JAKA 节卡

JAKA Robot

App User Manual

JAKA Zu® APP

JAKA 节卡

JAKA Robot

App User Manual

JAKA Zu® APP

App Version: 1.7

Note:

The definition of collaborative robots follows international ISO standards and national standards to protect the safety of operators. We do not recommend directly applying the robot arm to circumstances where the object is human body. However, when the object is human body, users or application developers should configure a safe, reliable, fully tested and certified safety protection system for the robot arm to protect personnel safety on the premise that they can fully evaluate personnel safety.

The content contained in this user manual is the exclusive property of Shanghai JAKA Robotics Ltd (Hereinafter collectively referred to as JAKA), and shall not be used in any form without the written consent of JAKA.

The user manual is subject to revision and improvement on a regular basis by JAKA and its contents are subject to change without notice. Please check the actual product information carefully before using this manual.

The information contained in the user manual is not a commitment of JAKA, and JAKA is not responsible for any errors that may occur in this manual and any accidental or consequential damages caused by this manual and the products it introduces. Please read this manual carefully before installing and using the product.

The pictures in this manual are for reference only, please refer to the actual product.

If the robot arm is modified or disassembled, JAKA will not be responsible for after-sales service.

JAKA also reminds the user that safety equipment must be used and safety provisions must be observed when using and maintaining JAKA robots.

Programmers of JAKA robot and designers and debuggers of the robot system shall be familiar with the programming mode and system application installation of JAKA robots.

Manual Instructions

This manual mainly contains software overview, operating environment, and usage of JAKA Zu App.

This manual is intended for users who have received basic training in robot, which will facilitate installation and usage of JAKA Zu software.

Read more

For more information about our products, please scan the QR code on the right to visit our official website www.jaka.com.

Contents

Chapter 1 Preface

1.1 Preface

JAKA Zu is an App integrated with functions of robot demonstrators. The software is provided with the functions of manual manipulation, program writing, parameter configuration and information monitoring of the robot, and is the control software (demonstrator) of JAKA Zu collaborative robot. JAKA Zu replaces the cumbersome handheld robot demonstrator, and integrates demonstrator functions into a lightweight App that can be installed in mobile device or PC. It is possible to manipulate robot with mobile device or PC (with JAKA Zu installed inside) connected to the robot on the same LAN, without traditional cumbersome demonstrators and long control cables.

1.2 Software Function

1) Users can read general robot information and event logs;
2) Users can manipulate the robot manually;
3) Users can manage and set robot I/O modules;
4) Users can manipulate the robot by writing programs;
5) Users can configure relevant robot parameters;

1.3 Operation Environment

The recommended hardware configuration for JAKA Zu software is shown in Table 1-1:

Table 1-1 Terminal Configuration Table

Terminal Type	Tablet Computer	Terminal Type	Computer
Operating System	Android 8.0 and above	Operating System	Windows 7 32/64bit and above
Processor	Kirin 695 or Snapdragon 660 and above	Processor	Intel Core I3 and above
Storage Capacity	32GB	Storage Capacity	32GB
System Memory	4GB	System Memory	4GB
Screen Size	8.0 inches and above	Video Card	Intel HD Graphics 4000 and above
Network Communication	Wi-Fi	Network Communication	Wi-Fi or wired NIC

Chapter 2 Software Introduction

2.1 Interface Overview

2.1.1 Interface Layout

The main interface of JAKA Zu software mainly consists of function bar, switch bar and menu bar. See Figure 2-1.

Menu Bar: Including electric control cabinet, robot and software management. The main function includes electric control cabinet shutdown, App settings, robot arm settings, electric control cabinet settings, robot connection and robot display, etc. (**Note:** Android version of App is provided without window size control function).

Switch Bar: Including robot arm power switch and enable switch. The main function is to power on/off robot arm and enable/disable robot.

Function Bar: Including programming control, motion control, I/O control and monitoring information. The main function includes robot program management, robot motion control, electrical control cabinet I/O management and robot state monitoring.

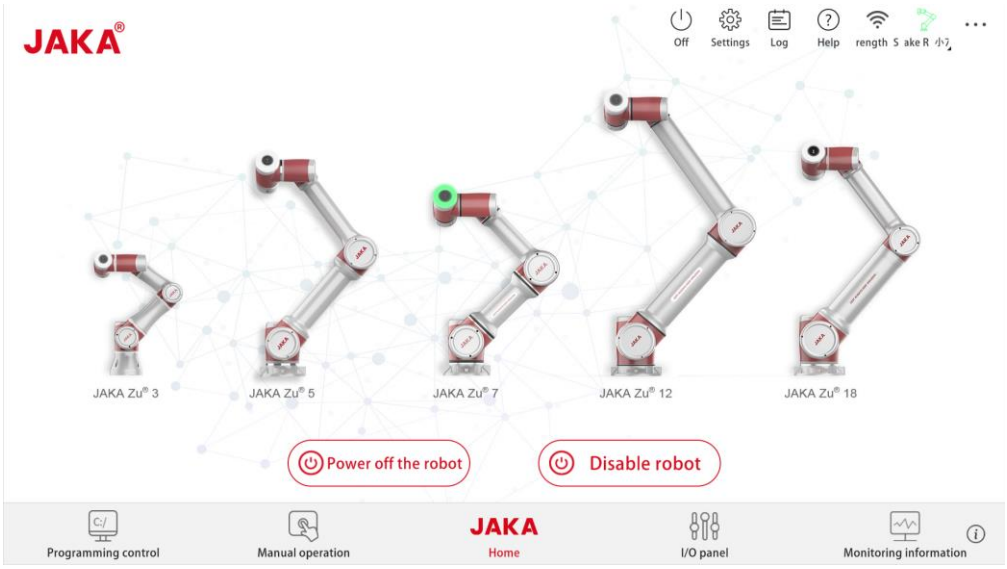


Figure 2-1 Main Interface of JAKA Zu Software

2.1.2 Interface Help

There is a "Help" button 

in the upper right corner of the App main interface. When you first use JAKA Zu software, you can click the "Help" button in the upper right corner to get information about the main interface of JAKA Zu software. The "Help" interface is shown in Figure 2-2 below.

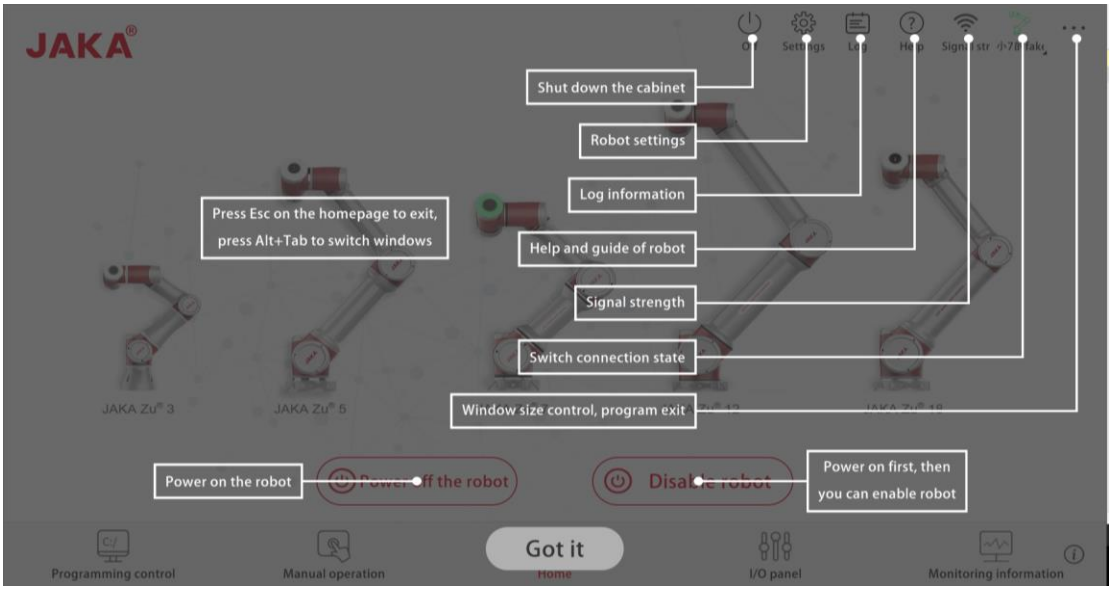


Figure 2-2 "Help" Interface of JAKA Zu Software

2.2 Interface Features

JAKA Zu software is demonstration and control software for JAKA Zu Cobot, with functions of robot parameters definition, robot log information reading, robot connection and control, robot arm state control, robot programming and control, robot manual control, robot I/O signal control and robot information monitoring, etc.

2.2.1 Electrical Cabinet Shutdown

You can shut down the electrical control cabinet by clicking the "Shutdown" button Off in the upper right corner (See Figure 2-3) or shutdown electrical control cabinet with external handle by clicking the "Shutdown" button.

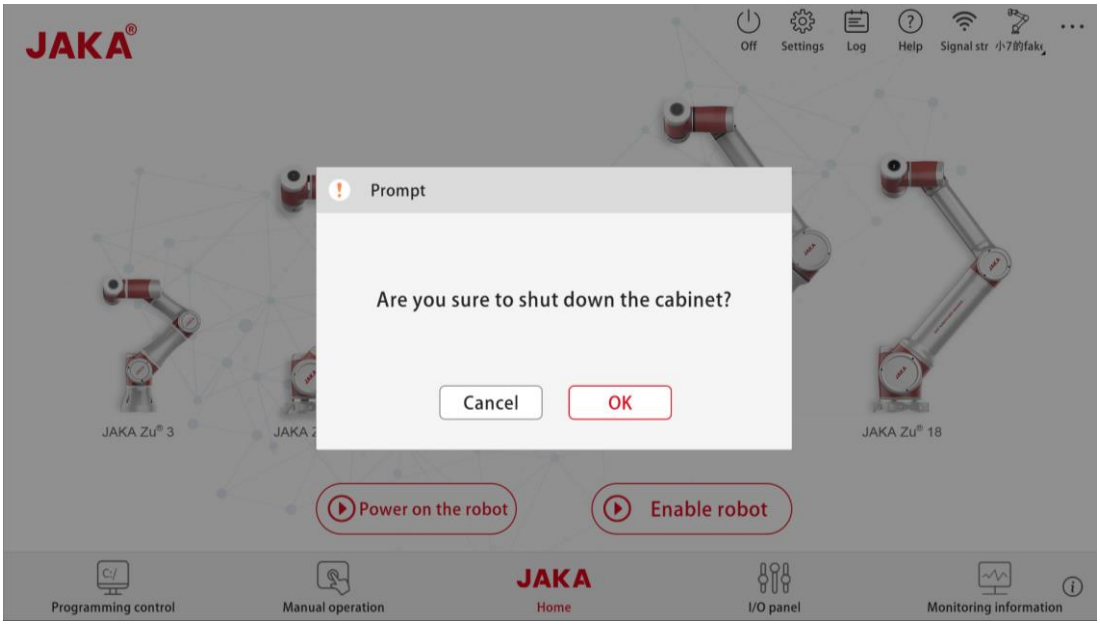


Figure 2-3 Shutdown Interface of Electrical Control Cabinet

2.2.2 Robot Log

Log records information of robot operation events, which is divided into messages, warnings and errors. Log information is important for storing historical data and diagnosing problems.

When there is an error alarm during robot operation, you can refer to the log information to check the cause, and to examine by yourself. For problem cannot be solved, you can contact JAKA technicians and inform them of the log information to solve the problem quickly. Click the "Log" button Log in the upper right corner to view the log. Click "Refresh" button to refresh log content and view the latest log information. See Figure 2-4.

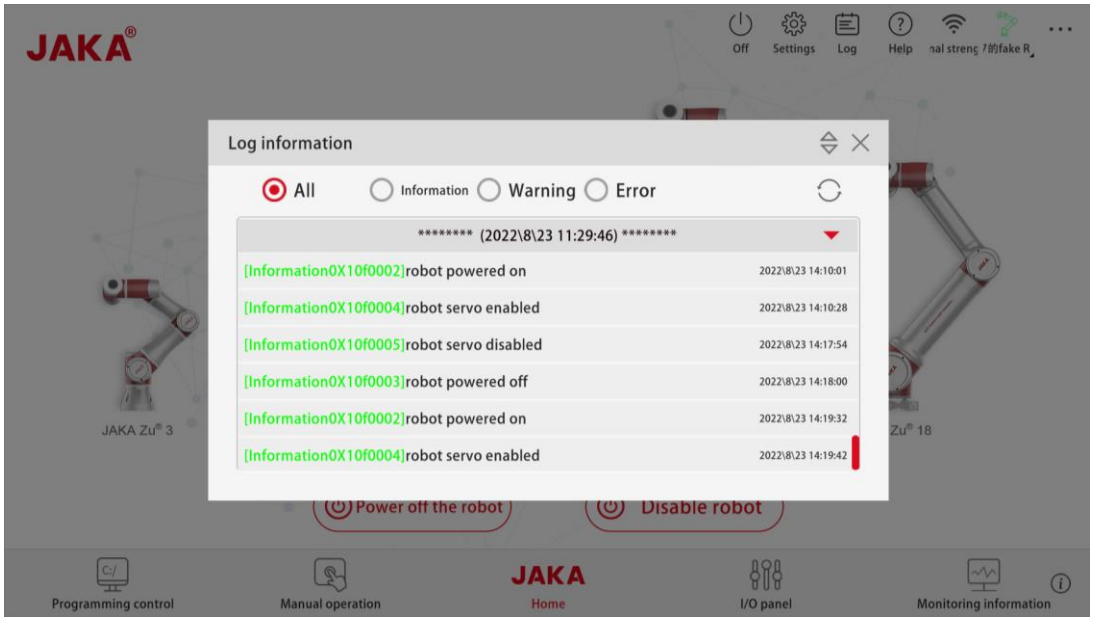


Figure 2-4 Log Information Interface

2.2.2.1 Information

The information will be stored when robot state changes, and you can check robot state change through state information, which is used to check whether the robot state switch is successful or whether robot state switch is abnormal in a certain period of time. The state information window is shown in Figure 2-5 below:

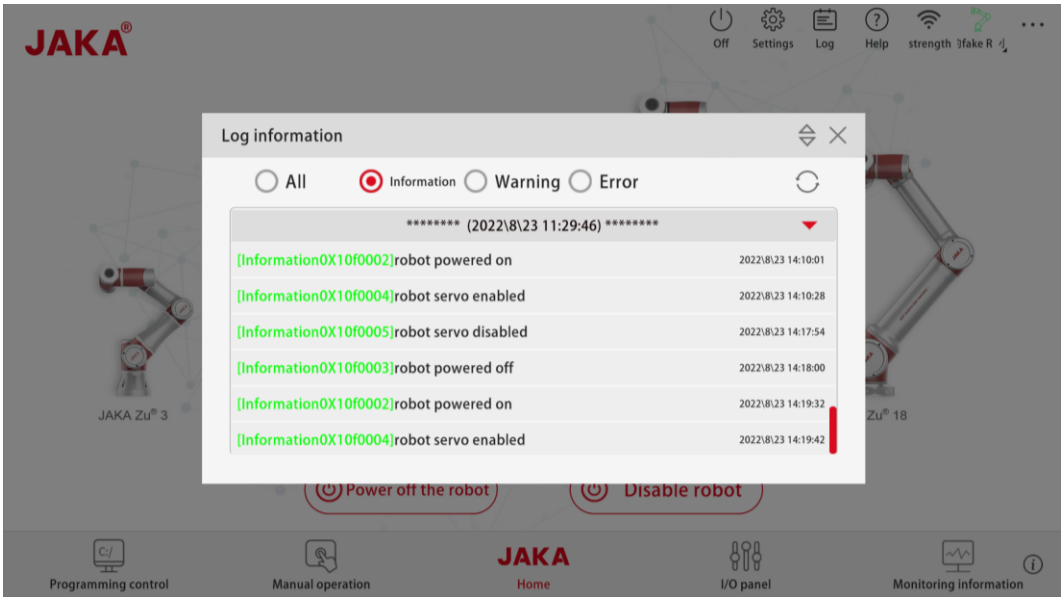


Figure 2-5 Information Interface

2.2.2.2 Warning

Warnings are pop-up messages that appear when abnormal JAKA Zu software operation or abnormal robot state occurs, which will be stored in warning message log in real time for troubleshooting and abnormality monitoring. See Figure 2-6.

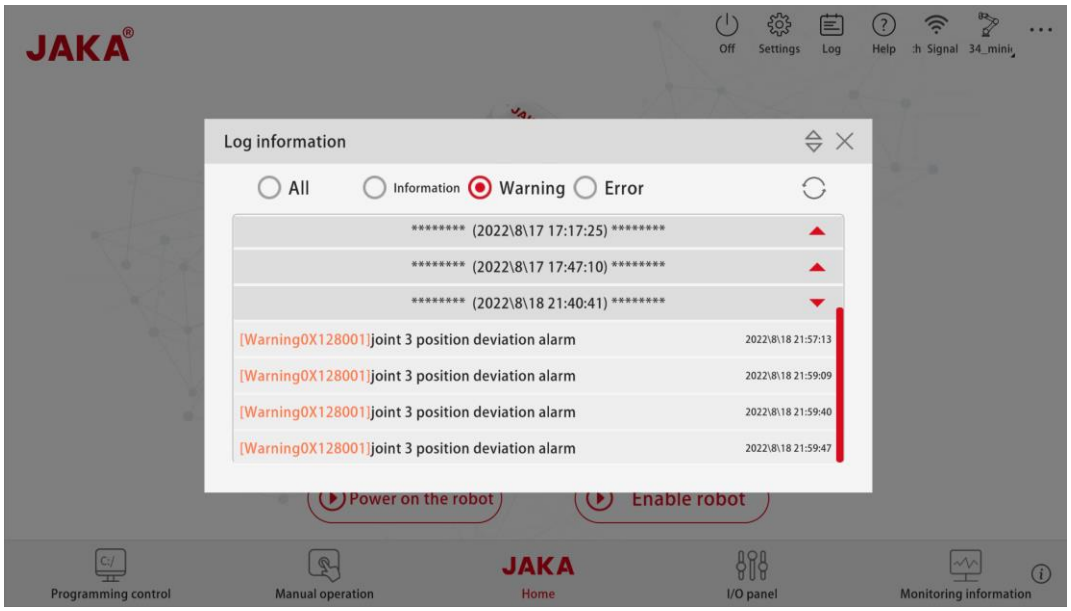


Figure 2-6 Warning Message Interface

2.2.2.3 Error

Errors are messages that pop up when wrong JAKA Zu software operation or robot error occurs. Error messages in the JAKA Zu software or on the robot arm will be stored in log information for troubleshooting and analysis. See Figure 2-7.

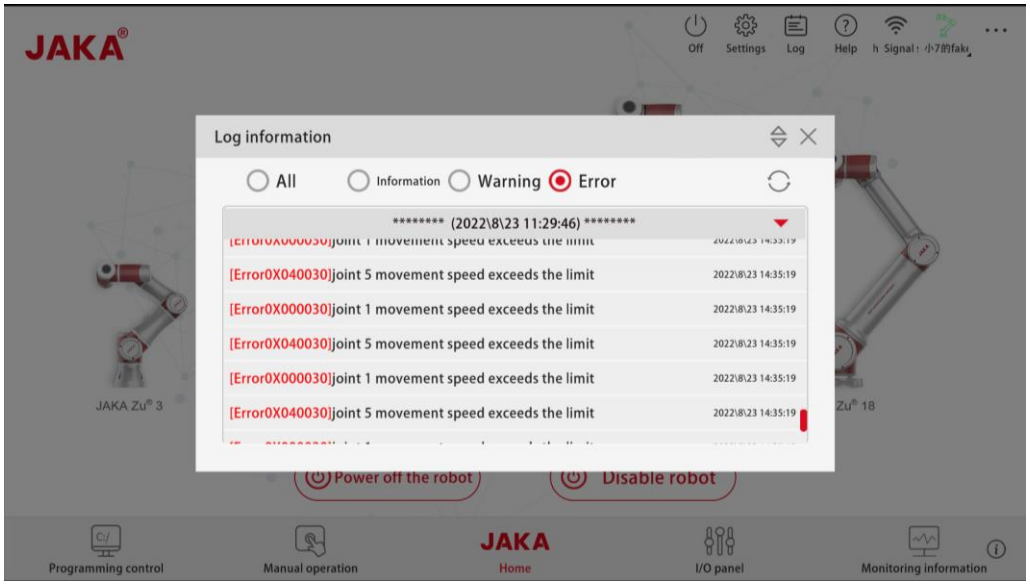


Figure 2-7 Error Message Interface

2.2.3 Signal Strength

There is a "WLAN" icon in the upper right corner of JAKA Zu software main screen, which allows you to see the network state when connecting to the robot by Wi-Fi LAN.

2.2.4 Robot Connection

JAKA Zu software can manage multiple robots with one App. Connect JAKA Zu software to the same LAN as the robots, click the "Not Connected" button in the function bar and click the robot to be connected, then a login interface will pop up. Enter your password, and click "Connect to the robot". The connection interface is shown in Figure 2-8.

The login interface allows you to select three different operation authorities: operator, technician and administrator.

When the robot program is not running, only administrators and technicians can edit and change system, operation settings, safety settings, program settings, hardware and communications.

Operator: With permission to "start/stop program", "turn on/off the robot", and "view log information and robot status"; default password: 0

Technician: With permission to "edit program" (related settings required for program editing, etc.), including all operator permissions; default password: 0000

Administrator: With permission to operate all software functions; default password: jakazuadmin

Note:

1. Programming control interface: The operator can open and browse the program and variable monitoring in the program, and is allowed to perform operations necessary for program executing and

observing, including run, pause, stop, program rate adjustment; but is not allowed to perform operations of creating, editing, saving, saving as, advanced operations, single-step debug, and program lock.

2. I/O panel interface: The operator can only observe and browse information such as I/O names and state.

3. Setup Interface: The administrator can edit, modify and view all settings in "Settings"; the operator can view all pages except user management; the technician can view all pages except user management, and is allowed to modify settings except safety settings (the technician can configure the robot pose setting in security settings).

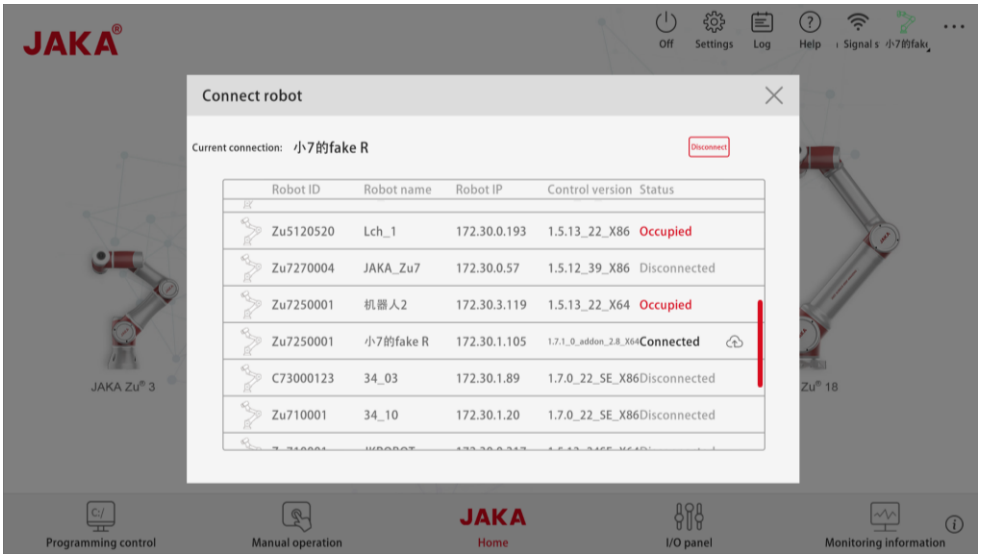


Figure 2-8 Robot Connection Interface

2.2.4.1 How to log into the robot

First, connect JAKA Zu software to the LAN in which the robot is located to enter the software main interface. Second, click the "Not Connected" button 

in the upper right corner to visit the robot connection interface, where you can view all the robot information (robot ID, name, IP address, controller version and state information) in the current LAN. After that, select the robot to be connected based on the robot information, and click to enter login interface. Finally, select the login authority, enter the password, and click "Connect the robot". The corresponding operation is shown in Figure 2-9 below:

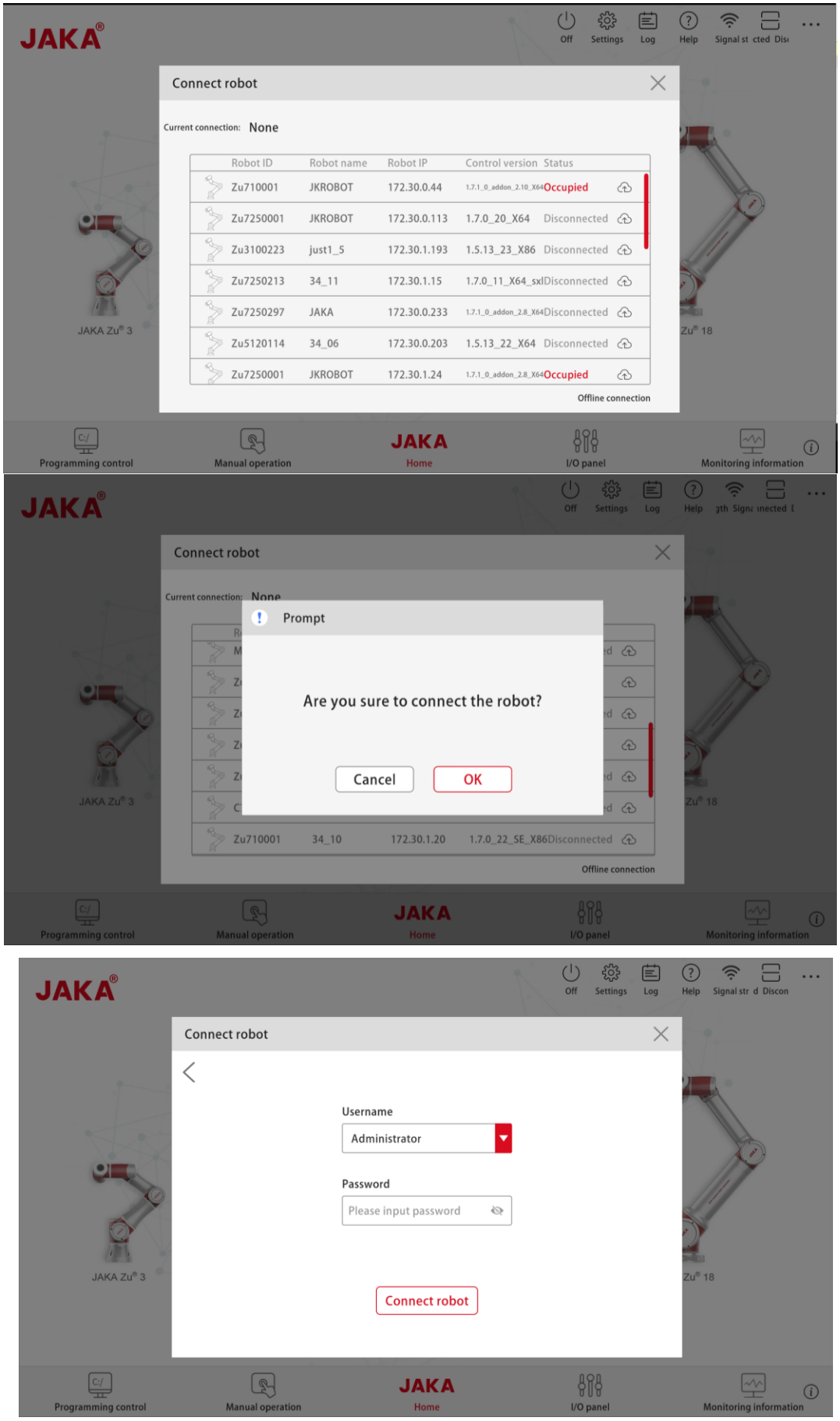


Figure 2-9 Robot Connection Steps

2.2.4.2 Offline Connection

JAKA Zu software supports offline connection function, which is mainly used to connect virtual robots for virtual operation experience and learning. Click the "Not Connected" button in the upper right corner of the software main interface to enter robot connection interface, where you can see the "Offline Connection" button in the lower right corner. Click to enter offline connection interface, select user name, enter the password, and fill in the robot address (**Note:** robot address is the trial code applied in WeChat Official Account). Now the virtual robot is connected. The virtual connection interface is shown in Figure 2-10 below.

Note: The virtual robot is applied in WeChat Official Account of "JAKA Robot".

For virtual operation experience, it is required to ensure that App version is matched with virtual robot.

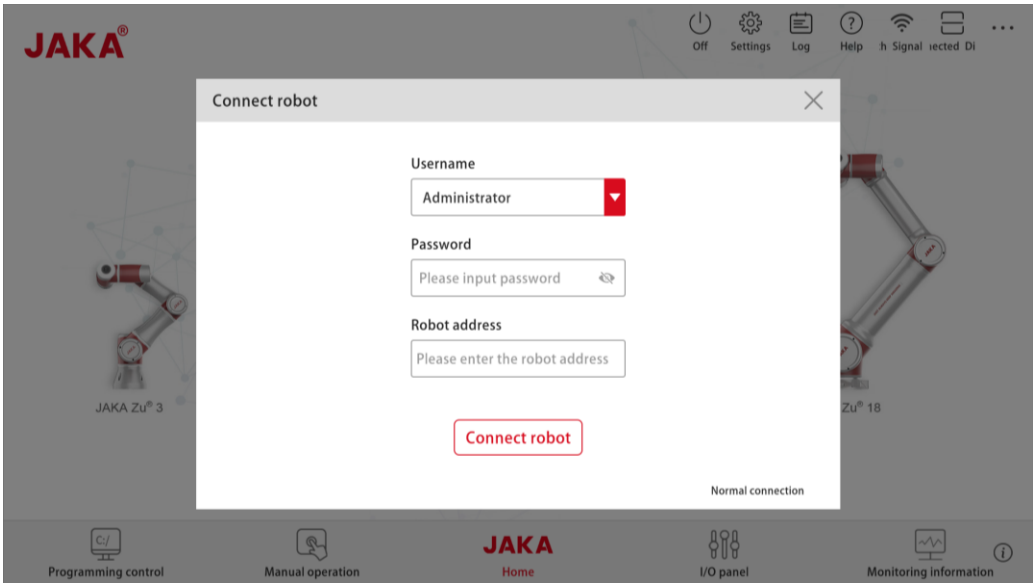


Figure 2-10 Robot Connection Steps

2.2.4.3 Version Upgrade

JAKA Zu software supports cross-version update operation. Version update operation can be performed with robot unconnected. Click the "Not Connected" button in the upper right corner of the software main interface to enter robot connection interface, where you can see the "Update" button on the right of the robot information. Click to enter the login interface, enter the administrator password, select the corresponding update file, and click the "Update" button to update the robot. The operation interface is shown in Figure 2-11 below.

Note: When the robot is connected, click the "Update" button on the right to enter update interface in the settings for updating.

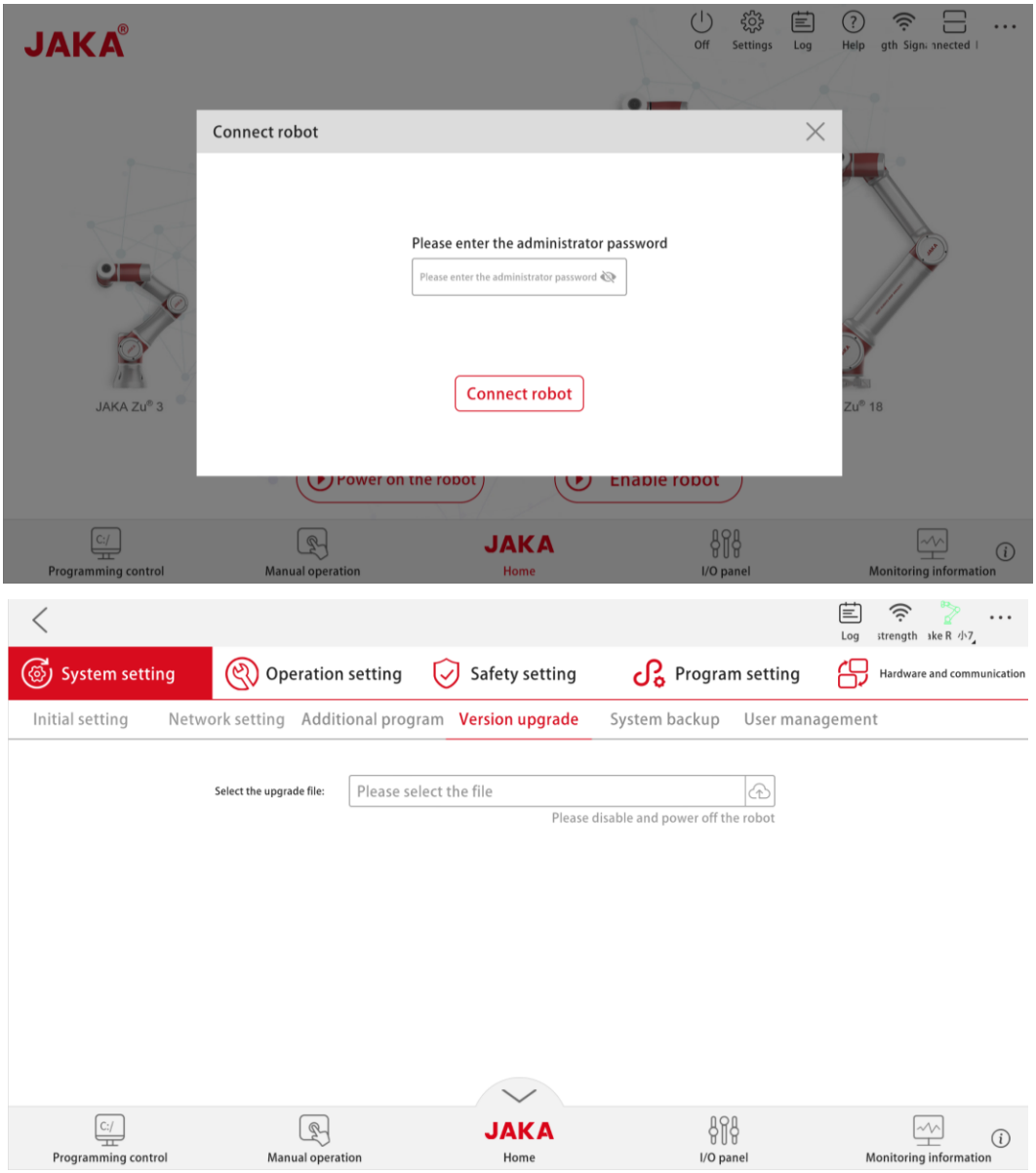


Figure 2-11 Robot Version Update Steps

2.2.4.4 How to view connection information

To view the information of connected robots, click the "Robot" icon  in the upper right corner, and then click the "Zoom in" button  at the bottom of the pop-up interface to enter robot connection interface, where you can view information about the connected robots and the robots in the LAN. See Figure 2-12 below.

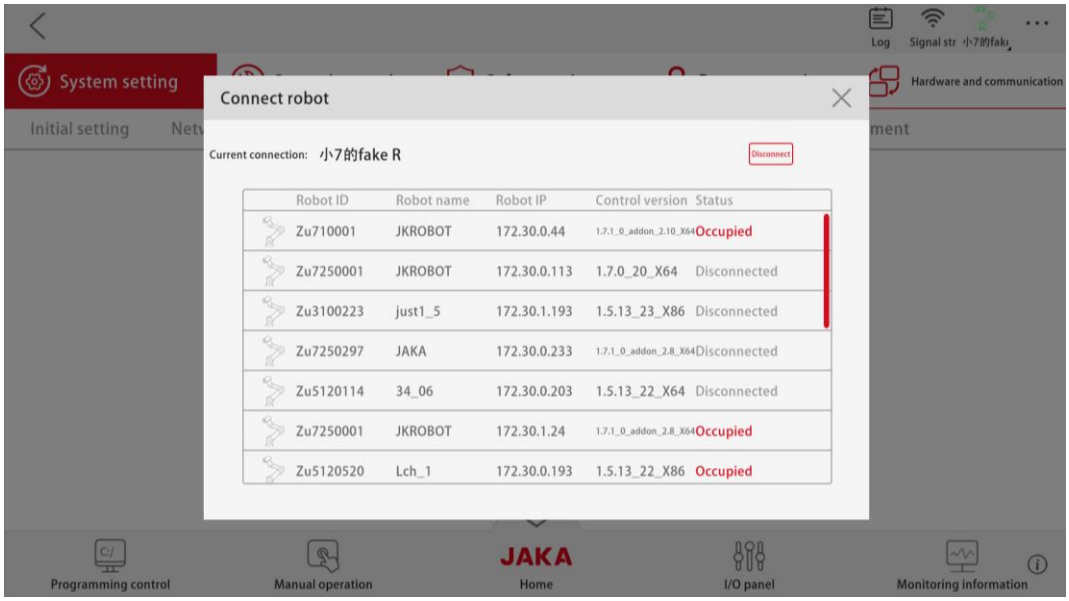


Figure 2-12 Robot Connection Interface

2.2.5 Robot Start-up/Shutdown

Before starting the robot, it is required to start the electrical control cabinet first. Press and hold the handle switch button for 1~2s to start the electrical control cabinet. The robot arm cannot be connected and powered up/enabled until the electrical control cabinet starts successfully (the JAKA breathing light on handle turns blue).

First, click "Power on the robot". When the robot end indicator in the software main screen turns blue or the robot end indicator turns blue, it means that the robot is powered on successfully, as shown in Figure 2-13-1. Then click "Enable robot". When the robot end indicator in the homepage turns green or the robot end indicator turns green, it means that the robot arm is successfully enabled. Now the robot is successfully started. See Figure 2-13-2.

Figure 2-13-1 Robot Switch Control Interface

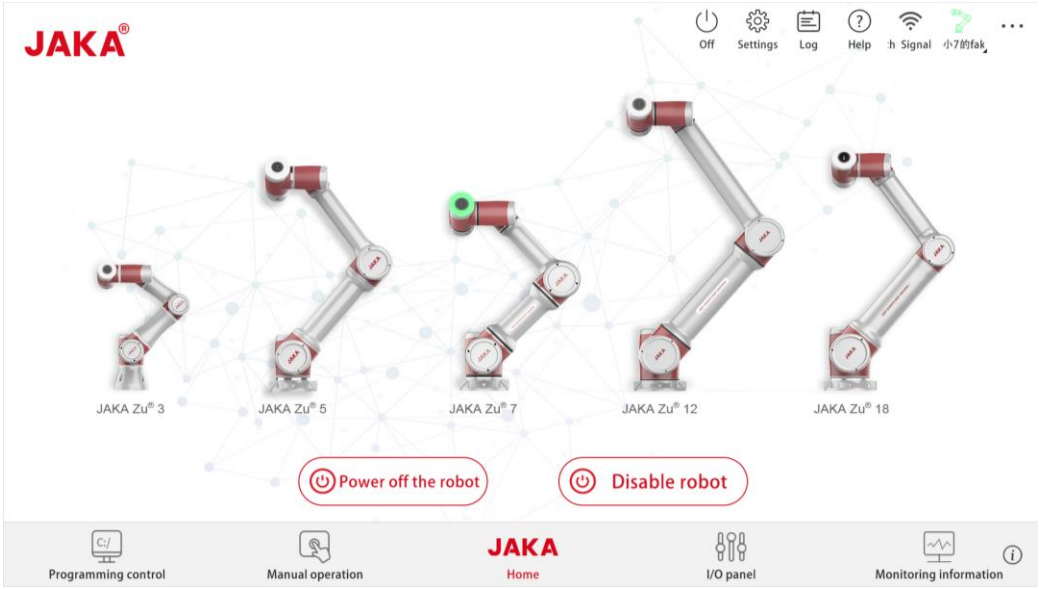


Figure 2-13-2 Robot Switch Control Interface

Robot arm shutdown steps are as follows: First, click "Disable robot". When the robot end indicator in the software main interface turns blue or the robot end indicator turns blue, it means that the robot arm is successfully unenabled. Then click "Power off the robot". When the robot end indicator in the software main interface becomes colorless or the robot end indicator turns off, it means that the robot arm is successfully powered off. Now the robot is successfully shut down.

How to shutdown electrical control cabinet: Press and hold the handle switch button for 3~5s to power off the electrical control cabinet, or click the "Shutdown" button 

on the upper menu bar of JAKA Zu software for shutdown. When the JAKA breathing light on the handle turns off, it means that the electrical control cabinet is shut down successfully.

Note:

1. Before powering on the robot, it is required to power on the electrical control cabinet first.
2. Before enabling the robot arm, it is required to power on the robot first.
3. Before robot powering off, it is required to disable the robot first.
4. It is strictly forbidden to power off the electrical control cabinet before powering off the robot, as unknown failure may occur when powered off during robot operation.

2.2.6 Robot State Monitoring

JAKA Zu software can monitor robot information. You can access the information monitoring interface from the software homepage to view information about the robot, controller and joints. Cabinet temperature, instantaneous robot power and robot current values can be monitored in electrical cabinet information.

Current, voltage, temperature, torque, limit state and speed limit state information for joints 1-6 can be monitored in joint information, as shown in Figure 2-14.

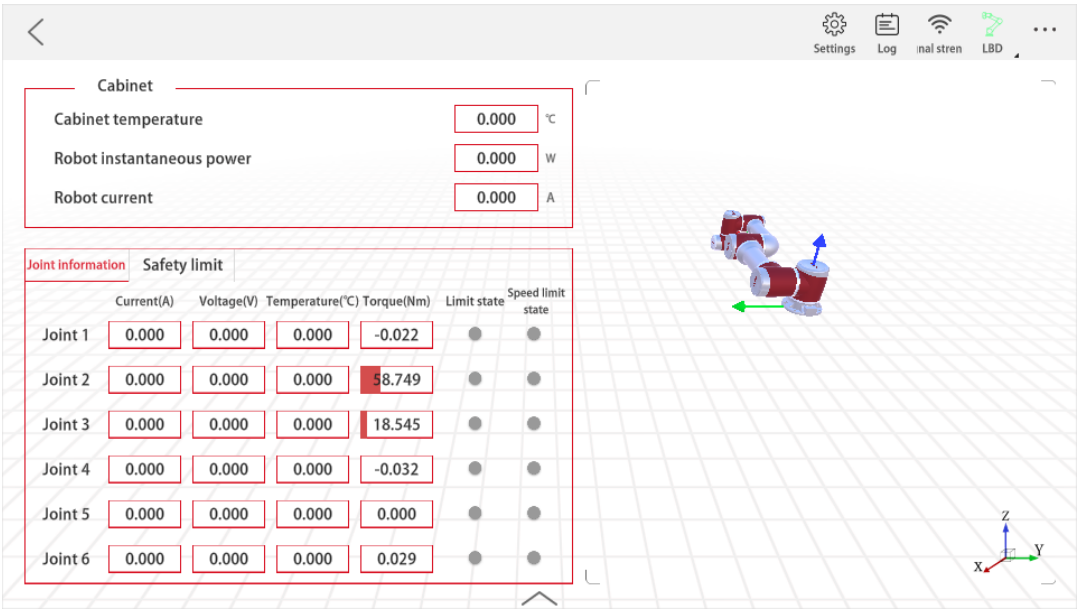


Figure 2-14 Information Monitoring Interface

The speed limit and limit state of the robot can be seen in the safety restriction, and the corresponding indicator lights up when the robot is in a certain limit mode. See Figure 2-15.

