Programming Instructions

<<<<

Catalogue

Appendix-Instruction Set

> Bind variable parameter description

The following instructions add bind variables to the positional variable type:

P$INT: When the local integer variable (INT) is assigned a certain value, the local point P is the point represented by the value.

Usage example: I001 = 2 P$I001 is equivalent to P0002

P$GINT: When the global integer variable (GINT) is assigned a certain value, the local point P is the point represented by the value.

Usage example: GI001 =3 P$GI001 is equivalent to P0003

GP$INT: When the local integer variable (INT) is assigned a certain value, the global point GP is the point represented by the value.

Usage example: I001 = 4 GP$I001 is equivalent to GP0004

GP$GINT: When the global integer variable (GINT) is assigned a certain value, the global point GP is the point represented by the value.

Usage example: GI001 = 5 GP$GI001 is equivalent to GP0005

E$INT: When the local integer variable (INT) is assigned a certain value, the local point E is the point represented by the value.

Usage example: I001 = 6 E$I001 is equivalent to E0006

E$GINT: When the global integer variable (GINT) is assigned a certain value, the local point E is the point represented by the value.

Usage example: I001 = 7 E$I001 is equivalent to E0007

GE$INT: When the local integer variable (INT) is assigned a certain value, the global point GE is the point represented by the value.

Usage example: I001 = 8 GE$I001 is equivalent to GE0008

GE$GINT: When the global integer variable (GINT) is assigned a certain value, the global point GE is the point represented by the value.

Usage example: I001 = 9 GE$I001 is equivalent to GE0009

> Motion control class

MOVJ-Point to point

Function

Move to the target point using joint interpolation. This instruction is used in the section where the robot is not constrained by trajectory in moving to the target point. The robot runs at the fastest speed in space.

Parameter description

P/GP: Use either a local position variable (P) or a global position variable (GP). When the value is "New", inserting this instruction will create a new P variable and record the robot's current position into that P variable.

VJ: The speed of the joint interpolation, the range is 1-100, and the unit is percentage. The actual movement speed is the maximum axis speed in the robot joint parameters multiplied by this percentage.

PL: Position level, range 0-5.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to the same value as VJ

DEC: Deceleration rate, the range is 1-100, the unit is percentage. It is recommended to set it to the same value as VJ.

TIME: Time, the range is a non-negative integer, and the unit is ms. Early execution time of the next instruction.

Note: When modifying the speed of MOVJ instruction, the acceleration & deceleration will be automatically displayed in a multiple of 1:1 with the speed. If you need to modify the acceleration or deceleration, you can do it manually.

Usage examples

MOVJ P0001 VJ = 10 % PL = 1 ACC = 10 DEC = 10 0

MOVJ GP0002 VJ = 10 % PL = 0 ACC = 7 DEC = 11 0

MOVL-Linear

Function

Move to the target point using linear interpolation. During the robot's movement to the target point, the movement trajectory of the robot end is a straight line.

Parameter description

P/GP: Use either a local position variable (P) or a global position variable (GP). When the value is "New", inserting this instruction will create a new P variable and record the robot's current position into that P variable.

V: Motion speed, the range is 1-1000 (The default Cartesian parameter maximum speed is 1000, the range varies according to the actual filled Cartesian parameter), the unit is mm/s.

PL: Position level, range 0-5.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

DEC: Deceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

TIME: Time, the range is a non-negative integer, and the unit is ms. Early execution time of the next instruction.

Note: When modifying the speed of MOVL instruction, the acceleration & deceleration will be automatically displayed in a multiple of 1:10 with the speed. If you need to modify the acceleration or deceleration, you can do it manually.

Usage examples

MOVL P0003 V = 200 mm/s PL = 2 ACC = 20 DEC = 20 0

MOVC-Circular

Note: MOVC, MOVCA, MOVCADOUBL, MOVS, MOVCEXT and other motion instructions that include circular, full circle and curve cannot be used alone, and their early execution speed, PL and other parameters are affected by the first (circular/full circle) instruction

Function

The robot moves in a circle through the 3 points taught by circular interpolation.

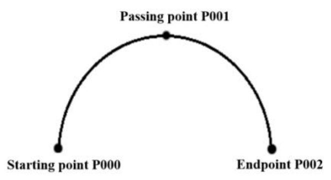
If the robot axis is taught by circular interpolation, the movement command is MOVC. The robot needs a MOVJ or MOVL instruction plus two MOVC instructions to walk a complete arc curve.

The starting point of single arc and the first arc of continuous arc can only be MOVJ or MOVL.

Single arc

When there is only one arc, as shown in the figure below, use circular interpolation to teach the three points of P000-P002.

If joint interpolation or linear interpolation is used to teach P000 before entering the arc, the trajectory of P000-P001 will automatically become a straight line.

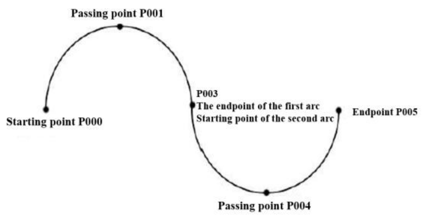


P000-Joint/Linear

P001-P002-Circular

Continuous arc

As shown in the figure below, when there are 2 or more consecutive circular arcs with curvature change, the arcs will eventually separate one by one. Therefore, please add joint or linear interpolation points at the connection point of the previous arc and the next arc.



P000-Joint/Linear

P001-P002-Circular

Parameter description

P/GP: Use either a local position variable (P) or a global position variable (GP). When the value is "New", inserting this instruction will create a new P variable and record the robot's current position into that P variable.

V: Motion speed, the range is 1-1000 (The default Cartesian parameter maximum speed is 1000, the range varies according to the actual filled Cartesian parameter), the unit is mm/s.

PL: Position level, range 0-5.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

DEC: Deceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

TIME: Time, the range is a non-negative integer, and the unit is ms. Early execution time of the next instruction.

Note: When modifying the speed of MOVC instruction, the acceleration & deceleration will be automatically displayed in a multiple of 1:10 with the speed. If you need to modify the acceleration or deceleration, you can do it manually.

Usage examples

MOVJ P0001 VJ = 10 % PL = 0 ACC = 1 DEC = 1 0

MOVC P0002 V = 100 mm/s PL = 0 ACC = 10 DEC = 10 0

MOVC P0003 V = 100mm/s PL = 0 ACC = 5 DEC = 8 0

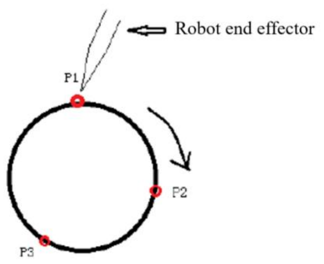
MOVCA-Full circle

Function

The robot walks a complete circle by teaching the starting point (MOVJ or MOVL) and two passing points (MOVCA) of the circle.

Instruction insertion prerequisites

Click on the "Tools" button in the upper status bar and select the previously calibrated tool hand.



Insertion steps for four instructions:

Click "Insert", click "Coordinate Switch Class", select "SWITCHTOOL" instruction, select the previously calibrated tool hand number.

Move to any point of the circle to be drawn such as P1 in the figure, click on "Insert", click on the "Motion Control" class and select MOVJ or MOVL.

Then move to any point of the circle to be drawn such as P2 in the figure (different from the point in step 2), click the "Coordinate System" button in the upper status bar, select the "Tool" coordinate system, click "Insert", click on the "Motion Control" class and select MOVCA

Move to any point of the circle to be drawn such as P3 in the figure (different from the points in steps 2 and 3), click the "Coordinate System" button in the upper status bar, select the "Tool" coordinate system, and click "Insert", click the "Motion Control" class, select MOVCA

Parameter description

P/GP: Use either a local position variable (P) or a global position variable (GP). When the value is "New", inserting this instruction will create a new P variable and record the robot's current position into that P variable.

V: Motion speed, the range is 1-1000 (The default Cartesian parameter maximum speed is 1000, the range varies according to the actual filled Cartesian parameter), the unit is mm/s.

PL: Position level, range 0-5.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

DEC: Deceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

TIME: Time, the range is a non-negative integer, and the unit is ms. Early execution time of the next instruction.

SPIN:

Attitude unchanged: Take a full circle trajectory with the same attitude as the P0001 calibrated attitude

6-axis non-rotating: Walk a full circle trajectory according to the calibrated attitude, while the 6-axis is fixed

6-axis rotating: Walk a full circle trajectory according to the calibrated attitude, and the 6-axis will rotate 360 degrees at the same time

Note: When modifying the speed of MOVCA instruction, the acceleration & deceleration will be automatically displayed in a multiple of 1:10 with the speed. If you need to modify the acceleration or deceleration, you can do it manually.

Usage examples

MOVJ P0001 VJ = 10 ％ PL = 0 ACC = 10 DEC = 10SPIN=1 0

MOVCA P0002 V = 100 mm/s PL = 0 ACC = 10 DEC = 10SPIN=1 0

MOVCA P0003 V = 100mm/s PL = 0 ACC = 10 DEC = 10SPIN=1 0

MOVS-Curve interpolation

Function

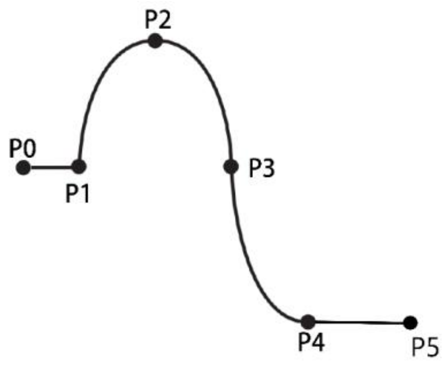
In welding, cutting, fusion welding, priming and other operations, if the free curve interpolation is used, the teaching operations for irregularly curved workpieces can become easier.

