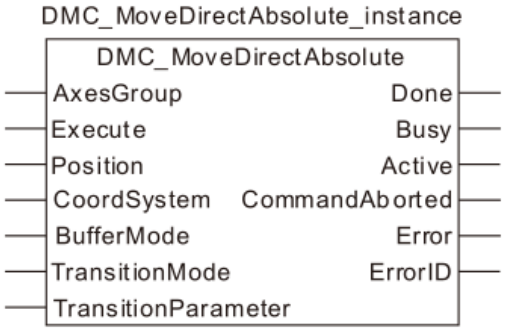
- **DMC_GroupStop**
  - Used to stop the motion of the current axes group.

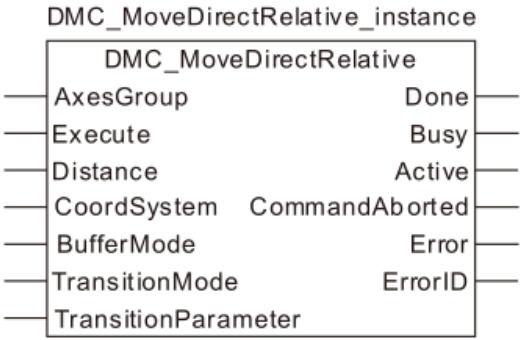


How to Use Axes Group Function

Axes Group Instructions

- **DMC_MoveDirectAbsolute —Rapid positioning (Absolute coordinates)**
- **DMC_MoveDirectRelative —Rapid positioning (Relative coordinates)**
  - Used to control **all axes** in an axes group to move from current positions to the terminal positions at the specified velocity.
  - The values of the array, the input parameter **Position/ Distance** represent the terminal positions for rapid positioning.



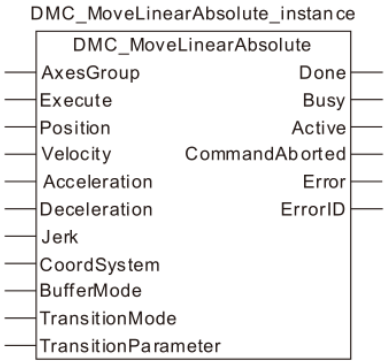


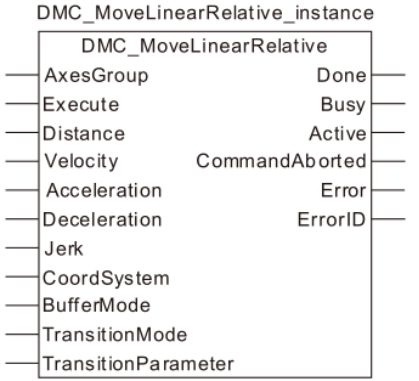
How to Use Axes Group Function

Axes Group Instructions

- **DMC_MoveLinearAbsolute -- Linear interpolation (Absolute coordinates)**
- **DMC_MoveLinearRelative -- Linear interpolation (Relative coordinates)**

  - Controls the **terminal mechanism** to perform the linear interpolation function. The values in the array input parameter Position/ Distance represent the end position of the linear interpolation.
  - Can set the velocity, acceleration and deceleration.
  - Through setting the values of **BufferMode, TransitionMode** and **TransitionParameter**, the transition mode for linear interpolation and previous interpolation instructions can be changed and the effect acquired is similar to G51/G52.





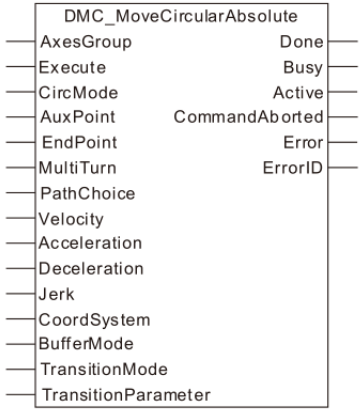
How to Use Axes Group Function

Axes Group Instructions

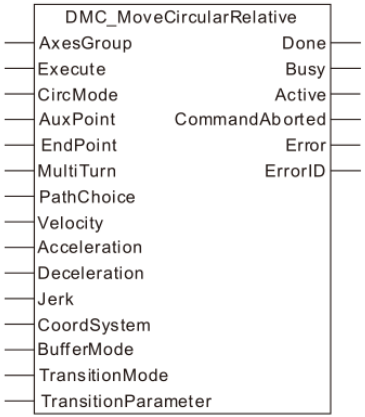
- **DMC_MoveCircularAbsolute – Circular arc interpolation (Absolute coordinates)**
- **DMC_MoveCircularRelative– Circular arc interpolation (Relative coordinates)**