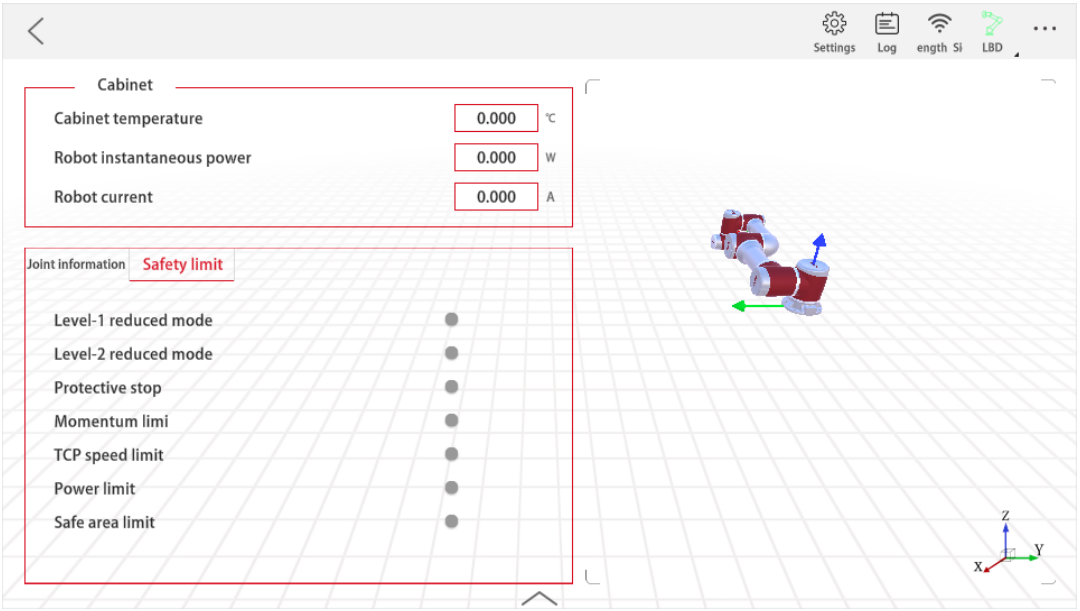


Figure 2-15 Safety Restriction Interface

2.2.7 Version Information

When JAKA Zu software is connected to the robot, you can view the information about corresponding version of software, controller and robot, as well as switch languages, adjust software sound, visit official website, find customer support and check for updates. See Figure 2-16.

Note: The soft keyboard on/off function is limited to the PC version.

Figure 2-16 Software Version Information Interface

2.2.7.1 Version Language

JAKA Zu software supports Chinese, English and Japanese. Open the software to enter the main interface, and click the "About" button ⓘ at the bottom right corner to switch different language version by clicking language information in the language settings. When selected as automatic mode, the App will automatically read the current device system language and switch the software language.

2.2.7.2 Version Information

Open the JAKA Zu software and connect to the robot to enter the software main interface. Click the "About" button ⓘ at the bottom right corner to open the version information window, where you can view the App version, servo version, SCB version, controller version and robot number.

Note: The servo version can be checked only after the robot arm is powered on.

2.2.7.3 Customer Service Support

JAKA Zu software supports technical customer service function. When you need technical consultation: Open the software to enter the main interface; Click the "About" button ⓘ at the bottom right corner to enter "About" interface; Then click "Click to open WeChat QR code" button to visit "WeChat customer service QR code" pop-up window for technical consultation via WeChat scan function. See Figure 2-17 below.

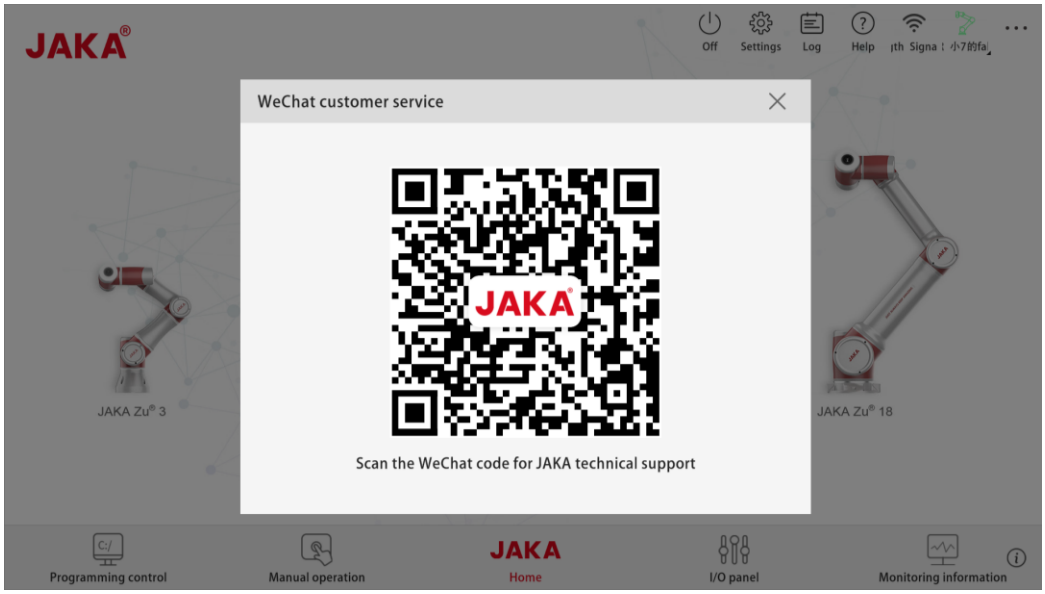


Figure 2-17 Customer Service Information Interface

2.2.7.4 Software Update

JAKA Zu software supports version search function, which allows users to find the corresponding matching version or the latest version according to version requirements. Open the software to enter the main interface; Click the "About" button ⓘ at the bottom right corner; Then click "Check for updates" to enter the software update interface. Select the acquisition type according to the search intention; Enter the controller version number, electrical control cabinet number and robot number; Click "Search" to check information of corresponding version; Then click "Version Information" button ⓘ on the right to view the current version update information. Click "Download" button on the right to download the corresponding file to the local memory device. See Figure 2-19 below.

Note:

1."Check for updates" function of the software requires that the network connected to the device in which the software is installed support the Internet access function.

2.When the acquisition type is selected as "specified supporting software", the controller version number is required to be entered manually; when the acquisition type is selected as "latest software", the controller version number is not required to be entered.

3.The electrical control cabinet number can be viewed in the lower right corner label of the electrical control cabinet door.

4.The robot number can provide number information of the robot base nameplate.

5.When "Check for update automatically" is checked, the robot will automatically check for updates once powered on, and a pop-up window will remind users if there is any update. See Figure 2-18.

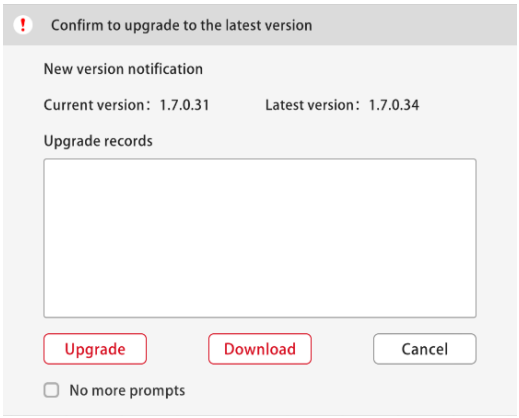


Figure 2-18　Software Update Interface

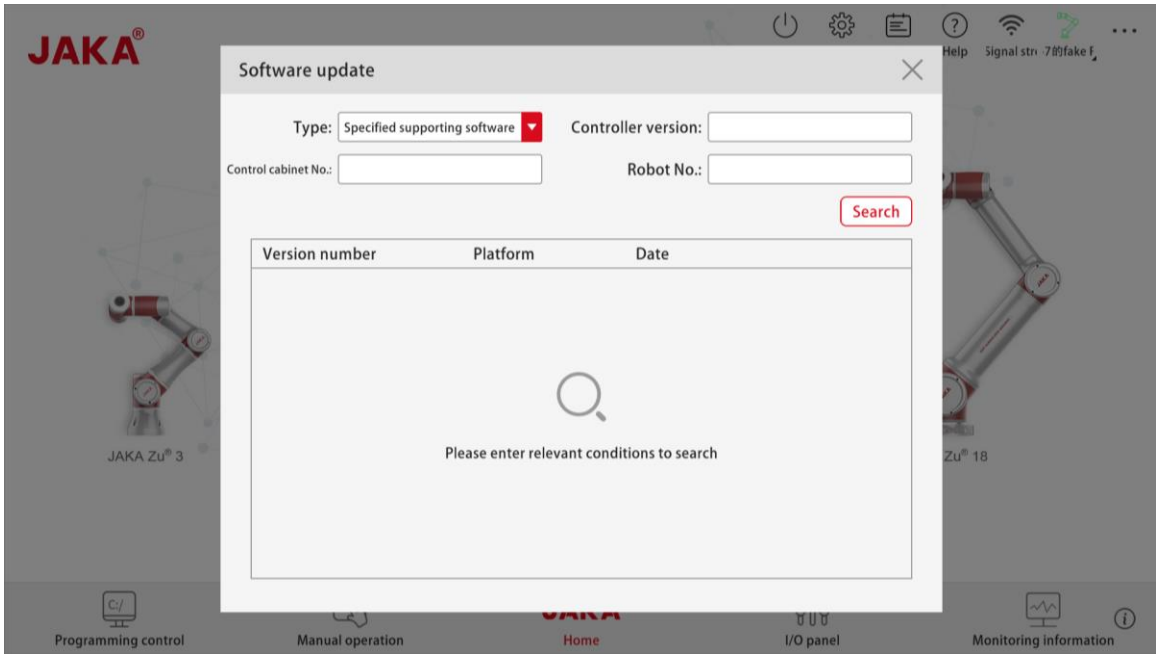


Figure 2-19　Software Update Interface

Chapter 3 Parameter Settings

Click the setting button at the top right of the App to enter the setting interface. There are 5 major sections in the setting interface: system settings, operation settings, safety settings, program settings, hardware and communication.

3.1 How to Configure Robot Parameters

3.1.1 System Settings

3.1.1.1 Time and Name Change

The initial settings are settings of robot name and system time. See Figure 3-1.

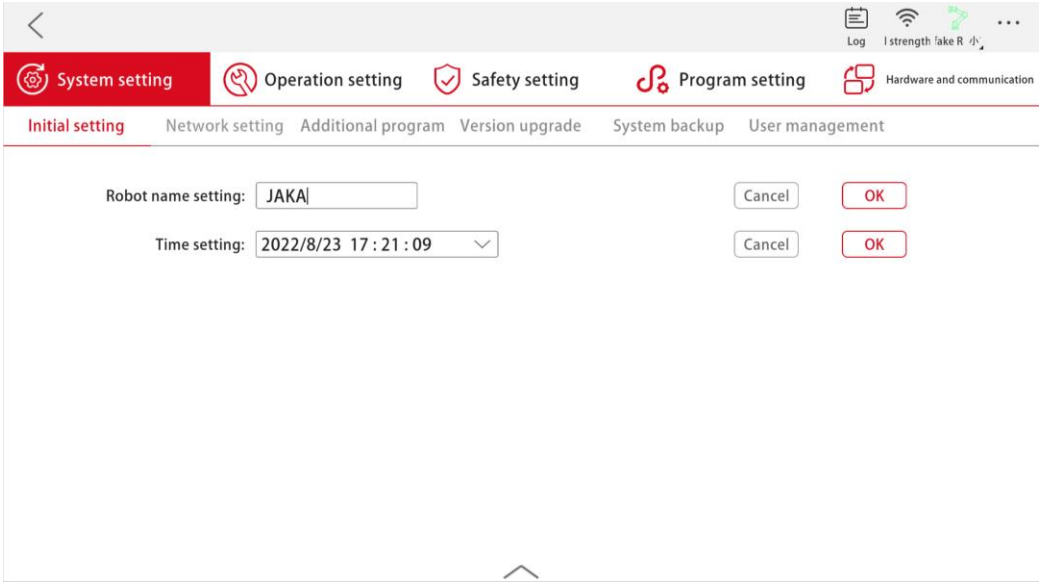


Figure 3-1 Initial Setting Interface

(1) Robot Name Settings

Role: After changing the robot name, you can quickly find the robot to be connected by robot name in robot connection interface. After connecting to a robot, you can also quickly know which robot is currently connected by the information in the upper right corner of the App.

(2) Time Settings

Role: Modify the system time so that the time in the controller matches the user's local time.

Note: After setting the time, the "OK" button will take effect again after a 3-second delay.

3.1.1.2 Robot IP Address Modification

In this interface, you can choose how the robot gets its IP address. See Figure 3-2.

An automatic IP address will be given by default in factory setting. If you need to fix the robot IP, it

is required to make sure that all subsequent devices that communicate with the robot over network (including Pad for debugging) are located under the same subnet as the robot.

Note:

1. To fix the robot IP address, it is required to configure it in powered off and disabled state.
2. When fixing IP, after clicking OK, the network configuration will be prompted to restart, and you can see App "Disconnect" pop-up window.
3. V2.1 electrical control cabinet: ①Default setting of control cabinet panel's network port is 10.5.5.x network segment; when this network port is used, it is required to configure IP address of the connected device within 10.5.5.101~10.5.5.254; or configure the device to obtain IP address dynamically. ②Default setting of the network port at the bottom of control cabinet is a dynamic IP address. Note that the two network ports cannot be set in the same IP network segment.
4. MiniCab electrical control cabinet: ①Default configuration of LAN1 network port is 10.5.5.x; When connecting to LAN1 port, the device IP address shall be configured within 10.5.5.101~10.5.5.254; or configure the device to obtain IP address dynamically. ②Default setting of LAN2 network port is a dynamic IP address. Note that the two network ports cannot be set in the same IP network segment.

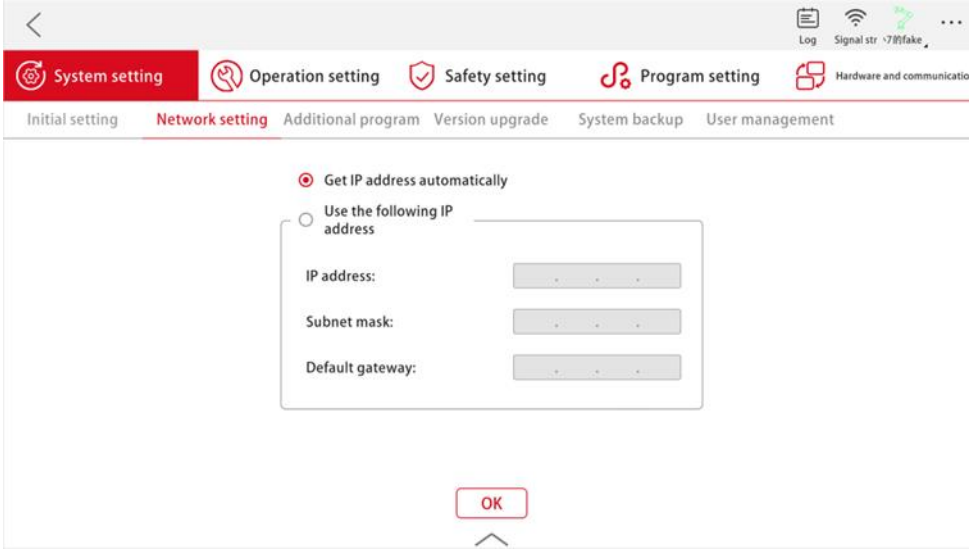


Figure 3-2 Network Setting Interface

3.1.1.3 Additional Programs

JAKA Zu software can be used for secondary development via AddOn functions, which can develop additional functions for specific situations. To use AddOn functions, please consult AddOn Usage and Development Manual or contact JAKA technical staff.

Click the "+" sign in the upper right corner to select additional package according to the file location, and click OK to upload additional package to the electrical control cabinet, or delete and export the uploaded additional program. See Figure 3-3.

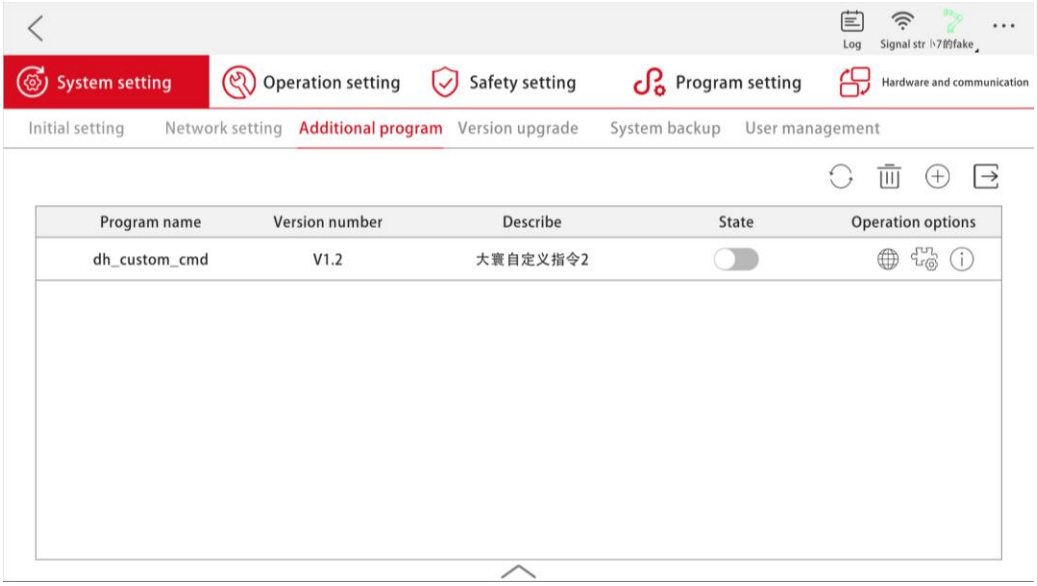


Figure 3-3 Additional Program Management Interface

3.1.1.4 Version Upgrade

Users can update the electrical control cabinet, PS/SCB, servo, CAN and OTA in the version update interface. See Figure 3-4 below.

Update steps are shown as follows:

1) Download the update package to local memory of your device；

2) Open App to connect the robot and make sure the robot is powering off；

3) Click "Settings" button on the upper right corner of the App homepage to enter "System Settings" → "Version Update" page; Click the white part of "Select File" to select the corresponding downloaded update package, and click "Upload" button (the name of the update package cannot be modified, you must keep the original file name)；

4) Select the content to be updated in pop-up window according to version update requirements (all selected by default, and it is recommended to update all contents except special situation)；

5) Click the "OK" button and wait for update to complete. The controller will restart automatically after the upgrade is completed；

6) After the restart, connect the robot via supporting JAKA Zu software, and check the version to see if the update is completed.

Note:

1. After downloading the update package, the update package name cannot be modified.

2. To update version 1.4, 1.5 or 1.6 to version 1.7, it is required to update the controller version to version 1.7 first, and then update overall contents. Please contact JAKA technical staff for operation.

3. To update version 1.7, it is required to ensure that the SCB version is above 02_50 and the PSCB version is above 02_00. If you find that the SCB/PSCB version does not match before updating, please contact JAKA technical staff.

4. Version 1.7 is compatible with update of older versions (below version 1.7). Connect the robot to enter Settings - System Settings - Version Update; Select "Upload older version upgrade package" and click "Upload", wait for restart to complete, then the update is completed. (**Note:** for version 1.7 update package, the suffix is .jaka; for older version of update package, the suffix is .tar.gz)

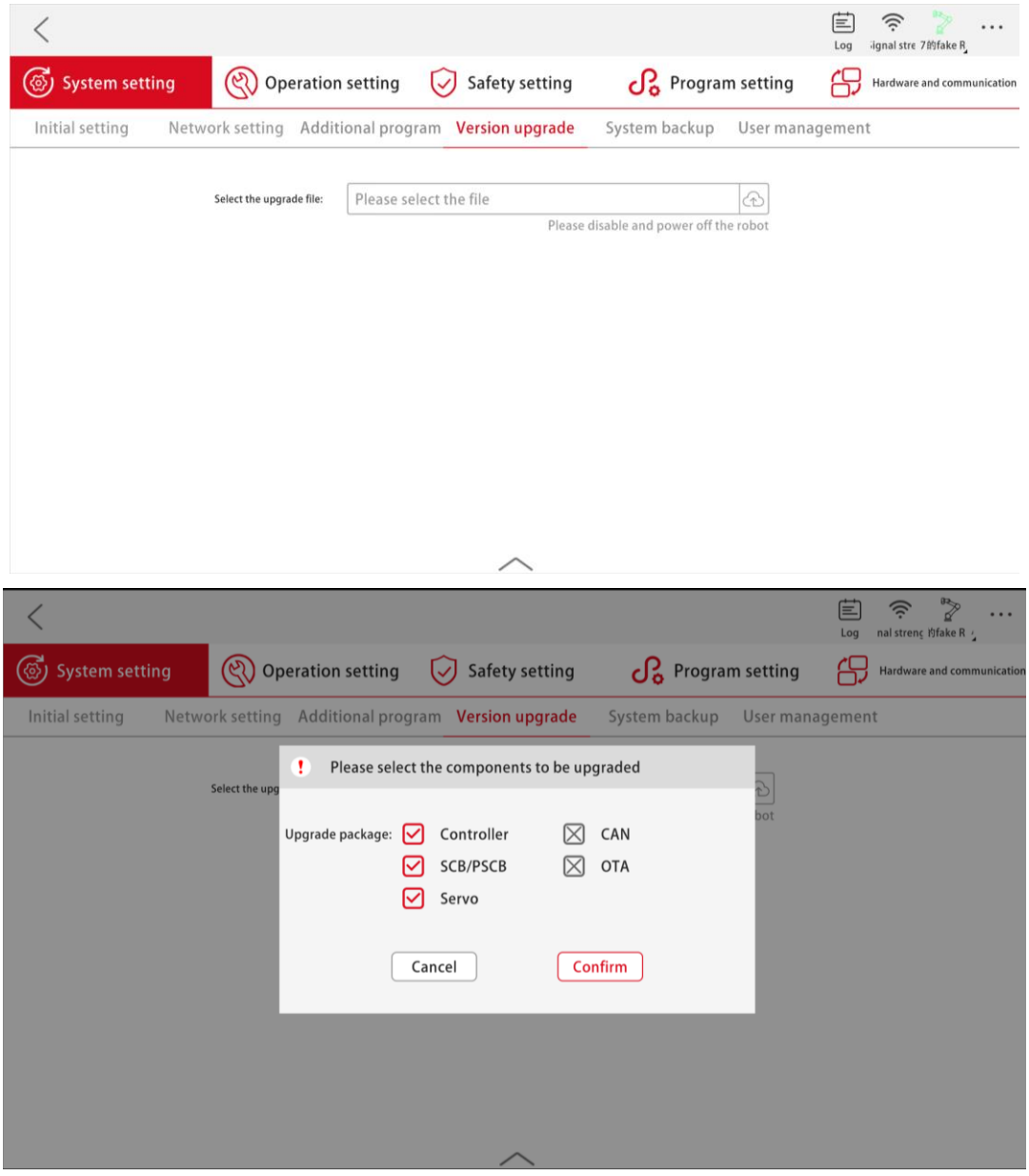


Figure 3-4 Version Update Interface

3.1.1.5 System Backup

System backup is composed of two parts: importing and exporting configuration files, and regular automatic program backup. See Figure 3-5 below.

(1) Importing and exporting configuration files

Role: Configuration files are user configuration parameters that are modified by the user when using the robot, which include controller setup parameters, IO name parameters, dynamic IO parameters, system variable parameters, and safety zone parameters.

How to export: Select the configuration file to be exported and click "Export File"; Select the path to save the configuration file, and click OK to complete the export.

How to import: Click "Import File" to select the save path of the configuration file; Select the configuration file to be imported and click OK; Restart the electrical control cabinet after the file is successfully imported. **Note:** For configuration file importing, it is required to keep controller versions consistent, and importing between different versions of configuration files may lead to controller abnormalities.

(2) Start regular automatic program backup

Role: After regular automatic program backup started and the program is modified, App will start timing according to the set time, and automatically save a backup of the current program when the time is up, which is named by "program name" + "." + "system time when saving".

When automatic backup started, you can open the program in the programming control interface to enter program file list and click "Hide Backup File" in the upper right corner to view it.

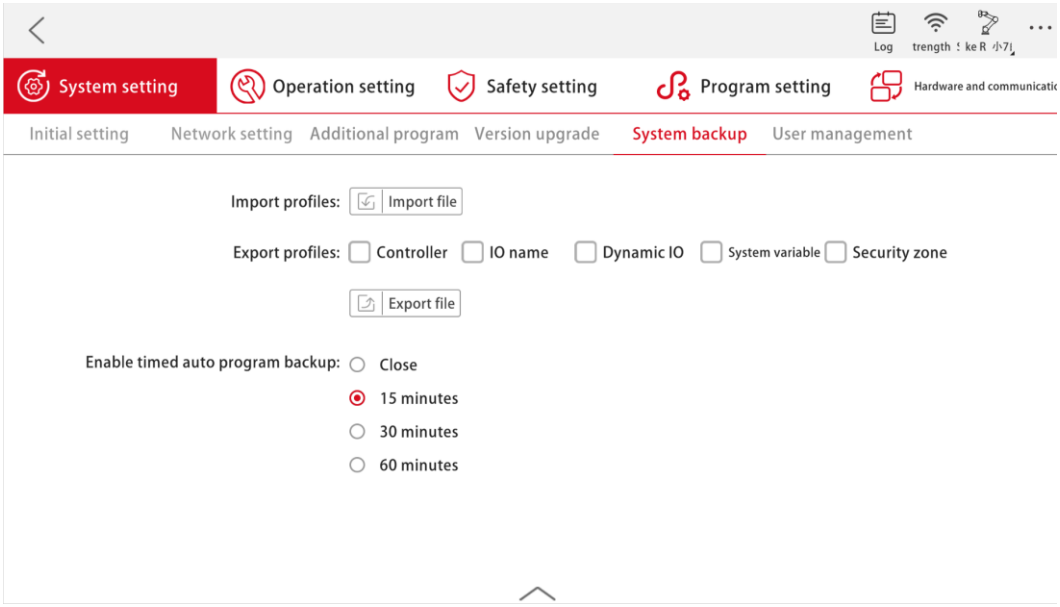


Figure 3-5 System Backup Interface

3.1.1.6 User (Password) Management

The administrator can change the password for different operation authorities under user management settings, as shown in Figure 3-6. (Default password: Administrator: jakazuadmin Technician: 0000 Operator: 0) .

Note: This password is the default initial password, please change it on first use.

Enter the setting interface and select "System Setting"; Click "User Management" to select the corresponding user level to be changed and click "Edit" button to make changes; Click the text box according to the prompts to enter and click OK to complete the change.

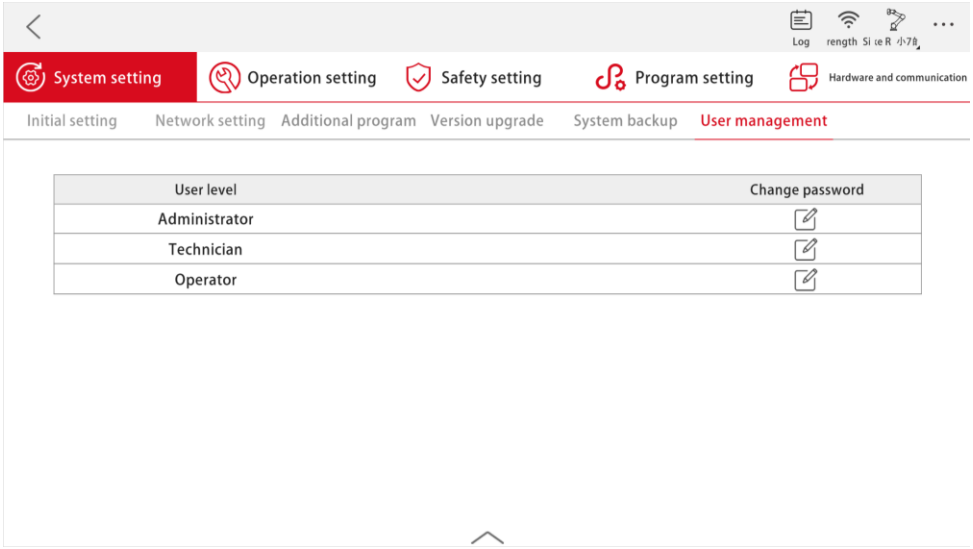


Figure 3-6 Password Change Interface

3.1.2 Operation Settings

3.1.2.1 Tool Coordinate System Settings

Tool coordinate system is a coordinate system whose origin is Tool Center Point (TCP), and the robot end position is the Cartesian coordinate of TCP in the current user coordinate system. The robot end pose is the pose of the tool coordinate system in the current user coordinate system, which is represented by RPY. "TCP" represents tool coordinate system in JAKA Zu App. The default tool coordinate system for the robot is the "flange coordinate system". The origin of flange coordinate system is the center of the flange robot end; The outward direction of flange end face is the Z-axis positive direction; The direction of the line connecting the flange center and TIO is the Y-axis negative direction; and the X-axis positive direction is defined according to the right-handed rule.

The flange coordinate system parameters cannot be modified, and in operations requiring high robot motion accuracy, TCP is generally set at the end of the robot end-effector, such as the gripper end, the center of suction cup, etc. Therefore, JAKA Zu robot provides 10 TCPs with editable settings for users, except the default tool coordinate system. Users can edit TCP parameters according to their needs by selecting "manual input", "four-point setup", "six-point setup", etc. The introduction and description of three methods are as follows.

(1) Manual input (as shown in Figure 3-7): Users calculate the position offset of the required tool coordinate system relative to the flange coordinate system based on their needs, fill these calculated data into the corresponding data box, and click OK to complete TCP parameter editing.

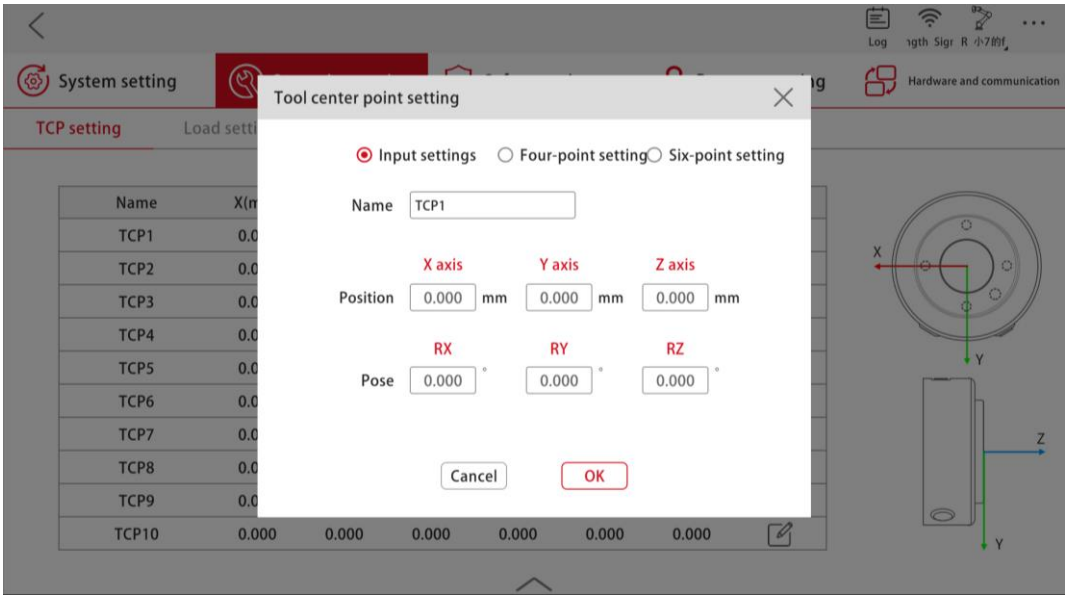


Figure 3-7 Input Setting Interface of Tool Coordinate System

(2) Four-point setup (as shown in Figure 3-8): After determining a fixed point in the space, users control the robot in four different positions to make the TCP end point reaches the fixed point, and the desired position offset of the tool coordinate system relative to the flange coordinate system can be automatically calculated via "four-point setup". The specific setting process is shown as follows:

1. Find a fixed reference point within reach of the robot, such as the vertex of a spiked cone.

2. Click "Set Position Point1" to manually operate the robot, so that the effector end can reach the reference point position. Click OK.

3. Click "Set Position Point2" to manually operate the robot, so that the effector end can reach the reference point position in joint angle different from that of position point 1. Click OK.

4. Click "Set Position Point3" to manually operate the robot, so that the effector end can reach the reference point position in joint angle different from those of position point 1 and position point 2. Click OK.

5. Click "Set Position Point4" to manually operate the robot, so that the effector end can reach the reference point position in joint angle different from those of position point 1, position point 2 and position point 3. Click OK.

6. Click OK to get the parameters of the desired tool coordinate system.

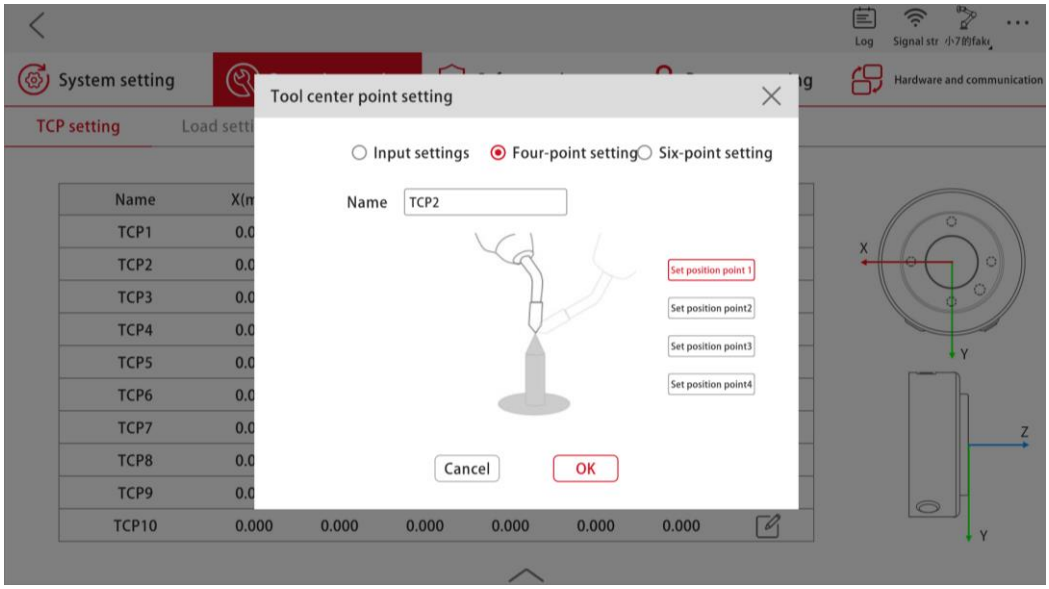


Figure 3-8 Four-point Setup Interface of Tool Coordinate System

(3) Six-point setup (as shown in Figure 3-9): Based on "Four-point Setup", determine the TCP axis direction by specifying two additional position points, so that TCP position and position parameters can be calculated automatically to get parameters of desired tool coordinate system. Six-point setup is generally applied when the robot end tool axis is not perpendicular or parallel to the robot end flange, and the Z axis of the tool coordinate system can be aligned with the robot end axis by "Six-point setup".

The specific setting process is shown as follows:

1. Find a fixed reference point within reach of the robot, such as the vertex of a spiked cone.

2. Click "Set Position Point1" to manually operate the robot so that the effector end can reach the reference point position. Click OK.

3. Click "Set Position Point2" to manually operate the robot, so that the effector end can reach the reference point position in angle different from that of position point 1. Click OK.

4. Click "Set Position Point3" to manually operate the robot, so that the effector end can reach the reference point position in angle different from those of position point 1 and position point 2. Click OK.

5. Click "Set Position Point4" to manually operate the robot, so that the effector end can reach the reference point position in angle different from those of position point 1, position point 2 and position point 3. Click OK.

6. Click "Set Position Point5", keep position point 4 unchanged, and move forward along the Z-axis positive direction of the desired tool coordinate system to get position point 5. Click OK.

7. Click "Set Position Point6", keep position point 5 unchanged, and move forward in the desired XOZ plane to get position point 6. Click OK.

8. Click OK to get the parameters of the desired tool coordinate system. The direction of the line between position point 4 and position point 5 above is the Z-axis positive direction of the desired tool coordinate system.

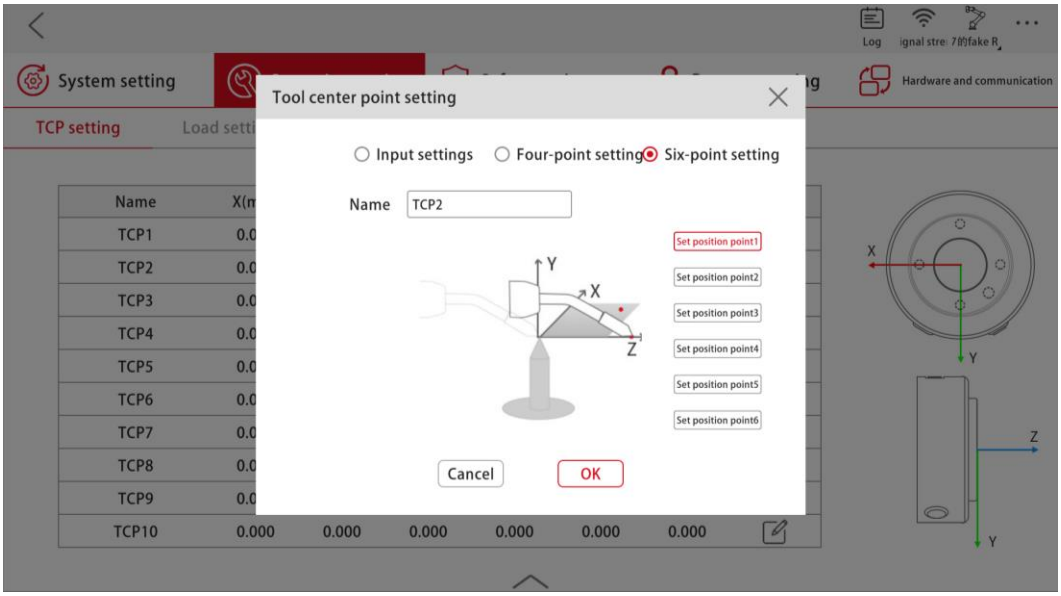


Figure 3-9 Six-point Setup Interface of Tool Coordinate System

3.1.2.2 Load Information Settings

The load is the mass and centroid position of all objects mounted on the robot end. Correct load information settings allows the controller to calculate the actual state of the robot correctly. If the load settings deviates significantly from the actual situation, the controller may incorrectly recognize collision while the robot is in motion, thus interrupting the robot motion. There are two ways to set the load: manual input mode and automatic identification mode.

(1) Manual input mode (as shown in Figure 3-10):

Users get the accurate load information according to calculation or measurement, and fill the mass and centroid position of the load into the corresponding input box correctly. Clicks OK to complete the setting.

Note: The centroid position is relative to flange center at the robot end, and the X, Y, and Z coordinates of the centroid position are also spatial values in the flange coordinate system. It is recommended to calculate in 3D design software to improve accuracy.

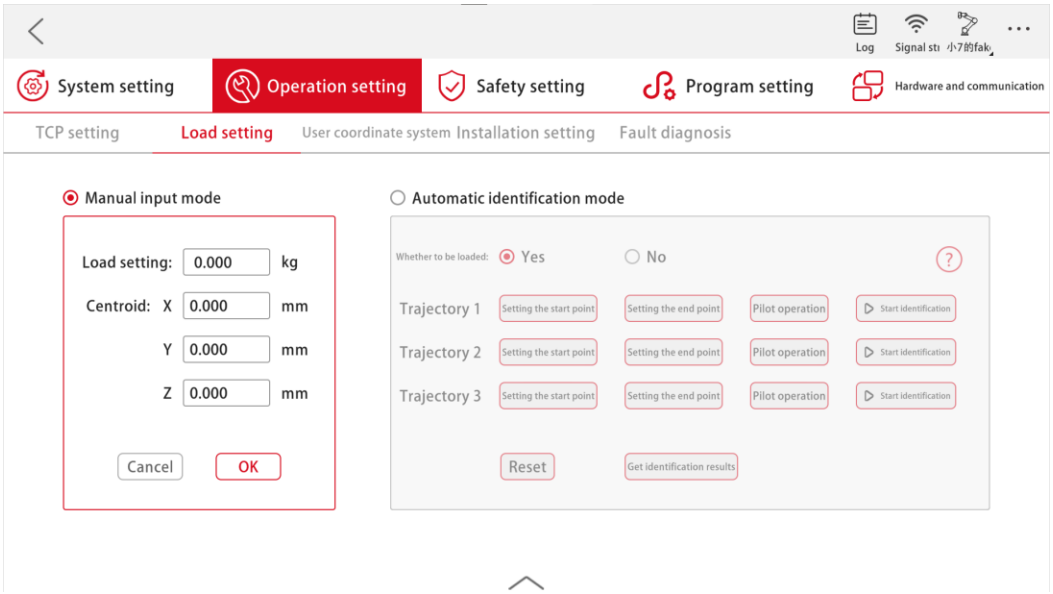


Figure 3-10 Load Manual Input Setting Interface

(2) Automatic identification mode (as shown in Figure 3-11):

Recognize and calculate the mass and centroid position of the load by robot motion from a fixed position.

Setup Process (Zu, C, Pro Series):

a Robot operation is divided into two modes: no-load and with-load. In general, the latter is preferred. Load the robot and confirm that there is no interference in the recognition trajectory. It is necessary to move the robot to a general position (0, 90°, 0, 0, 180°, 0) first, keeping it upright, while ensuring that the payload is on the outside of the whole robot to avoid interference to some extent.

b Trajectory 1. Click [Set initial point] to enter manual setting interface, where the pop-up window prompts [first, move the robot joint 2 to 90°, joint 3 to 0, joint 5 to 180°, and joint 4 is allowed to move in range (-60°, 60°), keep joint 4 and 6 in consistent angle]. Exit manual setting interface and save the initial point; Click [Set end point], and the movement limit is the same as that of [Set initial point]. Exit manual setting interface and save the end point.

c Trajectory 2. Click [Set initial point] to enter manual setting interface, where the pop-up window prompts [first, move the robot joint 2 to 90°, joint 3 to 0, joint 5 to 180°, and joint 4 is allowed to move in range (-60°, 60°). The angle value of joint 6 is 90° larger than that of joint 4]. Exit manual setting interface and save the initial point; Click [Set end point], and the movement limit is the same as that of [Set initial point]. Exit manual setting interface and save the end point.

d Trajectory 3. Click [Set initial point] to enter manual setting interface, where the pop-up window prompts [first, move the robot joint 2 to 90°, joint 3 to 0, joint 4 to 0, joint 5 is allowed to move in range (170°, 190°) and smaller than 180°]. Exit manual setting interface and save the initial point; Click [Set end point], and the pop-up window prompts [first, move the robot joint 2 to 90°, joint 3 to 0, joint 4 to 0, joint 5 is allowed to move in range (170°, 190°) and larger than 180°]. Exit manual

setting interface and save the end point.

e Press and hold [Set initial point] to go back to the initial point; Press and hold [Pilot operation] to the end point, and judge whether there is any interference in the trajectory. Click [Start identification]. If the robot is not at the initial point, it will prompt to move to the initial point.

f For no-load operation, it is not required to reset the trajectory. Click [Start identification] directly. if users set the trajectory, the pop-up window will prompt [trajectory has been recognized by load identification, do not repeat setting].