The trajectory is a spline curve passing through four points.

If free curve interpolation is used to teach the robot axis, the movement command is MOVS.

Single MOVS

Teach 4 points of P1-P4 as shown in the figure below. P1-P4 form a spline curve



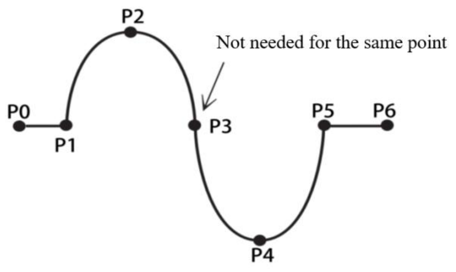
P0-Joint/Linear (The first motion instruction of the program cannot be MOVS)

P1-P4-Curve interpolation

P5-Joint/Linear

Continuous MOVS

A spline curve consisting of more than 4 points. P1-P5 form a spline curve



P0-Joint/Linear

P1-P5-Curve interpolation

P6-Joint/Linear

Note: The curve requires at least four curve points

Parameter description

P/GP: Use either a local position variable (P) or a global position variable (GP). When the value is "New", inserting this instruction will create a new P variable and record the robot's current position into that P variable.

V: Motion speed, the range is 1-1000 (The default Cartesian parameter maximum speed is 1000, the range varies according to the actual filled Cartesian parameter), the unit is mm/s.

PL: Position level, range 0-5.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

DEC: Deceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

TIME: Time, the range is a non-negative integer, and the unit is ms. Early execution time of the next instruction.

Note: When modifying the speed of MOVS instruction, the acceleration & deceleration will be automatically displayed in a multiple of 1:10 with the speed. If you need to modify the acceleration or deceleration, you can do it manually.

Usage examples

MOVJ P0001 VJ = 10 % PL = 0 ACC = 10 DEC = 10 0

MOVS P0002 V = 100 mm/s PL = 0 ACC = 10 DEC = 10 0

MOVS P0003 V = 100mm/s PL = 0 ACC = 10 DEC = 10 0

MOVS P0004 V = 100 mm/s PL = 0 ACC = 10 DEC = 10 0

MOVS P0005 V = 100mm/s PL = 0 ACC = 10 DEC = 10 0

IMOV-Increment

Function

Move from the current position according to the set incremental distance by means of joint or linear interpolation.

Parameter description

RP: Incremental variable, you can choose four kinds of coordinate systems: joint, Cartesian, tool and user, and fill in positive number for positive direction and negative number for negative direction for the corresponding axis. If not moving, fill in 0.

V/VJ:

When RP is the value in the joint coordinate system, here is VJ, i.e. the speed of joint interpolation, the range is 1-100, and the unit is percentage.

The actual movement speed is the maximum axis speed in the robot joint parameters multiplied by this percentage. When RP is the value in the Cartesian, tool, and user coordinate systems, here is V, i.e. the motion speed, the range is 1-1000 (The default Cartesian parameter maximum speed is 1000, the range varies according to the actual filled Cartesian parameter), and the unit is mm/s.

PL: Position level, range 0-5.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10% or VJ.

DEC: Deceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10% or VJ.

TIME: Time, the range is a non-negative integer, and the unit is ms. Early execution time of the next instruction

Note: When modifying the speed of IMOV instruction, the acceleration & deceleration will be automatically displayed in a multiple of 1:10 with the speed. If you need to modify the acceleration or deceleration, you can do it manually.

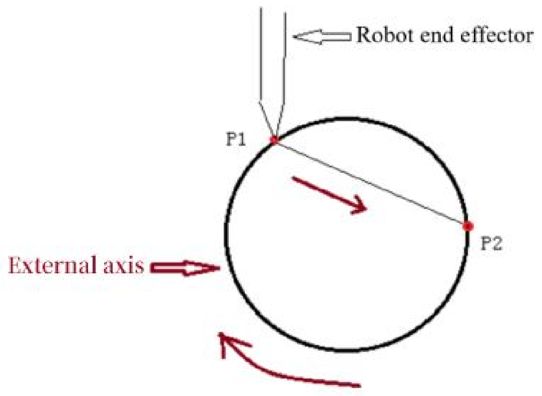
Usage examples

IMOV RP0001 V=10mm/s BF PL=0 ACC=1 DEC=1

MOVJEXT-External axis point to point

Function

The robot moves to the teaching position by means of joint interpolation, and the external axis moves by means of joint interpolation.



Parameter description

E: A variable that records robot and external axis position data simultaneously. When the value is "New", inserting this instruction will create a new E variable, and record the current position of the robot and the external axis to this E variable.

VJ: The speed of the joint interpolation, the range is 1-100, and the unit is percentage. The actual movement speed is the maximum axis speed in the robot joint parameters multiplied by this percentage. The external axis speed changes with the robot speed.

PL: Position level, range 0-5.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to the same value as VJ.

DEC: Deceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to the same value as VJ.

TIME: Time, the range is a non-negative integer, and the unit is ms. Early execution time of the next instruction

Note: When modifying the speed of MOVJEXT instruction, the acceleration & deceleration will be automatically displayed in a multiple of 1:1 with the speed. If you need to modify the acceleration or deceleration, you can do it manually.

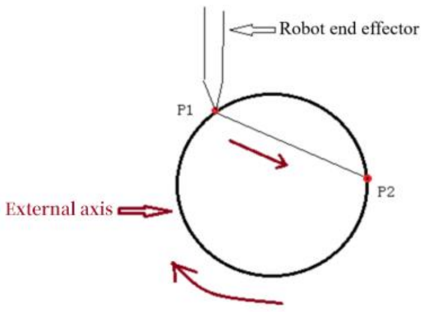
Usage examples

MOVJEXTE0001 VJ = 10 % PL = 0 ACC= 10 DEC = 10 0

MOVLEXT-External axis linear

Function

The robot moves to the teaching position by means of linear interpolation, and the external axis moves by means of joint interpolation.



Parameter description

E: A variable that records robot and external axis position data simultaneously. When the value is "New", inserting this instruction will create a new E variable, and record the current position of the robot and the external axis to this E variable.

V: Motion speed, the range is 1-1000 (The default Cartesian parameter maximum speed is 1000, the range varies according to the actual filled Cartesian parameter), the unit is mm/s. The external axis speed changes with the robot speed.

PL: Position level, range 0-5.

SYNC: Whether the robot moves synchronously with the external axis: when "Yes" is selected, the robot moves in a straight line in collaboration with the external axis; when "No" is selected, the robot moves in a straight line in space, and the external axis moves independently to the target angle.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

DEC: Deceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

Note: When modifying the speed of MOVLEXT instruction, the acceleration & deceleration will be automatically displayed in a multiple of 1:10 with the speed. If you need to modify the acceleration or deceleration, you can do it manually.

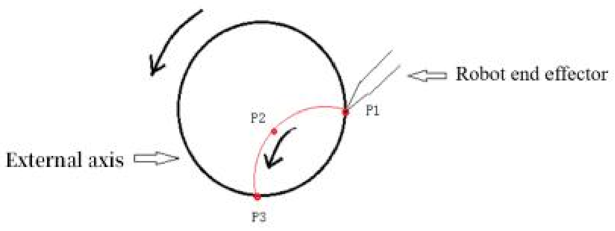
Usage examples

MOVLEXT E0002 V = 10 mm/s PL = 0 ACC = 1 DEC = 1 SYNC = 0 0

MOVCEXT-External axis circular

Function

The robot moves to the teaching position by means of circular interpolation, and the external axis moves by means of joint interpolation.



Parameter description

E: A variable that records robot and external axis position data simultaneously. When the value is "New", inserting this instruction will create a new E variable, and record the current position of the robot and the external axis to this E variable.

V: Robot motion speed, the range is 2-2000, the unit is mm/s. The external axis speed changes with the robot speed.

PL: Position level, range 0-5.

SYNC: Whether the robot moves synchronously with the external axis: when "Yes" is selected, the robot moves in an arc in collaboration with the external axis; when "No" is selected, the robot moves in an arc in space, and the external axis moves independently to the target angle.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

DEC: Deceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

TIME: Time, the range is a non-negative integer, and the unit is ms. Early execution time of the next instruction.

Note: When modifying the speed of MOVCEXT instruction, the acceleration & deceleration will be automatically displayed in a multiple of 1:10 with the speed. If you need to modify the acceleration or deceleration, you can do it manually.

Usage examples

MOVLEXT E0002 V = 10 mm/s PL = 0 ACC = 1 DEC = 1 SYNC = 1 0

MOVCEXT E0003 V = 10 mm/s PL = 0 ACC = 1 DEC = 1 SYNC = 1 0

MOVCEXT E0004 V = 10 mm/s PL = 0 ACC = 1 DEC = 1 SYNC = 1 0

SPEED-Global speed

Function

The motion speed of all motion class instructions below the SPEED instruction is: instruction speed * speed in the upper status bar * SPEED (%).

Parameter description

Global speed (%): Speed percentage: 1-200.

Usage examples

SPEED= 9 %

SAMOV-Fixed-point movement

Function

The robot moves to a preset absolute position by joint interpolation.

If you do not want to move an axis, please leave the coordinate of the axis blank. (Do not fill in 0)

Parameter description

AP: Absolute position; four coordinate systems can be selected: joint, Cartesian, tool, and user; if the corresponding axis is not filled in, the corresponding axis will not move.

