艾创科技
AITRON TECH

Operation Software Manual

Version：

V1.0

Please ensure that this manual reaches the end user of the product

Catalogue

1. Safety regulations

1.1. General rule

The main basis of this safety regulation is: combined with the actual work experience and results, under the premise of referring to the relevant robot standards or methods, to make it meet the specific requirements of the company's robot operation and other operations.The detailed requirements and specific implementation measures are put forward for the safety content of the operation or maintenance of the robot and its system:

1. Standardize the safety content of the robot and its system from installation and debugging, type test, field operation and later maintenance;
2. Specify specific implementation methods or preventive measures where safety problems may occur;
3. Provide safety guarantee for robot operation.

1.2. Regulatory requirement

The technical performance of the robot system and the preset installation position are described in detail in the corresponding instructions for the use of the robot and control cabinet.

Improper operation or non-use of robotic systems as prescribed may result in:

1. pose a threat to human body and life;
2. Cause harm and damage to the robot system and other property of the user;
3. The work efficiency of the robot system and operator is affected

1.3. Responsibility statement

Industrial robots comply with the current state of the art and current safety regulations; however, improper use may result in personal injury, robot systems and other damage.Industrial robots are only allowed to be used in accordance with regulations and safety awareness in the intact state of the robot, and if there is a safety hazard fault, it must be eliminated in time.

Yantai Aichuang Robot Technology Co., Ltd. is committed to providing reliable safety information, but does not assume responsibility for this.Even if everything is done in accordance with the safe operating instructions, there is no guarantee that industrial robots will not cause damage to people and property.

Industrial robots shall not be altered without the consent of Eltron.Additional parts (tools, software, etc.) that do not belong to our company may also be used in industrial robots, and if these parts cause damage to the robot, the responsibility is borne by the operator.

1.4. Safety precautions

It is prohibited to use the robot in the following situations, otherwise it will not only cause adverse effects on the robot and peripheral equipment, but also may cause injury to the user.

1. In an flammable environment
2. In an explosive environment
3. In the presence of a large amount of radiation environment
4. In water or high humidity environment
5. Methods of use for the purpose of transporting people and animals
6. Use as a foot carrier (climb on top of the robot, or dangle under the robot)

When connecting stop-related peripherals (safety fence, etc.) and various signals of the robot (external emergency stop, fence, etc.), it is necessary to confirm the robot stop action to avoid wrong connection and other abnormal situations.

1.5. Safe operating procedure

1. When teaching and manual robots, do not wear gloves to operate the teaching device and operation panel.
2. In the point operation of the robot, a lower speed ratio should be adopted to increase the control opportunity of the robot.
3. Consider the movement trend of the robot before pressing the click key on the teaching device.
4. The movement trajectory of the avoidance robot should be considered in advance and confirmed that the route is not interfered with.
5. The area around the robot must be clean, free of oil, water and impurities.

1.6. Production run

1. Before starting up and running, it must know all the tasks that the robot will perform according to the programmed program.
2. The position and status of all switches, sensors and control signals that will move the robot must be known.
3. Must know the location of the emergency stop button on the robot control cabinet and peripheral control equipment, and be ready to use these buttons in emergency situations.

Never assume that the robot has finished the program without moving, because the robot is most likely waiting for an input signal to keep moving.

2. Installation work

⚠ Danger
Robot control cabinet power ground must be grounded, the company will not be responsible for damage to components, electric shock, fire, other accidents or failures caused by ungrounded wire!

2.1. Transport

⚠ Caution

Carry out the handling according to the handling Angle value of each axis stipulated by each robot model.

⚠ Danger

Do not stay or work under hanging heavy objects.

The hoisting and fixing of the components and the direction of the crane driver must be carried out by experienced personnel.Loose parts and larger components must be carefully secured to the lifting device during replacement to avoid hazards.

2.2. Unpacking inspection

⚠ Caution

1. Visually inspect the robot to make sure it is not damaged.Do not install robots that are damaged or lack parts; otherwise, serious accidents and injuries may occur;

2. Before moving the robot, please check the stability of the robot to avoid tilt danger;

3. Cable packaging is vulnerable to mechanical damage. Cable packaging, especially connectors, must be handled carefully to avoid damage to cable packaging, otherwise the robot cannot operate normally;

4. When unpacking, please confirm carefully: whether there is damage during transportation;Whether the type and specification of the machine nameplate are consistent with the order requirements.In case of model discrepancy or device omission, please contact the manufacturer or supplier as soon as possible.

2.3. Install

2.3.1. Wiring diagram

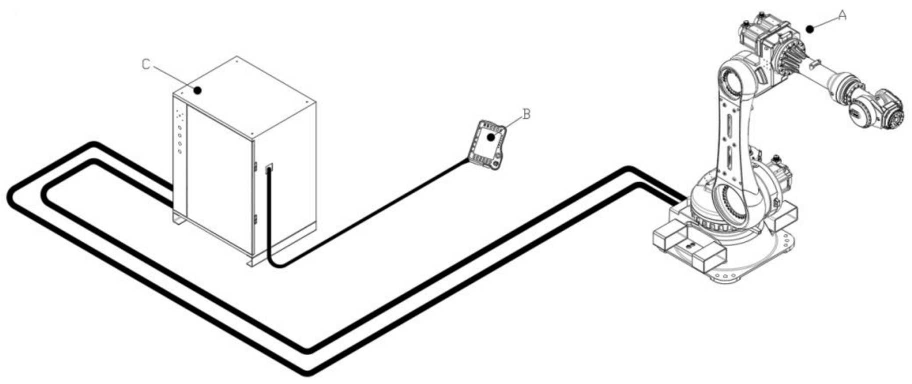


Figure 2-1 Robot wiring diagram

A Robot body

B Demonstrator

C Control cabinet cabinet

2.3.2. Matters needing attention

1. The control cabinet must be installed and operated outside the potential explosion area;
2. It is strictly prohibited for any personnel to step on the cable or motor of the robot to carry out barbaric operation on the machine;
3. Fix the control cabinet and the robot body without tilting or shaking;
4. Disconnect the power supply before connecting the power cable on the control cabinet.

2.4. Control cabinet installation

Installation environment

1. Ambient temperature: The ambient temperature has a great impact on the life of the controller. Do not allow the ambient temperature of the controller to exceed the allowable temperature range (0 ° C ~45 ° C).

2. Install the controller vertically on the surface of the flame-retardant object in the installation cabinet, and there should be enough space around for heat dissipation.

3. Install it in a place that is not susceptible to vibration.The vibration should not be greater than 0.6G.Take special care to stay away from equipment such as punches.

4. Avoid installation in direct sunlight, damp, and watery places.

5. Avoid places with corrosive, flammable and explosive gases in the air.

6. Avoid installation in places with oil pollution and dust, and the installation site pollution registration is PD2.

7.NRC series products are installed in cabinets and need to be installed in the final system. The final system should provide corresponding fire protection shells, electrical protection shells, and mechanical protection shells, and comply with local laws and regulations and relevant IEC standards, as shown in the figure

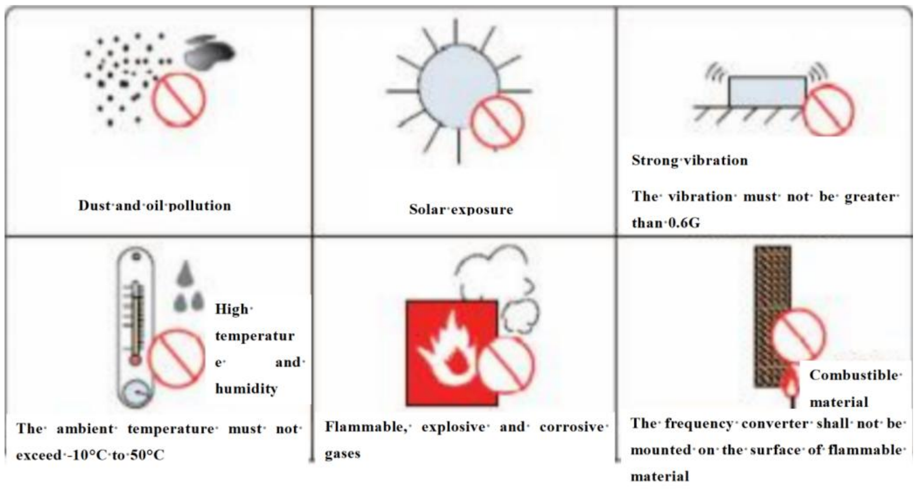

Dust and oil pollution
Solar exposure
Strong vibration
The vibration must not be greater than 0.6G
High temperature and humidity
The ambient temperature must not exceed -10°C to 50°C
Flammable, explosive and corrosive gases
Combustible material
The frequency converter shall not be mounted on the surface of flammable material

Figure 2-2 Installation requirements for the control cabinet

Installation position

1. The control cabinet should be installed outside the robot's action range (outside the safety bar).

2. The control cabinet should be installed in a position where the robot can be seen clearly.

3. The control cabinet shall be installed in a position that is convenient for opening the door for inspection.

4. The control cabinet must be at least 500mm away from the wall to keep the maintenance channel unblocked.

2.5. Cable requirements

1. Classify cables

Class 1: sensitive signals (low-voltage analog signals, high-speed encoder signals, high-speed communication signals, plus or minus 10V analog signals, low-speed 422, 485 signals, digital input and output signals).

Class 2: Interference signals (low voltage power supply, contactor control line, motor line with recorder High voltage AC power line, motor line without recorder).

2. Cable selection Input/Output Main loop cables are recommended to use symmetrical shielded cables.Compared with four-core cables, the use of symmetrical shielded cables can reduce the electromagnetic radiation of the entire conduction system.

3. Recommended power cable type - symmetrical shielded cable, as shown in the figure.

4. The recommended signal cable type is twisted pair shielded cable, as shown in the figure.

Caution

Twisted-pair shielded cables are recommended for digital signal cables.

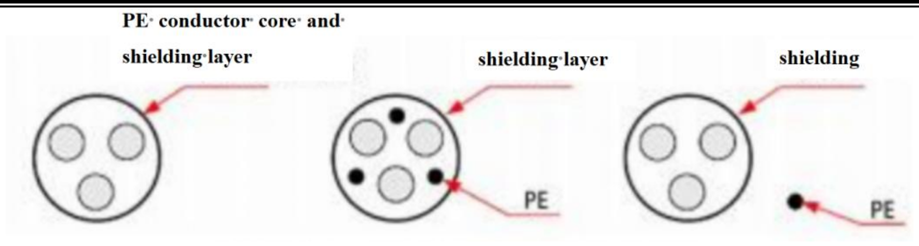


Figure 2-3 Symmetrical shielded cable

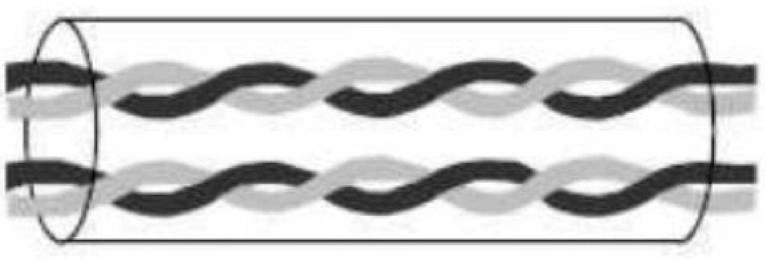


Figure 2-4 twisted pair shielded cable

3. Recommended communication cable types - Shielded communication cable,the crystal head used in the figure must be a metal shell to be shielded， The shielding layer of the communication cable is compacted with the shielding iron shell of the crystal head.

2.6. Wiring requirement

1. Power cables should be laid away from all signal cables.

2. Motor cables, input power cables and control loop cables should not be routed in the same slot.

3. Avoid electromagnetic interference caused by the coupling of motor cable and control loop for long distance and walking line.

4. Keep at least 100mm distance between cables of different grades in the same slot.Cables of different grades are arranged separately. When long-distance cables are routed in the same direction, the distance between cables of different grades should be maintained at least 100mm.The metal part of the controller is directly connected to the backplane using a conductor as the backplane (using a zinc plate that has not been sprayed).Keep the cables separated according to the grade. If the cables of different grades must be crossed, they should be kept at 90° .

2.7. Grounding requirement

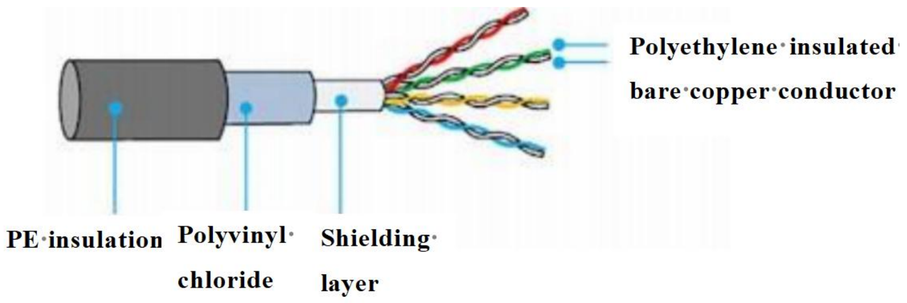


Figure 2-5 Shielding communication cables

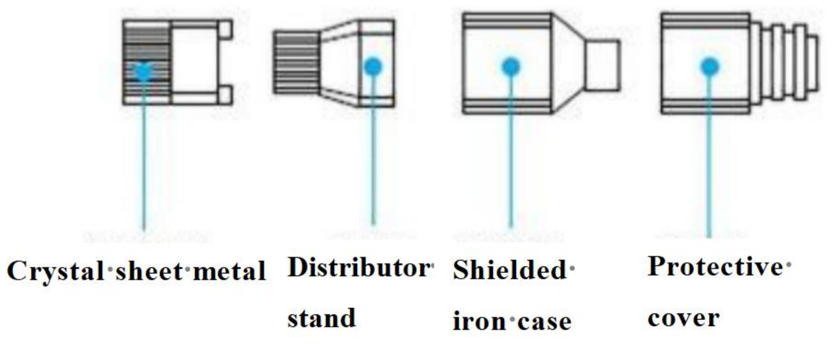
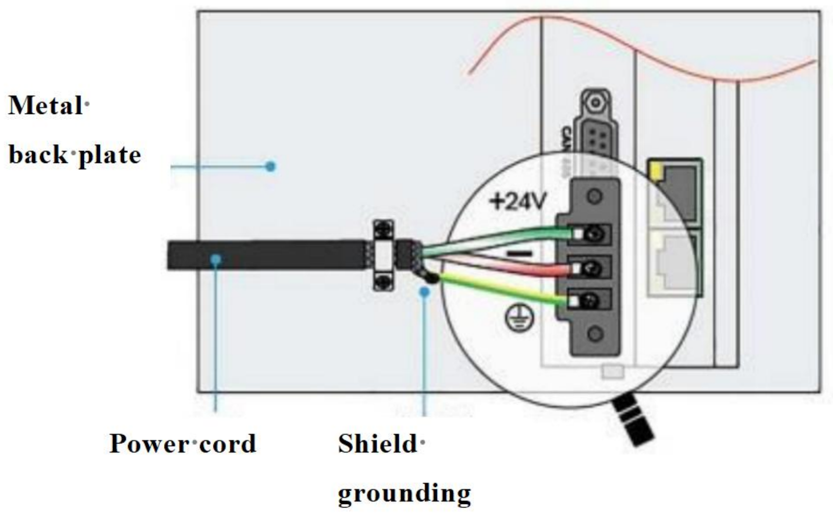


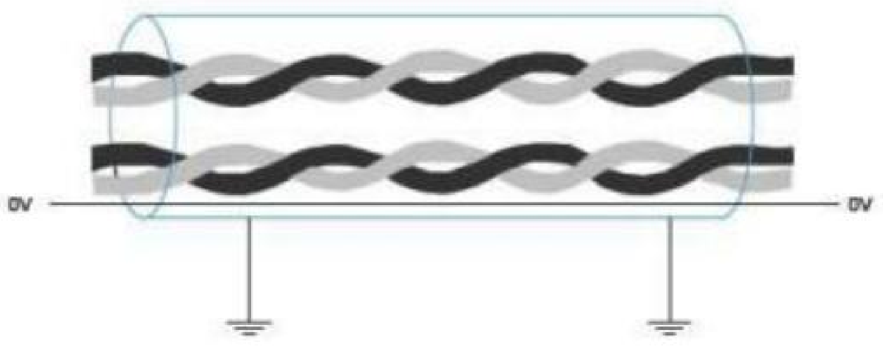
Figure 2-6 Schematic of a shielded metal shell crystal head

Be sure to ground the grounding terminal, otherwise there may be a risk of electric shock or interference resulting in misoperation.

1. Power cable grounding requirements, as shown in the figure



Differential signal cables (CAN/RS485/RS422) adopt twisted-pair shielded cables, and the shielded layer must be connected to 0V at both ends of the cable, as shown in the figure.



Wiring precautions

1. The personnel involved in wiring and inspection must be professionals with appropriate skills

2. The product must be reliably grounded, the grounding resistance should be less than 4 ohms, and the neutral line (neutral line) cannot be used instead of the ground line

3. The wiring must be correct and firm to avoid product failure or unintended consequences

4. The surge absorption diode connected with the product must be connected in the specified direction, otherwise it will damage the product

5. Disconnect the power supply before inserting or removing the plug or opening the chassis

6. Try to avoid the signal cable and power cable passing through the same pipe, the distance should be more than 30mm

7. Signal line, encoder (PG) feedback line, please use multi-stranded wire and multi-core stranded shielded wire.The maximum length of the command input line is 3m, and the maximum of the PG feedback line is 20m.The signal cable of the coding cable is a group of twisted pairs, the power cable is a group of twisted pairs, and the battery cable is a group of twisted pairs

8. Do not frequently ON/OFF the power supply.When the power supply needs to be continuously ON/OFF repeatedly, please control it less than once in a minute.Since there is a capacitor in the power supply part of the servo unit, frequent ON/OFF will cause the performance of the main circuit components inside the servo unit to deteriorate.

9. Check the power and voltage of the switching power supply of the control system.Ensure that the voltage of the controller, teaching box, and IO module is not less than 50W. The specific power depends on the load of the IO module.

10. It is recommended that the servo switching power supply and the controller system switching power supply be used separately to prevent the servo interference with the control system.

11. The network cable connecting the control system and the servo system needs to use the super six shielded network cable.

12. If one axis corresponds to one servo, the network cables need to be connected in the order of the axis.

13. Connect cables in the sequence of controller, servo, and IO board.

2.8. Inspection work

Pre-power inspection

1. The robot has been correctly installed according to the instructions, and is firm and reliable;

2. The electrical connection is correct, and the power parameters (such as voltage, frequency, interference, etc.) are within the specified range;

3. Other facilities (such as water, air, gas, etc.) are properly connected and within the prescribed limits;

4. Communication connection is correct;

5. Peripheral devices are properly connected to the system;

6. The limit device for limiting space has been installed;

7. Safety protection measures have been adopted;

8. The surrounding environment meets the regulations (such as lighting, noise level, humidity, temperature, air pollution, etc.).

Post-power inspection

1. When using the robot, put the power supply (control power supply) of the control cabinet ON (Danger: please be sure to close the device door of the control cabinet.

2. The functions of the robot system control device, such as starting, stopping and operation mode selection, meet predetermined requirements, and the robot can move according to predetermined operating system commands;

3. Each axis of the robot can move within the expected limited range;

4. Emergency stop, safety stop circuit and device are effective;

5. Disconnect and isolate from external power supply;

6. The function of teaching device is normal;

7. Safety guard and interlock function properly, other protective devices (such as fences, warnings) in place;

8. Under the "manual" operation mode, the robot can run normally and has the ability to work.

3. Installation protection

3.1. Security protection preview

The robot system has the following safety protection devices:

1. Emergency shutdown button
2. Operation mode selection switch
3. Mechanical terminal card
4. Software limit switch

⚠ Note

Do not run the robot system with the safety guard removed or turned off.

3.1.1. Emergency off button

The robot's emergency stop button is located on the control panel of the instructor. When the emergency stop button is pressed, the robot driver will be shut off immediately.

⚠ Danger

In case of any danger to personnel or equipment, the emergency shutdown button must be pressed.If you need to continue running, you must turn the emergency shutdown button to unlock it and confirm the stop information.

3.2. Safety of users

Before using an automatic system, we must first try to ensure the safety of the robot users.When the user enters the range of the robot, it is very dangerous. Measures must be taken to prevent the user from entering the range of the robot.The following lists the precautions. Take proper measures to ensure the safety of personnel and equipment.

1. Personnel using Aitron robots shall be trained by Aitron Company.

2. In order to ensure the safety of robot users, display or sound devices such as alarm lights and buzzers shall be set outside the security zone such as guardrail or grating to inform users of the current state of the robot.Because when the robot stops running, there is waiting for the start signal state or delay waiting state, in such a state, the robot may continue to run at any time, strictly prohibit the use of personnel into the robot movement range, otherwise it will cause personal injury accidents, or even serious casualties.

3. Be sure to set up safety fence and safety door around the robot, so that users cannot enter the safety fence when the safety door is closed.Interlock switch, safety lock and warning light (sound) should be set on the safety door. When the user opens the safety door, the robot will stop and the light (sound) showing the current stopped state of the robot will be displayed.

4. The ground cable of the peripheral device must be connected to the ground cable of the robot.

5. Peripheral devices should be placed as far as possible out of the robot's range of motion.

6. If possible, mark the maximum range of motion of the robot (including the holding tool) should be drawn on the ground as far as possible.

7. As far as possible, the foot mat alarm switch or photoelectric switch is set on the ground. When the user is about to enter the movement range of the robot, the alarm is sent through the buzzer or warning light to stop the robot, so as to ensure the safety of the user.

8. The transfer switch used to control the power supply on and off of the control cabinet is equipped with a keyhole, and the safety lock should be installed as required, so that personnel other than those responsible for operating the robot cannot connect the power supply of the robot system.

9. When transporting and installing the robot, please strictly follow the methods required by Ai Chuang Robot Company to prevent human injury caused by the robot tipping.

10. Note that the power supply of the robot system should be disconnected before debugging peripheral equipment.

11. When the robot is installed and used for the first time, please be sure to proceed at a low speed and then gradually accelerate to ensure that there is no abnormality.

Do not use the robot in the following situations, otherwise it will not only cause adverse effects on the robot and peripheral equipment, but also may cause injury to the user.

1. In an inflammable environment
2. In an explosive environment
3. In the presence of large amounts of radiation
4. In water or high humidity environment
5. Methods of use for the purpose of transporting people and animals
6. Use as a foot carrier (climb on top of the robot, or dangle under the robot)

When connecting stop-related peripheral devices (safety fence, etc.) and various signals of the robot (external emergency stop, fence, etc.), it is necessary to confirm the robot stop action to avoid wrong connection and other abnormal situations.

3.3. Robot stop method description

The Aitron robot has three stopping methods as follows.

3.3.1. Robot power off stop

Through the power transfer switch of the control cabinet panel, the power supply of the control system is cut off, so that the robot's action stops quickly.In this method, the servo power supply is disconnected, so the deceleration trajectory of the robot is not controlled, and the robot will have large vibration, which will have a certain impact on the motor brake and reducer. Frequent use of serious will lead to robot failure.Avoid operations that are powered off and stopped in daily situations.

⚠ Note

When the robot peripheral emergency stop signal trigger, the teaching device emergency stop signal trigger, the control cabinet panel emergency stop signal trigger, all three fail at the same time, the power should be cut off immediately through the power transfer switch.

3.3.2. Robot control stop

This method can make the robot slow down and stop quickly, and disconnect the motor power and brake power.When the external stop signal (lead access IO board in the control cabinet) is triggered, or the safety door signal (lead access IO board in the control cabinet) is triggered, or the emergency stop signal of the demonstrator is triggered, or the control cabinet panel signal is triggered, the robot enters the control stop mode, the demonstrator displays the corresponding alarm information, the robot stops, and the motor and brake power supply is disconnected.

⚠ Note

External Stop Signal, Safety door signal, and Emergency Stop signal of indicator are all connected to the I/O board in the control cabinet. When the I/O board fails abnormally, the emergency stop signal may fail.After triggering such signals, the robot user should confirm again that the brake is disconnected (the motor makes an obvious "click" sound), and the alarm screen appears on the page of the demonstrator before entering the movement range of the robot.If the robot cannot be effectively stopped, the emergency stop button on the panel of the control cabinet should be operated first, and then the power supply of the robot system should be cut off through the power transfer switch of the control cabinet.

3.4. Robot hold stop

This is how the servo power is maintained to slow down and stop the robot.The robot slows down to stop, suspending the execution of a program.Aitron robot stop mode is as follows:

Emergency stop mode	mode	Control cabinet emergency stop	Teach scram	External emergency stop	The safety door stopped abruptly	Switch off	Run down
A	remote	E-Stop	E-Stop	E-Stop	E-Stop	P-Stop	B-Stop
	automatic	E-Stop	E-Stop	E-Stop	E-Stop	P-Stop	B-Stop
	manual	E-Stop	E-Stop	—	—	P-Stop	B-Stop

B	remote	E-Stop	E-Stop	E-Stop	E-Stop	P-Stop	B-Stop
	automatic	E-Stop	E-Stop	E-Stop	E-Stop	P-Stop	B-Stop
	manual	E-Stop	E-Stop	E-Stop	E-Stop	P-Stop	B-Stop

E-Stop: Motor and brake power disconnected stop

P-Stop: power disconnected brake stop

B-Stop: hold stop

One: invalid

4. The Basic knowledge of robotics

4.1. Coordinate system

In industrial robot systems, programmers first need to understand the multiple coordinate systems defined in the system and their interrelationships.The industrial robot system mainly includes four coordinate systems: joint coordinate system, rectangular coordinate system, tool coordinate system and user coordinate system.The coordinate system contains three-dimensional x,y,z coordinate information and three-dimensional rx,ry,rz space attitude information.The robot's motion trajectory is composed of multiple target points. Similar to the coordinate system, the target point contains three-dimensional x,y, and z coordinate information and rx,ry, and rz space attitude information.Any target point belongs to a user coordinate system, that is, the coordinate information of the target point is relative to the user coordinate system to which it belongs.When a programmer modifies the coordinate information of a user coordinate system, the coordinate information of all target points belonging to that user coordinate system with respect to the global coordinate system will change.

4.1.1. Joint coordinate system

Individual movement of each joint axis of the robot.

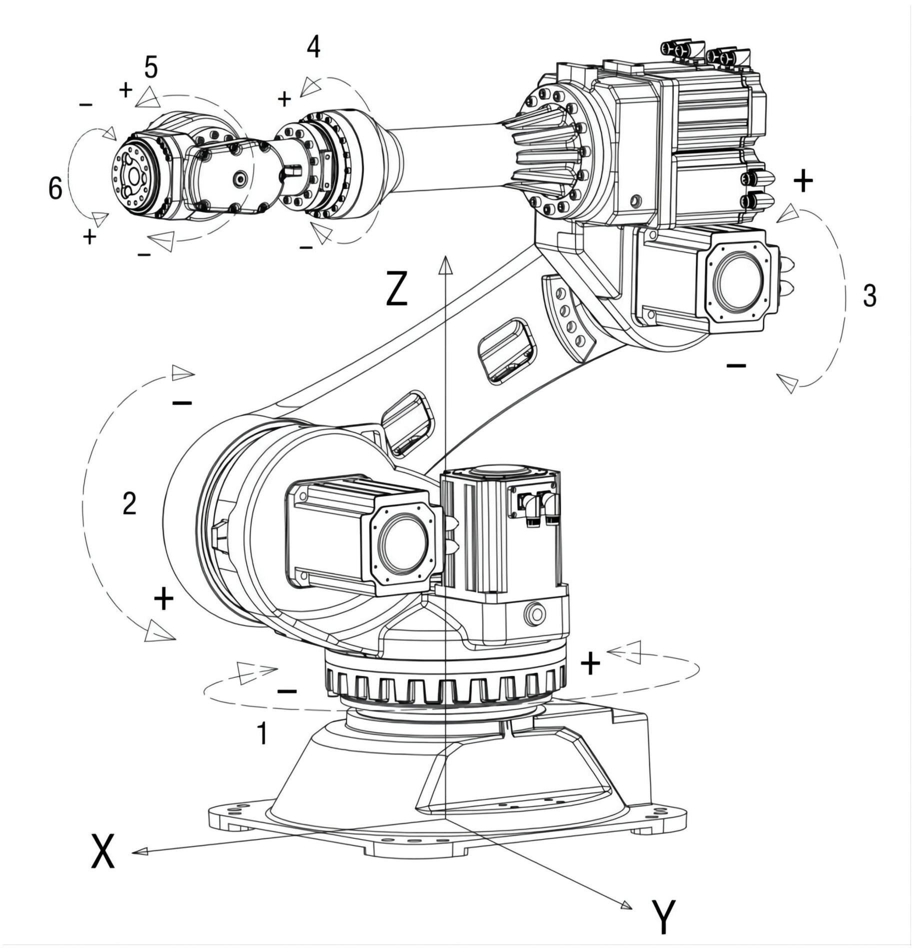


Figure 4-16 Axis robot joint direction

4.1.2. Rectangular coordinate system

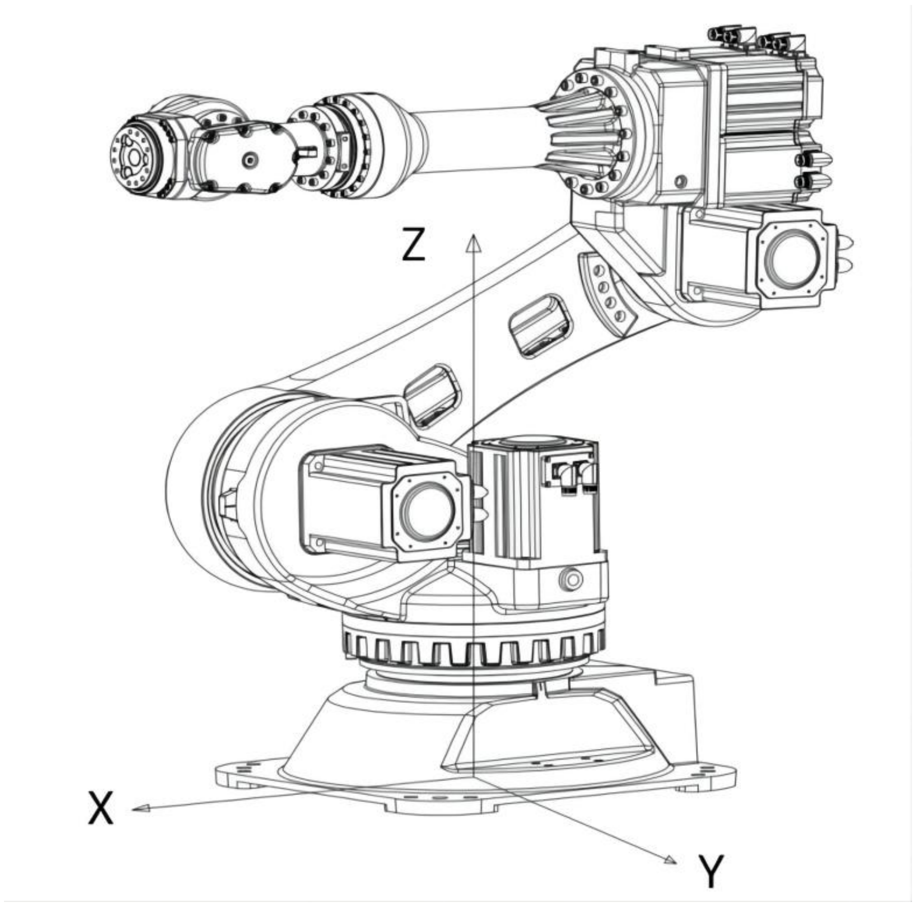
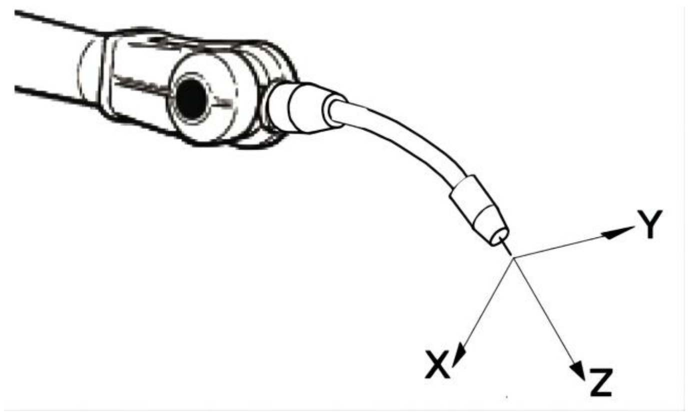


Figure 4-2 robot base coordinate system

The installation position information of the robot is described, that is, the position of the robot base in the global coordinate system.In general, the robot base is installed vertically on the ground, and the base coordinate system and the global coordinate system coincide.However, if the robot base is raised or the robot is inverted or mounted on the wall, the base coordinate system and the global coordinate system do not coincide.

4.1.3. Tool coordinate system



This section describes the installation position information of the processing tool.The tool coordinate system describes the position information of the tool relative to the end flange of the robot.

4.1.4. User coordinate system

The installation position information of the workpiece to be machined is described.The user coordinate system describes the position information of the workpiece relative to the global coordinate system.

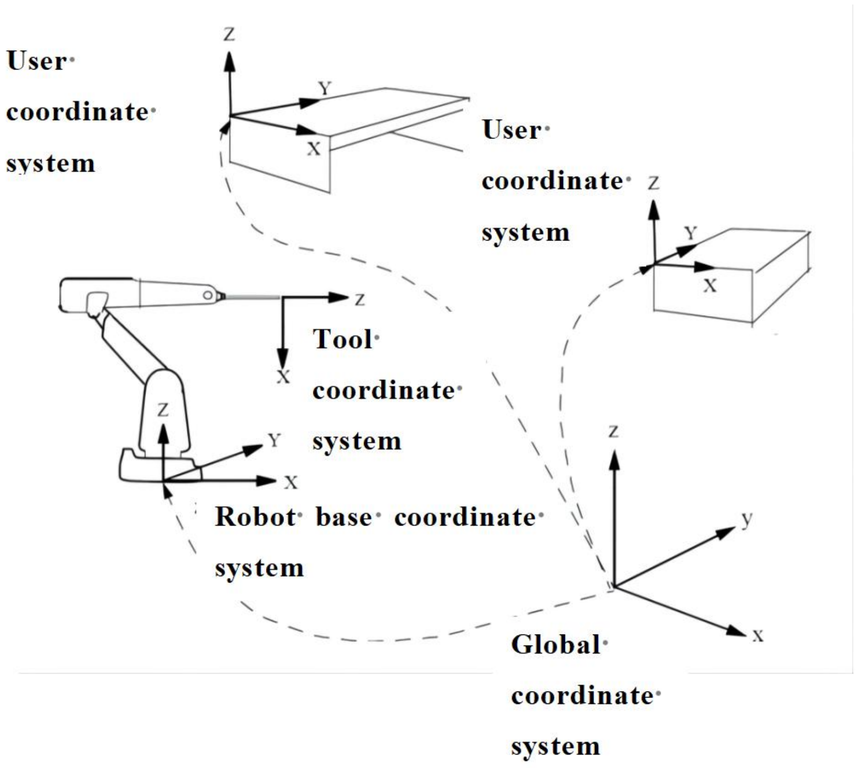


Figure 4-2 User coordinates

By default, a tool coordinate system tool0 overlapped with the robot flange, indicating that there was no tool installed in front of the robot.A user coordinate system, user0, coincides with the global coordinate system.As shown in the following picture.

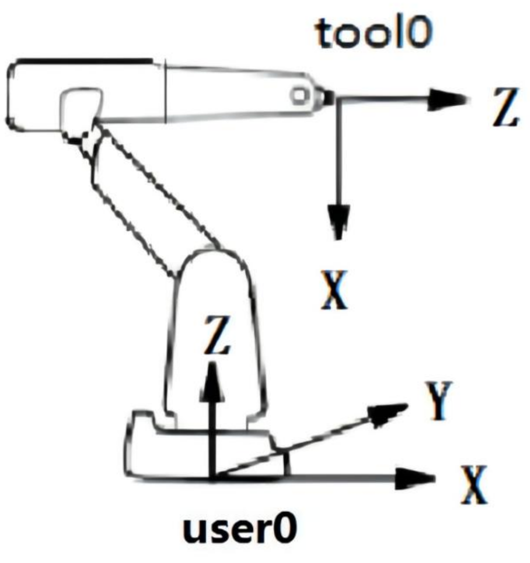


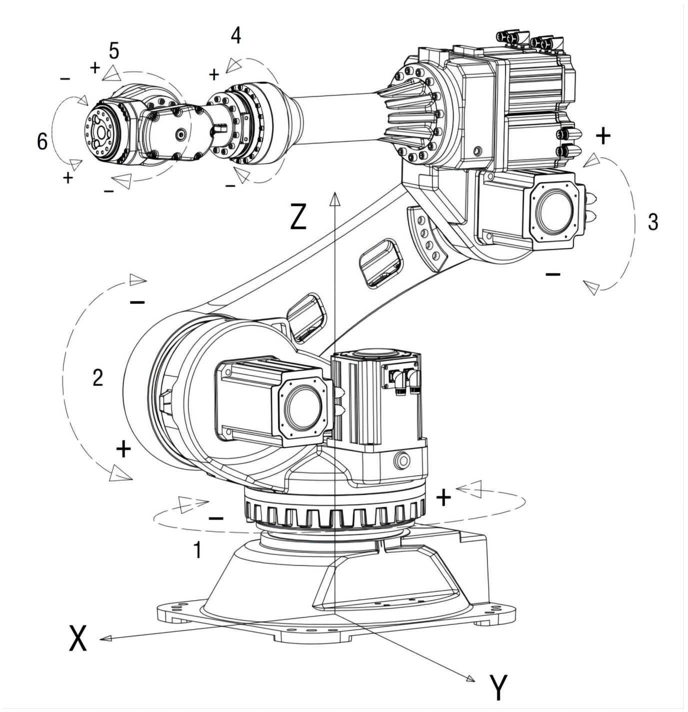
Figure 4-3 tool0 and user0

4.2. Coordinate axis operation

4.2.1. Joint coordinate axis operation

In the joint coordinate system, each axis of the robot can operate independently.

When the robot does not have the shaft operation key, no action is done.



Axis name		Axis operation	Movement
Fundamental axis	S-axis	S+/S-	Body rotation
	L-axis	L+/L-	Lower arms move back and forth
	U-axis	U+/U-	Upper arms up and down
Wrist shaft	R-axis	R+/R-	Wrist rotation
	B axis	B+/B-	Up-and-down wrist movement
	T-axis	T+/T-	Wrist rotation

4.2.2. Cartesian axis operation

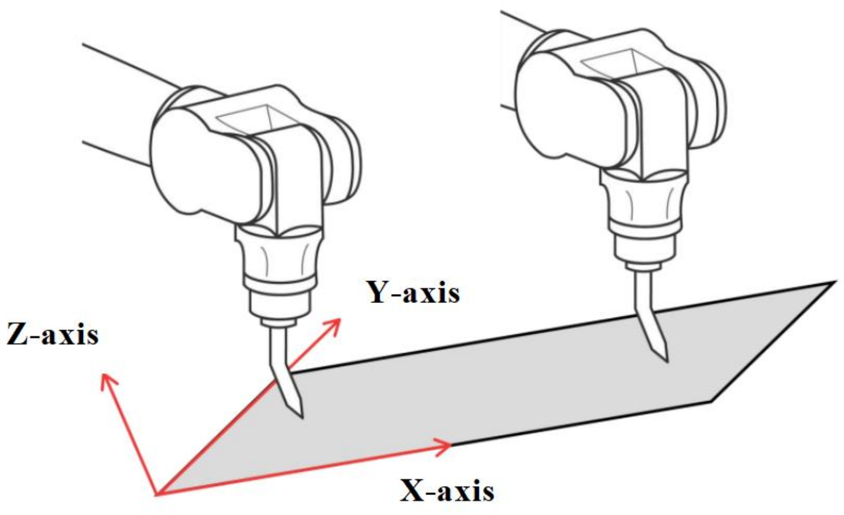
The robot moves in a rectangular coordinate system parallel to the body axes X, Y and Z

Axis name		Axis operation	Movement
Fundamental axis	X-axis	X+/X-	It moves parallel to the X-axis
	Y-axis	Y+/Y-	It moves parallel to the Y-axis

	Z-axis	Z+/Z-	It moves parallel to the Z-axis
Attitude axis	A-axis	A+/A-	Rotate around the X-axis
	B axis	B+/B-	Rotate around the Y-axis
	C-axis	C+/C-	Rotate around the Z-axis

4.2.3. Tool coordinate axis operation

The movement of the tool coordinate is not affected by the change of the position or posture of the robot, and the movement is mainly based on the effective direction of the tool.Therefore, the tool coordinate motion is best used in applications where the tool position is always the same as the workpiece and moves in parallel.As shown in the following picture.



Axis name		Axis operation	Movement
Fundamental axis	TX-axis	TX+/TX-	It moves parallel to the TX-axis
	TY-axis	TY+/TY-	It moves parallel to the TY-axis
	TZ-axis	TZ+/TZ-	It moves parallel to the TZ-axis
Attitude axis	TA-axis	TA+/TA-	Rotate around the TX-axis
	TB axis	TB+/TB-	Rotate around the TY-axis
	TC-axis	TC+/TC-	Rotate around the TZ-axis

4.2.4. User coordinate axis operation

In the user coordinate system, X, Y and Z axes of any Angle are set at any position in the robot's action range, and the robot moves parallel to these axes.

Axis name		Axis operation	Movement
Fundamental axis	UX-axis	UX+/UX-	It moves parallel to the UX-axis
	UY-axis	UY+/UY-	It moves parallel to the UY-axis
	UZ-axis	UZ+/UZ-	It moves parallel to the UZ-axis

Attitude axis	UA-axis	UA+/UA-	Rotate around the UX-axis
	UB axis	UB+/UB-	Rotate around the UY-axis
	UC-axis	UC+/UC-	Rotate around the UZ-axis

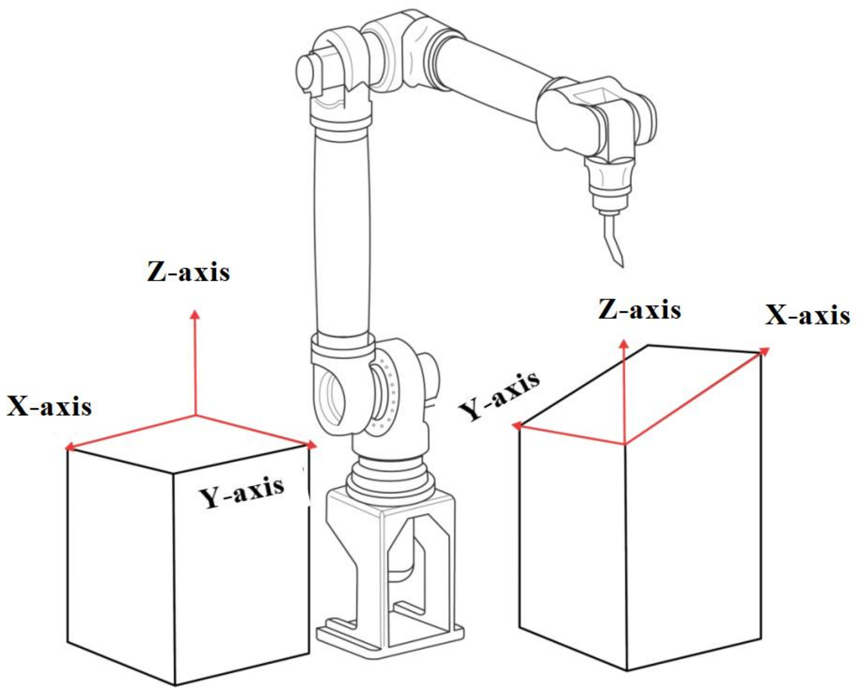
4.2.5. Examples of use of user coordinate system

Through the use of user coordinates, various teaching operations can be made easier.

Here are a few examples to illustrate.

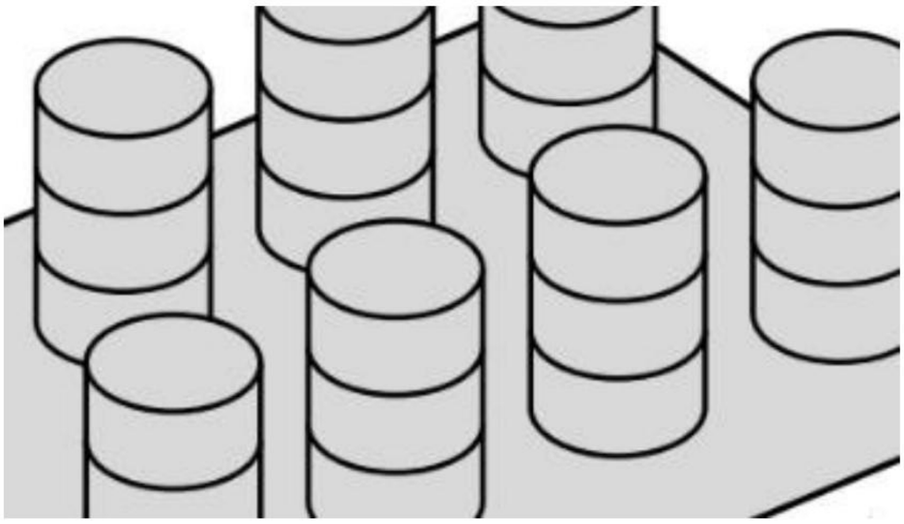
When there are multiple fixture tables:

If the user coordinates set by each fixture table are used, manual operation can be made easier.



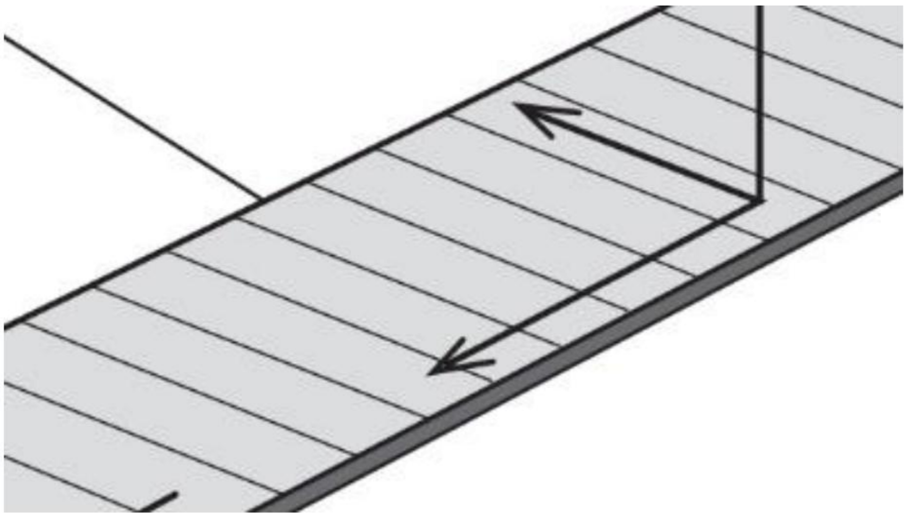
When engaged in arrangement and placement operations:

If the user coordinates are set on the tray, then it becomes easier to set the displacement increase value when moving in parallel.



When running in sync with the conveyor belt:

Specify the direction of the conveyor belt movement.