Setup Process (Cobot Series):

a Robot operation is divided into two modes: no-load and with-load. In general, the latter is preferred. Load the robot and confirm that there is no interference in the recognition trajectory. It is necessary to move the robot to a general position (0, 0, 90°, 0, 0, 0) first, keeping joint 3, 4, 5, 6 in horizontal state.

b Trajectory 1. Click [Set initial point] to enter manual setting interface, and the pop-up window prompts [first, move the robot joint 2, 4, 6 to 0, joint 3 is allowed to move within (60°, 120°), and it is not required to set joint 1 and 5]. Exit manual setting interface and save the initial point; Click [Set end point], and the movement limit is the same as that of [Set initial point]. Exit manual setting interface and save the end point.

c Trajectory 2. Click [Set initial point] to enter manual setting interface, and the pop-up window prompts [first, move the robot joint 2, 5 to 0, joint 3 to 90°, joint 4 is allowed to move within (-60°, 60°), and the angle of joint 4 and 6 adds up to 90°]. Exit manual setting interface and save the initial point; Click [Set end point], and the movement limit is the same as that of [Set initial point]. Exit manual setting interface and save the end point.

d Trajectory 3. Click [Set initial point] to enter manual setting interface, and the pop-up window prompts [first, move the robot joint 2, 5 to 0, joint 3 to 90°, joint 4 is allowed to move within (-60°, 60°), and the angle of joint 4 and 6 adds up to 0]. Exit manual setting interface and save the initial point; Click [Set end point], and the movement limit is the same as that of [Set initial point]. Exit manual setting interface and save the end point.

e Press and hold [Set initial point] to go back to the initial point; Press and hold [Pilot operation] to the end point, and judge whether there is any interference in the trajectory. Click [Start identification]. If the robot is not at the initial point, it will prompt to move to the initial point.

f For no-load operation, it is not required to reset the trajectory. Click [Start identification] directly. if users set the trajectory, the pop-up window will prompt [trajectory has been recognized by load identification, do not repeat setting].

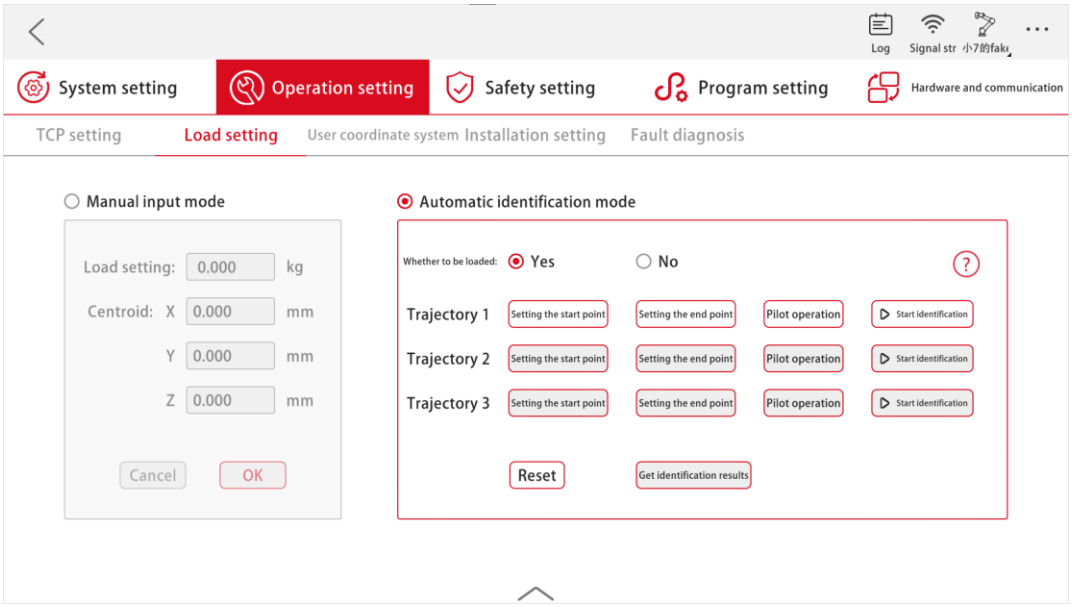


Figure 3-11 Automatic Load Recognition Setting Interface

3.1.2.3 User Coordinate Settings

The default user coordinate system for the robot is the world coordinate system. The world coordinate system takes the robot base center as the origin, the direction of the base pointing vertically to the robot arm as the Z-axis positive direction, the direction of the line connecting the base center to the interface of the heavy load line as the X-axis positive direction, and the Y-axis positive direction is defined according to the right-handed spiral rule. This TCP parameter cannot be modified.

In addition to the world coordinate system, JAKA Zu robots provide 10 user coordinate systems with editable settings for users. Users can choose "Manual input" or "Three-point setup" to edit user coordinate system parameters according to their needs. The following is the introduction and description of these three methods.

(1) Manual input (as shown in Figure 3-12): Users calculate the position offset of the required user coordinate system relative to the world coordinate system based on their needs, fill these calculated data into corresponding data boxes, and click OK to complete parameter editing.

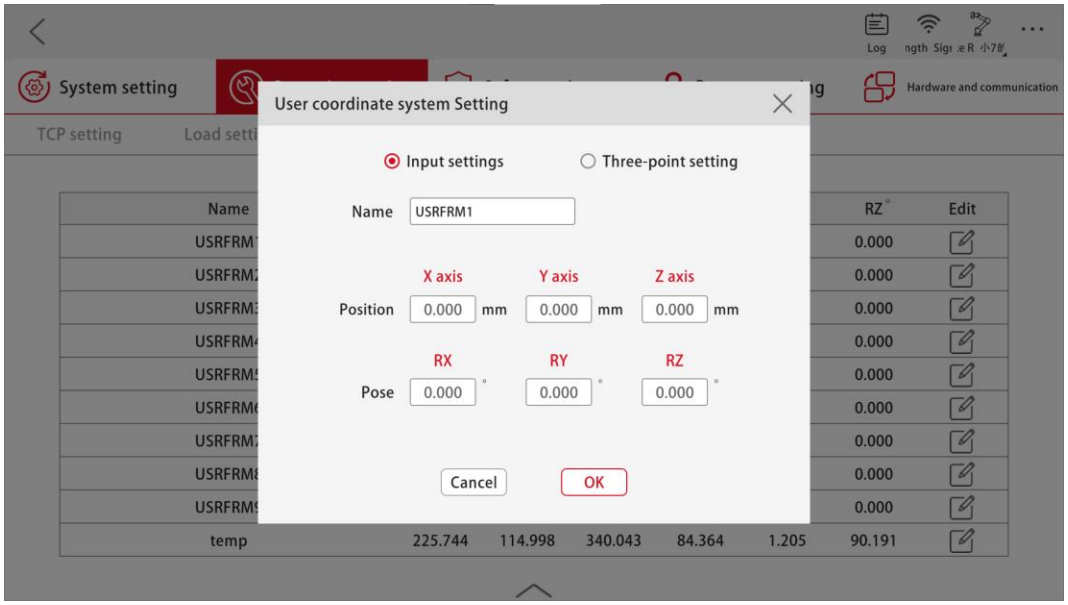


Figure 3-12 User Coordinate System Input Setting Interface

(2) Three-point setup (as shown in Figure 3-13):

Role: The parameters in the X, Y and Z axis directions of the desired user coordinate system are calculated automatically from the 3 position points set by the user. The X, Y and Z axis directions of the TCP generated by "Three-point setup" are consistent with those of the world coordinate system.

Definition of the three points:

Position point 1: The origin of the user coordinate system.

Position point 2: Any point on the X-axis positive direction of user coordinate system.

Position point 3: Any point on the first quadrant of the X-Y plane of user coordinate system.

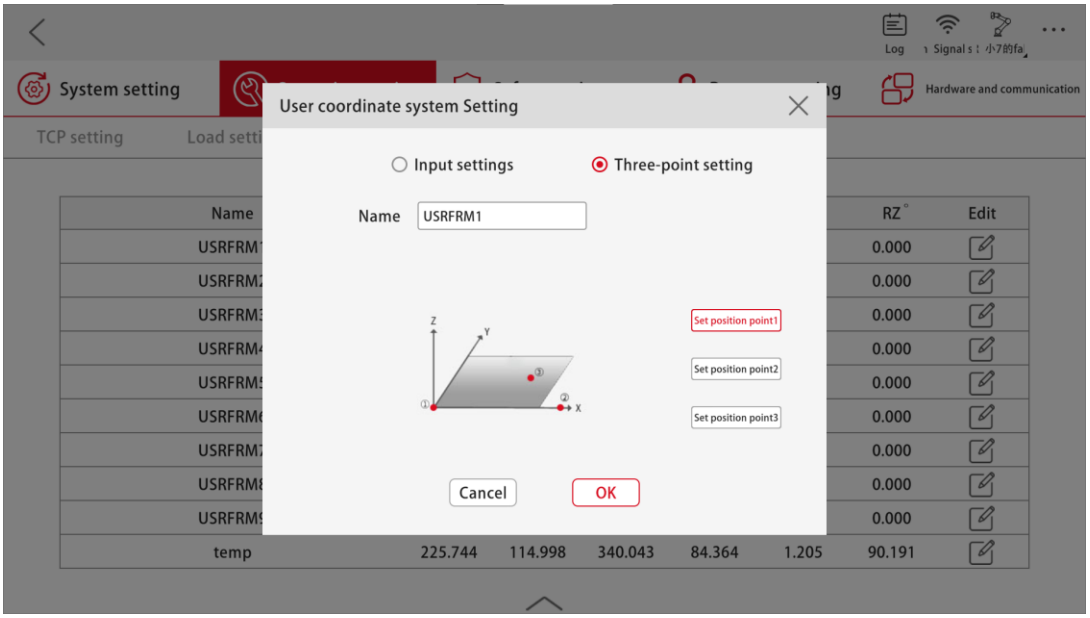


Figure 3-13 Three-point Setup Interface of User Coordinate System

3.1.2.4 Installation Position Settings

Zu Robot supports installation at any angle. After Zu Robot mounting, it is required to enter the

installation position information of Zu Robot into the software to ensure safety during operation. Click "Settings" in the function bar; Select "Installation Settings"; Adjust the installation position of the robot model in the software according to actual robot mounting position; Click the corresponding button in top right side to select reverse installation, side installation or front installation (default front installation); and click the directional triangle button to select the installation angle to ensure that installation position of the robot model in the software is consistent with the actual situation. Click OK to take effect. See Figure 3-14.

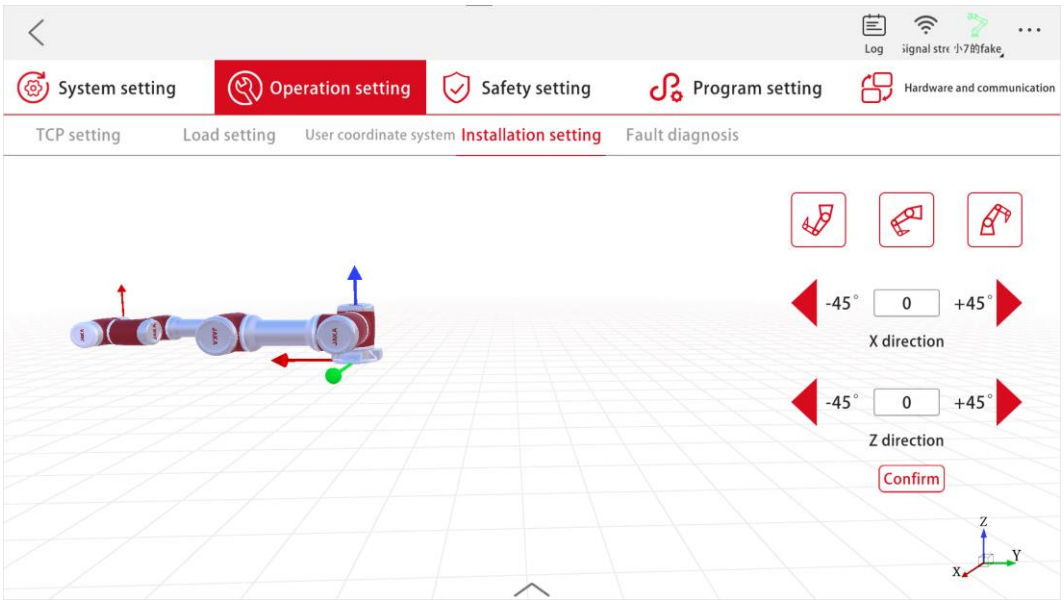


Figure 3-14 Installation Setting Interface

3.1.2.5 Failure Diagnosis

When robot failure occurs, the controller automatically saves the information related to the failure as a zip file named by the current system time, and displays it in the failure diagnosis interface. See Figure 3-15.

However, users can also click "Log Failure" button in the failure diagnosis interface even if the robot operates well, and App will pop-up "Start failure diagnosis". Then "Failure diagnosis in progress" prompt button appears next to the log button. Failure diagnosis will be completed automatically in about 30 seconds. Click "Failure diagnosis in progress" button in the upper menu bar or "Stop" button to stop failure diagnosis in advance.

Note: After clicking the diagnosis button for the first time, you can click the diagnosis button again to extend the diagnostic time.

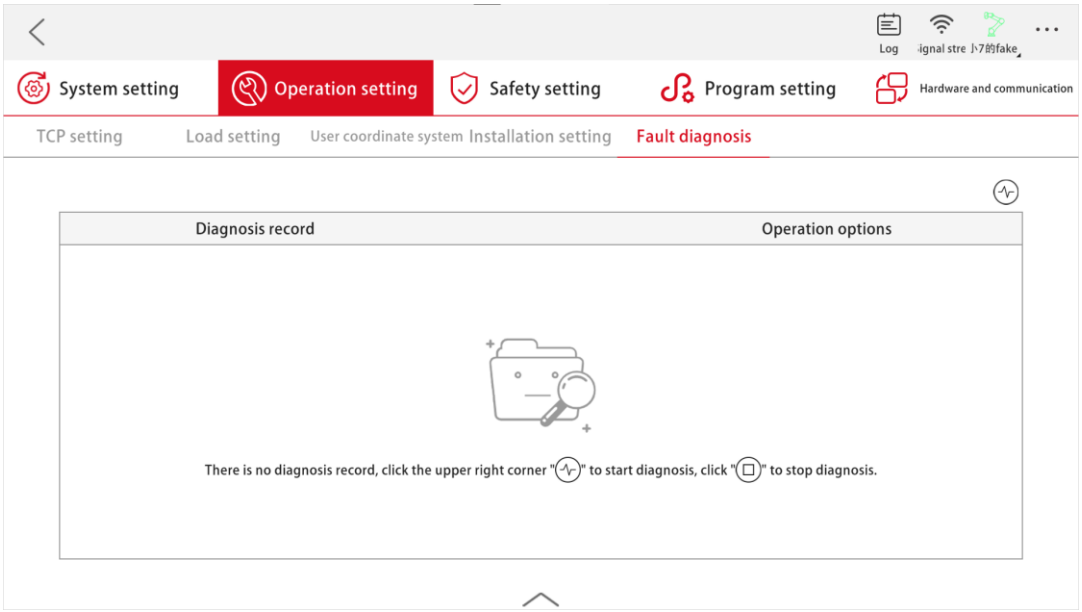


Figure 3-15 Failure Diagnosis Interface

3.1.3 Safety Settings

3.1.3.1 Joint Limit Settings

In the joint limit interface, users can set the soft limit angle, joint speed limit, and error alarm threshold for each joint. See Figure 3-16.

Note: The error alarm threshold is the robot alarm prompt when the robot arm motion offset error is greater than the alarm threshold.

Note: The default values of joint positive limit, joint negative limit, and joint speed limit are the maximum range, and users can only modify them within the default range.

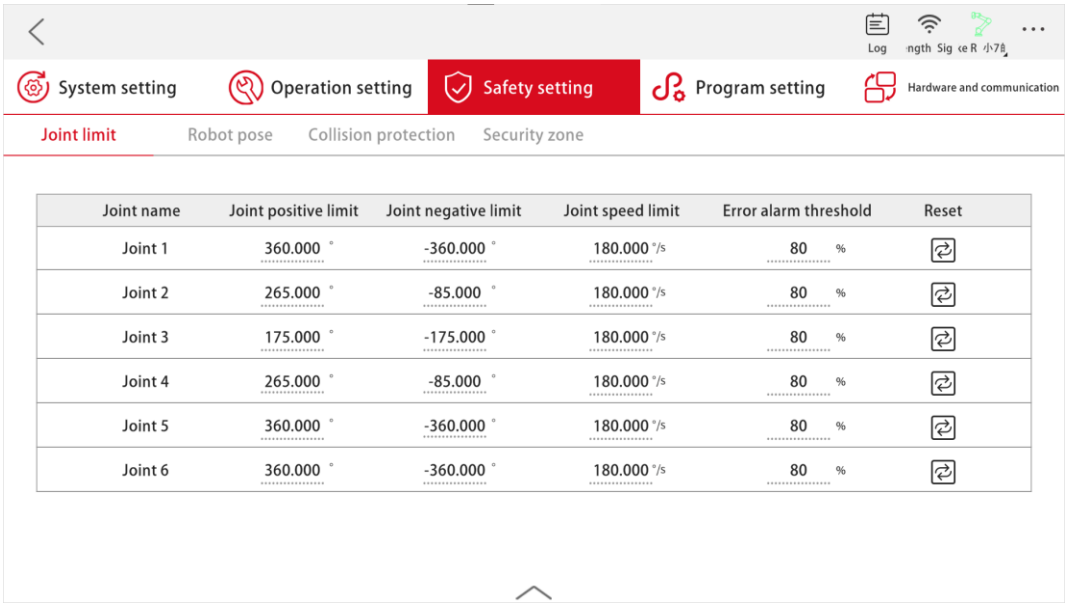


Figure 3-16 Joint Limit Interface

3.1.3.2 Robot Pose Settings

In the robot pose interface, there are three poses: factory pose, open pose, and safe pose. See

Figure 3-17.

Factory pose: The robot state when packing.

Open pose: Zero position pose for joint zeroing.

Safe pose: Safe pose for user editing. Users can set a safe pose as the reset pose, which can be reached via robot handle. When the robot reaches its reset pose, the function IO "Safe position" will be triggered.

Click "Modify" button to modify the pose position. Press and hold "Move to target point" to move the robot to the target position.

Safety pose error: Allows the user to set the safety error within which the functional DO signal output of safety pose is valid.

Moving speed: Users can adjust the moving speed to control the robot speed to the target point.

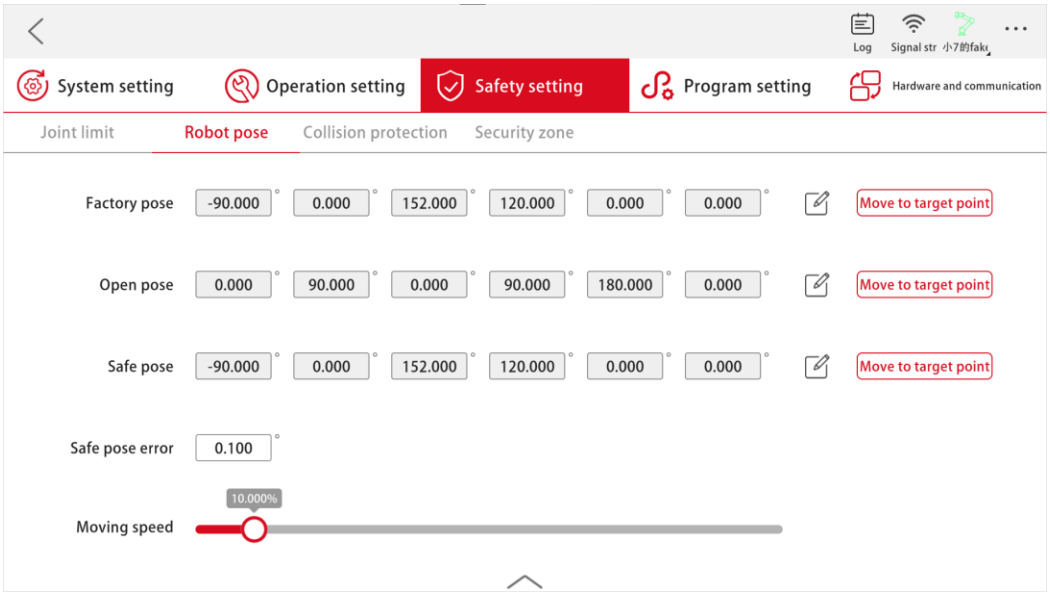


Figure 3-17 Robot Pose Setting Interface

3.1.3.3 Protection System Setting

Operating with external vision function, JAKA robot can realize safety protection. Customers can define the deceleration area and collaboration area according to the actual working condition, and when people or external objects enter the corresponding area, the robot will recognize and enter corresponding deceleration or stop state to prevent safety accidents caused by inadvertent entry.

Operation steps:

① Extended I/O setting on App (as shown in Figure 3-18 to 3-21)

Open JAKA Zu software and connect to the robot to enter I/O panel interface. Click "Edit" button in the upper right corner, and click "Add" button to add an extended IO module. Set DI0 to the reduction mode, and DI1 to the protective stop mode. When the extended I/O function is set up, turn on visual protection and switch the extended I/O to running status to establish real-time communication between visual protection devices and the controller.

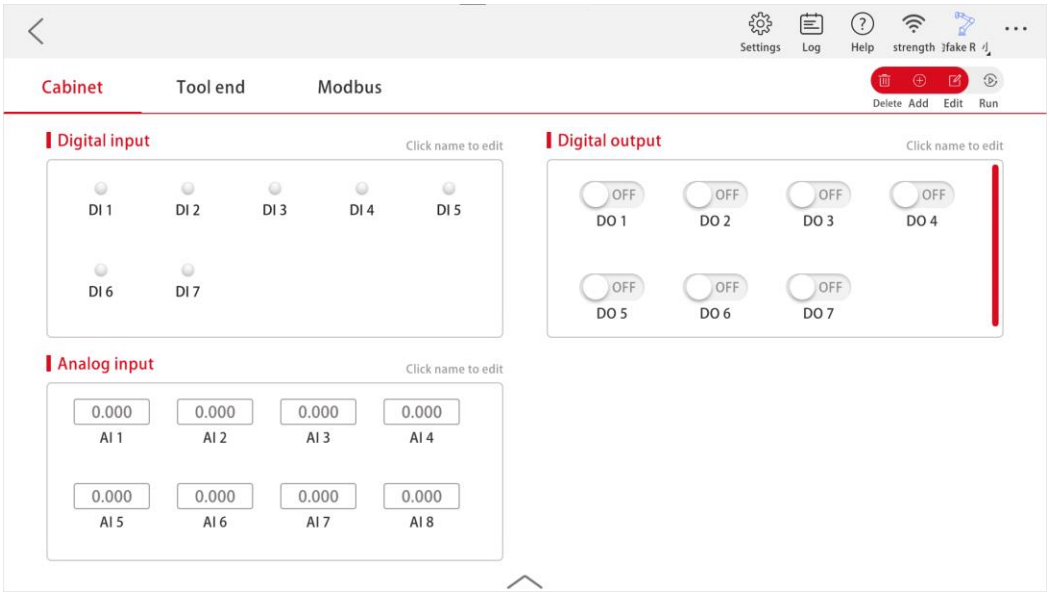


Figure 3-18 Schematic Diagram of Extended I/O Addition

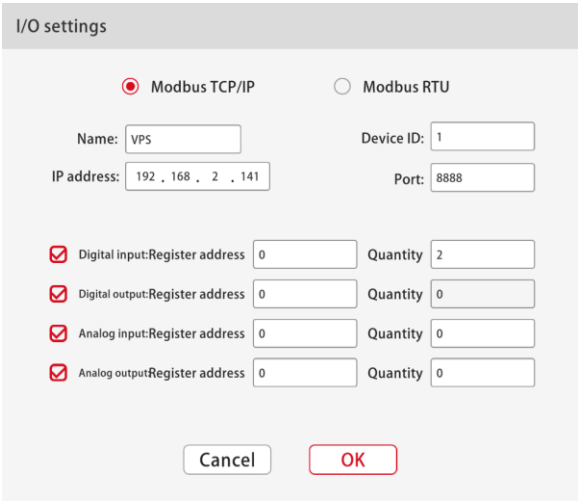


Figure 3-19 Extended I/O Configuration Interface

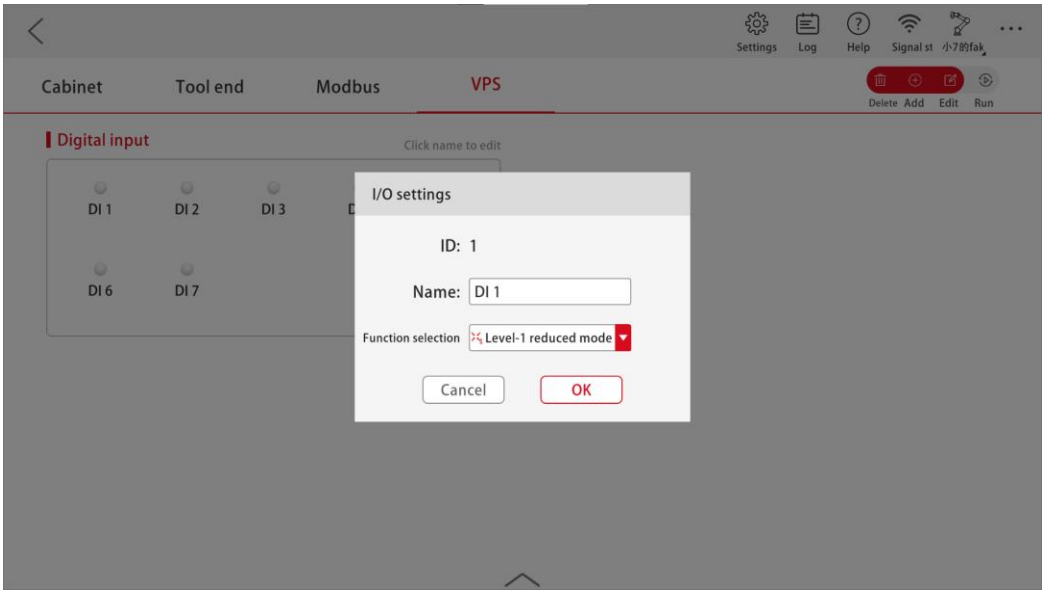


Figure 3-20 Extended I/O Digital Input 1 Setting Interface

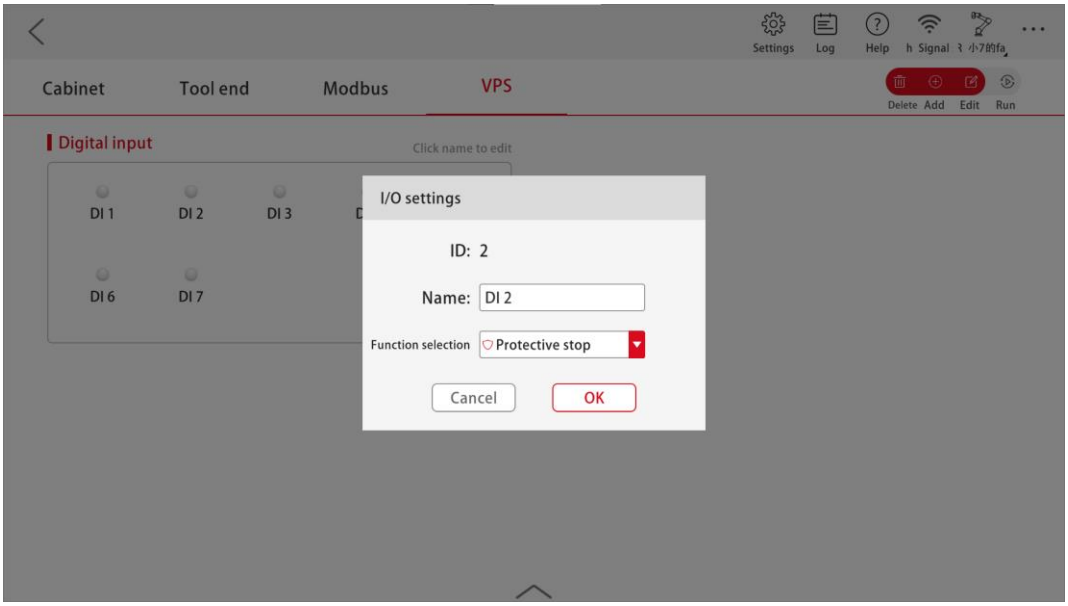


Figure 3-21 Extended I/O Digital Input 2 Setting Interface

② Parameter Settings on App

As shown in Figure 3-22, click to enter "Settings" - "Safety Settings" - "Protection System" and click "Connection Settings" button. When the controller and the protective camera are under the same LAN, the information of the currently connected protective camera will be displayed. Click "Parameter Settings" to configure the shape, sensitivity, sensitometry and reduction magnification of the protection area. The protection system supports three types of protection area shape settings, including rectangle, circle and self-defined curves.

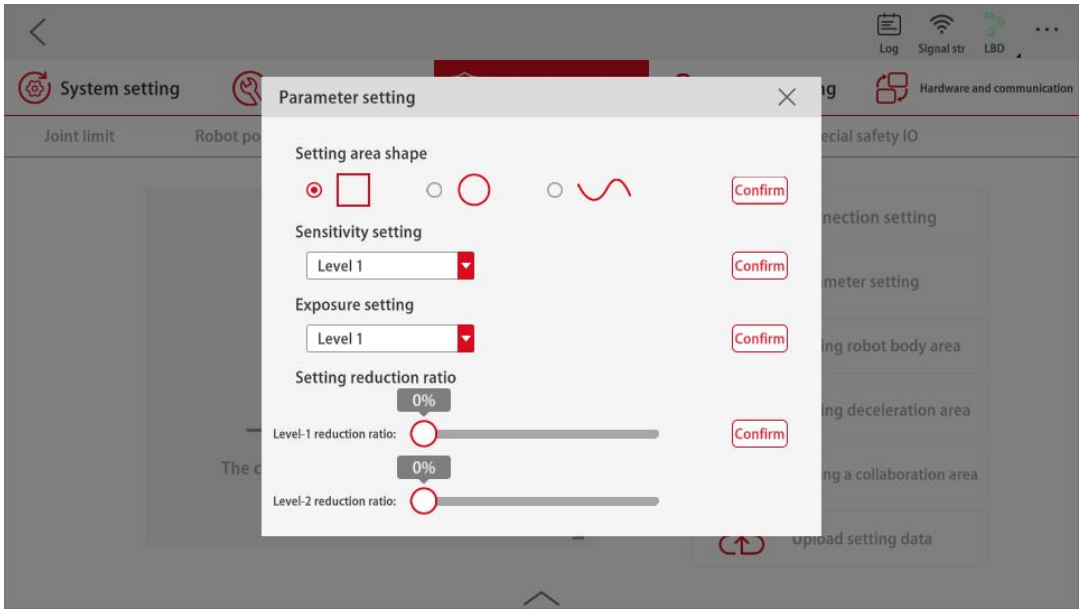


Figure 3-22 Visual Protection Connection Setting Interface

There are three areas in the protection system: robot arm area, deceleration area, and collaboration area. As shown in Figures 3-23 to 3-25. The function of these three areas are shown as follows:

Robot Arm Area

As the name implies, it is the area occupied by the robot itself. Since the robot is in motion during actual program operation, it is required to exclude the active area of the robot from the algorithm in order to avoid detecting the robot as an intruding object. In this module, the robot arm area can be set as rectangle, circle or self-defined curves. Setting Method: Operate the job program that the robot will execute in actual operation. Enter the visual protection setting interface for setting up robot body area, and ensure that the robot is within the set robot body area circle at all times.

Deceleration Area

This area is the first level of protection area. When an object invades the area, visual protection devices will send information to the controller by which the robot enters reduction mode. The area is set according to the shape selected by the user in the beginning, as shown in the following figure. Since the initial setting mode selected is rectangle, so the shape of self-defined deceleration area is rectangle. After selecting the area, click "Upload Settings Data" to send the setting information to the processor and store it locally. After uploading, the area settings take effect, and will remain in effect when the visual protection system is turned on again. Reset and upload settings again to change area settings if necessary.

Collaboration Area

This area is the second level of protection area. When an object invades the area, visual protection devices will send information to the controller by which the robot enters protective stop mode. The area is set according to the shape selected by the user in the beginning, as shown in the following figure. Since the initial setting mode selected is rectangle, so the shape of self-defined deceleration area is rectangle. After selecting the area, click "Upload Settings Data" to send the setting information to the processor and store it locally. After uploading, the area settings take effect, and will remain in effect when the visual protection system is turned on again. Reset and upload settings again to change area settings if necessary.

Figure 3-23 Robot Arm Area Setting Interface

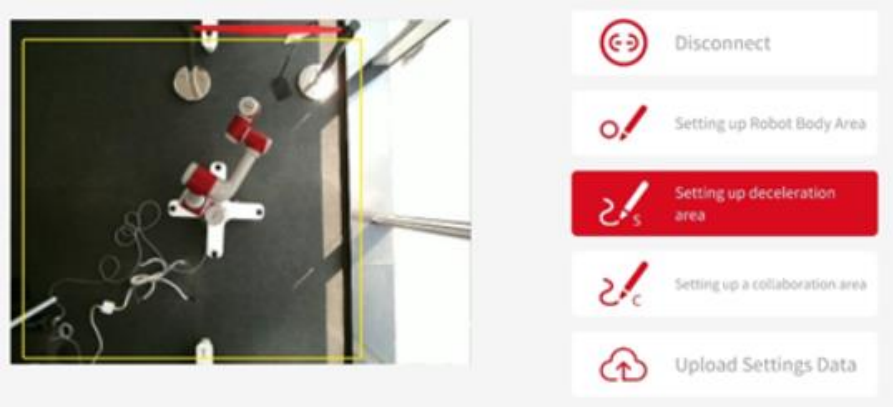


Figure 3-24 Deceleration Area Setting Interface

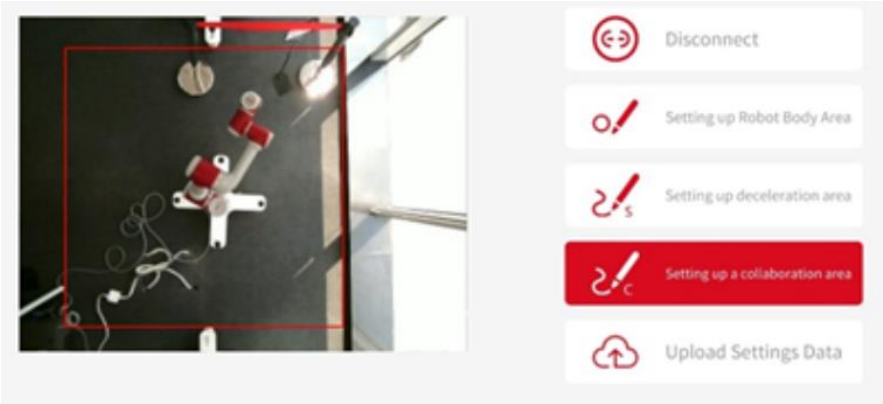


Figure 3-25 Collaboration Area Setting Interface

The visual protection function can only take effect on the robot after setting the extended I/O as running mode on App. If you want to turn off the visual protection function, please set the extended I/O to edit mode or disconnect the network cable to stop the visual protection program.

Note:

a. Ensure that there are no interferences when setting self-defined areas;

b. The algorithm will take real-time images as the reference image for visual safety protection. Therefore, it is required to ensure that there are no other objects interfering in the set area when setting self-defined areas, i.e. in a state without object intrusion;

c. Ensure that the self-defined area setting logic is correct;

d. Since the protection level of the collaboration area is higher than that of the deceleration area, it is required to ensure that the collaboration area is inside the deceleration area when setting self-defined areas. When setting areas by means of custom curves, it is required to ensure that the drawn curve is a closed region as far as possible.

3.1.3.4 Collision Protection Settings

Collision protection setting interface includes collision setting and collision handling settings. There are two collision setting methods: quick settings and customization (See Figure 3-26).

(1) Quick settings: Users can quickly set robot collision protection sensitivity through quick settings.

(2) Customization: Users can also select customization to set collision protection sensitivity according

to their actual needs. Force limit is the limit level of collision force.

Note: Force limit levels are divided into 1-5 according to restrictions, and level 0 means no force limit. The smaller the limit value of momentum, TCP speed and power, the slower the robot will run, and users can set the limit value according to the actual situation.

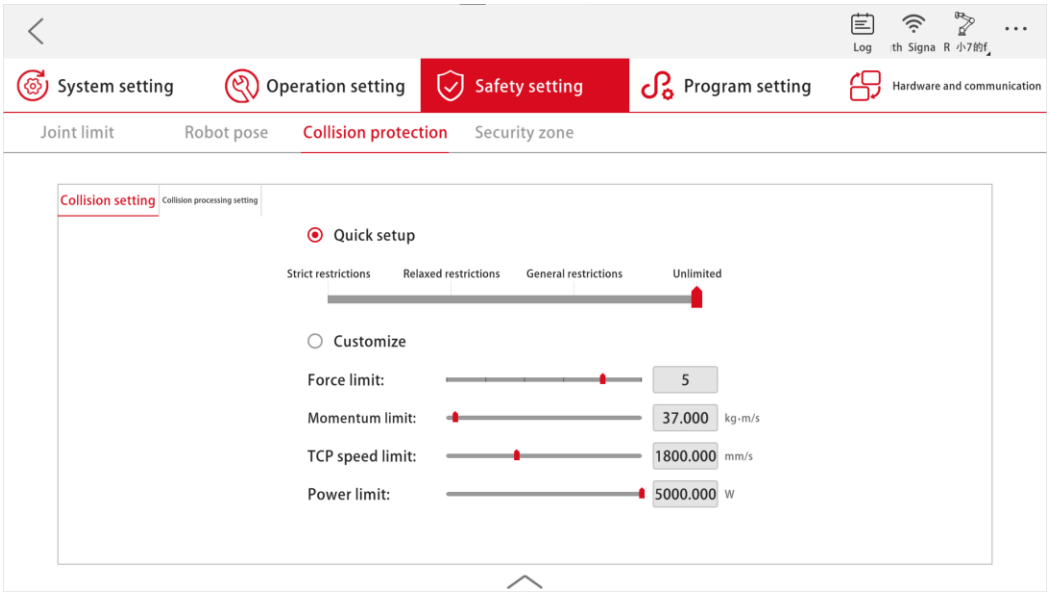


Figure 3-26 Collision Protection Setting Interface

In manual mode, if a collision occurs, the robot will not bounce back whether in motion or at rest, and the robot can be pushed within a certain range if the external force continues to act.

If a collision is detected by robot during automatic operation, there will be corresponding collision handling, as shown in Figure 3-27; the collision handling interface allows users to define handling method when a collision is detected by robot. Users can specify collision handling method during program runtime - to suspend the program (no bounce) or to terminate it and bounce.

- Suspend the program

The program is suspended without bounce. Subsequently, the user program can be controlled to continue running by the proceed operation function.

- Terminate the program and bounce

The program is terminated with bounce. In this interface, users can set the bounce angle, which ranges from 0 to 3°.

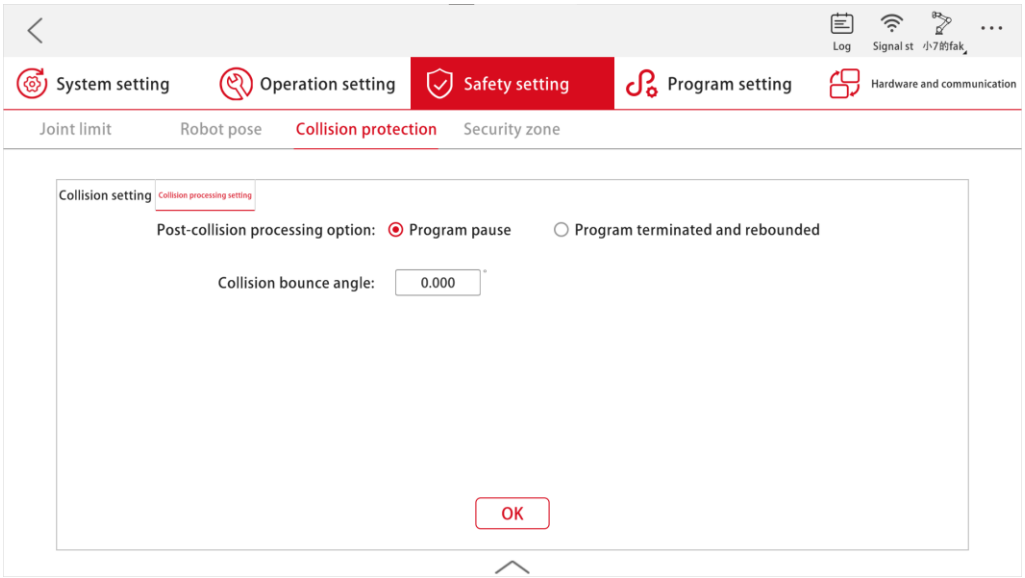


Figure 3-37 Collision Handling Setting Interface

3.1.3.5 Safety Area Settings

In order to avoid collisions between the robot and other objects during operation, users can set safety areas so that the robot end stops when it reaches the safety area.

Safety areas can be enabled in two ways: Enable when powered up and enable in operation. "Enable when powered up" means that the safety area is enabled when the robot is powered up. Once the robot reaches the safety area, it stops moving; "Enable in operation" means that the safety area is enabled only when the robot is in operation mode, and the safety area will not take effect in drag&drop or manual operation mode (See Figure 3-28).

Users can set up a total of 6 different safety areas, and safety area setting steps are shown as follows:

1. Click the arrow on the right of the safety plane option to be set to expand the editing list.
2. Enter the plane name.
3. Click the gear icon on the right of plane point 1 to enter manual operation interface. Operate the robot to the desired safety area and click OK.
4. Click the gear icon on the right of plane point 2 to enter manual operation interface. Operate the robot to another position in the desired safety area and click OK.
5. Click the gear icon on the right of plane point 3 to enter manual operation interface. Operate the robot to another position in the desired safety area and click OK. The controller will calculate the spatial location of safety area based on these three plane points, and display it in the 3D model on the left.
6. The safety plane point is set to determine the safety side of safety area. Click the gear icon on the right of the safety plane point to enter manual operation interface. Operate the robot to the safe side of the safety area, and click OK.
7. Enable this safety area in "Enable or not" column.
8. Safety distance (mm) is the distance between the robot end and the safety area. If the safety distance $\leqslant$ the value set by the user, the safety area will be triggered and the robot will stop.

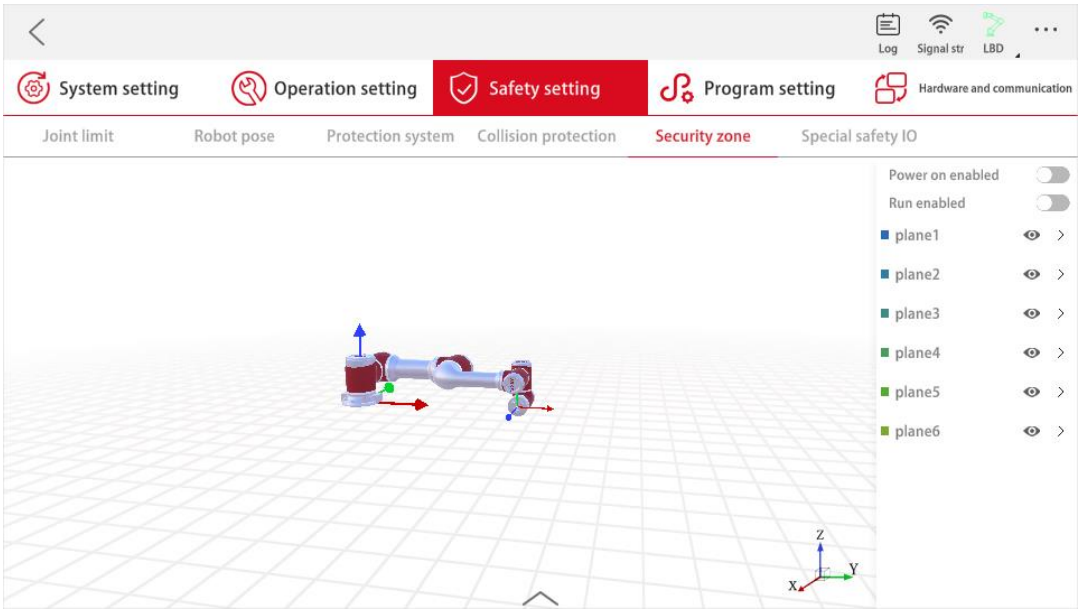


Figure 3-28 Safety Area Interface

3.1.3.6 Safety-related IO Settings

Note: This function is only applicable to version 2.1 of electrical control cabinet, and the rest versions are not supported at the moment.