  - Used to control the **terminal mechanism** to perform the circular arc/helical interpolation.
  - Can set the direction for the circular arc/helical interpolation, methods and number of circles.
  - Can set the velocity, acceleration and deceleration.
  - Through setting the values of **BufferMode**, **TransitionMode** and **TransitionParameter**, the transition mode of circular arc/ helical interpolation instructions can be changed and the interpolation effect is similar to G 51.

DMC_MoveCircularAbsolute_instance



DMC_MoveCircularRelative_instance



Exercise on Axes Group Function

How to Use Axes Group Function

TransitionMode

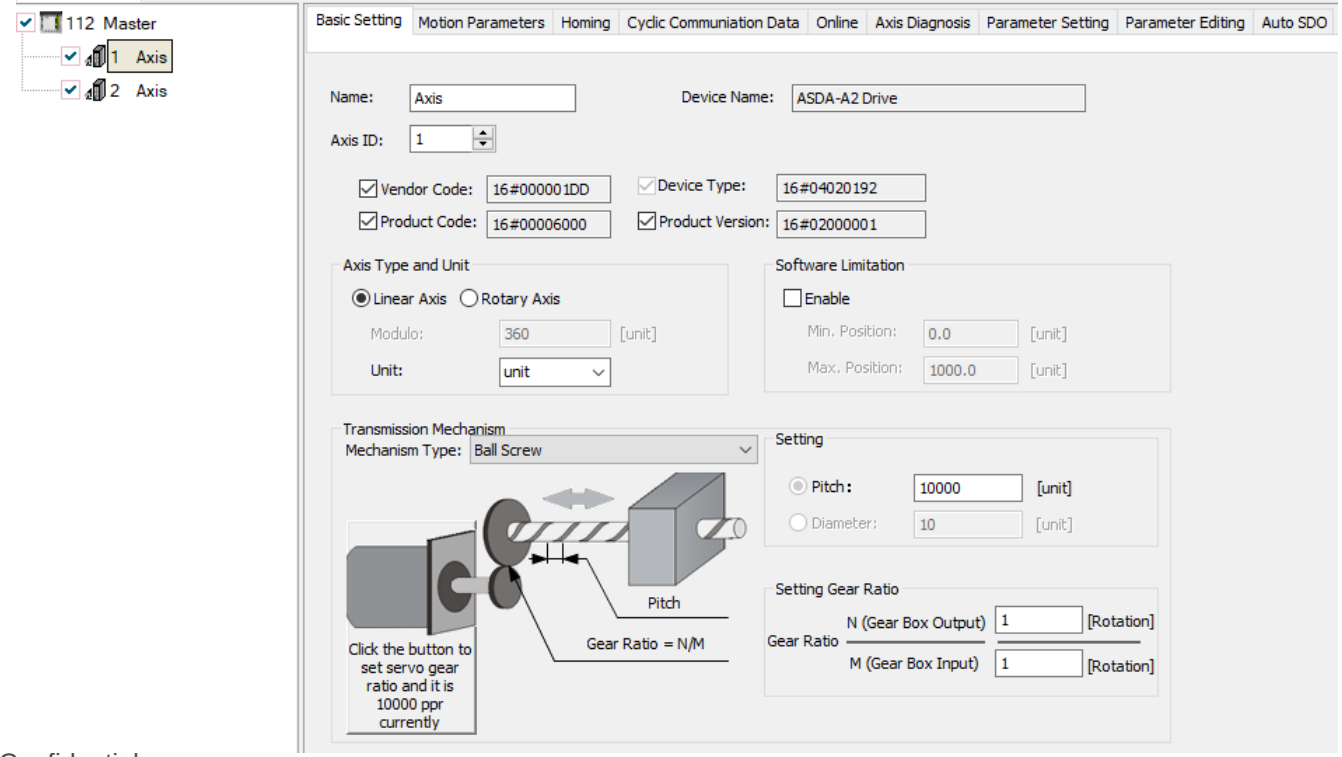
- In the process that an axes group is running under the control of a motion instruction, another motion instruction may be triggered to execute and there are three transition modes for current and previous two motion instructions to choose. **TransitionMode** and **BufferMode** need be matched to use together.
- The parameter **TransitionParameter** value should be set when the **mcTMCornerDistance** mode is used and the parameter represents the transition radius.
- The transition mode is determined together by the parameter **TransitionMode** of current and previous two instructions. When the parameter settings of the two instructions are same, the transition action will happen. Otherwise the execution of the two instructions will both stop.
- See transition modes that axes group instructions support:

	mcBuffered	mcBlendingPrevious
mcTMNone (G50)	DMC_MoveDirectAbsolute DMC_MoveDirectRelative DMC_MoveLinearAbsolute DMC_MoveLinearRelative DMC_MoveCircularAbsolute DMC_MoveCircularRelative	_
mcTMConstantVelocity (G52)	_	DMC_MoveLinearAbsolute DMC_MoveLinearRelative
mcTMCornerDistance (G51)	_	DMC_MoveLinearAbsolute DMC_MoveLinearRelative DMC_MoveCircularAbsolute DMC_MoveCircularRelative

How to Use Axes Group Function

Program Example

Axis Parameter Example

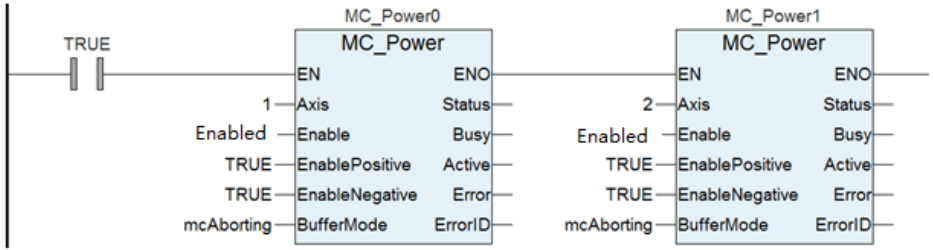


How to Use Axes Group Function

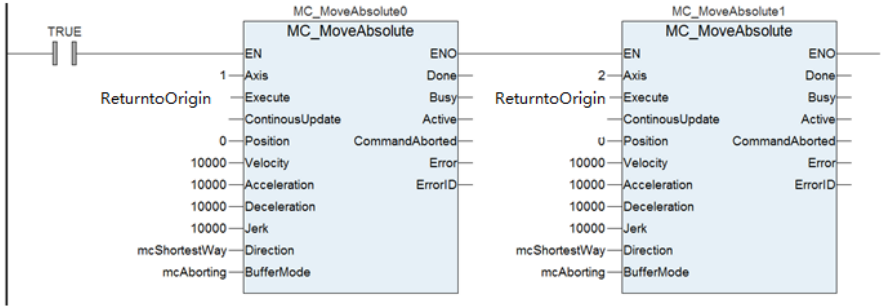