4.3. External axis

Use the external axis button to switch to the external axis can be clicked to teach the external axis;The external axis supports only the joint movement.

Axis name	Axis operation	Movement
Axis 01	J1+/J1-	External axis 1 axis rotational motion
Axis 02	J2+/J2-	External axis 2 axis rotational motion
Axis 03	J3+/J3-	External axis 3 axis rotational motion
Axis 04	J4+/J4-	External axis 4 axis rotational motion
Axis 05	J5+/J5-	External axis 5 axis rotational motion

4.4. Coordinate attitude

There are many ways to express the attitude of coordinate system. RX, RY and RZ are used in this system.Rotations of the x,y, and z axes with respect to the right-handed Cartesian coordinate system are called RX, RY, and RZ rotations, respectively.Any rotation can be decomposed into a sequence of three basic rotations, the sequence order of the attitude RX, RY, RZ rotation matrix is first RZ rotation about the Z axis, then RY rotation about the Y axis, and finally RX rotation about the X axis.

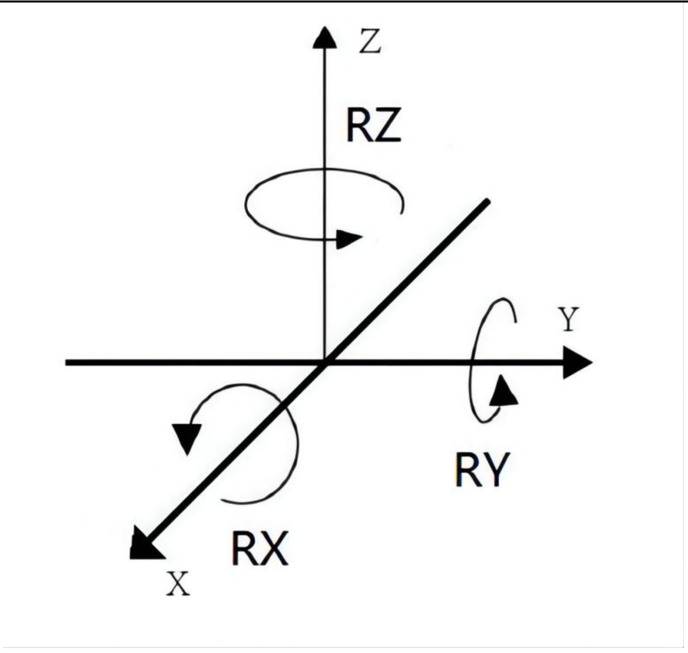


Figure 4-5 Posture in the coordinate system

4.5. Robot load

The load of the tool mounted on the end flange of the robot or the workpiece being machined contains mass, center of gravity and inertia tensor information.The position of the center of gravity is relative to the tool0 coordinate system of the robot flange, and the inertia tensor is relative to the reference coordinate system with the center of gravity as the origin and the attitude is consistent with the tool0.In order to simplify the calculation, it is usually only necessary to extract the principal inertia information Ixx, Iyy, Izz of the inertia tensor.

4.6. Singular point

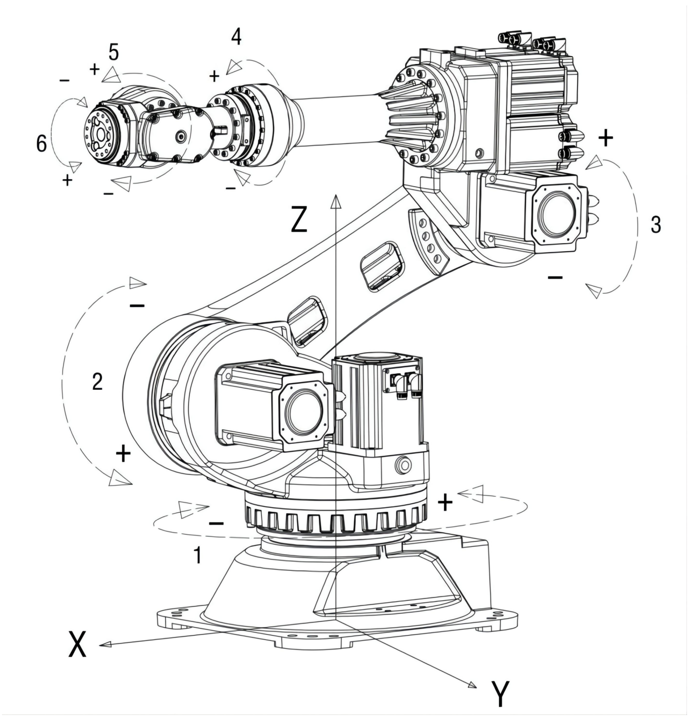
The robot may encounter singularity during its movement.The singularity will lead to the failure of linear and circular motion control calculation, but will not affect the joint motion of the robot.

When the user manually teaches the robot to move in a straight line or an arc through the instructor, the robot will stop moving if it approaches the singularity position.

There are two types of singularity points. One is that the robot reaches the limit position in the three-dimensional space and cannot move further.The other is related to the mechanical structure of the robot, for certain positions in three-dimensional space, each joint of the robot may have infinite ways to reach.

For standard 6-axis robots, the axes of 4, 5, and 6 meet at a point in space called the wrist point.When the robot is extended, the 2-axis axis, the 3-axis axis and the wrist point are in the same straight line, and the 3-axis is at a certain Angle, and the robot is in the limit position (the robot arm has been extended to reach the limit position).Therefore, when the 3-axis of the robot approaches this specific Angle, the robot is in the singularity position, and the linear and circular motions cannot work properly due to calculation failure.

When the 5 axis is 0 degrees or 180 degrees, the axis directions of the 4 axis and the 6 axis are collinear in three-dimensional space, and there are infinite combinations of the 4 and 6 axes to keep the position and attitude of the end point of the robot unchanged.Therefore, when the robot's axis 5 is close to 0 degrees or 180 degrees, the robot is in the singularity position, and the linear and circular motions cannot work properly due to calculation failure.



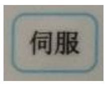
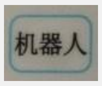
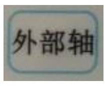
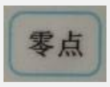
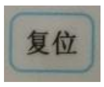
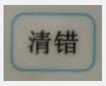
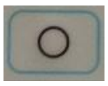
Another singularity position for standard 6-axis robots is that when the wrist point passes through the axis of 1 axis, there can be infinite combinations of 1, 2 and 3 axes to keep the position of the robot wrist point unchanged, and the linear and circular motions cannot work properly due to calculation failure.

In the process of operating the robot, or in the process of executing the robot program, the operator should try to avoid passing through or near the singularity position of the robot.When the robot passes through or approaches the singularity, the Angle of some joints may change rapidly. Please pay attention to safety.It is also possible that the robot motion fails, and it is necessary to move the robot away from the singularity position by joint motion.

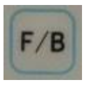
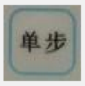
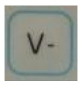
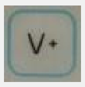
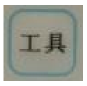
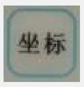
5. Instruction key and interface introduction

5.1. T30 Teaching device physical keys

Left side

	Switch current servo state
	Switch the current robot. (Only available in multi-machine mode)
	Switch between the current robot and external axis. (Only available when there is an external axis)
	Home button
	Recovery site button
	The error is cleared after the servo reports an error. (Only valid in teaching mode)）
	Reserved

Down side

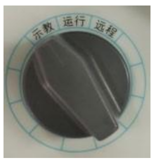
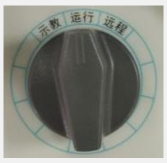
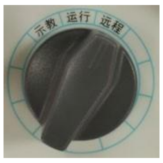
	Switch between sequential execution or reverse execution when single-step running program in teaching mode.
	Run the program step by step in the teaching mode.
	Reduce the teaching or running speed.
	Increase the teaching or running speed.
	Switch tool hand (reserved).
	Switch whether to execute the program in a single step in the teaching mode in order or in reverse order.

Right side

停止	Pause the program in run mode

启动	Start program in run mode
−	When teaching, the corresponding axis runs in the negative direction
+	When teaching, the corresponding axis runs in the positive direction

Key switch

	On the left, switch to teaching mode
	In the middle, switch to run mode
	On the right, switch to remote mode

Emergency button

	Press emergency stop

Wheel knob

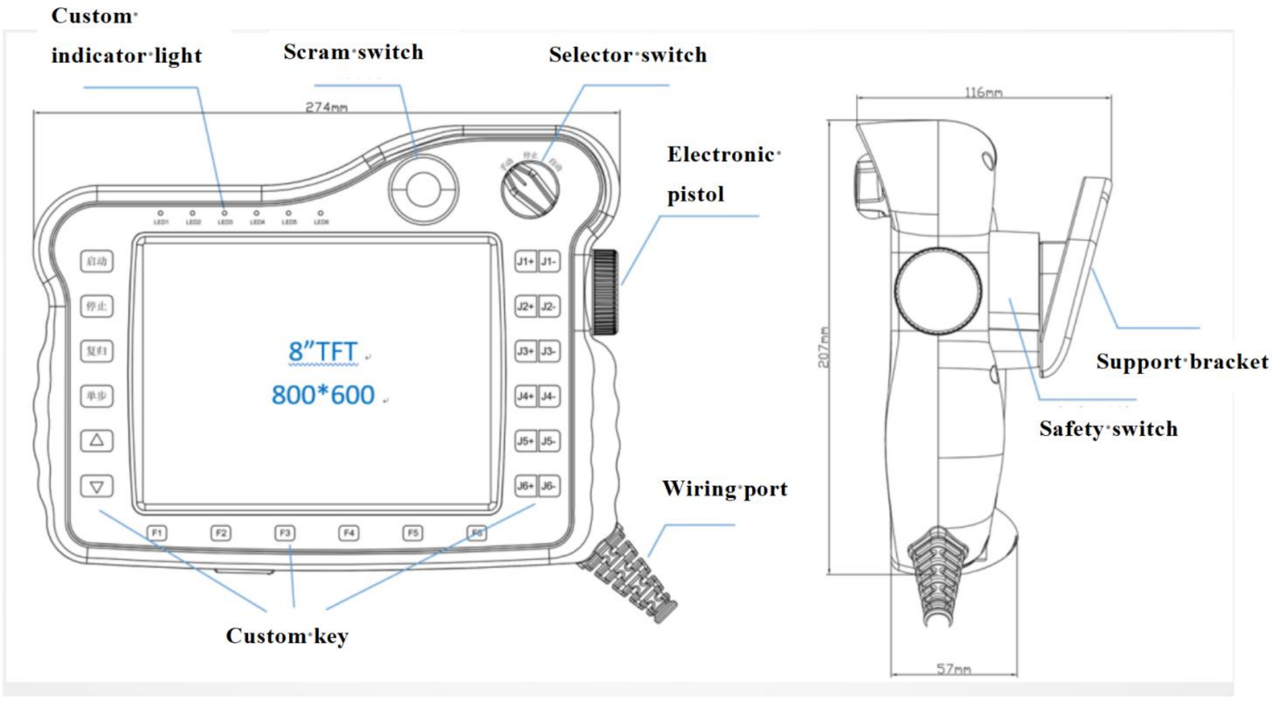
	Program interface selection to switch the previous line and the next line

Deadman key

	Three-stage key Press in the middle to power on the robot Press the bottom to power off the robot Release the key to power off the robot

5.2. Dimensional information

5.2.1. The teaching apparatus is as follows:



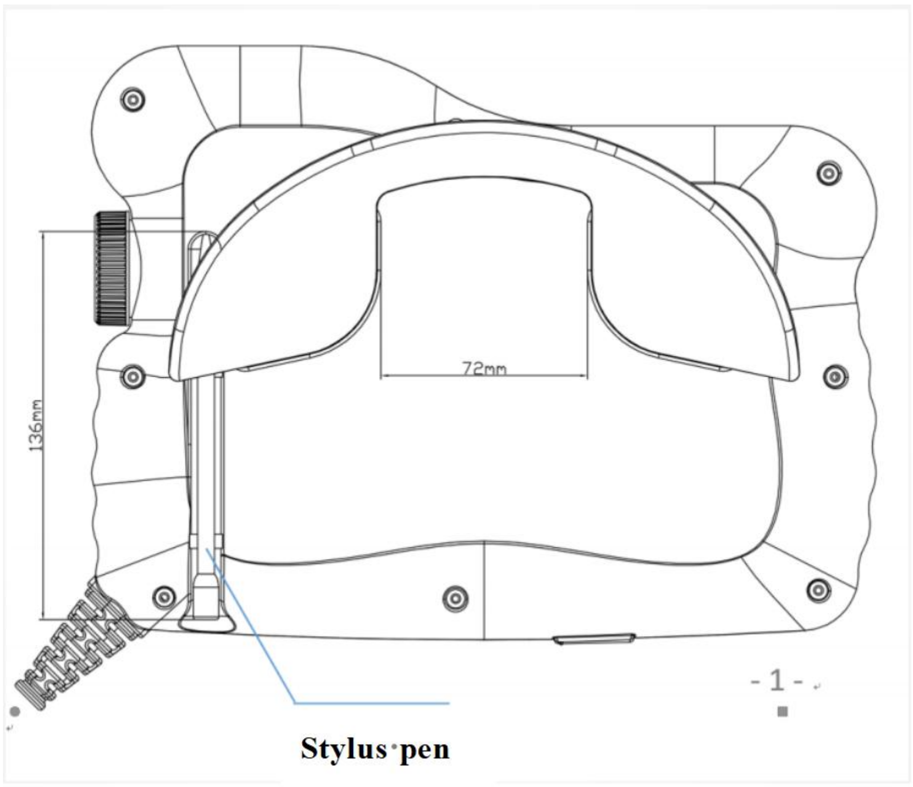


Figure 5-1 Size of the indicator

5.2.2. The diagram of the external switching box is as follows

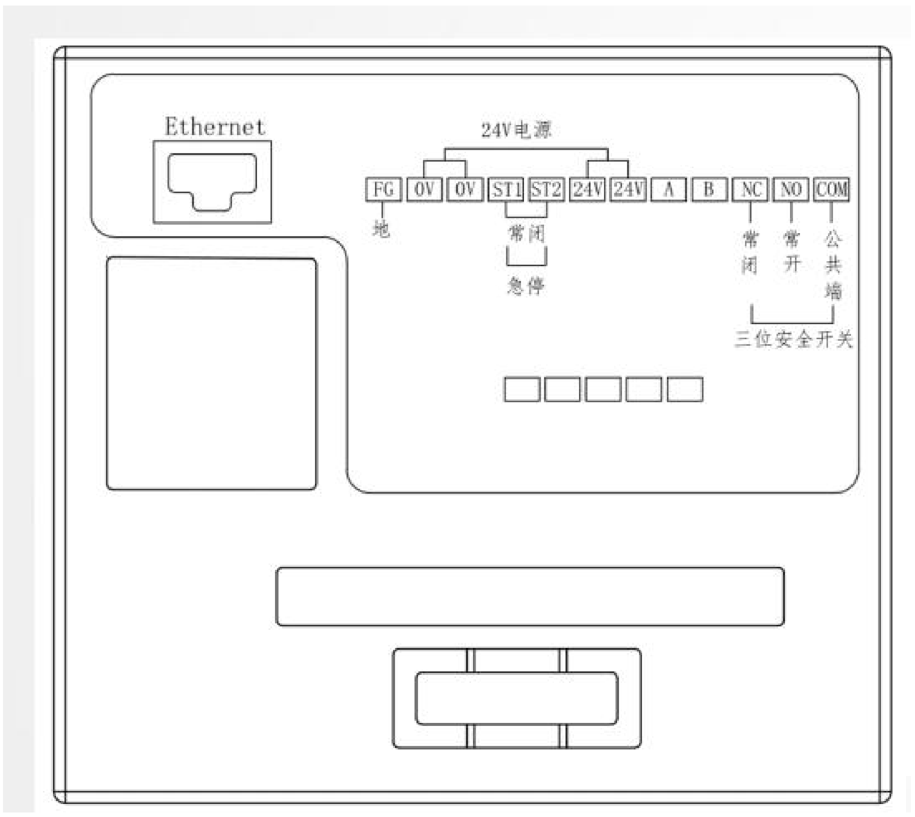


Figure 5-2 shows the conversion box

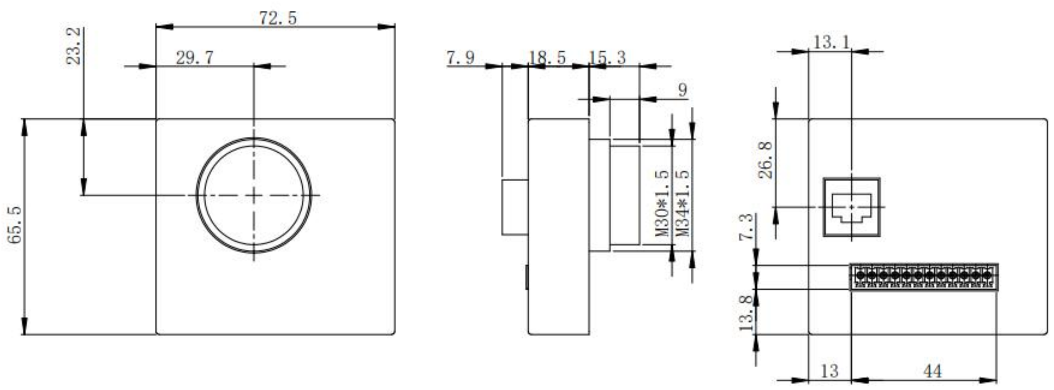


Figure 5-3 Dimensions of an adapter box

5.2.3. Adapter box installation drawing

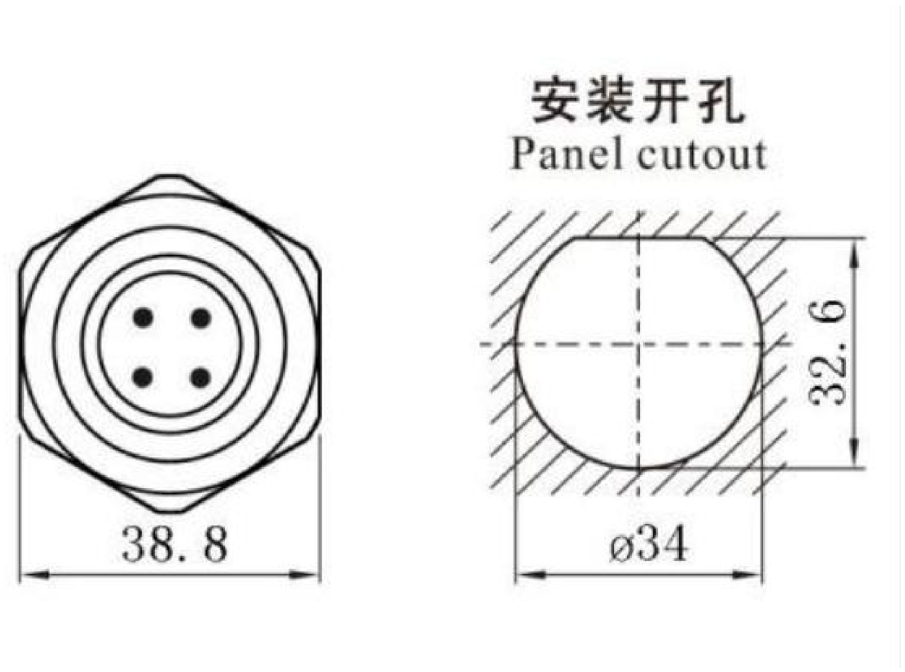
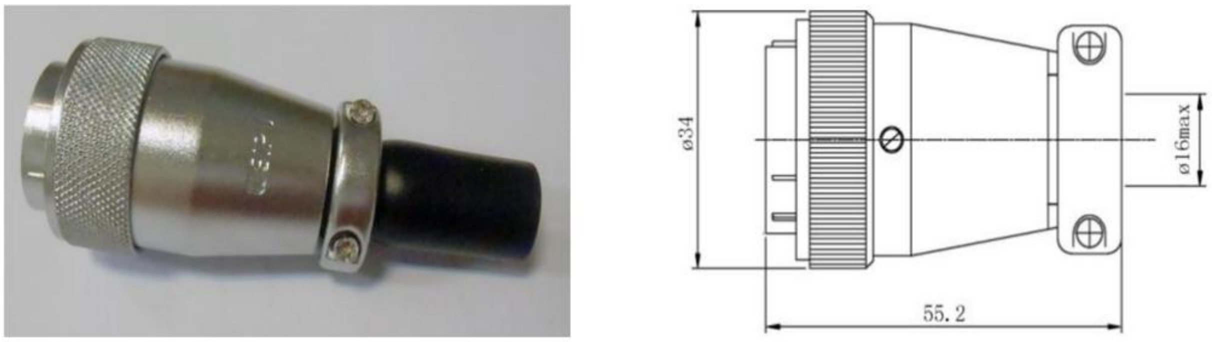


Figure 5-4 Dimensions of the adapter box

5.2.4. Connection port

Aviation plug size drawing:



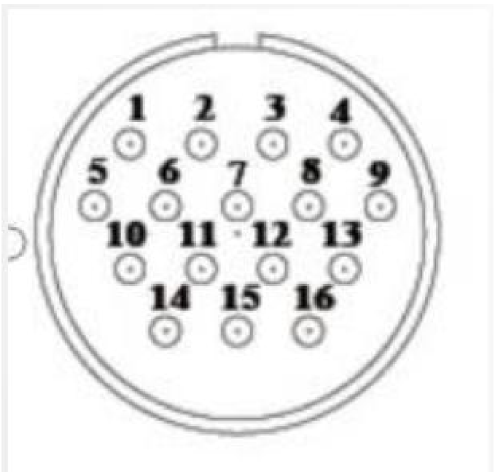


Figure 5-5 Dimensions of an aviation plug

Adapter box connector definition

pin	Name description
1	EtherNet TX+
2	EtherNet TX-
3	EtherNet RX+
4	EtherNet RX-
5	The public terminal was enabled
6	RS422_T+/CAN_H/The enable switch is always on
7	RS422T-/CANL/Enable switch normal closing point
8	RS422R-/RS485B/Emergency stop always open
9	RS422R+/RS485A/Emergency stop normal group A
10	Power supply 24V
11	Power supply 24V
12	Power supply 0V
13	Power supply 0V
14	Casing grounding
15	Emergency stop normal closing point 2
16	Emergency stop normal closing point 1

5.2.5. Product characteristics

processor	TI Sitara AM335x ARM Cortex-A8 32-bit RISC Microprocessor, up to 1GHz
memory	512MB DDR3, 4GB eMMC
Liquid crystal screen	TFT 8inch 800*600
touch	Reinforced type 4-wire resistance screen
Operating system	LINUX 2.6
panel	Function key: 15 key axis key: 14 key indicator light: 6
USB port	USB 2.0:1个

Communication interface	RS485,RS422, CAN, Ethernet
Functional component	Emergency stop switch: 1;Selector switch: 1;Electronic hand wheel: 1 PCS;Touch pen: 1 piece;Enable switch (three-position) : optional;
Rated input voltage/current	DC 24V/0.5A
Operating ambient temperature	-40~85℃
Ambient humidity	≤90%
weight	1.2Kg

5.2.6. Disassemble and assemble the teaching device

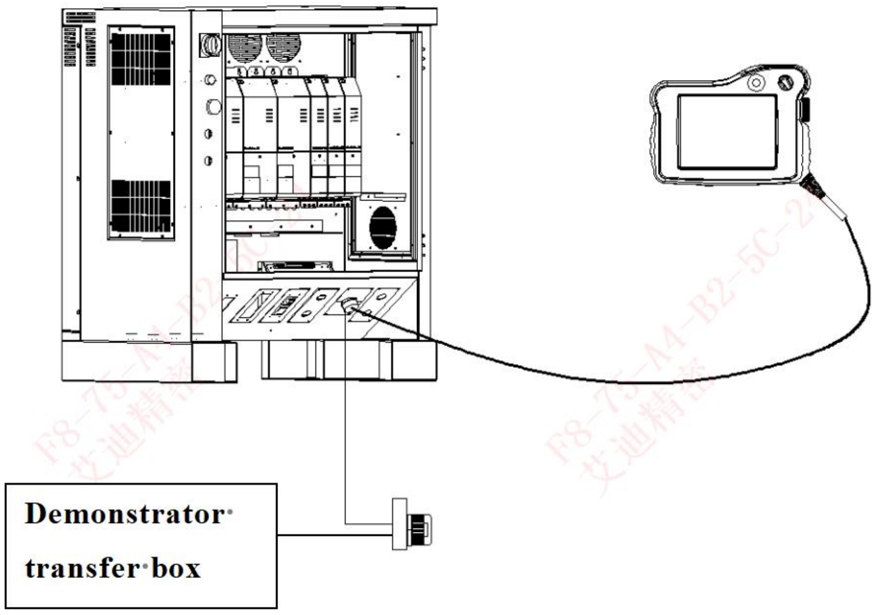


Figure 5-6 Removing and assembling an indicator

How to connect the demonstrator: Turn off the power, connect the demonstrator cable to the demonstrator switching box, and then use the power.

How to remove the indicator: turn off the power and unscrew the cable of the indicator from the converter box.

5.3. Operating system Introduction

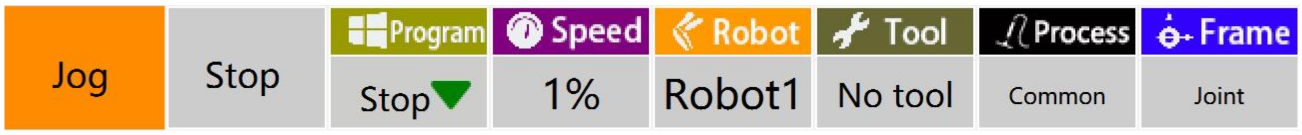
5.3.1. Basic description

This section mainly summarizes each part of the program interface.The left side of the program is the function key and other functions are shown in the table

Op	The Administrator/Technician/Operator switching page is displayed
Settings	The robot function setting screen is displayed
Function	Open the robot process selection screen
X=/ Var	Open the robot variable setting screen
Status	The robot status view screen is displayed
Project	The project preview screen is displayed
Job	Open the program command interface
Log	The error log page is displayed
Monitor	The robot monitoring screen is displayed
14:15 Thurs 2023/07/13	Date and time display

5.3.2. Status introduction

At the top of the program is the status bar, which displays the various states of the robot.



Mode state: Switch by rotating [mode selection key], including teaching mode, remote mode and operation mode.

Servo state: After starting the program, press the [Mot] button to switch from the "servo stop" state to the "servo Ready" state.

When the DEADMAN key is pressed in instruction mode or the program is run in Reproducing mode or Loop mode, the servo state switches to "Servo Run".

Program state: The running state of the program. When the program is run STEP by step in "teaching mode" or run in "reproduction mode" or "cycle mode", the program state is switched to "running" state.

Point speed increases or decreases the point speed by pressing [V-] and [V+] at the bottom of the indicator

Speed increase: Press the [V+] (Speed increase) button at the bottom of the indicator. Each time, the clicking speed changes in the following order: Fretting 1% → fretting 2% → low 5% → low 10% → medium 25% → medium 50% → high 75% → high 100%

Speed reduction: Press the [V-] (Speed reduction) button at the bottom of the teaching box. Each time, the clicking speed changes in the following order: high 100% → high 75% → medium 50% → medium 25% → low 10% → low 5% → fretting 2% → fretting 1%

Robot status: Switch by pressing the [Rob] button at the bottom of the indicator.They are "Robota" and "Robotb".

Tool status: Switch by pressing the [Jog] button at the bottom of the indicator.They are Tool 1, Tool 2, Tool 3, Tool 4, Tool 5, Tool 6, Tool 7, Tool 8, and Tool 9.

Process mode: [in teaching state] can be switched manually.They are "universal", "welding", "palletizing", "laser cutting" four states.

Coordinate system: Press the "coordinate System Switch" button on the left side of the instructor to switch.There are "joint coordinate system", "rectangular coordinate system", "tool coordinate system" and "user coordinate system".

6. Robot teaching and running

This chapter will explain the specific programming steps of Ai Chuang robot control system, as well as the detailed description of each control instruction.

6.1. Robot Preparing

6.1.1. Boot and safety confirmation

This section mainly describes the teaching operation before the boot and ensure that the safety measures are complete.

6.1.1.1. Turn on

Operation procedure

1. Check whether the cables of the servo, controller, and teaching box are properly connected.

2. Turn the main power switch ON the cabinet panel to the On position.

3. Press the green servo start button on the cabinet panel.

6.1.1.2. Safety confirmation

For safety reasons, please confirm that the emergency stop button is normal before teaching.

Confirmation of the use of the emergency stop button: Before the robot is used, please confirm the emergency stop button on the control cabinet and teaching box respectively, and check whether the servo power is disconnected when it is pressed.

1. Press the emergency stop button on the control cabinet and teaching box;

2. Ensure that the servo power is off, the indicator displays servo error, and the control cabinet servo error light is on;

3. Clear the servo error, the servo error light of the control cabinet is off, and "servo stop" is displayed on the teaching device;

4. Press the "MOT" button on the instruction box to make the servo in the servo ready state after confirming that it is normal;

6.1.2. Ready to teach

After the demonstrator is powered on and the servo has no error, confirm that the demonstrator is in the demonstrator mode. If not, rotate the key to select the mode and switch the demonstrator to the demonstrator mode.Press "MOT" (Servo Ready) on the indicator, then the "Servo Status" column at the top of the program interface will display as "Servo Ready" and blink

Servo ready

.The machine can only be enabled in the servo-ready state.

Press the "DEADMAN" button on the back of the indicator, then you can hear the sound of the robot being powered on, and the "Servo status" column shows "Servo operation" in green, indicating that the servo power is successfully connected.

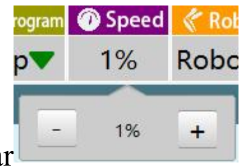
6.2. Point operation

This section mainly describes the related matters of manual operation by using the physical keys of the teaching-programming pendant in the teach mode. It includes the definition and setting of coordinate system, the method of manual operation, the speed setting and the be sure of each status during manual operation. It takes a lot of practice to become proficient.

6.2.1. Teach Speed Control

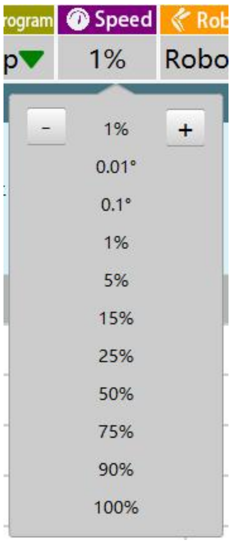
In the teach mode, the motion speed of the manually manipulation robot is modified by pressing the

{V+} (speed increase) key or {V-} (speed decrease) key on the hand-held manipulation teaching- programming pendant. For each press the manual speed changes in the following order, which is be sure by the speed display in the status area.

You can also click on the speed item in the status bar 

, which will bring up a

drop-down menu. Clicking {-} and {+} can also add or subtract speed. Click on the middle number will pop up the speed option, you can quickly select several commonly used speed.



Speed increase: press the {V+} (speed increase) key at the bottom of the teaching-programming pendant. Each press, the manual speed will change in the following order:

Inching motion 0.01°→inching motion 0.1°→1\%→5\%→10\%→15\%→25\%→ 50\%→75\%→100\%

Speed down: press the {V-} (speed down) key at the bottom of the teaching-programming pendant. Each press, the manual speed will change in the following order:

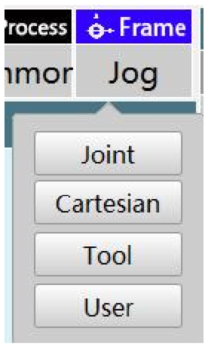
High 100\%→high 75\%→mid 50\%→mid 25\%→low 15\%→low 10\%→low 5\%→micro motion 1\%→inching motion 0.1°→inching motion 0.01°

Inching motion: inching motion speed under the joint coordinate system is 0.01 ° and 0.1 ° two grades. In the rectangular, tool and user coordinate system, there are two grades of 0.1mm and 1mm. The teach speed is based on the percentage, and the actual speed is the percentage in the status bar multiply the maximum speed of the point movement. The maximum speed of point movement is set in the setting-robot parameter-point movement interface, Please refer to the chapter of robot setup for detailed parameters and setting methods.

6.2.2. Description and Switching of Coordinate System

There are four coordinate systems in this product, namely joint coordinate system, rectangular coordinate system, tool coordinate system and user coordinate system.

- All points in the joint coordinate system are the angle values of the joint axis of the robot relative to the mechanical zero of the axis.
- Rectangular coordinate system is also known as the "base coordinate system". All points are the coordinate value (unit mm) of the robot tip (center of flange) relative to the center of the robot base.
- All points in the tool coordinate system are the coordinate value (unit mm) of the tool tip (TCP point) carried by the robot relative to the center of the robot base. See the chapter on Tool and User Coordinates for its definition and usage.
- The user coordinate system is also known as the " workpiece coordinate system", and all points are the coordinate values (unit mm) of the tool tip (flange center) of the robot relative to the origin of user coordinate system (without tool). See the chapter on Tool and User Coordinates for its definition and usage.
- In the teach mode, press the {Coordinate System Switch} key in the physical keys area on the left side of the teaching-programming pendant. Every time the key is pressed, the coordinate system will be switched in the following order and be sure by the display of the top status bar.
- You can also click on the coordinate system column of the status bar to pop up the coordinate system selection menu and switch by clicking on the corresponding coordinate system.



Joint→rectangle→tool→user

6.2.3. Point operation

1.To perform the pointing operation of the robot, the following steps are specified:

2.Starting up.

3.Check whether the emergency stop key is intact or pressed.

4.Press the {MOT} key of the teaching-programming pendant to be sure that the servo status is "servo preparation".

5.Select the coordinate system you want to use.

6.Adjust to the appropriate speed.

7.Press the {DEADMAN} key (the orange key on the back of the teaching-programming pendant), and do not release.

8.Use the keys in the physical key area on the right side of the teaching-programming pendant to operate the robot to move.

9.Release the {DEADMAN} key.

6.3. Programming

This section will mainly introduce the operation of the instructions of this product. It includes the operation of creating, modifying, deleting, copying and renaming programs, inserting, modifying, deleting and copying instructions, as well as the specific function description of each instruction, and provides specific examples. If you want to master it skillfully, you need to use it many times.

6.3.1. Program New/Open/Delete/Rename/Copy

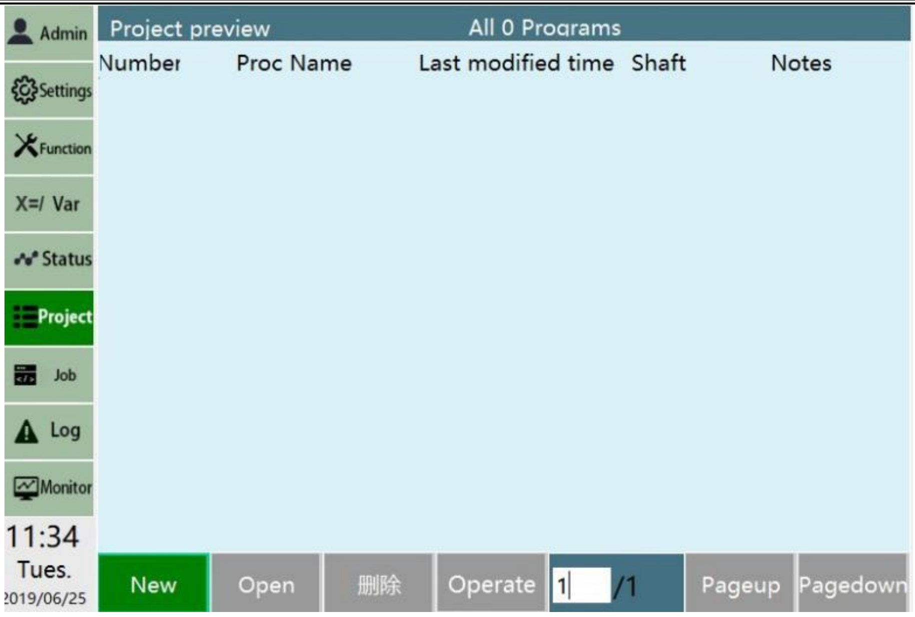
Users need to enter the program interface and use the bottom button to perform related operations to insert/modify/delete/copy/rename instructions of the program.

6.3.1.1. New Program

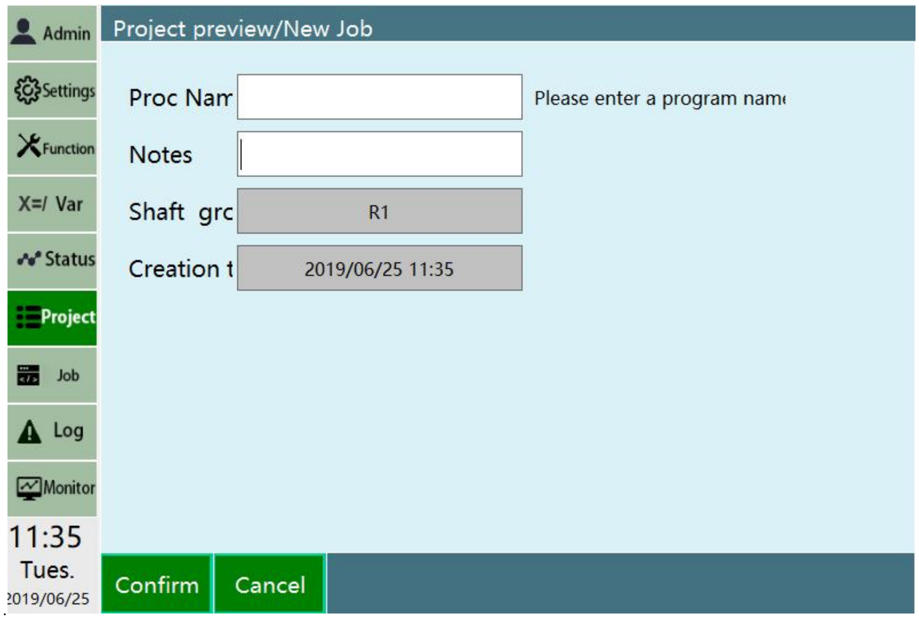
New programs need to be created by clicking on the {New} button at the bottom of the program interface.

The new program is under the selected program. The relevant steps are as follows：

Enter the program interface；



1.In the Program Creation window, enter the corresponding program name and other parameters



Proc Nar

Please enter a program nam

Notes

Shaft grc R1

Creation t 2019/06/25 11:35

2.Click on the {OK} button at the bottom, the program is created successfully, and jump into the new program. If you want to cancel the new program, click on the {Cancel} button.

3.To cancel the new program, click the "Cancel" button.

Caution

The program name must be a string of two or more characters starting with a letter/Chinese character.The program name cannot be the name of an existing program.

6.3.1.2. Program Open

To open an existing job file, the user needs the following steps:

1.Open the "engineering" interface;

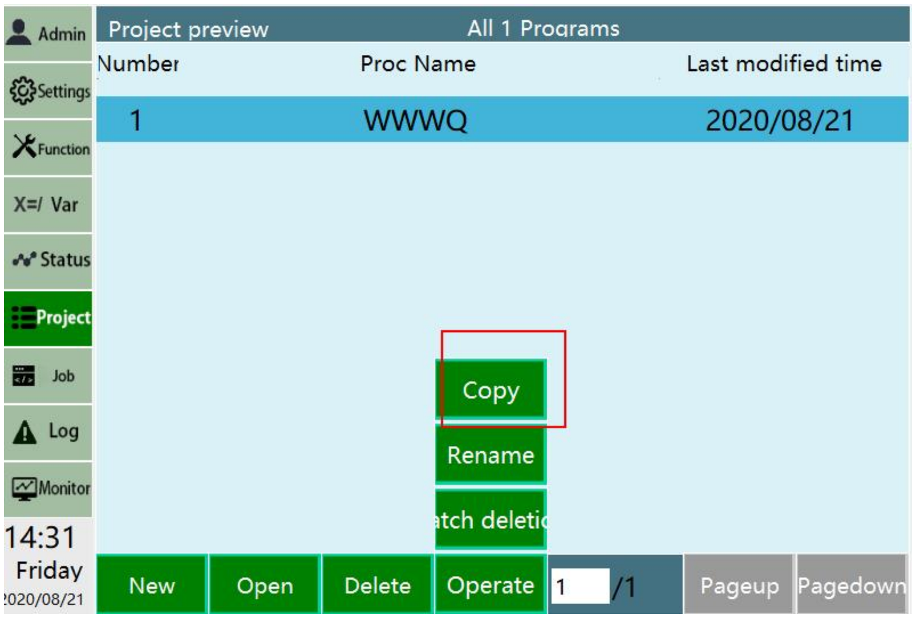
2.Select the program you want to open.

3.Click on the {Open} button at the bottom. The program was successfully opened.

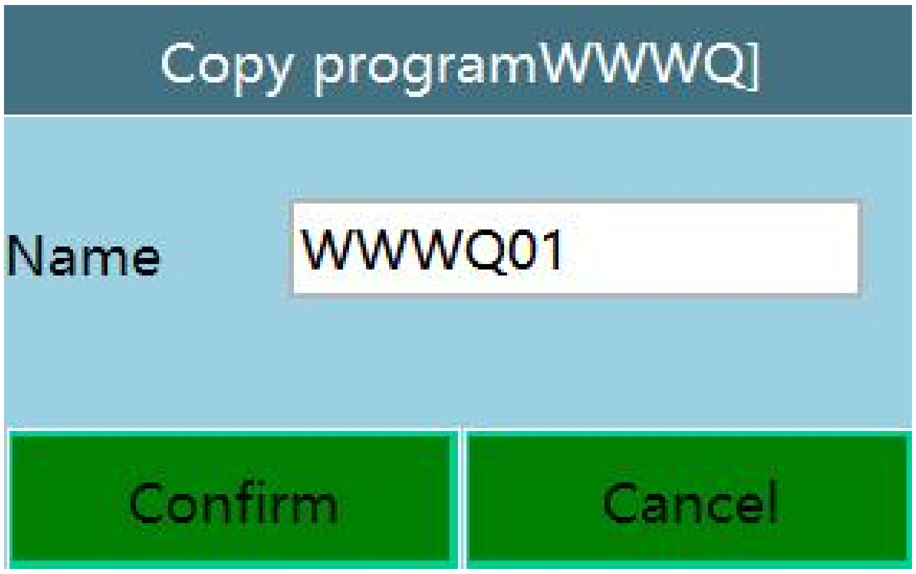
6.3.1.3. Program Copy

To copy an existing job file (which can only be copied as a whole), users need to do the following steps:

1.Select the program to copy;



2.Click on the {Operation} button at the bottom and then click on the {Copy} button.



3.Click on {OK} otherwise {Cancel}; you can also change the name of the file.

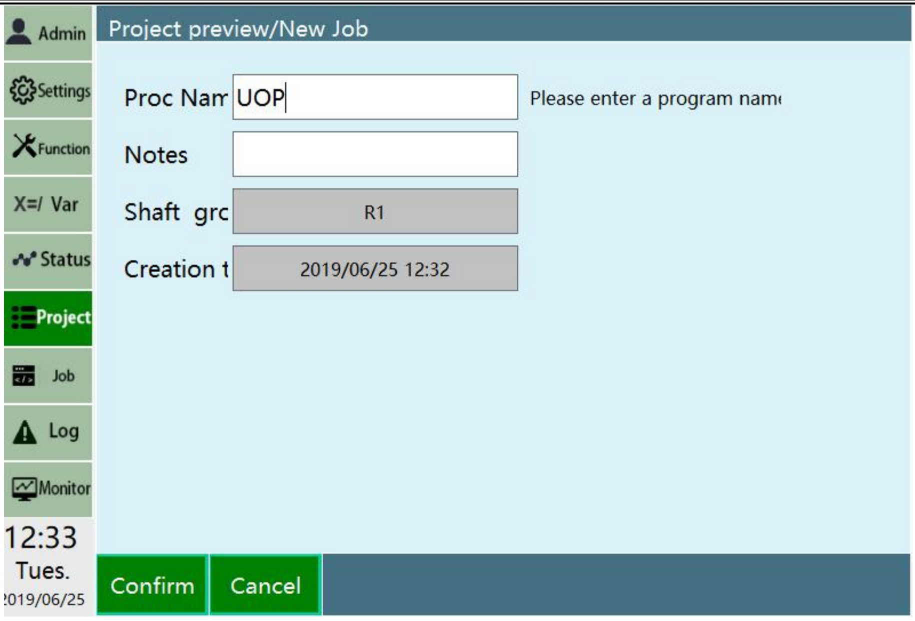
6.3.1.4. Program Rename

The rename operation can modify the name of the selected program. The operation steps are as follows:

1.Select the program you want to rename.

2.Click on {Operation}, and then click on {Rename}.

3.Enter the name you want to modify in the pop-up window.



Project preview/New Job

Proc Nam	UOP	Please enter a program name
Notes		
Shaft grc	R1	
Creation t	2019/06/25 12:32	

4.Click on the {OK} button. If you want to cancel the renaming operation, click on the {Cancel} button.

Caution

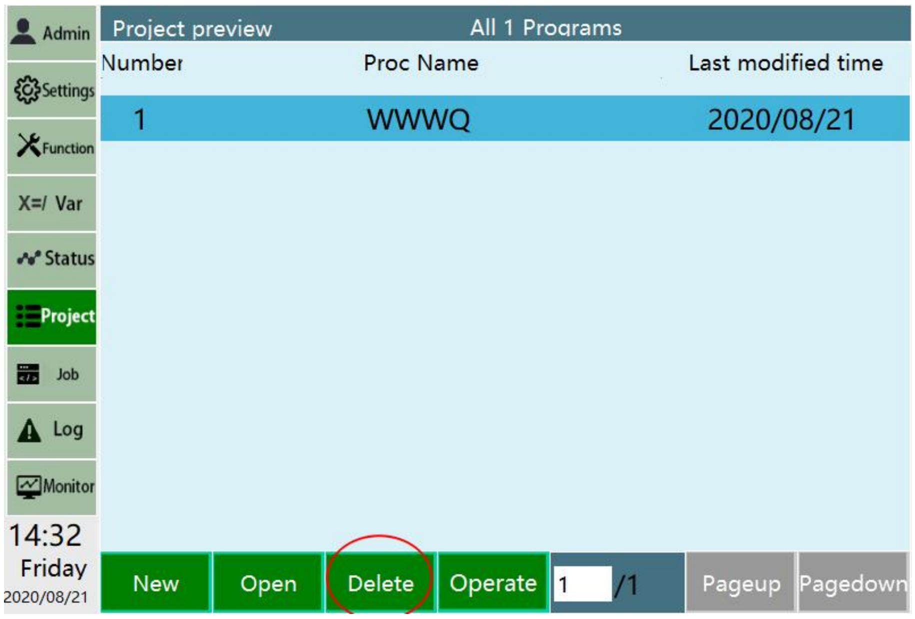
The program name of a renamed program cannot be the name of an existing program.

6.3.1.5. Program Delete

Delete operation can delete the selected program. The relevant steps are as follows:

1.Select the program you want to delete.