V/VJ:

When AP is the value in the joint coordinate system, here is VJ, i.e. the speed of joint interpolation, the range is 1-100, and the unit is percentage.

The actual movement speed is the maximum axis speed in the robot joint parameters multiplied by this percentage. When AP is the value in the Cartesian, tool, and user coordinate systems, here is V, i.e. the motion speed, the range is 1-1000 (The default Cartesian parameter maximum speed is 1000, the range varies according to the actual filled Cartesian parameter), and the unit is mm/s.

PL: Position level, range 0-5.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10% or VJ.

DEC: Deceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10% or VJ.

TIME: Time, the range is a non-negative integer, and the unit is ms. Early execution time of the next instruction.

Note: When modifying the speed of SAMOV instruction, the acceleration & deceleration will be automatically displayed in a multiple of 1:1 with the speed. If you need to modify the acceleration or deceleration, you can do it manually.

Usage examples

SAMOV AP0001 VJ= 10 %PL= 2 ACC= 10 DEC= 10

MOVJDOUBLE-Dual robot point to point

Function

When set to two robots, make two robots move to the target position by joint interpolation at the same time. Start and stop at the same time.

Parameter description

E: A variable that records the position data of two robots at the same time. When the value is "New", inserting this instruction will create a new E variable and record the current positions of the two robots to this E variable.

VJ: The speed of joint interpolation, the range is 1-100, and the unit is percentage. The actual movement speed is the maximum speed of the axis in the robot joint parameters multiplied by this percentage. The speeds of the two robots are synchronized.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to the same value as VJ.

DEC: Deceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to the same value as VJ.

TIME: Time, the range is a non-negative integer, and the unit is ms. Early execution time of the next instruction.

Note: When modifying the speed of MOVJDOUBLE instruction, the acceleration & deceleration will be automatically displayed in a multiple of 1:1 with the speed. If you need to modify the acceleration or deceleration, you can do it manually.

Usage examples

MOVJDOUBLE E0001 VJ = 10 % PL = 0 ACC= 10 DEC = 10 0

MOVLDOUBLE-Dual robot linear

Function

When set to two robots, make two robots move to the target position by linear interpolation at the same time. Start and stop at the same time.

Parameter description

E: A variable that records the position data of two robots at the same time. When the value is "New", inserting this instruction will create a new E variable and record the current positions of the two robots to this E variable.

V: Robot motion speed, the range is 1-1000 (The default Cartesian parameter maximum speed is 1000, the range varies according to the actual filled Cartesian parameter), and the unit is mm/s. The speeds of the two robots are synchronized.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

DEC: Deceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

TIME: Time, the range is a non-negative integer, and the unit is ms. Early execution time of the next instruction.

Note: When modifying the speed of MOVLDOUBLE instruction, the acceleration & deceleration will be automatically displayed in a multiple of 1:10 with the speed. If you need to modify the acceleration or deceleration, you can do it manually.

Usage examples

MOVLDOUBLE E0001 V = 100 mm/s PL = 0 ACC= 10 DEC = 10 0

MOVCDOUBLE-Dual robot circular

Function

When set to two robots, make two robots move to the target position by circular interpolation at the same time. Start and stop at the same time.

Parameter description

E: A variable that records the position data of two robots at the same time. When the value is "New", inserting this instruction will create a new E variable and record the current positions of the two robots to this E variable.

V: Robot motion speed, the range is 1-1000 (The default Cartesian parameter maximum speed is 1000, the range varies according to the actual filled Cartesian parameter), and the unit is mm/s. The speeds of the two robots are synchronized.

PL: Position level, range 0-5.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

DEC: Deceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

TIME: Time, the range is a non-negative integer, and the unit is ms. Early execution time of the next instruction.

Note: When modifying the speed of MOVCDOUBLE instruction, the acceleration & deceleration will be automatically displayed in a multiple of 1:10 with the speed. If you need to modify the acceleration or deceleration, you can do it manually.

Usage examples

MOVLDOUBLE E0001 VJ = 10 % PL = 0 ACC= 10 DEC = 10 0

MOVCDOUBLE E0002V = 100 mm/s PL = 0 ACC= 10 DEC = 10 0

MOVCDOUBLE E0003V = 100mm/s PL = 0 ACC= 10 DEC = 10 0

MOVCADOUBLE-Dual robot full circle

Function

When set to two robots, make the two robots move to the target position at the same time through full circle interpolation. Start and stop at the same time.

Parameter description

E: A variable that records the position data of two robots at the same time. When the value is "New", inserting this instruction will create a new E variable and record the current positions of the two robots to this E variable.

V: Robot motion speed, the range is 1-1000 (The default Cartesian parameter maximum speed is 1000, the range varies according to the actual filled Cartesian parameter), and the unit is mm/s. The speeds of the two robots are synchronized.

PL: Position level, range 0-5.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

DEC: Deceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10%.

TIME: Time, the range is a non-negative integer, and the unit is ms. Early execution time of the next instruction.

Note: When modifying the speed of MOVCADOUBLE instruction, the acceleration & deceleration will be automatically displayed in a multiple of 1:10 with the speed. If you need to modify the acceleration or deceleration, you can do it manually.

Usage examples

MOVLDOUBLE E0001 VJ = 10 % PL = 0 ACC= 10 DEC = 10 0

MOVCADOUBLE E0002 V = 100 mm/s PL = 0 ACC= 10 DEC = 10 0

MOVCADOUBLE E0003 V = 100 mm/s PL = 0 ACC= 10 DEC = 10 0

MOVCOMM-External point

Function

Move to the point sent by the external device to the controller through Modbus or TCP in the specified interpolation mode.

Parameter description

Interpolation method: The interpolation method used when moving to the target point, including joint, linear, curve.

V/VJ:

When B is the value in the joint coordinate system, here is VJ, i.e. the speed of joint interpolation, the range is 1-100, and the unit is percentage.

The actual movement speed is the maximum axis speed in the robot joint parameters multiplied by this percentage. When B is the value in the Cartesian, tool, and user coordinate systems, here is V, i.e. the motion speed, the range is 1-1000 (The default Cartesian parameter maximum speed is 1000, the range varies according to the actual filled Cartesian parameter), and the unit is mm/s.

PL: Position level, range 0-5.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10% or VJ.

DEC: Deceleration ratio, the range is 1-100, the unit is percentage. It is recommended to set it to V*10% or VJ.

TIME: Time, the range is a non-negative integer, and the unit is ms. Early execution time of the next instruction.

Usage examples

MOVCOMM MOVL VJ= 10 mm/s PL = 0 ACC = 1 DEC = 1 0

EXTMOV-External axis follow

Function

Instruction for external axes to follow the robot at a speed that is a multiple of the robot's linear speed or at a constant speed.

Parameter description

External axis: One axis of O1-O5 can be selected to follow

Type:

Follow type: Change speed with robot real time linear speed

K: External axis speed (°/s)=K*Linear speed (mm/s)

Constant speed type: Run at a constant speed

Speed value source: optional INT/DOUBLE/GINT/GDOUBLE/Hand-filled.

Variable name: Used to select variables when the speed value source is INT/DOUBLE/GINT/GDOUBLE

Hand-filled value: When the speed value source is hand-filled, it is used to input the speed value for constant operation

Usage examples

EXTMOV O1 FOLLOW 22.22

GEARIN-Electronic gear

Function

An instruction to make a certain axis of an external axis move along with a certain axis of the robot.

Parameter description

Spindle: J1~J6 axes of the robot can be selected

External axis: One axis of O1-O5 can be selected to follow (O3 is not supported at the moment)

Proportional relationship K: Follow axis speed (°/s) = K*Spindle speed (°/s)

Usage examples

GEARIN J1 O1 22.22

MRESET-Reset external axis multi-turn rotation amount

Function

The maximum and minimum limits set according to the external axis rotation axis. When the external axis rotates beyond the limit, use this instruction to reset the external peripheral coordinates and continue to rotate, so that the external axis will not report an error due to overrun.

Parameter description

MRESET: all axes or single axis

Usage examples

MRESET 0

DRAG_TRAJECTORY-Drag teaching

Function

The robot runs according to the previously recorded trajectory.

Parameter description

Track name: The name of the robot track

Playback rate: Motion speed (0~500%)

Usage examples

DRAG_TRAJECTORY Track1 20%

SWITHCPAYLOAD-Switch load parameters

Function

In actual operation, the actual load and load parameter match

SWITHCPAYLOAD instruction is used to switch load parameters

It affects collision detection and torque feedforward

Parameter description

Load number: You can fill in the tool number or use the bind variable function

Usage examples

SWITHCPAYLOAD 1

MOVARCH-Arch motion

Function

Allows the robot to move along an arch-type trajectory

Parameter description

P/GP: Use either a local position variable (P) or a global position variable (GP). When the value is "New", inserting this instruction will create a new P variable and record the robot's current position into that P variable.

V: Motion speed, the range is 1-1000 (The default Cartesian parameter maximum speed is 1000, the range varies according to the actual filled Cartesian parameter), and the unit is mm/s.

PL: Position level, range 0-5.

ACC: Acceleration ratio, the range is 1-100, the unit is percentage.

DEC: Deceleration rate, the range is 1-100, the unit is percentage.

Displacement axis: (X, Y, Z) The axis that is displaced during arch-type movement, and the standard arch-type movement is displaced in the Z-axis direction.