Program Example

◆ Programming Example

- **Axes are enabled.**



- **Axes return to the origins.**

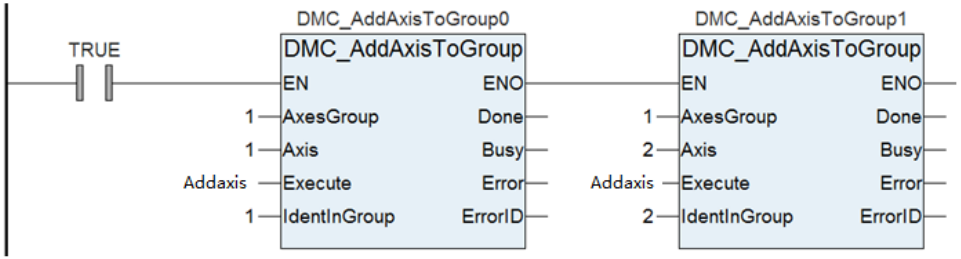


How to Use Axes Group Function

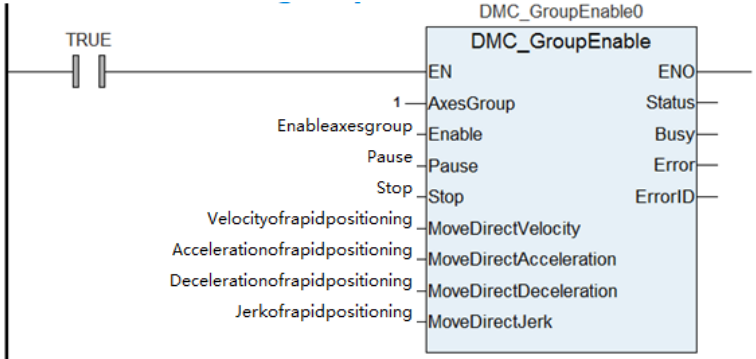
Program Example

◆ Programming Example

- **Add axes to an axes group.**



- **Enable the axes group.**

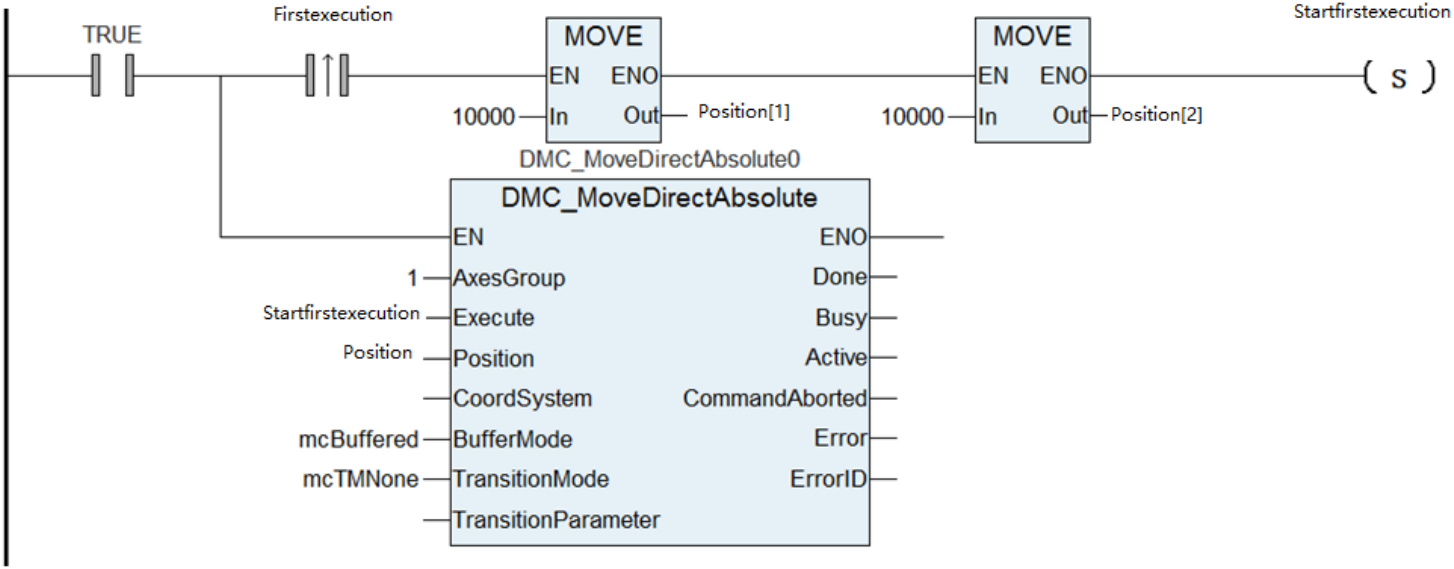


How to Use Axes Group Function

Program Example

◆ Programming Example

- **Executing the axes group motion instruction, the axes group is controlled to starts to move.**