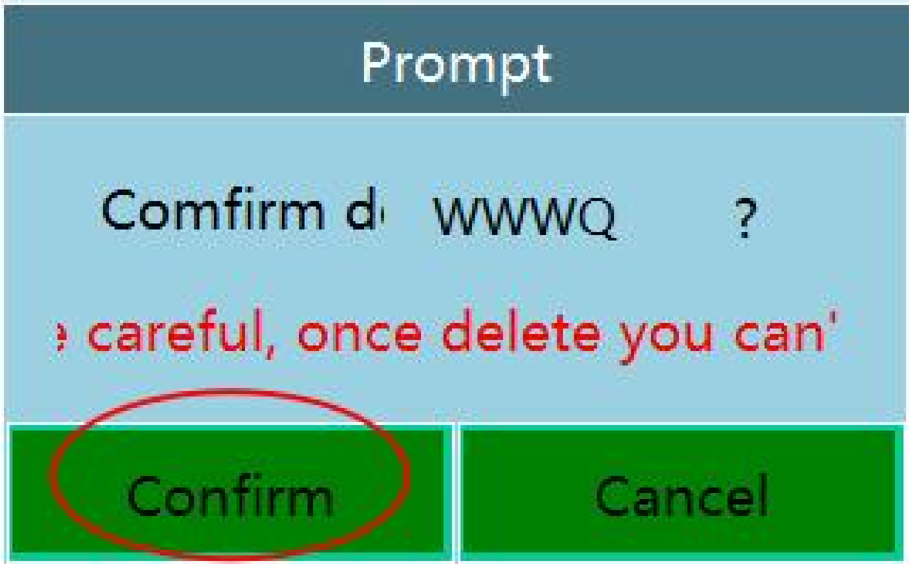
2.Click on the {Delete} button;



Project preview All 1 Programs

Number	Proc Name	Last modified time
1	WWWQ	2020/08/21

3.Click on the {OK} button in the pop-up window. If you want to cancel the deletion operation, click on the {Cancel} button.



Prompt

Comfirm d WWWQ ?

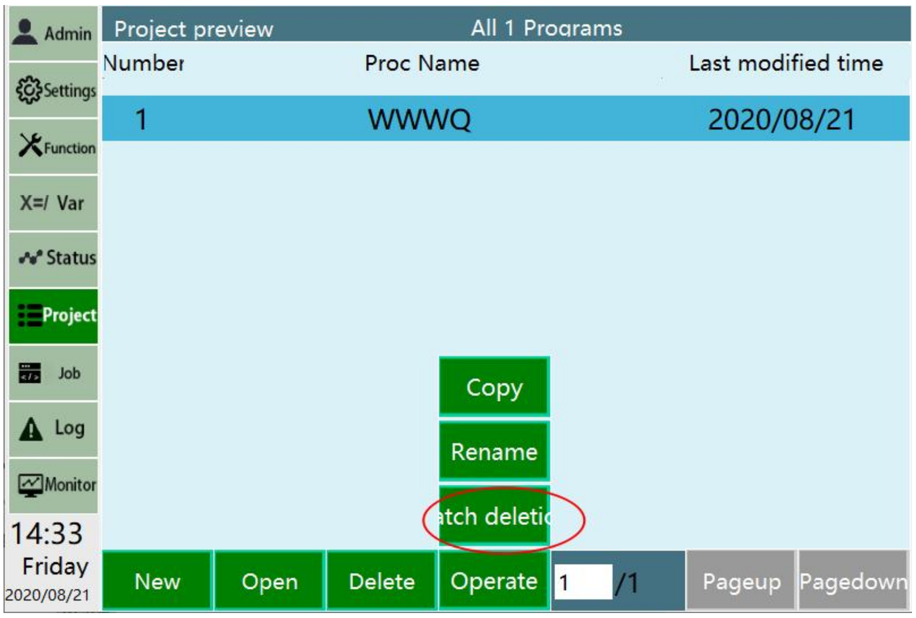
careful, once delete you can'

6.3.1.6. Batch Delete

Batch deletion can delete more than one program file at a time. The method of use is as follows:

1.Enter the engineering interface;

2.Click on the operation of the bottom menu bar - {Batch Deletion} button;

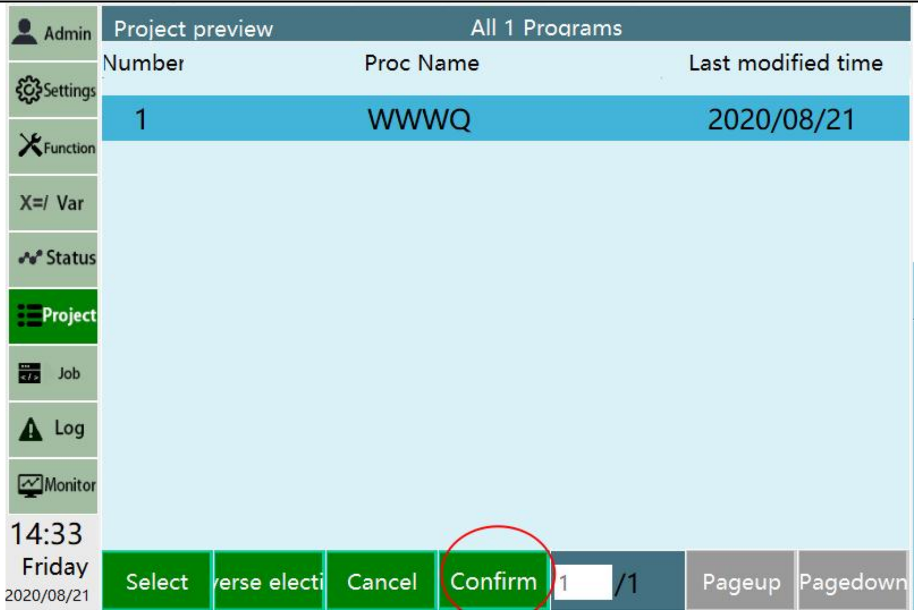


Project preview All 1 Programs

Number	Proc Name	Last modified time
1	WWWQ	2020/08/21

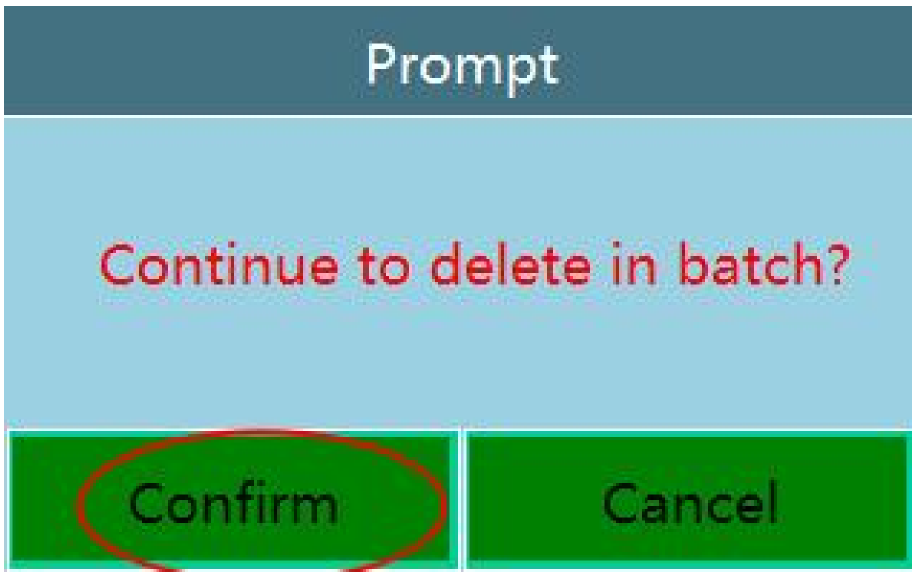
14:33 Friday 2020/08/21

3.Select the program files that need to be deleted (only the files on the current page can be selected, but not on the previous or next page). Click on the {Select All} button to select all the program files on this page;



Number	Proc Name	Last modified time
1	WWWQ	2020/08/21

4.Click on the {Confirmation Button} button and then pop up the confirmation box and click on the {Confirmation} button to delete the batch successfully.



Continue to delete in batch?

6.3.2. Instruction Operation

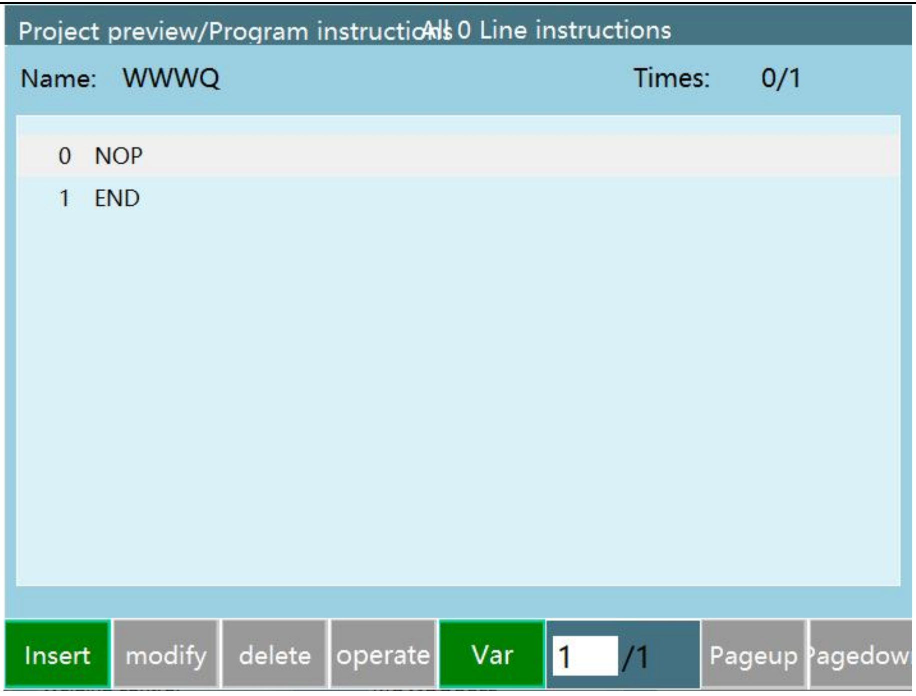
Users need to enter the program preview interface to perform instruction-related operations such as insertion/modification/deletion by using the bottom button.

6.3.2.1. Insertion

The insertion of instructions needs to be operated by using the {Instruction Menu} button at the bottom of the program preview interface.

Inserted instructions are below the selected instruction line The relevant steps are as follows:

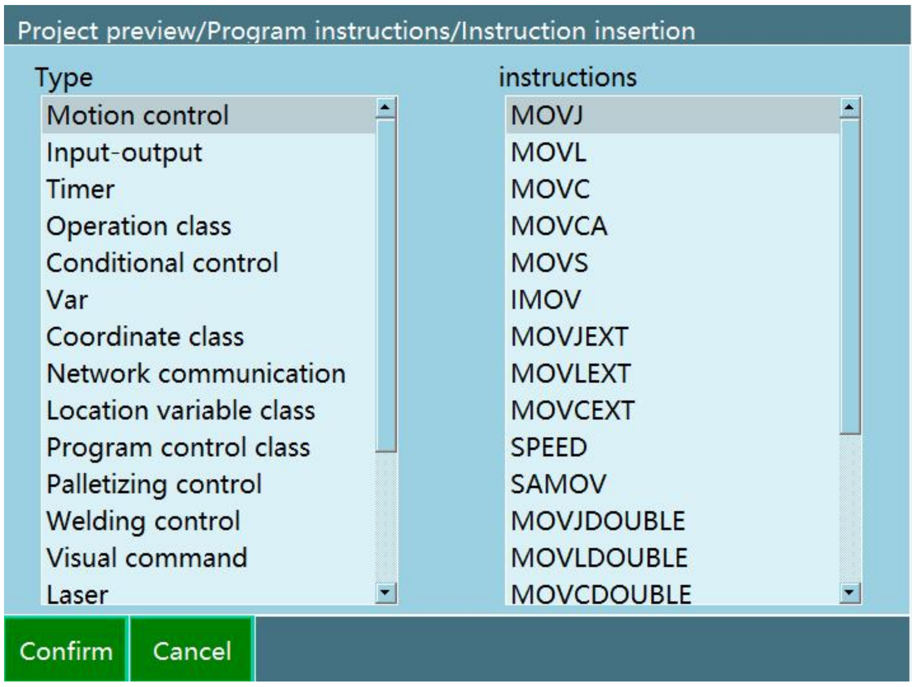
1.Enter the program preview interface;


Project preview/Program instructions All 0 Line instructions

Name: WWWQ Times: 0/1

```
0  NOP
1  END
```

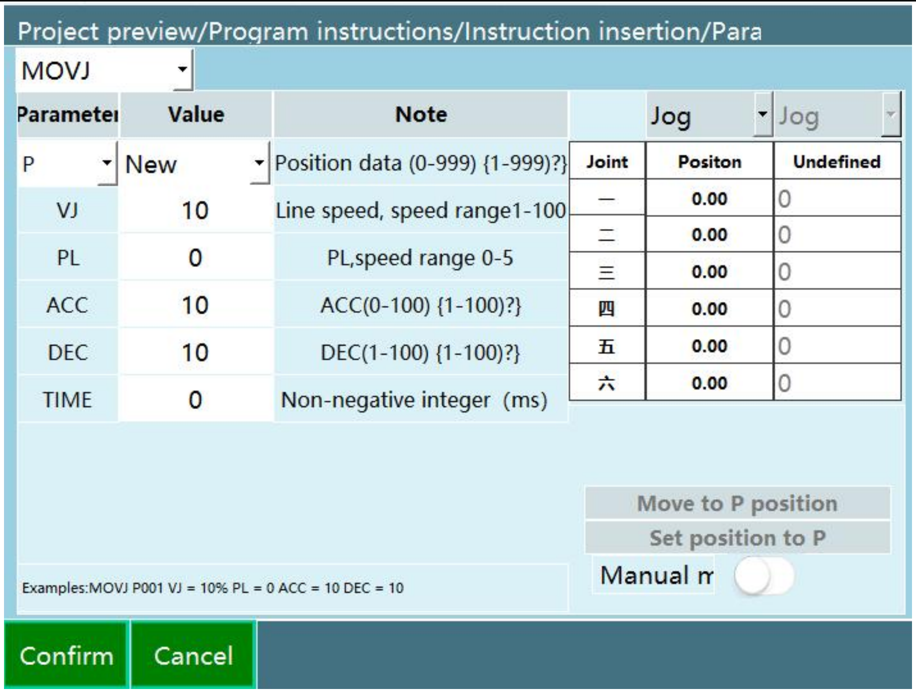
2.Click on the {Insert} button to pop up the instruction type menu;


Project preview/Program instructions/Instruction insertion

Type	instructions
Motion control	MOVJ
Input-output	MOVL
Timer	MOVC
Operation class	MOVCA
Conditional control	MOVS
Var	IMOV
Coordinate class	MOVJEXT
Network communication	MOVLEXT
Location variable class	MOVCEXT
Program control class	SPEED
Palletizing control	SAMOV
Welding control	MOVJDOUBLE
Visual command	MOVLDOUBLE
Laser	MOVCDOUBLE

3.Click on the type of instructions needed to insert instructions, such as motion control classes, as shown in the figure;

4.Click on the instructions you need to insert, such as MOVL, as shown in the figure;


Project preview/Program instructions/Instruction insertion/Para

MOVJ

Parameter	Value	Note
P	New	Position data (0-999) {1-999)?}
VJ	10	Line speed, speed range1-100
PL	0	PL,speed range 0-5
ACC	10	ACC(0-100) {1-100)?}
DEC	10	DEC(1-100) {1-100)?}
TIME	0	Non-negative integer (ms)

Jog Jog

Joint	Positon	Undefined
一	0.00	0
二	0.00	0
三	0.00	0
四	0.00	0
五	0.00	0
六	0.00	0

Examples:MOVJ P001 VJ = 10% PL = 0 ACC = 10 DEC = 10

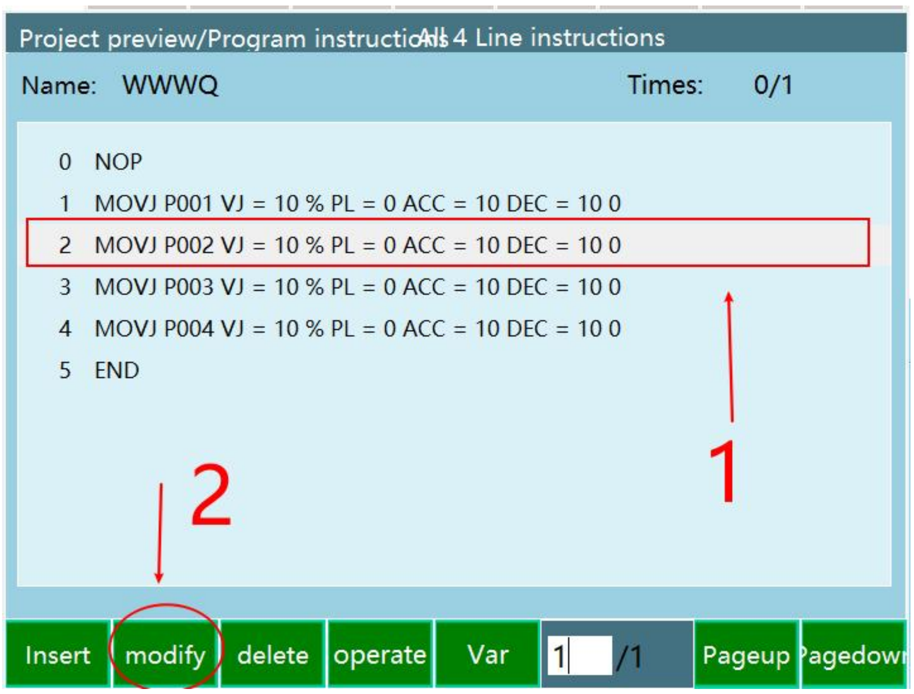
5.Set the relevant parameters of the inserted instructions;

6.Click on the {Confirmation} button at the bottom of the program;

6.3.2.2. Instruction Modification

Users can easily modify the parameters of the inserted instructions by using the "modify" command. The steps to modify the instruction parameters are as follows:

1.Selected inserted rows (except NOP rows and END);

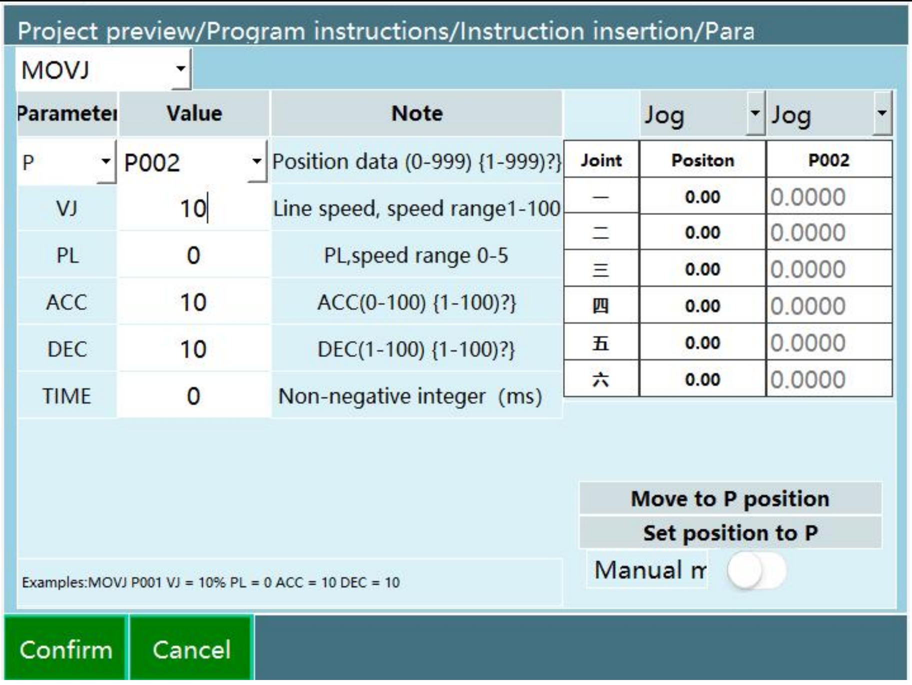

Project preview/Program instructio All 4 Line instructions

Name: WWWQ Times: 0/1

```
0  NOP
1  MOVJ P001 VJ = 10 % PL = 0 ACC = 10 DEC = 10 0
2  MOVJ P002 VJ = 10 % PL = 0 ACC = 10 DEC = 10 0
3  MOVJ P003 VJ = 10 % PL = 0 ACC = 10 DEC = 10 0
4  MOVJ P004 VJ = 10 % PL = 0 ACC = 10 DEC = 10 0
5  END
```

3.Click on the {Modify} button at the bottom of the program.

4.Modify the relevant parameters;


Project preview/Program instructions/Instruction insertion/Para

MOVJ

Parameter	Value	Note
P	P002	Position data (0-999) {1-999)?}
VJ	10	Line speed, speed range1-100
PL	0	PL,speed range 0-5
ACC	10	ACC(0-100) {1-100)?}
DEC	10	DEC(1-100) {1-100)?}
TIME	0	Non-negative integer (ms)

Examples:MOVJ P001 VJ = 10% PL = 0 ACC = 10 DEC = 10

Joint	Positon	P002
一	0.00	0.0000
二	0.00	0.0000
三	0.00	0.0000
四	0.00	0.0000
五	0.00	0.0000
六	0.00	0.0000

5.Click on {OK} button at the bottom when the modification is complete;

6.Successful instruction modification.

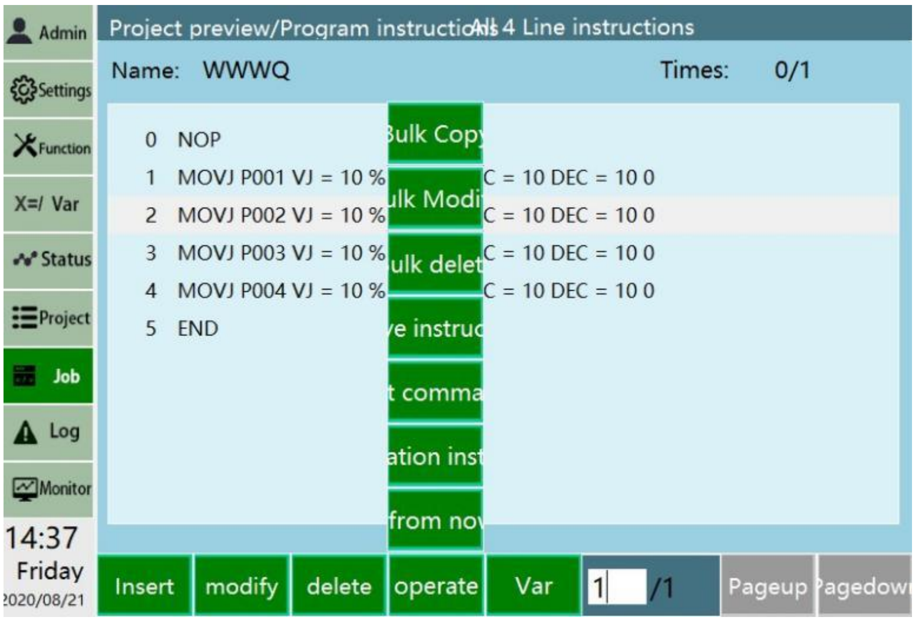
Caution

The program name of a renamed program cannot be the name of an existing program.

6.3.2.3. Batch Copy

Users can replicate the required instructions to the designated place through the "batch copy" operation. The steps are as follows:

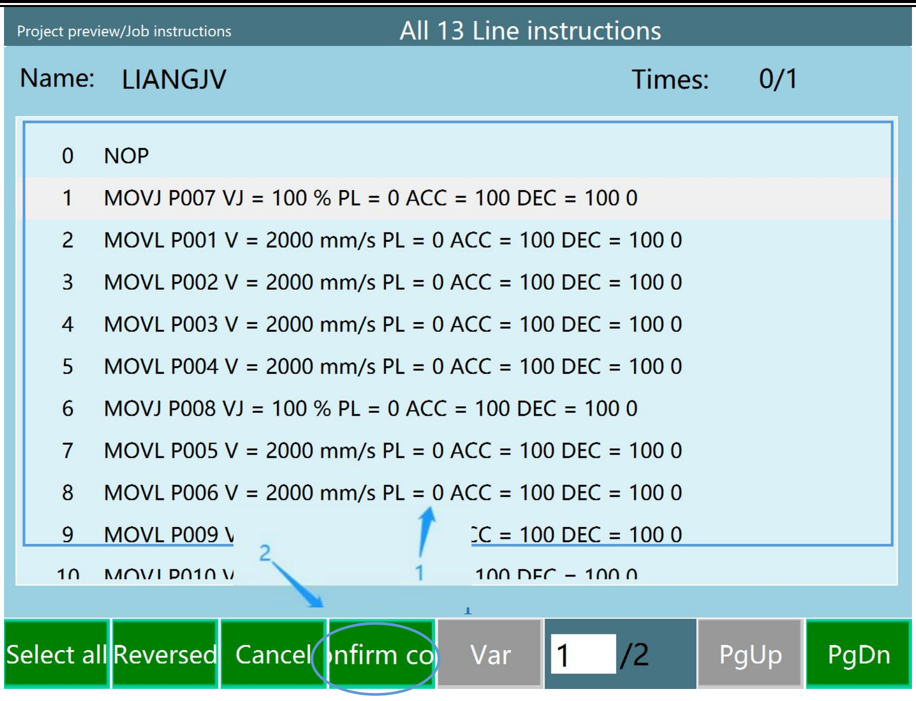
1.First click on the bottom "Operation" button:


Project preview/Program instructions All 4 Line instructions

Name: WWWQ Times: 0/1

```
0  NOP
1  MOVJ P001 VJ = 10 %      C = 10 DEC = 10 0
2  MOVJ P002 VJ = 10 %      C = 10 DEC = 10 0
3  MOVJ P003 VJ = 10 %      C = 10 DEC = 10 0
4  MOVJ P004 VJ = 10 %      C = 10 DEC = 10 0
5  END
```

2. Select the required instructions:

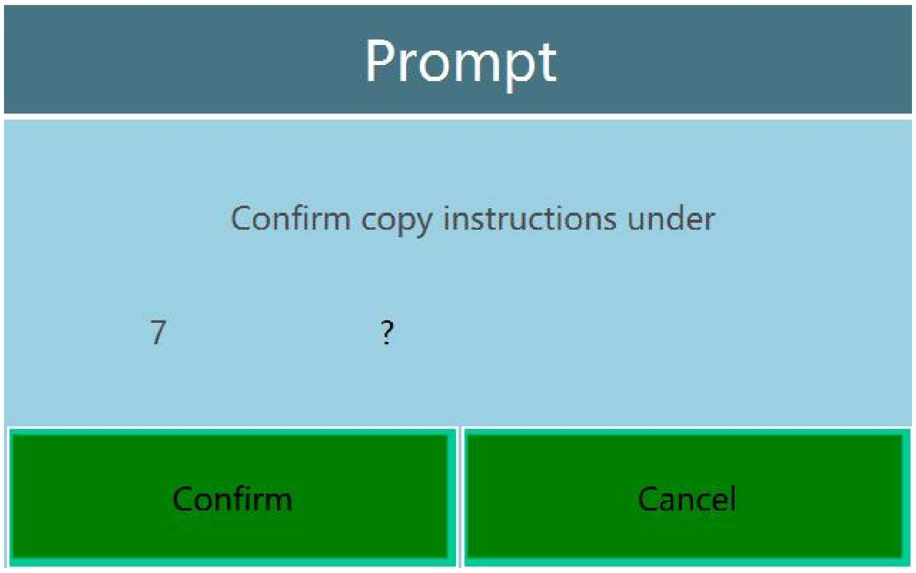


Project preview/Job instructions All 13 Line instructions

Name: LIANGJV Times: 0/1

```
0   NOP
1   MOVJ P007 VJ = 100 % PL = 0 ACC = 100 DEC = 100 0
2   MOVL P001 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
3   MOVL P002 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
4   MOVL P003 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
5   MOVL P004 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
6   MOVJ P008 VJ = 100 % PL = 0 ACC = 100 DEC = 100 0
7   MOVL P005 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
8   MOVL P006 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
9   MOVL P009 V                       C = 100 DEC = 100 0
10  MOVL P010 V                       100 DEC = 100 0
```

3.Click "Confirm Copy" button, pop up the button below, fill in the position you paste;



6.3.3. Instruction Description (Instruction Specification)

This section mainly describes the functions of the instructions and the functions and specifications of the relevant parameters. Provide some examples of specific application scenarios.

6.3.3.1. Motion Control Class

Motion control instructions include MOVJ, MOVL, MOVC, IMOV, NOVCA, MOVJEXT, MOVLEXT, MOVCEXT and other instructions.

When inserting these motion control instructions, if the P-point is selected as the new one, a new P-variable will be created automatically at the same time, and the current robot position will be written into the variable. Running the command runs to the position where the robot inserts the command.

The target points of all motion instructions in this system use position variables, local position variables are P, global position variables are G. See the chapter of position variable for the method of using specific position variable.

The functions and scope of each instruction and related parameters are as follows:

MOVJ

When the robot moves to the target point, it is used in the non-trajectory constrained interval. If the joint is used to interpolate the teaching robot axis, the moving command is MOVJ.

For safety reasons, usually, use joint interpolation to teach the first step. The default speed is VJ = 10, which is the maximum speed of 10\%.

MOVJ	function	Moving to teach position by joint interpolation	
	parameters	position data, base axis position data, tool axis position data	Display in the interface
		VJ= reproduction speed	VJ：1-100
		PL= positioning level	PL：0~5
		NWALL	
		UNTIL	
		ACC= acceleration adjustment ratio	ACC：1-100
		DEC= deceleration adjustment ratio	DEC：1-100
	use example	MOVJ P001 VJ=10% PL=2 ACC=10 DEV=10	

MOVL

Use the linear trajectory to move in the program point of the linear interpolation teach. If linear interpolation is used to teach the robot axis, the mobile command is MOVL.

Linear interpolation is often used in welding operations.

When linear interpolation is used, the wrist attitude of the robot remains unchanged.

MOVL	function	Moving to teach position by joint interpolation	
	parameters	position data, base axis position data, tool axis position data	Display in the interface
		V= reproduction speed	V：2-9999
		PL= positioning level	PL：0~5
		NWALL	
		UNTIL	

		ACC= acceleration adjustment ratio	ACC：1-100
		DEC= deceleration adjustment ratio	DEC：1-100
	use example	MOVL P001 V=100 PL=2 ACC=10 DEV=10	

MOVC

The robot moves through three dotted circles taught by arc interpolation.

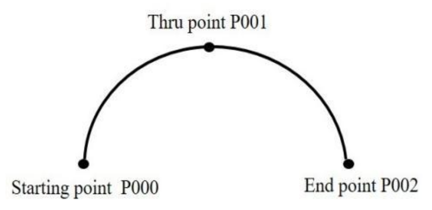
If arc interpolation is used to teach the robot axis, the mobile command is MOVC.

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

■ Single arc

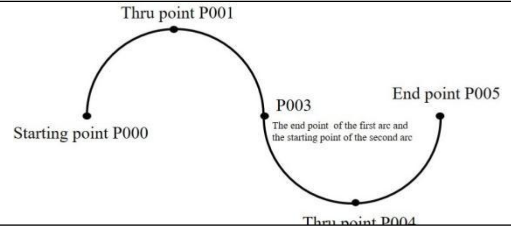
When there is only one arc, as shown in the table below, three points of P1-P3 are taught by arc interpolation.

If P0 before entering the arc is taught by joint interpolation or linear interpolation, the trajectory of P0-P1 will automatically become a straight line.

	point	Interpolation method	command
	P000	joint linear	MOVJ MOVL
	P001-P002	arc	MOVC

■ Continuous arc

As shown in the table below, when there are more than two consecutive arcs whose curvature changes, the arcs will eventually be separated one by one. Therefore, as shown in Figure 4, join the points of joint and linear interpolation at the connection point between the former arc and the latter arc.

	point	Interpolation method	command
 Thru point P001 Starting point P000 P003 The end point of the first arc and the starting point of the second arc End point P005 Thru point P004			
	P000	Joint linear	MOVJ MOVL

	P001-P002	arc	MOVC
	P003-P004	arc	MOVC

MOVC	function	Arc interpolation moves to the target position. The three-point arc method is adopted. The first point before the arc is the first point, and the two MOVCs are the middle point and the target point. Note: The first motion control class instruction of the job file cannot be MOVC.	
	parameters	position data, base axis position data, tool axis position data	Display in The interface
		V= reproduction speed	V：2-9999
		PL= positioning level	PL：0~5
		NWALL	
		UNTIL	
		ACC= acceleration adjustment ratio	ACC：1-100
		DEC= deceleration adjustment ratio	DEC：1-100
	use example	MOVC P001 V=100 PL=2 ACC=10 DEV=10	

IMOV

IMOV	function	Move by the joint position or linear interpolation from the current position according to the set incremental value distance	
	parameters	B= position data	BF：base coordinates RF：robot coordinates TF：tool coordinates UF：user coordinates
		V= reproduction speed	V：2-9999
		PL= positioning level	PL：0~5
		user coordinates	Display B parameter status
		UNTIL	

		ACC= acceleration adjustment ratio	ACC：1-100
		DEC= deceleration adjustment ratio	DEC：1-100
	use example	IMOV B001 V=100 PL=2 ACC=10 DEV=10	

MOVS

In welding, cutting, welding, primer painting and other jobs, if the use of free curve interpolation, for irregular curve workpiece teaching jobs can be easy.

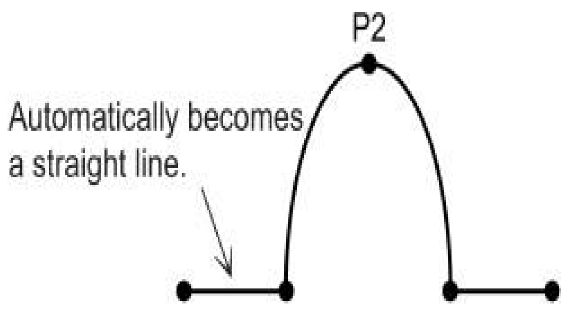
The trajectory is a parabola passing through three points,.

If the free curve is used to interpolate the teaching robot axis, the moving command is MOVS.

■ Single free curve

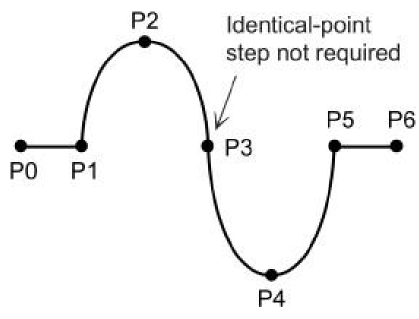
As shown in the table below, the three points of teaching P1-P3 are interpolated with free curve.

If joint interpolation or linear interpolation is used to teach the P0 point before entering the free curve, the trajectory of P-P1 will automatically become a straight line.

	point	Interpolation method	command
	P000	Joint linear	MOVJ MOVL
	P001-P003	free curve	MOVS
	P004	Joint linear	MOVJ MOVL

■ Continuous free curve

The trajectory is established by coincidence parabola synthesis.。

	point	Interpolation method	command
 P2 Identical-point step not required P5 P6 P3 P0 P1 P4	P000	Joint linear	MOVJ MOVL
	P001-P005	free curve	MOVS
	P006	joint	MOVJ MOVL

Unlike arc interpolation, the joint of two free curves cannot be the same or have no other instructions.

Establishment of synthetic trajectory in the case of coincidence parabola.

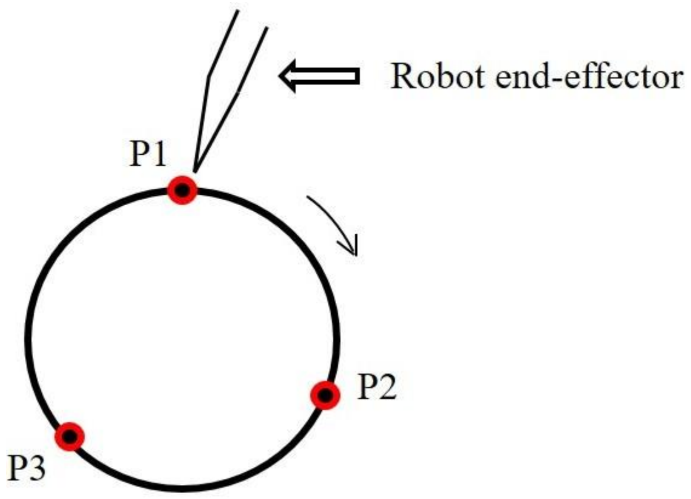
MOVS	function	Move to the teaching position in the form of free curve

		interpolation.	
	parameters	position data, base axis position data, tool axis position data	Display in the interface
		V= reproduction speed	V：2-9999
		PL= positioning level	PL：0~5
		NWALL	
		UNTIL	
		ACC= acceleration adjustment ratio	ACC：1-100
		DEC= deceleration adjustment ratio	DEC：1-100
	use example	MOVS P001 V=100 PL=2 ACC=10 DEV=10	

MOVCA

To teach the robot to walk a complete circle, the mobile command is MOVCA. Instruction insertion premise

Click on the {Tool} button in the status bar above and select the tool that has been calibrated before.



Insertion steps, a total of four instructions.

{movca}Click on the {Insert}, click on the {Coordinate Switching Class}, select SWITCHTOOL, and select the tool number previously calibrated.

Move to any point of the circle you want to draw as shown in Figure P1, click on the {Insert}, click on the {Motion Control Class}, and select {movj} or {movl};

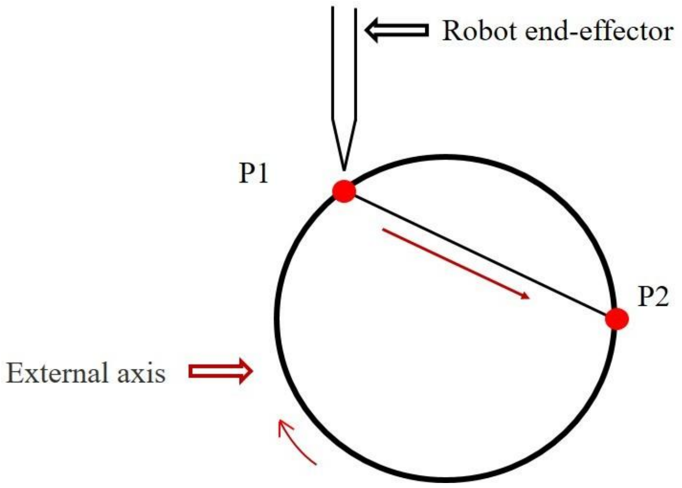
Move to any point of the circle you want to draw as shown in Figure P2 (unlike the point in Step 2). Click on the {Coordinate System} button in the upper status bar, select the "Tool" coordinate system, click on the {Insert}, click on the {Motion Control Class}, and select {movca}.

Move to any point of the circle you want to draw as shown in Figure P3 (unlike the point in Step 2 or 3). Click on the {Coordinate System} button in the upper status bar, select the "Tool" coordinate system, click on {Insert}, click on {Motion Control Class}, and select .

MOVCA	function	Based on the principle of determining a circle by three points, draw a circle. The three-point circle drawing method is adopted. The first point is in front of the circle and the two MOVCAs are in the middle of the circle. Note: The first motion control class instruction of the job file cannot be MOVCA.	
	parameters	position data, base axis position data, tool axis position data	Display in the interface
		V= reproduction speed	V：2- 9999
		PL= positioning level	PL：0~5
		NWALL	
		UNTIL	
		ACC= acceleration adjustment ratio	ACC：1-100
		DEC= deceleration adjustment ratio	DEC：1-100
	use example	MOVCA P001 V=100 PL=2 ACC=10 DEV=10	

MOVJEXT

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

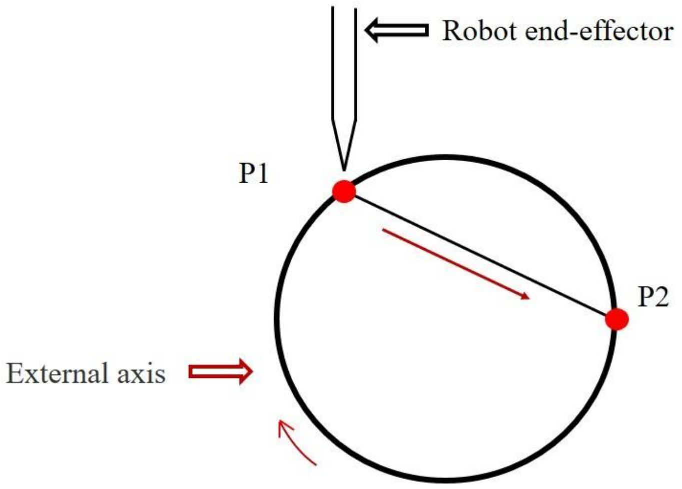


MOVJEXT	function	The robot moves to the teaching position by means of joint interpolation,and the external axis is compensated by joint

		difference.	
	parameters	position data, base axis position data, tool axis position data	Display in the interface
		VJ= reproduction speed	VJ：1-100
		PL= positioning level	PL：0~5
		NWALL	
		UNTIL	
		ACC= acceleration adjustment ratio	ACC：1-100
		DEC= deceleration adjustment ratio	DEC：1-100
	use example	MOVJEXT P001 VJ=10% PL=2 ACC=10 DEV=10	

MOVLEXT

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

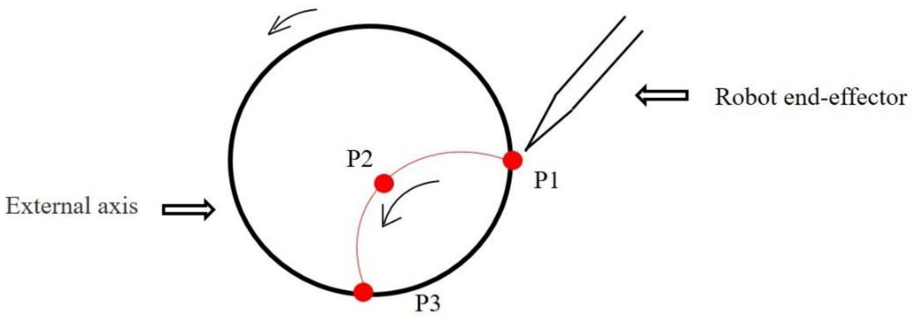


MOVLEXT	function	The robot moves to the teaching position by arc interpolation, and the external axis is compensated by joint difference.	
	parameters	position data, base axis position data, tool axis position data	Display in the interface
		V= reproduction speed	V：2- 9999
		PL= positioning level	PL：0~5
		NWALL	
		UNTIL	

		ACC= acceleration adjustment ratio	ACC:1-100
		DEC= deceleration adjustment ratio	DE：1-100
	Use example	MOVL P001 V=100 PL=2 ACC=10 DEV=10	

MOVCEXT

The robot moves to the teaching position by arc interpolation, and the external axis is compensated by joint difference.



MOVCEXT	function	The robot moves to the teaching position by arc interpolation, and the external axis is compensated by joint difference.	
	parameters	position data, base axis position data, tool axis position data	Display in the interface
		V= reproduction speed	V：2-9999
		PL= positioning level	PL：0~5
		NWALL	
		UNTIL	
		ACC= acceleration adjustment ratio	ACC：1-100
		DEC= deceleration adjustment ratio	DEC：1-100
	use example	MOVCEXT P001 V=100 PL=2 ACC=10 DEV=10	

SAMOV

Robots move to a set absolute position by joint interpolation.

Caution

If you don't want to move an axis, leave it blank at its coordinates. (Do not fill in 0)

SAMOV	function	Robots move to a set absolute position by joint interpolation.	
	parameters	B=position data	BF：base coordinates RF：robot coordinates TF：tool coordinates UF：user coordinates
		V= reproduction speed	V：2-9999
		PL= positioning level	PL：0~5
		user coordinates	Display B parameter status
		UNTIL	
		ACC= acceleration adjustment ratio	ACC：1-100
		DEC= deceleration adjustment ratio	DEC：1-100
	Use example	SAMOV B001 V=100 PL=2 ACC=10 DEV=10	

SPEED

The movement speed of all the motion instructions below the SPEED instruction is as follows: the instruction speed * the speed of the upper status bar * the percentage of SPEED.

SPEED	function	setting global speed	
	parameters	globalspeed（%）	speed percentage：1-200
	use example	SPEED 200	

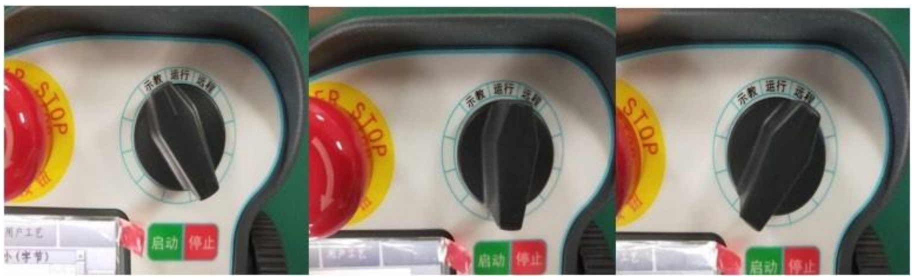
6.3.3.2. Timer class

TIMER timing

TIMER	function	delay	
	parameters	time	0-9999s
	use example	TIMER=100s	

6.4. Program running

The program can be run in three mode states, including "Teach", "Run" and "remote", corresponding to "teach", "Run" and "remote mode" respectively.The user can switch between the three modes by using the mode select key in the upper right corner of the instructor.



6.4.1. Teaching mode

In the teaching mode, the robot can complete the point operation, job file programming, system parameter setting and other operations.In the process of job file programming can use the "STEP" function to the job file for a single step operation.

6.4.1.1. Use STEP for track confirmation

After selecting the inserted instruction line, the user can press the "DEADMAN" button and click the "STEP" (single step) button in the physical key area at the bottom of the indicator to carry out a single step operation on the job file in the program (do not release the "DEADMAN" button during the robot movement). The single step operation can only run the selected instruction line.

STEP Running speed = command speed * Speed ratio of the upper status bar.The specific steps are as follows:

1. Select the command line for which you want to step.
2. Press the DEADMAN button to power on the robot.
3. Press the [STEP] button, and the robot will execute the command of the selected line and stop after the execution.
4. Select the line to move down automatically, and press the [STEP] button again if you want to run the next line of instructions in a single step.

6.4.2. Operating mode

In the running mode, you can click the "Run times" button in the lower left corner to set the running times of the program.The default is to run once.Click the "loop Run" button in the pop-up box to make the program run in an infinite loop.

In running mode, the number of run times and the total number of set run times are displayed above the program, in the format of "number of run times/number of set run times".

During operation, the number of runs can be modified. After modification, the robot will stop after the number of runs set.For example, if the original set of 200 runs has been run 156 times, and the number of runs is set to 3 times, the robot will stop after continuing to run for 3 times.

Run speed = command speed * Speed ratio of the upper status bar.

6.4.3. Remote mode

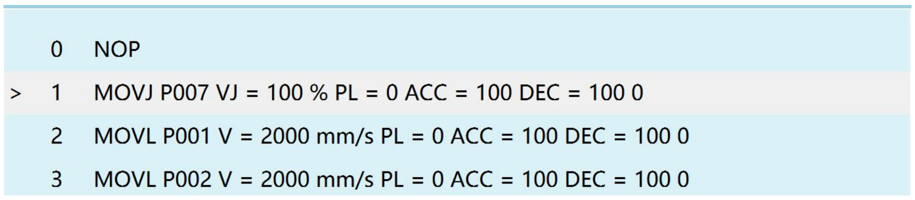
Remote mode supports two types of external devices, digital IO and Modbus touch screens.