Displacement distance: the distance that needs to be displaced on the displacement axis. The standard arch-type movement is 25mm displacement on the Z axis.

TIME: Time, the range is a non-negative integer, and the unit is ms. Early execution time of the next instruction.

View trajectory diagram: You can view the arch-type motion trajectory diagram

Note: When modifying the speed of MOVARCH instruction, the acceleration & deceleration will be automatically displayed in a multiple of 1:1 with the speed. If you need to modify the acceleration or deceleration, you can do it manually.

Usage examples

MOVARCH P0001 V=10 PL=0 ACC=10 DEC=10 X 10 0

MOVARCH GP0001 V=10 PL=0 ACC=10 DEC=10 X 10 0

> Input and output class

DIN-IO input

Function

Read the digital input value of the IO board and store it into an integer or Boolean variable.

Parameter description

Port value storage: Store the input value in the variable name and variable type of the target variable.

Input IO board: If there are multiple EtherCAT IOs, you can choose the corresponding IO board.

Input group number (Number of input channels): The input is read according to the group, which is a group of 1 channel, 4 channels and 8 channels. For a group of 1 channel, 16 DIN ports are 16 groups; for a group of 4 channels, 1-4, 5-8, 9-12, 13-16 as a group; for a group of 8 channels, 1-8, 9-16 as a group. The data read into the variable is to convert the input port value from binary to decimal and store it in the variable.

For example: a group of 8 channels, the value of port 1-8 is 10110101, then starting from port 8 is 10101101. Convert it to decimal as 173, then store it in variable as 173.

Usage examples

DIN I001 IN#(5)

DOUT-IO output

Function

Set the corresponding IO port on the IO board to high or low.

Parameter description

Output IO board: If there are multiple EtherCAT IOs, you can choose the corresponding IO board.

Output group number (Number of output channels): The output is output according to the group, which is a group of 1 channel, 4 channels and 8 channels. For a group of 1 channel, 16 DOUT ports are 16 groups; for a group of 4 channels, 1-4, 5-8, 9-12, 13-16 as a group; for a group of 8 channels, 1-8, 9-16 as a group.

Output value (Variable source): Divided into manual selection and variable type. Manual selection is to check the box below, the selected output is 1, and the unselected output is 0. Example: When the output group number is 4-way output, the second group, port 1 and port 3 are selected in the selection box below, and the other two are left blank, then when this command is run, the output ports of the IO board are numbered 5-8 The output value of the port is 1010. When INT, GINT, BOOL, GBOOL is selected as the variable source, the corresponding variable value will be converted into binary and output to the IO board.

Example: If the variable value is 173, it will be 10101101 when converted to binary. If 8 channels are a group, and the binary value is output from port 8, then the value of port 8-1 is 10101101, and the value of port 1-8 is 10110101.

Variable name: When the variable source is INT, GINT, BOOL, GBOOL, select the variable name to be output here.

Time: The time to reverse the output, the output is reversed after the specified time. For example, DOUT1=1, time is 2, then DOUT1 outputs high level for 2 seconds and then turns to low level. If the time is 0, it will output high level continuously.

Error stop processing: Output value hold, that is, when an error is reported, the io continues to output according to the parameters set by the instruction; Time-out stop, that is, stop at the end of the timing

Usage examples

DOUTOT#(1) I001 0

AIN-Analog input

Function

Read the input value of the corresponding analog input port into the target variable.

Parameter description

Analog input port: The analog input port to be read.

Variable value source: The variable type of the target variable.

Variable name: The variable name of the target variable.

Usage examples

AIN D001 B001

AOUT-Analog output

Function

Set the output value of the corresponding analog output port to the defined value.

Parameter description

Analog output port: The port to be output.

Variable value source: The variable type of the value to be output.

New parameter: When the variable value is customized, enter the hand-filled data here, the range is 0-10V, and the corresponding port will output the value.

Variable name: The variable name of the variable whose value is to be output.

Usage examples

AOUTAOUT1 1.1

PULSEOUT-Pulse output

Function

Output on pin 4 (PWM+) of the DB9 terminal on the R1 PWM IO board according to the set pulse frequency and number.

Parameter description

Number: Number of pulses.

Frequency: Pulse frequency.

Usage examples

PULSEOUT RATE = 100 SUM = 100

READ_DOUT-Read output

Function

Read the output status of the digital output port and store it in the target variable.

Parameter description

Output IO board: If there are multiple EtherCAT IOs, you can choose the corresponding IO board.

Variable type: The variable type of the target variable to be stored.

Variable name: The variable name of the target variable to be stored.

Output group number (Number of output channels): The value of the output port is read according to the group, which is a group of 1 channel, 4 channels and 8 channels. For a group of 1 channel, 16 DOUT ports are 16 groups; for a group of 4 channels, 1-4, 5-8, 9-12, 13-16 as a group; for a group of 8 channels, 1-8, 9-16 as a group.

For example: a group of 8 channels, the value of port 1-8 is 10110101, then starting from port 8 is 10101101. If it is converted to decimal, it is 173, then the variable is 173.

Usage examples

READ_DOUT I001 OT#(1)

> Timer class

TIMER-Delay

Function

Delay for the set value, and then continue to run.

Parameter description

Variable value source: You can manually fill in the value in the new parameter. You can also select the bind variable in "More" option, which will delay the length of time corresponding to the variable value.

Usage examples

TIMER T= 10

> Operation class

ADD-Add

Function

Addition operation (+), A=A+B.

Parameter description

Variable: The variable type of the summand A, you can fill in by hand or select the variable type in "More" option.

Variable value: The variable type of the addend B, you can fill in by hand or select the variable type in "More" option.

Usage examples

ADD GI001 22; Meaning: GI001=GI001+22

ADD GI002 I003; Meaning: GI002=GI002+I003

SUB-Subtract

Function

Subtraction operation (-), A = A - B.

Parameter description

Variable: The variable type of the minuend A, you can fill in by hand or select the variable type in "More" option.

Variable value: The variable type of the subtrahend B, you can fill in by hand or select the variable type in "More" option.

Usage examples

SUB GI001 22; Meaning: GI001=GI001-22

SUB GI002 I003; Meaning: GI002=GI002-I003

MUL-Multiply

Function

Multiplication (*), A=A*B.

Parameter description

Variable: The variable type of the multiplicand A, you can fill in by hand or select the variable type in "More" option.

Variable value: The variable type of the multiplier B, you can fill in by hand or select the variable type in "More" option.

New parameter: When the variable value source selects "Custom", this input box is valid, and the filled value is the value of B.

Source parameter: When the variable value source selects "Variable", here is the variable name of B.

Usage examples

MUL GI001 22; Meaning: GI001=GI001*22

MUL GI002 I003; Meaning: GI002=GI002*I003

DIV-Divide

Function

Division operation (÷), A = A÷B.

Parameter description

Variable: The variable type of the dividend A, you can fill in by hand or select the variable type in "More" option.

Variable value: The variable type of the divisor B, you can fill in by hand or select the variable type in "More" option.

Usage examples

DIV GI001 22; Meaning: GI001=GI001÷22

DIV GI002 I003; Meaning: GI002=GI002÷I003

MOD-Modulo

Function

Modulo operation (Mod), A=A Mod B.

Parameter description

Variable: The variable type of the dividend A, you can fill in by hand or select the variable type in "More" option.

Variable value: The variable type of the divisor B, you can fill in by hand or select the variable type in "More" option.

Usage examples

MOD GI001 22; Meaning: GI001=GI001 Mod 22

MOD GI002 I003; Meaning: GI002=GI002 Mod I003

SIN-Sine

Function

Sine operation (sin), A=sin(B), the unit of B is radians (rad).

Parameter description

Variable: The variable type of the result value A, you can fill in by hand or select the variable type in "More" option.

Variable value: The variable type of sine radian value B, you can fill in by hand or select the variable type in "More" option.

Usage examples

SIN GI001 22; Meaning: GI001=sin(22)

SIN GI002 I003; Meaning: GI002=sin(I003)

COS-Cosine

Function

Cosine operation (cos), A=cos(B), the unit of B is radians (rad).

Parameter description

Variable: The variable type of the result value A, you can fill in by hand or select the variable type in "More" option.

Variable value: The variable type of the cosine radian value B, you can fill in by hand or select the variable type in "More" option.

Usage examples

COS GI001 22; Meaning: GI001=cos(22)

COS GI002 I003; Meaning: GI002=cos(I003)

ATAN-Arctangent

Function

Arctangent operation (cos), A=arctan(B), the unit of B is radians (rad).

Parameter description

Variable: The variable type of the result value A, you can fill in by hand or select the variable type in "More" option.

Variable value: The variable type of the arctangent radian value B, you can fill in by hand or select the variable type in "More" option.

Usage examples

ATAN GI001 22; Meaning: GI001=arctan(22)

ATAN GI002 I003; Meaning: GI002=arctan(I003)

LOGICAL_OP-Logic operation

Function

Logic operation (and/or/not), B001 = I001andI002.

Parameter description

Parameter 1 type: The variable type of parameter 1 involved in the operation.

Parameter 1 name: The variable name of parameter 1 involved in the operation.

Operation type: Logical AND (&&), logical OR (||), logical NOT (!).

Parameter 2 type: The variable type of parameter 2 involved in the operation.

Parameter 2 name: The variable name of parameter 2 involved in the operation.

Result storage variable type: The type of variable where the result of the operation is stored.

Result storage variable name: The name of the variable where the result of the operation is stored.