The safety signal can be bound to the physical I/O of electrical control cabinet panel in safety settings, and the electrical control cabinet I/O can be set as a special safety I/O signal (double-loop signal) to control robot safety action and monitor robot safety state. The safety control and safety state are shown in Figure 3-29:

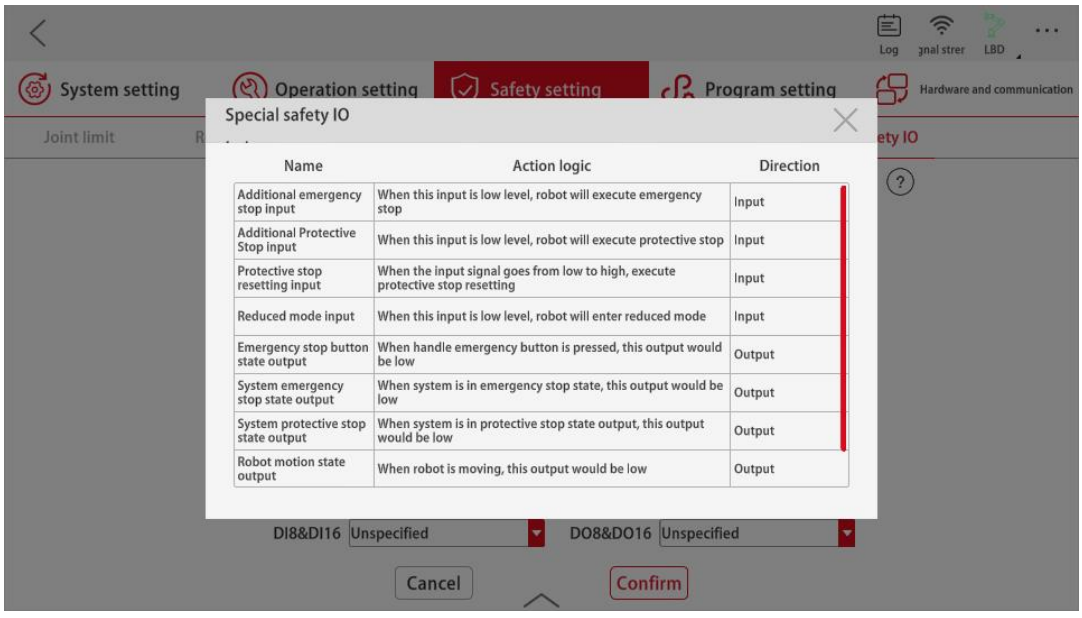


Figure 3-29 Safety Signal Description Interface

To set I/O for safety, first ensure the robot is in powered off and unenabled state; Click "Settings" to enter safety settings; Click "Safety-related I/O" to select the I/O to be set; Click the drop-down triangle to select the desired function, and click OK to take effect (See Figure 3-30).

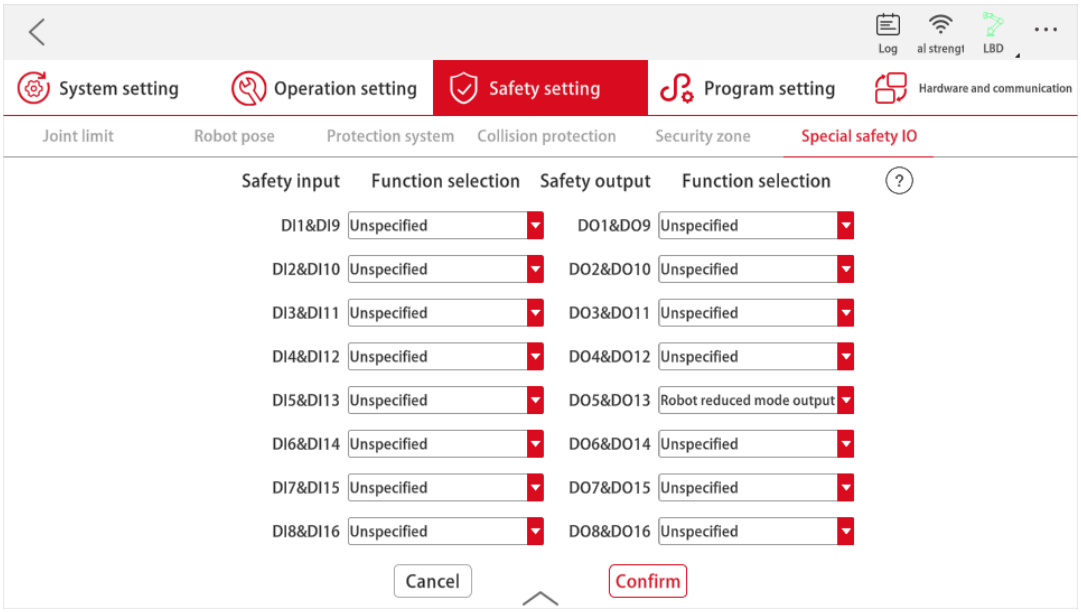


Figure 3-30 Safety-related IO Setting Interface

3.1.4 Program Settings

3.1.4.1 Default Program Management when Start

In program settings, users can specify a certain program as the default program. The robot can automatically load the default program when powered on by the options in program setting interface. The advantage is that it skips the process of logging in, powering up, enabling and selecting the program to be operated via App. Once the robot is powered on, you can operate the default robot program by enabling the robot and pressing the start button on the handle or the "start program" in the function I/O (See Figure 3-3).

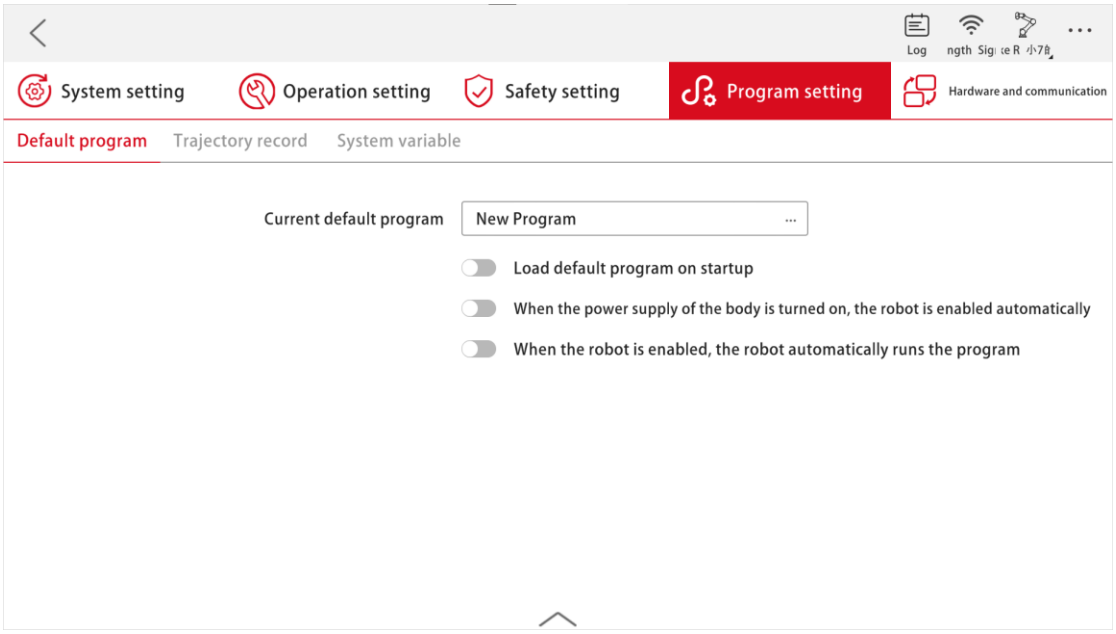


Figure 3-31 Default Program Setting Interface

3.1.4.2 Trajectory Recording Function

In "Trajectory Record" setting interface, users can make the controller record the robot trajectory and generate trajectory files by dragging or manual operation. During programming, these trajectory files can be called by trajectory record instructions, which allows the robot to reproduce demonstration trajectory in the program. (**Note:** The trajectory reproduction function can only record path information, and cannot record trajectory speed information.) As shown in Figure 3-32.

Movement trajectory recording method:

1. Click the “Settings” button to enter trajectory motion editing interface. Set the sampling accuracy of velocity, acceleration, position and posture of the sampled trajectory. If the motion distance of the trajectory is short, it is recommended to increase the sampling accuracy of position and posture to 0.1.

2. Click "Add" button, and the screen will pop up "Are you sure to start trajectory record?". Click "Yes", and "Trajectory record in progress" button will appear in the state bar on top of App.

3. The robot will make desired movement trajectory in drag mode or by manual operation.

4. After completing the action demonstration, click "Trajectory record in progress" button in the upper menu bar to stop movement trajectory recording, and trajectory files will be generated in the trajectory interface.

5. Click "Edit" button to modify the name of trajectory file.

6. You can call trajectory recording instructions for trajectory reproduction in programming control interface.

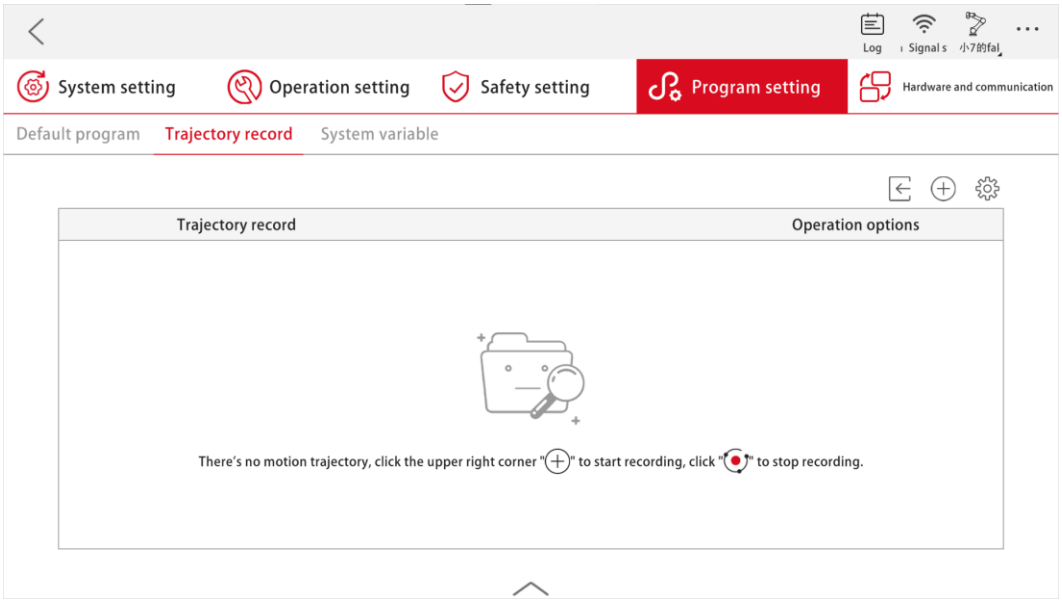


Figure 3-32 Trajectory Record Setting Interface

3.1.4.3 System Variables

JAKA Zu software can create system variables for use by all programs. System variables can only be numeric type variables, which are stored independently in the controller. The variable values will not be changed and reset due to program start and termination, robot power state, or electrical control cabinet shutdown. System variables can be called and modified in any of the programs. As shown in Figure 3-33.

System variables can be added by following steps: In "Program settings" - "System Variables", click

"Add" button ⊕ to enter system variable editing interface. Fill in the variable name and initial value in system variable editing interface, and click OK to add system variables.

Note: Up to 100 system variables can be stored.

In programming control interface, system variables can also be created by "New Variables" in variable instruction column.

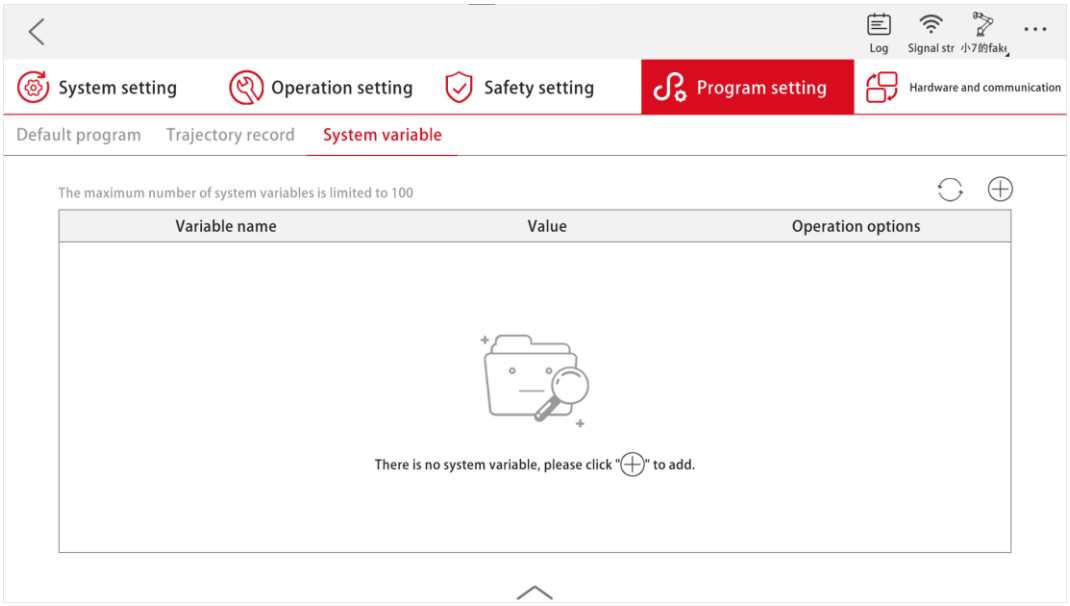


Figure 3-33 System Variable Setting Interface

3.1.5 Hardware and Communication

3.1.5.1 Modbus Communication Settings

JAKA Zu series robots support Modbus TCP/IP and Modbus RTU communication methods in which the robot serves as Modbus Server. Modbus IO usage is detailed in section 3.2.3. Modbus Address table is detailed in Appendix II. The steps are as follows:

Modbus TCP/IP:

Select Modbus TCP/IP mode in "APP Settings" → "Hardware & Communication" → "Modbus Setup" to modify the parameters and click OK. Restart the electrical control cabinet after "Setting succeeded" pop-up to complete connection. As shown in Figure 3-34.

Port Number: Consistent with Client port number.

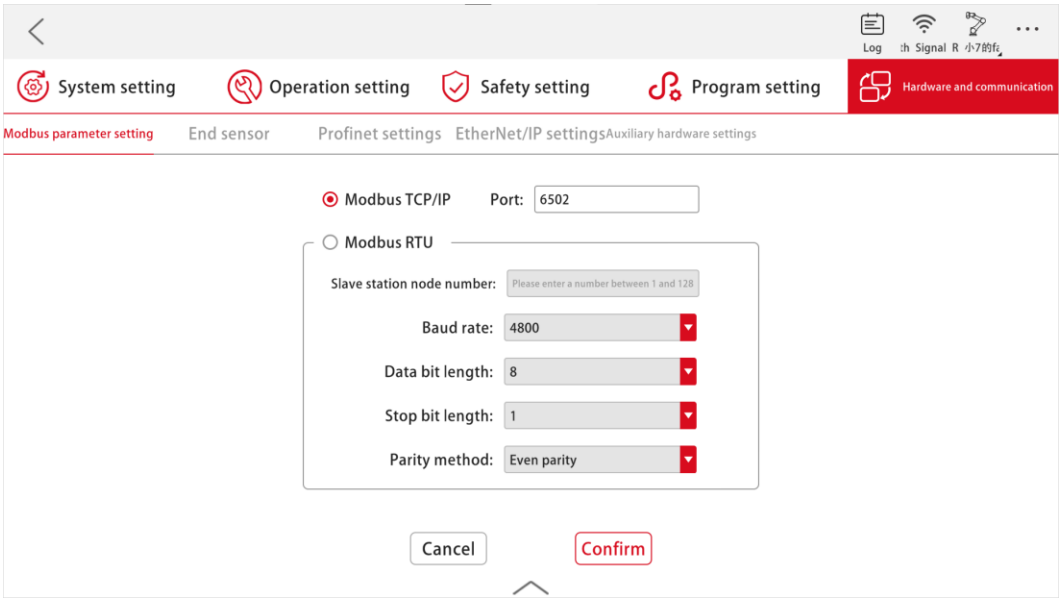


Figure 3-34 Modbus Parameter Interface

After establishing the connection on Client side, it is required to write a program to read the robot state or control robot I/O signals according to the register address and function code in the Modbus Address table, which can be viewed in "I/O Monitor" →"Modbus" of App. As shown in Figure 3-35.

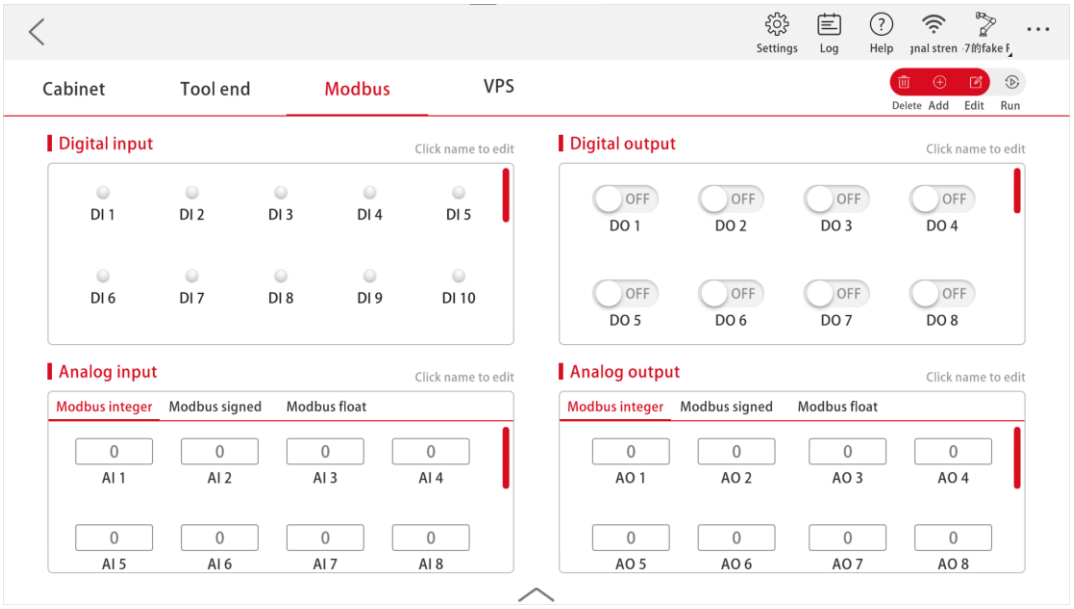


Figure 3-35 IO Panel Modbus Interface

Modbus RTU:

Open the setting interface and set Modbus Slave parameters of the robot. The setting interface on PLC is required to be configured according to Slave parameters of the robot. As shown in Figure 3-36.

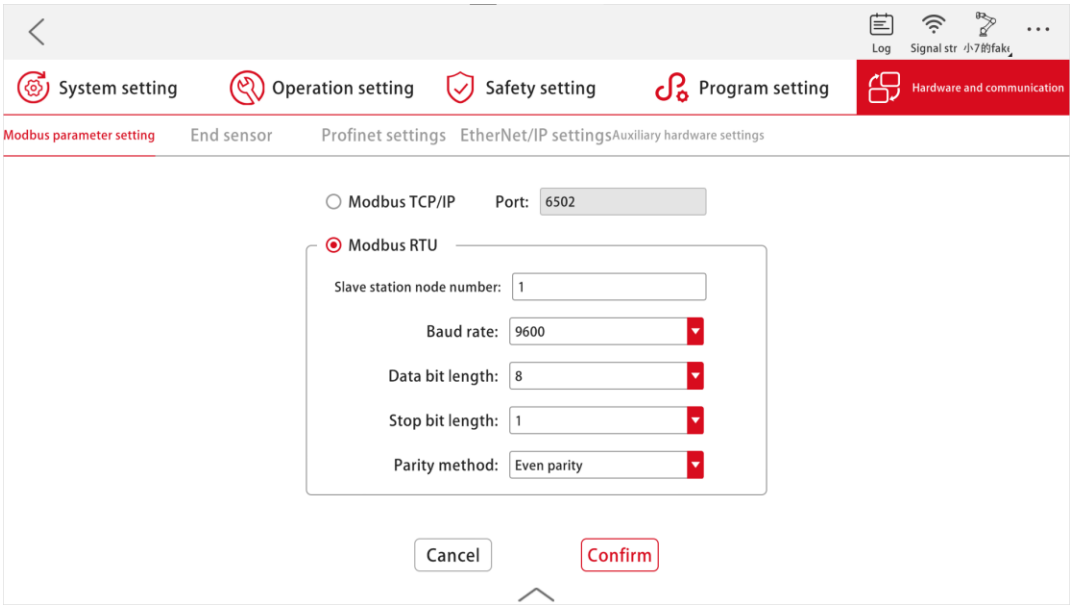


Figure 3-36 Modbus Parameter Interface

After establishing the connection on Client side, it is required to write a program to read robot state or control robot I/O signals according to the register address and function code in the Modbus Address table, which can be viewed in Modbus column of App I/O panel interface. As shown in Figure 3-35. The Modbus Address table is detailed in the Appendix.

3.1.5.2 Terminal Sensors

JAKA Zu series robots can be equipped with end force sensors. Take type 1 sensor as an example, when the robot is connected to the force control sensor, click [Setup] → [Hardware & Communication] → [End Sensor] to enter the configuration interface, as shown in Figure 3-37. In this interface, you can select different sensor types and set the corresponding IP and port numbers according to the actual situation. In addition, you can also set payload parameters through [Payload Settings] or automatically recognize the payload through the auto recognition function, and set the safety force value through [Sensor Limit]. For details and other types of sensors, see the JAKA User Manual of Force Control Product.

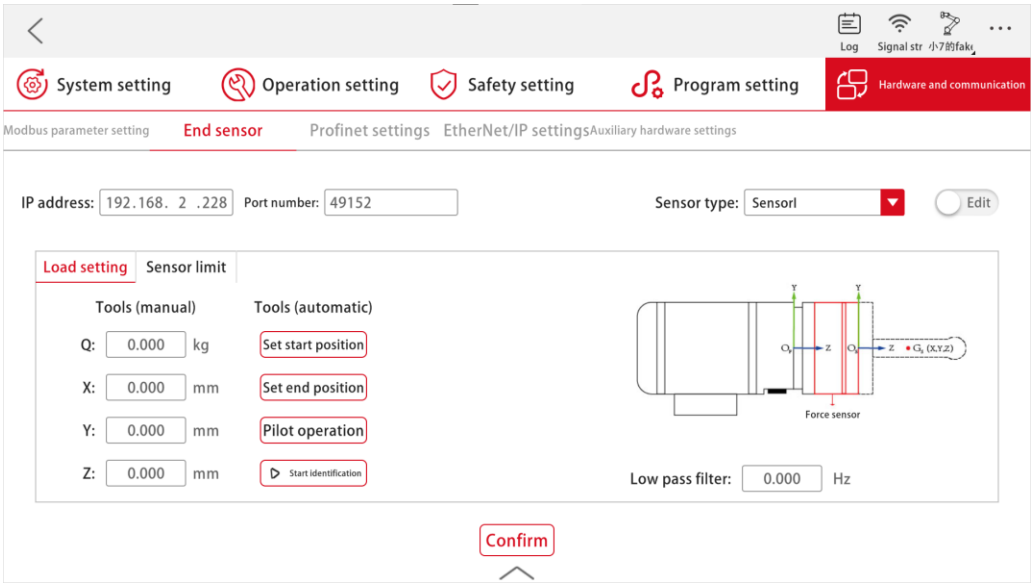


Figure 3-37 Force Control Sensor Setting Interface

3.1.5.3 Profinet Communication Settings

JAKA robot supports Profinet communication protocol and can be used as a Profinet slave station to connect with external devices. Open JAKA Zu software and connect the robot to enter "Settings"-"Hardware & Communication"-"Profinet Settings" interface. This interface displays state information of the connection between the controller and Profinet devices (See Figure 3-38). The enable function turns on or off the Profinet function, which is off by default.

Profinet function can only interact with external PLC when it is enabled, with Profinet IO information displayed in the IO interface. (See section 3.2.4 for details of Profinet IO. See Appendix III for details of Profinet Address table.)

After enabled, Profinet function can only take effect after electrical control cabinet is restarted. The reset button is used to reset the communication configuration of Profinet devices in PLC side. After turning on the enable button, manually restart the electrical control cabinet with the handle to complete Profinet communication settings.

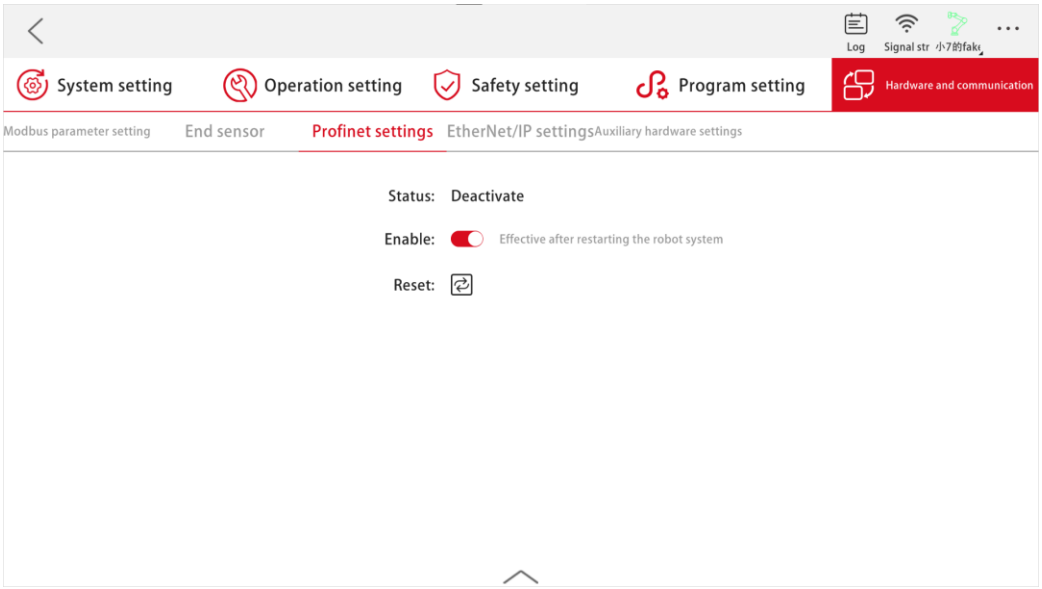


Figure 3-38 Profinet Communication Settings

3.1.5.4 Ethernet/IP Communication Settings

JAKA robot supports Ethernet/IP communication protocols and can be used as Ethernet/IP slave station to connect with external devices. Open JAKA Zu software and connect the robot to enter "Settings"-"Hardware & Communication"-"Ethernet/IP" interface. This interface displays state information of the connection between the controller and Ethernet/IP devices, as shown in Figure 3-39.

The Ethernet/IP enable function can be turned on or off, which is off by default. Ethernet/IP function can only interact with external PLC for Ethernet/IP communication when it is enabled, with Ethernet/IP IO information displayed in the IO screen (see section 3.2.5 for details of Ethernet/IP; see Appendix IV for details of Ethernet/IP Address table.). After turning on the enable button, the IP configuration interface information will appear, and users can configure robot Ethernet/IP communication IP address (either automatic IP address or fixed IP address) according to the actual usage.

After enabled, Ethernet/IP function can only take effect after electrical control cabinet is restarted. After turning on the enable button, configure the robot IP address and manually restart the electrical control cabinet with the handle to complete Ethernet/IP communication settings.

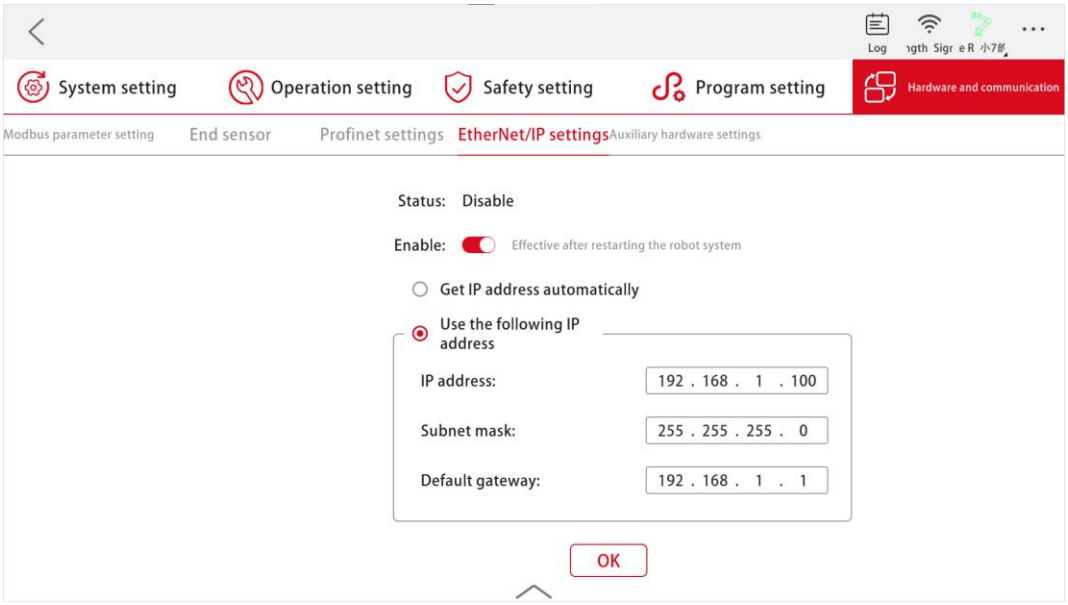


Figure 3-39 Profinet Communication Settings

3.1.5.5 Auxiliary Hardware Settings

There are three buttons on JAKA robot end, which are: End light button, FREE button and POINT button. JAKA Zu software allows users to configure functions of the three buttons at the robot arm end. Open JAKA Zu software and connect the robot to enter "Settings"-"Hardware & Communication"-"Auxiliary Hardware Settings" interface. Users can click the corresponding drop-down button of "Button Option" selection box to switch the function selection.

Functions of end light button option: Function disable, program pause or start, function to enter drag mode.

Functions of FREE button option: Function disable, function to enter drag mode.

Functions of POINT button option: Function disable, function to record the current point.

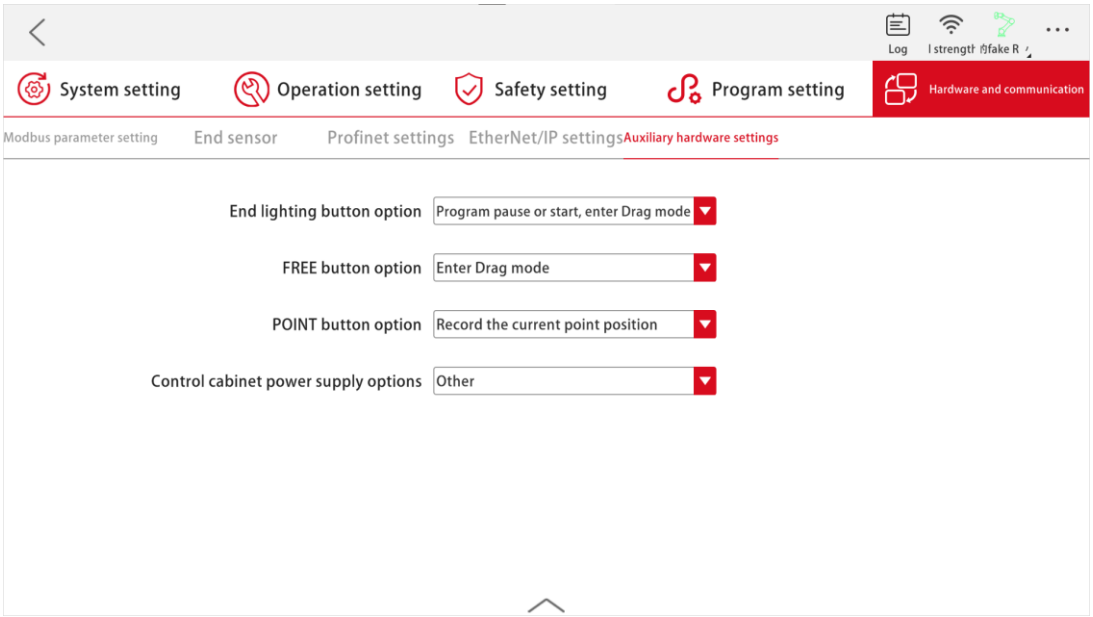


Figure 3-40 Auxiliary Hardware Setting Interface

3.1.5.6 End IO

Note: This function is only available for the V3 version of end TIO robot.

There is a small IO interface (TIO) at the robot end (tool end), which supports 2 digital inputs, 2 digital outputs and 2 analog inputs. The 2 digital outputs can be multiplexed as high-speed RS485 channels, and the 2 analog inputs can be multiplexed as low-speed RS485 channels. Besides, it supports configurable voltage outputs (12V/24V/0V) to power external extended devices. And it can be configured in "App Settings" → "Hardware & Communication" → "End IO".

① Voltage Output Configuration

The tool-end TIO is provided with end power output function, which allows the user to choose whether to output voltage or not, and the output voltage magnitude of enable voltage can be selected as 12V or 24V. Configure the voltage output enable and output magnitude in [Settings] -> [Hardware & Communication] -> [Voltage Output] interface, as shown in Figure 3-41 below.

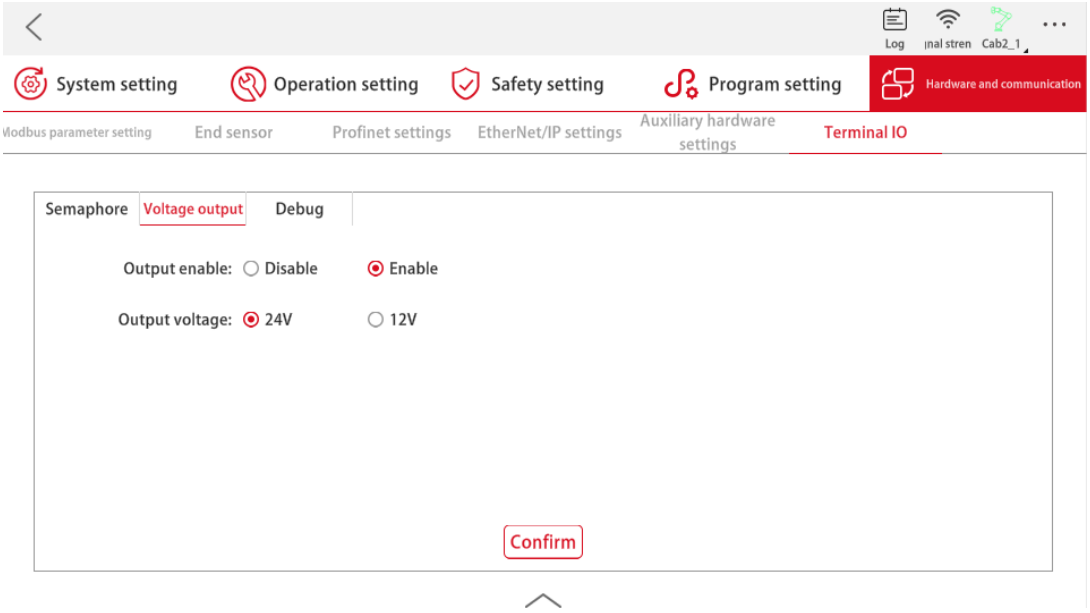


Figure 3-41 TIO Power Supply Output Function Configuration

② RS485

RS485 Configuration:

2 RS485 channels can be configured only after the corresponding pins are multiplexed as RS485 channels. Take RS485 channel 1 as an example, first multiplex the DO pins as RS485 channel 1. After completing the setting, the configuration option button for RS485 channel will appear. Click [RS485 Configuration] and RS485 channel configuration interface will pop up, including communication parameters and function mode configuration, as shown in Figure 3-42, 3-43 and 3-44.

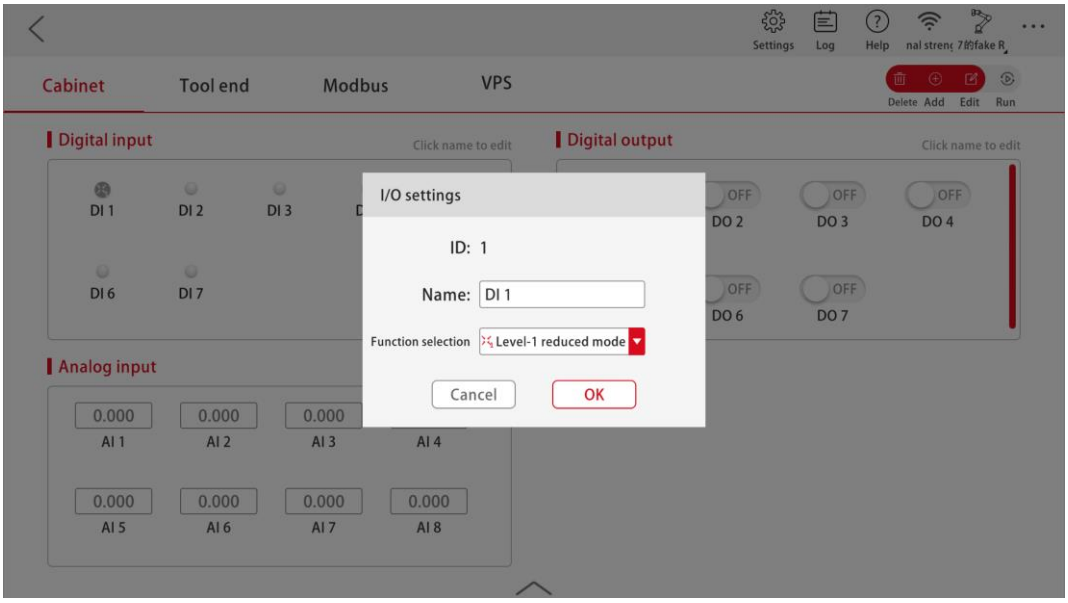


Figure 3-42 Digital Output Multiplexed as RS485 Channel 1

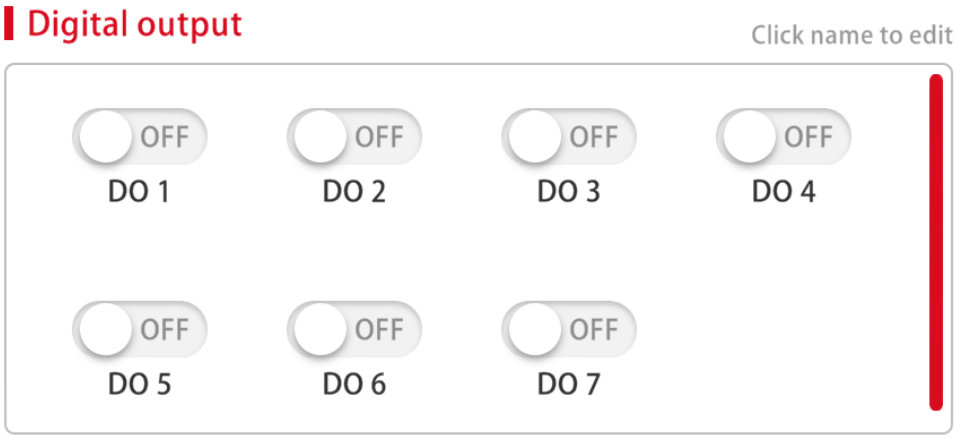


Figure 3-43 RS485 Configuration Options

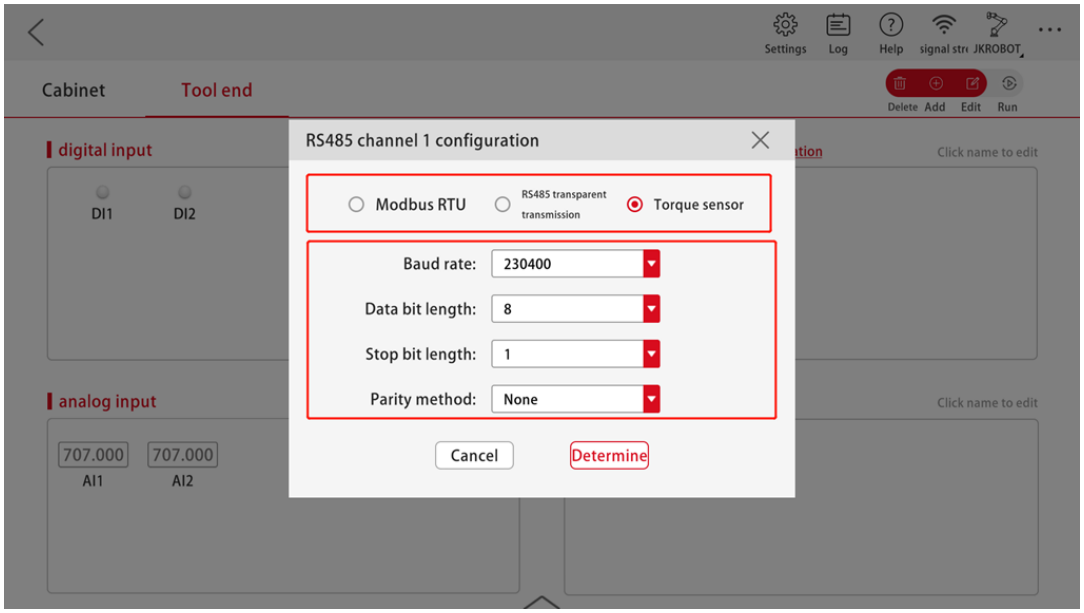


Figure 3-44 RS485 Channel Configuration Interface

RS485 Channel Configuration:

When RS485 channel is used, it is required to set its mode with three options: passthrough mode, Modbus RTU mode, and torque sensor mode, as shown in Figure 3-44:

- Passthrough mode does not support any devices, and subsequent versions are upgraded for use;
- Modbus RTU is mainly used for supporting devices such as external grippers;
- The torque sensor mode is mainly used to connect the specified model of torque sensor, which is TIO end torque sensor for specified sensor VI.

Note: For 2 RS485 channels currently available, it is recommended that torque sensor use RS485 channel 1 in preference and the gripper use either RS485 channel. And only one of the 2 RS485 channels can be used by torque sensor.

RS485 Channel Communication Parameters Configuration:

Regardless of what mode the RS485 channel is, it is required to configure the serial communication parameters of the RS485 channel, including baud rate (up to 2,250,000), data bits (supports 8/9), stop bits (supports 1/2) and parity bits (supports odd parity/even parity/no parity). When the channel mode is set as Modbus RTU, the Modbus slave station node ID is required to be specified additionally.