The device priority is Modbus> Digital IO. When two external devices are connected, you can enable digital IO by using the Modbus touch screen.

When the demonstrator is unplugged, the remote IO signal is triggered and the remote mode will be automatically entered.

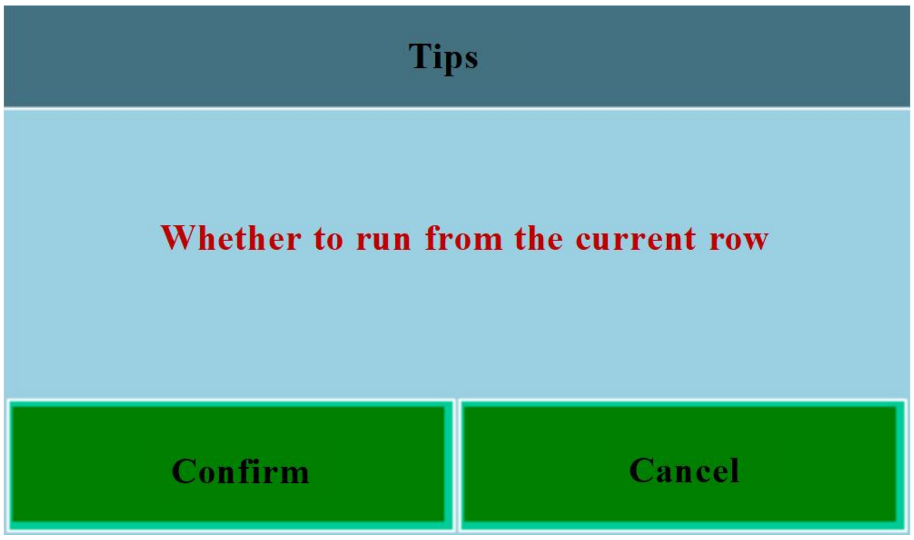
6.4.4. Run from the current row

In the teaching mode, open the job file, select a line, click the operation button, click "Run from", and the ">" symbol will appear in the job file;



```
    0   NOP
>   1   MOVJ P007 VJ = 100 % PL = 0 ACC = 100 DEC = 100 0
    2   MOVL P001 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
    3   MOVL P002 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
```

Switch to Run mode, click Start run will prompt pop-up window.



Tips

Whether to run from the current row

Click the Confirm button to run from the selected row, click Cancel to run from the first row.

6.4.5. Breakpoint run

6.4.5.1. Run mode breakpoint

During operation (except for the first instruction), the operation will be interrupted when switching to other modes, and the variable state and program running position at the time of interruption will be saved as a breakpoint. When running again, a prompt box will pop up asking "whether to continue running the current program". If you select "OK", you will continue running from the breakpoint; if you select "Cancel", the breakpoint will disappear and start running again from the first instruction.

Viewing the status of a breakpoint: After the breakpoint is switched to the teaching mode, you can power on the breakpoint to view the position/value variable status.

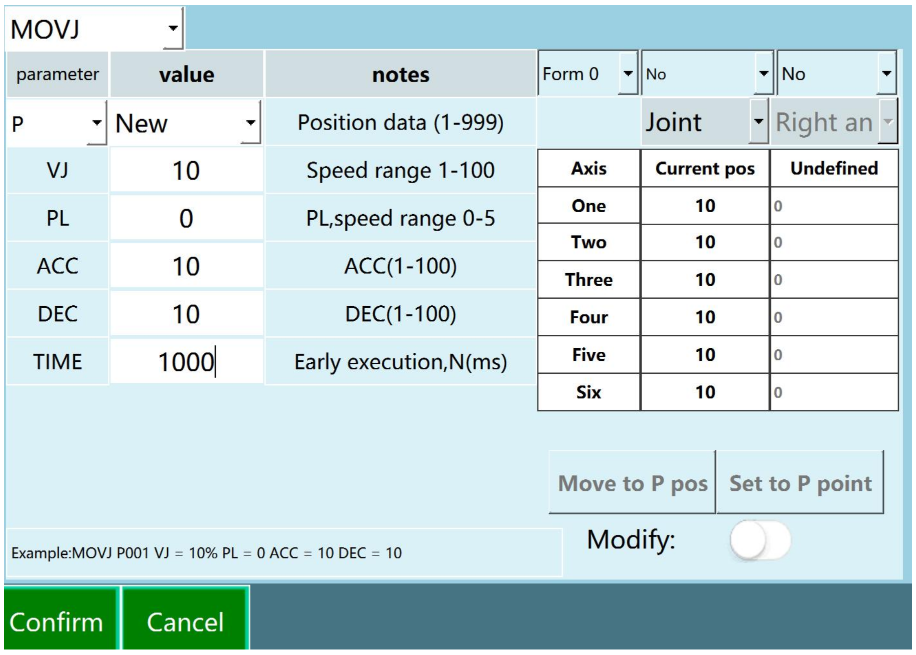
For example, the initial state of P001 and I001 is shown in the figure. During running, P001J1+1 and I001+1 are changed.

At the sixth line, P001J1=1 and I001=2, break point is generated when switching to the teaching mode. After switching to the teaching mode, check that P001 and I001 are displayed as initial values. At this time, press "DEADMAN" to power on, and P001J1=1 and I001=2 are displayed.

Breakpoint release: After the breakpoint is generated, return to zero, reset, run a single step command, run other programs, run to this point, select "Cancel" in the pop-up box, insert/delete/move/cut/copy instructions, modify local value/local position variables, restart the controller, modify robot parameters and other operations will remove the breakpoint, and then run the program from the first instruction.

6.4.5.2. Teaching mode break point

Breakpoints exist in the teaching mode. If there is a command to change local variables during a single-step program, you can power off the program and then power it on to view the local variable value at breakpoint.



parameter	value	notes
P	New	Position data (1-999)
VJ	10	Speed range 1-100
PL	0	PL,speed range 0-5
ACC	10	ACC(1-100)
DEC	10	DEC(1-100)
TIME	1000	Early execution,N(ms)

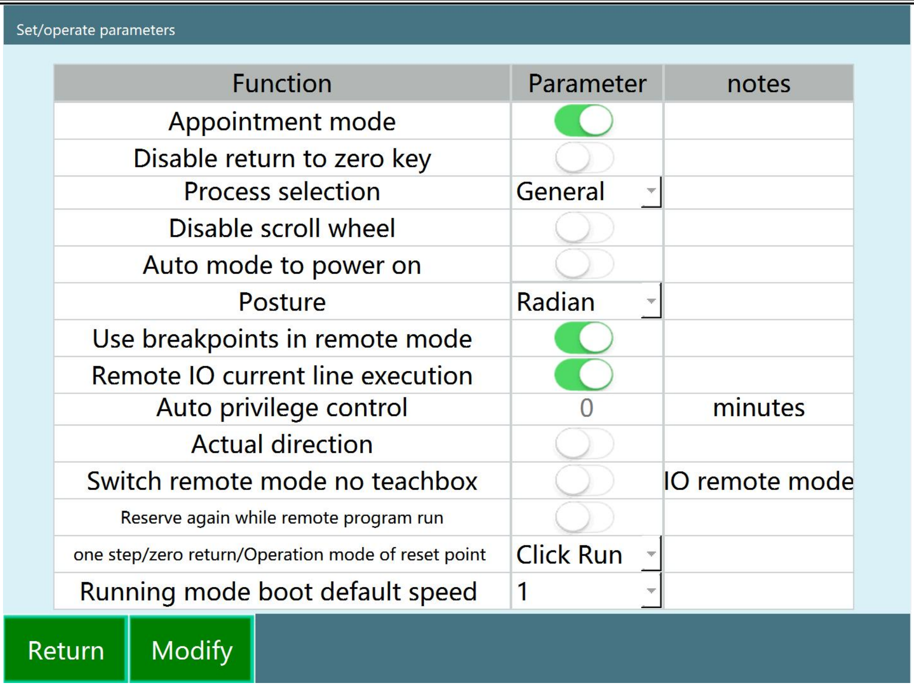
Axis	Current pos	Undefined
One	10	0
Two	10	0
Three	10	0
Four	10	0
Five	10	0
Six	10	0

Example:MOVJ P001 VJ = 10% PL = 0 ACC = 10 DEC = 10

Return to zero, reset, power off in single step instruction, run other programs, run to this point, modify local value/local position variable and single step instruction, restart the controller, modify robot parameters and other operations will lift the "breakpoint".

6.4.5.3. Remote mode breakpoint

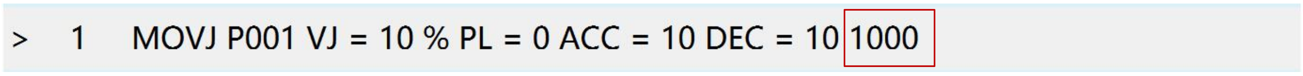
Use the IO reservation program to execute breakpoints by default. If remote breakpoints are not needed, disable them in Settings - Operation Parameters - Remote Mode Whether to use breakpoints


Set/operate parameters

Function	Parameter	notes
Appointment mode	✓	
Disable return to zero key		
Process selection	General	
Disable scroll wheel		
Auto mode to power on		
Posture	Radian	
Use breakpoints in remote mode	✓	
Remote IO current line execution	✓	
Auto privilege control	0	minutes
Actual direction		
Switch remote mode no teachbox		IO remote mode
Reserve again while remote program run		
one step/zero return/Operation mode of reset point	Click Run	
Running mode boot default speed	1	

6.4.6. Advanced executive function

The motion command takes effect when the time parameter is set. The parameter point is ms.



```
>   1    MOVJ P001 VJ = 10 % PL = 0 ACC = 10 DEC = 10 1000
```

In the example program above, the MOVJ instruction is followed by the DOUT instruction;If the TIME parameter of MOVJ instruction is set to 1000ms, the next instruction will be executed 1s in advance. For example, if MOVJ instruction executes 3s, the MOVJ instruction will run for 2 seconds to execute DOUT and continue to execute MOVJ to P001.

6.5. Robot motion speed

The speed of teaching mode, operation mode and remote mode of the system should be set separately

6.5.1. Teaching mode speed

Kinematic speed of joint joint = maximum kinematic speed of joint axis * teaching speed (maximum speed limit = maximum kinematic speed of joint axis *50);

Right-angle pointing speed = maximum pointing speed of right-angle coordinates * teaching speed (maximum speed limit =300mm/s);

Zero return speed = rated speed * teaching speed *10 (maximum speed limit = rated speed *10);

Joint return safety point speed = rated speed * teaching speed *10 (maximum speed limit = rated speed *10);

Speed of straight line return to safety point =100mm/s* Teaching speed (maximum speed limit =100mm/s);

Speed of movement to the point = teaching speed * teaching speed;

Single step joint speed = rated speed * teaching speed * command speed (maximum speed limit = rated speed *30);

One-step right-angle speed = teaching speed * instruction speed (maximum speed limit =300mm/s);

6.5.2. Operating mode speed

Running point-to-point speed = rated speed * Running speed * command speed

Running straight-line speed = running speed * command speed

6.5.3. Remote mode speed

Remote point-to-point speed = rated speed * remote speed * command speed

Remote linear speed = remote speed * command speed

6.5.4. Remote I/O speed change mode

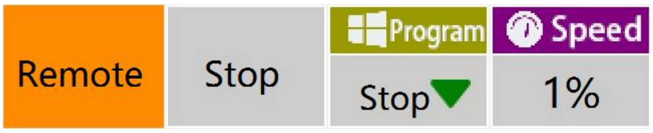
⚠ Caution

The remote mode teaching box does not allow you to change the speed. You need to set it in advance in the teaching mode. The default remote speed is 15

1. Go to the Settings - Robot Parameters - Motion parameters interface.

2. Click Modify to modify the speed of the remote mode, click Save, and switch to the running mode.

3. The modification is successful.



7. Tool and User Coordinates

7.1. Tool Calibration

7.1.1. Tool coordinate system

Flange Center: The origin of the default tool coordinate system, flange center point to flange positioning hole direction is + X direction, vertical flange outward direction is + Z direction, finally according to the right hand rule can determine the Y direction. The new tool coordinates are derived from the relative default tool coordinates.

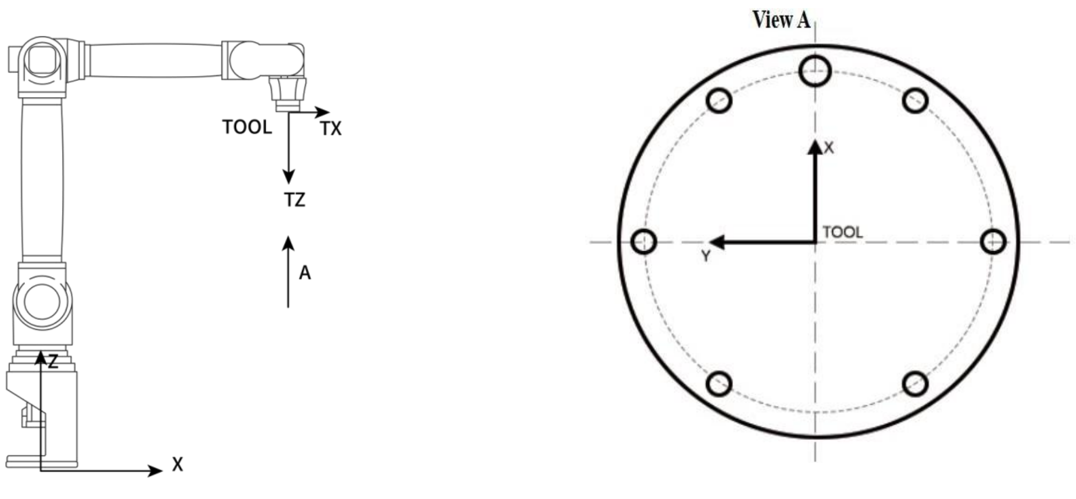


Figure 7.1 Tool coordinate system and flange

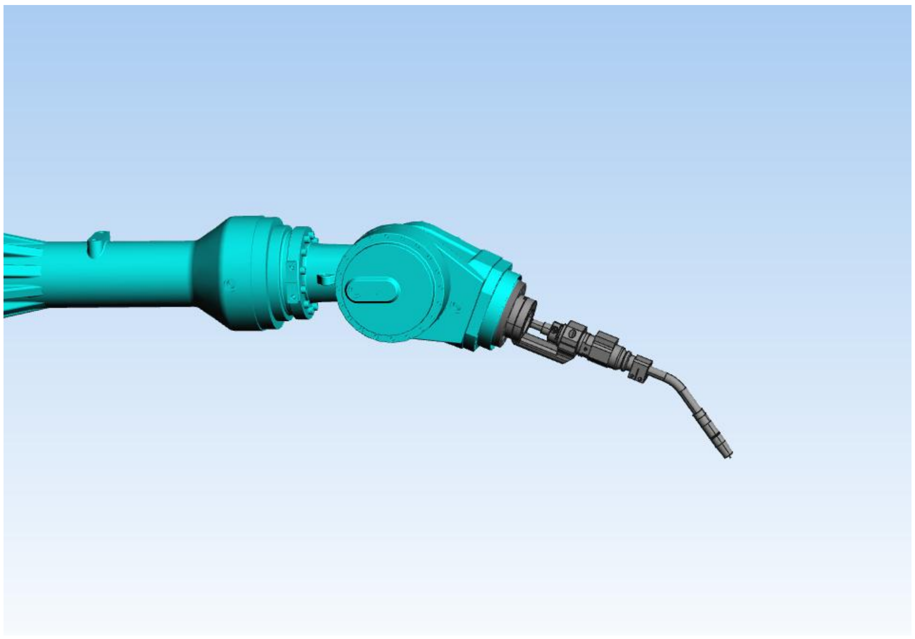
TCP:TOOL CENTER PO INT,the tool center point

Robot trajectory and speed: refers to the trajectory and speed of the TCP point.

TCP is generally set in the center of the hand claw, the end of the wire, the front end of the spot welding static arm and so on.

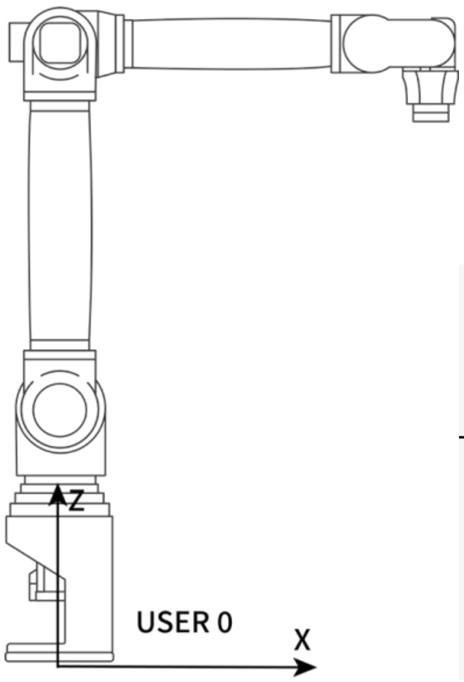
In order to describe the position of an object in space, it is necessary to fix a coordinate system on the object, and then determine the pose of the coordinate system (origin position and three axis pose), that is, 7 DOF are needed to completely describe the pose of the rigid body [1].For industrial robots, a Tool is installed on the end flange to operate.In order to determine the pose of the Tool, a Tool coordinate TCS(ToolCoordinateSystem) is bound to the tool, and the origin of TCS is TCP (ToolCenterPoint).In the trajectory programming of robot, it is necessary to record the pose of TCS in other coordinate systems into the program for execution.

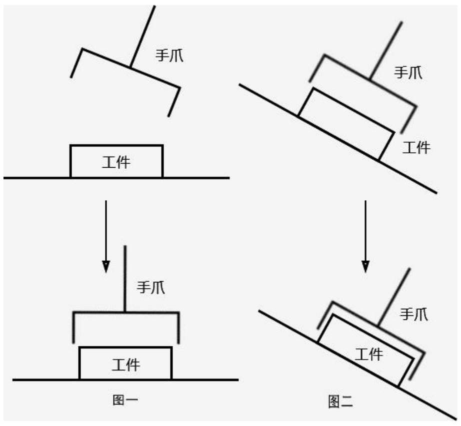
Industrial robots generally define a TCS in advance. The XY plane of TCS is bound on the flange plane of the sixth axis of the robot, and the origin of TCS coincides with the center of the flange.Clearly TCP is in the center of the flange disk.The ABB robot calls TCP tool0 and the REIS robot _tnull.Although the default TCP can be used directly, in practical use, such as welding, the user usually defines the TCP point to the tip of the wire (in fact, the position of the welding gun tool coordinate system in the tool0 coordinate system), then the recorded position in the program is the position of the tip of the wire, and the recorded attitude is the attitude of the welding gun rotating around the tip of the wire.



think

From Thinking 1, we know that the tool coordinate system is a research object in motion, but what role does it play in the actual debugging process?Think about how to adjust the posture and position of the claw in Figure 1 and Figure 2.





Two conjectures can be made from consideration:

Hypothesis 1: If the claw in Figure 1 has a rotation point, make the claw directly around the rotation point selection is OK.

Conjecture 2: If there is a claw in Figure 2, it can be moved directly.

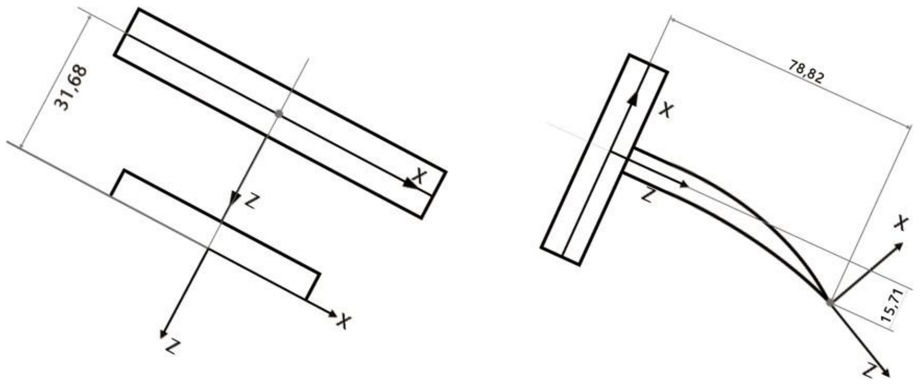
Conclusion: The function of establishing tool coordinate system:

1. Establish the TCP point of the tool (i.e., the center point of the tool) to facilitate the adjustment of the tool state.

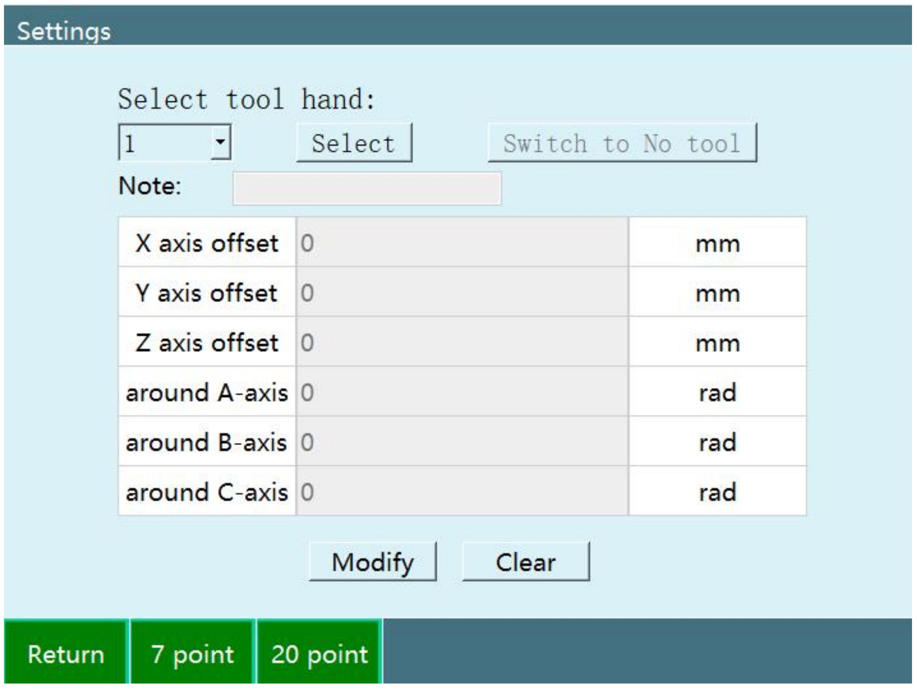
2. determine the tool feed direction, convenient tool position adjustment.

7.1.2. Characteristics of tool coordinate system

The new tool coordinate system is obtained from the change of the default tool coordinate system. The position and direction of the new tool coordinate system always maintain the absolute position and attitude relationship with the flange, but it is always changing in space.



Click "Tool Hand calibration" in Settings to enter the interface of tool hand calibration, as shown in the figure


Settings

Select tool hand:

1 Select Switch to No tool

Note:

X axis offset	0	mm
Y axis offset	0	mm
Z axis offset	0	mm
around A-axis	0	rad
around B-axis	0	rad
around C-axis	0	rad

If there are detailed parameters of the tool, in this interface, users can directly fill in the relevant parameters of the tool end offset without seven-point calibration.

When entering this interface, the size parameters of the tool saved in the controller will be read automatically (default items are 0). If you change the tool hand, please fill them in again.

Detailed parameter setting steps are as follows:

1.Open the tool calibration interface, the following table is an introduction to each parameter:

parameters	action
X-axis direction migration	The migration length (mm) of the tool end relative to the center of the flange along the Cartesian coordinate system X-axis
Y-axis direction migration	The migration length (mm) of the tool end relative to the center of the flange along the Cartesian coordinate system Y-axis
Z-axis direction migration	The migration length (mm) of the tool end relative to the center of the flange along the Cartesian coordinate system Z-axis
Migration about axis A	The migration angle (°) of the tool end relative to the center of the flange around the X-axis of the Cartesian coordinate system
Migration about axis B	The migration angle (°) of the tool end relative to the center of the flange around the Y-axis of the Cartesian coordinate system
Migration about axis C	The migration angle (°) of the tool end relative to the center of the flange around the Z-axis of the Cartesian coordinate system

2.Click on the {Modify} button.

3.Fill in the parameters corresponding to the tool, in which the functions of the parameters are shown in the table above.

4.After be sure that it is correct, click on the {Save} button and and the setting is successful.

⚠ Caution

Place flange parallel to horizontal plane before data collection.

Click the "Clear" button to clear the filled parameter.

If you click the "Back" or "Seven Point calibration" button at the bottom of the operation area during parameter setting, you will jump to the corresponding interface, and the unsaved setting parameters will not be retained.

7.1.3. Six-point calibration

Six-point method calibration steps:

First point: Robot axis 5 is straight down

Second point: The robot rotates the C-axis 180° based on the first point

Third point: the B-axis Angle of robot is 35°

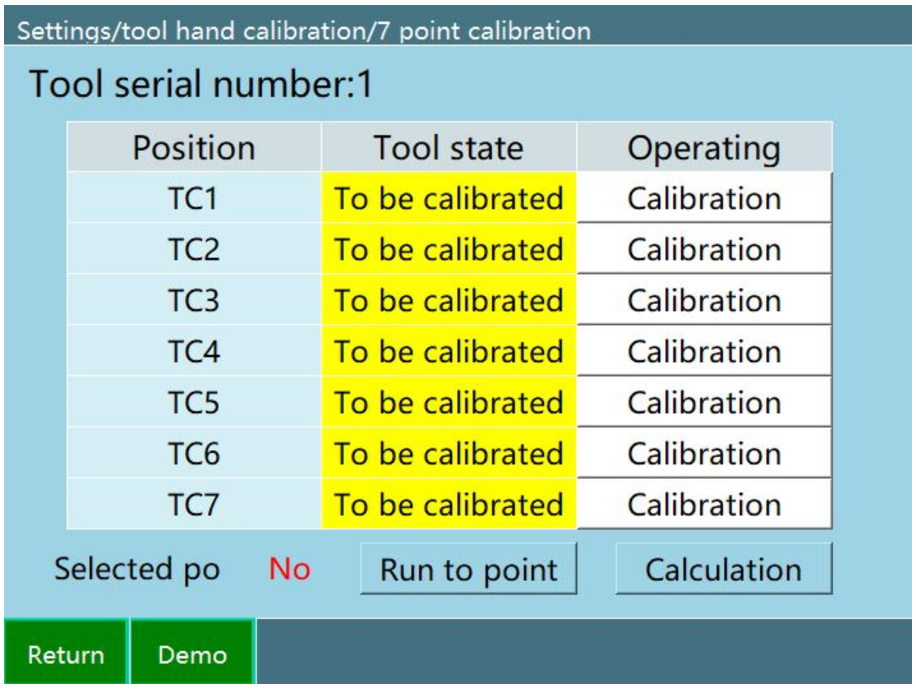
Fourth point: The robot returns to zero, and then the tool tip is vertical

Fifth point: The robot moves -x based on fourth point

Sixth point: The robot moves +Y based on fifth point

7.1.4. Seven-point calibration

Click on the {Seven-Point Calibration} button at the bottom to enter the seven-point calibration interface, as shown in the figure.



Settings/tool hand calibration/7 point calibration

Tool serial number:1

Position	Tool state	Operating
TC1	To be calibrated	Calibration
TC2	To be calibrated	Calibration
TC3	To be calibrated	Calibration
TC4	To be calibrated	Calibration
TC5	To be calibrated	Calibration
TC6	To be calibrated	Calibration
TC7	To be calibrated	Calibration

Selected po No

Without the detailed parameters of the tool, TCP calibration can be carried out, and the size parameters of the tool can be calculated automatically. The specific calibration steps are as follows:

1. First point: Place the end of the tool perpendicular and facing the reference point

2. Second point: Switch the robot to a position B+45° and align the end point with the reference point

3. Third point: Switch the robot to A position of A+45° and align the end point with the reference point

4. The fourth point: switch the robot to A position A-45° and align the end point with the reference point

5. Fifth point: Place the end of the tool perpendicular to the reference point (same as TC1)

6. Point six: The robot moves -x based on point five

7. Point 7: The robot moves +Y on the basis of point 6

8. Click "Run to this point" to check whether the calibration is accurate;

9. Click the "Calculation" button, the calibration is successful.

10. If you are not satisfied with a point in the calibration process, you can click the corresponding [Cancel calibration] button on the line to cancel the calibration and then calibrate the point again.

11. Click the "Demo" button at the bottom to open the "Demo" interface and explain how to calibrate the tool.Click the [Back] button at the bottom to return to the "Tool Hand calibration"

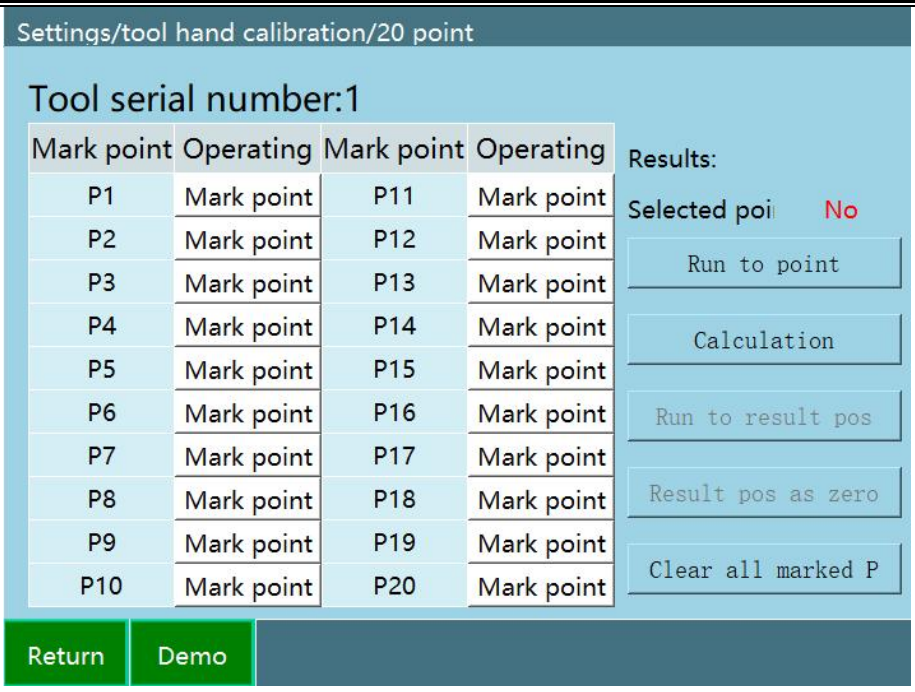
interface.

7.1.5. Twelve/Fifteen-Point Calibration

Twelve/Fifteen/Twenty-Point Calibration share a calibration interface, and the first fifteen points of calibration are the fifteen-point calibration method.

Twelve-Point Calibration means that Fifteen-Point calibration does not mark the last three points (thirteen-fifteen),and the calibration result is only the offset of the XYZ axis direction of the tool hand, without the value of rotation around ABC.

Click on the {Twenty-Point Calibration} button at the bottom to enter the Twenty-point calibration interface, as shown in the figure.


Settings/tool hand calibration/20 point

Tool serial number:1

Mark point	Operating	Mark point	Operating
P1	Mark point	P11	Mark point
P2	Mark point	P12	Mark point
P3	Mark point	P13	Mark point
P4	Mark point	P14	Mark point
P5	Mark point	P15	Mark point
P6	Mark point	P16	Mark point
P7	Mark point	P17	Mark point
P8	Mark point	P18	Mark point
P9	Mark point	P19	Mark point
P10	Mark point	P20	Mark point

Results:

Selected poi No

Find a reference point (the tip of the calibration cone is the reference point) and ensure that this reference point is fixed.Start inserting position points. For each insert point, click "Mark the point" to insert 15 points. The specific steps are as follows:

1. First point: The robot returns to zero point, aligns the robot tip with the tip of the calibration cone through rectangular coordinates, and calibrates the first point

2. Second point: On the basis of the first point, rotate C 180 degrees through the rectangular coordinate system;Tip alignment marks the second point

3. The third point: the robot returns to zero, and aligns the robot tip with the calibration cone tip through the rectangular coordinate system;Calibrate the third point (same as the first point)

4. The fourth point: On the basis of the third point, B- is made through the rectangular coordinate system, the degree is located at 30° -60° , and the tip alignment marks the fourth point

5. The fifth point: On the basis of the fourth point, make B+, J5>-90° through the rectangular coordinate system, align the robot tip with the tip of the calibration cone, and calibrate the fifth point

6. The sixth point: Select the first point and move the robot to the first point. On the basis of the first point, do B+, J5>-90° through the rectangular coordinate system, and calibrate the sixth point with tip alignment

7. The seventh point: On the basis of the first point, make B-, J5>-90° through the rectangular coordinate system, and align the tip to mark the seventh point

8. The eighth point: On the basis of the seventh point, make A+ through the rectangular coordinate system, rotate 90° , J5>-90° , and align the eighth point with the tip.

9. The ninth point: On the basis of the seventh point, make A- through the rectangular

coordinate system, rotate 90°, J5>-90°, and align the tip to mark the ninth point

10. Tenth point: The robot returns to the first point, moves the five axes through the joint coordinate system, so that the five axes are upward,J5<-90°, aligns the tips, and calibrates the tenth point

11. The eleventh point: On the basis of the tenth point, the robot does A+ through the rectangular coordinate system, rotates 90°, J5<-90°, and marks the eleventh point with tip alignment

12. The twelfth point;On the basis of the tenth point, the robot does A- through the rectangular coordinate system, rotates 90°, J5<-90°, and marks the eleventh point with tip alignment

13. Point 13: The robot returns to the zero position and adjusts the robot attitude so that the tool tip at the end of the robot faces vertically down. Align the calibration tip with the calibration cone and calibrate the thirteenth point

14. The 14th point: On the basis of the 13th point, X- is made through the rectangular coordinate system, the robot shifts a distance, and directly clicks to calibrate the 14th point

15. Point 15: On the basis of point 14, do Y+ through the rectangular coordinate system, shift the robot some distance, and directly click to calibrate point 15

When you have finished marking, click Calculate.

[Cancel calibration] : If you are not satisfied with a point in the calibration process, you can click the corresponding [Cancel calibration] button on the line to cancel the calibration and then calibrate the point again.

[Run to the point] : After a point is calibrated, click [Run to the point], and the robot will run to the point.

[Mark the result position as zero] : Set the position after calibration compensation as the zero position of the current robot.

[Clear all calibration points] : The calibration points will be saved in the controller, and the calibration results will be cleared only after clicking Cancel calibration, clear all calibration points and switching tool to enter the calibration interface.

Click the [Back] button at the bottom to return to the "Tool Hand calibration" interface.

Caution

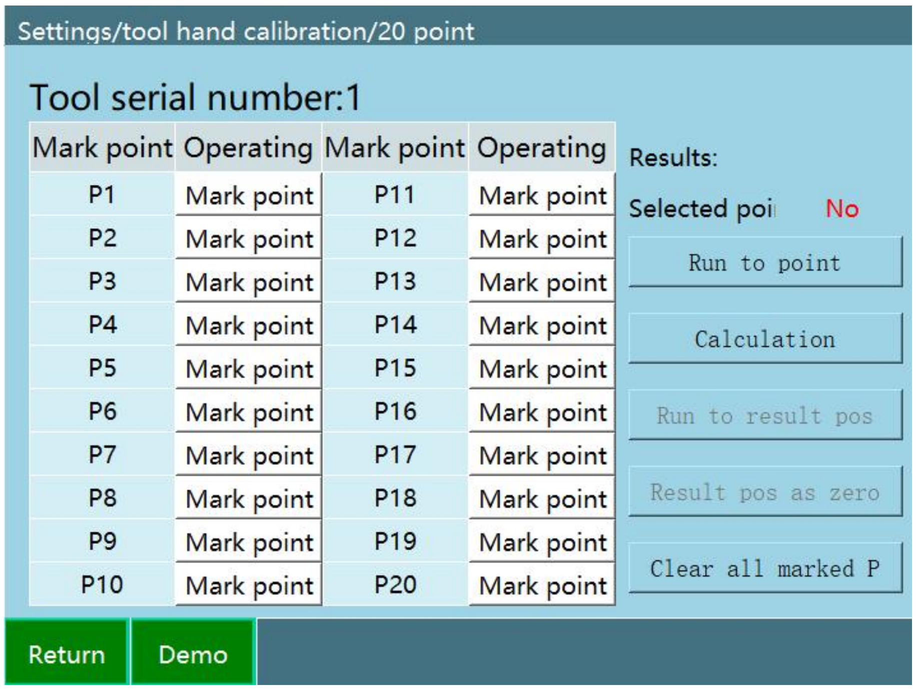
The position of each point, please try to take any direction of the posture.If the position is rotated in a certain direction, sometimes the accuracy is not accurate.

Please keep the reference point fixed during the calibration process, otherwise the calibration error will increase.

7.1.6. Twenty-Point Calibration

Twelve/Fifteen/Twenty-Point Calibration share a calibration interface, and calibrate all twenty points is to use the Twenty-Point Calibration method.

Click on the {Tool Calibration} button at the bottom of the interface to enter the "twenty point calibration" interface, as shown in the figure.



Settings/tool hand calibration/20 point

Tool serial number:1

Mark point	Operating	Mark point	Operating
P1	Mark point	P11	Mark point
P2	Mark point	P12	Mark point
P3	Mark point	P13	Mark point
P4	Mark point	P14	Mark point
P5	Mark point	P15	Mark point
P6	Mark point	P16	Mark point
P7	Mark point	P17	Mark point
P8	Mark point	P18	Mark point
P9	Mark point	P19	Mark point
P10	Mark point	P20	Mark point

Results:

Selected poi No

Specific calibration steps are as follows:

1. The first point: the vertical reference point at the end of the robot tool hand
2. Second point: Move A+20° on the basis of the first point
3. Third point: Move A+20° on the basis of the first point
4. The fourth point: Move A+60° based on the first point
5. Fifth point: Move A-20° from the first point
6. Sixth point: Move A-40° from the first point
7. Seventh point: Move A-60 degrees from the first point
8. Point 8: Move B+20° from point 1
9. Point 9: Add B+40° to point 1
10. Tenth point: Move B+60° from the first point
11. Point Eleven: Move B-20 degrees from point one
12. Point 12: Move B-40 degrees from point 1
13. Point 13: Move B-60 degrees from point 1
14. Point 14: Move C+30° from point 1
15. Point 15: Move C+45° from point 1

16. Point 16: Move C+60° from point 1

17. Point 17: Move C+90° from point 1

18. Point 18: Move C-30° from point 1

19. Point 19: Move C-60 degrees from point 1

20. Point 20: Move C-90° from point 1

When you have completed the 20-point mark, click Calculate.

[Cancel calibration] : If you are not satisfied with a point in the calibration process, you can click the corresponding [Cancel calibration] button on the line to cancel the calibration and then calibrate the point again.

[Run to the point] : After a point is calibrated, click [Run to the point], and the robot will run to the point.

[Mark the result position as zero] : Set the position after calibration compensation as the zero position of the current robot.

[Clear all calibration points] : The calibration points will be saved in the controller, and the calibration results will be cleared only after clicking Cancel calibration, clear all calibration points and switching tool to enter the calibration interface.

⚠ Caution

The position of each point, please try to take any direction of the posture.If the position is rotated in a certain direction, sometimes the accuracy is not accurate.

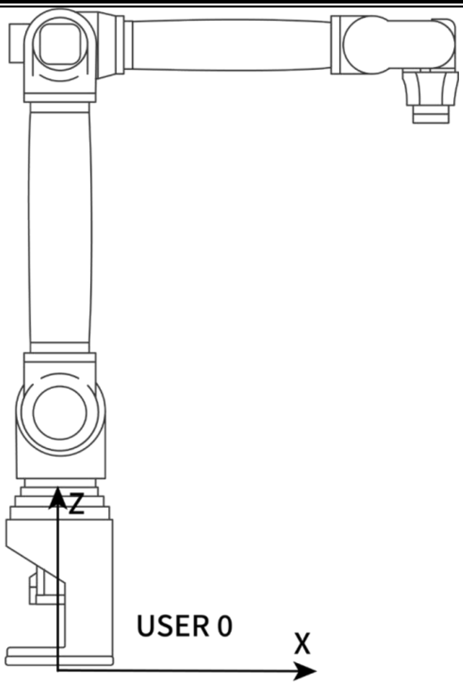
Please keep the reference point fixed during the calibration process, otherwise the calibration error will increase.

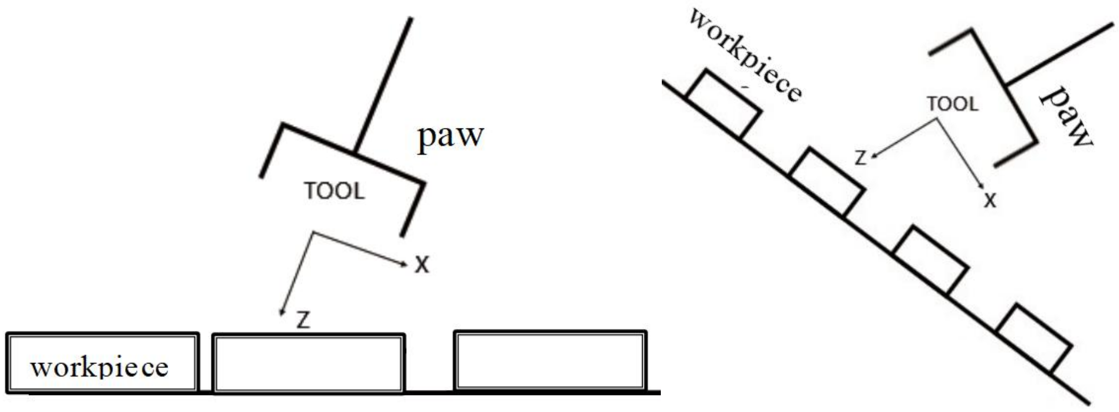
7.2. User coordinate system

7.2.1. User coordinate system action

Definition: Default user coordinate system: The default user coordinate system User0 coincides with the rectangular coordinate system.The new user coordinate system is based on the default user coordinate system change.

Thinking: From Thinking 1 we know that the user coordinate system is a reference object in motion, but what role does it play in the actual debugging process?





Speculation: As can be seen from the figure, it would be difficult to debug the position of each workpiece using the default user coordinate system User0 or cartesian coordinate system, but it would be convenient to have a coordinate system with two directions directly parallel to the workbench.

Conclusion:

User coordinate system action

1. Determine the reference coordinate system;

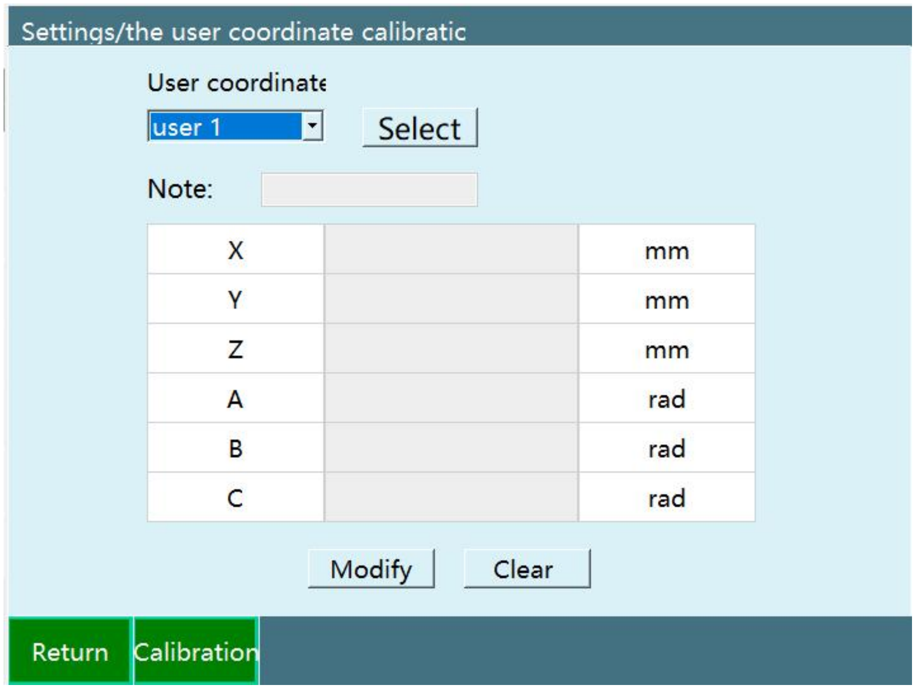
2. Determine the movement direction of the workbench for convenient debugging.

Characteristics of user coordinate system

The new user coordinate system is obtained according to the change of the default user coordinate system User0. The position and attitude of the new user coordinate system do not change with respect to the space.

7.2.2. User coordinate parameter Settings

Click the "User coordinate Calibration" button in the "Setting" interface to enter the "User coordinate" interface, as shown in the figure.



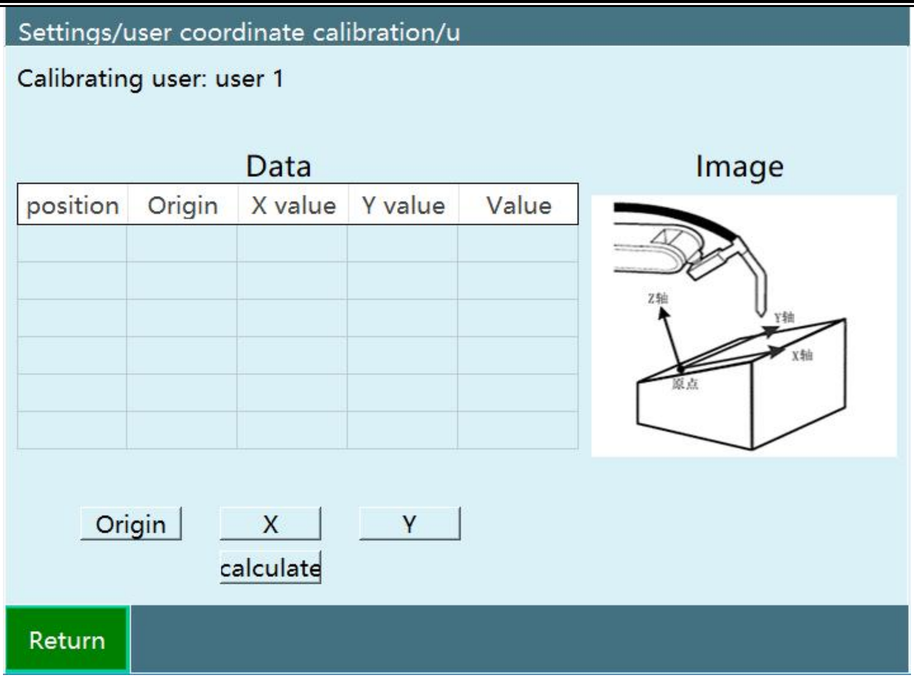
The parameters of the user coordinates are as follows

parameters	function
X value	Migration of user coordinate origin from the X-axis direction of robot base origin
Y value	Migration of user coordinate origin from the Y-axis direction of robot base origin
Z value	Migration of user coordinate origin from the Z-axis direction of robot base origin
A value	The rotation angle (radian) of the user coordinate system relative to the Cartesian coordinate system around the X-axis direction
B value	The rotation angle (radian) of the user coordinate system relative to the Cartesian coordinate system around the Y-axis direction
C value	The rotation angle (radian) of the user coordinate system relative to the Cartesian coordinate system around the Z-axis direction

If you have exact values, please fill in directly. Note that the three values of ABC are radian.