Usage examples

LOGICAL_OPB001 = I001AND 10; Meaning: Variable I001, constant 10 logic and operation results are stored in B001

> Condition control class

Note: When conditional judgment needs to use strings for comparison, the actual comparison is the ASCII code value corresponding to the character

CALL-Call subprogram

Function

Call another program, after the called program has finished running, the program will return to the next line of the original program below the CALL instruction to continue running.

Parameter description

CALL: The name of the called program.

Usage examples

CALL [Program]; Meaning: Call the program Program

CALL_LUAFILE-Call LUA file

Function

Call the Lua file uploaded from upgrade.

Parameter description

CALL_LUAFILE: Call Lua file

Number of incoming parameters: The number of incoming parameters in the Lua file

Incoming parameter selection: Select the number and value of the required incoming parameters (the number should be the same as the actual Lua file)

Number of output parameters: The number of output parameters in the Lua file

Output parameter selection: Select the number and value of the required output parameters (the number can be less than the actual Lua file)

Usage examples

CALL_LUAFILE [$demo.lua$] IN (1.0,2.0,3.0,) OUT (2.0,2.0)

That is, call the Lua file demo.lua, pass three values into the demo, namely 1, 2, 3, and the demo will send out two values, both of which are 2.

IF-If

Function

If the condition of IF instruction is satisfied, the instruction between IF and ENDIF will be executed, if the condition of IF instruction is not satisfied, the program will jump to ENDIF instruction and continue to run the instruction below ENDIF without running the instruction between IF and ENDIF.

The judgment condition of IF is (comparand 1 comparison mode comparand 2), for example, comparand 1 is 2, comparand 2 is 1, comparison mode is ">", then 2>1, the judgment condition is established; if comparison mode is "<" or "==", the judgment condition is not established.

The IF instruction can be used alone or in combination with the ELSEIF and ELSE instructions. Note that the ELSEIF and ELSE instructions cannot be used independently of the IF instruction!

Note: When the beginning of the program is IF and the last line is ENDIF instruction, please insert a TIMER (delay, 0.1s) instruction above the IF instruction or below the ENDIF instruction, otherwise the program will crash when the conditions of the IF instruction are not met.

When inserting an IF instruction, an ENDIF instruction will be inserted at the same time. When deleting an IF instruction, please be careful to delete the corresponding ENDIF instruction, otherwise the program will not run.

IF instruction can nest another IF instruction or other conditional judgment instructions such as WHILE and JUMP.

Now the IF instruction supports multi-condition judgment and judges in order. If there are parentheses, first judge the ones inside the parentheses, and then judge the ones outside the parentheses. Up to 5 judgment conditions are supported.

Parameter description

Parameter type: The type of comparand 1: variable or digital/analog input value.

Parameter name:

If the selected parameter type is "Variable" (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), then this is the variable name of comparand 1.

If the selected parameter type is input value (DIN, AIN), then here is the port number of the digital input or analog input

Comparison mode:

== equal to

< less than

> greater than

<= less than or equal to

>= greater than or equal to

!= not equal to

Variable value source: The type of comparand 2: custom, variable or digital/analog input value.

New parameter:

If the variable value source is "Custom", then this is not selectable.

If the variable value source is "Variable" (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), then this is the variable name of the comparand 1.

If the variable value source is input value (DIN, AIN), then here is the port number of the digital input or analog input

Source parameter: If the variable value source is "Custom", fill in the value of the comparand 2 directly here.

Usage examples

IF(GI001>=D001)

Other instructions, such as MOVJ, etc.

ENDIF

ELSEIF-Else if

Function

The ELSEIF instruction must be inserted between IF and ENDIF. An ELSE instruction or multiple ELSEIF instructions can also be inserted between ELSEIF and ENDIF.

When the condition of IF is satisfied, the ELSEIF instruction and the instructions between ELSEIF and ENDIF will be ignored, only the instructions between IF and ELSEIF will be run, and then jump to the line of instruction below ENDIF to continue running.

When the condition of IF is not satisfied, the program will jump to the ELSEIF instruction to judge the judgment condition of ELSEIF. If it is met, run the instructions between ELSEIF and ENDIF, and then continue to run the instruction below ENDIF; if it is not met, jump directly to the line of instruction below ENDIF to continue running.

If multiple ELSEIFs are nested in IF and ENDIF, when the judgment condition of IF is not established, the judgment condition of the first ELSEIF is first judged, if it is established, the instruction between the first ELSEIF and the second ELSEIF is executed; if it is not established, then judge the judgment condition of the second ELSEIF, and so on.

Note: When deleting the IF instruction, the corresponding ELSE and ENDIF instructions must be deleted, otherwise the program will not run.

Now the ELSEIF instruction supports multi-condition judgment and judges in order. If there are parentheses, first judge the ones inside the parentheses, and then judge the ones outside the parentheses. Up to 5 judgment conditions are supported.

Parameter description

Parameter type: The type of comparand 1: variable or digital/analog input value.

Parameter name:

If the selected parameter type is "Variable" (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), then this is the variable name of comparand 1.

If the selected parameter type is input value (DIN, AIN), then here is the port number of the digital input or analog input

Comparison mode:

== equal to

< less than

> greater than

<= less than or equal to

>= greater than or equal to

!= not equal to

Variable value source: The type of comparand 2: custom, variable or digital/analog input value.

New parameter:

If the variable value source is "Custom", then this is not selectable.

If the variable value source is "Variable" (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), then this is the variable name of the comparand 1.

If the variable value source is input value (DIN, AIN), then here is the port number of the digital input or analog input

Source parameter: If the variable value source is "Custom", fill in the value of the comparand 2 directly here.

Usage examples

IF(GI001>=D001)

Other instruction 1, such as MOVJ, etc.

ELSEIF(D001<9)

Other instruction 2, such as MOVJ, etc.

ENDIF

ELSE-Else

Function

The ELSE instruction must be inserted between IF and ENDIF, but only one ELSE instruction can be embedded in an IF instruction.

When the judgment condition of IF is established, the instruction between IF and ELSE will be executed and then jump to the next line of instruction of ENDIF to continue running, without running the instruction between ELSE and ENDIF.

When the judgment condition of IF is not established, the program will jump to the instruction between ELSE and ENDIF to continue running, without running the instruction between IF and ELSE.

Note: When deleting the IF instruction, the corresponding ELSE and ENDIF instructions must be deleted, otherwise the program will not run.

Parameter description

None

Usage examples

IF(GI001<9)

Other instruction 1, such as MOVJ, etc.

ELSE

Other instruction 2, such as MOVJ, etc.

ENDIF

WAIT-Wait

Function

WAIT means wait, you can choose whether to have a wait time or not. If the "TIME" option is not checked, the program will stay in the WAIT instruction and wait until the judgment condition is met. If the "TIME" option is checked, the program will wait for the length of the parameter and then continue to run the next instruction. If the condition becomes true while waiting, the next instruction will be run immediately.

Now the WAIT instruction supports multi-condition judgment and judges in order. If there are parentheses, first judge the ones inside the parentheses, and then judge the ones outside the parentheses. Up to 5 judgment conditions are supported.

Parameter description

Parameter type: The type of comparand 1: variable or digital/analog input value.

Parameter name:

If the selected parameter type is "Variable" (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), then this is the variable name of comparand 1.

If the selected parameter type is input value (DIN, AIN), then here is the port number of the digital input or analog input

Comparison mode:

== equal to

< less than

> greater than

<= less than or equal to

>= greater than or equal to

!= not equal to

Variable value source: The type of comparand 2: custom, variable or digital/analog input value.

New parameter:

If the variable value source is "Custom", then this is not selectable.

If the variable value source is "Variable" (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), then this is the variable name of the comparand 1.

If the variable value source is input value (DIN, AIN), then here is the port number of the digital input or analog input

Source parameter: If the variable value source is "Custom", fill in the value of the comparand 2 directly here.

TIME:

Optional, if you do not select this option, you will wait forever until the condition is met.

If selected, you can fill in the waiting time (seconds), and after that time, even if the condition is still not met, the program will still jump to the next line and continue to run.

Is PL continuous: refers to whether the smoothness of the trajectory curve of the robot will be interrupted

Continuous: The robot running curve is relatively smooth after the conditions are met

Not continuous: The smoothness of the robot's trajectory is interrupted after the conditions are met

Filter time:

Optional, no effect if not selected.

If selected, it refers to the required input signal duration. When the input signal duration meets the filter time (without waiting for TIME), jump to the next line to continue running. If the filter time is not met, wait for the TIME time and then jump to the next line to continue running.

Usage examples

WAIT (GI001==2) T = 2 F = 1

WHILE-Loop

Function

When the condition of WHILE instruction is satisfied, the instruction between WHILE and ENDWHILE will be run cyclically. If the judgment condition is not satisfied before running to WHILE instruction, the program will jump to ENDWHILE instruction directly when running to the WHILE instruction without running the instruction between WHILE and ENDWHILE; if the judgment condition becomes not satisfied during running the instruction between WHILE and ENDWHILE, the program will continue to run until it reaches ENDWHILE line, it will not loop but continue to run the instruction below ENDWHILE.

The judgment condition of WHILE is (comparand 1 comparison mode comparand 2), for example, comparand 1 is 2, comparand 2 is 1, comparison mode is ">", then 2>1, the judgment condition is established; if comparison mode is "<" or "==", the judgment condition is not established.

Note: When inserting an WHILE instruction, an ENDWHILE instruction will be inserted at the same time. When deleting an WHILE instruction, please be careful to delete the corresponding ENDWHILE instruction, otherwise the program will not run.