Note: Communication parameters misconfiguration will result in TIO+ not being able to communicate properly with external devices.

③ Torque Sensor Usage

TIO currently supports VI model torque sensor. The configuration steps to be performed when using the torque sensor are shown as follows:

- Install the torque sensor by connecting the connection terminal of the torque sensor to TIO end connector, with XY direction of the sensor in line with that of the end flange;
- TIO power output configuration: Set the voltage to 24V and enable it, and the sensor indicator is constantly on;
- 2 RS485 channels can be configured only after the corresponding pins multiplexed as RS485 channels;

  Take RS485 channel 1 as an example, first multiplex the DO pin as RS48 channel 1. After completing the settings, configuration option button for RS485 channel will appear.

  Click [RS485 configuration] and RS485 channel configuration interface will pop-up, including communication parameters and function mode configuration.

  Multiplex the corresponding IO pins as RS485 function. Only one of the 2 RS485 channels can be used by torque sensor.

After completing the above steps, TIO can communicate with the external torque sensor. However, the following configuration is still required to actually use the end torque sensor extended by TIO in the robot controller.

- In torque sensor configuration interface, select sensor type VI and click [OK] to ensure that

the sensor brand is set successfully. Click [run] to achieve normal sensor communication;

- Enable the torque sensor.

If the torque sensor is still not enabled properly, check the pin function multiplexing configuration and the RS485 communication parameter configuration. After normally enabled, the torque sensor actually enters working condition. The state value of the torque sensor can be monitored in App interface, and the force control function can be used in the operation program. As shown in Figure 3-45.

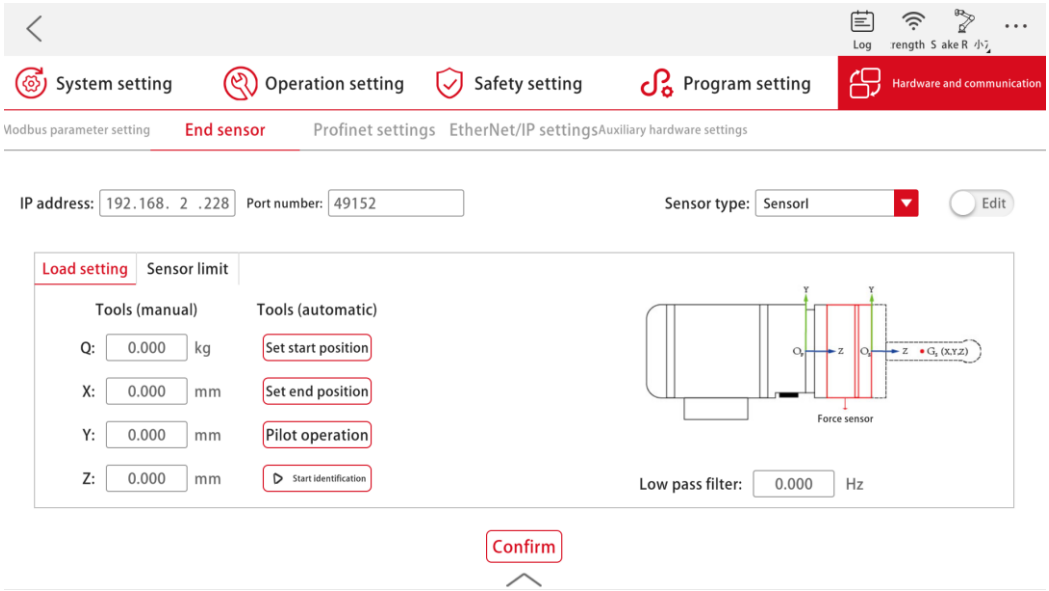


Figure 3-45 Channel Selection of Torque Sensor

When using a TIO channel torque sensor, the communication parameters is required to be set in the RS485 channel configuration interface. The interface shown in Figure 3-44 only displays the communication parameters of RS485 channel in use. If TIO RS485 channel is not properly configured based on the torque sensor and TIO channel is selected, the torque sensor will not work properly. The use of TIO channel-based torque sensor, including switch on/off and payload settings of the torque sensor, is the same as that of Ethernet communication.

If TIO channel is selected for the current torque sensor and is connected for communication, switching the RS485 channel configuration or pin configuration of the corresponding TIO will lead to an error, and the torque sensor must be turned off.

④ Gripper Usage

Currently, grippers with Modbus RTU communication are supported. Please check the definition of terminal connectors before connecting grippers to TIO terminals to ensure proper wire sequence connection. For specific communication methods, please refer to section ②RS485 as above.

⑤ Signal Variable

JAKA controller provides signal variables mechanism for users to check. Users need to define all the state variables to be checked in advance, and subsequently obtain the values of state variables by refreshing and checking.

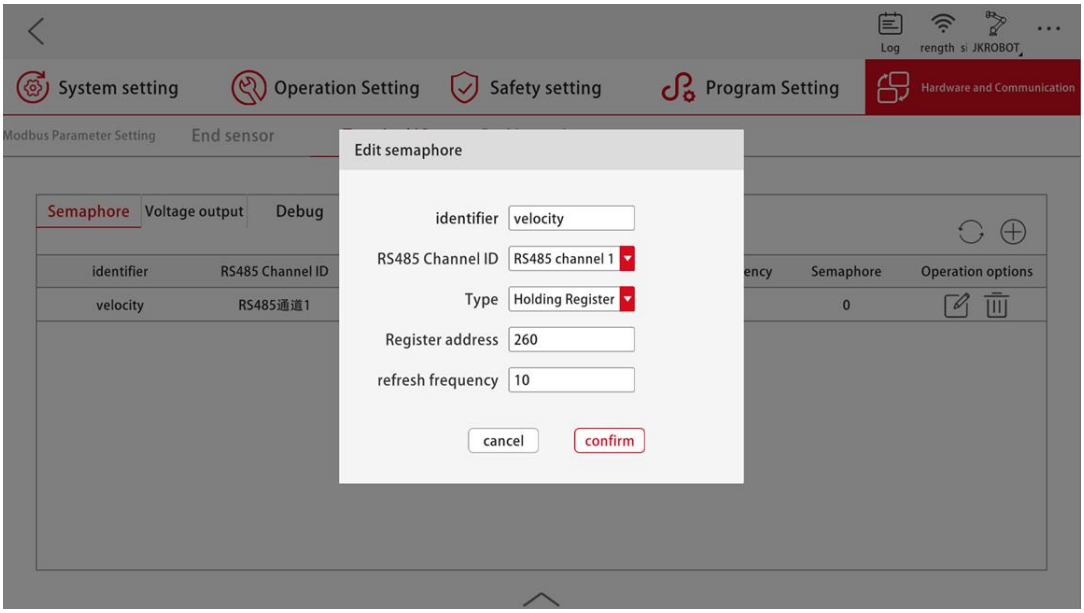


Figure 3-46 Definition of Signal Variable

The defining interface of signal variable is shown in Figure 3-46, and the meaning of each signal variable is shown as follows:

Signal variable identification: The unique identifier of signal variable (Unicode and special characters are not supported) for subsequently refreshing, accessing and deleting;

RS485 channel: It specifies TIO RS485 channel from which the signal variable is sourced (either RS485-1 or RS485-2).

Signal variable type: The type of signal variable data. This parameter corresponds to Modbus function code: 01 for coil register; 02 for discrete input; 03 for input register; 04 for holding register. Others are not supported currently.

Signal variable address: Refers to Modbus register address corresponding to the signal variable. In combination with RS485 channel configuration and signal variable types, this address can access specified registers of Modbus RTU slave station.

Note: The signal variable is defined when TIO+ related IO pins have been multiplexed as RS485 channels and set to Modbus RTU mode, and mode change or pin multiplexing will cause signal variable configuration loss.

Refreshing and checking signal variables:

Once the signal variable is defined, it can be debugged or monitored in the debug interface, or used directly in the operation program. Both methods provide interfaces for refreshing and checking signal variables. Refreshing will actually trigger the data interaction between the controller and TIO+ external gripper. Since the interaction process between the controller and TIO+ external device is asynchronous with refreshing operation instructions, it is necessary to wait for a certain interval (recommended value is 100ms) after refreshing to ensure that the refreshed value is obtained. In addition, refreshing can specify the refreshing frequency. When the frequency is 0, it will be treated as a one-time refresh, and when the frequency is greater than 0, it will be combined with the communication bandwidth at the bottom to meet the requirements to the maximum extent.

You can manually refresh the value of signal variable by clicking "Refresh" icon in the interface, as shown in Figure 3-47.

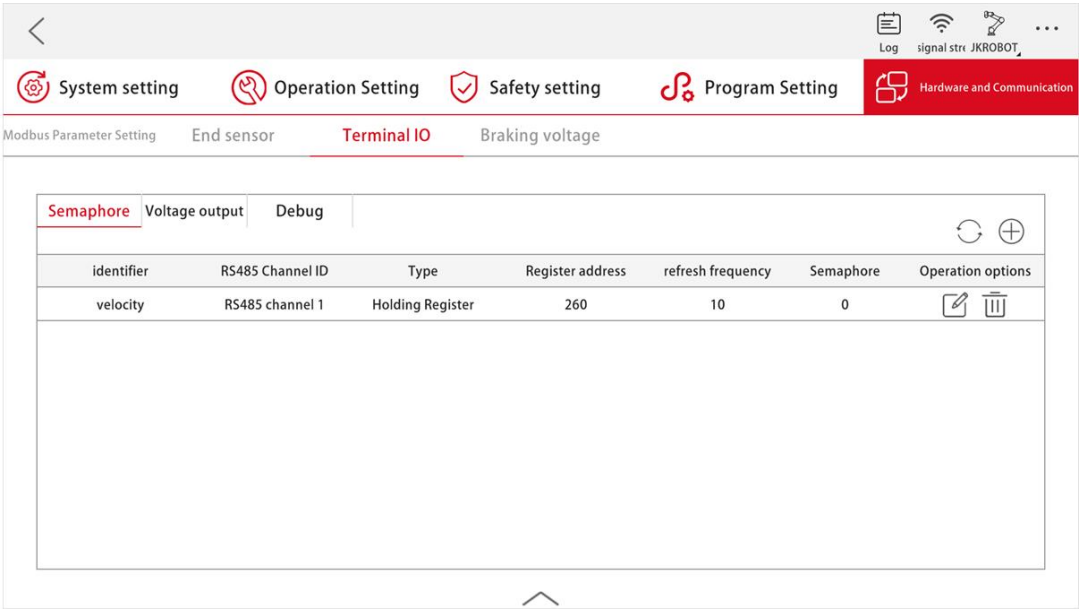


Figure 3-47 Signal Variable Monitoring Interface

Deleting signal variables:

Signal variables can be deleted manually in the debug interface. Click the "Delete" icon on the right of the corresponding signal variable in Signal Variable Monitoring interface, and a confirmation interface will pop up. The corresponding signal variable will be deleted after confirmation, as shown in Figure 3-48.

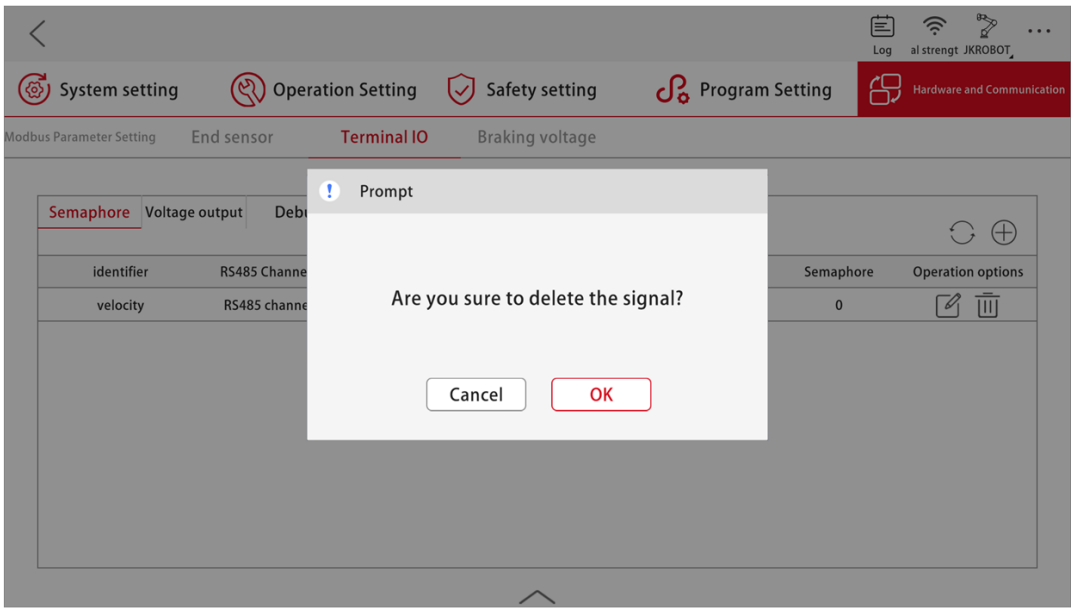


Figure 3-48 Deleting Signal Variables

⑥ Sending Immediate Instructions

Immediate instructions mainly refer to controllers' immediate control instructions to the external gripper of TIO, including gripper position control, speed control and force control. (**Note:** When users enter immediate instruction data, it is only required to write the instruction data, and CRC checksum value will be added automatically by the internal control system.)

Immediate instructions can be sent in TIO Device Debug interface, as shown in Figure 3-48 below. Users can enter hexadecimal data instructions and click "Send" to send it.

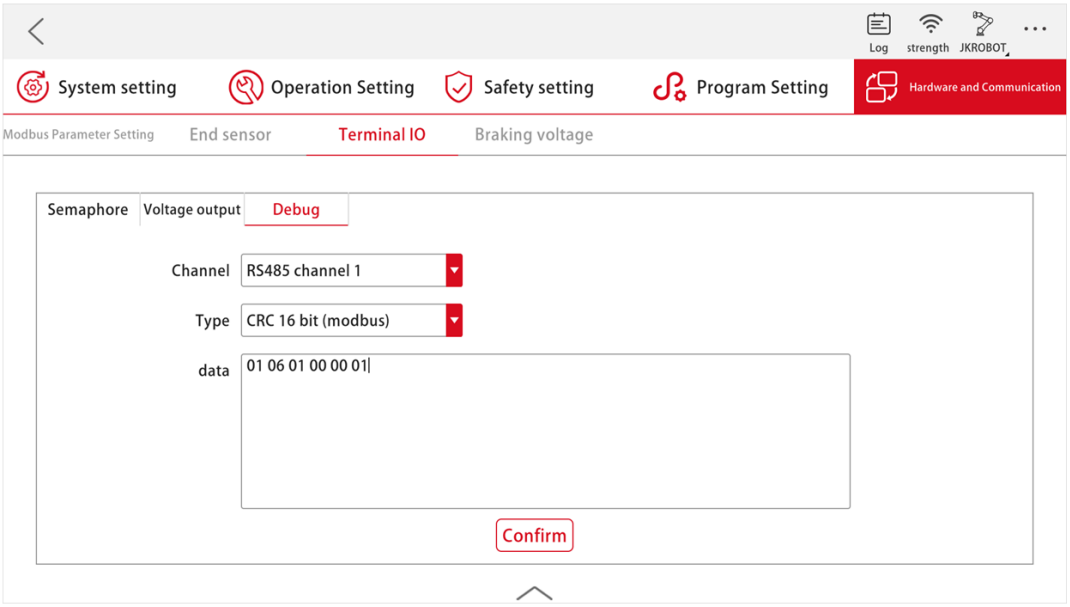


Figure 3-49 Instruction Control of External Devices

⑦ TIO+ Support in Operation Programs

Provides instructions in operation program for refreshing and checking signal variables. The definition, modification and deletion of signal variables are required to be added manually in the debug interface. It also provides immediate sending instructions for immediate control of external devices. UI instructions currently provided are shown in the figure.

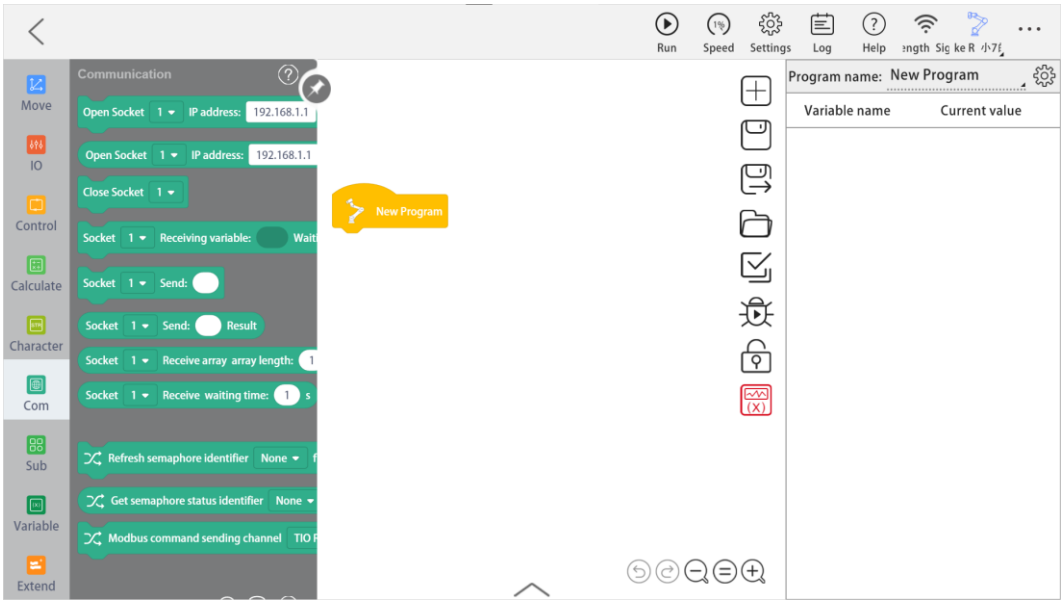


Figure 3-50 TIO+ Support in Operation Programs

3.1.5.7 Braking Voltage

Note: This function is only available for MiniCab version of electrical control cabinet.

MiniCab has an integrated voltage brake circuit to release the electrodynamic force generated by the

robot during deceleration and braking. When using external power supply, it needs to be set to avoid overvoltage protection or controller damage. When setting the braking voltage, it is required to power off the robot arm before operation. The setting path is as follows:

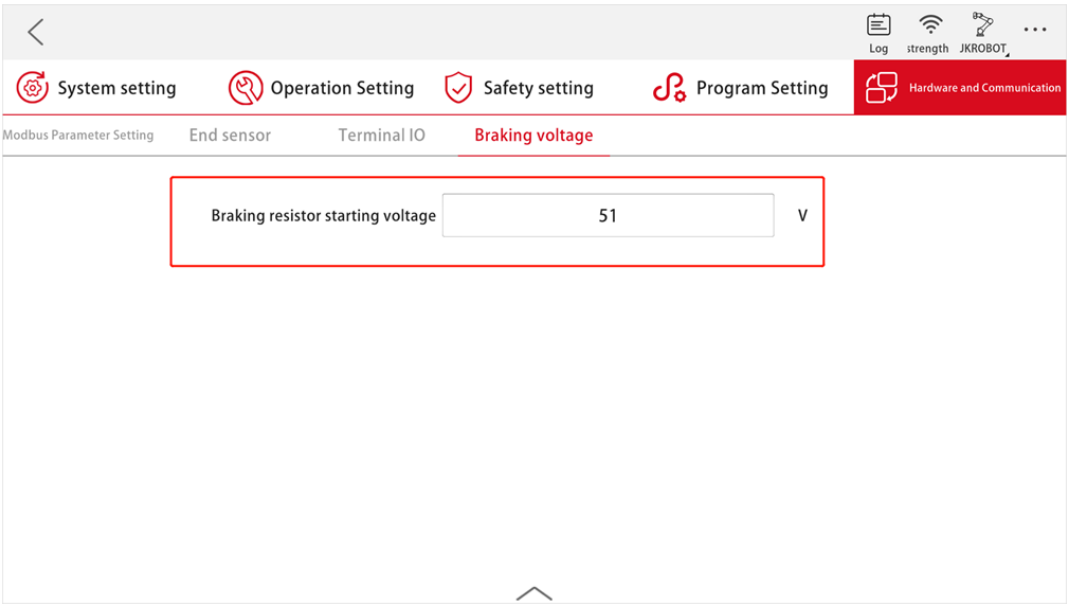


Figure 3-51 Braking Voltage Setting Path

The relationship between the voltage setting value V_{Brake} and the input voltage V_{IN} is $V_{Brake} \geqslant (V_{IN}+3)V$. According to common input voltage type, the recommended setting value and power supply type correspondence table are as follows:

Table 3-1 Correspondence Table of Recommended Setting Values between Input Voltage Type and Power Supply Type

Power Supply Type	Voltage (V_{IN}[1])	Brake Resistor Starting Voltage (V_{Brake})
48V modular power supply	48V	51V
48V lithium battery	54.6V	58V
24V modular power supply [2]	24V	27V

Note: 1. Change the voltage value to typical values;

2. 24V modular power supply is for Minicobot;

3. When $V_{Brake} < (V_{IN}+1)V$, the internal logic will be powered on for protection and APP will prompt "Robot arm voltage or voltage configuration is abnormal".

3.2 How to Configure I/O Information

The electrical I/O of the robot's electrical control cabinet can be viewed and set via JAKA Zu software I/O monitoring interface (**Note:** Power off when editing settings for the I/O module). I/O module panel is

divided into electrical control cabinet I/O, tool-side I/O, Modbus I/O, Profinet I/O and Ethernet/IP I/O. Click the "Up" arrow below to call the menu bar, and click "I/O Monitoring" to open I/O module interface (for the homepage, it is not required to click the "Up" arrow).

Note:

V2.1 version of electrical control cabinet is provided with 16 inputs and 16 outputs. The I/O panel interface shows 16 DI and 16 DO respectively when connecting V2.1 electrical control cabinet via APP. (V2.1 version of the electrical control cabinet is PNP type)

Electrical control cabinet of Minicab is provided with 7 NPN-type input and output multiplexed interfaces, and only one interface form can be selected each time.

3.2.1 Electrical Control Cabinet IO

3.2.1.1 V1.0 Version of Electrical Control Cabinet

V1.0 version of electrical control cabinet is provided with 8 physical inputs, 8 physical outputs and 8 physical analog inputs. When V1.0 version of electrical control cabinet is connected via App, the I/O panel interface will display the physical signals in the electrical control cabinet. (V1.0 version of electrical control cabinet is NPN type). As shown in Figure 3-52.

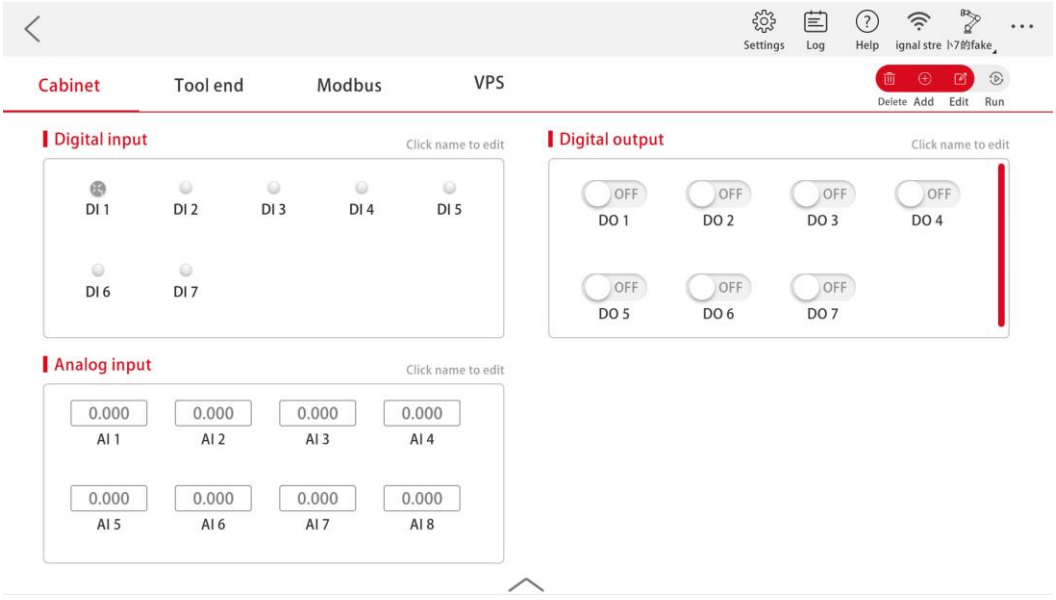


Figure 3-52 Schematic Diagram of Electrical Control Cabinet I/O

Digital inputs state in electrical control cabinet can be monitored by digital input interface. Click the digital input signal to edit the corresponding DI name and select DI function settings. Select the function to be set for this DI in the function selection drop-down box and click OK. When this DI signal is triggered, the corresponding function will be enabled, as shown in Figure 3-53.

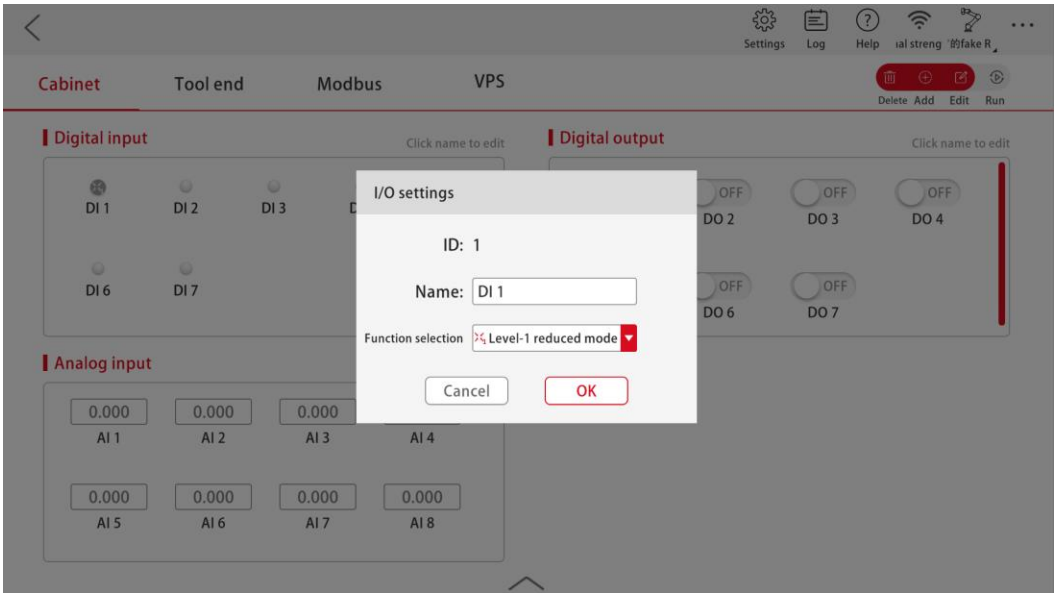


Figure 3-53 Schematic Diagram of DI Settings

Digital outputs state in electrical control cabinet can be monitored by digital output interface. Click the digital output signal to edit the corresponding DO name and select DO function settings. Set predefined state variables of the system bound to DO in the function selection drop-down box and click OK. This DO signal can reflect the state of the bound system state variables in real time, as shown in Figure 3-54.

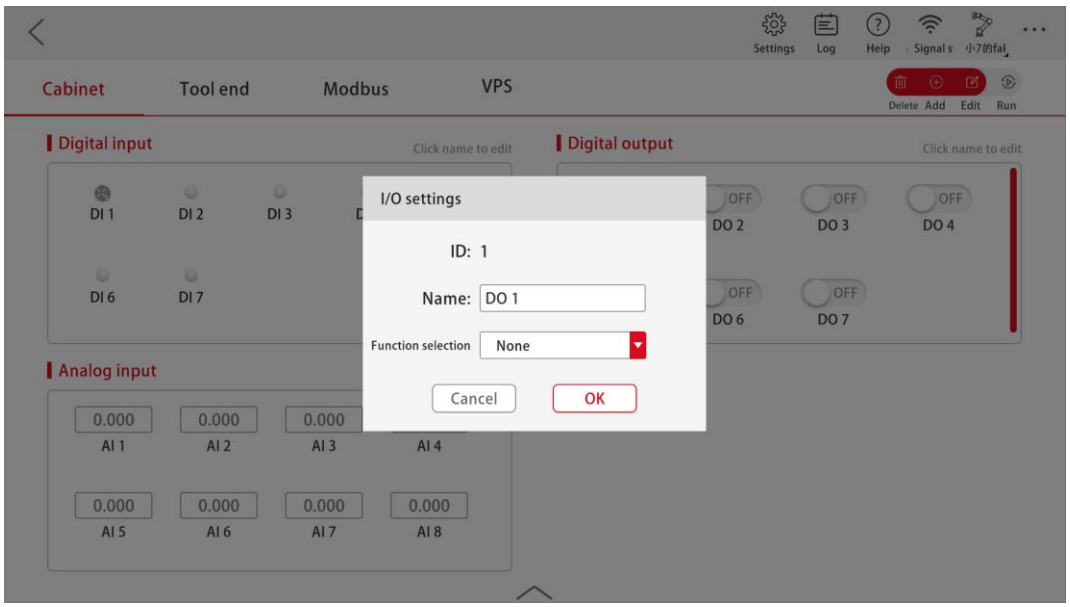


Figure 3-54 Schematic Diagram of DO Settings

Analog signal variable can be monitored by analog input interface. You can set the name of analog input AI in I/O interface. Click the analog input signal and enter a self-defined name in the "Name" text box. Click OK to take effect. (As shown in Figure 3-55)

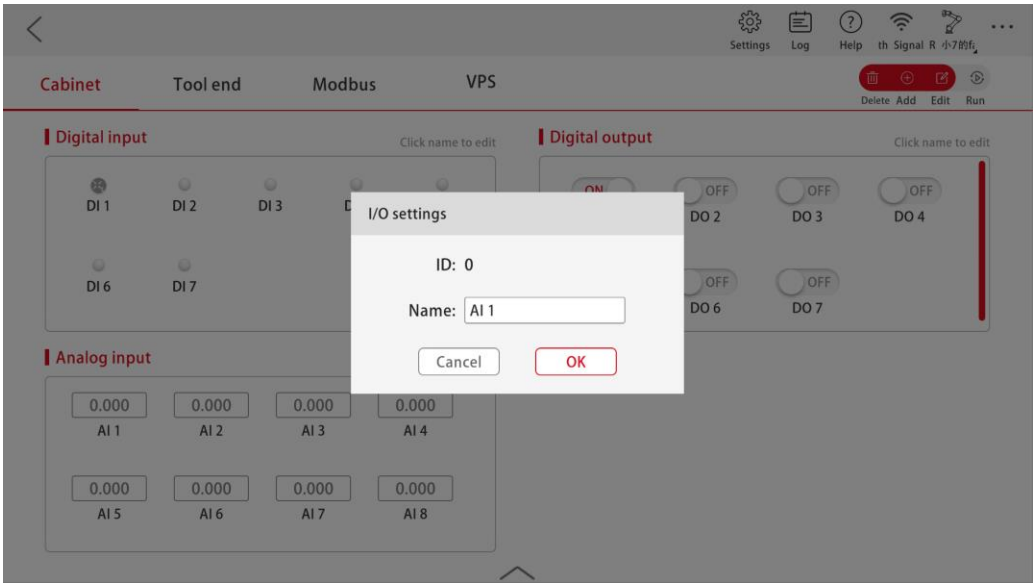


Figure 3-55 AI Setting Diagram

3.2.1.2 V2.1 Version of Electrical Control Cabinet

V2.1 version of electrical control cabinet is provided with 16 physical inputs, 16 physical outputs and 2 physical analog signals. When V2.1 version of electrical control cabinet is connected via App, physical signals in the electrical control cabinet will be displayed in I/O panel interface. (V2.1 version of electrical control cabinet is PNP type). As shown in Figure 3-56.

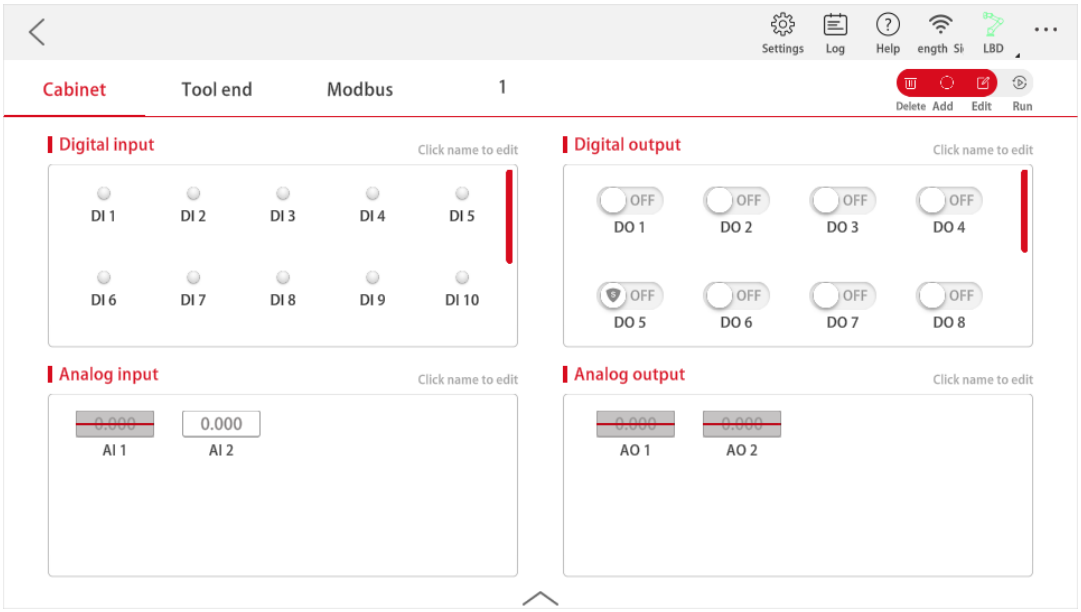


Figure 3-56 Electrical Control Cabinet I/O Diagram

Digital inputs state in electrical control cabinet can be monitored by digital input interface. Click the digital input signal to edit the corresponding DI name and select DI function setting. Select the function to be set for this DI in the function selection drop-down box and click OK. When this DI signal is triggered, the corresponding function will be enabled. As shown in Figure 3-57.

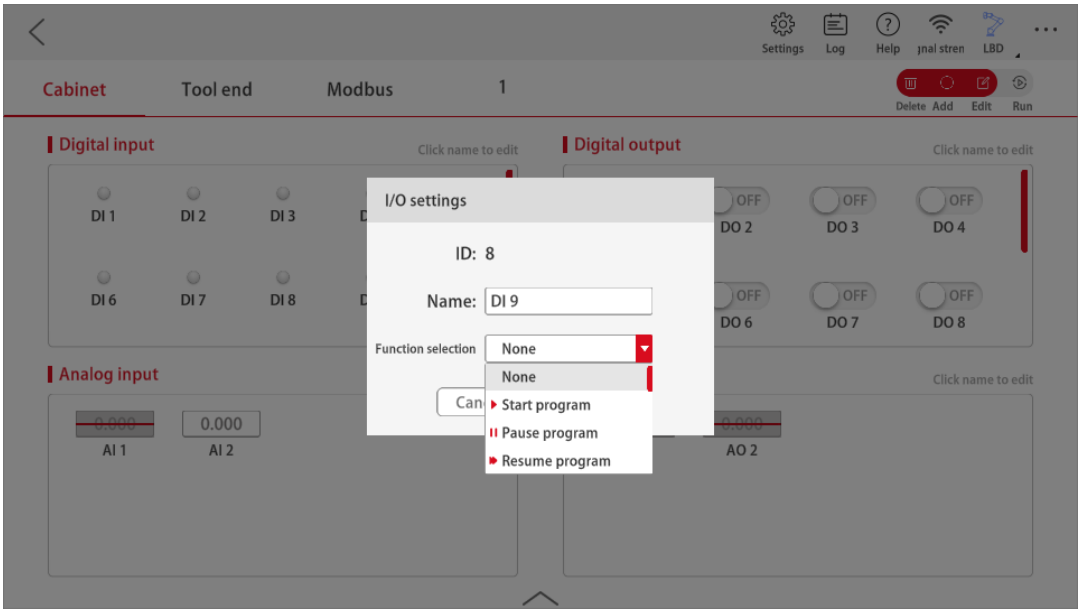


Figure 3-57 Schematic Diagram of DI Settings

Digital outputs state in electrical control cabinet can be monitored by digital output interface. Click the digital output signal to edit the corresponding DO name and select DO function setting. Set predefined state variables of the system bound to DO in the function selection drop-down box and click OK. This DO signal can reflect the state of bound system state variables in real time. As shown in Figure 3-58.

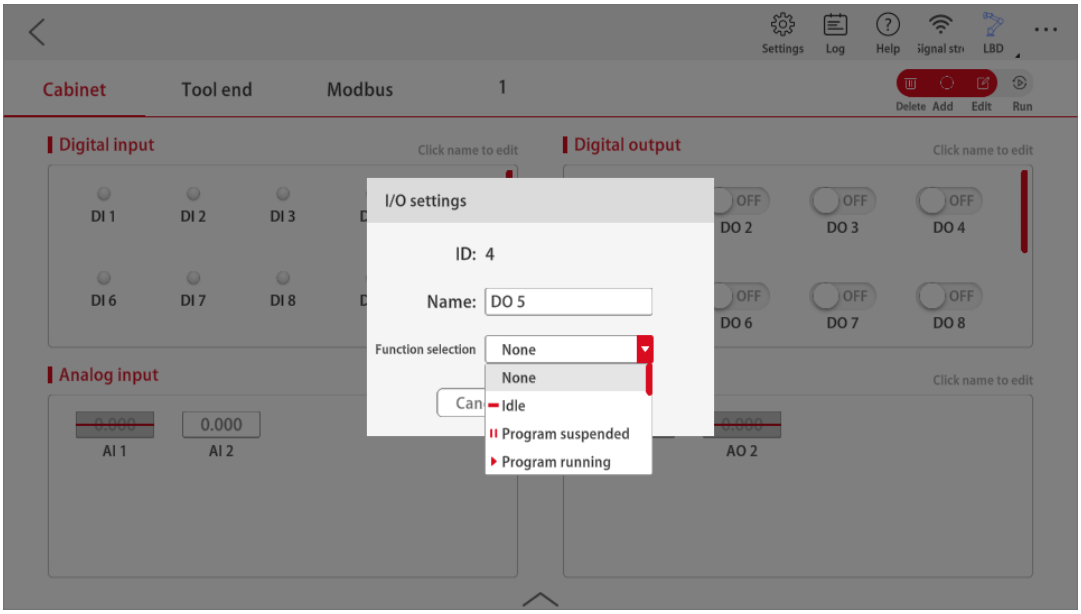


Figure 3-58 Schematic Diagram of DO Settings

Analog signal variables can be monitored by analog input/output interface. You can set the function of analog input AI in I/O interface. Click the analog input signal to select the function of this signal. Click "Function Selection" drop-down window to select the corresponding function, and click OK to take effect. Currently AI signal supports following functions: voltage input, voltage output, current input and current output. (As shown in Figure 3-59); You can set the name and value of the analog signal by clicking it, and

analog port can be configured in the same interface (As shown in Figure 3-60).

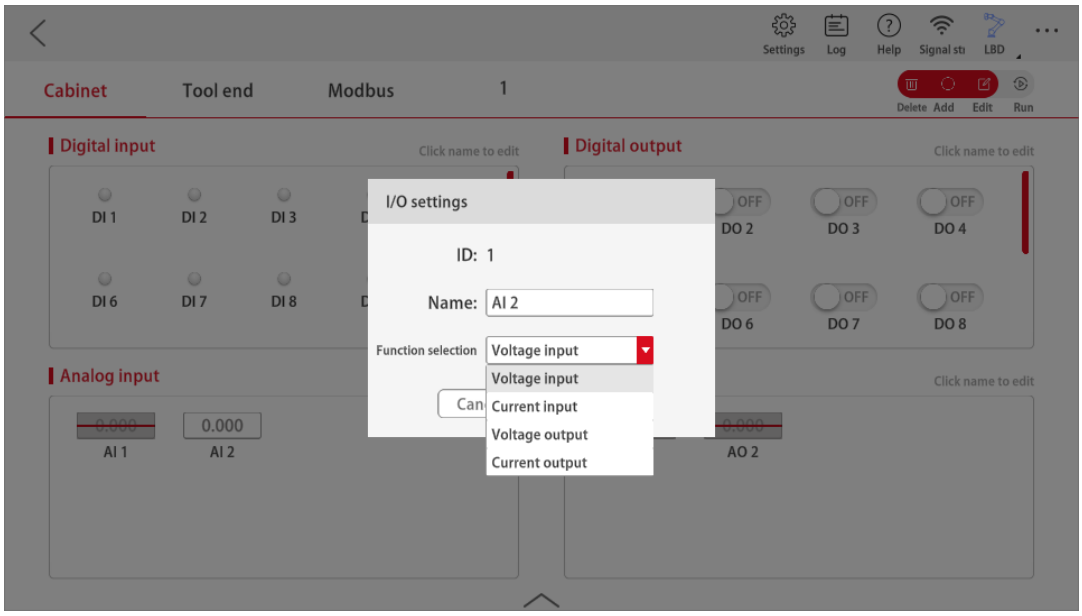


Figure 3-60 AI Setting Diagram

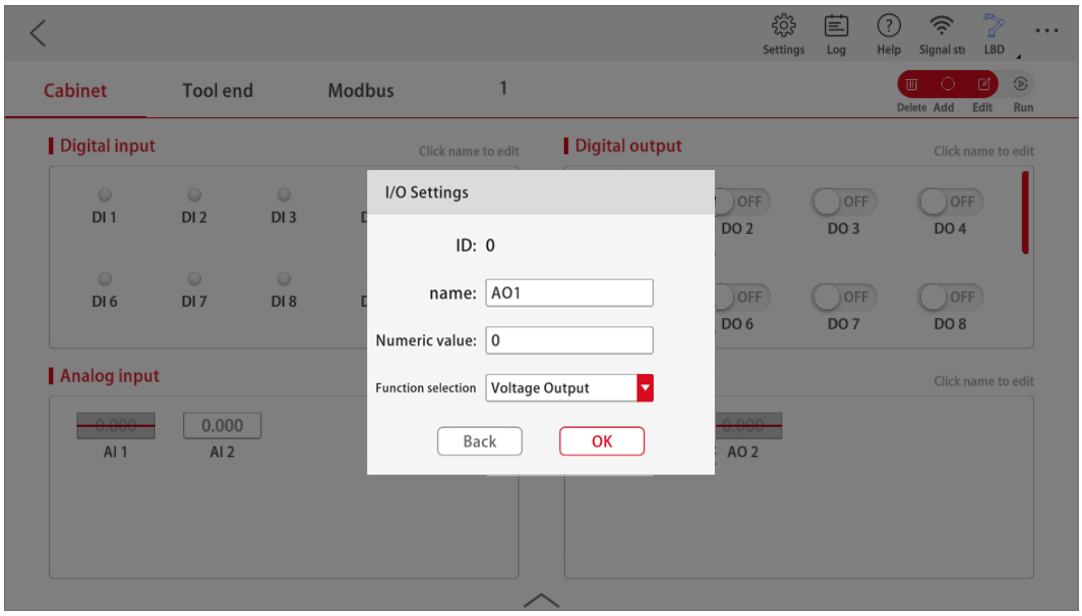


Figure 3-60 AO Setting Diagram

3.2.1.3 Electrical Control Cabinet of MiniCab

Electrical control cabinet of MiniCab is provided with 7 physical digital signals. When it is connected via App, physical signals in the electrical control cabinet will be displayed in I/O panel interface. (MiniCab is NPN type). As shown in Figure 3-61.