Exercise on axes group function

Robot Function

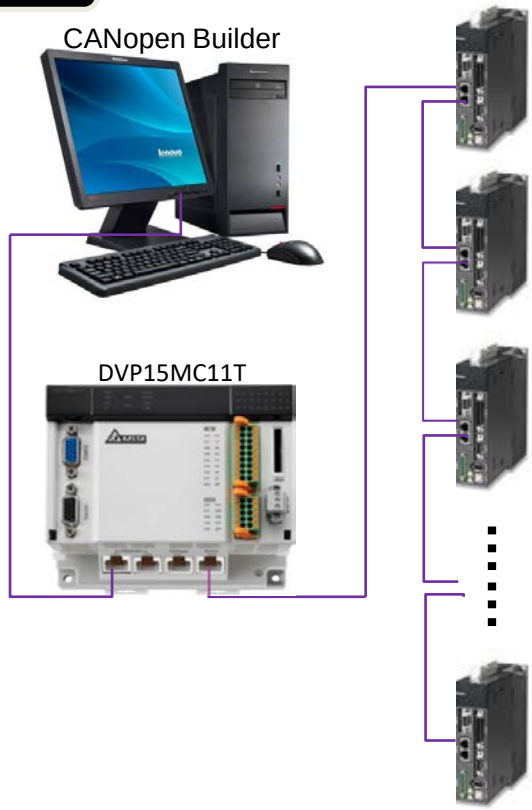
Robot Function

Robots Supported

Robot	Appearance	Application
Cartesian-coordinate robot		Assembly
SCARA		Welding, assembly, carrying and packaging
DELTA		Sifting and picking, carrying and packaging

Robot Function

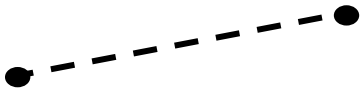
System Architecture



Robot Function

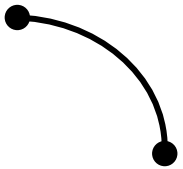
Processing Path for Robot

- **Point to point: MoveDirect**



- **Linear interpolation: MoveLinear**

- **Circular arc interpolation: MoveCircular**



Robot Function

Operation Method

Path Fixing

- Define a motion path
- Edit CNC path file
- Call existing instructions of the robot

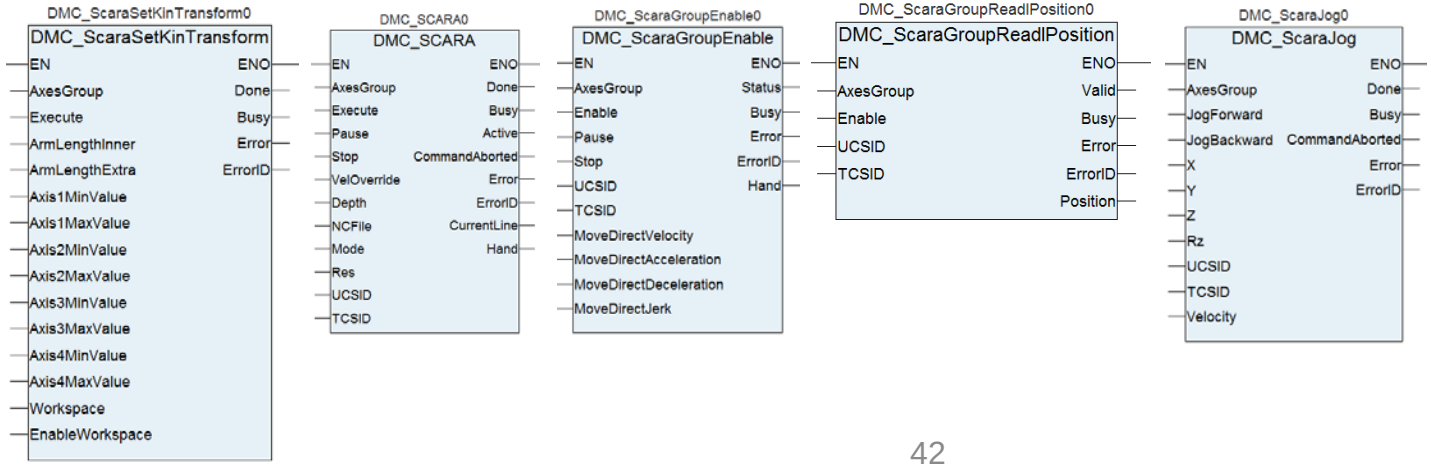
Path Change

- Define the path algorithm
- Use the instruction DMC_GroupEnable of the robot to enable the axes group
- Perform the linear interpolation or circular interpolation instructions based on the acquired path.