When the beginning of the program is WHILE and the last line is ENDWHILE instruction, please insert a TIMER (delay, 0.3s) instruction at the beginning or end of the program, otherwise the program will crash when the conditions of the WHILE instruction are not met.

The WHILE instruction can be nested with multiple WHILE, IF or JUMP and other judgment class instructions.

Now the WHILE instruction supports multi-condition judgment and judges in order. If there are parentheses, first judge the ones inside the parentheses, and then judge the ones outside the parentheses. Up to 5 judgment conditions are supported.

Parameter description

Parameter type: The type of comparand 1: variable or digital/analog input value.

Parameter name:

If the selected parameter type is "Variable" (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), then this is the variable name of comparand 1.

If the selected parameter type is input value (DIN, AIN), then here is the port number of the digital input or analog input

Comparison mode:

== equal to

< less than

> greater than

<= less than or equal to

>= greater than or equal to

!= not equal to

Variable value source: The type of comparand 2: custom, variable or digital/analog input value.

New parameter:

If the variable value source is "Custom", then this is not selectable.

If the variable value source is "Variable" (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), then this is the variable name of the comparand 1.

If the variable value source is input value (DIN, AIN), then here is the port number of the digital input or analog input

Source parameter: If the variable value source is "Custom", fill in the value of the comparand 2 directly here.

Usage examples

WHILE(GI001<2)

Other instruction 1, such as MOVJ, etc.

WHILE(D001<10)

Other instruction 2, such as MOVJ, etc.

ADD D001 1

ENDWHILE

Other instruction 3

ADD GI001 1

ENDWHILE

LABEL-Label

Function

The target label of JUMP instruction jump.

Parameter description

Label name: The name of the label, which needs to be a string starting with a letter.

Usage examples

LABEL *A1

JUMP-Jump

Function

JUMP is used to jump and must be used in conjunction with the LABEL (label) instruction.

JUMP can be set with or without judgment conditions. When the JUMP instruction is set to have no judgment condition, the program will jump directly to the corresponding LABEL instruction when running to the instruction and continue to run the next instruction below LABEL.

When the JUMP instruction is set to have a judgment condition, if the condition is satisfied, jump to the LABEL instruction line; if the condition is not satisfied, ignore the JUMP instruction and continue to run the next instruction below the JUMP instruction.

LABEL label can be inserted above or below JUMP, but cannot jump across programs.

LABEL label name must consist of two or more characters starting with a letter.

Inserting the LABEL label has no effect on the operation of the program, but it must comply with the rules of program operation, for example, it cannot be inserted above the MOVC instruction or the local variable definition instruction.

Now the JUMP instruction supports multi-condition judgment and judges in order. If there are parentheses, first judge the ones inside the parentheses, and then judge the ones outside the parentheses. Up to 5 judgment conditions are supported.

Parameter description

Label name: The label name of the inserted LABEL instruction, optional.

Judgment condition:

Optional, if selected, the judgment condition can be set. If not selected, the program will jump directly after running to JUMP.

Parameter type: The type of comparand 1: variable or digital/analog input value.

Parameter name:

If the selected parameter type is "Variable" (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), then this is the variable name of comparand 1.

If the selected parameter type is input value (DIN, AIN), then here is the port number of the digital input or analog input

Comparison mode:

== equal to

< less than

> greater than

<= less than or equal to

>= greater than or equal to

!= not equal to

Variable value source: The type of comparand 2: custom, variable or digital/analog input value.

New parameter:

If the variable value source is "Custom", then this is not selectable.

If the variable value source is "Variable" (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), then this is the variable name of the comparand 1.

If the variable value source is input value (DIN, AIN), then here is the port number of the digital input or analog input

Source parameter: If the variable value source is "Custom", fill in the value of the comparand 2 directly here.

Usage examples

MOVJ

LABEL *C1

Other instruction 1, such as MOVJ, etc.

JUMP *C1 WHEN (I001==0)

Other instruction 2

UNTIL-Until

Function

The UNTIL instruction is used to jump out of a motion process. This means that the program pauses during one motion and starts the next one. When the condition is met, the program immediately pauses and starts an instruction below the ENDUNTIL instruction, regardless of whether the robot is currently running or not.

The judgment condition of UNTIL is (comparand 1 comparison mode comparand 2), for example, comparand 1 is 2, comparand 2 is 1, comparison mode is ">", then 2>1, the judgment condition is established; if comparison mode is "<" or "==", the judgment condition is not established.

Note: When inserting an UNTIL instruction, an ENDUNTIL instruction will be inserted at the same time. When deleting an UNTIL instruction, please be careful to delete the corresponding ENDUNTIL instruction, otherwise the program will not run.

Now the UNTIL instruction supports multi-condition judgment and judges in order. If there are parentheses, first judge the ones inside the parentheses, and then judge the ones outside the parentheses. Up to 5 judgment conditions are supported.

Parameter description

Parameter type: The type of comparand 1: variable or digital/analog input value.

Parameter name:

If the selected parameter type is "Variable" (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), then this is the variable name of comparand 1.

If the selected parameter type is input value (DIN, AIN), then here is the port number of the digital input or analog input

Comparison mode:

== equal to

< less than

> greater than

<= less than or equal to

>= greater than or equal to

!= not equal to

Variable value source: The type of comparand 2: custom, variable or digital/analog input value.

New parameter:

If the variable value source is "Custom", then this is not selectable.

If the variable value source is "Variable" (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), then this is the variable name of the comparand 1.

If the variable value source is input value (DIN, AIN), then here is the port number of the digital input or analog input

Source parameter: If the variable value source is "Custom", fill in the value of the comparand 2 directly here.

Usage examples

UNTIL(GI001<2)

Other instructions

ENDUNTIL

MOVJ P0003

CRAFTLINE-Process jump

Function

Special process instruction, after running this instruction in the program, the program will jump to the corresponding line in the special process interface.

Parameter description

New parameter: The corresponding line number in the special process interface.

Usage examples

CRAFTLINE 22

CMDNOTE-Instruction note

Function

Instruction note, you can use this instruction to add note in the appropriate position of the program for debugging.

If a note instruction is inserted, when single-stepping this instruction, the program will jump to the next line of instruction to run, and there will be no error prompt.

Parameter description

Note content: The note content supports Chinese and English, case, number input and symbol input.

Usage examples

##iINEXBOT$$; Meaning: The content of the note is "INEXBOT".

POS_REACHABLE-Judgment of reachability

Function

Reachability judgment instruction, used to judge whether the target point can be reached, if the point can be reached, set the variable to 1, if not, set it to 0.

Parameter description

Position variable name: P point, G point.

Motion type: MOVJ, MOVL.

Status storage variable type: BOOL, GBOOL.

Status storage variable name: BOOL,GBOOL variable name.

Usage examples

POS_REACHABLE MOVJ P001 B001; Meaning: Calculate if it is possible to run to position P001 using MOVJ interpolation, if it is possible to reach, the value of B001 is 1, if not, the value of B001 is 0.

CLKSTART-Timing start

Function

The CLKSTART instruction is used for timing. Run this instruction to start the timing and record the time in a local or global DOUBLE variable.

Accuracy of timing instruction is two decimal places (i.e. 10ms, error ±2ms)

Parameter description

Serial number: The serial number of the timer, you can use 32 timers to time separately at the same time.

Storage variable type: Store the timed time into a local DOUBLE variable or a global GDOUBLE variable.

Storage variable name: The variable name of the variable that stores the time.

Usage examples

CLKSTARTID= 1 D001; Meaning: Process number 1 starts timing, and the timing result is stored in D001.

CLKSTOP-Timing stop

Function

The CLKSTOP instruction is used to stop the timing of the timer of the corresponding serial number. The value stored in the variable will not reset to zero after stopping.

Parameter description

Serial number: The serial number of the timer to stop timing.

Usage examples

CLKSTOPID=1; Meaning: Process number 1 stops timing

CLKRESET-Timing reset

Function

The CLKRESET instruction is used to reset the timer to zero for the corresponding serial number. If this instruction is not used, the next time the CLKSTART instruction is run, the timing will be accumulated.

Parameter description

Serial number: The serial number of the timer to be reset to zero.

Usage examples

CLKPESETID=1; Meaning: Reset the timing result of process number 1.

READLINEAR-Read linear speed

Function

Read the linear speed of the robot into the variable in real time

Parameter description

Variable type: The type of the storage variable, optional GINT/GDOUBLE.

Variable name: The name of the storage variable.

Usage examples

READLINEAR GDOO1

CALL_LUASTRING-Call LUA statement

Function

Realize corresponding function or operation by calling Lua statement

Parameter description

Statement: the Lua statement to be entered

More: hand-filled and variables;

Hand-filled: enter the corresponding and correct Lua statement by yourself, and you can directly step or run

Variables: Write Lua statements into string variables, and realize their functions by calling the corresponding string variables

Usage examples

CALL_LUASTRING [statement]

or CALL_LUASTRING string variable

> Variable class

SET-Assignment

(Define the variable and go directly to the local variable interface to cancel the instruction)

Function

Define local integer, float, and Boolean variables, and assign values at the same time.

Parameter description

Variable: Click "More" to select the desired variable type.

Variable value source: Assign value to the above variable, you can fill in by hand or select the variable type in "More" option.

Usage examples

INTI001 = 11

INT I002 = GI003

FORCESET-Write to file

Function

During the running of the program, all calculations and assignment operations are to change the values in the cache, and the values will not be stored in the system file, that is, the values of all global variables will be restored when the program stops running.

To force the global value variables in the content to be written to the file, you can use the FORCESET instruction.