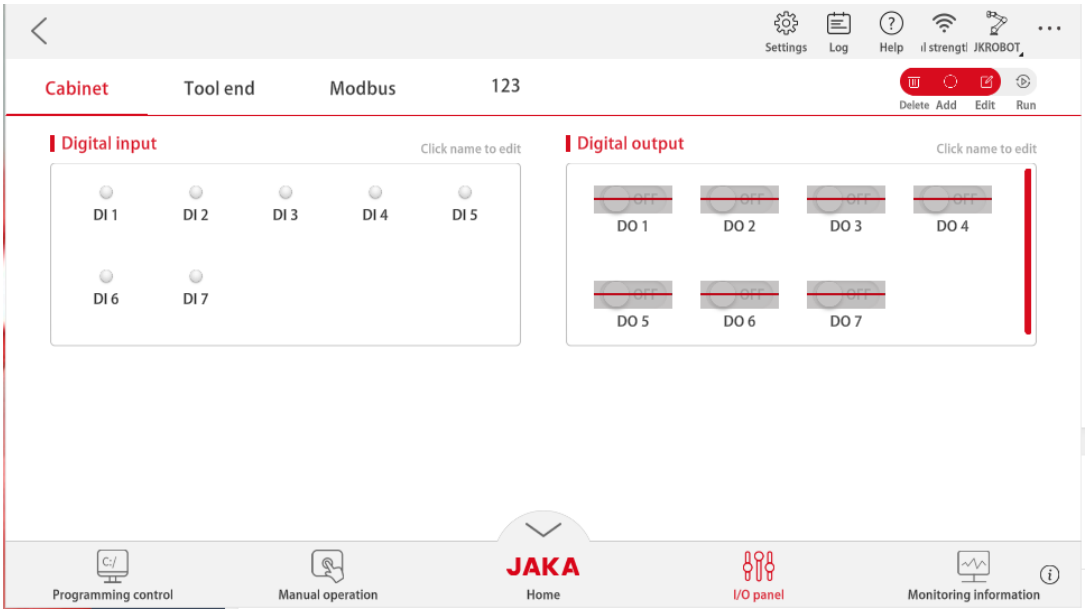


Figure 3-61 Electrical Control Cabinet I/O Diagram

Digital inputs state in electrical control cabinet can be monitored by digital input interface. Click the digital input signal to edit the name, select the type and set functions for DI. In the type selection, you can define the current interface as a digital input or a digital output. Select the function to be set for this DI in the function selection drop-down box and click OK. The function will be enabled when this DI signal is triggered. As shown in Figure 3-62.

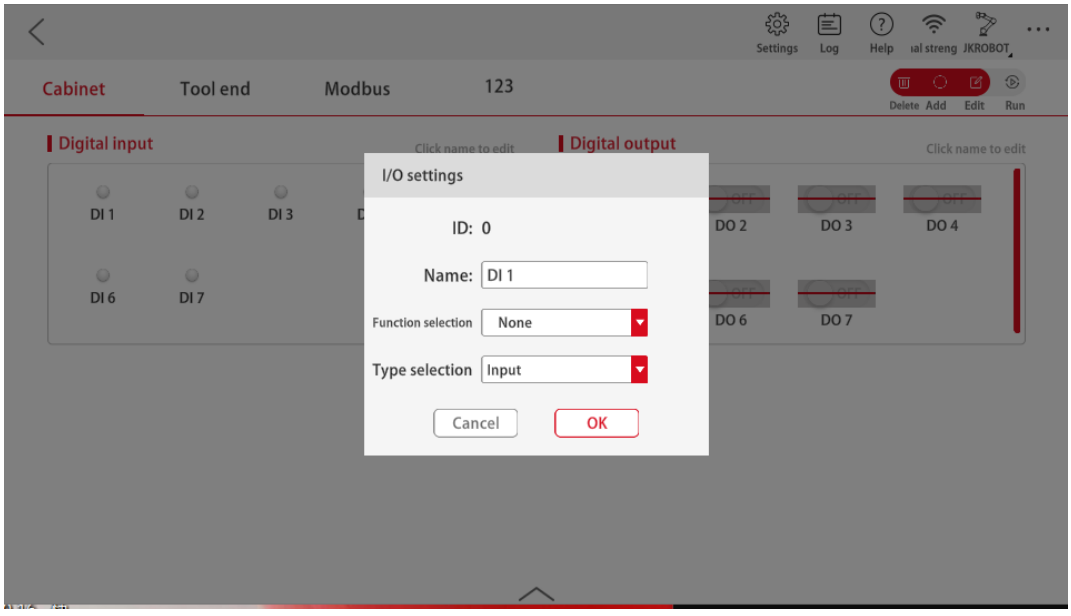


Figure 3-62 Schematic Diagram of DI Settings

Digital outputs state in electrical control cabinet can be monitored by digital output interface. Click the digital output signal to edit the name, select the type and set functions for DO. In the type selection, you can define the current interface as a digital input or a digital output. Set predefined state variables of the system bound to DO in the "Function Selection" drop-down box and click OK. This DO signal can reflect

the state of the bound system state variable in real time. As shown in Figure 3-63.

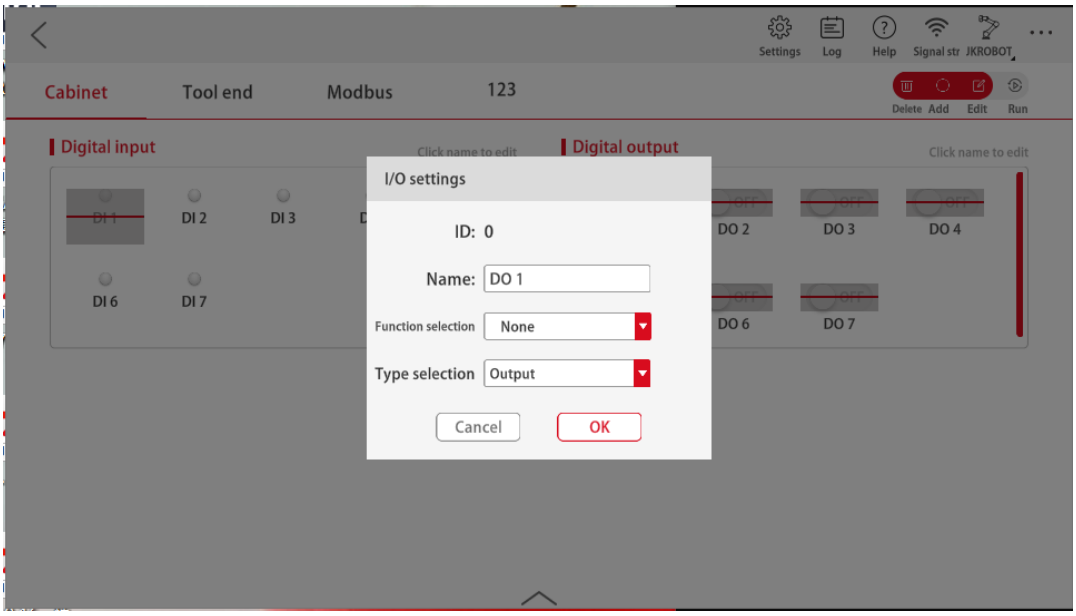


Figure 3-63 Schematic Diagram of DO Settings

3.2.2 Tool-side IO

The robot arm end tool TIO is available in two versions: V2 and V3. When using V3 version, the end IO settings will appear in "Hardware & Communication" interface in the setting interface for configuration. When using different versions of the end tool TIO, there are different operations of the tool side IO in IO panel, which are shown as follows:

3.2.2.1 V2 Version of Tool Side IO

V2 version of robot arm end (tool side) is provided with 2 digital inputs, 2 digital outputs and 1 analog voltage input. As shown in Figure 3-64.

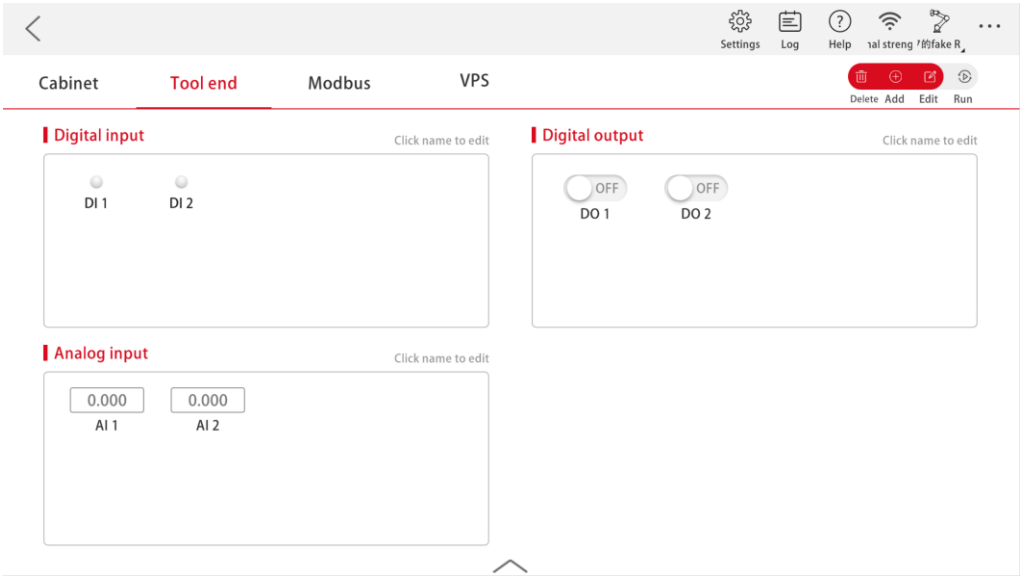


Figure 3-64 Schematic Diagram of Tool Side I/O

The state of digital inputs in the tool side can be monitored in digital input interface. Click the digital input signal to edit the corresponding DI name and select DI function settings. Select the function to be set for this DI in the function selection drop-down box and click OK. When this DI signal is triggered, the corresponding function will be enabled. As shown in Figure 3-65.

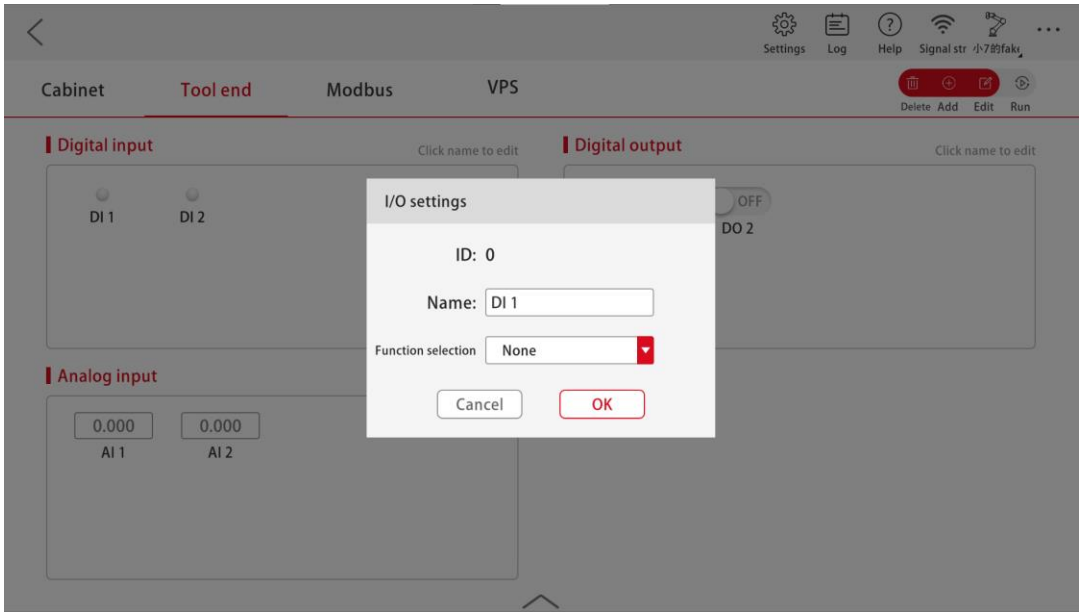


Figure 3-65 Schematic Diagram of DI Settings

The state of digital outputs in the tool side can be monitored in digital output interface. Click the digital output signal to edit the corresponding DO name and select DO function settings. Set predefined state variables of the system bound to DO in the function selection drop-down box and click OK. This DO signal can reflect the state of bound system state variables in real time. As shown in Figure 3-66.

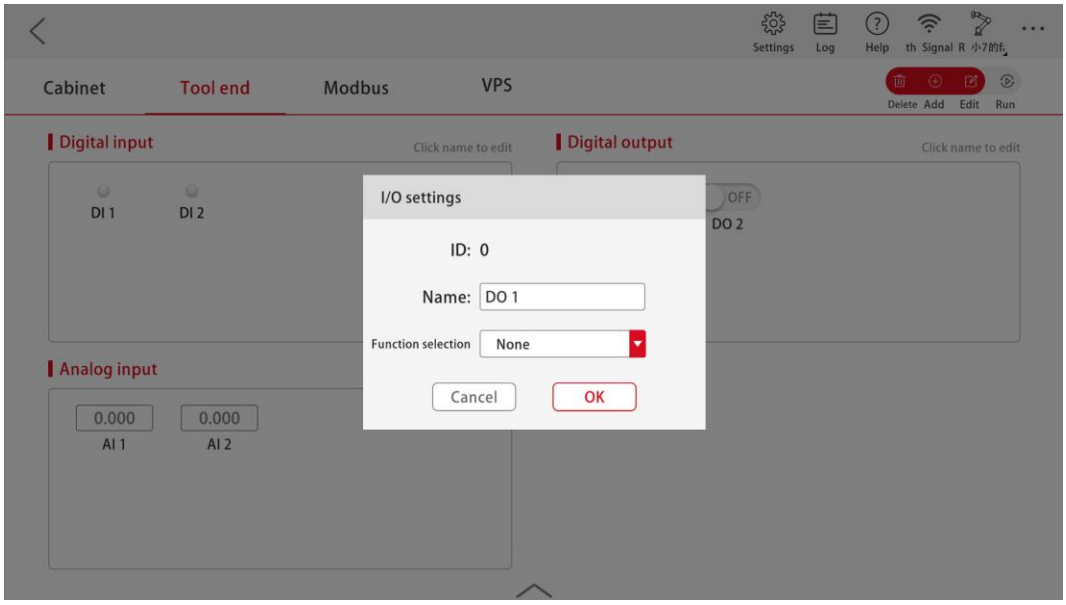


Figure 3-66 Schematic Diagram of DO Settings

Analog voltage signals can be monitored by analog input interface. The name of the analog input

signal can be changed by clicking it (As shown in Figure 3-67);

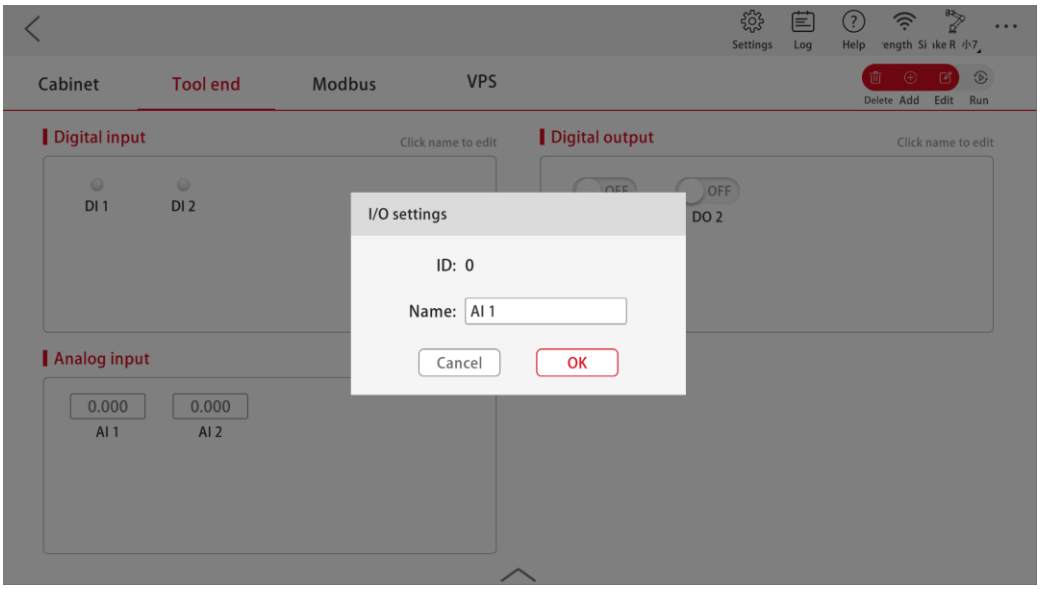


Figure 3-67 Schematic Diagram of AI Settings

3.2.2.2 V3 Version of Tool Side I/O

V3 version of robot arm end (tool side) supports 2 digital inputs, 2 digital outputs and 2 analog inputs. The 2 digital outputs can be multiplexed as high-speed RS485 channels, and the 2 analog inputs can be multiplexed as low-speed RS485 channels. Besides, it supports configurable voltage outputs (12V/24V/0V) to power external extended devices.

Digital Input Configuration

2 DIs in the TIO function can be set as different input modes respectively, i.e. NPN type input or PNP type input. The two DIs are configured as NPN inputs by default once TIO is powered up. Clicking the pin name in the [Main Menu] -> [IO Monitor] -> [Tool Side] -> [Digital Input] interface, and IO setting interface will pop up, as shown in Figure 3-68 below, where you can set the input type of the corresponding pin in "Mode Settings" drop-down box.

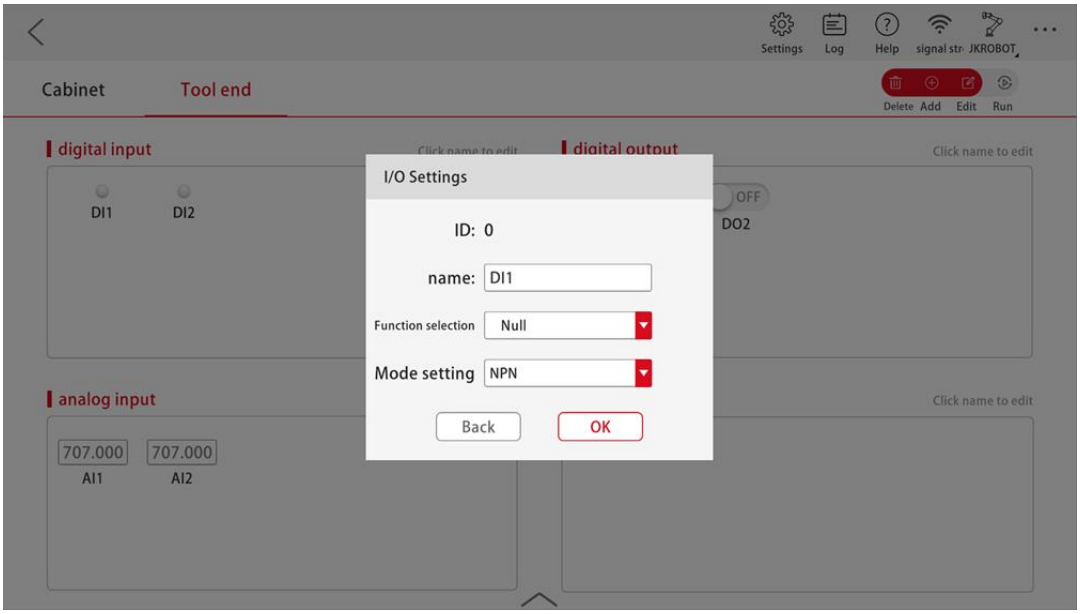


Figure 3-68 Mode Setting of TIO Digital Input Pins

Digital Output Configuration

2 DOs in the TIO function can be multiplexed as RS485 channel pins. When used as DO, it can also be set as different output modes, such as NPN output, PNP output, push pull output, etc. The two DOs are configured as NPN outputs by default once TIO is powered up. Clicking the pin name in the [Main Menu] -> [IO Monitor] -> [Tool Side] -> [Digital Output] interface, and IO setting interface will pop up, as shown in Figure 3-69 below, where you can set the output type of the corresponding pin in "Mode Settings" drop-down box.

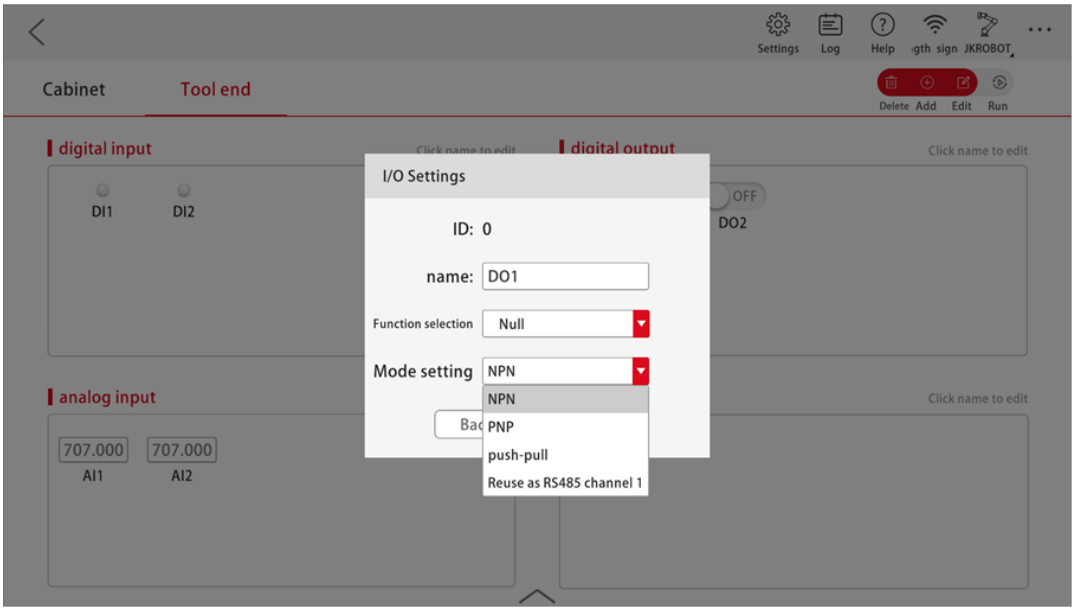


Figure 3-69 Mode Setting of TIO Digital Output Pins

Note: When used as RS485 channels, 2 DOs must be set as RS485 channel 1 mode simultaneously. When one of the DOs is modified to RS485 channel 1 mode in the interface, the other will also be set as RS485 channel 1; When one of the DOs is modified from RS485 channel 1 mode to other mode, the other will be automatically set as NPN output mode.

Analog Signal Configuration

TIO function provides two multiplexable analog input channels, and the 2 analog input pins can be multiplexed as RS485 channel 2. TIO will be used as an analog input by default when powered up. Clicking the pin name in the [Main Menu] -> [IO Monitor] -> [Tool Side] -> [Analog Input] interface, and IO setting interface will pop up, as shown in Figure 3-70 below, where you can set the mode of the corresponding pin in "Mode Settings" drop-down box.

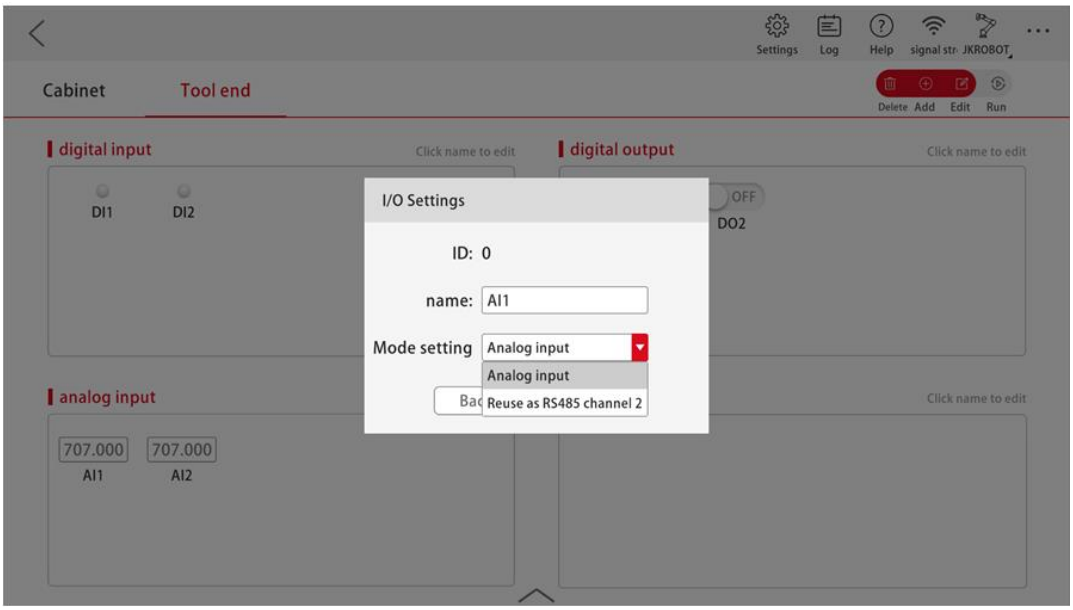


Figure 3-70 Mode Setting of TIO Analog Input Pins

3.2.3 Modbus IO

JAKA robot electrical control cabinet supports Modbus communication protocol, and can be used as a Modbus communication slave station to interact with external devices. The Modbus settings for digital input, digital output, analog input and analog output of the electronic control cabinet (including V1.0, V2.1 and MiniCab) are shown in Figure 3-71 below. I/O signals in Modbus window are the I/O data accessible by the robot and external devices through Modbus communication protocol. The controller supports 128 digital inputs and 128 digital outputs as a Modbus device; There are 16 integer analog inputs and 16 integer analog outputs for Modbus; There are 16 signed number analog inputs and 16 signed number analog outputs for Modbus; There are 32 floating-point number analog inputs and 32 floating-point number analog outputs for Modbus. Modbus register address definition is detailed in the Appendix.

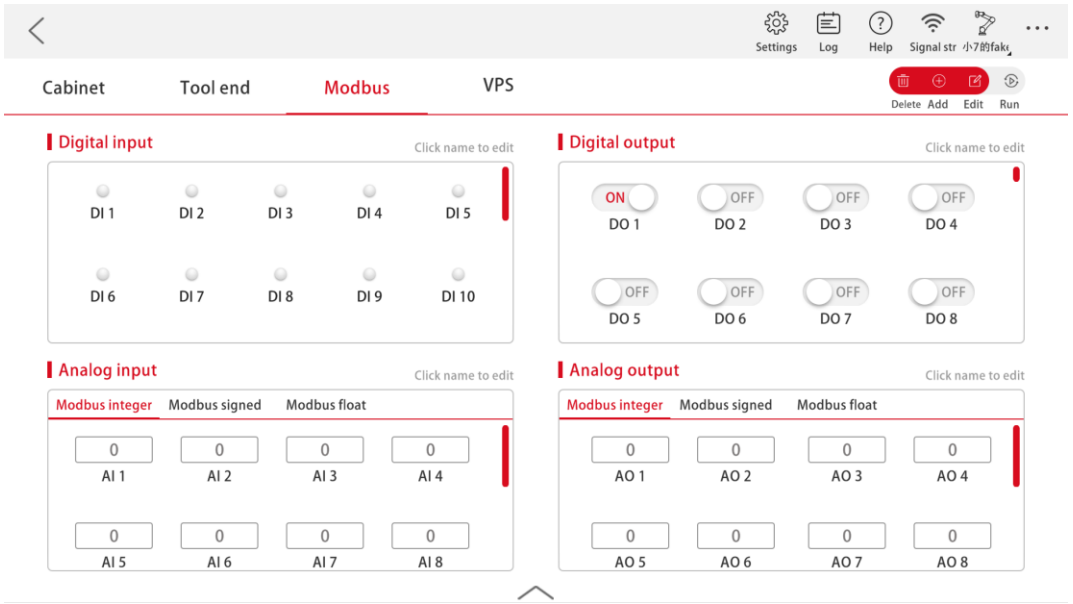


Figure 3-71 Schematic Diagram of Modbus I/O

The state of digital inputs in Modbus can be monitored by digital input interface. Click the digital input signal to edit the corresponding DI name and select DI function settings. Select the function to be set for this DI in the function selection drop-down box and click OK. When this DI signal is triggered, the corresponding function will be enabled. As shown in Figure 3-72.

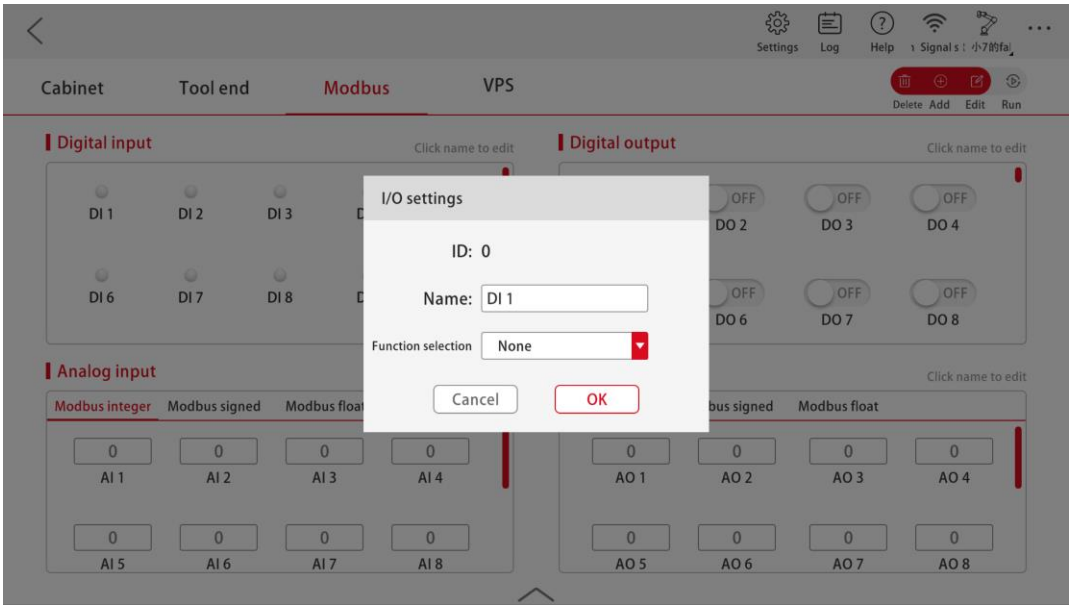


Figure 3-72 Schematic Diagram of DI Settings

The state of digital outputs in Modbus can be monitored by digital output interface. Click the digital output signal to edit the corresponding DO name and select DO function settings. Set predefined state variables of the system bound to DO in the function selection drop-down box and click OK. This DO signal can reflect the state of bound system state variables in real time. As shown in Figure 3-73.

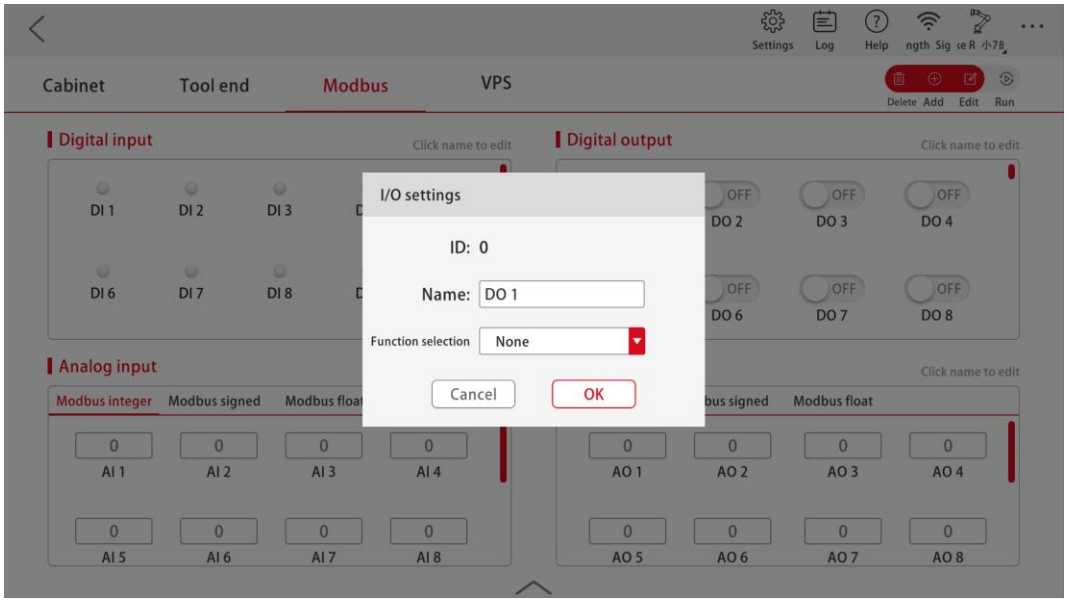


Figure 3-73 Schematic Diagram of DO Settings

Modbus analog signal variables can be monitored by analog input/output interface. The name of analog input AI can be set in analog input interface. (As shown in Figure 3-74); The name and value of analog output AO can be set in analog output interface (As shown in Figure 3-75).

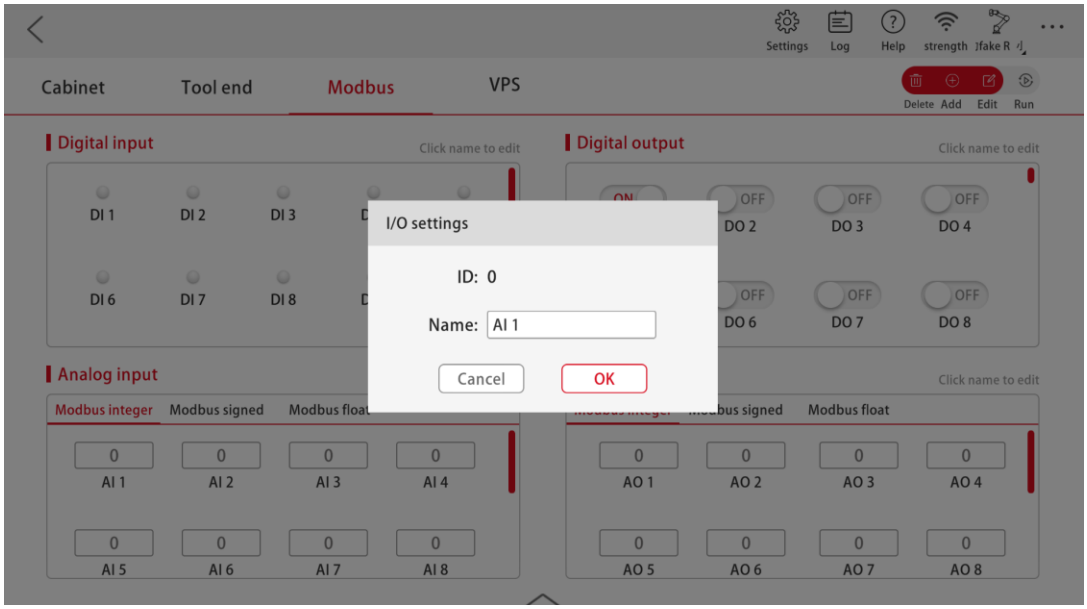


Figure 3-74 Schematic Diagram of AI Settings

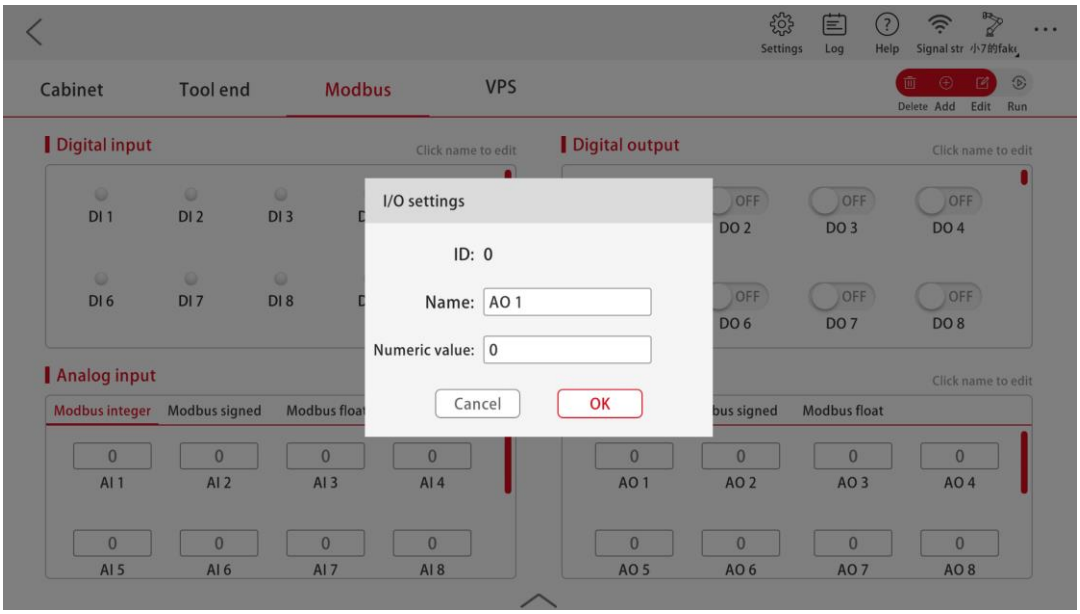


Figure 3-75 Schematic Diagram of AO Settings

3.2.4 Profinet IO

JAKA robot electrical control cabinet supports Profinet communication protocol, and can be used as a Profinet IO-Device slave station to interact with external devices. When the Profinet IO function is enabled, IO interface will display Profinet IO data in the electrical control cabinet tab. The Profinet settings for digital input, digital output, analog input and analog output of the electronic control cabinet (including V1.0, V2.1 and MiniCab) are shown in Figure 3-83 below. I/O signals in Profinet window are the I/O data accessible by the robot and external devices through Profinet communication protocol. The controller supports 64 digital inputs and 64 digital outputs as a Profinet device; There are 32 signed number analog inputs and 32 signed number analog outputs for Profinet; There are 32 floating-point number analog inputs and 32 floating-point number analog outputs for Profinet. Profinet register address definition is

detailed in the Appendix. For GSDML-XML form device description files, please consult relevant JAKA technical staffs.

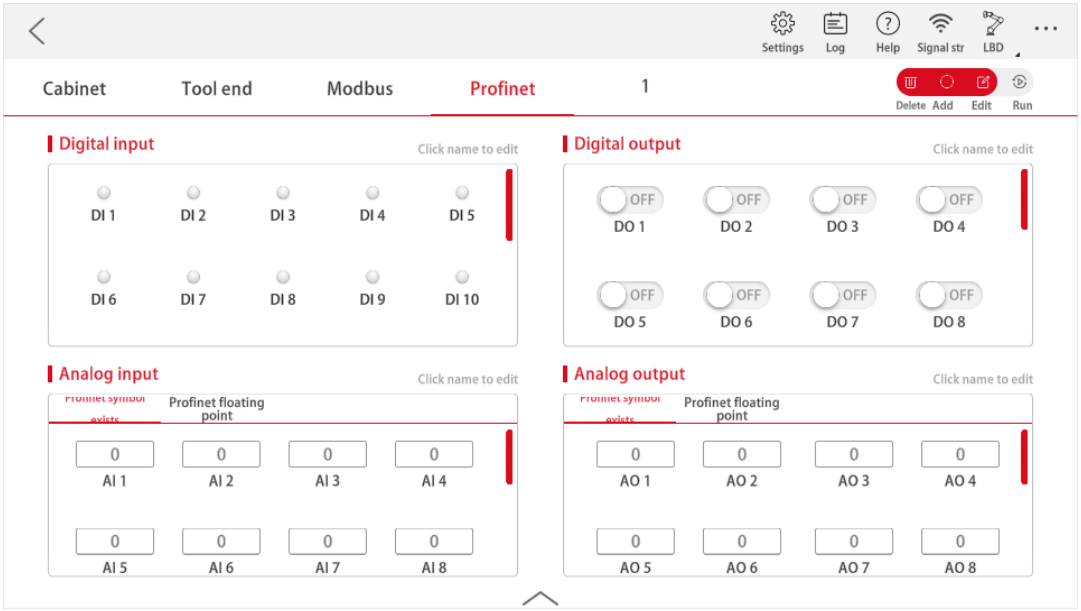


Figure 3-76 Schematic Diagram of Profinet I/O

The state of digital inputs in Profinet can be monitored by digital input interface. Click the digital input signal to edit the corresponding DI name and select DI function setting. Select the function to be set for this DI in the function selection drop-down box and click OK. When this DI signal is triggered, the corresponding function will be enabled. As shown in Figure 3-77.

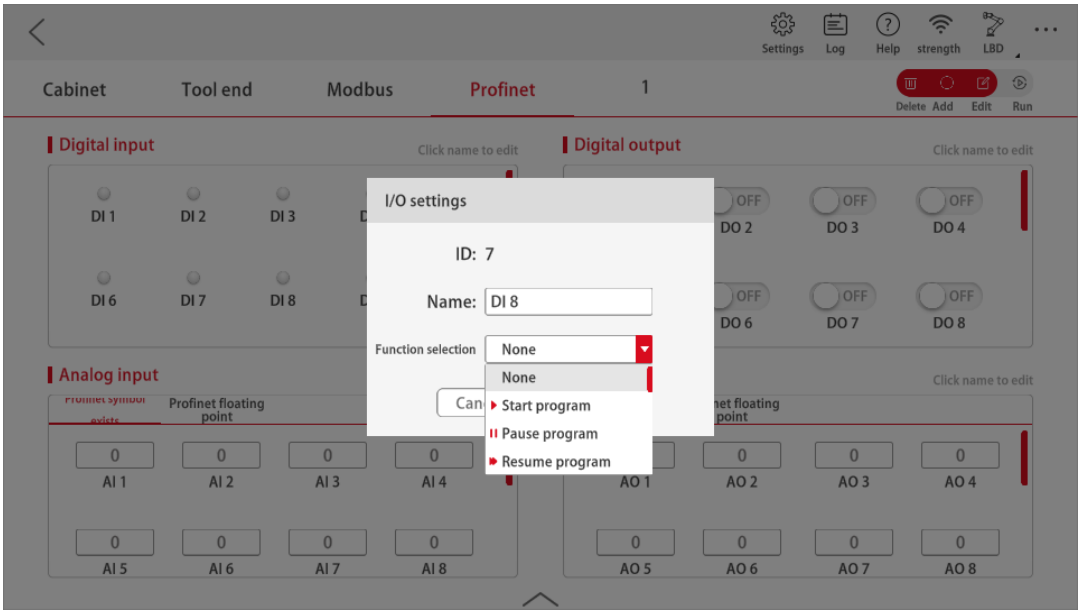


Figure 3-77 Schematic Diagram of DI Settings

The state of digital outputs in Profinet can be monitored by digital output interface. Click the digital output signal to edit the corresponding DO name and select DO function settings. Set predefined state variables of the system bound to DO in the function selection drop-down box and click OK. This DO signal can reflect the state of bound system state variables in real time. As shown in Figure 3-78.