7.2.3. User Coordinate System Calibration

Click on the {User Calibration} button at the bottom of the "User Coordinate Calibration" interface to enter the "User Calibration" interface, as shown in the figure.


Settings/user coordinate calibration/u

Calibrating user: user 1

Data

position	Origin	X value	Y value	Value

Please follow the following steps to calibrate the user coordinate system:

1. Move the end of the robot to the desired origin of the user coordinate system, and click the "Calibration origin" button;

2. Move the robot at any distance relative to the origin of the user coordinate system to the position expected to be the positive direction of the X-axis of the user coordinate system, and click the "Calibration X-axis" button;

3. Move the robot at any distance relative to the origin of the user coordinate system to the position expected to be the positive direction of the Y-axis of the user coordinate system, and click the "Calibration Y-axis" button.

⚠ Caution

If the Y-axis of the user coordinate system is not calibrated accurately, the system will automatically compensate.

8. Numerical Variable

This chapter mainly describes the variables of the control system.

	type	Quantity	Example
Global numerical variables	Global Integer Variable、GINT	Each job file can contains 999	GI001
	Global Floating Point Variable、GDOUBLE		GD001
	Global Bool Variable、GBOOL		GB001
Lobal numerical variables	Lobal Integer Variable、INT	Each job file can contain 999	I001
	Lobal Floating Point Variable、DOUBLE		D001
	Lobal Bool Variable、BOOL		B001

8.1. Variable class instruction

- INT

Defining and assigning a local INT variable requires inserting instructions into the program header.

INT	function	Define local INT variables and assignment	
	parameters	variable name	0-999
		variable value source	Constant INT DOUBLE BOOL GINT GDOUBLE GBOOL
		new parameter	constant
		source parameter	existing variable name
	use example	INT I001=1	

- DOUBLE

Defining and assigning a local DOUBLE variable requires inserting instructions into the program header.

DOUBLE	function	Define local DOUBLE variables and assignment	
	parameters	variable name	0-999
		variable value source	Constant INT DOUBLE BOOL GINT

			GDOUBLE GBOOL
		new parameter	constant
		source parameter	existing variable name
	use example	DOUBLE D001=1	

- BOOL

Defining and assigning a local BOOL variable requires inserting instructions into the program header.

BOOL	function	Define local DOUBLE variables and assignment	
	parameters	variable name	0-999
		variable value source	Constant INT DOUBLE BOOL GINT GDOUBLE GBOOL
		new parameter	constant
		source parameter	existing variable name
	use example	BOOL A001=1	

- SETINT

Assignment to INT variables.

SETINT	function	Assign INT variables.	
	parameters	variables	INTGINT
		variable value source	Constant INT DOUBLE BOOL GINT GDOUBLE GBOOL
		new parameter	constant

		source parameter	existing variable name
	use example	SETINT I001=1	

- SETDOUBLE

  Assignment to DOUBLE variables.

SETDOUBLE	function	Assignment to DOUBLE variables.	
	parameters	variables	INTGINT
		variable value source	Constant INT DOUBLE BOOL GINT GDOUBLE GBOOL
		new parameter	constant
		source parameter	existing variable name
	use example	SETDOUBLE D001=1	

- SETBOOL

  Assignment to BOOL variables.

SETBOOL	function	Assignment to BOOL variables.	
	parameters	variables	INT GINT
		variable value source	Constant INT DOUBLE BOOL GINT GDOUBLE GBOOL
		new parameter	constant
		source parameter	existing variable name
	use example	SETBOOL A001=1	

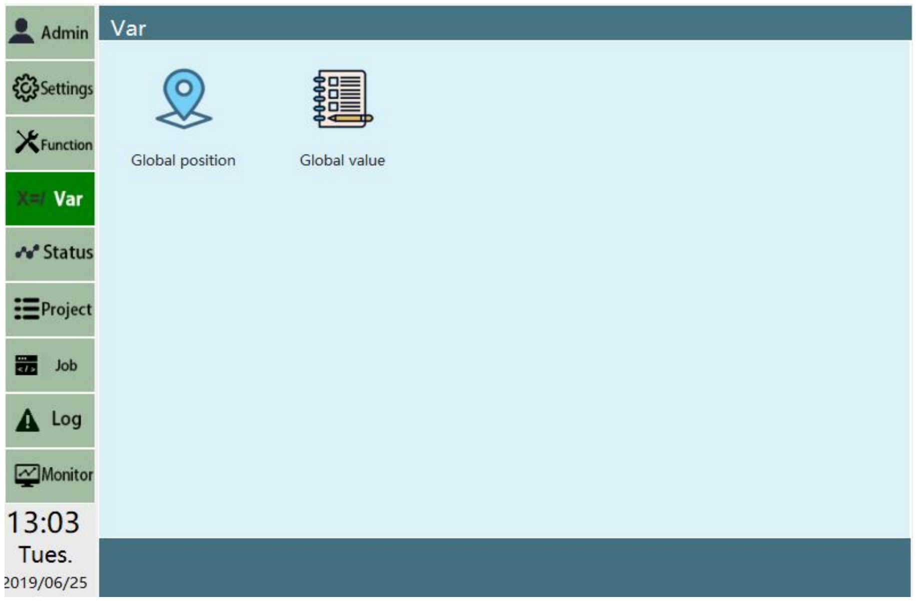
- FORCESET

  In the process of running the program, the global variable value in the current cache is written into the variable file.

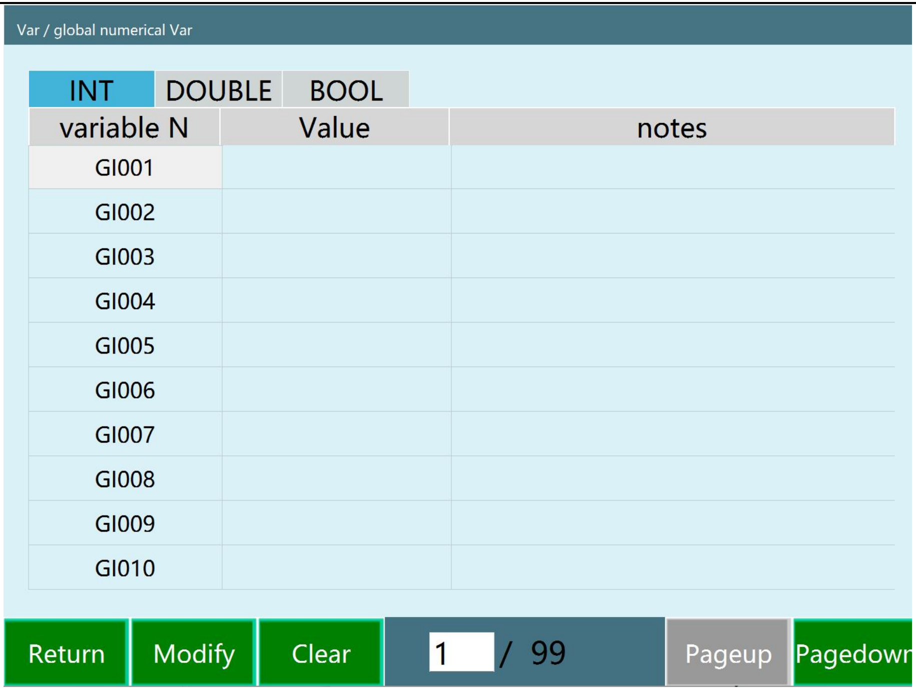
FORCESET	function	In the process of running the program, the global variable value in the current cache is written into the variable file.	
	parameters	variable type	GINT GDOUBLE GBOOL
		variable name	variable name
	use example	FORCESET GI001	

8.2. Global value variable

The global value variable is a variable that can act on all robots and all programs, such as program AA of Robot 1 and program BB of Robot 2, which can use the same global value variable at the same time. This section will mainly explain the use of the global variable interface, as well as the use of position and numerical variables.



Imagine that the robot needs so many instructions to complete a process, if we insert instructions every time, set variables, how tedious work, based on this we add a value variable to facilitate the call.For example, "WHILE (INT001=10) END (WHILE)" such an instruction, in the robot to complete a certain process of the program, we directly call your pre-set value variable.At the same time, the global value variable can be used to transfer information between the main program, the called subroutine and the background program for logical judgment.Numerical variables store values, including integer variables, real variables, Boolean variables three kinds.


Var / global numerical Var

INT DOUBLE BOOL

variable N	Value	notes
GI001		
GI002		
GI003		
GI004		
GI005		
GI006		
GI007		
GI008		
GI009		
GI010		

Caution

Global variable assignments are saved directly to arguments

8.2.1. Global Boolean variable

Global Boolean variables store bytes. In this interface, you can modify the value and comment of each variable.The meanings of each parameter are as follows:

1. The variable name is the number of the variable. The name of the global Boolean variable is GBxxx.

2. The value is the value of the variable, and the range of the value of Boolean variable is "0/1".

3. Comment is the comment defined by the user for the variable to facilitate the user to mark the function of the variable. The range can be any value and can be in Chinese.

8.2.2. Global integer type variable

Global integer variables are stored as integers. You can modify the value and comment of each variable on this interface.The meanings of each parameter are as follows:

1. The name of a variable is the number of the variable. The name of a global integer variable is GIxxx.

2. The value is the value of the variable, and the range of the integer type variable is integer.

3. Comment is the comment defined by the user for the variable to facilitate the user to mark the function of the variable. The range can be any value and can be in Chinese.

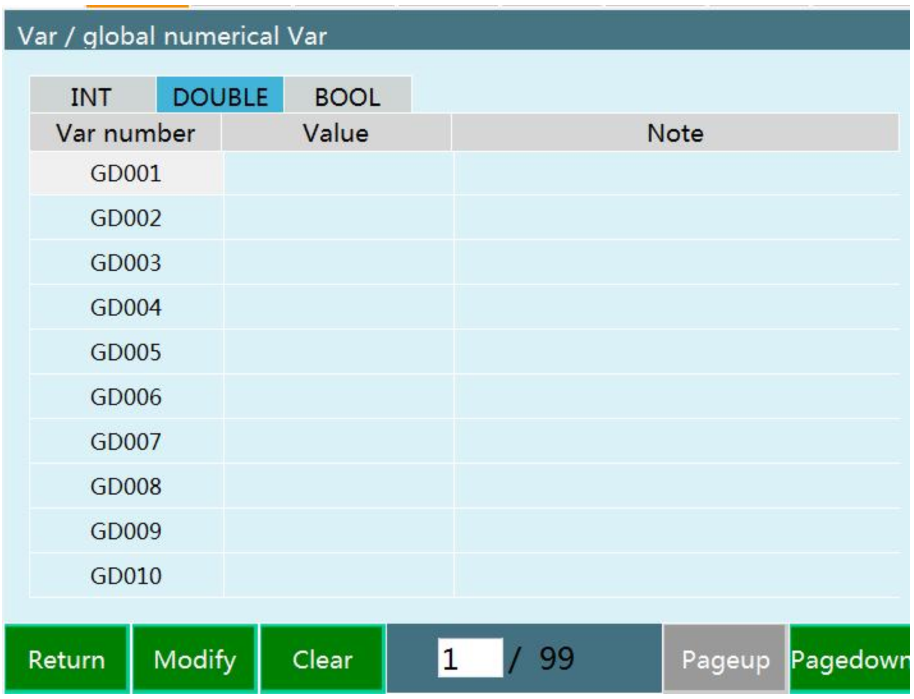
8.2.3. Global floating point variable

Global real variables are saved as real numbers, and the value, content, and comment of each variable can be modified on this interface.The meanings of each parameter are as follows:

1. The variable name is the number of the variable. The global real variable name is GDxxx.

2. The value is the value of the variable. The range of floating-point variables is real.

3. Comment is the comment defined by the user for the variable to facilitate the user to mark the function of the variable. The range can be any value and can be in Chinese.


Var / global numerical Var

INT | DOUBLE | BOOL

Var number	Value	Note
GD001		
GD002		
GD003		
GD004		
GD005		
GD006		
GD007		
GD008		
GD009		
GD010		

Click on the data type you want to modify, select the variable name, and click on {Modify}, then you can modify the values and comments. Then click on {Save}. Click on {Clear} to clear the data you choose.

8.2.4. Use of Global Numerical Variable

8.2.4.1. Define Global Value Variable

Define variables before using them. The methods for defining variables are as follows:

1.Click on the {Variable} button on the left to enter the variable interface;

2.Click on the global value variable;

3.Select the corresponding variable number and click on the {Modify} button;

4.Fill in the required values at the values and notes;

5.Variables that are not manually defined are defaulted to zero.

8.2.4.2. Assign Values to Global Variable by Calculating Instructions

Global variables can be calculated by ADD, SUB, MUL, DIV and MOD instructions. Note: Global Bool variables cannot be calculated !

ADD

Add operation (+).

Formula: variable type (variable name) = variable type (variable name) + variable value source (parameter)

To calculate global integer or global value variables, select GINT or GDOUBLE at the variable type. If the source of variable value is customized, the parameters can be manually filled in at the "new parameters". It can also be used for other variable values.

Case 1:

Premise: GI001=1 Instruction: ADDGI0011

Significance:GI001=GI001 Result: GI001=2

Case 2:

Premise: GI001=1 GI002=2

Instruction: ADD GI001 GI002 Significance: GI001=GI001+GI002 Result: GI001=3

SUB

Subtraction operation (-)

Formula: variable type (variable name) = variable type (variable name)- variable value source (parameter)

To calculate global integer or global value variables, select GINT or GDOUBLE at the variable type.

If the source of variable value is customized, the parameters can be manually filled in at the "new parameters". It can also be used for other variable values.

Case :

Premise:GD001=3.4 Instruction: SUB GD001 1.1 Significance: GI001=GI001-1.1

Result: GD001=2.3

MUL

Multiply operation(*)

Formula: variable type (variable name) = variable type (variable name) * variable value source (parameter)

To calculate global integer or global value variables, select GINT or GDOUBLE at the variable type. If the source of variable value is customized, the parameters can be manually filled in at the "new parameters". It can also be used for other variable values.

Case :

Premise: GD001=3.4 GI001=2

Instruction: MUL GD001 GI001

Significance: GD001=GD001*GI001

Result: GD001=6.8

DIV

Division operation (DIV)

Formula: variable type (variable name) = variable type (variable name) DIV variable value source (parameter)

To calculate global integer or global value variables, select GINT or GDOUBLE at the variable type. If the source of variable value is customized, the parameters can be manually filled in at the "new parameters". It can also be used for other variable values.

Case :

Premise: GD001=3.4 GI001=2

Instruction: DIV GD001 GI001

Significance: GD001=GD001÷GI001 Result: GD001=1.7

MOD

Remainder operation（MOD）

Formula: variable type (variable name) = variable type (variable name) MOD variable value source (parameter)

To calculate global integer or global value variables, select GINT or GDOUBLE at the variable type. If the source of variable value is customized, the parameters can be manually filled in at the "new parameters". It can also be used for other variable values.

Case :

Premise: GD001=14 GI001=3

Instruction: MOD GD001 GI001

Significance: GD001=GD001 MOD GI001

Result: GD001=2

8.2.4.3. Assign Values Directly to Global Variables

Through SETBOOL, SETINT and SETDOUBLE instructions, the value of variables can be changed directly when the program is running.

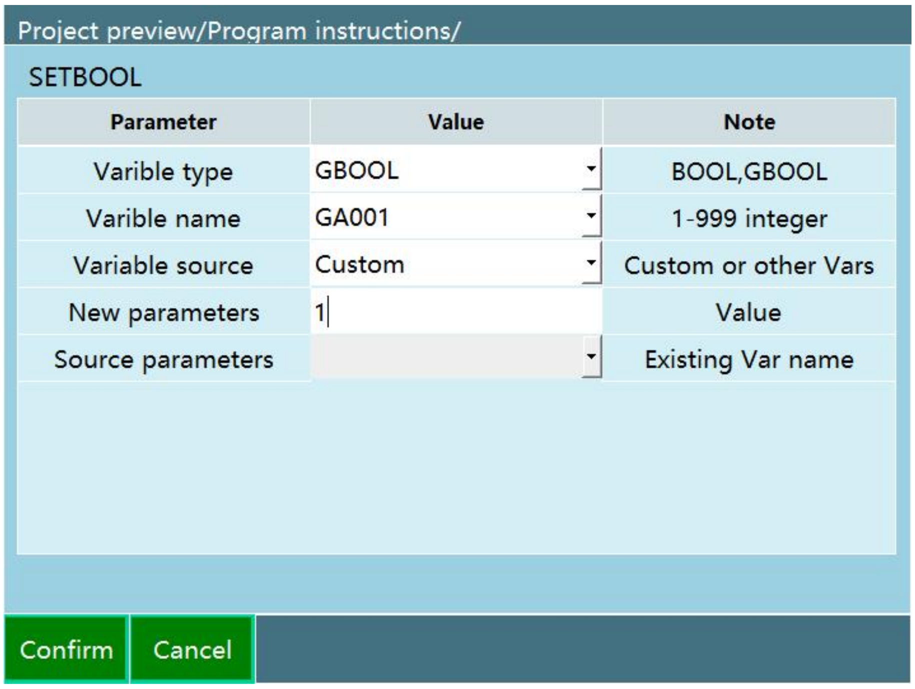
1.In the program, click on the {Insert} button；

2.Select "variable class "；

3.To change the global BOOL variable, select the SETBOOL instruction and click on {OK}；

4.Select "GBOOL" at the type of variable; select the previously defined global BOOL variable; variable value source selected "customized". Fill in the value that needs to be changed at the new parameter, and if you need to change the value of the variable to 1, fill in 1 here.

For example, if you need to change the value of GA001 variable to 1 when running the program, fill in the parameters as shown in the figure below.



Parameter	Value	Note
Varible type	GBOOL	BOOL,GBOOL
Varible name	GA001	1-999 integer
Variable source	Custom	Custom or other Vars
New parameters	1	Value
Source parameters		Existing Var name

SETINT and SETDOUBLE are used to set INT and DOUBLE type variables respectively, the usage is the same as above.

8.2.4.4. Use Global Variables to Count

In the process of running the program, all calculation and assignment operations are to change the values in the cache, and the values in the "variable-global value" interface will not be modified, that is, when the program stops running, the values of all global variables will be restored.

To count a loop process (such as a WHILE inner loop), you can use the FORCESET instruction. Using scene: There is a process between a WHILE and ENDWHILE instruction. There is an ADD GI001 1 instruction in the process, that is, every time a loop is made between WHILE and ENDWHILE, the value of GI001 variable is increased by one, that is, the number of execution times of the process is increased by one. After the program stops running, the value of GI001 is reduced to 0, so the number of operation times of the process can not be seen.

Solution: Insert a FORCESET GI001 instruction after the Add GI001 1 instruction. When the program is finished, the value of GI001 can be seen in the "variable-global value" interface, which represents the number of times the program runs.

Click on the {Insert} button in the "program" interface；

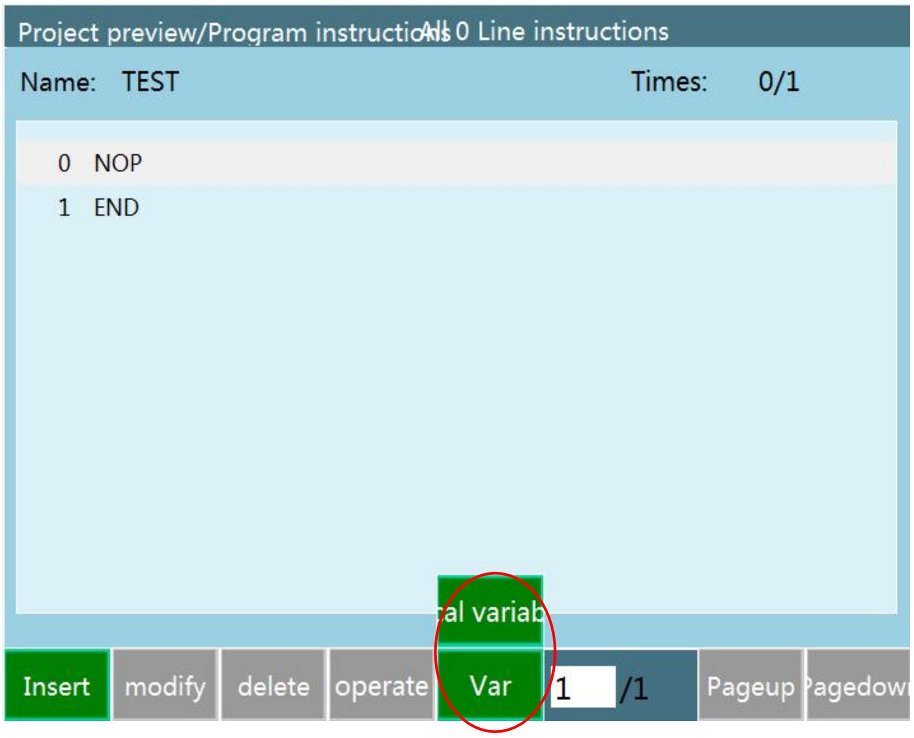
Select "variable class "-"FORCESET", and click on {OK}；

Select the variable type. If you want to change the global integer variable, select GINT and select "GI001" for the variable name；

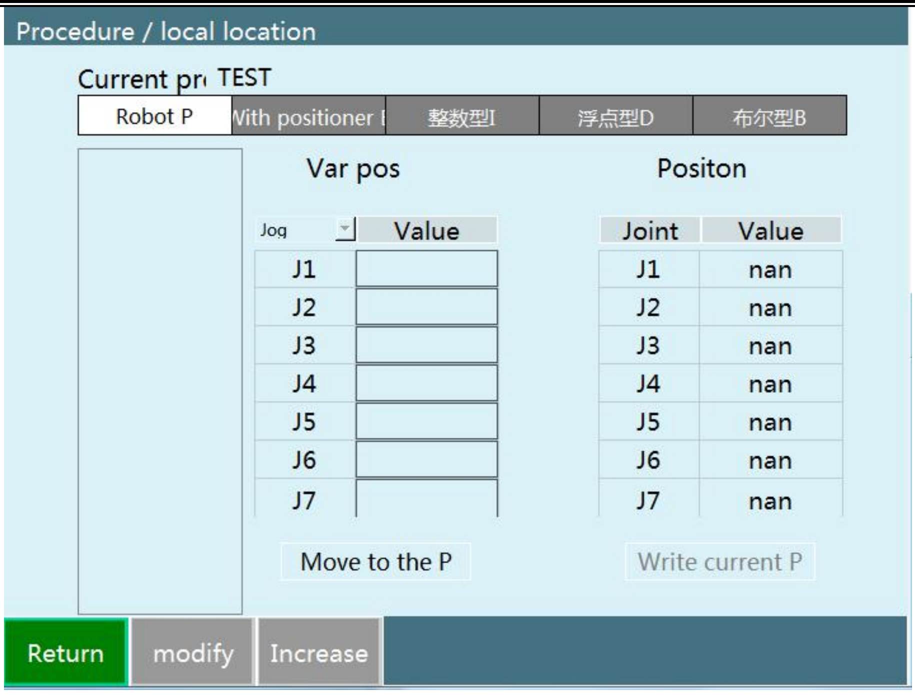
Click on the {Insert} button to complete.

8.3. Local numerical variables

Local variables can only be used for the defined program itself, such as variables of program A can not be used in program B.



Numerical variables store values, including integer variables, real number variables, and Boolean variables.All defined local numerical variables can only be used in the current program, and other programs and background programs cannot be used.


Procedure / local location

Current pr TEST

Robot P | With positioner | 整数型I | 浮点型D | 布尔型B

Var pos

Jog	Value
J1	
J2	
J3	
J4	
J5	
J6	
J7	

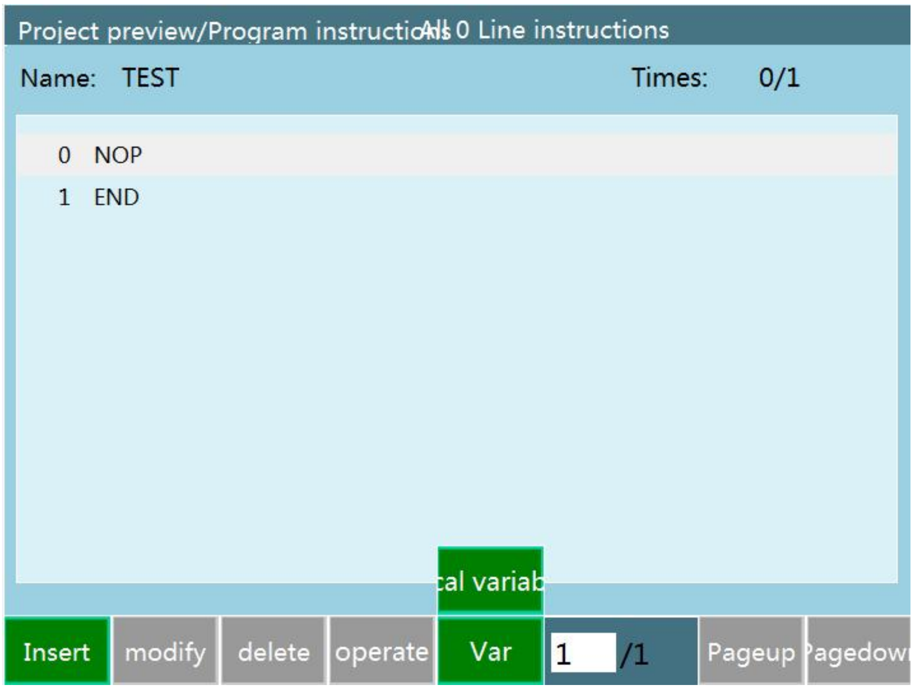
Positon

Joint	Value
J1	nan
J2	nan
J3	nan
J4	nan
J5	nan
J6	nan
J7	nan

8.3.1. Use of local variables

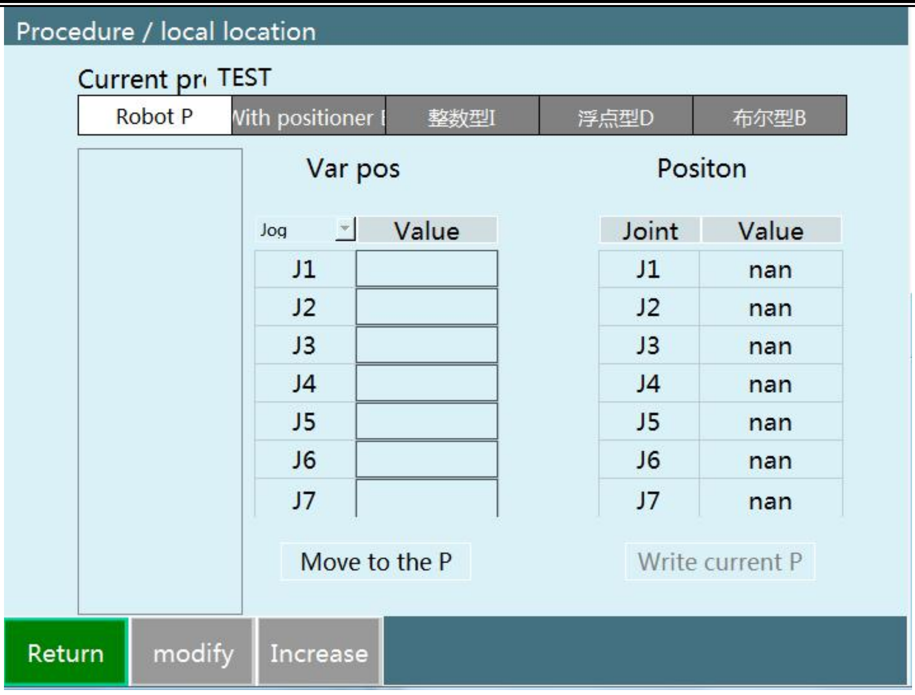
8.3.1.1. Define Local Variables

Defining local variables is different from defining global variables. To define a local variable, you need to click the variable-local variable page setting on the program page.


Project preview/Program instructions All 0 Line instructions

Name: TEST Times: 0/1

```
0   NOP
1   END
```



Jog	Value
J1	
J2	
J3	
J4	
J5	
J6	
J7	

Joint	Value
J1	nan
J2	nan
J3	nan
J4	nan
J5	nan
J6	nan
J7	nan

Int I

Local integer variables are used to store integer variables. The variable name is Ixxx.

The default value is 0. When you need to modify, select the variable name to be modified, enter the value, and click Save.

Floating Point Variable D

Local real variables are used to store real variables. The variable name is Dxxx.

The default value is 0. When you need to modify, select the variable name to be modified, enter the value, and click Save.

Bool variable B

Local Bool variables are used to store Bool variables. The variable name is Bxxx.

The default value is 0. When you need to modify, select the variable name to be modified, enter the value, and click Save.

8.3.1.2. Assignment of Local Variables Using Calculation Instructions

The method of calculating and assigning local variables using the ADD, SUB, MUL, DIV, and MOD instructions is the same as the calculation method for global variables.

8.3.1.3. Assign Values Directly to Variables

The method of directly assigning a local variable using the SETINT, SETDOUBLE, and SETBOOL instructions is the same as the method of directly assigning a global variable.

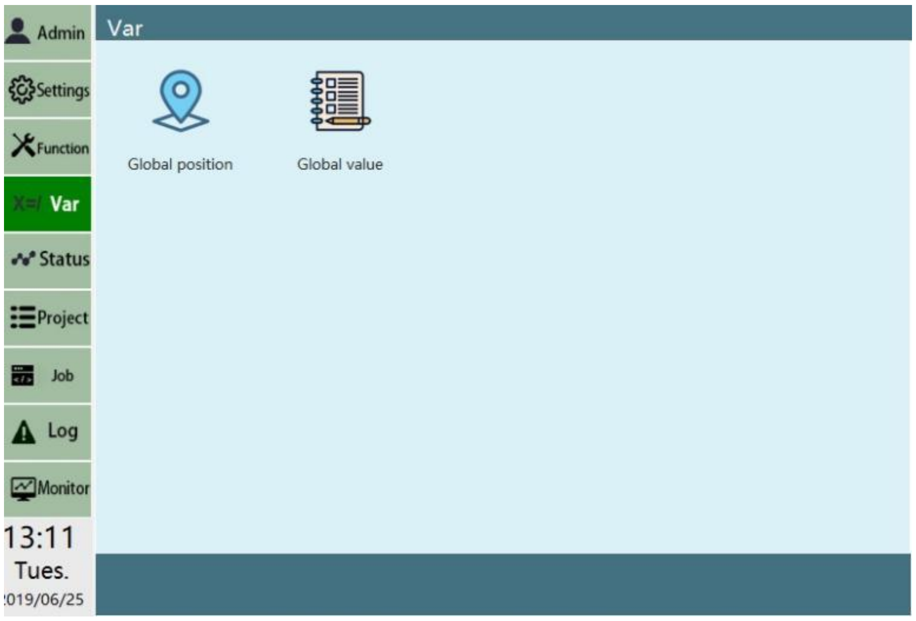
9. Position variables

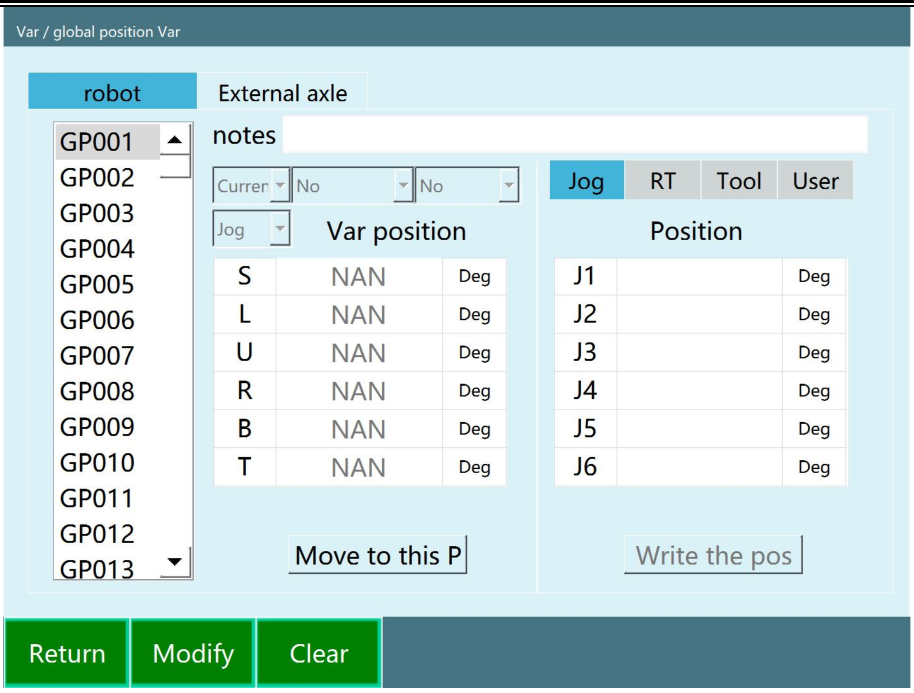
This chapter mainly describes the variable settings of the control system.

The global position variable	Global location、G	G001
The local location variable	Local locationP point	P001
	Local locationEpoint	E001
	Local locationS point (IMOV)	S001
	Local locationR point(SAMOV)	R001

9.1. Global position variable

The global position variable (G) is available in all job files for a robot. Defining global position variables needs to be done on the “variables - global position” interface.





S	NAN	Deg
L	NAN	Deg
U	NAN	Deg
R	NAN	Deg
B	NAN	Deg
T	NAN	Deg

J1		Deg
J2		Deg
J3		Deg
J4		Deg
J5		Deg
J6		Deg

Global location variables are defined as follows:

1. The Variables - Global Location page is displayed.

2. Select the variable to be defined, for example, G001.

3. Teach the robot to the position that needs to be defined and switch the coordinate system to the required coordinate system, such as the rectangular coordinate system;

4. Click the "Modify" button;

5. Click the "Record current Point" button;

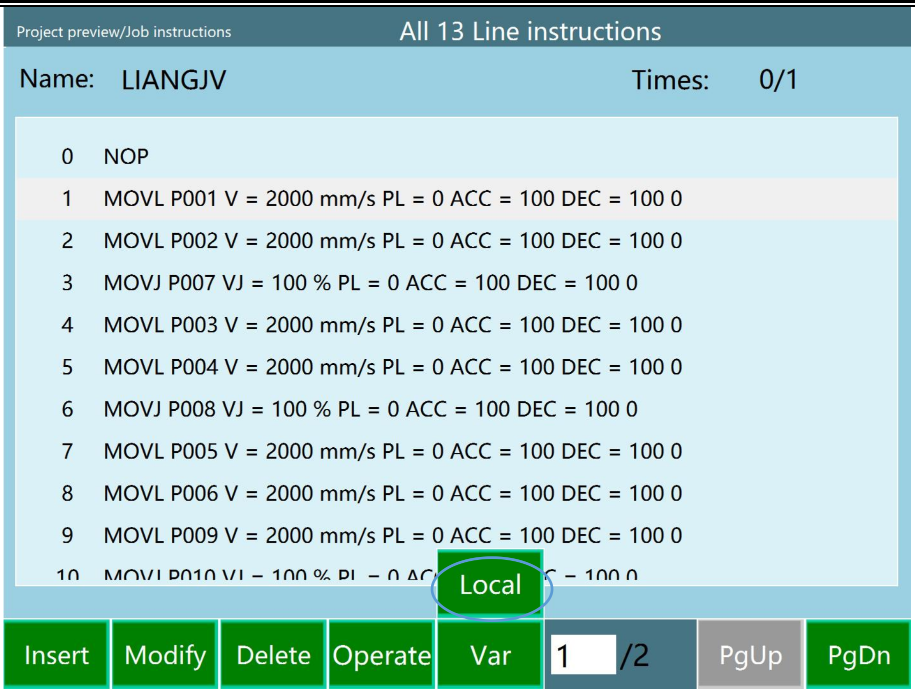
6. Click the "Save" button.

9.2. Local position variable

The local location variable (P) can only be used for a single job file and cannot be used across all job files.The definition of local position variable is only automatically defined when "New" variable is selected when MOVJ, MOVL, MOVC and other motion class instructions are inserted.

9.2.1. The local location variable is set in method 1

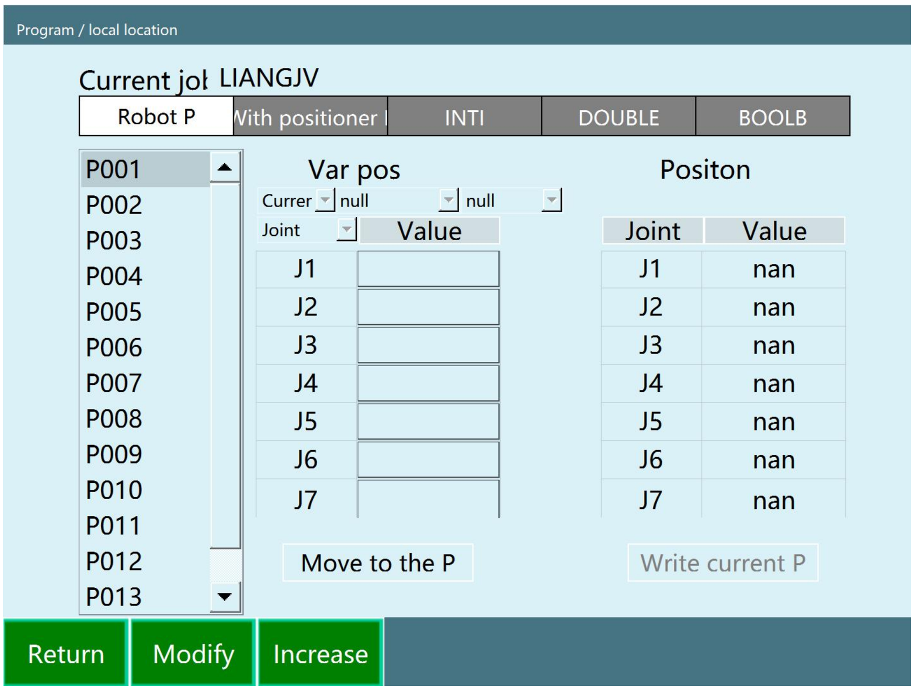
1. Click Program - Variable - Local Variable to enter the view interface of local variable



Project preview/Job instructions All 13 Line instructions

Name: LIANGJV Times: 0/1

```
0   NOP
1   MOVL P001 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
2   MOVL P002 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
3   MOVJ P007 VJ = 100 % PL = 0 ACC = 100 DEC = 100 0
4   MOVL P003 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
5   MOVL P004 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
6   MOVJ P008 VJ = 100 % PL = 0 ACC = 100 DEC = 100 0
7   MOVL P005 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
8   MOVL P006 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
9   MOVL P009 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
10  MOVJ P010 VJ = 100 % PL = 0 AC       C = 100 0
```



Program / local location

Current jol LIANGJV

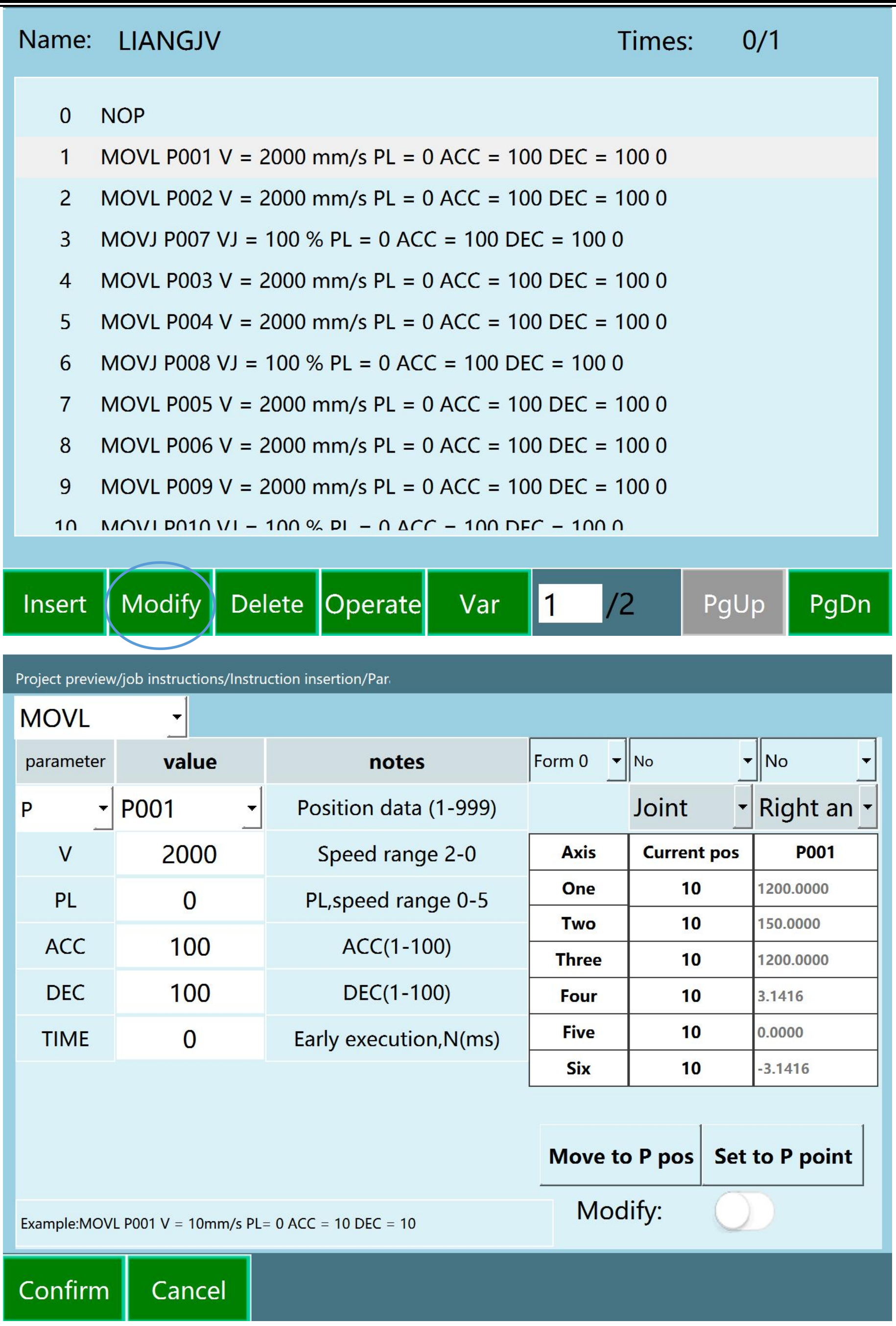
Joint	Value
J1	
J2	
J3	
J4	
J5	
J6	
J7	

Joint	Value
J1	nan
J2	nan
J3	nan
J4	nan
J5	nan
J6	nan
J7	nan

2. Local position variables can be modified, added, run to the point, write the current position and other functions.

9.2.2. The local location variable is set in method 2

1. Create or modify the MOVJ instruction to access the instruction interface



Name: LIANGJV Times: 0/1

```
0   NOP
1   MOVL P001 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
2   MOVL P002 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
3   MOVJ P007 VJ = 100 % PL = 0 ACC = 100 DEC = 100 0
4   MOVL P003 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
5   MOVL P004 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
6   MOVJ P008 VJ = 100 % PL = 0 ACC = 100 DEC = 100 0
7   MOVL P005 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
8   MOVL P006 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
9   MOVL P009 V = 2000 mm/s PL = 0 ACC = 100 DEC = 100 0
10  MOVJ P010 VJ = 100 % PL = 0 ACC = 100 DEC = 100 0
```

Project preview/job instructions/Instruction insertion/Par

MOVL

parameter	value	notes
P	P001	Position data (1-999)
V	2000	Speed range 2-0
PL	0	PL,speed range 0-5
ACC	100	ACC(1-100)
DEC	100	DEC(1-100)
TIME	0	Early execution,N(ms)

Form 0 | No | No

Joint | Right an

Axis	Current pos	P001
One	10	1200.0000
Two	10	150.0000
Three	10	1200.0000
Four	10	3.1416
Five	10	0.0000
Six	10	-3.1416

Example:MOVL P001 V = 10mm/s PL= 0 ACC = 10 DEC = 10

Modify:

2. The current position column displays the robot position in the selected coordinate system;Column P001 shows the robot position in the selected coordinate system at point P

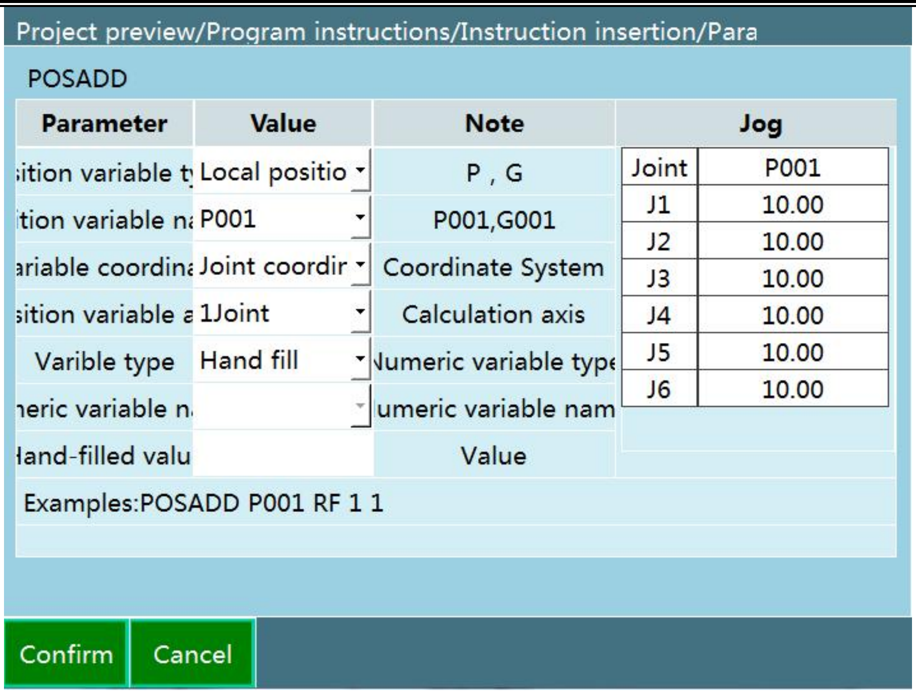
3. Move the robot to point P and power on in teaching mode;

4. Set the current position as point P, click and save the current position to local point P;Manually modify, open can manually fill P point coordinates.

9.3. Use of Position Variable Calculation Class Instructions

9.3.1. POSADD Instruction

The position variable addition operation (+), which can add the value of the single axis of the position variable (global, local), and then assign it to the axis.



Project preview/Program instructions/Instruction insertion/Para

POSADD

Parameter	Value	Note
ition variable t	Local positio	P , G
tion variable n	P001	P001,G001
ariable coordin	Joint coordir	Coordinate System
ition variable a	1Joint	Calculation axis
Varible type	Hand fill	Numeric variable type
eric variable n		umeric variable nam
Hand-filled valu		Value

Jog

Joint	P001
J1	10.00
J2	10.00
J3	10.00
J4	10.00
J5	10.00
J6	10.00

Examples:POSADD P001 RF 1 1

The variable name of the position variable can be a value variable, such as 1001=50, then P$1001 is P1001。

This instruction can add a single axis of a position variable in any coordinate system, regardless of the coordinate system in which the position variable is inserted, but it will be converted to the original coordinate system in assignment. For example, if the second axis of P001 variable is added, the P001 coordinate is in the joint coordinate system (0,0,0,0,0,0). You need to add 10 to the Z axis for this point. Convert P001 to Cartesian coordinates (500,0,1000,0,0,0), then add 10 to the Z axis, i.e (500,0,1010,0,0,0), and finally convert to joint coordinates (0,-1,1,0,1,0) and assign this value to P001.