Introduction to Instructions

- **Instructions for SCARA robot:**
  - **DMC_ScaraSetKinTransform -- SCARA parameters setting instruction**

  Used for setting mechanism parameters and work space limitation for SCARA robot
  - **DMC_SCARA** -- SCARA G code motion instruction
  - **DMC_ScaraGroupEnable –** Enable SCARA axes group
  - **DMC_ScaraGroupReadPosition–** Read SCARA current position in Cartesian space
  - **DMC_ScaraJog** -- SCARA jog instruction in Cartesian space

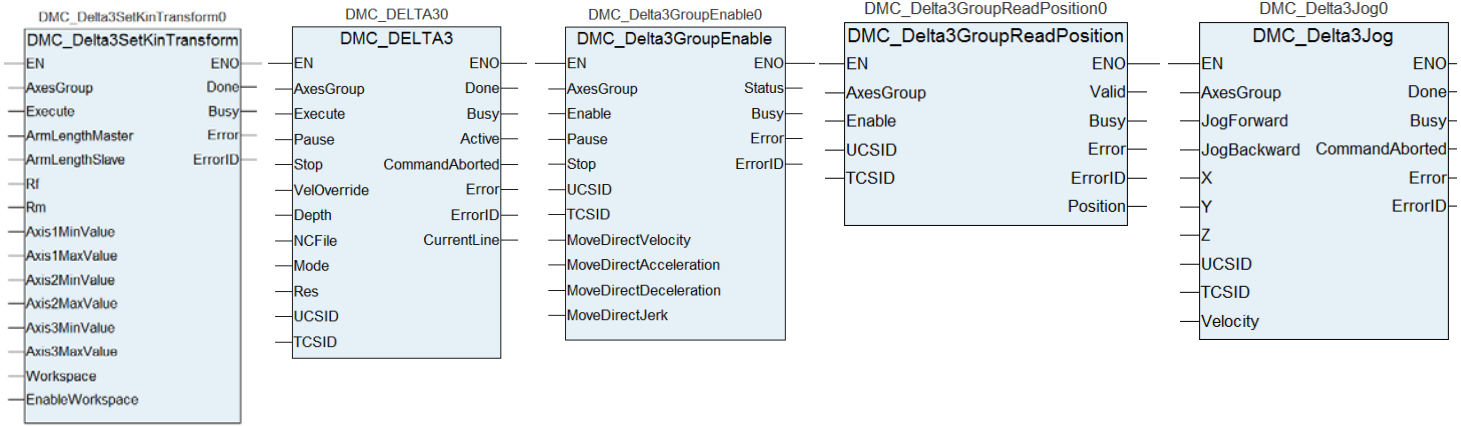


Robot Function

Introduction to Instructions

- **Instructions for Delta3 robot**
  - **DMC_Delta3SetKinTransform-- Delta3 parameters setting instruction**

  Used for setting mechanism parameters and work space limitation for Delta3 robot
  - **DMC_Delta3** -- Delta3 G code motion instruction
  - **DMC_Delta3GroupEnable --** Enable Delta3 axes group
  - **DMC_Delta3GroupReadPosition--**Read Delta3 current position in Cartesian space
  - **DMC_Delta3Jog** -- Delta3 jog instruction in Cartesian space