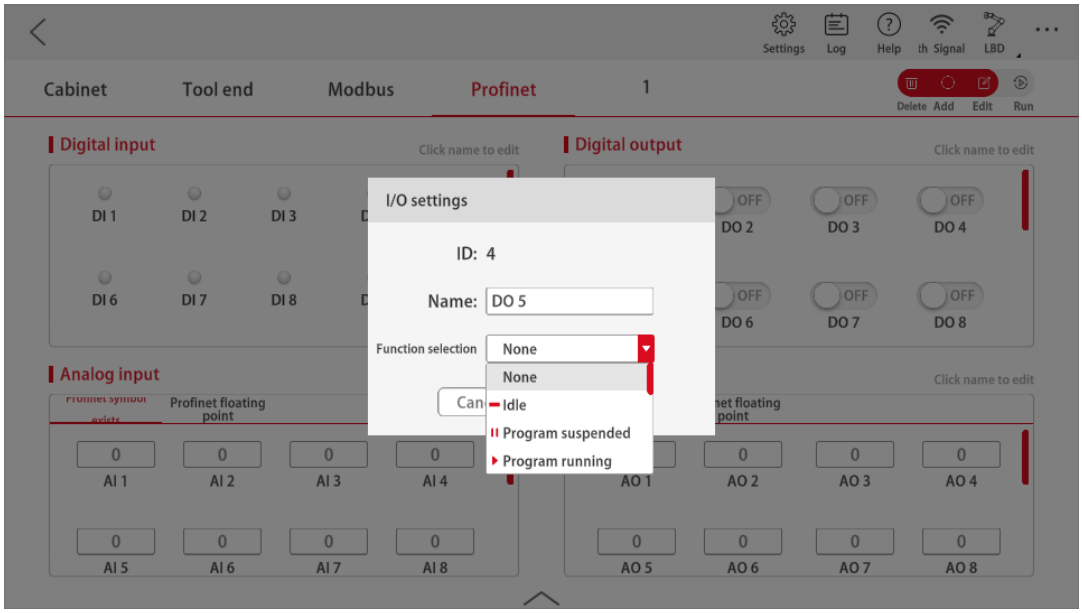


Figure 3-78 Schematic Diagram of DO Settings

Profinet analog signal variables can be monitored by analog input/output interface. The name of the analog input AI can be set in analog input interface. (As shown in Figure 3-79). The name and value of the analog output AO can be set in analog output interface (As shown in Figure 3-80).

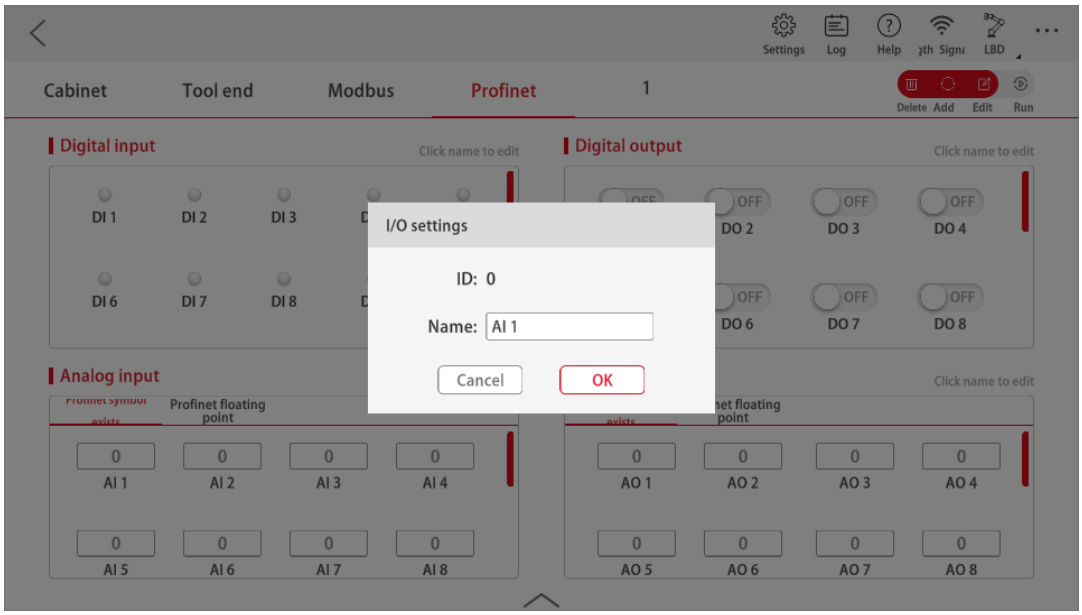


Figure 3-79 Schematic Diagram of AI Settings

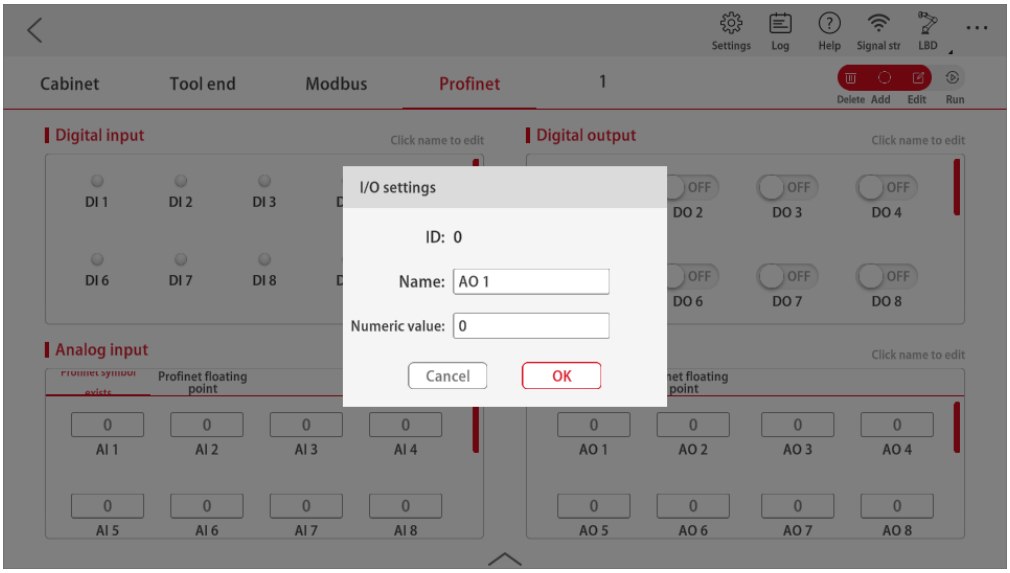


Figure 3-80 Schematic Diagram of AO Settings

3.2.5 Ethernet/IP IO

JAKA robot electrical control cabinet supports Ethernet/IP communication protocol, and can be used as an Ethernet/IP communication "adaptor" to interact with external devices. After Ethernet/IP IO function is enabled, IO interface will display Ethernet/IP IO data in the electrical control cabinet tab. The Ethernet/IP settings for digital input, digital output, analog input and analog output of the electronic control cabinet (including V1.0, V2.1 and MiniCab) are shown in Figure 3-88 below. I/O signals in Ethernet/IP window are the I/O data accessible by the robot and external devices through Ethernet/IP communication protocol. The controller supports 64 digital inputs and 64 digital outputs as an Ethernet/IP device; There are 24 signed number analog inputs and 24 signed number analog outputs for Ethernet/IP; There are 24 floating-point number analog inputs and 24 floating-point number analog outputs for Ethernet/IP. Ethernet/IP register address definition is defined in appendix.

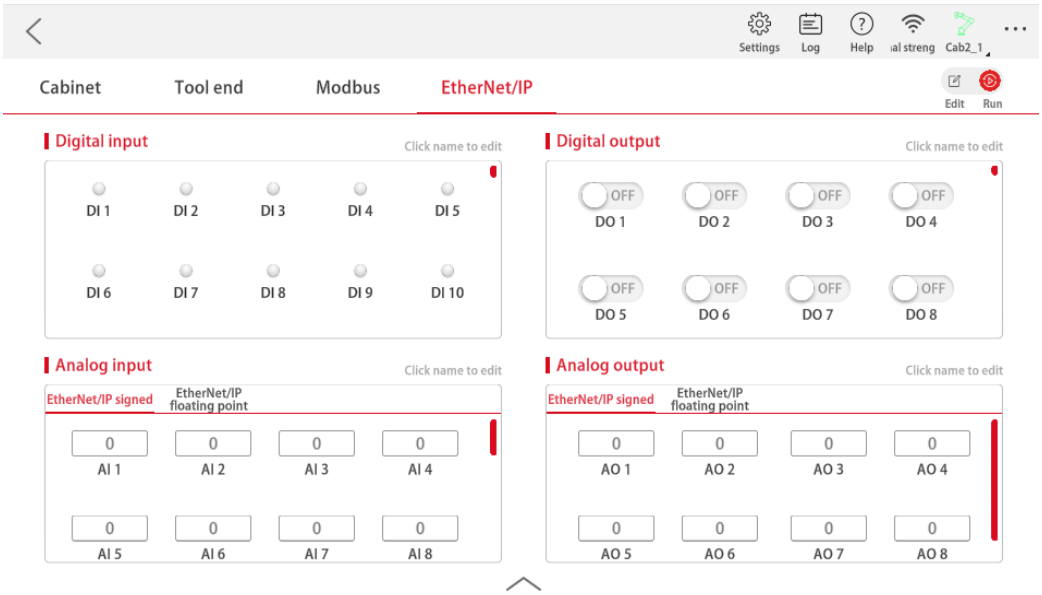


Figure 3-81 Schematic Diagram of Ethernet/IP I/O

The state of digital inputs in Ethernet/IP can be monitored by digital input interface. Click the digital input signal to edit the corresponding DI name and select DI function settings. Select the function to be set for this DI in "Function Selection" drop-down box and click OK. When this DI signal is triggered, the corresponding function will be enabled. As shown in Figure 3-82.

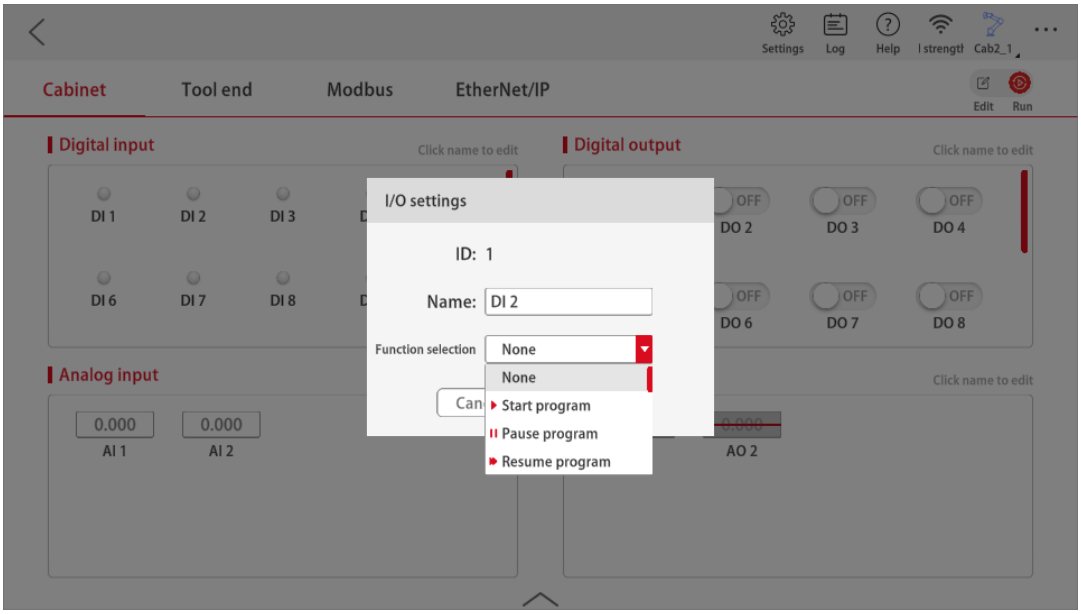


Figure 3-82 Schematic Diagram of DI Settings

The state of digital outputs in Ethernet/IP can be monitored by digital output interface. Click the digital output signal to edit the corresponding DO name and select DO function settings. Set predefined state variables of the system bound to DO in "Function Selection" drop-down box and click OK. This DO signal can reflect the state of the bound system state variable in real time. As shown in Figure 3-83.

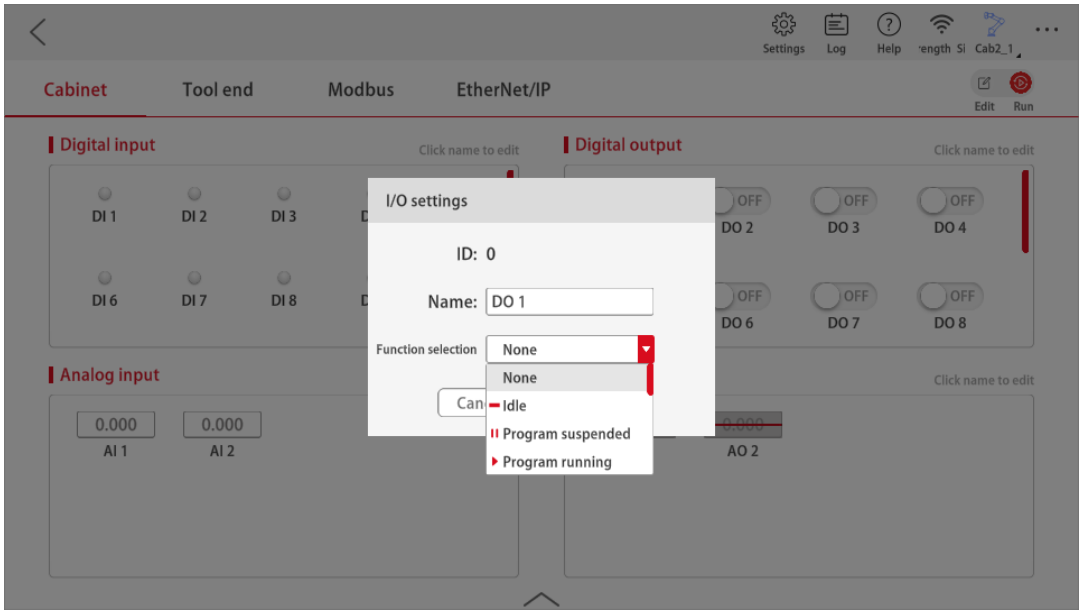


Figure 3-83 Schematic Diagram of DO Settings

Ethernet/IP analog signal variables can be monitored by analog input/output interface. The name of analog input AI can be set in analog input interface. (As shown in Figure 3-84); The name and value of the

analog output AO can be set in analog output interface (As shown in Figure 3-85).

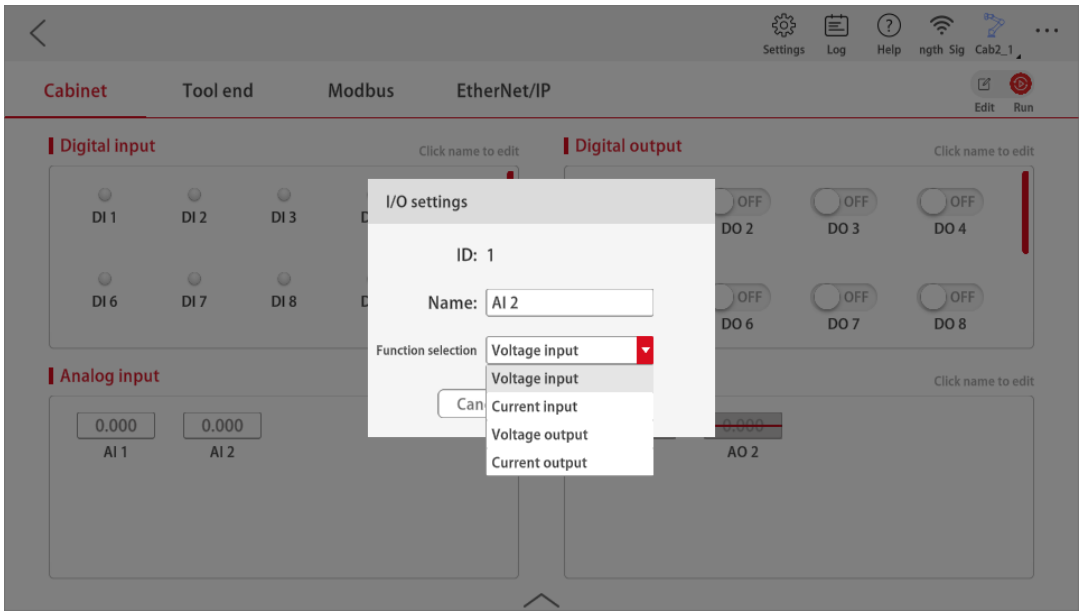


Figure 3-84 Schematic Diagram of AI Settings

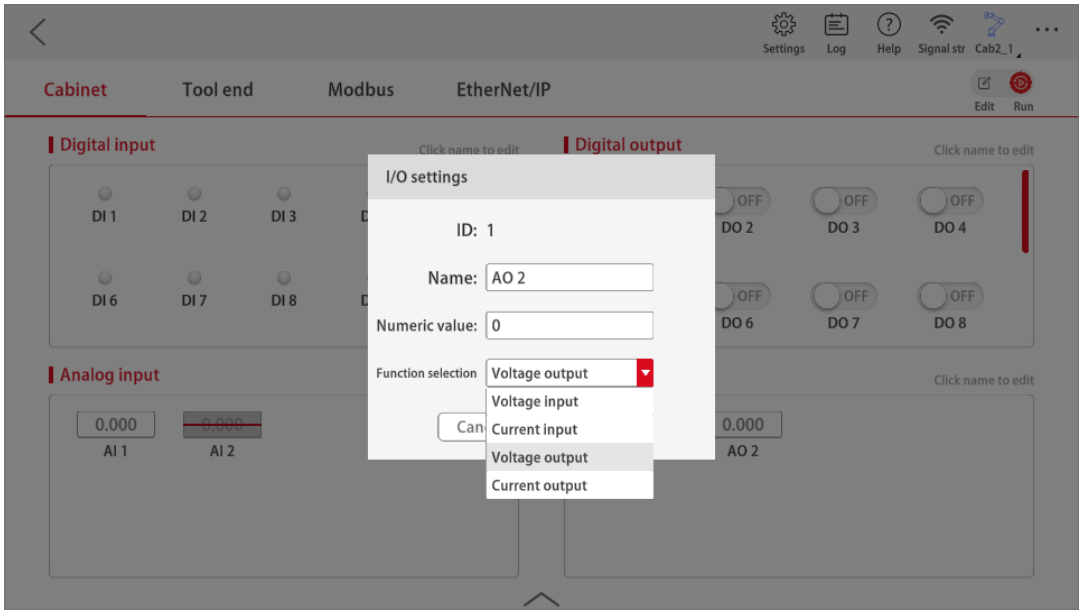


Figure 3-85 Schematic Diagram of AO Settings

3.2.6 Function IO

DI function can be set in I/O interface. Select the function to be set for this DI in "Function Selection" drop-down box and click OK. When this DI signal is triggered, the corresponding function will be enabled. Currently DI signal supports following functions: start program, pause program, resume program, stop program, power on, power off, enable robot, disable robot, level 1 reduction mode, level 2 reduction mode, protective stop, back to original position, clear fault, enter drag mode and exit drag mode. As shown in Table 3-2. Function trigger conditions are shown in the following table:

Table 3-2 Function Trigger Mode Table

Function Name	Trigger Method

Start program	Rising edge signal
Pause program	Rising edge signal
Resume program	Rising edge signal
Stop program	Rising edge signal
Power on	Rising edge signal
Power off	Rising edge signal
Enable robot	Rising edge signal
Disable robot	Rising edge signal
Level 1 reduction mode	Low level signal
Level 2 reduction mode	Low level signal
Protective stop	Low level signal
Back to original position	High level signal
Clear fault	Rising edge signal
Enter drag mode	Rising edge signal
Exit drag mode	Rising edge signal

Note:

For fault clear, only collision alarm can be cleared, and other abnormal alarms cannot be cleared.

Level 2 reduction rate should be less than that of Level 1, and reduction rate is set in "Settings" - "Safety Settings" - "Protection System".

You can set DO function in I/O interface. Set predefined state variables of the system bound to DO in "Function Selection" drop-down box and click OK. This DO signal can reflect the state of the bound system state variable in real time. Currently the following states can be bound to DO signal: Idle, program pause, program operating, error, powered on, enabled, in motion, in rest, powered on, system E-stop button state, robot reduction state, system protective stop state, safety position, etc.

Note:

The idle state is the state in which the robot is not operating, and is independent of the robot arm state.

The error state is the state in which the robot triggered collision alarm.

The in-motion state is the state which is triggered when the robot is in motion (program operation, manual control, secondary development control motion, etc.), and is independent of the program state.

The in-rest state is the state in which the robot is at rest (program paused, no program operating, program finished, waiting for a signal, etc.), and is independent of the program state.

The powered-on state is the state in which the electrical control cabinet is in the power-on state, and is independent of the robot arm state.

The robot reduction state is the state to define whether the robot triggers Level 2 reduction mode.

In Level 2 reduction mode, the end speed cannot exceed 250mm/s.

The safety position is triggered when the robot is in the reset pose position (set in "Robot Pose" of "Safety Settings").

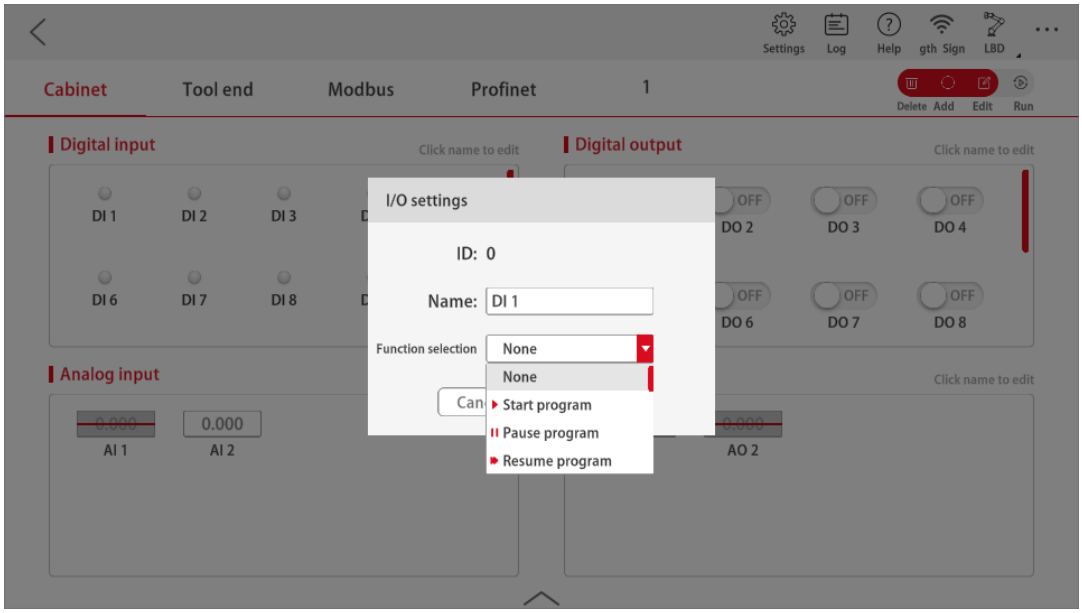


Figure 3-86 Schematic Diagram of Function DI

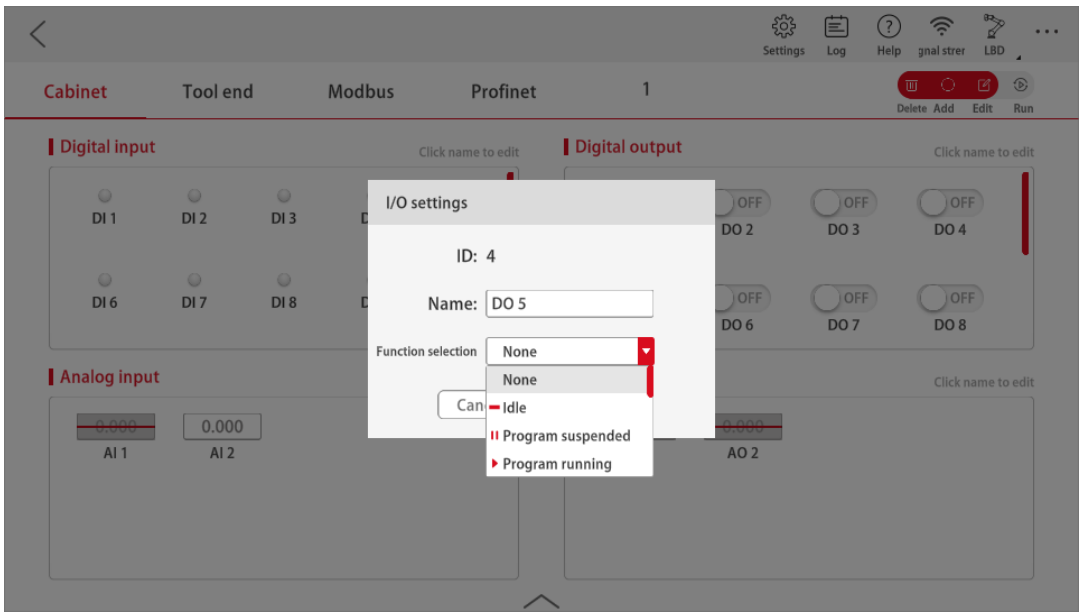


Figure 3-87 Schematic Diagram of Function DO

3.2.7 How to Add Extended I/O

I/O panel supports dynamic I/O configuration function to make the controller as Client side/master station in Modbus communication. The function is located in the upper right corner of I/O panel interface and has two states: operating and editing. When switched to editing state, Dynamic I/O can be configured by clicking '+'.

The configuration is divided into Modbus-TCP and Modbus-RTU. Modbus settings are connected to the communication interface of electrical control cabinet, in which RTU mode is connected via USB or Modbus RTU, and TCP mode is connected via Ethernet. The setting steps are shown as follows:

Note: Maximum number of extended IO: 32 for AIO and 64 for DIO, and up to 8 modules are supported for expansion.

3.2.7.1 Modbus-TCP

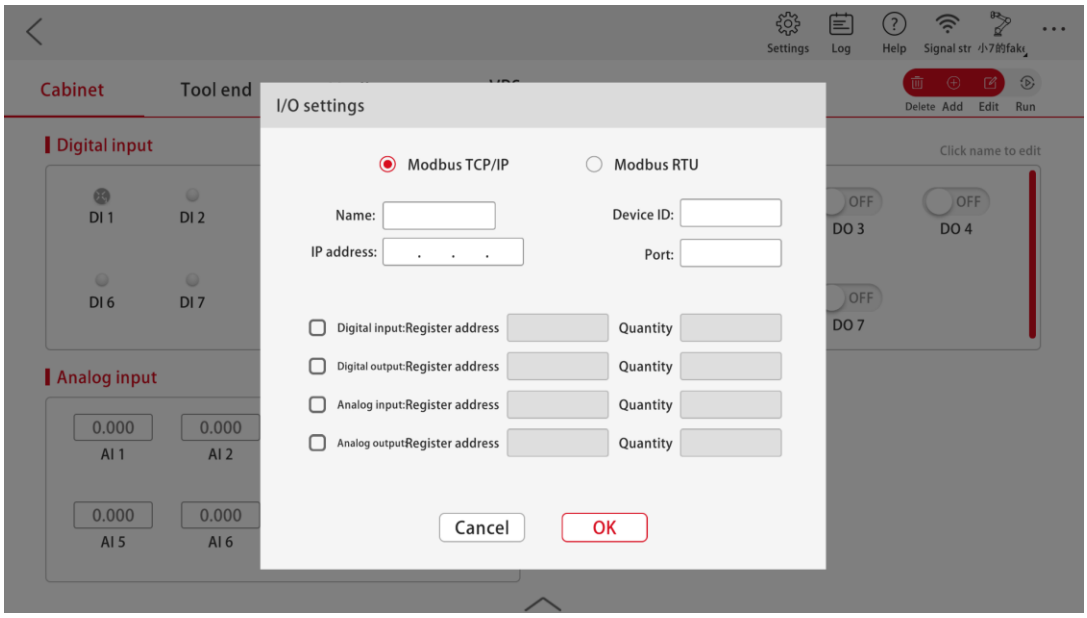


Figure 3-88 Modbus-TCP Interface

Name: Modbus-TCP naming

IP Address: Enter the IP address of Modbus-TCP

Device ID: Fill in the device number

Port Number: Fill in the port number

Then configure the register addresses and numbers of DI, DO, AI, and AO of TCP.

Click OK, and the newly defined Modbus module will be displayed in I/O monitoring interface.

3.2.7.2 Modbus-RTU

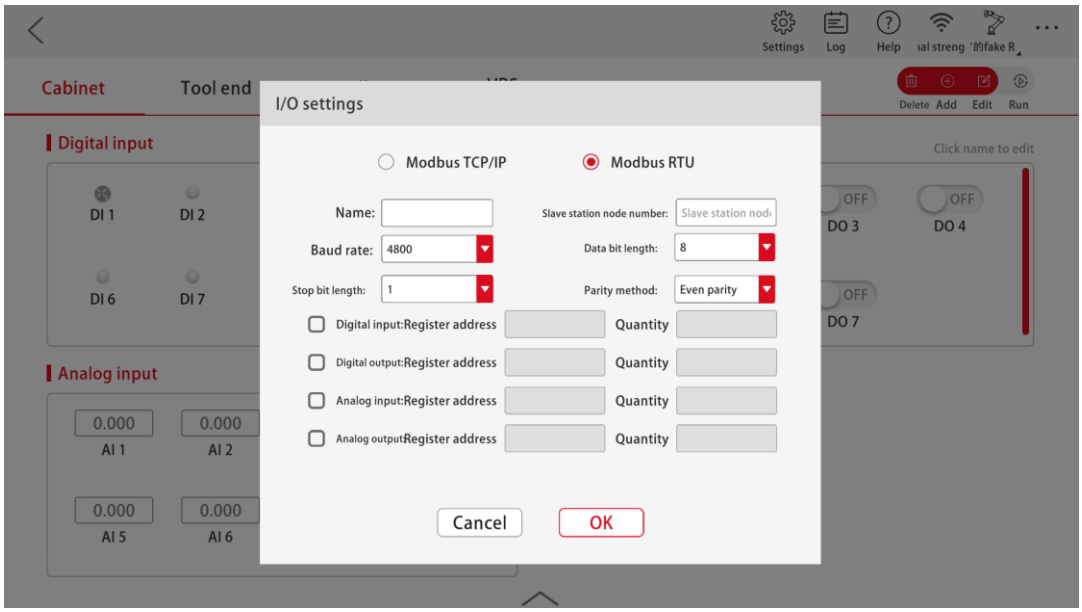


Figure 3-89 Modbus-RTU Interface

Name: Modbus-RTU naming

Baud Rate: Baud rate of the configured Modbus-RTU

Stop Bit Length: Stop bit length of the configured Modbus-RTU

Slave Node Number: Station number of the configured Modbus-RTU

Data Bit Length: Data bit length of the configured Modbus-RTU

Verification Method: Verification method of the configured Modbus-RTU

Register Address & Number: Fill in the information according to Modbus-RTU parameters.

Click OK, and the newly defined Modbus module will be displayed in I/O monitoring interface.

Chapter 4 Motion Operations

4.1 Motion Control

Click "Manual Operation" on homepage to enter manual operation interface. If you are in another interface, click the " Up" arrow below the screen and "Interface Switch" tab will pop up. Click "Manual Operation" to enter manual operation interface.

In JAKA Zu software, two types of coordinate system can be self-defined for Zu Robot, which are tool coordinate system and user coordinate system. Both tool coordinate system and user coordinate system can be self-defined and used in "General Settings" option. The two coordinate systems are presented separately in the robot model in manual operation interface. As shown in Figure 4-1.

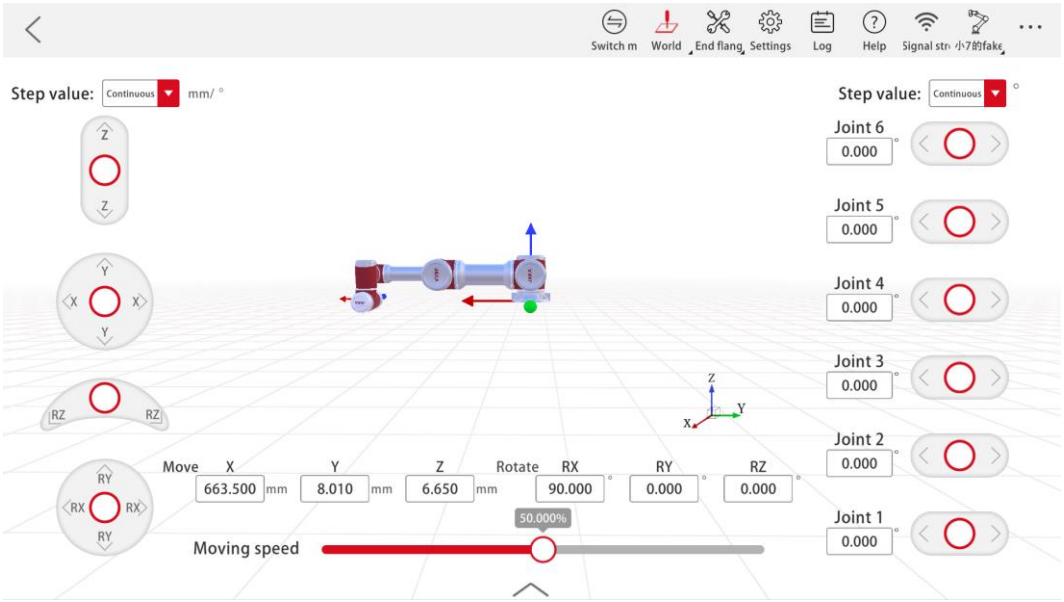


Figure 4-1 Manual Operation Window Diagram

4.1.1 How to Switch Motion Reference Coordinate System

Click "Switch JOG Coordinate System" button in manual operation interface. When the color of user coordinate system icon turns red, it means the user coordinate system is currently in use; When the color of tool coordinate system icon turns red, it means the tool coordinate system is currently in use. (User coordinate system is world coordinate system by default, and tool coordinate system is end flange coordinate system by default).

4.1.2 Switching of User Coordinate System

Click the small triangle at the bottom right corner of the user coordinate system icon to pull down the user coordinate system list, and click any user coordinate system name to switch to user coordinate system. As shown in Figure 4-2.

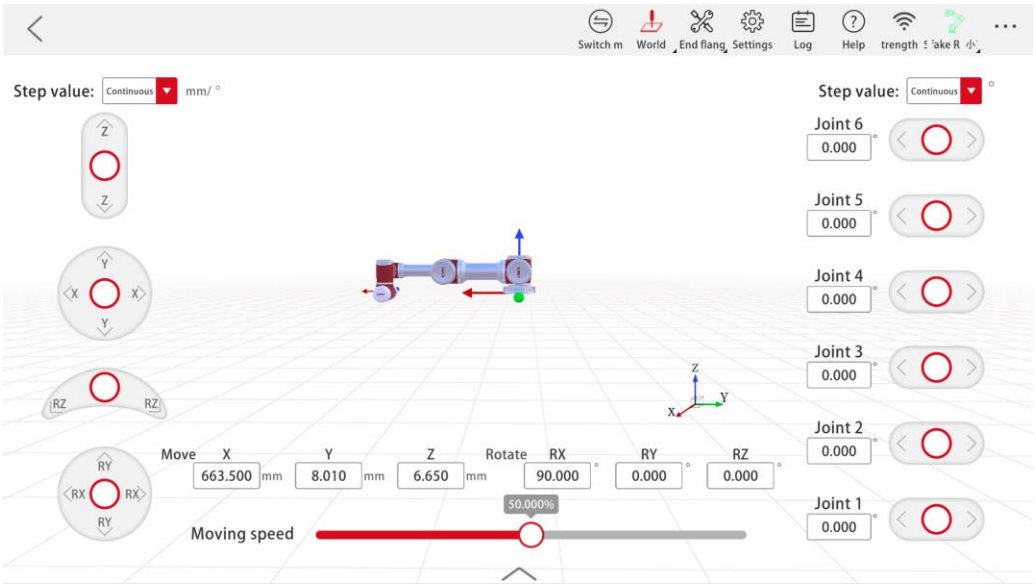


Figure 4-2 Switching Manual Operation Interface to User Coordinate System Interface

4.1.3 Switching of Tool Coordinate System

Click the small triangle at the bottom right corner of the tool coordinate system icon to pull down the tool coordinate system list, and click any tool coordinate system name to switch to tool coordinate system. As shown in Figure 4-3.

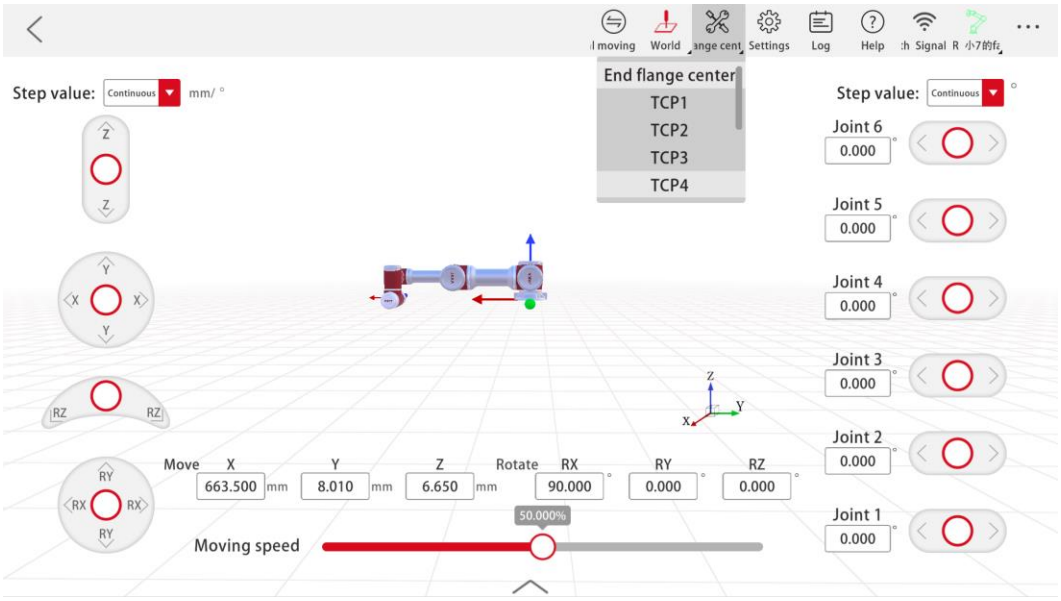


Figure 4-3 Switching Manual Operation Interface to Tool Coordinate System Interface

4.1.4 How to Precisely Control Robot Motion

There are step value options at the top of both sides of manual operation interface, which respectively control the step value of the virtual joystick below. Control the distance or angle of the robot motion under manual operation by changing the step value. The smaller the step value, the more accurate the robot motion. As shown in Figure 4-4

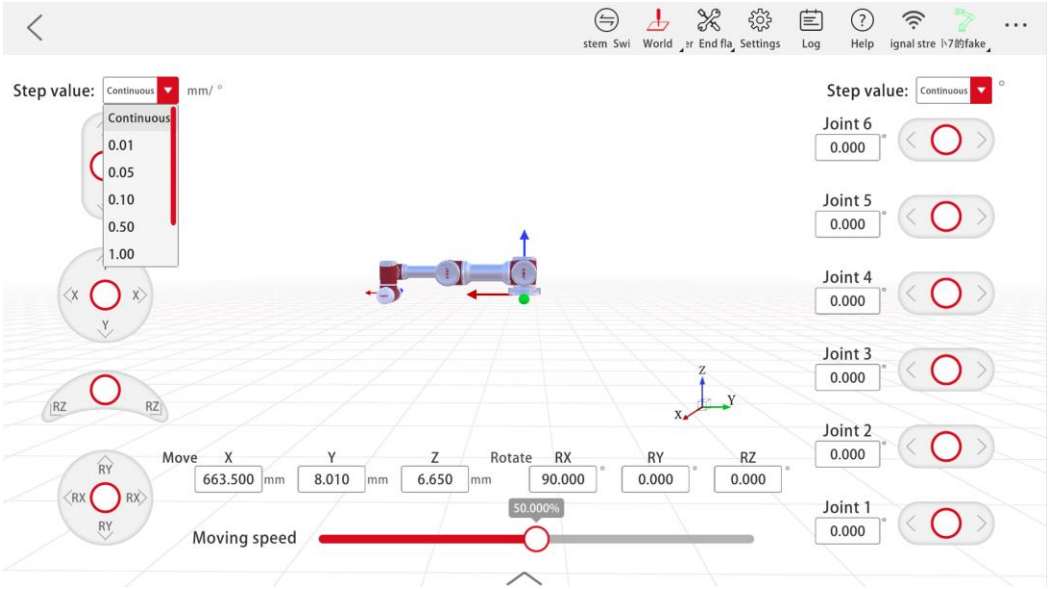


Figure 4-4 Step Value System in Manual Operation Interface

4.1.5 Motion Speed Control

Adjust the robot moving speed by dragging the motion speed bar at the bottom of the manual interface. You can also click the percentage above the motion speed bar to enter a specific percentage speed.

4.2 Spatial Motion

Spatial motion refers to tool coordinate system origin of the current robot in Cartesian space or task space. Users can choose to move in the user coordinate system or in the tool coordinate system. As shown in Figure 4-5. Spatial motion is the joint motion of individual joints, and the manual operation for robot spatial motion is shown below:

Slide and hold the virtual joystick on the left side of the interface, and the robot's tool coordinate system origin will perform corresponding spatial motion in the current user coordinate system.

When released, the virtual joystick will automatically return to the initial point and the robot will immediately stop.

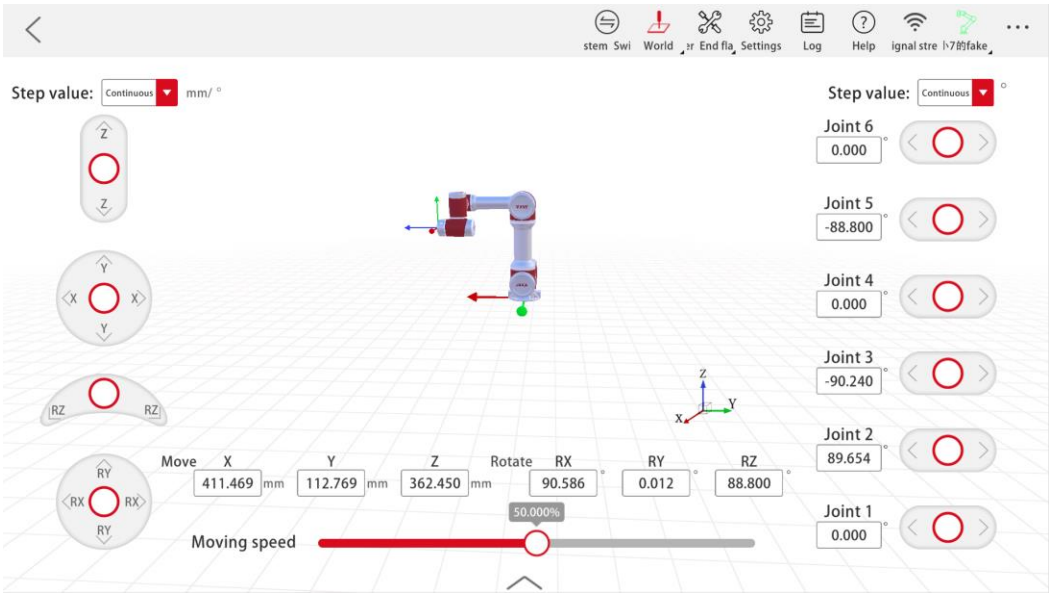


Figure 4-5 Spatial Motion Interface

4.3 Joint Motion (MoveJ)

JAKA Zu robot is composed of six joints (as shown in Figure 4-6), and joint movement refers to independent movements of these joints by manual operation, as shown in Figure 4-7. The manual operation of single joint movement is shown below:

Slide and hold the virtual joystick on the right side of the interface, and the corresponding robot joint will immediately rotate in the corresponding direction.

When released, the virtual joystick will automatically return to the initial point and the robot will immediately stop.