Formula: position variable = position variable {Coordinate System (Axis)} + value variable or number

To calculate a global integer or global value variable, select GINT or GDOUBLE at the variable type.

If the source of variable value selects to fill in manually, the parameters can be filled in manually at the "new parameters". It can also be used for other variable values.

9.3.2. POSSUB Instruction

The position variable subtraction operation (-), which can subtract the value of the single axis of the position variable (global, local), and then assign it to the axis.

The meaning and method of this instruction are similar to the POSADD instruction.

Formula: position variable = position variable {Coordinate System (Axis)} - value variable or number.

To calculate a global integer or global numeric variable, select GINT or GDOUBLE at the variable type.

If the source of variable value selects to fill in manually, the parameters can be filled in manually at the "new parameters". It can also be used for other variable values.

9.3.3. POSSET Instruction

Position variable assignment, which can assign the value of position variable (global, local) on a single axis directly.

The meaning and method of this instruction are similar to the POSADD instruction. Formula: position variable {Coordinate System (Axis)} = value variable or number.

To calculate a global integer or global numeric variable, select GINT or GDOUBLE at the variable type.

If the source of variable value selects to fill in manually, the parameters can be filled in manually at the "new parameters". It can also be used for other variable values.

9.3.4. READPOS Instruction

Read the position variable coordinate instruction, which can read the coordinate value of the position variable into the numerical variable.

When the coordinate value of "current position" is selected to read, the coordinate value is read when the robot runs to that position.

Formula: value variables (I, D.GI, GD) = position variables {Coordinate System (Axis)}

9.3.5. USERFRAME_SET Instruction

Modify the user coordinate system instruction, which allows the user to modify the value of an axis of the user coordinate system parameters. After modification, all points using user coordinates are migration.

For example, P001, P002 and P003 all use user coordinate system 1 and insert USERFRAME_SET instruction to add 10 to the X parameter of user coordinate system 1, then the position variables of P001, P002 and P003 are migration by 10mm to the X axis.

9.3.6. TOOLFRAME_SET Instruction

Modify the tool coordinate command.This command can modify the value of one axis of the tool coordinate system.After modification, the trajectory in the program used will change with the modification of the tool coordinate system value.For example, the original tool offset is (0,0,200,0,0,0).Use this command to modify the offset of the Z-axis direction to 100, and the center position of the 6-axis flange will be offset down by 100mm during operation.If it is changed to the corresponding tool hand, the tool hand with the Z-axis offset of 200mm is changed to the tool hand with the Z- axis offset of 100mm, and the tip position remains unchanged.

9.3.7. COPYPOS Instruction

Copy point instruction.Copy the current position, local position variable, global position variable, etc. to another local or global position variable.

For example, copy the current position to the local position variable.Source location variable type: current location, source location variable name: not selected,Target position variable type: local position variable, target position variable name:P001.

9.4. Morphological parameter

Morphological parameters are only available for 6-axis robots

The form value is the decimal conversion value of the robot axis 1, 3 and 5

Conversion mode

For example, the first axis of a six-axis robot is 59 degrees, the second axis is 69 degrees, the third axis is 79 degrees, the fourth axis is 89 degrees, the fifth axis is 99 degrees, and the sixth axis is 109 degrees.

Axis 1/3/5 is taken, and the point position range between -90 and +90 is 1, not 0;So the result is as follows.

Axis	1 axis	3 axis	5Axis
Binary value	0	0	1

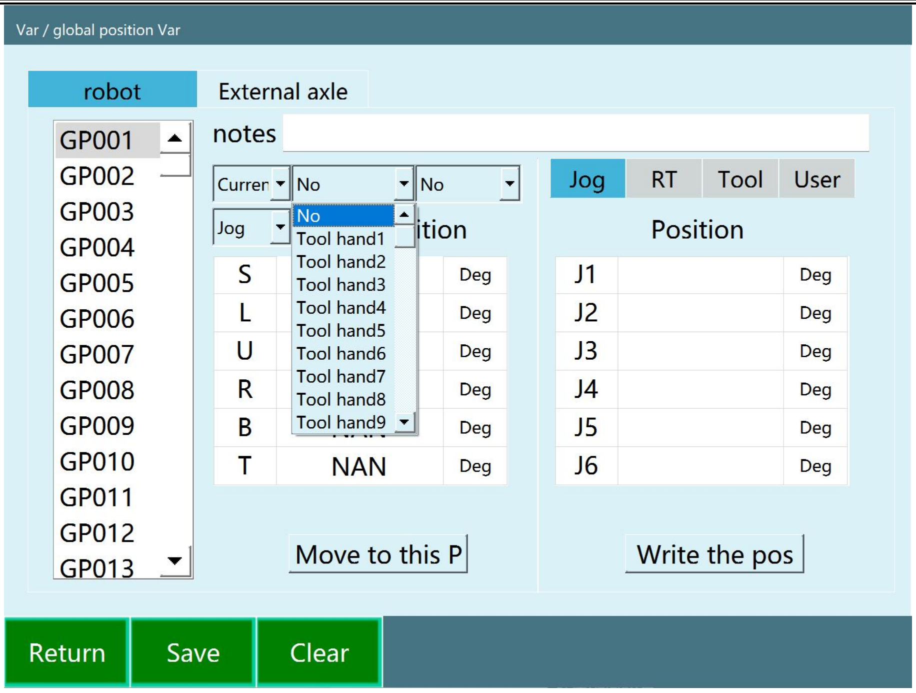
Binary 001= decimal 1

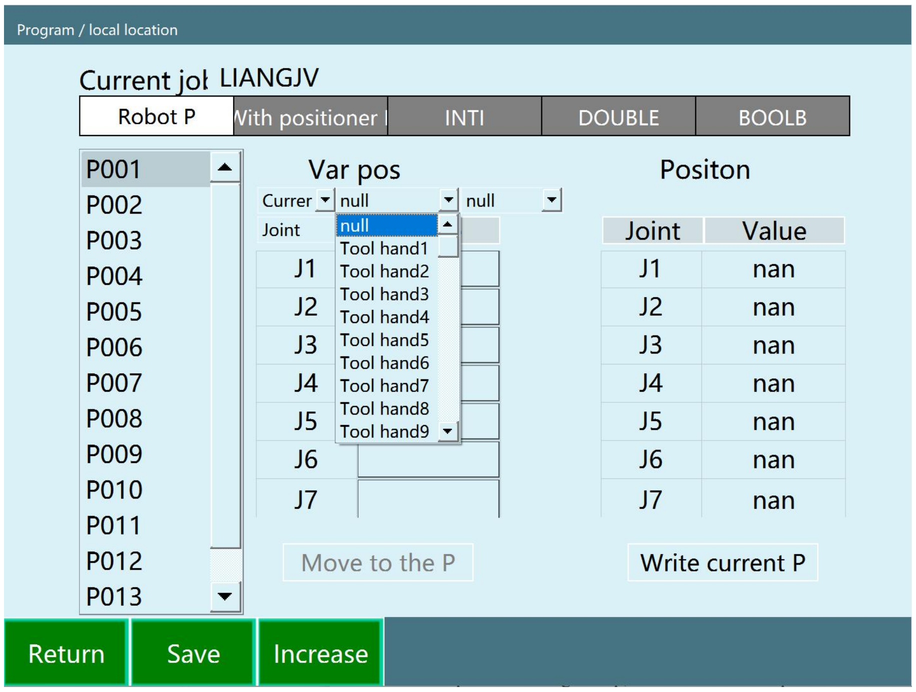
The form value is the decimal result plus 1, and the form value of the point is 2.

9.5. Tool hand parameter

Set cartesian coordinate, tool coordinate, user coordinate point binding tool hand, no binding select none;If the tool hand and point parameters are different during movement, it cannot run.

For example, if you bind tool hand 2 and use the tool hand to run the instructions using this point in a single step, an error will be reported.



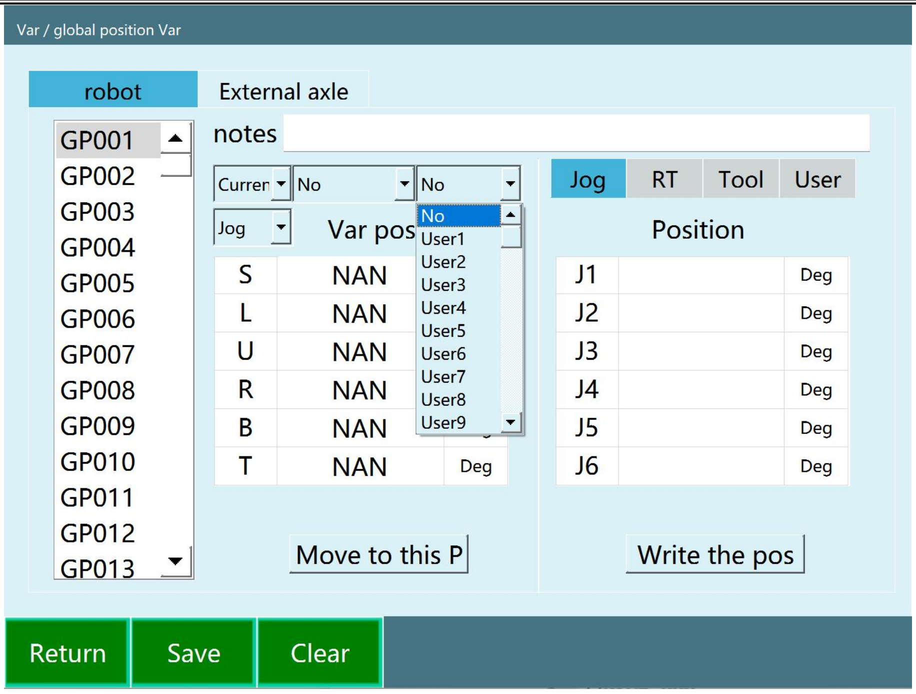


Joint	Value
J1	nan
J2	nan
J3	nan
J4	nan
J5	nan
J6	nan
J7	nan

9.6. User coordinate parameter

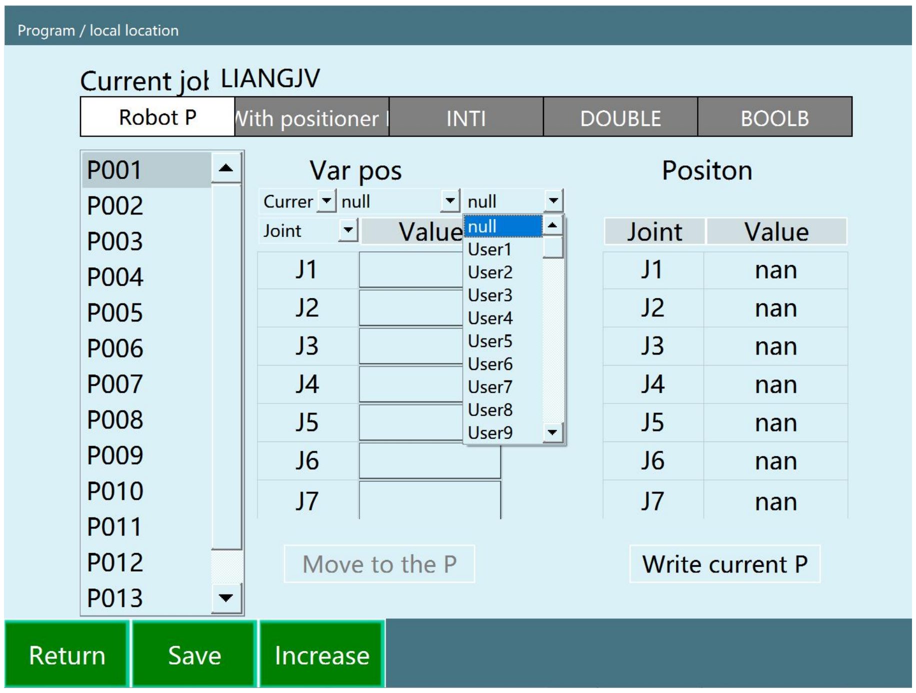
Set user coordinates point bit bind user coordinates, no bind select none;If the user coordinates and point parameters are different when moving, it cannot run.

For example, if you bind user coordinate 1 and use user coordinate 5 to run instructions using this point in a single step, an error will be reported.



S	NAN	
L	NAN	
U	NAN	
R	NAN	
B	NAN	
T	NAN	Deg

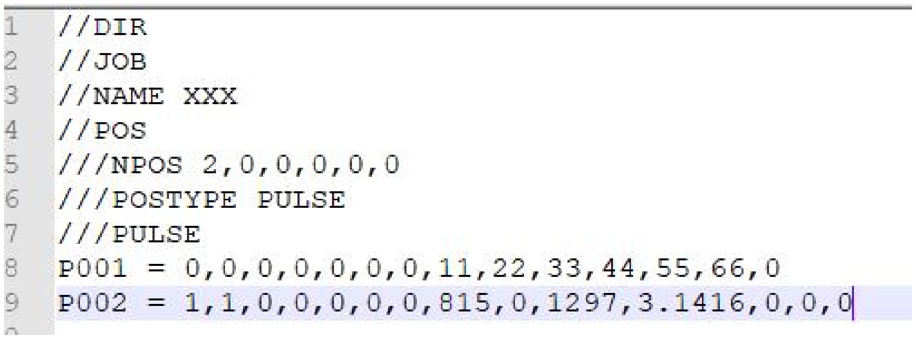
J1		Deg
J2		Deg
J3		Deg
J4		Deg
J5		Deg
J6		Deg



Joint	Value
J1	nan
J2	nan
J3	nan
J4	nan
J5	nan
J6	nan
J7	nan

9.7. Program local point parameter description

This function describes the format in which points are saved in the program.



```
//DIR
//JOB
//NAME XXX
//POS
///NPOS 2,0,0,0,0,0
///POSTYPE PULSE
///PULSE
P001 = 0,0,0,0,0,0,0,11,22,33,44,55,66,0
P002 = 1,1,0,0,0,0,0,815,0,1297,3.1416,0,0,0
```

Such as,1,0,0,0,0,0,815,0,1297,3.1416 P002 = 1, 0, 0

Point data is decomposed as follows:

P002	Point name	Point name P001-P999
1	Coordinate system	0：joint 1：Right Angle 2：tool 3：user
1	Angle/radian	0：Angle (junction)1：Radians (right Angle point, tool point, user point)
0	Form/right hand and left hand	Six-axis is the shape parameter, and four-axis SCARA is the left and right hand parameter
0	tool	Tool hand number
0	user	User coordinate number
0	reserve	reserve
0	reserve	reserve
815	1 axis	Point position 1 axis coordinates
0	2 axis	Point position 2 axis coordinates
1297	3 axis	Point position 3 axis coordinates
3.1416	4 axis	Point position 4 axis coordinates
0	5 axis	Point position 5 axis coordinates
0	6 axis	Point position 6 axis coordinates
0	7 axis	Point position 7 axis coordinates

10. Use of Conditional Judgment Class Instructions

Conditional judgment class instruction includes CALL, IF, WHILE, WAIT, JUMP and other instructions.

10.1. Instruction Description

10.1.1. CALL

CALL instruction is used to call a subroutine.

In this system, there is no distinction between the main program and the subroutine when establishing the program. When one program calls another program, the called program is the subroutine.

The two programs cannot call each other, that is, after program A calls program B, program B cannot call program A.

Parameter	Meaning
Program name	The program name of the called program

Case

Premise: Two programs, Job1 and Job2, have been established, and CALL instructions have been inserted into Job1.

Instruction: CALL [Job2] Meaning: Call subroutine Job2

Process: When the instruction of Job1 runs to the CALL instruction, the program jumps to the program Job2. After running all the instructions of the program Job2, the program jumps back to the next line of the CALL [Job2] instruction of the program Job1 and continues to run.

10.1.2. IF

If the condition of IF instruction is satisfied, the instruction between IF and ENDIF is executed. If the condition of IF instruction is not satisfied, then jump to ENDIF instruction and continue to run the instruction under ENDIF, and the instruction between IF and ENDIF is not executed.

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

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

Note that when the beginning of the program is IF and the last action is ENDIF instruction, please insert a 0.1 second TIMER (Delay) instruction above or below the IF instruction.

Otherwise, if the condition of the IF instruction is not satisfied, the program will crash.

When IF instruction is inserted, ENDIF instruction will be inserted at the same time. When deleting IF instruction, please n

Parameter	Meaning
Parameter type	The type of the comparison number 1,the input value of a variable or a number or an analog
Parameter name	If the type of the previous selection is a variable (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), here is the variable name of comparison number 1 If the type selected in the previous item is the input value (DIN, AIN), then here is the port number for digital or analog input
Comparison method	== equal to < less than more than <= less than or equal to >= more than or equal to != not equal to
Variable value source	The type of the comparison number 2, customize or input values of variables or numbers or analogs
New parameter	If the type of the previous selection is custom, it is not optional here.If the type of the previous selection is a variable (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), here is the variable name of comparison number 1.If the type selected in the previous item is the input value (DIN, AIN), then here is the port number for digital or analog input.
Source parameter	If the variable value source is selected as custom, the value of comparison number 2 is filled in directly here.

Case 1

Premise: Global variables or local variables have been defined, such as GI001=8 Instructions: IF (GI001<9)

Other instructions, such as MOVJ, etc. ENDIF

Meaning: If GI001<9, run the instruction between IF and ENDIF, if it is not satisfied, it will not run.

Process: Because GI001=8<9, the condition is valid, the instruction between IF and ENDIF is run, and the instruction following ENDIF is continued after running.

Case 2

Premise: Global variables or local variables have been defined, such as GI001=5, D001=8.88 Instructions: IF(GI001>=D001)

Other instructions, such as MOVJ, etc. ENDIF

Meaning: If GI001>=D001, run the instruction between IF and ENDIF, if it is not satisfied, it will not run.

Process: Because GI001 = 5, D001 = 8.88, 5 < 8.88, the condition is not valid, and the instruction between IF and ENDIF will not be run. The program jumps to the next line of instruction under ENDIF and continues to run.

Case 3

Premise: An external IO equipment is connected, such as the input value of port 10 of digital IO is 1.

Instructions: IF (DIN10=1)

Other instructions, such as MOVJ, etc. ENDIF

Meaning:If the input value of the digital IO port 10 is equal to 1, the instruction between IF and ENDIF is run, but if it is not satisfied, it will not run.

Process: Because the input value of port 10 of digital IO is 1,that is, DIN10 = 1, the condition is satisfied. After running the instructions between IF and ENDIF, the instructions under ENDIF are continued.

10.1.3. ELSE

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 the IF is valid, the instruction between the IF and the ELSE is executed, and the next line of the jump to the ENDIF instruction continues to run, instead of running the instruction between ELSE and ENDIF.

When the judgment condition of the IF is not valid, it will jump to the instruction running between ELSE and ENDIF, instead of running the instruction between IF and ELSE.

Note that when you delete IF instructions, you need to delete the corresponding ELSE and ENDIF instructions, otherwise the program will not run.

Case 1

Premise: Global variables or local variables have been defined, such as GI001=8 Instructions: IF (GI001<9)

Other instructions 1, such as MOVJ, etc. ELSE

Other instructions 2, such as MOVJ, etc. ENDIF

Meaning: If GI001<9, the instruction 1 between IF and ELSE is run, and if it is not, the instruction 2 between ELSE and ENDIF is run.

Process: Because GI001=8<9, the condition is valid, the instruction between IF and ELSE is run, and the instruction following ENDIF is continued after running.

Case 2

Premise: Global variables or local variables have been defined, such as GI001=5, D001=8.88 Instructions: IF(GI001>=D001)

Other instructions 1, such as MOVJ, etc. ELSE

Other instructions 2, such as MOVJ, etc. ENDIF

Meaning: If GI001>=D001, the instruction 1 between IF and ELSE is run, and if it is not, the instruction 2 between ELSE and ENDIF is run.

Process: Because GI001 = 5, D001 = 8.88, 5 < 8.88, the condition is not valid. Instruction 2 between ELSE and ENDIF will be run, and then the instructions under ENDIF will continue to run.

10.1.4. ELSEIF

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

When the condition of IF is not satisfied, it will jump to ELSEIF instruction to judge the condition of ELSEIF. If it is satisfied, it will run the instruction between ELSEIF and ENDIF, and then continue to run the instruction under ENDIF. If it is not satisfied, it will jump directly to one line of instruction under ENDIF to continue to run.

If multiple ELSEIFs are nested in IF and ENDIF, the first ELSEIF judgment condition is judged when the judgment condition of IF is not valid, and if it is, the instructions between the first ELSEIF and the second ELSEIF are run; if not, the second ELSEIF judgment condition is judged, and so on. Note that when you delete IF instructions, you need to delete the corresponding ELSEIF and ENDIF instructions, otherwise the program will not run.

Case 1

Premise: Global variables or local variables have been defined, such as GI001=8 Instructions: IF (GI001<9)

Other instructions 1, such as MOVJ, etc. ELSEIF (GI001>7)

Other instructions 2, such as MOVJ, etc. ENDIF

Meaning: If GI001<9, the instruction 1 between IF and ELSEIF is run. If it is not satisfied, the judgment condition of ELSEIF is judged. If it is satisfied, the other instruction 2 is run. If it is not satisfied, the instruction jumped to the ENDIF continues to run.

Process: Because GI001=8<9, the condition is valid, the instruction between IF and ELSEIF is run, and the instruction following ENDIF is continued after running.

Case 2

Premise: Global variables or local variables have been defined, such as GI001=5, D001=8.88 Instructions: IF(GI001>=D001)

Other instructions 1, such as MOVJ, etc. ELSEIF (D001<9)

Other instructions 2, such as MOVJ, etc. ENDIF

Meaning: If GI001>=D001, the instruction 1 between IF and ELSE is run. If it is not satisfied, the judgment condition of ELSEIF is judged. If it is satisfied, other instruction 2 is run. If it is not satisfied, the instruction jumped to ENDIF continues to run.

Process: Because GI001 = 5, D001 = 8.88, 5 < 8.88, the condition is not valid, the condition of ELSEIF is judged, because D001 = 8.88 < 9, if the condition is valid, the other instruction 2 is run.

Case 3

Premise: Global variables or local variables have been defined, such as GI001=5, D001=8.88 Instructions: IF(GI001>=D001)

Other instructions 1, such as MOVJ, etc. ELSEIF(D001>9)

Other instructions 2, such as MOVJ, etc. ELSE

Other instructions 3, such as MOVJ, etc.

ENDIF

Meaning: If GI001>=D001, the instruction 1 between IF and ELSE is run. If it is not satisfied, the judgment condition of ELSEIF is judged. If it is satisfied, other instruction 2 is run. If it is not satisfied, the instruction jumped to ENDIF continues to run.

Case 3

Premise: Global variables or local variables have been defined, such as GI001=5, D001=8.88 Instructions: IF(GI001>=D001)

Other instructions 1, such as MOVJ, etc. ELSEIF(D001>9)

Other instructions 2, such as MOVJ, etc. ELSE

Other instructions 3, such as MOVJ, etc.

ENDIF

Meaning: If GI001>=D001, the instruction 1 between IF and ELSE is run. If it is not satisfied, the judgment condition of ELSEIF is judged. If it is satisfied, other instruction 2 is run. If it is not satisfied, the instruction jumped to ENDIF continues to run.

Process: Because GI001 = 5, D001 = 8.88, 5 < 8.88, the condition is not valid, the condition of ELSEIF is judged, because D001 = 8.88 < 9, if the condition is valid, the other instruction 3 is run.

Case 4

Premise: Global variables or local variables have been defined, such as GI001=5, D001=8.88 Instructions: IF(GI001>=D001)

Other instructions 1, such as MOVJ, etc. ELSEIF(D001>9)

Other instructions 2, such as MOVJ, etc. ELSEIF (GI001<6)

Other instructions 3, such as MOVJ, etc. ELSEIF (GI001>4)

Other instructions 4, such as MOVJ, etc. ENDIF

Meaning: If GI001>=D001, the instruction 1 between IF and ELSE is run. If it is not satisfied, the judgment condition of the first ELSEIF is judged. If D001>9 is satisfied, other instructions 2 are run. If not, the second ELSEIF is judged. The judgment condition is that if GI001<6, other instructions 3 are run, if not, the third ELSEIF is judged, and so on.

Process: Because GI001 = 5, D001 = 8.88, 5 < 8.88, then the condition is not valid, judging the ELSEIF judgment condition, because D001 = 8.88 < 9, the condition is not valid, judging the second ELSEIF, GI001 = 5 < 6, if the condition is valid, then run the other instruction 3, and then jump to the instruction under ENDIF to continue running.

Other instructions 1,

Such as MOVJ, etc. ELSEIF(D001>9)

Other instructions 2, such as MOVJ, etc. ELSEIF (GI001<6)

Other instructions 3, such as MOVJ, etc. ELSEIF (GI001>4)

Other instructions 4, such as MOVJ, etc. ENDIF

Meaning: If GI001>=D001, the instruction 1 between IF and ELSE is run. If it is not satisfied, the judgment condition of the first ELSEIF is judged. If D001>9 is satisfied, other instructions 2 are run. If not, the second ELSEIF is judged. The judgment condition is that if GI001<6, other instructions 3 are run, if not, the third ELSEIF is judged, and so on.

Process: Because GI001 = 5, D001 = 8.88, 5 < 8.88, then the condition is not valid, judging the ELSEIF judgment condition, because D001 = 8.88 < 9, the condition is not valid, judging the second ELSEIF, GI001 = 5 < 6, if the condition is valid, then run the other instruction 3, and then jump to the instruction under ENDIF to continue running.

10.1.5. WHILE

When the condition of WHILE instruction is satisfied, the instruction between WHILE and ENDWHILE will be run circularly. If the judgment condition is not satisfied before running to the WHILE instruction, it will jump to the ENDWHILE instruction instead of the instruction between WHILE and ENDWHILE when running to the WHILE instruction; if the judgment condition becomes unsatisfactory during the process of running the instruction between WHILE and ENDWHILE, it will continue to run until running to the ENDWHILE line, and it will not circulate but continue to run the instruction under ENDWHILE.

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

Note that inserting the WHILE instruction will also insert the ENDWHILE instruction. To delete the WHILE instruction, delete the corresponding ENDWHILE instruction at the same time, otherwise the program will not run.

When the program starts with WHILE and the last instruction is ENDWHILE, insert a 0.3 second TIMER (Delay) instruction at the beginning or end of the program. Otherwise, when the condition of WHILE instruction is not satisfied, the program will crash.

When instructions in WHILE do not have motion instructions or may fall into a dead cycle in some cases, please insert a 0.3 second TIMER (Delay) instruction between WHILE and ENDWHILE. Otherwise, when the conditions of WHILE instructions are satisfied, the program may crash.

The WHILE instruction can be used to nest multiple judgment instructions such as WHILE, IF or JUMP at the same time.

Parameter	Meaning
Parameter type	The type of the comparison number 1,the input value of a variable or a number or an analog
Parameter name	If the type of the previous selection is a variable (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL),then here is the variable name of comparison number 1.If the type of the previous selection is input value (DIN, AIN), then here is the port number of the digital input or analog input.
Comparison method	== equal to < less than more than <= less than or equal to >= more than or equal to != not equal to
Variable value source	The type of the comparison number 2, customize or input values of variables or numbers or analogs
New parameter	If the type of the previous selection is custom, it is not optional here. If the type of the previous selection is a variable (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), here is the variable name of comparison number 1. If the type selected in the previous item is the input value (DIN, AIN), then here is the port number for digital or analog input.
Source parameter	If the variable value source is selected as custom, the value of comparison number 2 is filled in directly here.

Case 1

Premise: The variable GI001=1has been defined. Instruction:WHILE (GI001<2)

Other instructions ENDWHILE

Meaning: When GI001 < 2, other instructions between WHILE and ENDWHILE are circulated. Until the condition is not valid, the instructions running to ENDWHILE will not be recycled, but continue to run the instructions under ENDWHILE.

Process: Because GI001 = 1 < 2, other instructions between WHILE and ENDWHILE are circulated. Until the condition is not valid, the instructions running to ENDWHILE will not be recycled, but continue to run the instructions under ENDWHILE.

Case 2

Premise: The variable GI001=1, D001=7 has been defined. Instruction: WHILE (GI001<2)

Other commands 1, MOVJ, etc. WHILE (D001<10)

Other instructions 2, MOVJ, etc. ADD D001 1

ENDWHILE

Other instructions 3

ADD GI001 1 ENDWHILE

Meaning: When GI001 < 2, all instructions between WHILE and ENDWHILE will be run circularly. When running to WHILE (D001 < 10), D001 < 10 will be judged. If it is valid, other instructions 2 and ADD instructions will be run circularly until D001 > = 10, jump out of the intermediate WHILE instructions, continue to run other instructions 3 and ADD instructions, and then recycle until GI001 > = 2 jumps out. WHILE.

Process: Initial GI001 = 1 < 2, D001 = 7 < 10, so the judgment conditions of the two WHILE instructions are all valid at first, and the other instructions 2 and ADD instructions between WHILE (D001 < 10) and intermediate ENDWHILE will be circulated. D001 = 10 will be added to D001 once per loop. After three loops, D001 = 10 will be added. The intermediate judgment conditions will not be valid. Continue to run other instructions 3 and ADD GI001 1 instructions, GI001 plus 1 once per loop, and G after running 1 time. I001 = 2, the condition is not valid, continue to run the instructions under ENDWHILE.

Other instructions 2, MOVJ, etc. ADD D001 1

ENDWHILE

Other instructions 3

ADD GI001 1 ENDWHILE

Meaning: When GI001 < 2, all instructions between WHILE and ENDWHILE will be run circularly. When running to WHILE (D001 < 10), D001 < 10 will be judged. If it is valid, other instructions 2 and ADD instructions will be run circularly until D001 > = 10, jump out of the intermediate WHILE instructions, continue to run other instructions 3 and ADD instructions, and then recycle until GI001 > = 2 jumps out. WHILE.

Process: Initial GI001 = 1 < 2, D001 = 7 < 10, so the judgment conditions of the two WHILE instructions are all valid at first, and the other instructions 2 and ADD instructions between WHILE (D001 < 10) and intermediate ENDWHILE will be circulated. D001 = 10 will be added to D001 once per loop. After three loops, D001 = 10 will be added. The intermediate judgment conditions will not be valid. Continue to run other instructions 3 and ADD GI001 1 instructions, GI001 plus 1 once per loop, and G after running 1 time. I001 = 2, the condition is not valid, continue to run the instructions under ENDWHILE.

10.1.6. WAIT

WAIT is waiting, you can select whether there is waiting time. When the "TIME" option is not checked, the WAIT instruction will remain waiting until the judgment condition is valid. If the "TIME" option is checked, the next instruction will continue to run after waiting for the parameter for a long time. If the condition becomes valid while waiting, the next instruction is executed immediately.

Parameter	Meaning
Parameter type	The type of the comparison number 1,the input value of a variable or a number or an analog
Parameter name	If the type of the previous selection is a variable (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL),then here is the variable name of comparison number 1. If the type of the previous selection is input value (DIN, AIN), then here is the port number of the digital input or analog input.

Comparison method	== equal to < less than more than <= less than or equal to >= more than or equal to != not equal to
Variable value source	The type of the comparison number 2, customize or input values of variables or numbers or analogs
New parameter	If the type of the previous selection is custom, it is not optional here. If the type of the previous selection is a variable (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), here is the variable name of comparison number 1. If the type selected in the previous item is the input value (DIN, AIN), then here is the port number for digital or analog input.
Source parameter	If the variable value source is selected as custom, the value of comparison number 2 is filled in directly here.
TIME	Options, if not selected, wait forever until the condition is valid. If you selected, you can fill in the waiting time (seconds). After the waiting time, even if the condition is still invalid, it will jump to the next line and continue to run.
Whether continuous	If you select "Yes", the PL of the previous instruction and the PL of the next instruction can be continuous when the conditions are met before running the instruction. If you choose otherwise, PL will be interrupted.

Case

Premise: The variable GI001=1 has been defined. Instruction: WAIT(GI001==2)T = 2

Meaning: When GI001 is not equal to 2, the program stays in this instruction and waits, but after waiting more than two seconds it will no longer wait, jumping to the next program to continue running. If the condition is satisfied during the waiting process, jump to the next line immediately to continue running.

Process: Because GI001 is not equal to 2, the program stays in this instruction to wait, but after waiting more than two seconds it will no longer wait, jumping to the next program to continue running.

10.1.7. LABEL

LABEL tags instruction needs to be used in conjunction with JUMP instruction. A separate label instruction is meaningless.

parameter	meaning
Tag name	The value is a string starting with a maximum of eight characters

10.1.8. JUMP

JUMP is used for jumps and must be used in conjunction with the LABEL (label) instructions. JUMP can set whether there is a judgment condition or not. When set to no judgment condition, running to the instruction will jump directly to the corresponding LABEL instruction and continue to run the next line of instructions of LABEL.

When 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 line of the JUMP instruction.

LABEL tags can be inserted above or below JUMP, but it cannot be jumped across programs.

The label name of LABEL must be two or more characters beginning with the letter.

Inserting LABEL tags has no effect on the running of programs, but it should conform to the rules of program running, such as not inserting on MOVC instructions or on local variable definition instructions.

parameter	meaning
Tag name	LABEL name that has been inserted into the label directive, option

Judgment condition	Option, if selected, you can set the judgment condition, if not, run to JUMP directly
Parameter name	If the type of the previous selection is a variable (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), then the variable name is the comparison number 1. If the type of the previous selection is an input value (DIN, AIN), then the port number is the digital input or analog input
Mode of comparison	== Equal to < less than > Greater than <= is less than or equal to >= greater than or equal to != not equal to
Variable value source	Compare the type of number 2, custom or variable or digital, analog quantity input value
New parameter	If the previous type is Custom, this parameter is unavailable If the type selected above is a variable (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), then this is the variable name of the comparison number 1 If the type selected in the previous option is an input value (DIN, AIN), this is the port number of the digital input or analog input
Source parameter	If the source of the variable value is custom, enter the value of comparison number 2 here

Case 1

Premise: The variable GI001=1has been defined. Instruction:WHILE (GI001<2)

Other instructions ENDWHILE

Meaning: When GI001 < 2, other instructions between WHILE and ENDWHILE are circulated. Until the condition is not valid, the instructions running to ENDWHILE will not be recycled, but continue to run the instructions under ENDWHILE.

Process: Because GI001 = 1 < 2, other instructions between WHILE and ENDWHILE are circulated. Until the condition is not valid, the instructions running to ENDWHILE will not be recycled, but continue to run the instructions under ENDWHILE.

Case 2

Premise: The variable GI001=1, D001=7 has been defined. Instruction: WHILE (GI001<2)

Other commands 1, MOVJ, etc. WHILE (D001<10)

Other instructions 2, MOVJ, etc. ADD D001 1

ENDWHILE

Other instructions 3

ADD GI001 1 ENDWHILE

Meaning: When GI001 < 2, all instructions between WHILE and ENDWHILE will be run circularly. When running to WHILE (D001 < 10), D001 < 10 will be judged. If it is valid, other instructions 2 and ADD instructions will be run circularly until D001 > = 10, jump out of the intermediate WHILE instructions, continue to run other instructions 3 and ADD instructions, and then recycle until GI001 > = 2 jumps out. WHILE.

Process: Initial GI001 = 1 < 2, D001 = 7 < 10, so the judgment conditions of the two WHILE instructions are all valid at first, and the other instructions 2 and

10.1.9. UNTIL

UNTIL instruction is used to jump out during a movement. That is, during one movement of the robot, pause and start the next process. When the condition is satisfied, regardless of whether the current robot is running or not, immediately pause and start an instruction under the ENDUNTIL instruction.

The judgment condition of UNTIL is (comparison number 1 comparison method comparison number 2), for example, comparison number 1 is 2, comparison number 2 is 1, comparison method is

">", then 2>1, the judgment condition is valid; if the comparison mode is "" <" or "==", the judgment condition is not valid.

Note that the ENDUNTIL instruction is inserted at the same time as the UNTIL instruction is inserted. To delete UNTIL instructions, delete the corresponding ENDUNTIL instructions at the same time, otherwise the program will not run.

Parameter	Meaning
Parameter type	The type of the comparison number 1,the input value of a variable or a number or an analog
Parameter name	If the type of the previous selection is a variable (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL),then here is the variable name of comparison number 1. If the type of the previous selection is input value (DIN, AIN), then here is the port number of the digital input or analog input.
Comparison method	== equal to < less than more than <= less than or equal to >= more than or equal to != not equal to
Variable value source	The type of the comparison number 2, customize or input values of variables or numbers or analogs
New parameter	If the type of the previous selection is custom, it is not optional here. If the type of the previous selection is a variable (INT, DOUBLE, BOOL, GINT, GDOUBLE, GBOOL), here is the variable name of comparison number 1. If the type selected in the previous item is the input value (DIN, AIN), then here is the port number for digital or analog input.
Source parameter	If the variable value source is selected as custom, the value of comparison number 2 is filled in directly here.

Case Premise: The variable GI001=1 has been defined. Instructions: UNTIL (GI001<2) Other instructions ENDUNTIL MOVJ P003 Meaning: When running "other instructions" between UNTIL and ENDUNTIL, if GI001 becomes a value of < 2, the current action is suspended and the MOVJ P003 instruction is jumped to; if GI001 is always > 2, the MOVJ P003 instruction is run after running other instructions.

10.1.10. CRAFTLINE

Process skip instruction. Use with special process. Use with special craft skipping

Parameter	Meaning
New parameter	Fill in the number of lines in the special process program

10.1.11. CMDNOTE

Instructions comments. You comment contan use this instruction to add comments to the appropriate position of the program for easy debugging

Parameter	Meaning
Comment content	Support Chinese and English

10.1.12. POS_REACHABLE

The judgment instruction is reached. Used to judge whether the target point can be reached. If the point can be reached, the variable is set to 1, otherwise it is set to 0

Parameter	Meaning
Location variable name	P point and G point can be selected
Exercise type	Can choose MOVJ, MOVL
State is stored in variable type	Can be stored in BOOL, GBOOL

Example Prerequisite: BOOL variable A001 has been defined, and position variable P001 has been defined Command: POS_REACHABLE MOVJ P001 A001 Meaning: Determine whether to use MOVJ interpolation to run to P001 position. A001 value of 1 means reachable, A001 value of 0 means unreachable.	
State is stored in variable name	BOOL, GBOOL variable name

10.1.13. CLKSTART

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

Parameter	Meaning
Serial number	The serial number of the timer can be counted separately by using 32 timers at the same time.
Stored variable type	Store the timed time into the local DOUBLE variable or the global GDOUBLE variable.
Save the variable name	The variable name of the variable where the time is stored.

10.1.14. CLKSTOP

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

Parameter	Meaning
Serial number	The serial number of the timer to stop timing.

10.1.15. CLKRESET

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

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

11. multithreading

11.1. Local background task programming

Programs that need to be run in background tasks need to be done in Settings - Background Tasks, which is programmed the same way as writing normal programs.

Caution

It is best to insert a 0.2s delay in the WHILE loop and in the last line of the entire program.

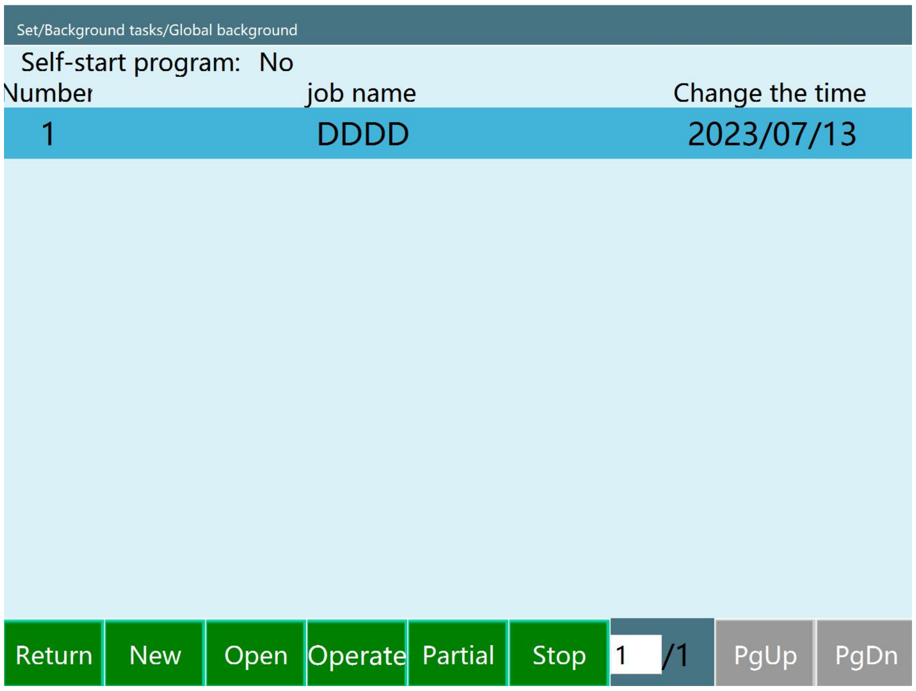
When editing a background task program, you want to debug a way that only provides a "STEP" run.To run debugging as a whole, insert the PTHREAD_START directive into the main program and run it to debug.

The background task is executed only once when it is started. If loop judgment is needed, it can be used with the WHILE instruction.

11.2. Global background task programming

Programs that need to run in background tasks need to be done in Settings - Background Tasks and are programmed the same as local background.Setting the global background mode:

1. Edit the global background program
2. Click Operation - Set to Start in the global background operation area
3. Above the program list: "Start up: program name"
4. Start immediately Click the Start button in the operation area. Otherwise, it will start automatically after the restart.


Set/Background tasks/Global background

Self-start program: No

Number	job name	Change the time
1	DDDD	2023/07/13

You can view the global background running status in the Monitor-Running-Background task

11.3. Main program programming

To run background tasks in the main program, you need to insert the PTHREAD_START (start thread) instruction into the program.To exit a background task, insert the PTHREAD_END (exit thread) directive.

Background tasks start only after running PTHREAD_START, and the background program does not pause when the main program pauses.Background task stop conditions:

1. The program runs to the PTHREAD_END instruction;
2. The program stops and the robot stops enabling.
3. The background task runs until the END of the end line.

11.4. Program control class instruction

11.4.1. PTHREAD_START(Start thread)

Running the PTHREAD_START command starts background tasks.This instruction is located in the program control class instruction.

When inserting this command, click the input box of [Value], and the list of established background tasks will automatically pop up. Select the background tasks to be run, and click the button of "OK" to select this program.

When the main program runs, the background task starts when the PTHREAD_START instruction is run.

11.4.2. PTHREAD_END(Close the thread)

Running the PTHREAD_END directive exits the corresponding background task that has been run, and inserts and modifies in the same way as PTHREAD_STAR.

11.4.3. PAUSERUN(Pause thread)

Running PAUSERUN suspends all tasks, the main program, or background tasks.

When inserting this command, click the drop-down box of type [Value], select the type of control, click the input box of program [Value], the established list of background tasks will automatically pop up, select the background tasks that need to be run, click the button of "OK", the program will be selected, all and main programs cannot be selected.

Caution

Press the stop button of the teaching box to pause the main program only.

When running the program, all tasks, main program, or background tasks are paused when the PAUSERUN command is run.

11.4.4. CONTINUERUN(Continue thread)

Running CONTINUERUN commands continues to run the main program or background tasks.

When inserting this command, click the drop-down box of type [Value], select the type of control, click the input box of program [Value], the established list of background tasks will automatically pop up, select the background tasks to be run, click the button of "OK", the program will be selected, and the main program cannot be selected.

When CONTINUERUN is run, the main program or background tasks continue to run.

11.4.5. STOPRUN(Stop running)

Running the STOPRUN command stops all tasks

This instruction can be directly clicked to confirm insertion, no need to set parameters.

11.4.6. RESTARTRUN(Rerun)

Running the RESTARTRUN instruction reruns all tasks.

This instruction can be directly clicked to confirm insertion, no need to set parameters

11.5. Instructions supported by background tasks

Currently, only the following instructions are supported in the program of background tasks:

category	instruction	content
Input and output class	DIN	IO input
	DOUT	IO output
	AIN	Analog input
	AOUT	Analog output
	READ_DOUT	Read output
Timer class	TIMER	Delay
Operation class	ADD	plus
	SUB	Less
	MUL	Multiply
	DIV	except
	MOD	mold
	SIN	Sine
	COS	Cosine
	ATAN	Arctangent
	LOGICAL_OP	logic operation
	IF	in case

Condition control class	ELSEIF	Otherwise if
	ELSE	otherwise
	WAIT	wait
	WHILE	Inner loop
	LABEL	label
	JUMP	Jump
	CLKSTART	Timing begins
	CLKSTOP	Time ends
	CLKRESET	Timer reset
Variable class	SETINT	Assigned integer
	SETDOUBLE	Assign floating point
	SETBOOL	Assign Boolean
	FORCESET	Write to file
Communication class	SENDMSG	send data
	PARSEMSG	Analytical data
	READCOMM	Read
	OPENMSG	Open data
	CLOSEMSG	Close data
	PRINTMSG	Output Data
	MSG_CONN_ST	Get information connection status
Position variable class	USERFRAME_SET	User coordinate modification
	TOOLFRAME_SET	Tool coordinate modification
	READPOS	Read point
	POSADD	Point plus
	POSSUB	Point minus

12. IO、Modbus and remote function

12.1. IO - input/output instruction

12.1.1. DIN

This instruction is used to read the numeric input state into a variable, which can be a local or global integer variable (INT, GINT) or a local or global floating point variable (DOUBLE, GDOUBLE).

Variable type: The type of the variable used to store the input state. It can be INT, GINT, DOUBLE, or GDOUBLE.

Variable name: Name of the variable that stores the input status, such as I001 and GD002. This variable must be defined in advance.

Input group number: You can set the 1/4/8 input status to be read simultaneously.

At this time, route 1 is group 1, and groups 1-16 correspond to ports 1-16 respectively.

In this case, each four channels is a group, that is, ports 1-4, ports 5-8, ports 9-12, and ports 13-16 are 1-4 respectively. In this case, group number can be 1-4. If you want to read the input status of ports 5-8 at the same time, you can enter group number 2.

IG#-8 input, each 8 channels is a group, that is, 1-8 is group 1, 9-16 is group 2.If you want to read the input status of ports 9-16 at the same time, enter 2 for group number.

If multiple ports are read at the same time, the port status is converted to a decimal number and stored in the variable.For example, if ports 9-16 are read and eight ports exist at the same time, their states are as follows

16	15	14	13	12	11	10	9
0	1	1	0	1	0	0	1

The binary value is 01101001, and the decimal value is 105.

Then I #(2)105 is saved in the system

12.1.2. DOUT

This instruction is used to output digital signals through the digital IO board.

Output Group number: Output 1/4/8 I/OS at the same time.

At this time, route 1 is group 1, and groups 1-16 correspond to ports 1-16 respectively.

When OGH#-4 is output, each 4 channels is in a group, that is, ports 1-4, ports 5-8, ports 9-12, and ports 13-16 are in groups 1-4. In this case, set group number to 1-4. If ports 5-8 are output at the same time, set group number to 2.

OG#-8 channel output, at this time every 8 channels is a group, that is, 1-8 is group 1, 9-16 is group 2.If you want to output ports 9-16 at the same time, enter 2 for group number.