Parameter description

Variable name: Click "More" to select the variable name you want to force writing to the file.

Usage examples

FORCESETGI001

> String class

STRING-SPELL-String append

Function

Add the characters you need to the variable of the original string or the variable of the empty string to form a new string variable

Parameter description

Variable: Variable type and name

Variable value: Constant or bind other variable

Usage examples

STRING_SPELL [S001 + S002]

STRING-SLICE-String index interception

Function

Intercept a part of a string in the string variable and store it in the specified variable

Parameter description

Variable: The variable name of the extracted string.

Start index: The start position of the index.

End index: The end position of the index.

Variable value: The location where the intercepted data is stored.

Usage examples

STRING_SLICE   S001 (I001, I001)   S001   I001

STRING-SPLIT-String separator splitting

Function

Split the string in the variable with one of the characters in the string variable, and store the splitted characters in the specified variable in order

Parameter description

Variable: The parameter where the search string is located.

Separator: The type of separator.

The first variable in which the data is stored: The first position where the queried data are stored in sequence.

Data storage number: Record the amount of extracted data.

Usage examples

STRING_SPLIT   S001 (I001, I001)   S001   I001

STRING-LOCATE-String positioning query

Function

Query the position of a character in a string variable, and store the position and quantity in the specified variable in turn

Parameter description

Variable: The parameter where the search string is located.

Variable to be indexed: The character to be searched.

The first variable in which the data is stored: The first position where the queried data are stored in sequence.

Data storage number: Record the amount of extracted data.

Usage examples

STRING_LOCATE S001 S002 I001 0

STRING-LENGTH-String length

Function

Calculate the length of a string in a string variable, and stores the calculated length in the specified variable.

Parameter description

Variable: The variable whose length is to be calculated.

Data storage variable: Record the amount of extracted data.

Usage examples

STRING_LENGTH S001 I001

STRING-TO-String to non-string

Function

Convert a string in a string variable to a non-string

Parameter description

String variable: The string to be translated.

Non-string variable: The target variable of translation.

Usage examples

STRING_TO S001 I001

TO-STRING-Non-string to string

Function

Convert a variable in a non-string variable to a string

Parameter description

Non-string variable: The variable to be translated.

String variable: The target variable of the translation.

Usage examples

TO_STRING I001 S001

> Coordinate switch class

SWITCHTOOL-Switch tool hand

Function

Switch the currently used tool hand coordinate system during program running.

Parameter description

Tool coordinate: The tool number of the tool hand coordinate system to switch to.

Usage examples

SWITCHTOOL (3)

SWITCHUSER-Switch user coordinate

Function

Switch the currently used user coordinate system during program running.

Parameter description

User coordinate: The serial number of the user coordinate system to switch to.

Usage examples

SWITCHUSER (3)

USERCOORD_TRANS-User coordinate transformation

Function

Superimpose the B and C user coordinate systems (×), and place the result in the A user coordinate system.

Parameter description

User coordinate A: The result is stored in this user coordinate system, here is the user coordinate system serial number.

User coordinate B: User coordinate system serial number.

User coordinate C: User coordinate system serial number.

Usage examples

USERCOORD_TRANS (1) (2) (3)

SWITCHSYNC-Switch external axis

Function

Switch the currently used external axis during program running.

Parameter description

External axis group number: The group number of the external axis to switch to.

Usage examples

SWITCHSYNC 1

Network communication class

SENDMSG-Send data

Function

Send a string message to another network device.

Parameter description

ID: Process number in the "Settings-Network Settings" interface.

Characters sent: The string to be sent. If you want to send variables, add $ before the variable. If you want to send $ character, two $ are needed. Escape characters and formatted output are supported.

Usage examples

SENDMSG ID = 1 #$D001#

PARSEMSG-Parse data

Function

Parse the data sent by another network device over TCP and store the data in multiple variables.

When a TCP receives multiple values, it will store the values into multiple variables respectively, and the variables used are the first variable, the second variable and so on. That is, if 3 values are sent, i.e. A, B, C, and the first variable set is named GI006, then A will be stored in GI006, B in GI007, and C in GI008.

Parameter description

ID: Process number in the "Settings-Network settings" interface.

The first variable in which the data is stored (First variable type): The type of the first storage variable. Click "More" to select variable type

Clear cache after parsing: Clear the cached data after parsing it

Data storage quantity: Record the number of extracted data by variable

First variable name: The name of the first storage variable.

Usage examples

PARSEMSG ID = 1 GI006CLEARCAHE= 0; Meaning: Store the received data in the variable GI001, and clear the cached data after parsing is completed

READCOMM-Read data

Function

Read the points sent by Ethernet or Modbus, store the points in the position variable, and store the number in the numeric variable.

Parameter description

Process number: The process number of the network communication to open the communication.

Communication method: Use Ethernet communication or Modbus communication.

Position variable type: Global position variable and local position variable can be selected.

Position variable name: The name of the position variable; store the received points, if there are multiple points, the position variables will be extended. For example, the instruction position variable is filled with GP003, and when three points are received, they are stored in GP003, GP004, and GP005 respectively.

Variable type: Global integer and local integer can be selected.

Variable name: Name of variable; store the number of received points.

Note: Currently available only for Modbus.

Usage examples

READCOOMID=1 EHTERNETTOG001I001

OPENMSG-Open data

Function

Open the network communication.

Parameter description

ID: Process number in the "Settings-Network settings" interface.

Usage examples

OPENMSGID= 1

CLOSEMSG-Close data

Function

Close the network communication.

Parameter description

ID: Process number in the "Settings-Network settings" interface.

Usage examples

CLOSEMSG ID = 2

PRINTMSG-Output information

Function

Print the string by means of prompt message.

Parameter description

Output character: The string to be printed. If you want to print variables, add $ before the variable. If you want to print $ character, two $ are needed. Escape characters and formatted output are supported.

Usage examples

PRINTMSG #this is $D001#

MSG_CONNECTION_STATUS-Get information connection status

Function

Get the connection status of a process number in the network settings

Parameter description

Process number: The process number of the network setting that needs to be known

Status storage variable name: Click "More" to select BOOL/GBOOL storage type

Usage examples

MSG_CONN_ST 1 B001

Position variable class

Note: For the newly added variables of the position variable type in the following instructions, please refer to the description of the bind variables in the motion control class section.

USERFRAME_SET-User coordinate modification

Function

Change the value of an axis of the user coordinate system.

Parameter description

User coordinate number: The user coordinate number whose value is to be changed.

User coordinate parameter: The user coordinate axis whose value is to be changed.

Variable type: You can choose hand-filled value or other variable.

Variable name: When selecting other variable, selecting the variable name here will assign the value of that variable to the axis corresponding to the user coordinate.

Hand-filled value: When the variable type selects hand-filled value, directly fill in the target value to be changed here.

Usage examples

USERFRAME_SET ID = 1 UX GI001

USERFRAME_SET ID = 2 UY 99

TOOLFRAME_SET-Tool coordinate modification

Function

hange the value of an axis of the tool coordinate system.

Parameter description

Tool coordinate number: The tool coordinate number whose value is to be changed.

Tool coordinate parameter: The tool coordinate axis whose value is to be changed.

Variable type: You can choose hand-filled value or other variable.

Variable name: When selecting other variable, selecting the variable name here will assign the value of that variable to the axis corresponding to the tool coordinate.

Hand-filled value: When the variable type selects hand-filled value, directly fill in the target value to be changed here.

Usage examples

TOOLFRAME_SET ID = 1 TX GI001; Meaning: Change the X-axis offset parameter of tool hand 1 to the variable value of GI001

TOOLFRAME_SET ID = 2 TY 99; Meaning: Change the X-axis offset parameter of tool hand 2 to 99

READPOS-Read points

Function

Read the value of an axis of a position variable into a float variable.

Parameter description

Variable type: The type of float variable to read, local or global.

Variable name: The variable name of the float variable to read.

Position variable type: The type of position variable to read, current position, local position variable or global position variable.

Position variable name: When the position variable type selects local position variable or global position variable, select the corresponding variable name here. If you choose PINT, PGINT, G$INT or G$GINT, select the corresponding integer variable name here. For example, if you select P$INT (variable name I001, I001=33), then the obtained position variable is P033.

Position variable coordinate system: The coordinate system where the position variable value to be read is located.

Position variable axis: The axis of the position value to be read in the corresponding coordinate system.

Usage examples

READPOS GD004 P$GI003 RF 1

POSADD-Point add

Function

Position variable addition operation (+). This instruction can add the value of a single axis of the position variable (global, local), and then assign it to the axis.

Parameter description

Position variable type: The type of the position variable to be changed, local or global.

Position variable name: The variable name of the position variable to be changed.

Position variable coordinate system: The corresponding coordinate system where the position variable axis to be changed.

Position variable axis: The axis of the position variable to be changed in the corresponding coordinate system.

Variable type: You can choose hand-filled value or other variable.

Numeric variable name: When selecting other variable, selecting the variable name here will add the value of that variable to the value of the corresponding axis of the position variable, and then assign the value to that position variable.

Hand-filled value: When variable type selects hand-filled value, filling in the target value directly here will add that value to the value of the corresponding axis of the position variable and assign it to that position variable.

Usage examples

POSADD P0001 RF 1 788

POSSUB-Point subtract

Function

Position variable subtraction operation (-). This instruction can subtract the value of a single axis of a position variable (global, local), and then assign it to the axis.

Parameter description

Position variable type: The type of the position variable to be changed, local or global.

Position variable name: The variable name of the position variable to be changed.