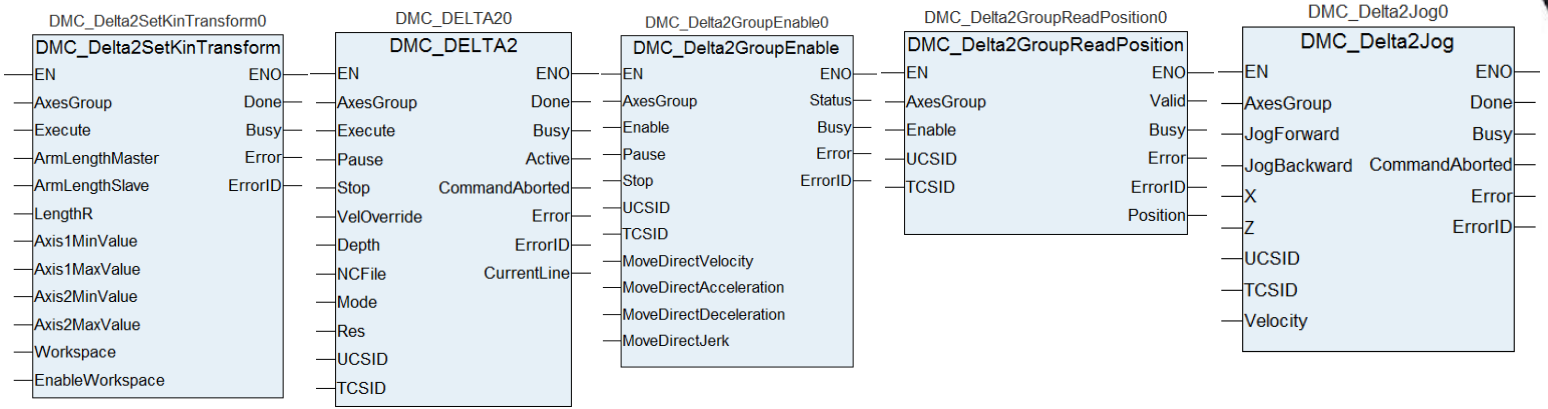


Robot Function

Introduction to Instructions

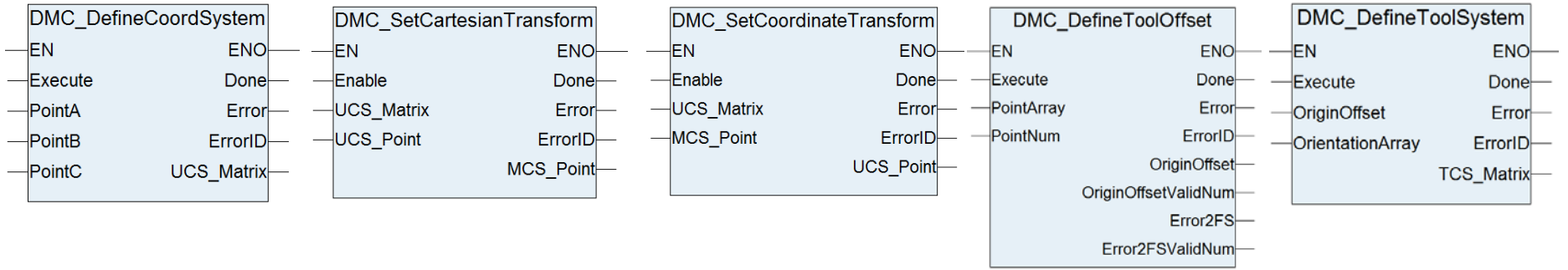
- **Instructions for Delta2 robot**
  - **DMC_Delta2SetKinTransform- - Delta2 parameters setting instruction**

  Used for setting mechanism parameters and work space limitation for Delta2 robot
  - **DMC_Delta2–** Delta2 G code motion instruction
  - **DMC_Delta2GroupEnable--**Enable Delta2 axes group
  - **DMC_Delta2GroupReadPosition--**Read Delta2 current position in Cartesian space
  - **DMC_Delta2Jog--** Delta2 jog instruction in Cartesian space



Introduction to Instructions

- **Coordinate system change instructions**
  - **DMC_DefineCoordSystem--Creates a user coordinate system by adopting the 3-point teaching method.**
  - **DMC_SetCartesianTransform--**Convert the point coordinates in the user coordinate system to the position and orientation in the machine-tool coordinate system.
  - **DMC_SetCoordinateTransform--**Convert the position and orientation in the machine-tool coordinate system into the point coordinates in the user coordinate system.
  - **DMC_DefineToolOffset** --Calculate the origin offset in the tool coordinate system through the multi-point teaching.
  - **DMC_DefineToolSystem--** Create a tool coordinate system by combing the orientation got from the teaching with the origin offset.