Output value: You can choose to choose by yourself or output by variable.

If you select Select by Yourself, select the status of each port in each group of I/OS. If you select the status of each port, the output value is 1 and if you do not select the status, the output value is 0.

If you choose to output through a variable, the variable value will be converted from base 10 to base 2 when output, as in DIN.

Time: Wait for the specified time after instruction execution, and then take the inverse output.

12.1.3. AIN

This instruction is used to read the single port input value of the analog IO board into a variable.

Analog input port: Select the input port to be read;

Source of variable values: Please select global floating point GDOUBLE or local floating point DOUBLE variable;

Variable name: Select the name of the variable that you want to read, for example, GD001.

12.1.4. AOUT

This instruction is used to set the output value of a single port on the analog IO board.Output values can be floating-point numbers.

12.1.5. PULSEOUT

This instruction is used to control the pulse output of the IO board that supports PWM.

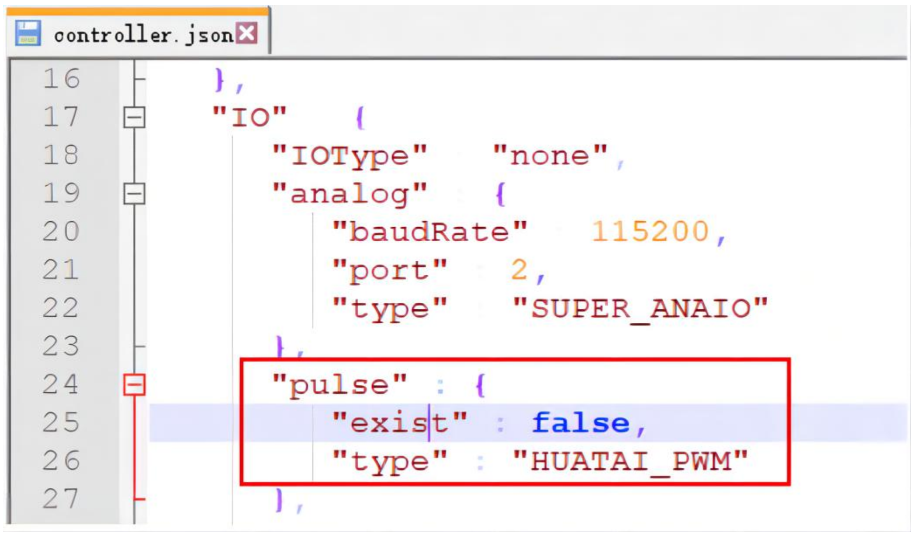
Number: the total number of pulses output;

Frequency: pulse output frequency;For example, if the default value is 100, the 1s output is 100

The IO boards that support this feature are as follows: Huatai IOPWM

INEXBOT R1 PWM

How to use:



```
    },
    "IO"    {
        "IOType"    "none",
        "analog"    {
            "baudRate"    115200,
            "port"    2,
            "type"    "SUPER_ANAIO"
        },
        "pulse" : {
            "exist" : false,
            "type" : "HUATAI_PWM"
        },
```

Modify the configuration file controller.json.

Find the exist parameter in "IO" - "pulse" and change it to "turn";

turn: function available;

false: The function is disabled.

Find the type parameter in "IO" - "pulse" and change it to the corresponding IO board.

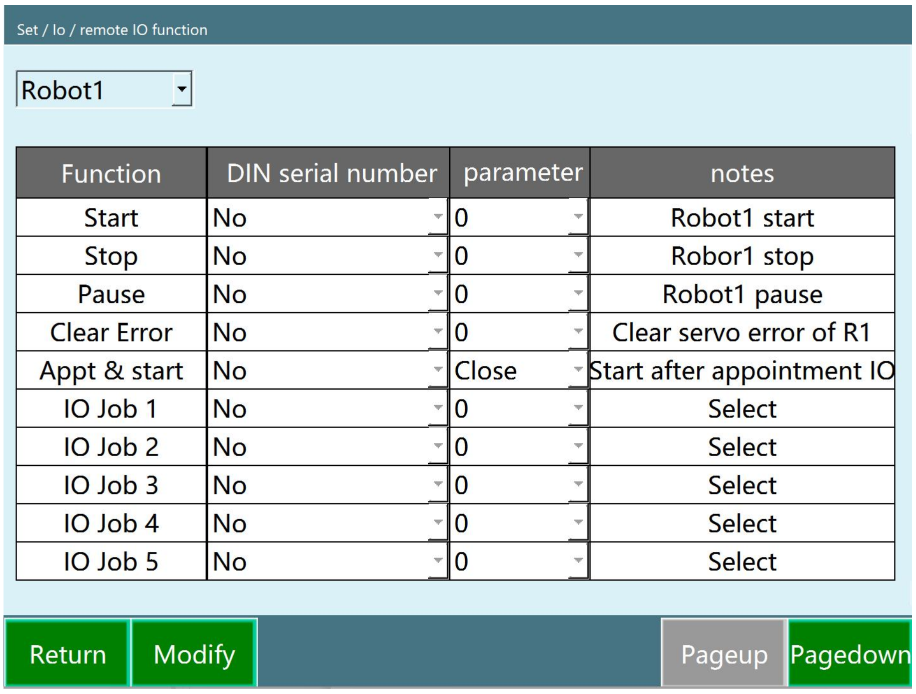
HUATAI_PWM: Huatai_IO

12.1.6. READDOUT

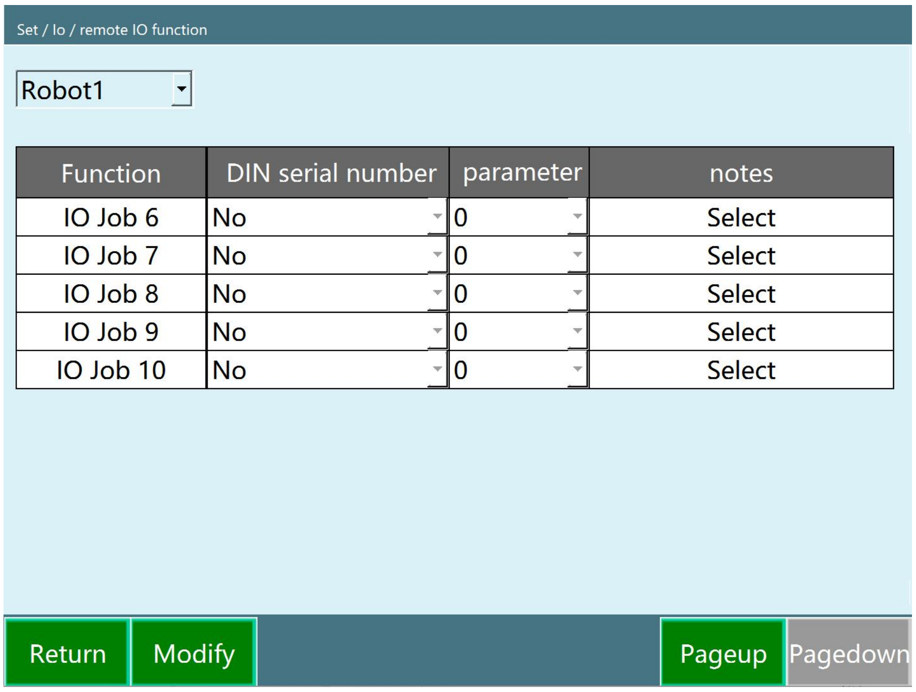
This instruction is used to read the output state of the current digital quantity IO board into a variable.It is used in the same way as DIN, except that the output state is read.

12.2. IO function selection Settings

In Setting /IO/ Function Selection, you can set the corresponding level of the I/O port corresponding to the remote IO control start, stop, pause, emergency stop, clear alarm and other functions, and set the program run by the IO module.



Function	DIN serial number	parameter	notes
Start	No	0	Robot1 start
Stop	No	0	Robor1 stop
Pause	No	0	Robot1 pause
Clear Error	No	0	Clear servo error of R1
Appt & start	No	Close	Start after appointment IO
IO Job 1	No	0	Select
IO Job 2	No	0	Select
IO Job 3	No	0	Select
IO Job 4	No	0	Select
IO Job 5	No	0	Select


Set / Io / remote IO function

Robot1

Function	DIN serial number	parameter	notes
IO Job 6	No	0	Select
IO Job 7	No	0	Select
IO Job 8	No	0	Select
IO Job 9	No	0	Select
IO Job 10	No	0	Select

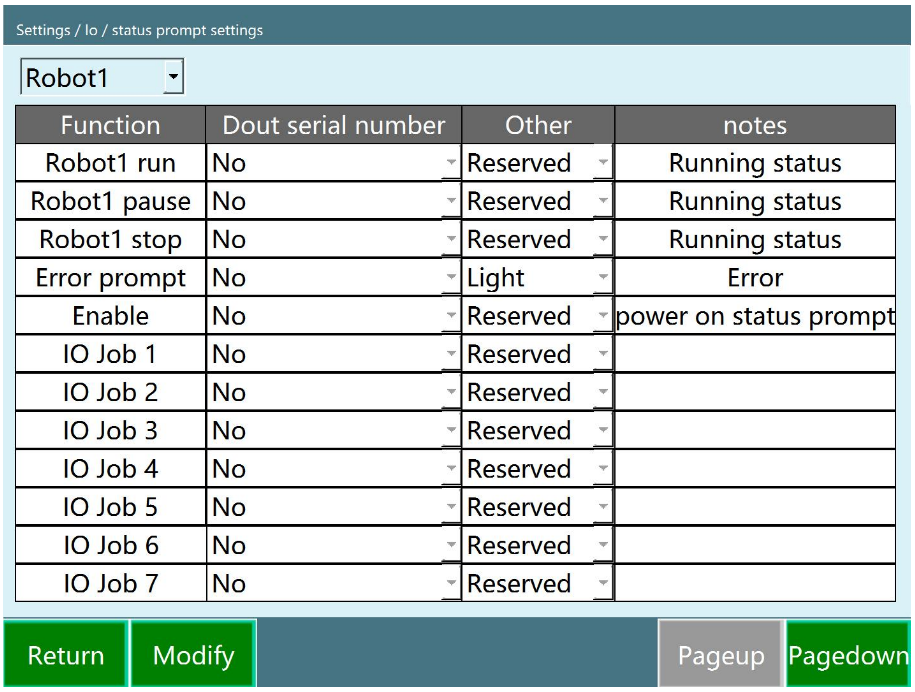
The program of the IO module can only select the program that has been set in the Remote Program Settings interface.

A maximum of 10 remote reservation programs are supported

Start with appointment: After opening, the first scheduled program is powered on immediately after the successful appointment, and other programs can be booked at this time.

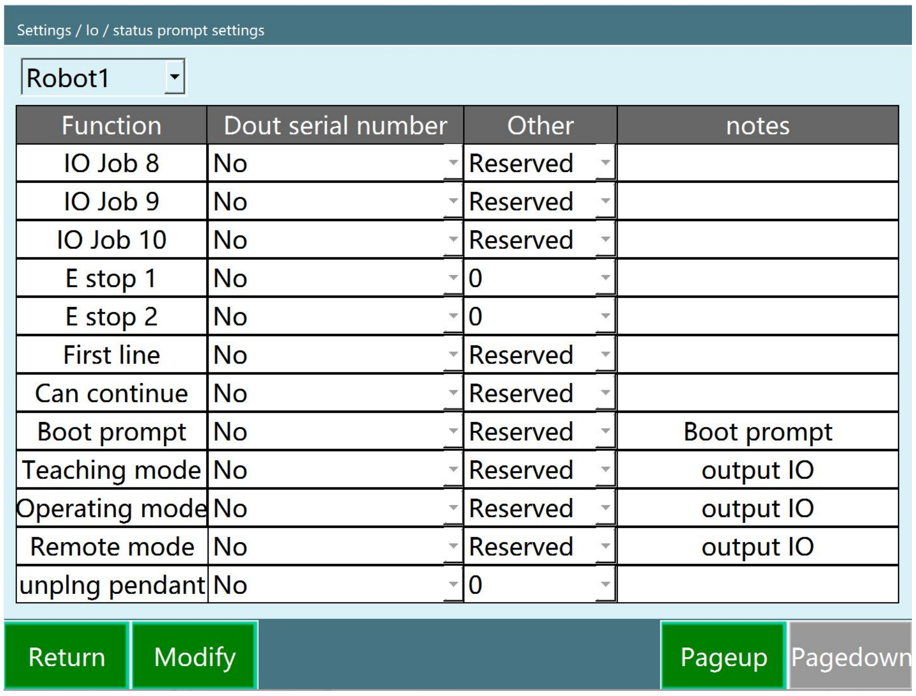
12.3. IO status prompt setting

On the status prompt setting screen, you can set the level of the I/O port corresponding to the startup prompt, robot running status, error prompt, enable, mode status reservation status, and emergency stop functions.


Settings / Io / status prompt settings

Robot1

Function	Dout serial number	Other	notes
Robot1 run	No	Reserved	Running status
Robot1 pause	No	Reserved	Running status
Robot1 stop	No	Reserved	Running status
Error prompt	No	Light	Error
Enable	No	Reserved	power on status prompt
IO Job 1	No	Reserved	
IO Job 2	No	Reserved	
IO Job 3	No	Reserved	
IO Job 4	No	Reserved	
IO Job 5	No	Reserved	
IO Job 6	No	Reserved	
IO Job 7	No	Reserved	



Function	Dout serial number	Other	notes
IO Job 8	No	Reserved	
IO Job 9	No	Reserved	
IO Job 10	No	Reserved	
E stop 1	No	0	
E stop 2	No	0	
First line	No	Reserved	
Can continue	No	Reserved	
Boot prompt	No	Reserved	Boot prompt
Teaching mode	No	Reserved	output IO
Operating mode	No	Reserved	output IO
Remote mode	No	Reserved	output IO
unplng pendant	No	0	

Robot1 operation: When robot 1 is running, it corresponds to the high output level of DOUT port

Robot1 pause: When robot 1 pauses, the corresponding DOUT port outputs high level

Robot1 stop: When robot 1 stops, the corresponding DOUT port outputs high level

Error prompt: When robot servo error is reported, it corresponds to the output of DOUT port, which can be set to be steady on or blinking

Enable: Output high level when the robot is powered on

Remote IO program 1-10 Output: remote IO program after reservation blinking, the IO program is always bright during operation.

Emergency stop 1: Forbid output high level or low level after the stop signal is triggered. You can set it by yourself.

Emergency stop 2: forbid output high level or low level after the stop signal is triggered. You can set it by yourself.

First line of main program: Output a signal with a high level parameter of 1, and the program cursor jumps to the first line of main program

Continuable execution: Output a signal with a high level parameter of 1 to run the paused program

Boot prompt: Controller boot output state, boot output high level

Teaching mode: High output level in teaching mode

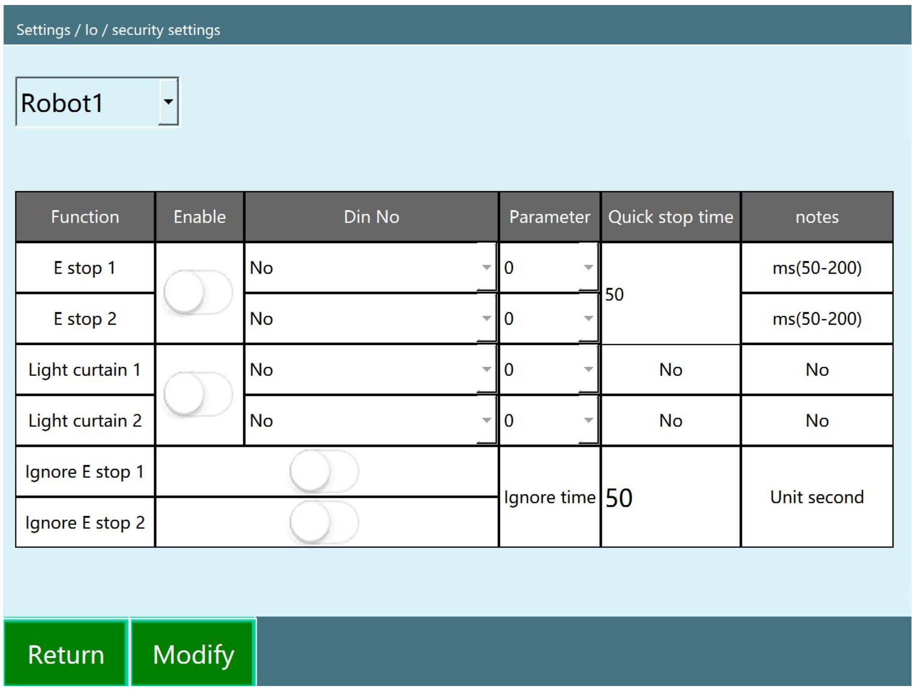
Running mode: Output high level in running mode

Remote mode: Output high level in remote mode

Unplug teaching box: After unplug teaching box, output high level or low level, which can be set by yourself.

12.4. IO security Settings

On the security Settings screen, you can set the level of the I/O port corresponding to functions such as emergency stop and safety light screen.After the I/O emergency stop is lifted, you need to click the clear button to clear the error before performing other operations.


Settings / Io / security settings

Robot1

Function	Enable	Din No	Parameter	Quick stop time	notes
E stop 1		No	0	50	ms(50-200)
E stop 2		No	0		ms(50-200)
Light curtain 1		No	0	No	No
Light curtain 2		No	0	No	No
Ignore E stop 1			Ignore time	50	Unit second
Ignore E stop 2					

Emergency stop: After the emergency stop signal is triggered, the robot is powered off and switched to servo stop

Safety light curtain: Trigger the safety light curtain robot to pause, press the start button again to continue running

Shielding emergency stop: After opening the shielding time, the emergency stop signal is shielded

12.5. IO reset

When the program stops running or an error is reported, the IO reset function can restore the IO output port to the initial state.I/O reset can be divided into three types: I/O reset, cutting mode stop and program error stop.

IO reset: In remote mode, when the reset signal is given, the robot will perform the action of returning to the reset point, and the IO port set on the interface will be reset to the reset value.

Switch mode stop: When the program is running, switching mode to teach or remote mode will cause the program to stop, and the I/O port set on the interface will be reset to the reset value.

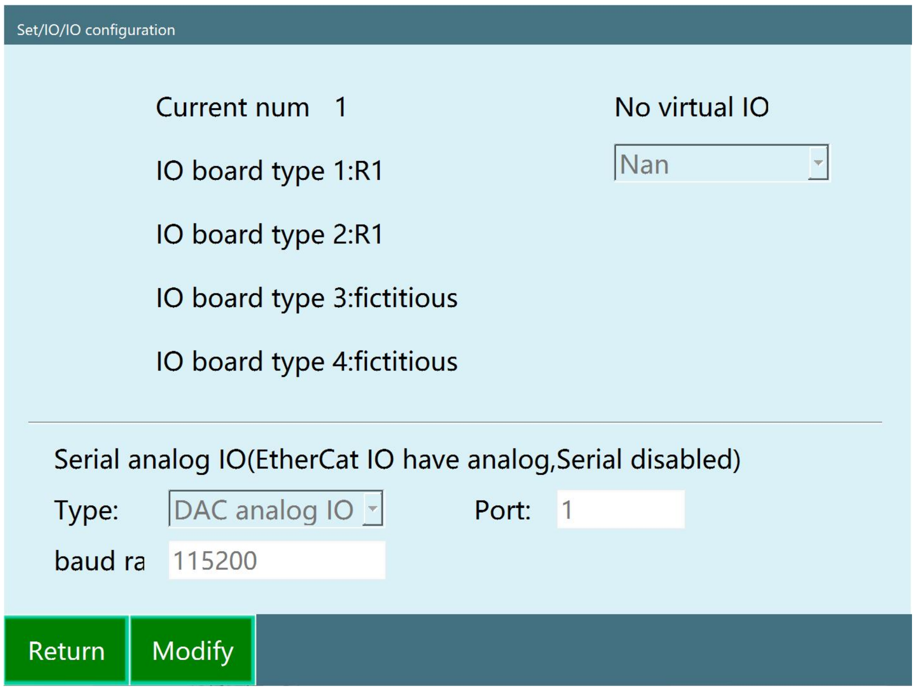
Program error stop: Program error (such as the server error) causes the program to stop, the I/O port set on the interface will be reset to the reset value.

Use steps:

1. The IO reset page is displayed.
2. Select the robot;
3. Click to enter the reset scenario (IO reset, cut mode stop, program error stop);
4. Select the IO board.
5. Turn on the Reset switch corresponding to the IO port to be reset.
6. Select the reset value (0/1), where 0 is low and 1 is high.

12.6. IO configuration

The system automatically identifies the IO model according to the hardware connection sequence.You can view the number and model of IO boards.Go to 【 Settings 】-【IO】-【IO Configuration 】.The input box is gray and no value can be entered.



Set/IO/IO configuration

Current num 1

No virtual IO: Nan

IO board type 1:R1

IO board type 2:R1

IO board type 3:fictitious

IO board type 4:fictitious

Serial analog IO(EtherCat IO have analog,Serial disabled)

Type: DAC analog IO

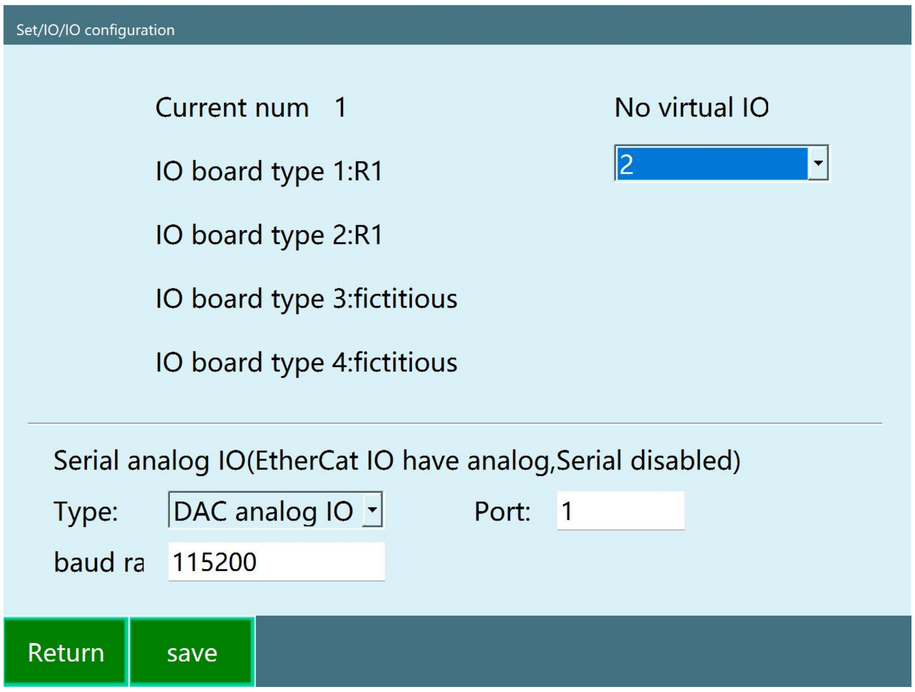
Port: 1

baud ra 115200

After you click Modify, the Modify button changes to save, and the number of virtual IO boards drop-down box selects the desired virtual IO

⚠ Caution

Virtual IO is only used for program debugging and program demonstration, and there is no IO signal access



Set/IO/IO configuration

Current num 1

No virtual IO

2

IO board type 1:R1

IO board type 2:R1

IO board type 3:fictitious

IO board type 4:fictitious

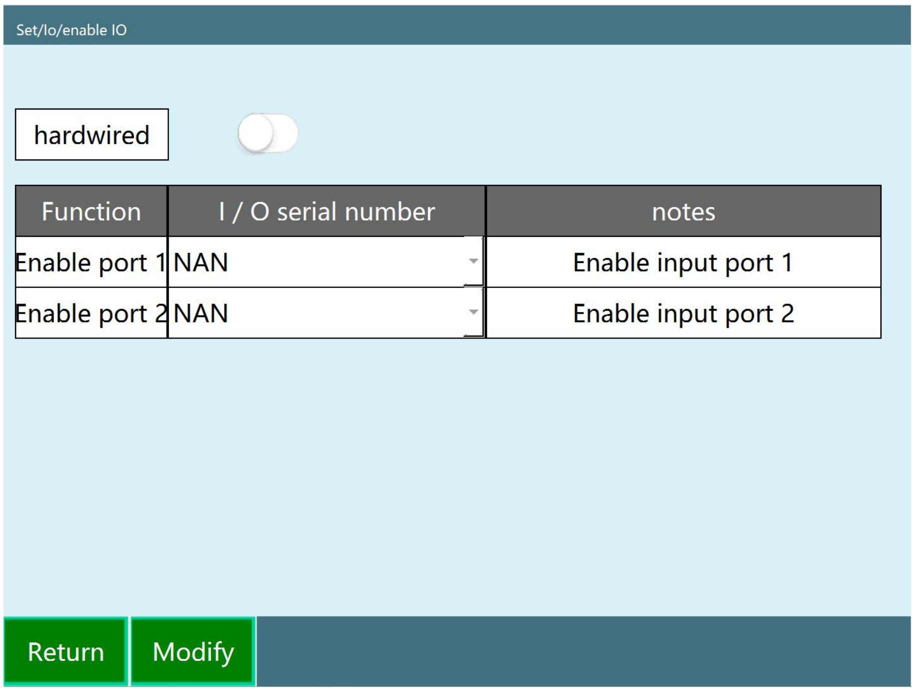
Serial analog IO(EtherCat IO have analog,Serial disabled)

Type: DAC analog IO　Port: 1

baud ra 115200

12.7. Enable IO

If the enable hardwired teaching box is used, after connecting the cable, select the corresponding DIN port on this page and turn on the enable switch. The enable function is controlled by the input signal of the IO board;Do not set the hard-wired teaching box if it is not enabled.After this function is enabled, the enable button of the teaching box becomes invalid and cannot be used.



Set/Io/enable IO

hardwired

Function	I / O serial number	notes
Enable port 1	NAN	Enable input port 1
Enable port 2	NAN	Enable input port 2

12.8. Alarm message

This function can customize the IO input and output port alarm content, alarm information priority is higher than other types of IO alarm information.

For example, set the I/O emergency stop signal port to 15 for receiving anti-collision I/OS. 1 triggers and 0 disengages.Then trigger DIN15 will report "Robot 1IO emergency stop is triggered";At this time, find DIN15 in the alarm message interface, and enter "trigger anti-collision" in the message bar, then trigger DIN15 again and the error "trigger anti-collision" is displayed.

12.9. Port name

A port name can contain a maximum of 5 Chinese characters or 10 English characters. After the port name is set, the port name is automatically displayed when you select the I/O port drop-down list.

12.10.Remote mode I/O reservation Description

12.10.1. Signal specification

	Feature	Support mode	Trigger/output mode	Instructions
Digital IO input	activate	Remote mode	Rising edge	When the parameter is 1, the signal is valid when 0 changes to 1
	stop	Remote mode	Rising edge	When the parameter is 1, the signal is valid when 0 changes to 1
	Pause	Remote mode	Rising edge	When the parameter is 1, the signal is valid when 0 changes to 1
	Clear alarm	Remote mode	Rising edge	When the parameter is 1, the signal is valid when 0 changes to 1
	Start by appointment	Remote mode	No	If the reservation is enabled, the system is powered on
	I/O programs 1-10	Remote mode	Pulse (period 0.6s)	When the parameter is 1, the signal is valid when 0-1-0, and the program reservation needs to be triggered for at least 0.6 seconds.
	Emergency stop 1	Teaching, running, remote	High level	Scan once every 1 millisecond and trigger when the scan is

	Emergency stop 2	Teaching, running, remote	Electric level	found
	Safety light Screen 1	Running (in operation), remote (in operation)	High level	
	Safety light Curtain 2	Running (in operation), remote (in operation)	High level	
	Shield emergency stop 1	Cooperate with emergency disuse	When the button is opened, the emergency stop function is shielded, and the emergency stop signal is detected again after the time is set	
	Mask emergency stop 2	Cooperate with emergency disuse		
Digital IO output	Boot prompt	No mode restriction	Steady on, output only at boot	Output high level
	Robot1 runs	Teaching, running, remote	Steady on	Output high level while the program is running
	Robot1 pauses	Teaching, running, remote	Steady on	Output high level when program pauses
	Robot1 stops	Teaching, running, remote	Steady on	Output high level when program stops
	Error prompt	No mode restriction	Steady on and blinking	Steady on Output High level blinking Output pulse (period 1s,0.5s on,0.5s off)
	enable	No mode restriction	Steady on	Output high level

	IO program 1-10 Scheduled output	Long range	Steady on and blinking	No light when not booked/booked;During reservation, the blinking period is 1.2s, on for 0.6s and off for 0.6s.It is always bright during operation, and the output level is high
	Emergency stop 1	Signal trigger time	High, low	When the parameter is 1, the output level is high
	Emergency stop 2	Signal trigger time		
	Pull out the teaching box	No mode restriction	High, low	Click to show the teaching box, output 1 or 0;If the parameter is set to 0, unplug the teaching box output 0, reconnect and set 1.
	executable	Signal trigger time	High level	Output a signal with a high level parameter of 1 to run the paused program
	Main program	Teaching, running, remote	High level	Output a signal with a high level parameter of 1, and the program cursor jumps to the first line of the main program

Caution

This section uses output 1 as an example of output high level.

12.10.2. Remote mode status description

Unscheduled: After entering the remote mode, the program is not scheduled, or the reservation is canceled after the reservation is made

Making a reservation: The reservation is successfully made

Running: The program is running display running

Reserved: When the program is finished or triggered to stop, the reservation is displayed

The speed can not be modified in remote mode, and the speed modification needs to be modified in advance in 【Settings - Robot parameters - Operation parameters】

12.10.3. Reservation procedure

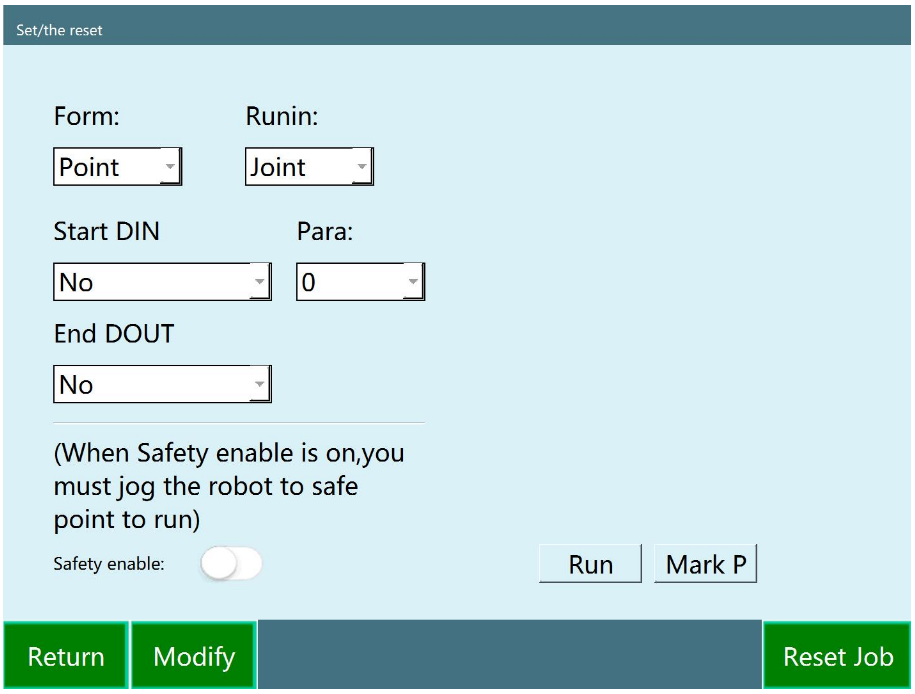
If the IO port of the corresponding program is triggered, the program is successfully booked. If the program is cancelled, the IO port of the corresponding program needs to be triggered again to start. The IO port of the corresponding trigger can be directly triggered to start the reservation.The start signal can not be set when booking to start.

12.10.4. Troubleshooting

After the IO function is set, go to Status-IO function status to check whether the IO function is set successfully or whether there are conflicting functions.

12.11.Reset point setting

The reset point function supports point-to-point, straight-line movement to a safe point, or custom reset trajectories and positions using reset programming instructions.


Set/the reset

Form: Point
Runin: Joint
Start DIN: No
Para: 0
End DOUT: No

(When Safety enable is on,you must jog the robot to safe point to run)

Safety enable:

Form: reset point, reset program;

Interpolation mode: joint, straight line;The motion speed of joint interpolation is 10% of the global velocity, and the motion speed of linear interpolation is 100mm/s.Reset program running speed is equal to instruction speed x status bar speed.

Safety enable: After opening, the program will determine whether the robot is at the reset point (safety point) position, and it must be at the reset point to continue to run the program;

Start DIN: reset point trigger signal;

Parameters: Reset point trigger signal 0 is valid or 1 is valid;

End DOUT: state signal output after recovery site;

Safety point range: the safety range error of each axis, the range is determined to be at the reset point (safety point);

Mark the point: Set the current robot coordinate as the reset point, and click "OK" to set it successfully.The set robot reset point can only run at the level corresponding to the port in the Remote Program Settings interface.

12.12.Description of remote mode control

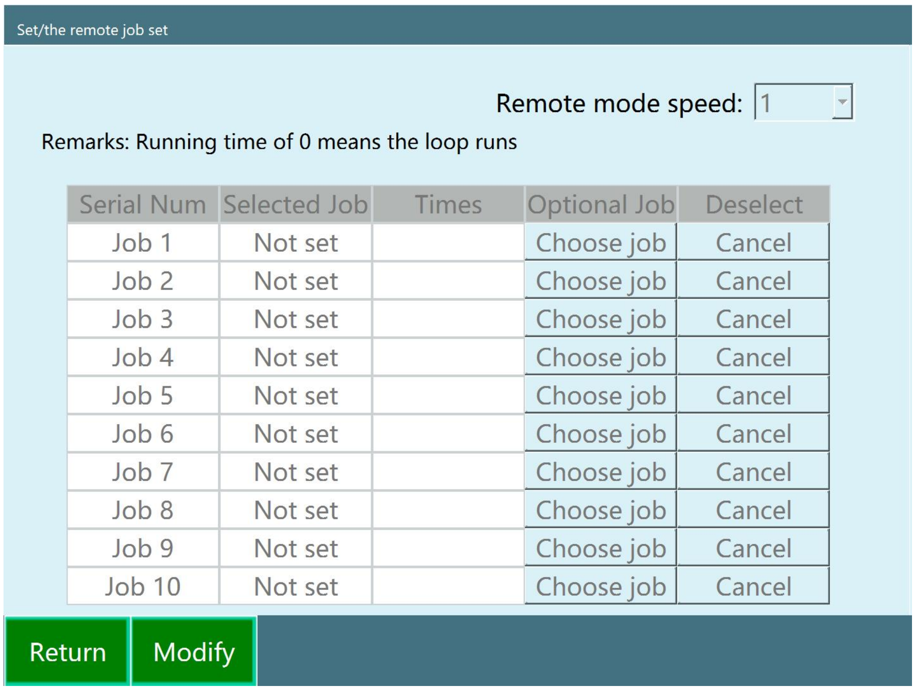
When the control system has an educator, touch screen and IO control device, the control priority is the educator > touch screen > IO control device.

After switching to remote mode, the control is switched to the touch screen.If there is no touch screen, switch to IO control.At this time, the interface of the instructor only displays the connection status between the Modbus module and the IO module and the IO program.

If both the touch screen and IO module are deployed, enable the IO module on the touch screen.

12.13.Remote IO control

12.13.1. Remote job set


Set/the remote job set

Remote mode speed: 1

Remarks: Running time of 0 means the loop runs

Serial Num	Selected Job	Times	Optional Job	Deselect
Job 1	Not set		Choose job	Cancel
Job 2	Not set		Choose job	Cancel
Job 3	Not set		Choose job	Cancel
Job 4	Not set		Choose job	Cancel
Job 5	Not set		Choose job	Cancel
Job 6	Not set		Choose job	Cancel
Job 7	Not set		Choose job	Cancel
Job 8	Not set		Choose job	Cancel
Job 9	Not set		Choose job	Cancel
Job 10	Not set		Choose job	Cancel

The remote program setting interface can set the programs used by the touch screen and the I/O control module.

If there are multiple robots, you can select the robot to be set at the robot, and set the programs of the robot.

The program used by the I/O control module must be set on the I/O function interface.

Click the Cancel button to cancel the selected program in the remote program interface.

The number of runs can be filled in the corresponding number, 0 represents cyclic operation.

12.14.Reservation mode

In the "Set/Operate Parameters";After the reservation mode is enabled, the remote IO program signal is triggered, the program reservation is successful, the startup signal is triggered, and the robot runs;After the reservation mode is disabled, the remote IO program signal is triggered, and the robot runs directly and other remote IO program signals are invalid at this time. The remote IO program signal can be triggered again after the robot runs.No need to set the start signal.

12.15.Use of Remote functions (IO)

12.15.1. Remote Function Overview

Set 10 remote programs and the number of runs of each program, queue 10 programs before running, run according to the order of the queue and the number of runs, stop waiting for queueing again after the completion of the queue.

12.15.2. Procedure for using the remote function

Write the program -- set up the remote program -- set up IO -- Switch to remote mode -- schedule sort -- Run

1. Write a program to create a new program and insert instructions, please ensure that the program can run normally.

2. Set up the remote program

Enter the "Settings - Remote program Settings" interface, set the program name and run times of program 1-program 10, if you want a single program to run in an infinite loop, set the run times of the program to 0.The program name here points to the program in the "Project" interface. When the instructions in the program are modified, the remote program will be automatically modified without resetting the remote program.If you have changed the program name, reset the program in the Remote program Settings screen.

3. Configure I/OS

On the IO-IO Functions page, set the I/O port and effective value for each function. If the effective value is 1, the I/O port takes effect when the effective value is 0, and the I/O port takes effect when the effective value is 0.

4. Switch to the remote mode

Rotate the Mode selection key to the Remote mode location or click the Mode state in the program to select Remote Mode.When the demonstrator is not connected to the controller, the startup controller automatically enters the remote mode.When the controller is connected to IO, Modbus device, and teach device at the same time, the priorities of the three devices are Teach device >Modbus device >IO device.When you switch to the remote mode, the Modbus device is valid, but the IO device is invalid. In this case, turn off the enable button in the Modbus device.

5. Sort by appointment

For example, the IO function in the IO function is set to

Running port 1 Valid value 1

Stop port 2 Valid value 1

Suspend port 3 The value is 1

Clear error port 4 Valid value 1

Program 1 Port 5 Valid value 1

Program 2 port 6 Valid value 1

Program 3 port 7 Valid value 1

Program 4 port 8 Valid value 1

Program 5 port 9 Valid value 1

Program 6 Port 10 Valid value 1

Program 7 Port 11 Valid value 1

Program 8 port 12 Valid value 1

Program 9 port 13 Valid value 1

Program 10 Port 14 Valid value 1

The order is to give port 6 a high level for 1 second and then release, then program 2 is in the first, give port 8 a high level for 1 second and release, program 4 is in the second, and so on.If you want to cancel the queue of a program in the queue, then give the corresponding IO port a high level for 1 second, and the program will be canceled in the queue.There can be only 10 programs in the queue, and the same program cannot be queued twice.When a program is running, you can add it back to the end of the queue.

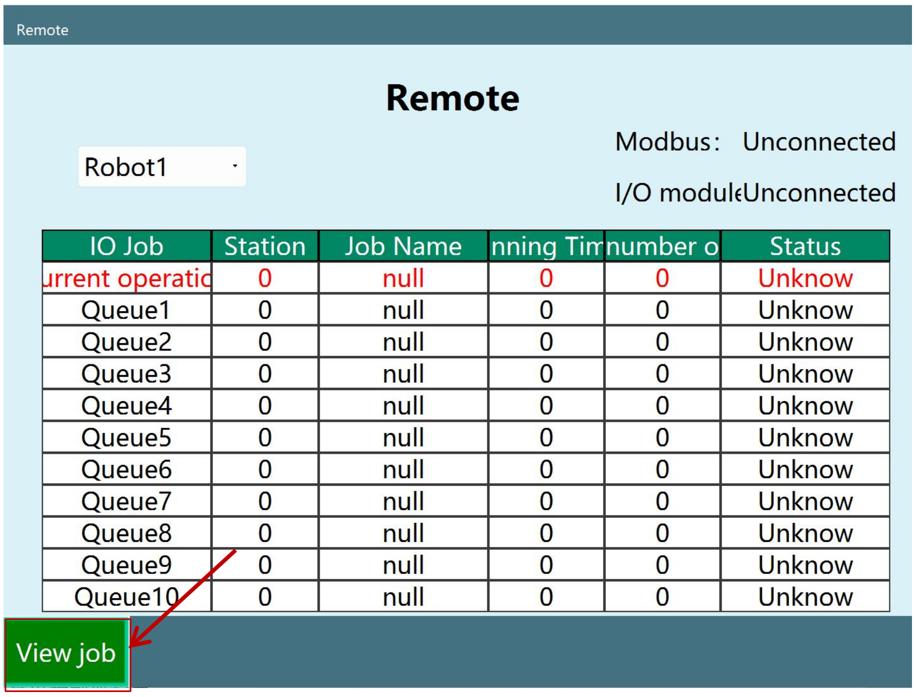
Step 6 Run

Give a high level to the functioning port, and the robot starts running according to the order and number of runs in the queue.After the completion of the operation, the servo is not powered off, and then the program is added to the queue, and the robot will immediately run the program.

When there is no program in the queue to make it run, then the robot is powered on and does not move, at this time the program is placed in the queue, and the robot immediately executes the program.

12.15.3. View operation

You can click the "View Program" button in the remote mode interface to view the details of the remote IO control program. modbus can also view the details through this function.



Modbus: Unconnected

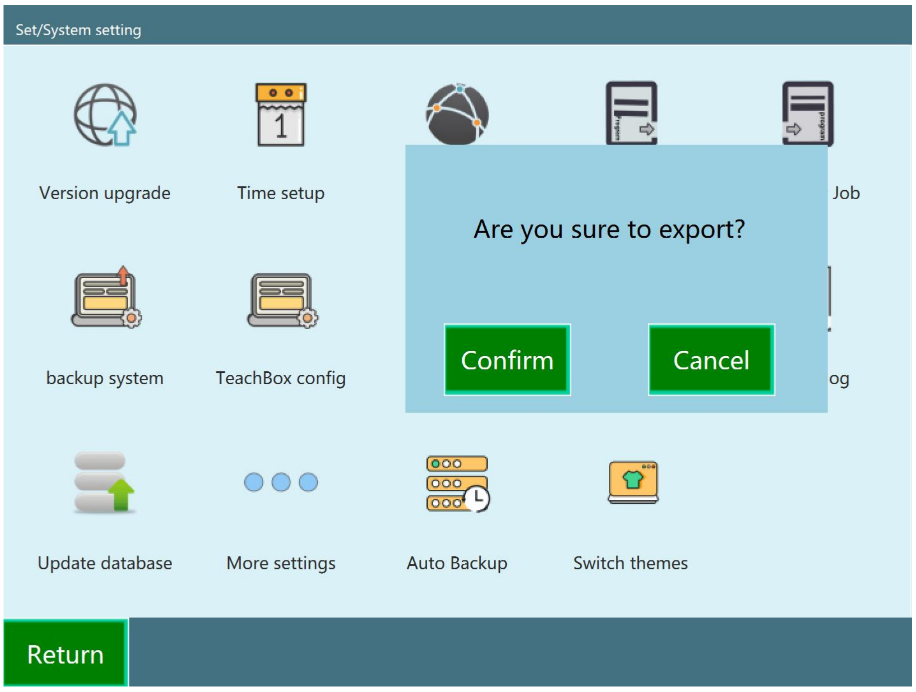
I/O module Unconnected

IO Job	Station	Job Name	nning Tim	number o	Status
urrent operatio	0	null	0	0	Unknow
Queue1	0	null	0	0	Unknow
Queue2	0	null	0	0	Unknow
Queue3	0	null	0	0	Unknow
Queue4	0	null	0	0	Unknow
Queue5	0	null	0	0	Unknow
Queue6	0	null	0	0	Unknow
Queue7	0	null	0	0	Unknow
Queue8	0	null	0	0	Unknow
Queue9	0	null	0	0	Unknow
Queue10	0	null	0	0	Unknow

12.16.MODBUS

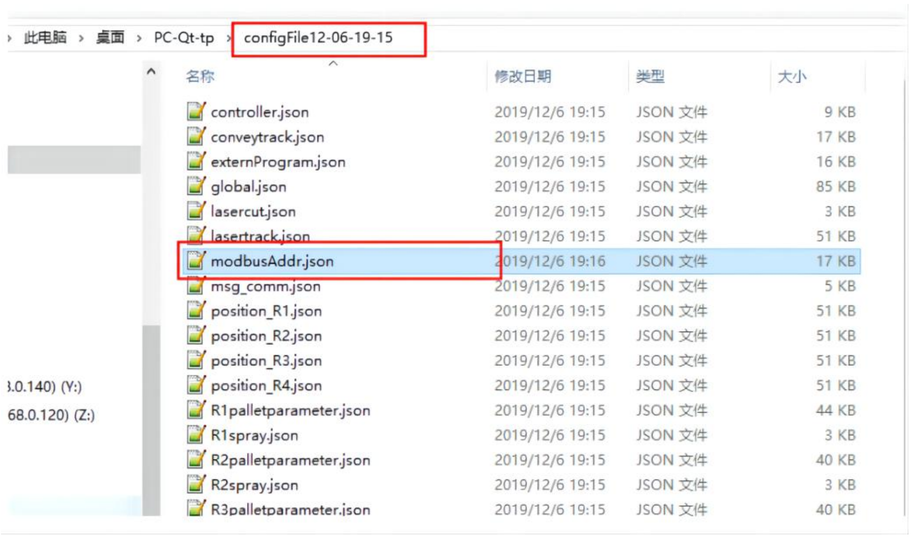
12.16.1. Modbus modifies the address code

1. Export the controller configuration



Are you sure to export?