Position variable coordinate system: The corresponding coordinate system where the position variable axis to be changed.

Position variable axis: The axis of the position variable to be changed in the corresponding coordinate system.

Variable type: You can choose hand-filled value or other variable.

Numeric variable name: When selecting other variable, selecting the variable name here will subtract the value of the variable from the value of the corresponding axis of the position variable, and then assign the value to that position variable.

Hand-filled value: When variable type selects hand-filled value, filling in the target value directly here will subtract that value from the value of the corresponding axis of the position variable and assign it to that position variable.

Usage examples

POSSUB P0001 RF 1 88

POSSET-Point change

Function

This instruction can modify the value of a single axis of a position variable (global, local).

Parameter description

Position variable type: The type of the position variable to be changed, local or global.

Position variable name: The variable name of the position variable to be changed.

Position variable coordinate system: The corresponding coordinate system where the position variable axis to be changed.

Position variable axis: The axis of the position variable to be changed in the corresponding coordinate system.

Variable type: You can choose hand-filled value or other variable.

Numeric variable name: When selecting other variable, selecting the variable name here will assign the value of the corresponding axis of the position variable to that position variable.

Hand-filled value: When variable type selects hand-filled value, filling in the target value directly here will assign the value of the axis corresponding to the position variable to the position variable.

Usage examples

POSSUB P0001 RF 1 88

COPYPOS-Copy point

Function

Copy the values of all axes of a position variable to another position variable.

Parameter description

Source position variable type: The type of the position variable to be read. You can select the current position, i.e. assign the current robot position to another position variable.

Source position variable name: The variable name of the position variable to be read.

Target position variable type: The variable type of the position variable being assigned.

Target position variable name: The variable name of the position variable being assigned.

Usage examples

COPYPOSG003 TOP001

COPYPOSCURPOSTOP002

POSADDALL-Point add all

Function

Position variable addition operation (+). This instruction can perform addition operation on the value of several axes of position variable (global, local), and then assign it to the axis.

Parameter description

Position variable type: The type of the position variable to be changed, local or global.

Position variable name: The variable name of the position variable to be changed.

Position variable coordinate system: The corresponding coordinate system where the position variable axis to be changed.

More: You can choose hand-filled value or other variable.

Numeric variable name: When selecting other variable, selecting the variable name here will add the value of the corresponding axis of the position variable to the value of that variable, and then assign it to the position variable.

Hand-filled value: When variable type selects hand-filled value, filling in the target value directly here will add that value to the value of the corresponding axis of the position variable and assign it to that position variable.

Usage examples

POSADDALL GP0001 RF I001 GI001 D001 GD001 10.1 10

POSSUBALL-Point subtract all

Function

Position variable subtraction operation (-). This instruction can perform subtraction operation on the value of several axes of position variable (global, local), and then assign it to the axis.

Parameter description

Position variable type: The type of the position variable to be changed, local or global.

Position variable name: The variable name of the position variable to be changed.

Position variable coordinate system: The corresponding coordinate system where the position variable axis to be changed.

More: You can choose hand-filled value or other variable.

Numeric variable name: When selecting other variable, selecting the variable name here will subtract the value of the variable from the value of the corresponding axis of the position variable, and then assign the value to that position variable.

Hand-filled value: When variable type selects hand-filled value, filling in the target value directly here will subtract that value from the value of the corresponding axis of the position variable and assign it to that position variable.

Usage examples

POSSUBALL GP0001 RF I001 GI001 D001 GD001 10.1 10

POSSETALL-Point change all

Function

This instruction can modify the values of several axes of position variable (global, local).

Parameter description

Position variable type: The type of the position variable to be changed, local or global.

Position variable name: The variable name of the position variable to be changed.

Position variable coordinate system: The corresponding coordinate system where the position variable axis to be changed.

More: You can choose hand-filled value or other variable.

Numeric variable name: When selecting other variable, selecting the variable name here will assign the value of the corresponding axis of the position variable to that position variable.

Hand-filled value: When variable type selects hand-filled value, filling in the target value directly here will assign the value of the axis corresponding to the position variable to the position variable.

Usage examples

POSSETALL GP0001 RF I001 GI001 D001 GD001 10.1 10

TOFFSETON-Trajectory offset start

Function

This instruction provides a real-time offset of the robot's trajectory.

Parameter description

Offset coordinate system: The coordinate system corresponding to the trajectory to be changed.

Offset type: You can choose hand-filled value or other variable type.

Offset: When the variable type is hand-filled value, filling in the target value directly here will add the robot's trajectory coordinates to this hand-filled value.

More: You can choose hand-filled value or other variable.

Numeric variable name: When selecting other variable, selecting the variable name here will assign the value of the corresponding axis of the position variable to that position variable.

Hand-filled value: When variable type selects hand-filled value, filling in the target value directly here will assign the value of the axis corresponding to the position variable to the position variable.

Usage examples

TOFFSETON RF GI001 I002 2 3 4 5

TOFFSETOFF-Trajectory offset end

Function

The trajectory offset ends, and the subsequent motion trajectories will no longer offset.

Usage examples

TOFFSETOFF

READPOSMSG-Read point information

Function

Read the value of point tool number, user coordinate number, coordinate system, attitude angle/radian and form information into an integer variable.

Parameter description

Variable type: You can choose between global position variable and local position variable.

Variable name: The name of the position variable.

Information: Tool number/user coordinate number/coordinate system/angle/radian/form.

More (Target variable type): The variable type of the position variable being read.

Target variable name: The variable name of the position variable being read.

Usage examples

READPOSMSG P0001 TOOL I001

POS_STRETCH-Point stretch

Function

Shorten or lengthen the length of the line and the ends of the arc, and change the middle point of the arc to change the trajectory of the arc.

Parameter description

Stretch type: Support line or arc instruction stretch.

Start point: The start point of a line or arc instruction.

Arc midpoint: The midpoint of the arc instruction.

End point: The end point of a line or arc instruction.

Start point offset: The distance by which the start point is shortened or stretched.

End point offset: The distance by which the end point is shortened or stretched.

Output start point position: Save the stretched start point position in the local point position or global point position.

Output end point position: Save the stretched end point position in the local point position or global point position.

Usage examples

POS_STRETCH LINE P0001 P0002 10 10 P0004 P0005

SETPOSMSG-Set point information

Function

Set the coordinate system, angle/radius, form, tool number, and user coordinate number of the point

Parameter description

Variable type: You can choose between global position variable and local position variable.

Coordinate system: Set the coordinate system number by local integer variable, global integer variable and "unchanged" option.

Angle/radian: Set angle/radian by local integer variable, global integer variable and "unchanged" option.

Form: Set the form by local integer variable, global integer variable and "unchanged" option.

Tool number: Set the tool number by local integer variable, global integer variable and "unchanged" option.

User coordinate number: Set user coordinate number by local integer variable, global integer variable or "unchanged" option.

Usage examples

SETPOSMSG P0001 1 1 1 1 1 1

> Program control class

PTHREAD_START-Start thread

Function

Start the background task. The background task ends when executed once. To edit the background task, please go to "Settings - Background Tasks" interface for programming.

Local background tasks will synchronize the stop and run of the main program, global background tasks will not do this.

Parameter description

Type: Select local background or global background

Background task: The name of the background task

Usage examples

PTHREAD_START [TTT]

PTHREAD_END-Exit thread

Function

Close the started background tasks.

Parameter description

Type: Select local background or global background

Background task: The name of the background task

Usage examples

PTHREAD_END [TTT]

PAUSERUN-Pause running

Function

Pause the program.

Parameter description

Type: The type of program to be paused, including all, main program, background program.

Program: Name of the program to be paused.

Usage examples

PAUSERUN [TTT]

PAUSERUN MAIN

PAUSERUN ALL

CONTINUERUN-Continue running

Function

Continue running the paused program (the stopped program cannot be continued).

Parameter description

Type: The type of program to continue running, including main program, local background program.

Program: The name of the program to continue running.

Usage examples

CONTINUERUN [TTT]

CONTINUERUN MAIN

STOPRUN-Stop running

Function

Stop all programs.

Parameter description

None

Usage examples

STOPRUN

RESTARTRUN-Rerun

Function

Rerun the stopped program.

Parameter description

None

Usage examples

RESTARTRUN

WINDOW-Popup instruction

Function

Pop up the filled content prompt window, the number of displayed buttons is the number of options, and save the value of the clicked button (option) into a local integer variable.

Parameter description

Prompt: The content displayed in the pop-up window.

Bind variable: Local integer variables

Number of options: 1-3 buttons

Option 1 content: Button 1 content

Option 1 value: Button 1 value

Option 2 content: Button 2 content

Option 2 value: Button 2 value

Option 3 content: Button 3 content

Option 3 value: Button 3 value

Usage examples

WINDOW #Prompt# I001 3 #Button 1# 1 #Button 2# 2 #Button 3# 3

PTHREAD_STATE-Thread state

Function

Insert thread instruction to view the status of the currently executed thread program, stop equals 1, pause equals 2, run equals 3

Parameter description

Type: You can choose local background, global background or main program

Background task: The name of the background task

Storage variable type: For example, the selected variable name is GI001, after opening the thread, the value of GI001 will change with the state. Stop: GI001=1; Pause: GI001=2; Run: GI001=3

Usage examples

PTHREAD_STATE[program file name] GINT GI001=0

LIGENT TECH CO., LTD

Website

YouTube

Official Website: http://ligentrobot.com
E-Mail: info@sotrobot.com
whatsapp:+86-13551010933