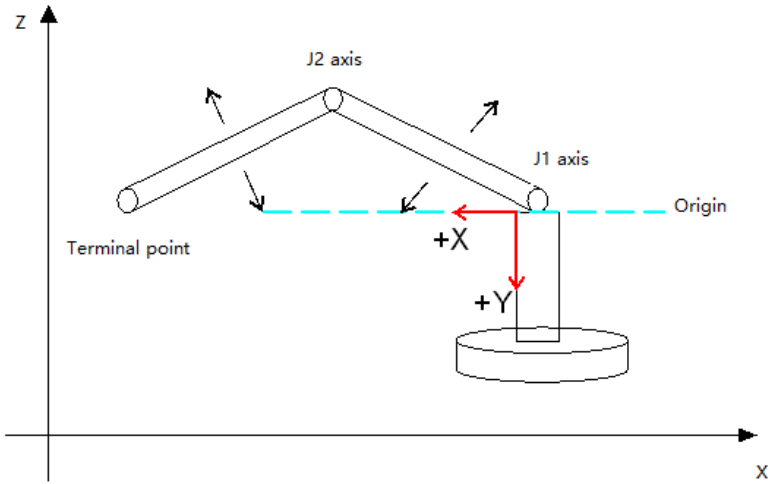


Example on Instructions

- **Mechanism model**
  The right figure is a simple model of a palletizing robot with two robotic arms respectively controlled by J1 axis and J2 axis to achieve the carrying and palletizing in a plane.

- **Model Analysis**
  This model is a simple SCARA robot model, which realizes the SCARA motion in the XY plane of the space.

- **Parameters Setting**
  To apply the SCARA model, you need to set up the following parameters.
  - The lengths of arm1 and arm 2 are 500mm and 800mm respectively.
  - The minimum limit of joint angle 1 is -45° and the maximum limit is 60° .
  - The minimum limit of joint angle 2 is 30° and the maximum limit is 140° .
  - The minimum limit of joint angle 3 is -10° and the maximum limit is 10° . (They need to be set though they are not useful here.)
  - The minimum limit of joint angle 4 is -360° and the maximum limit is 360° . (They need to be set though they are not useful here.)



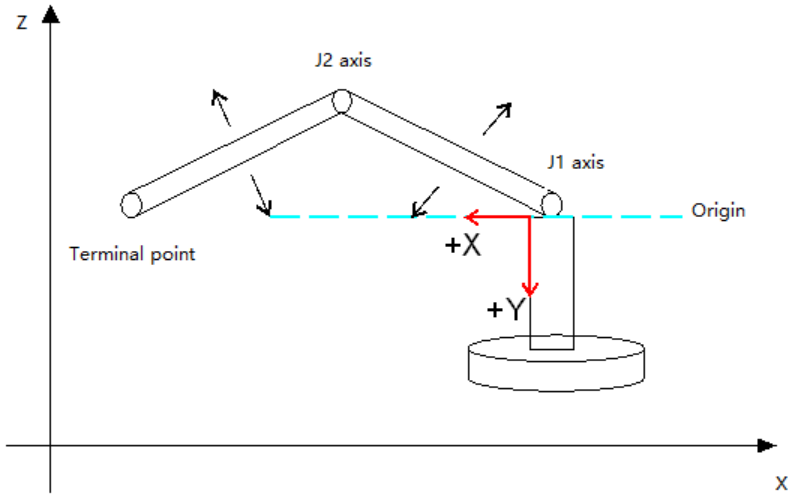
Robot Function

Example on Instructions

- **Define actions**
  Define the actions for carrying items. The actions start from (900,100), to (900, 0), to (800,0), and then to (800,100) where the items are put down.

- **Programming**
  Enable the axes, return them to the origins and add them to axes groups.

  - **Method 1: G codes**
    Write G codes, then set SCARA parameters via DMC_ScaraSetKinTransform and finally use DMC_SCARA to control axes groups to move according to G Codes.
  - **Method 2: Axes group instructions**
    Use DMC_ScaraSetKinTransform to set SCARA parameters, Use DMC_ScaraGroupEnable to enable axes group, and then use DMC_MoveLinearAbsolute or DMC_MoveLinearRelative to control axes groups to move.



Smarter. Greener. Together.

To learn more about Delta, please visit www.deltaww.com.