2. Find the configuration filemodbusAddr.json， in the configFile+ date folder.

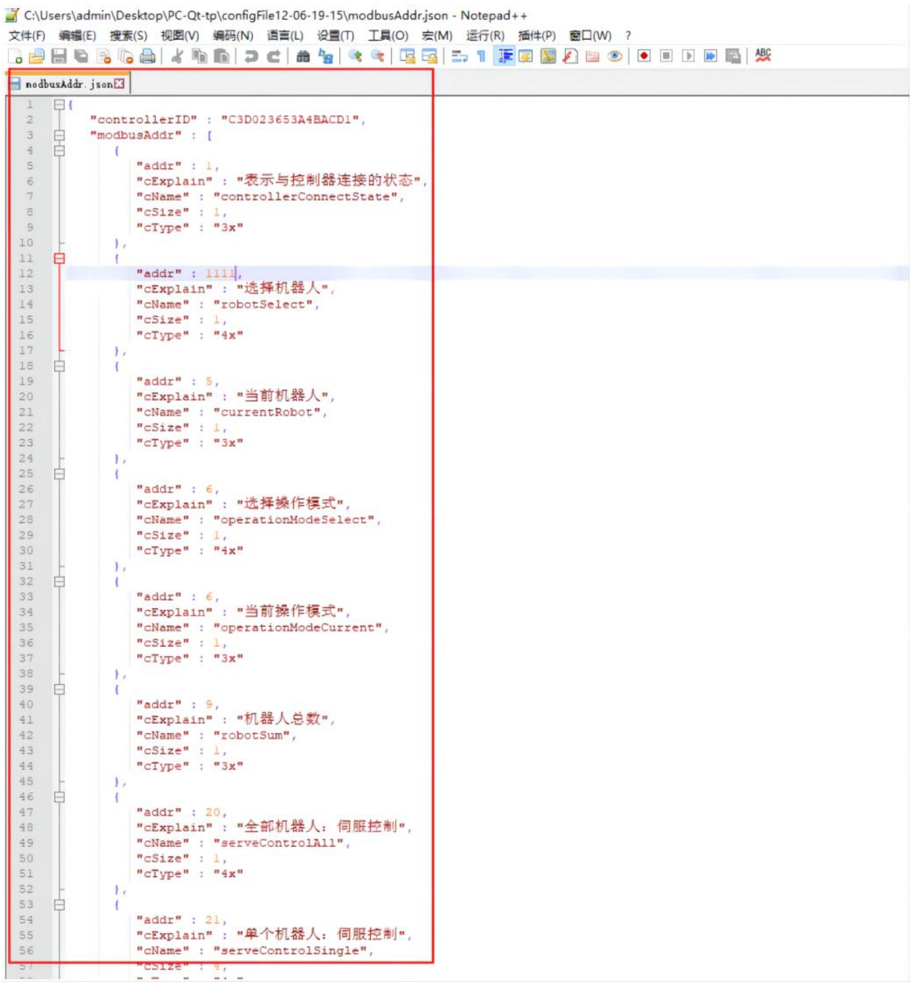


› 此电脑 › 桌面 › PC-Qt-tp › configFile12-06-19-15

名称	修改日期	类型	大小
controller.json	2019/12/6 19:15	JSON 文件	9 KB
conveytrack.json	2019/12/6 19:15	JSON 文件	17 KB
externProgram.json	2019/12/6 19:15	JSON 文件	16 KB
global.json	2019/12/6 19:15	JSON 文件	85 KB
lasercut.json	2019/12/6 19:15	JSON 文件	3 KB
lasertrack.json	2019/12/6 19:15	JSON 文件	51 KB
modbusAddr.json	2019/12/6 19:16	JSON 文件	17 KB
msg_comm.json	2019/12/6 19:15	JSON 文件	5 KB
position_R1.json	2019/12/6 19:15	JSON 文件	51 KB
position_R2.json	2019/12/6 19:15	JSON 文件	51 KB
position_R3.json	2019/12/6 19:15	JSON 文件	51 KB
position_R4.json	2019/12/6 19:15	JSON 文件	51 KB
R1palletparameter.json	2019/12/6 19:15	JSON 文件	44 KB
R1spray.json	2019/12/6 19:15	JSON 文件	3 KB
R2palletparameter.json	2019/12/6 19:15	JSON 文件	40 KB
R2spray.json	2019/12/6 19:15	JSON 文件	3 KB
R3palletparameter.json	2019/12/6 19:15	JSON 文件	40 KB

3.0.140) (Y:)
68.0.120) (Z:)

3. Open with text editing software such as Notepad++

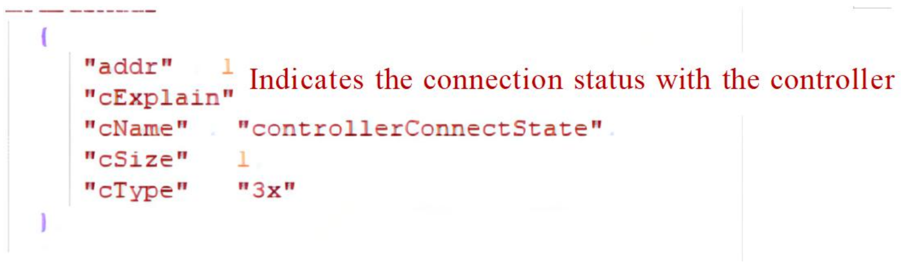


C:\Users\admin\Desktop\PC-Qt-tp\configFile12-06-19-15\modbusAddr.json - Notepad++

文件(F) 编辑(E) 搜索(S) 视图(V) 编码(N) 语言(L) 设置(T) 工具(O) 宏(M) 运行(R) 插件(P) 窗口(W) ?
modbusAddr.json

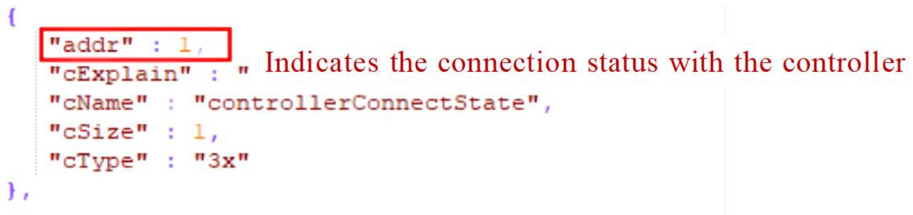
```
{
    "controllerID" : "C3D023653A4BACD1",
    "modbusAddr" : [
        {
            "addr" : 1,
            "cExplain" : "表示与控制器连接的状态",
            "cName" : "controllerConnectState",
            "cSize" : 1,
            "cType" : "3x"
        },
        {
            "addr" : 1111,
            "cExplain" : "选择机器人",
            "cName" : "robotSelect",
            "cSize" : 1,
            "cType" : "4x"
        },
        {
            "addr" : 5,
            "cExplain" : "当前机器人",
            "cName" : "currentRobot",
            "cSize" : 1,
            "cType" : "3x"
        },
        {
            "addr" : 6,
            "cExplain" : "选择操作模式",
            "cName" : "operationModeSelect",
            "cSize" : 1,
            "cType" : "4x"
        },
        {
            "addr" : 6,
            "cExplain" : "当前操作模式",
            "cName" : "operationModeCurrent",
            "cSize" : 1,
            "cType" : "3x"
        },
        {
            "addr" : 9,
            "cExplain" : "机器人总数",
            "cName" : "robotSum",
            "cSize" : 1,
            "cType" : "3x"
        },
        {
            "addr" : 20,
            "cExplain" : "全部机器人: 伺服控制",
            "cName" : "serveControlAll",
            "cSize" : 1,
            "cType" : "4x"
        },
        {
            "addr" : 21,
            "cExplain" : "单个机器人: 伺服控制",
            "cName" : "serveControlSingle",
            "cSize" : 4,
```

4. When opened, you can see a {...} contains a set of address code parameters. The system automatically generates an original address code



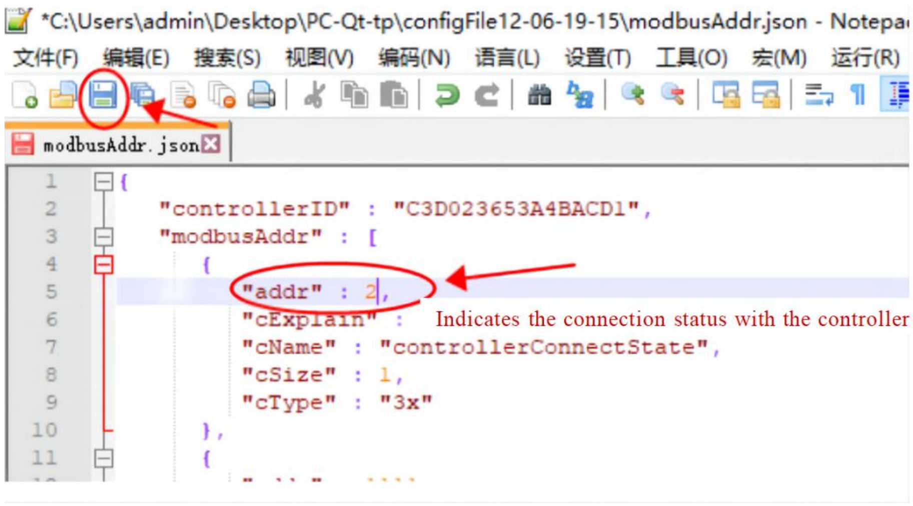
```
{
    "addr"      1  Indicates the connection status with the controller
    "cExplain"
    "cName"       "controllerConnectState"
    "cSize"       1
    "cType"       "3x"
}
```

5. Modify Address Change the number after the addr. If the number is 0, the address code function is invalid



```
{
    "addr" : 1,
    "cExplain" : " Indicates the connection status with the controller
    "cName" : "controllerConnectState",
    "cSize" : 1,
    "cType" : "3x"
},
```

6. Click Save when the modification is complete

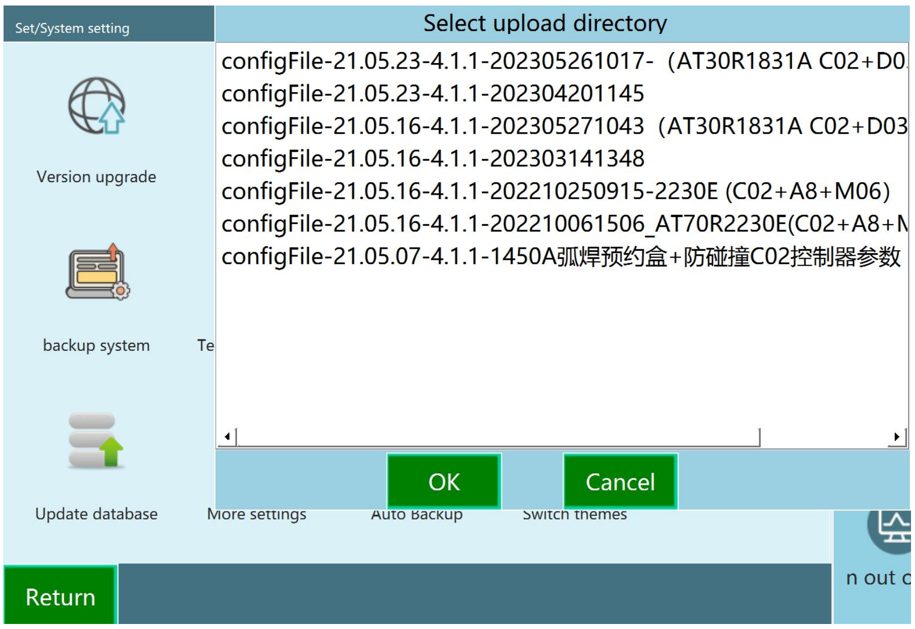


*C:\Users\admin\Desktop\PC-Qt-tp\configFile12-06-19-15\modbusAddr.json - Notepa

文件(F) 编辑(E) 搜索(S) 视图(V) 编码(N) 语言(L) 设置(T) 工具(O) 宏(M) 运行(R)
modbusAddr.json

```
{
    "controllerID" : "C3D023653A4BACD1",
    "modbusAddr" : [
        {
            "addr" : 2,
            "cExplain" :  Indicates the connection status with the controller
            "cName" : "controllerConnectState",
            "cSize" : 1,
            "cType" : "3x"
        },
        {
```

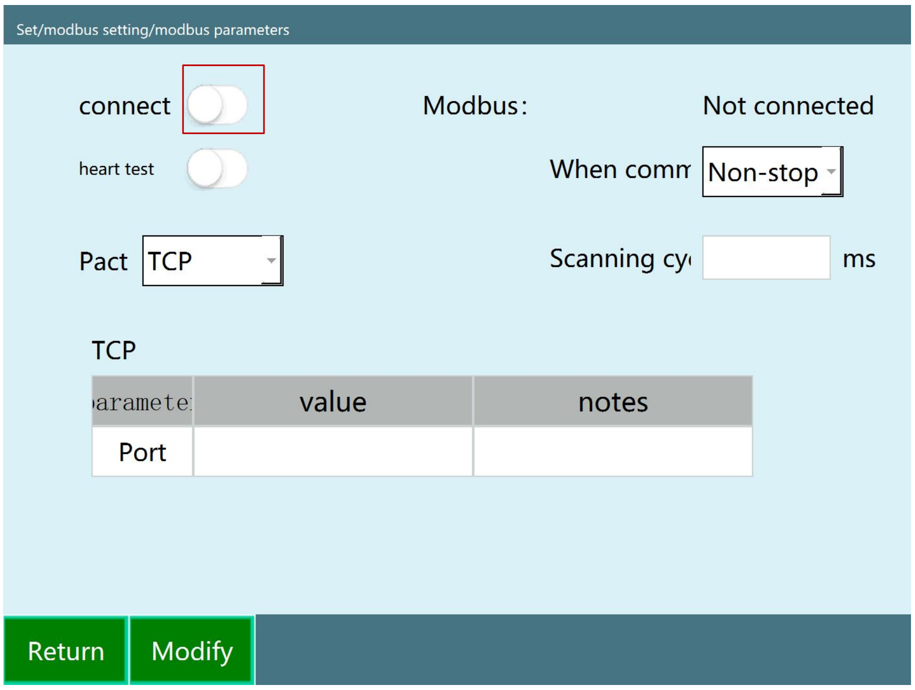
7. Then import the parameter to the controller again, and the restart takes effect



Select upload directory

configFile-21.05.23-4.1.1-202305261017- (AT30R1831A C02+D0
configFile-21.05.23-4.1.1-202304201145
configFile-21.05.16-4.1.1-202305271043 (AT30R1831A C02+D03
configFile-21.05.16-4.1.1-202303141348
configFile-21.05.16-4.1.1-202210250915-2230E (C02+A8+M06)
configFile-21.05.16-4.1.1-202210061506_AT70R2230E(C02+A8+N
configFile-21.05.07-4.1.1-1450A弧焊预约盒+防碰撞C02控制器参数

8. The modification takes effect after the parameters are restarted or the connection is re-opened. (The configuration file automatically restarts after being imported.)


Set/modbus setting/modbus parameters

connect

heart test

Modbus: Not connected

When comm Non-stop

Pact TCP

Scanning cy ms

TCP

parameter	value	notes
Port		

12.17.Use of Modbus

12.17.1. Overview of ModBus functions

Modbus function can replace part of the teaching box function, remote control robot operation, teaching, viewing status and so on.

Modbus Supported modbusTCP and modbusRTU protocols.

Modbus has two modes: teach and run.For details about the address code, see MODBUS Address Code List V20.02.xls.

12.17.2. Procedure for using the Modbus touch screen

This section uses Verenton touch screen and modbusTCP protocol as an example.The touchscreen model is MT6071iP.

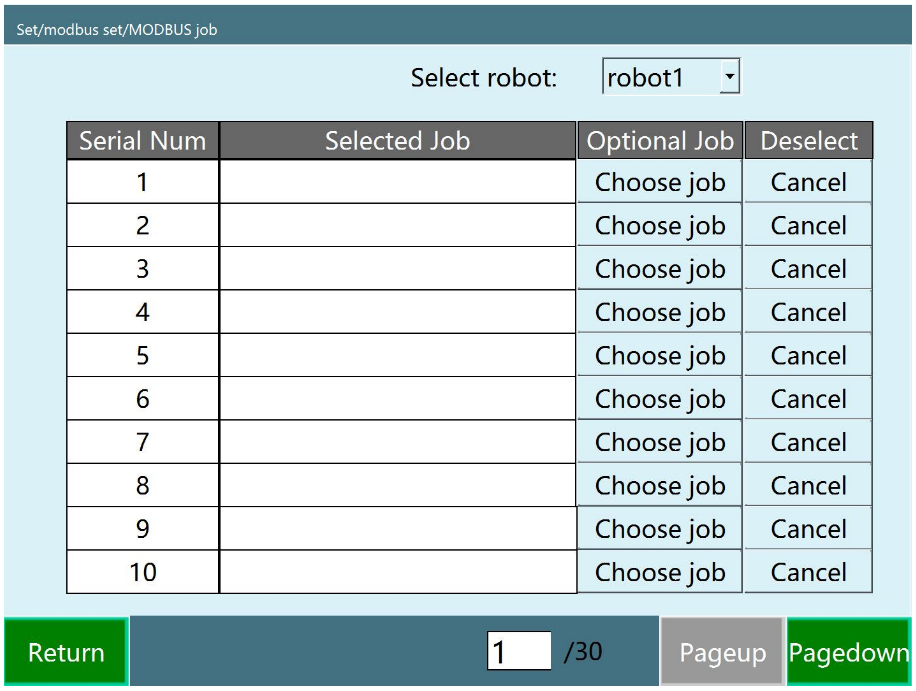
Write the program -- set the Modbus program -- set the Modbus parameters -- Switch to remote mode -- touch screen ready -- Select the program -- Run

1. Write the program

Use the instructor to write the program, to ensure that it can run normally.

2. Set the Modbus program

Set the program in "Settings -Modbus Settings -Modbus program", and the program name will be displayed in the list of selected programs after successful setting. A total of 300 programs can be set.


Set/modbus set/MODBUS job

Select robot: robot1

Serial Num	Selected Job	Optional Job	Deselect
1		Choose job	Cancel
2		Choose job	Cancel
3		Choose job	Cancel
4		Choose job	Cancel
5		Choose job	Cancel
6		Choose job	Cancel
7		Choose job	Cancel
8		Choose job	Cancel
9		Choose job	Cancel
10		Choose job	Cancel

3. Set Modbus parameters

In Settings -Modbus Settings -Modbus Parameters, set the protocol to TCP, set the controller as the master/slave station to the slave station, do not change the IP address, set the port to 502, and enable the connection.The effect takes effect after the controller is restarted.

12.18.Modbus Parameter Description

Connection: After Modbus Settings are complete, open the connection button and view the connection status on the right.

Protocol: TCP or RTU.

Master/slave: master station, slave station.

TCP parameter

IP: indicates the IP address of the Modbus device. This parameter is valid only when it is set to the primary station.

Port: Modbus device port

RTU parameter

Slave station ID: The default value is 1

Port: Serial port number of the controller

Baud rate: Set the baud rate corresponding to the touch screen

1. Switch to the mode

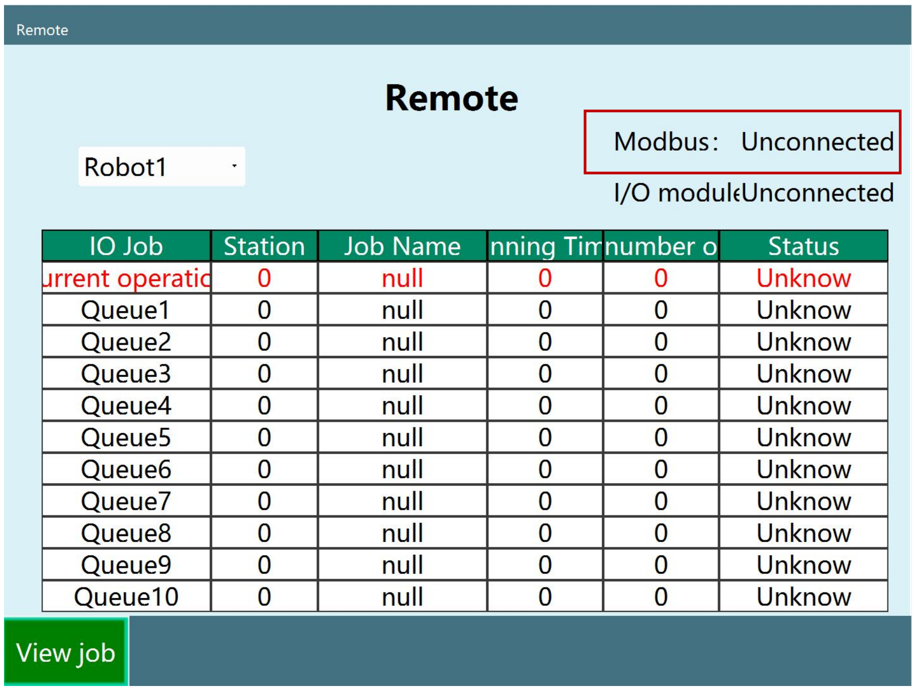
Rotate the Mode selection key to the Remote mode location or click the Mode state in the program to select Remote Mode.

Note: When the controller is connected to IO, Modbus device, and demonstrator at the same time, the priority of the three devices is demonstrator >Modbus device >IO device.When you switch to the remote mode, the Modbus device is valid, but the IO device is invalid. In this case, turn off the enable button in the Modbus device.

2. Prepare the touch screen

Connect the RJ45 network port on the touch screen, the teach device network port, and the Teach Box network port on the controller to the same switch.Touch screen Connects to controller IP address: 192.168.1.13, port: 502.

After the touch screen program is edited and run, modbus is not connected to the remote interface of the teaching box, and modbus is connected.



Remote

Remote

Robot1

Modbus： Unconnected

I/O modulєUnconnected

IO Job	Station	Job Name	nning Tim	number o	Status
urrent operatic	0	null	0	0	Unknow
Queue1	0	null	0	0	Unknow
Queue2	0	null	0	0	Unknow
Queue3	0	null	0	0	Unknow
Queue4	0	null	0	0	Unknow
Queue5	0	null	0	0	Unknow
Queue6	0	null	0	0	Unknow
Queue7	0	null	0	0	Unknow
Queue8	0	null	0	0	Unknow
Queue9	0	null	0	0	Unknow
Queue10	0	null	0	0	Unknow

3. Select a program

Use the touch screen to write 1 to type 4x address code 45, robot 1 select Demo Program 1;

Use the touch screen to write 5 to 4x type address code 61, robot 1 set the number of runs to 5 (does not take effect), use the touch screen to write 1 to 4x type address code 71, and confirm the change of the number of runs (number 5 takes effect);

4. Run

Use the touch screen to write 1 to the 4x type address code 29, cut to the server ready;Use the touch screen to write 1 to type 4x address code 19 to run the job file.

Appendix 1 Controller connection diagram

1. Controller

The controller is an important part of the robot, which is mainly used for the control of the robot and the completion of specific work tasks.

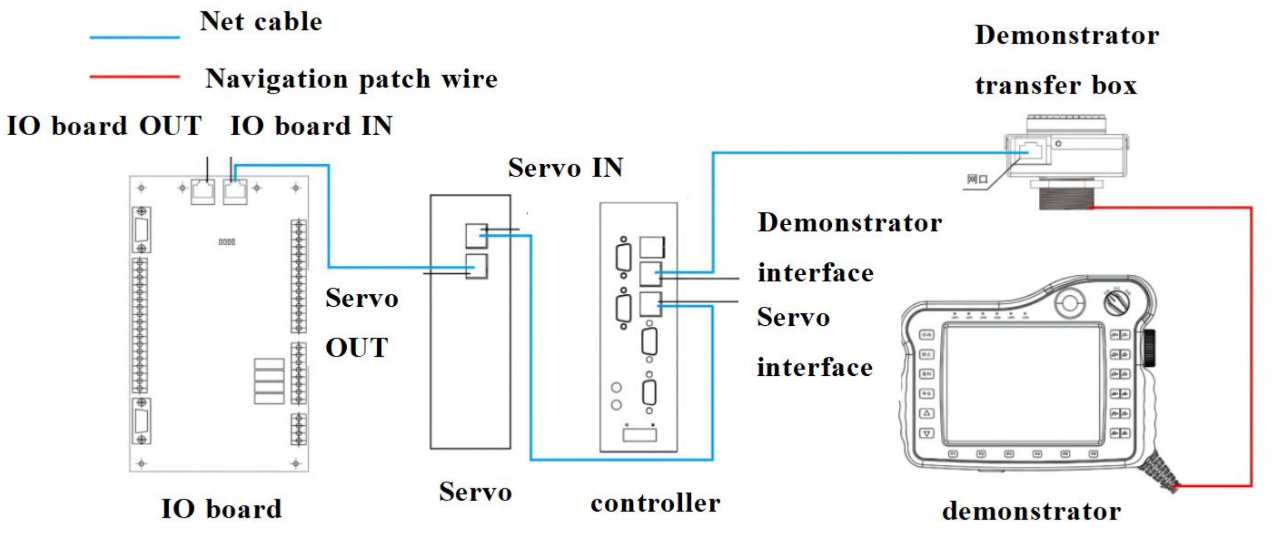


Figure 1-1 Controller connection diagram

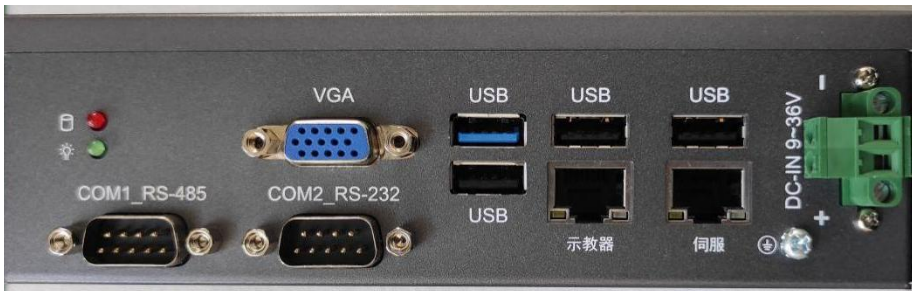


Figure 1-2 Appearance of the controller

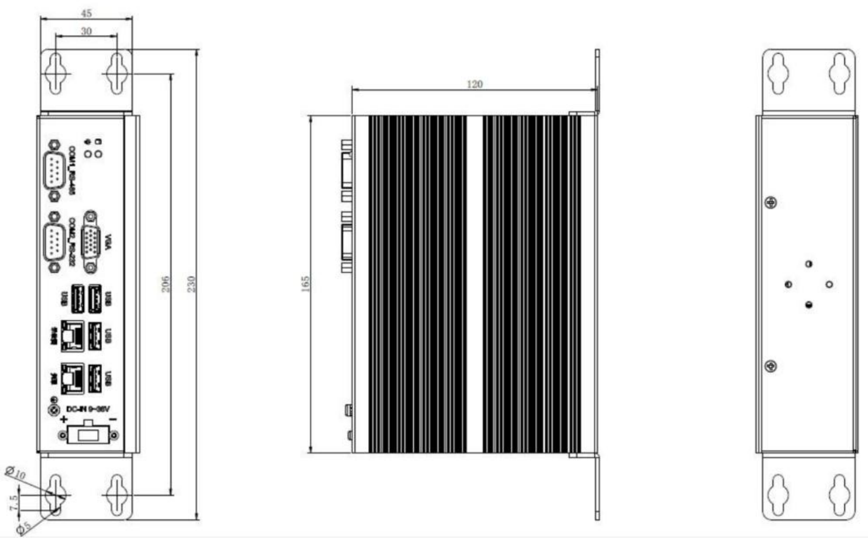


Figure 1-3 Dimensions of the controller

2. Description of external I/O interfaces

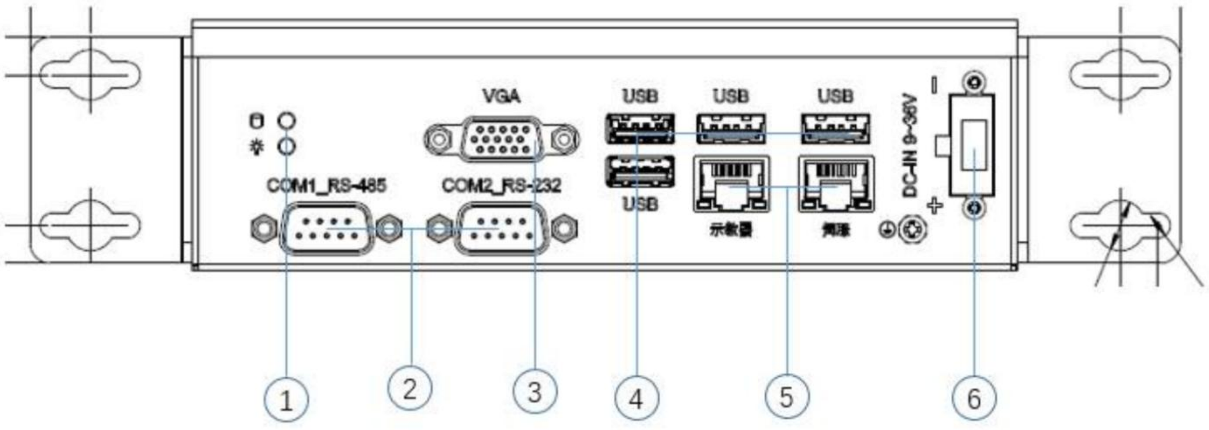


Figure 1-4 Front I/O panel

Serial number	name	Serial number	name
1	Power and HDD indicators	4	USB
2	COM	5	LAN
3	VGA	6	9~36V DC-IN

2.1 DC-IN power input

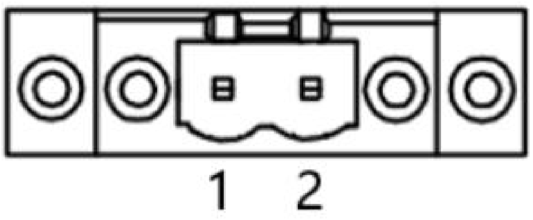


Figure 1-5 DC-IN ports

stitch	definition	stitch	definition
1	+9~36V	2	GND

2.2 HDMI/VGA display port

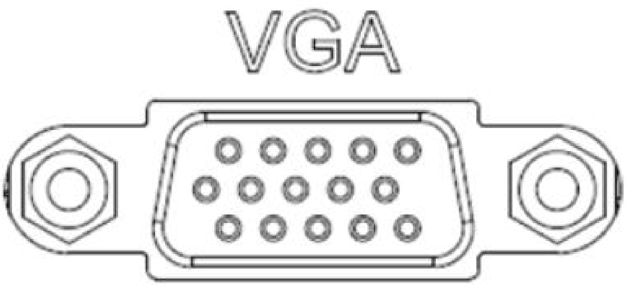


Figure 1-6 VGA ports

Display interface	Maximum resolution
VGA	2560×1600@60HZ

2.3 Serial port (COM1/COM2)

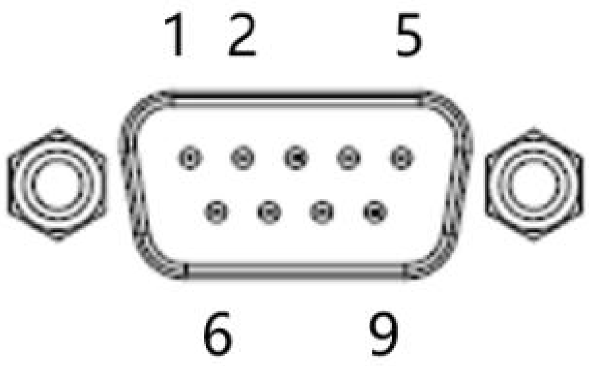


Figure 1-7 COM ports

stitch	RS232 definition	RS422 definition	RS485 definition
1	DCD	Tx-	Data-
2	RXD	Tx+	Data+
3	TXD	Rx+	NC
4	DTR	Rx-	NC
5	GND	GND	GND
6	DSR	-	-
7	RTS	-	-
8	CTS	-	-
9	RI	-	-

2.4 USB interface

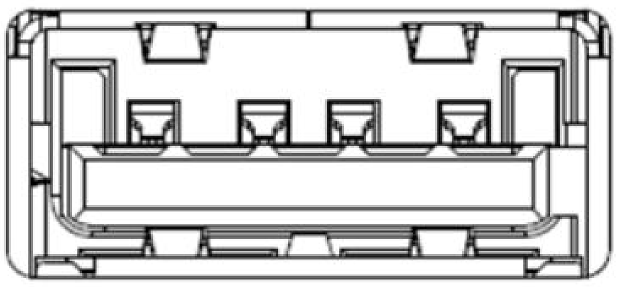


Figure 1-8 USB ports

USB Bus version	Theoretical maximum velocity	Rate designation	Maximum voltage/current
USB2.0	480Mbps(60MB/S)	High speed（High-Speed）	5V/500mA
USB3.0	5Gbps(500MB/S)	Superspeed （ Super-Speed USB）	5V/900mA

2.5 Net interface

Supports two 10/100/1000Mbps adaptive network ports. Network port indicators are defined as follows:

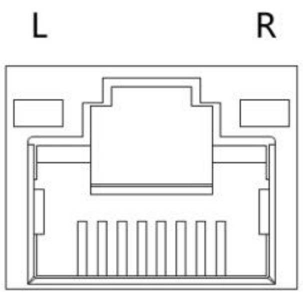


Figure 1-9 Network ports

Pilot lamp	Function definition	Indicated state
L	Network operating light	off:Not working green： working
R	Network port rate indicator	Off： 10Mbps green： 100Mbps orange： 1000Mbps

2.6 Status light

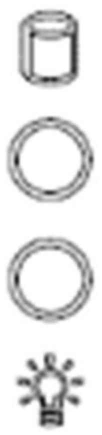


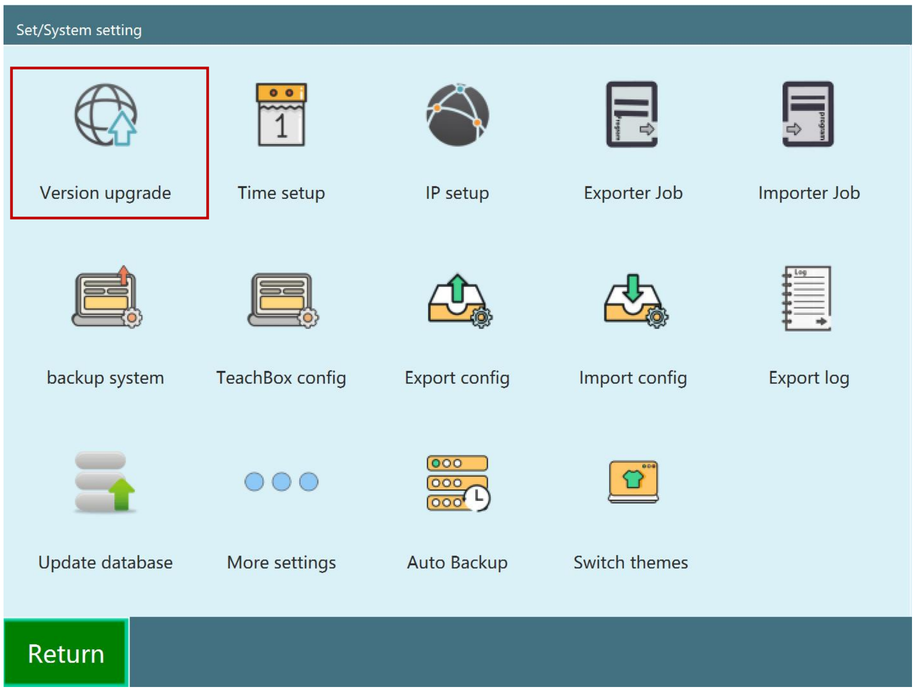
Figure 1-10 Status indicators

Indicator light marker	Function definition	Indicated state
🛢	SSD/Hard disk indicator	Green: SSD/ hard disk data read and write
💡	Power status indicator	Off： The system shuts down or hibernates Steady red: The system is powered on Red flashing: S3 sleep

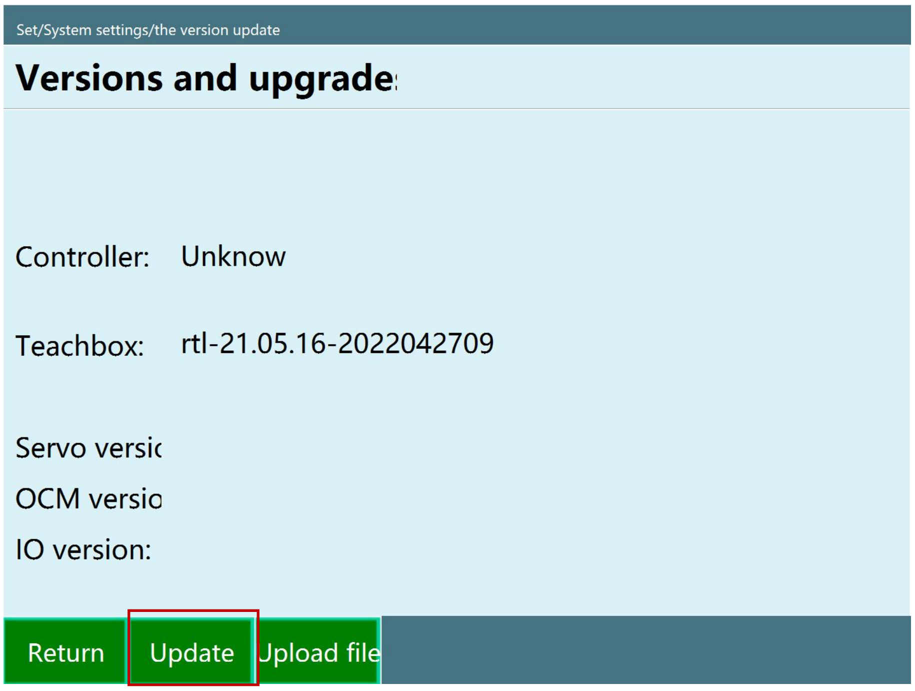
Appendix II Teaching apparatus operation

1. Version upgrade

(1) Switch permissions, log in to the administrator, and select the version upgrade in system Settings



(2) Insert the USB flash drive and click to detect the upgrade



Controller: Unknow

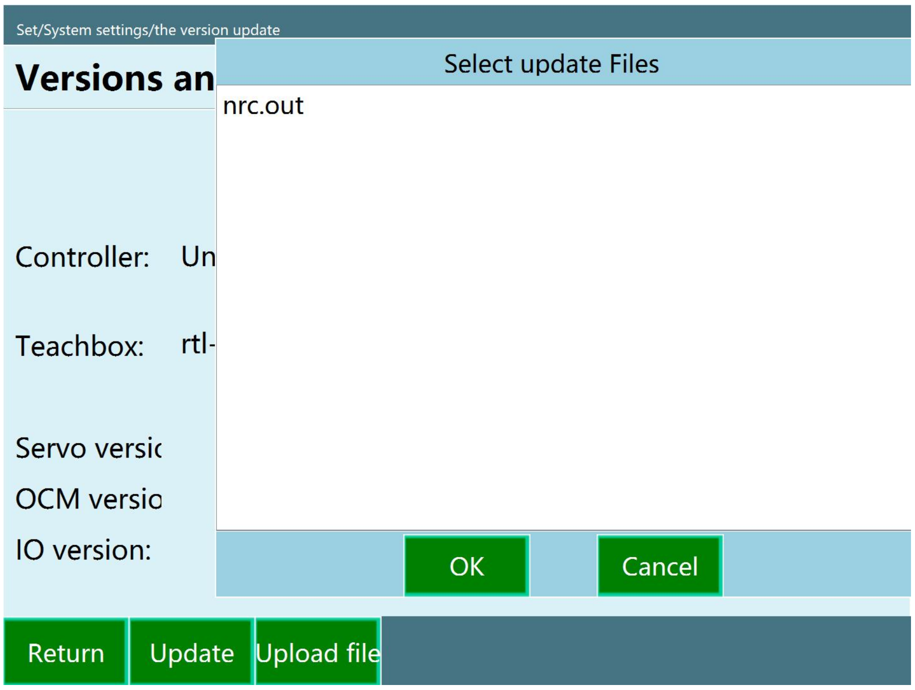
Teachbox: rtl-21.05.16-2022042709

Servo versic

OCM versio

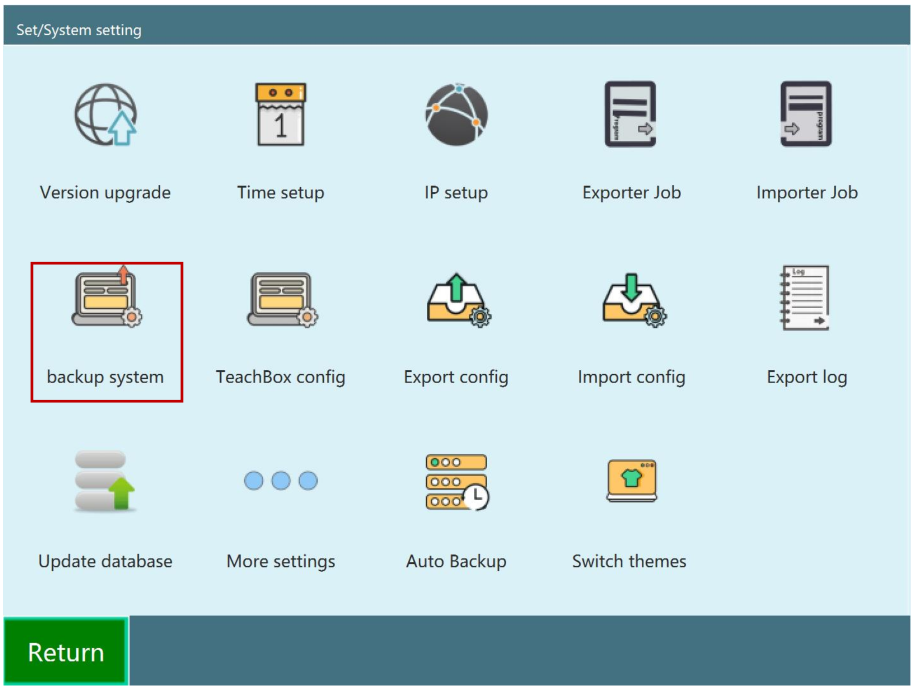
IO version:

(3) Click OK/Cancel to automatically upgrade

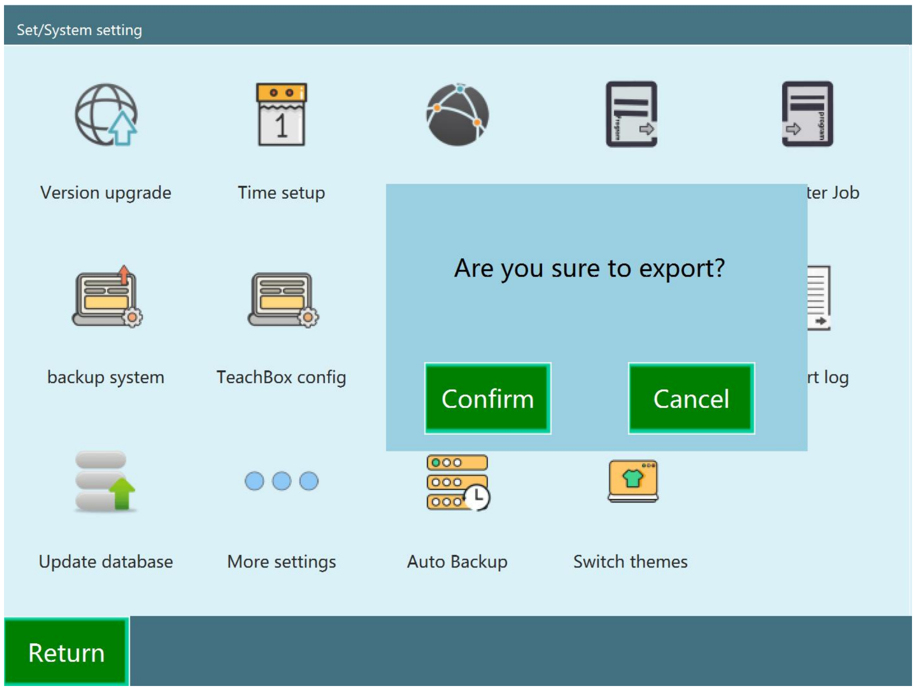


2. Backup export

(1) Switch permissions, log in to the administrator, and select the one-click backup system in System Settings

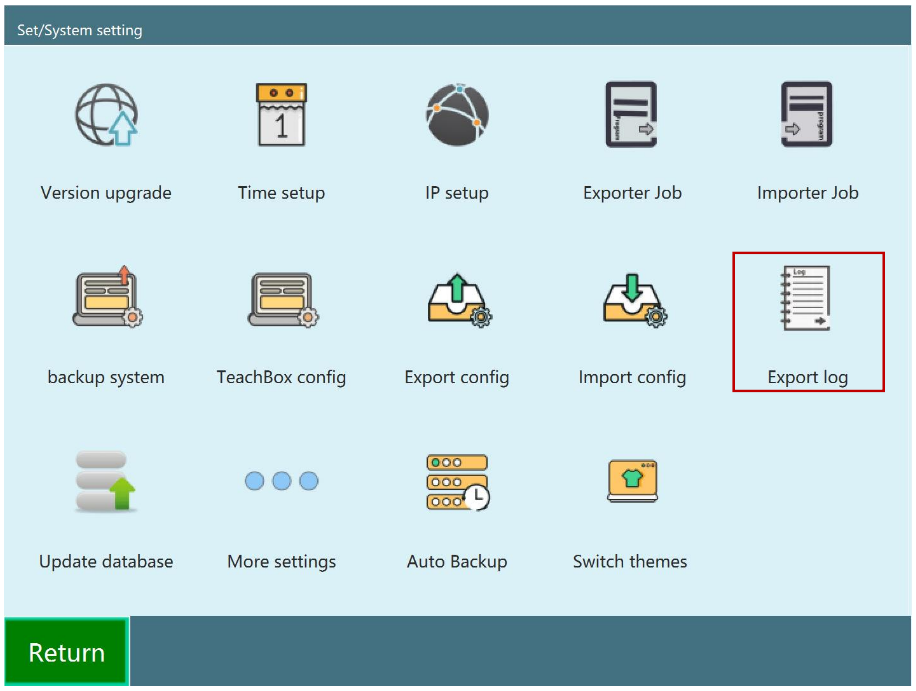


(2) Click OK to automatically back up to the U disk

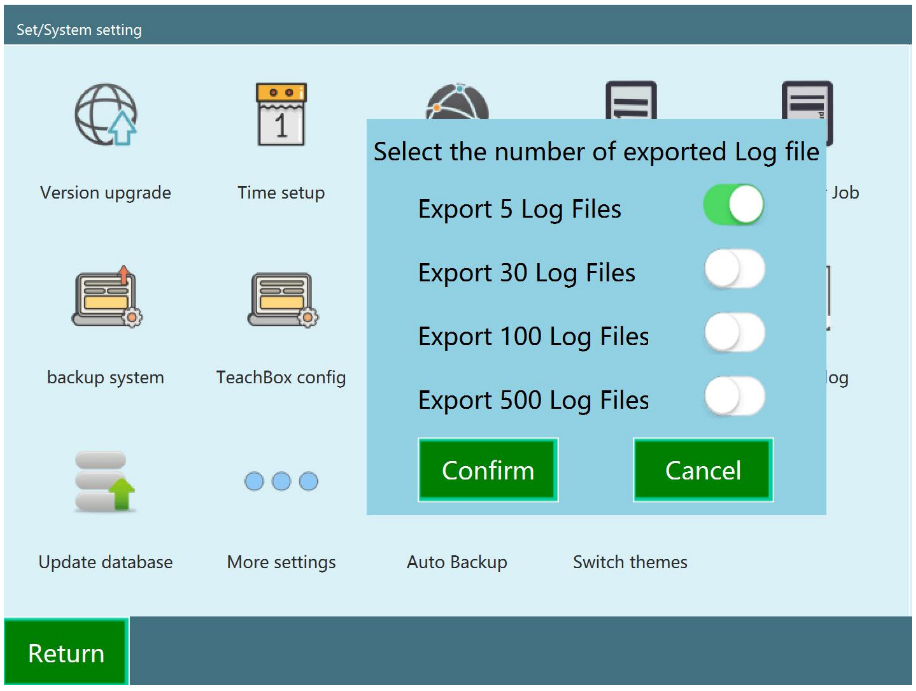


3. Export log

(1) Switch permissions, log in to the administrator, and select Export logs in System Settings



(2) Select the required options and guide to the USB flash drive


Set/System setting

Select the number of exported Log file

Export 5 Log Files	✓
Export 30 Log Files	
Export 100 Log Files	
Export 500 Log Files	