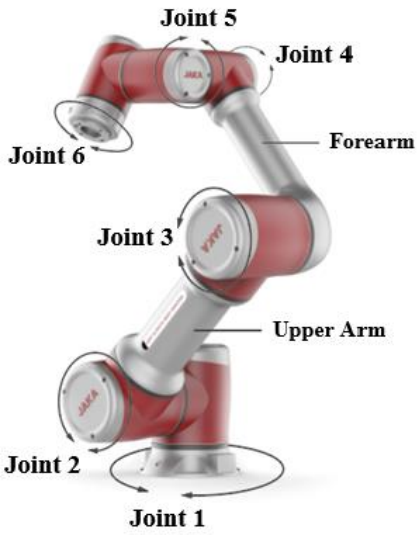


Figure 4-6 Schematic Diagram of Robot Joints

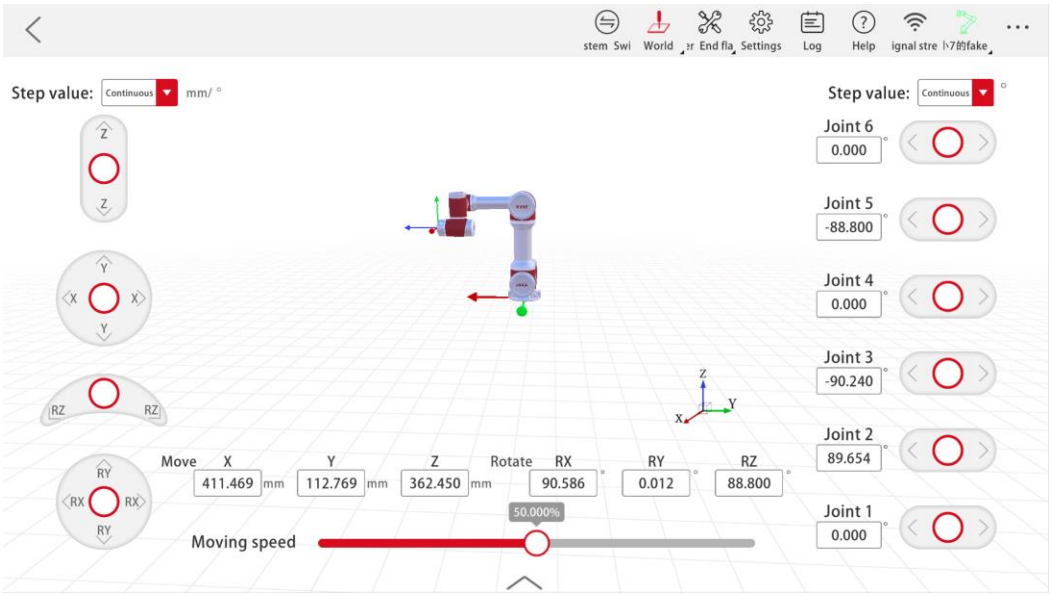


Figure 4-7 Joint Movement Interface

4.4 Position Motion

Position motion refers to specified movement positions of the robot by manual operation. Users can specify the joint position of the robot and the spatial position of the current tool coordinate system origin in the current user coordinate system. As shown in Figure 4-8. The manual operation of position motion is shown below:

Click any of the joint information or spatial position information boxes in manual operation interface to enter position motion interface.

（1）If you need the robot to move to the specified joint position, please enter the end position of six joints in the joint position area. Press and hold "Move to this point", and the robot will move to the specified position. Click OK to exit position motion interface, and position motion is completed.

（2）If you need the robot to move to the specified spatial position, please enter the spatial position of end point in TCP position area. After clicking "Calculate Joint Position", press and hold "Move to this point", and the robot will move to the specified position. Click OK to exit position motion interface, and position motion is completed.

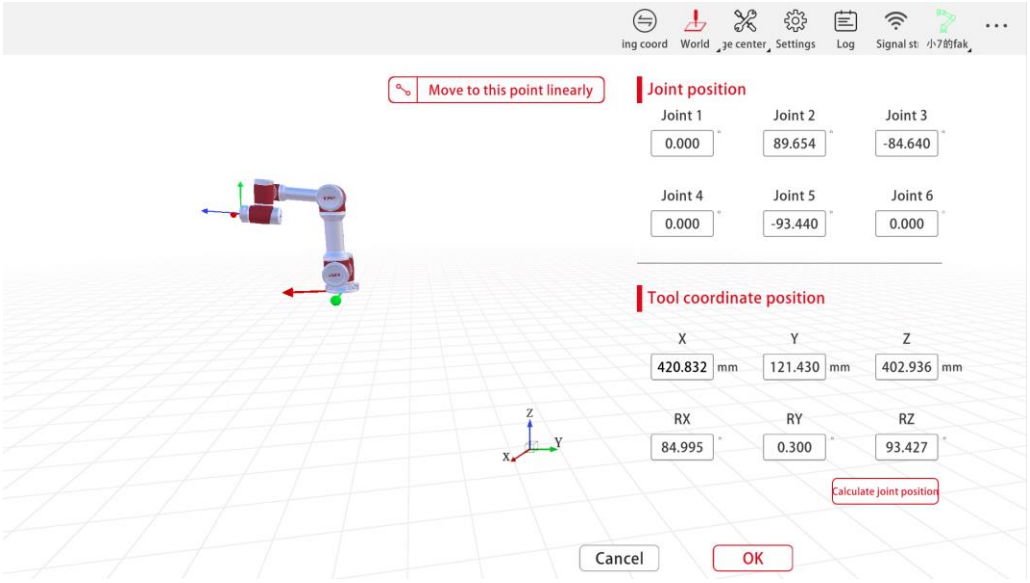


Figure 4-8 Position Motion Interface

Chapter 5 Start Programming

5.1 Instruction Introduction

JAKA Zu App provides a portable programming method that allows users to program JAKA Zu robot via basic programming skills, and the programming method is visual programming, which greatly improves work efficiency.

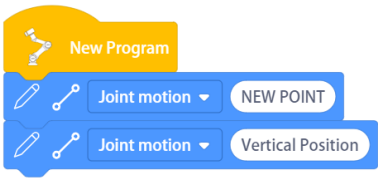
5.1.1 New Program

(1) Meaning: The program to be operated by robot.

(2) Usage: Place the instructions to be executed in order below new programs; Click the instruction to change program name.

(3) Example: Robot joint movement.

Let the robot do a joint movement from the starting point to vertical position.



Note: If it is not placed under "New Program", the module will not be executed.

5.1.2 Moving Instructions

5.1.2.1 Waypoint Variable Management

JAKA Zu software supports motion waypoint function. Users can define the motion waypoint in the waypoint variable management, and call them via corresponding motion waypoint instructions to control the robot movement to the set waypoint position.

In motion control interface, click the instruction bar on the left to enter "Movement Instruction" option, and then click "Settings" button  above to enter waypoint variable management interface. In waypoint variable management interface, users can add, modify and delete motion waypoints according to actual needs. As shown in Figure 5-1.

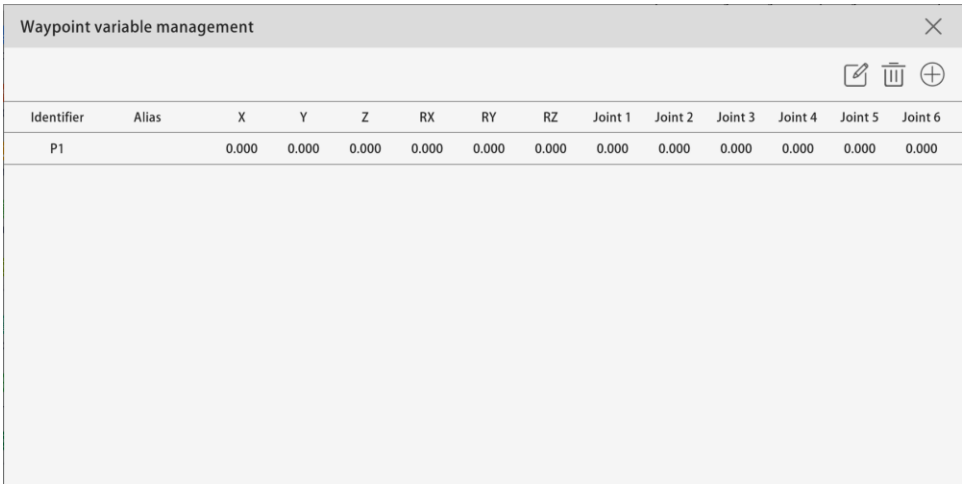
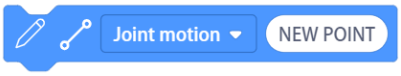


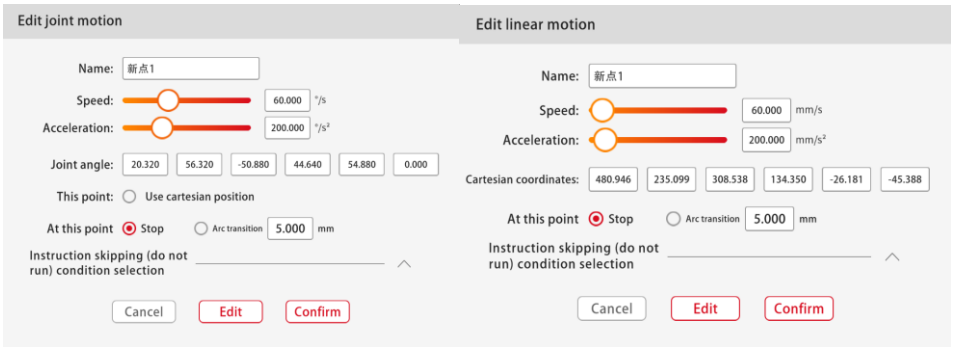
Figure 5-1 Waypoint Variable Management Interface

5.1.2.2 MoveJ or MoveL



(1) Meaning: Set endpoints or add variables, and make the robot perform joint movement or make its end perform linear motion.

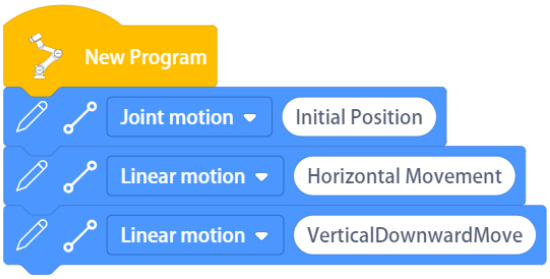
(2) Usage: Click the instruction to open "Settings" interface, in which you can edit the name, speed and acceleration of this instruction; You can use Cartesian coordinates (joint movement) to decide whether the robot stops here or performs arc transition; You can choose the stopping condition (the program will skip this instruction when the robot meets the state set by the stop condition) and edit the demonstration point.



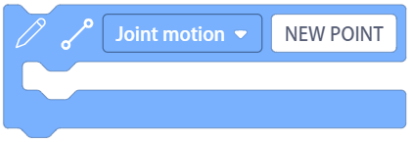
Note: In robot manual demonstration, make sure that the coordinate system is correct. Wrong coordinate system may lead to incorrect actual robot motion position.

In joint movement, if "Use Cartesian Coordinates" option is selected, the robot will calculate the corresponding joint angle value via Cartesian position saved in the current motion, and then perform joint movement to the set position.

(3) Example: After moving to the initial position, the robot joint will do a linear motion.

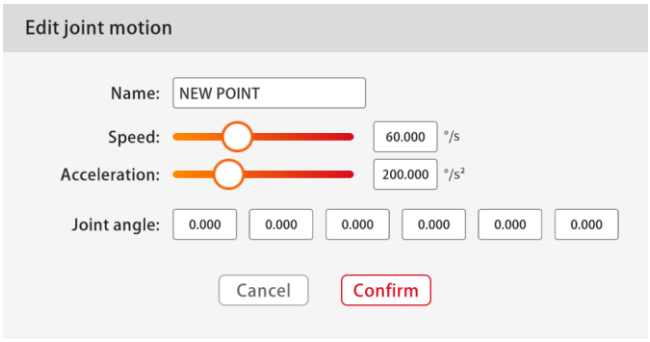


5.1.2.3 Movement Instruction Box



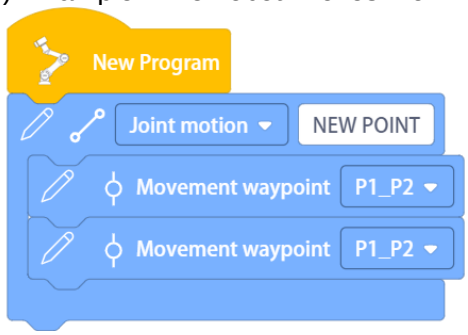
(1) Meaning: Set the name, speed and acceleration information of movement to make the robot perform MoveJ or its end to perform MoveL.

(2) Usage: Click the instruction movement switching function to switch between MoveJ or MoveL; click the instruction to open the editing interface. You can edit the name, speed and acceleration of this instruction.

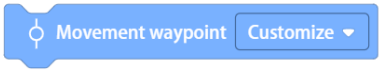


Note: This instruction is required to be used with the motion waypoint instruction.

(3) Example: The robot moves from waypoint 1 position to waypoint 2 position in MoveJ.



5.1.2.4 Movement Waypoints



(1) Meaning: Set the end point or call the waypoint defined in the waypoint variable management and make the robot perform MoveJ or its end perform MoveL.

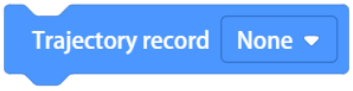
(2) Usage: Click the instruction selection function. You can customize the waypoint or call the waypoint variable management to define the waypoint, Click the instruction to open the setting interface, so you can edit the alias (custom mode cannot be modified), speed and acceleration of this instruction, choose whether to use shared parameters (speed and acceleration cannot be

modified in shared parameter mode), decide to use Cartesian coordinates (MoveJ), decide to stop here or perform arc transition, and select a stop condition (the program will skip this instruction when the robot meets the state set by the stop condition).

Note: This instruction is used with the motion instruction block, and is equivalent to the MoveJ instruction when used alone.

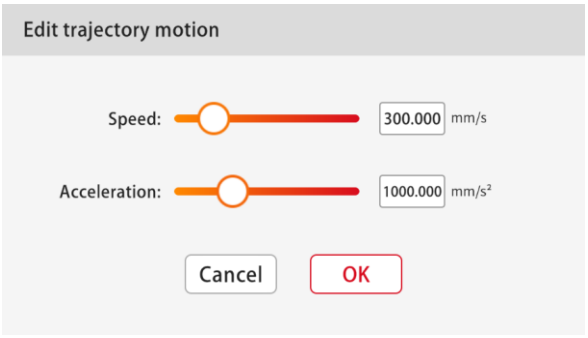
(3) Example: The robot moves from waypoint 1 position to waypoint 2 position in joint movement.

5.1.2.5 Trajectory Record

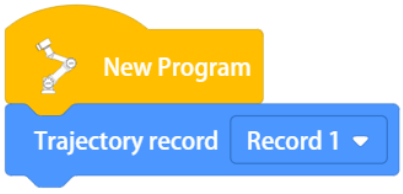


(1) Meaning: To reproduce the recorded trajectories, and the trajectories to be reproduced can be collected continuously by JOG or dragging.

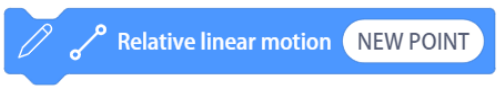
(2) Usage: In the instruction drop-down list, you can select the trajectory you need; Click the instruction to open "Settings" interface, in which you can edit the speed and acceleration of this instruction.



(3) Example: The robot is moving according to the recorded trajectory.



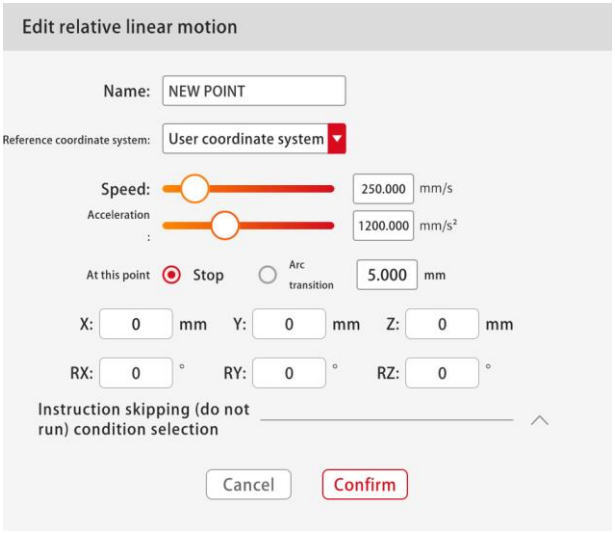
5.1.2.6 Relative MoveL



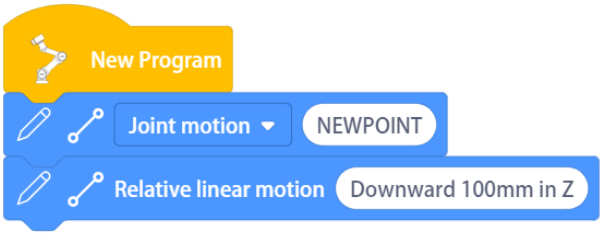
(1) Meaning: Set position increment to make the robot perform an end linear motion relative to the current position, where the position increment is the incremental value in the current user

coordinate system.

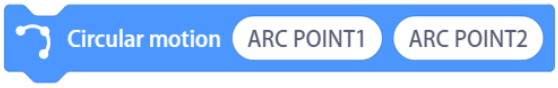
(2) Usage: Click the instruction to open "Settings" interface, in which you can edit the name, speed and acceleration of this instruction; You can choose the reference coordinate system, user coordinate system or tool coordinate system to decide whether the robot stops here or performs arc transition; You can choose the stop condition (the program will skip this instruction when the robot meets the state set by the stop condition) and edit the demonstration point.



(3) Example: Example: The robot end moves 100mm along the Z-axis negative direction of the current user coordinate system.

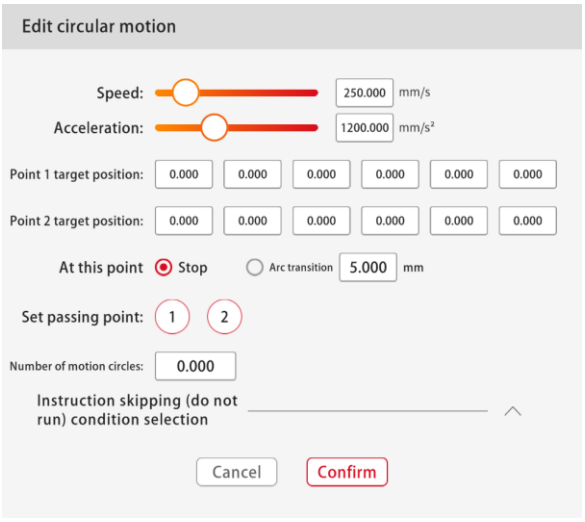


5.1.2.7 Arc Movement (MoveC)



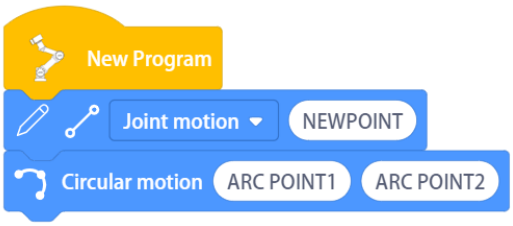
(1) Meaning: Set three points and make the robot end perform MoveC.

(2) Usage: Click the instruction to open "Settings" interface, in which you can edit the name, speed and acceleration of this instruction; You can demonstrate the passing point and the end point of the arc (the starting point is the current point), input the motion number (which is calculated from the initial point of the arc. If the number is 0, the actual end point of the arc is the set end point), and choose the stop condition (the program will skip this instruction when the robot meets the state set by the stop condition).

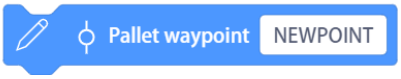


(3) Example: The robot end makes a full circular motion

Edit the initial point as well as two passing points in MoveC, and enter "1" in the "motion number".

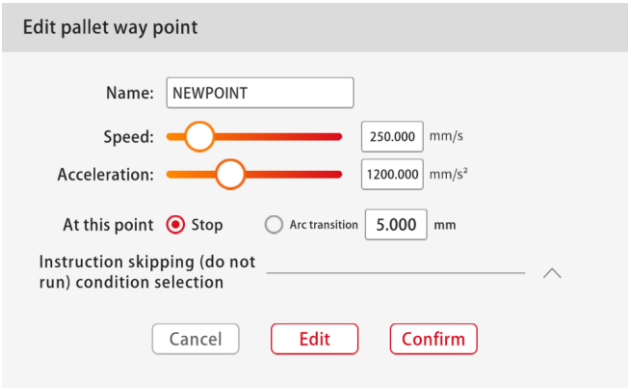


5.1.2.8 Pallet Waypoints



(1) Meaning: Set the end point of robot motion after reaching each pallet point in the pallet program.

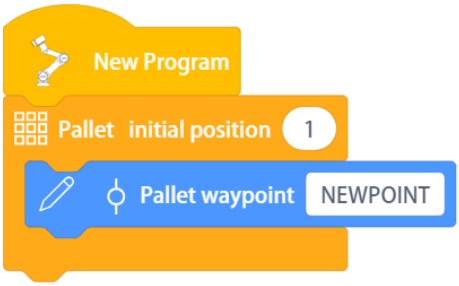
(2) Usage: Click the instruction to open "Settings" interface, in which you can edit the name, speed and acceleration of this instruction; You can decide whether the robot stops here or performs arc transition, choose the stop condition (the program will skip this instruction when the robot meets the state set by the stop condition), and edit the demonstration point (this instruction will take effect only when it is located in the pallet frame).



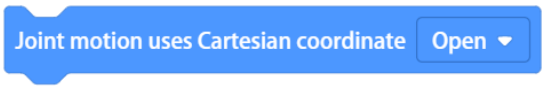
Note: This instruction will take effect only when it is located in the pallet frame.

(3) Example: Set trajectory

The robot moves 100mm vertically downward when it moves to each pallet point.



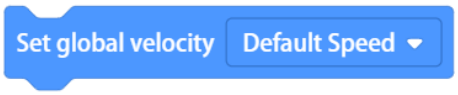
5.1.2.9 Apply Cartesian Coordinate System in MoveJ



(1) Meaning: The MoveJ performed by point positions recorded in the Cartesian coordinate system.

(2) Usage: By using this instruction before joint movement, the subsequent joint movement will be performed based on point positions recorded in the Cartesian coordinate system. Although the motion is still MoveJ, the robot will automatically perform kinematic inverse solution.

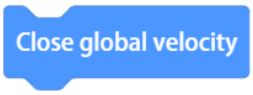
5.1.2.10 Set Global Speed



(1) Meaning: Enable global speed settings

(2) Usage: Select the appropriate speed variable in Drop-down list. After setting the global speed, the motion speed and acceleration of all subsequent MoveL and MoveJ instructions will be the same as the selected speed variable.

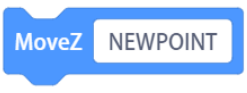
5.1.2.11 Disable Global Speed



(1) Disable global speed settings

(2) Usage: After performing this instruction, the global speed setting is disabled, and the original motion speed and acceleration are restored in the subsequent MoveL and MoveJ.

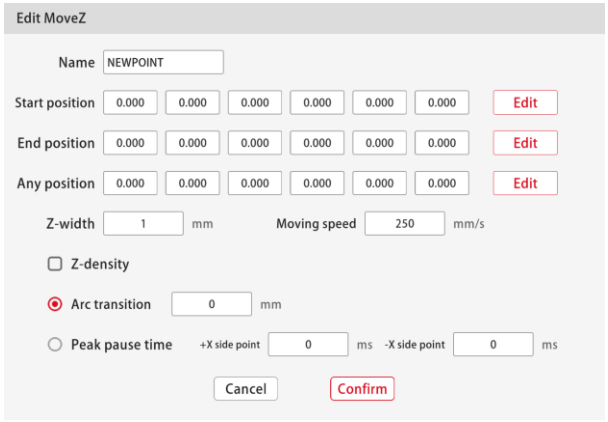
5.1.2.12 MoveZ



(1) Meaning: The end of the robot maintains the current state in a z-shaped trajectory movement.

(2) Usage: Click the instruction to open the setting interface. You can edit the start position, end position and any position of a movement (the robot will perform a Z-shaped movement in the plane

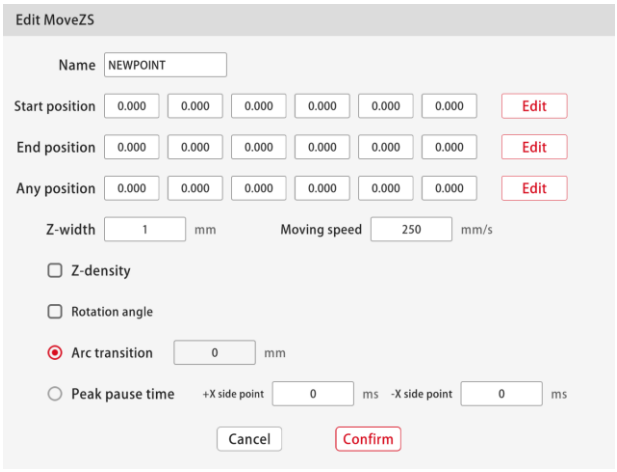
determined by these three points), the width, moving speed and density of the Z-shaped trajectory (if not set, the default is 45°), arc transition or peak pause duration (select one of two).



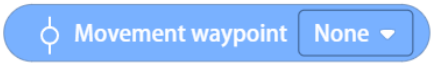
5.1.2.13 MoveZS

(1) Meaning: The end of the robot rotates RZ while moving in the set direction.

(2) Usage: Click the instruction to open the setting interface. You can edit the start position, end position and any position of the movement (the robot will execute the movement in the user coordinate system determined by these three points), the width, moving speed, density and rotation angle of the trajectory (default is 45° if not set), arc transition or peak pause duration (choose one of the two).



5.1.2.14 Obtain Motion Waypoint Parameters

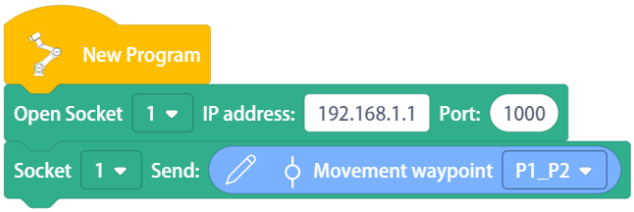


(1) Meaning: Get the information of motion waypoints in waypoint variable management.

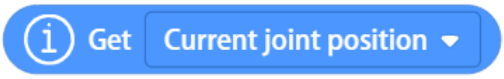
(2) Usage: It can be sent to the upper computer or assigned to the variable.

(3) Example: Retrieve the information of motion waypoint 1 in waypoint variable management.

The upper computer will receive the information of movement waypoint 1 in waypoint variable management.



5.1.2.15 Get Robot-related Parameters

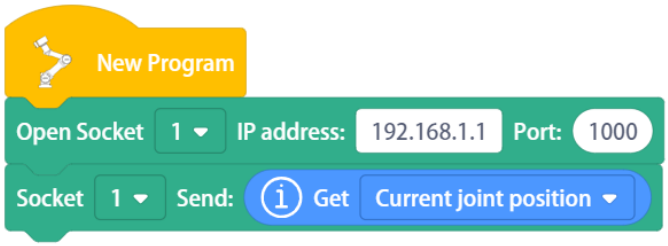


(1) Meaning: Get current joint position, tool-side center position, flange center position, end load, end force, collision sensitivity and system time of the robot.

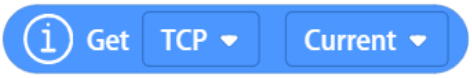
(2) Usage: It can be sent to the upper computer or assigned to a variable. Please refer to Appendix I for the data types of each parameter.

(3) Example: Read the current angle value

The upper unit will receive the current angle values of robot joints 1-6.



5.1.2.16 Get user coordinate settings or tool coordinate system settings of the robot

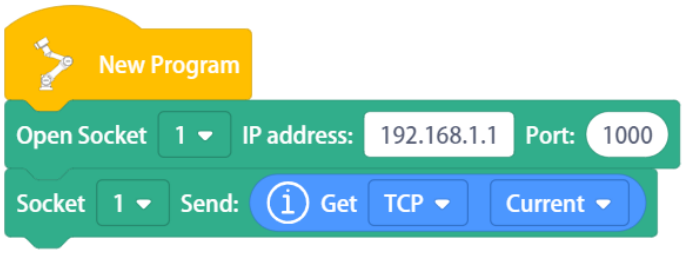


(1) Meaning: Get user coordinate settings or tool coordinate system settings of the robot.

(2) Usage: It can be sent to the upper computer or assigned to the variable.

(3) Example: Read current tool-side coordinate value.

The upper computer will receive the current tool-side coordinate value of the robot.

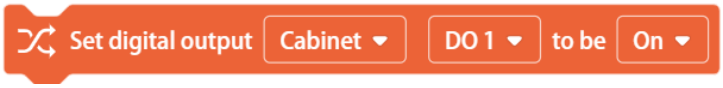


5.1.2.17 Cartesian Space Position () Accessible?

(1) Meaning: Based on the current position, determine whether a given Cartesian position is reachable by Cartesian spatial motion.

5.1.3 IO Instructions

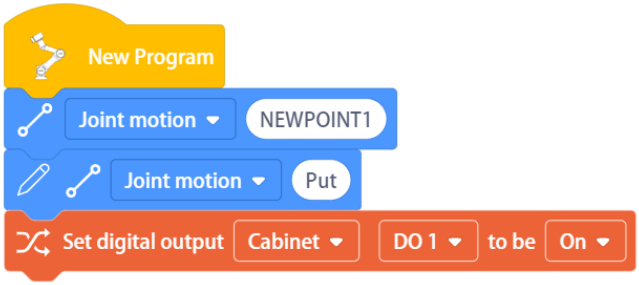
5.1.3.1 Set Digital Output () () to (On/Off)



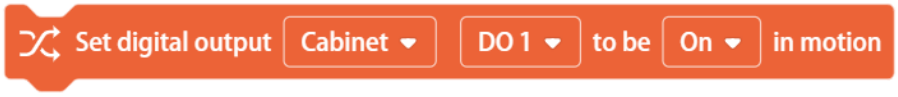
(1) Meaning: Immediate instructions to set DO on or off.

(2) Usage: When this instruction is executed by robot, the corresponding DO signal will switch to the set state.

(3) Example: Release the soft gripper. Release the soft gripper to place the object after moving to the object placement point.



5.1.3.2 Set Digital Output () () to (On/Off) in Motion



(1) Meaning: Non-immediate instructions to set DO on or off.

(2) Usage: By using this instruction after a MoveL instruction (with an arc transition), the robot will switch IO state to the state set by the instruction before the MoveL ends.

5.1.3.3 Waiting for () Seconds until Digital Input () () is (On/Off)

(1) Meaning: Waiting for () seconds until digital input () () is (on/off).

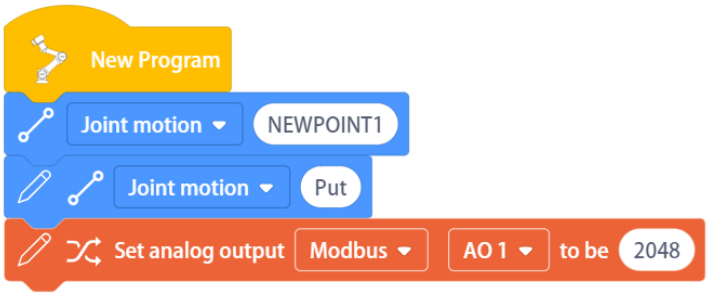
(2) Usage: If the IO state meets the state set by this instruction within () seconds or if the state is not met but the waiting time exceeds () seconds, the program will continue to execute (if the waiting time is set to 0 s, the robot will wait until the condition is met).

5.1.3.4 Set Analog Output () () to ()

(1) Meaning: Immediate instructions to set the analog output value.

(2) Usage: Set the AO value.

(3) Example: Set the value of electrical control cabinet AO1 to 2048 when the robot moves to the standby point



5.1.3.5 Set Analog Output () () to () in Motion

(1) Meaning: Non-immediate instructions to set the AO value.

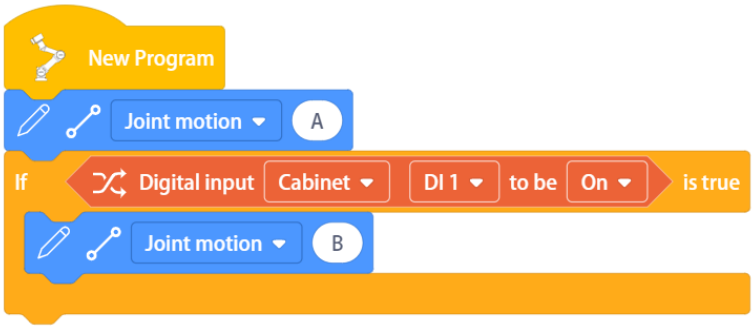
(2) Usage: By using this instruction after a MoveL instruction (with an arc transition), the robot will switch IO state to the state set by the instruction before the MoveL ends.

5.1.3.6 Digital Input () () is (On/Off)

(1) Meaning: Get the state of the corresponding DI.

(2) Usage: It can be used as a judgment condition in the program. It is true when the IO state is the same as the one set on the instruction.

(3) Example: Determine the state of the current digital input. If the current digital input DI1 is on, the robot will move from point A to point B.

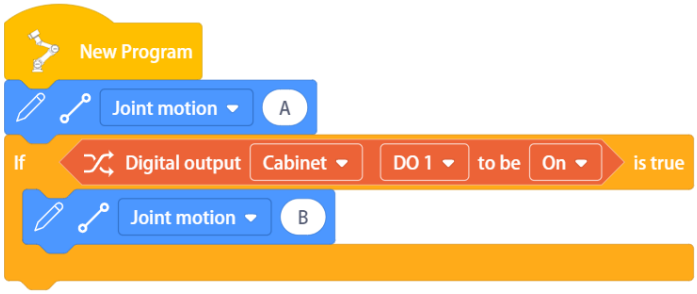


5.1.3.7 Digital Output () () is (On/Off)

(1) Meaning: Get the state of the corresponding DO.

(2) Usage: It can be used as a judgment condition in the program. It is true when the IO state is the same as the one set on the instruction.

(3) Example: Determine the state of the current digital output. If the current digital output DO1 is on, the robot will move from point A to point B.



5.1.3.8 Get Analog Output () ()

(1) Meaning: Get the AO value of current robot.

(2) Usage: It is possible to assign the analog output value to a variable by means of variable assignment.

5.1.3.9 Get Analog Input () ()

(1) Meaning: Get the AI value of current robot.

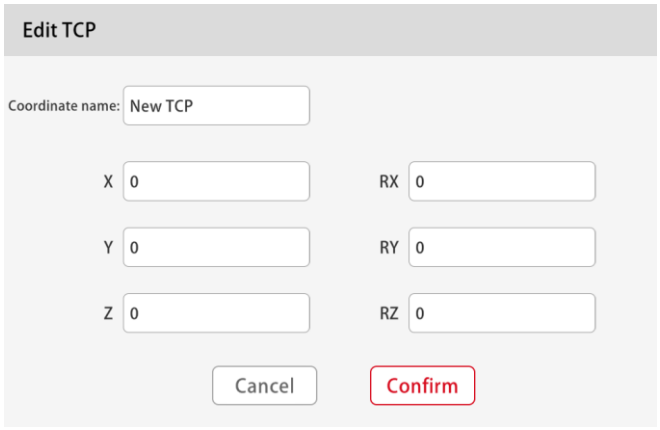
(2) Usage: It is possible to assign the analog output value to a variable by means of variable assignment.

5.1.4 Control Instructions

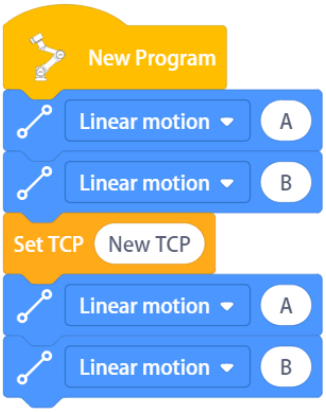
5.1.4.1 Set TCP

(1) Meaning: Set TCP coordinate system.

(2) Usage: Click the instruction to open "Settings" interface, in which you can edit the name of the coordinate system and the coordinate value. This coordinate system is only available in this program. (If not used, the robot uses end flange coordinate system by default.)



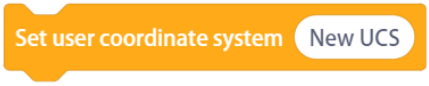
(3) Example: Create a new TCP coordinate system in the program and observe the change of robot end motion after setting.



5.1.4.2 Set TCP ()

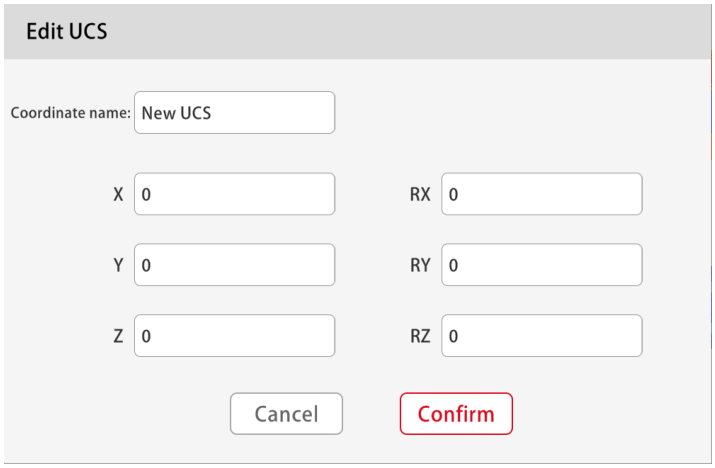
(1) Meaning: Select the set TCP.

5.1.4.3 Set User Coordinate System

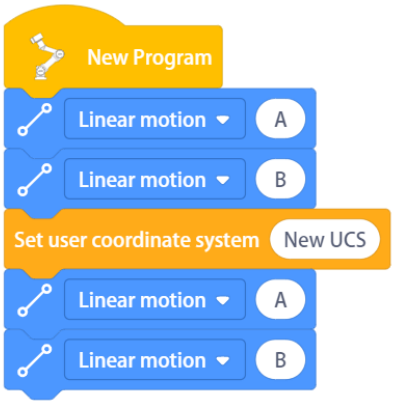


(1) Meaning: Set user coordinate system.

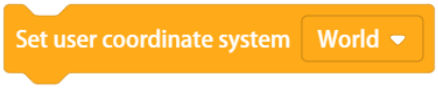
(2) Usage: Click the instruction to open "Settings" interface, in which you can edit the name of the coordinate system and the coordinate value. This coordinate system is only available in this program. (If not used, the robot uses world coordinate system by default).



(3) Example: Create a new user coordinate system in the program and observe the change in the robot end motion after setting.



5.1.4.4 Set User Coordinate System ()



(1) Meaning: Select the set user coordinate system.

5.1.4.5 Set Load () Centroid ()

(1) Meaning: Set the current load and centroid position.

5.1.4.6 Set Collision Sensitivity ()

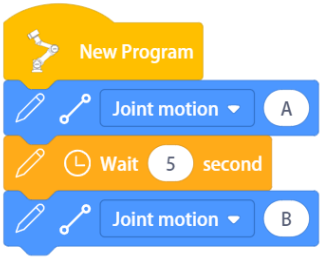
(1) Meaning: Set the current collision sensitivity.

5.1.4.7 Wait for () Seconds

(1) Meaning: The program is paused for () seconds.

Note: If the set waiting time is 0s, the robot will keep waiting.

(2) Example: The robot moves to point A and pauses for 5s, then continues to move.

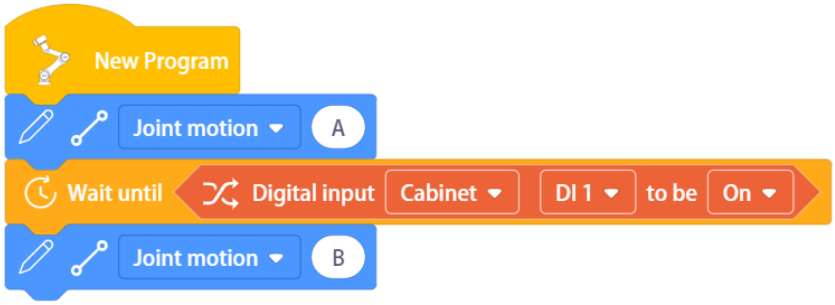


5.1.4.8 Wait Until ()

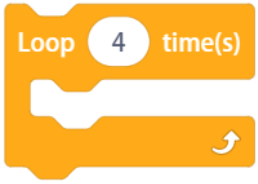
(1) Meaning: Wait until the condition is met, and continue the program.

(2) Usage: You can add a judgment condition in the condition box.

(3) Example: Wait until the digital input DI1 is on, and the robot moves from point A to point B.



5.1.4.9 Repeat for ( ) Times



(1) Meaning: Make the program inside its frame repeat for () times.

(2) Usage: By placing instructions to be repeated () times inside this frame, they will be repeated for () times when the program is running.

5.1.4.10 Keep Repeating

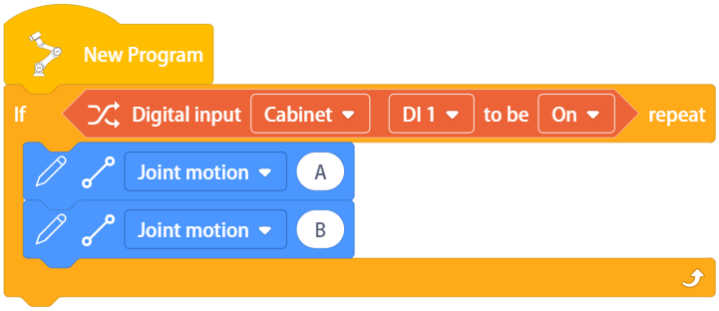
(1) Meaning: Keep the program inside its frame repeating.

(2) Usage: By placing instructions to be repeated inside this frame, they will keep repeating when the program is running.

5.1.4.11 If the () loop is executed

(1) Meaning: If the condition is met, the program in the frame will be executed in a loop.

(2) Example: If the condition is met, the robot will repeat the motion between points A and B.

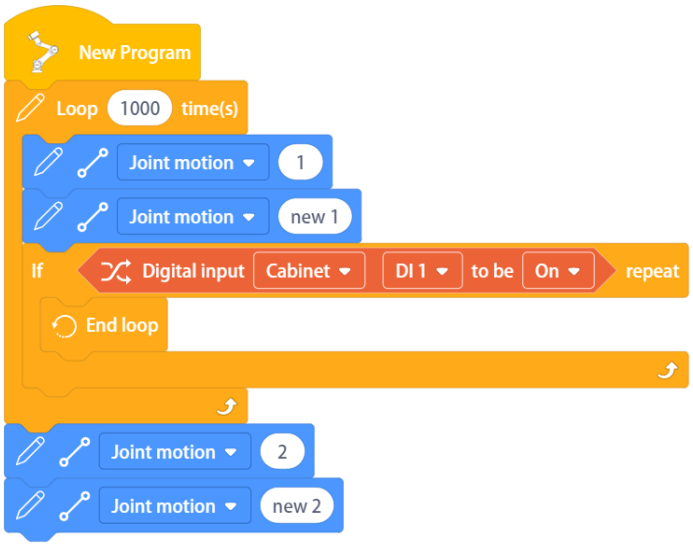


5.1.4.12 End the Loop

(1) Meaning: End the loop in which this instruction is placed.

(2) Usage: Add a judgment condition before this instruction, and immediately end the loop if the condition is met.

(3) Example: If DI1 is on, the program will immediately end the loop and start running directly from "MoveL <2>".

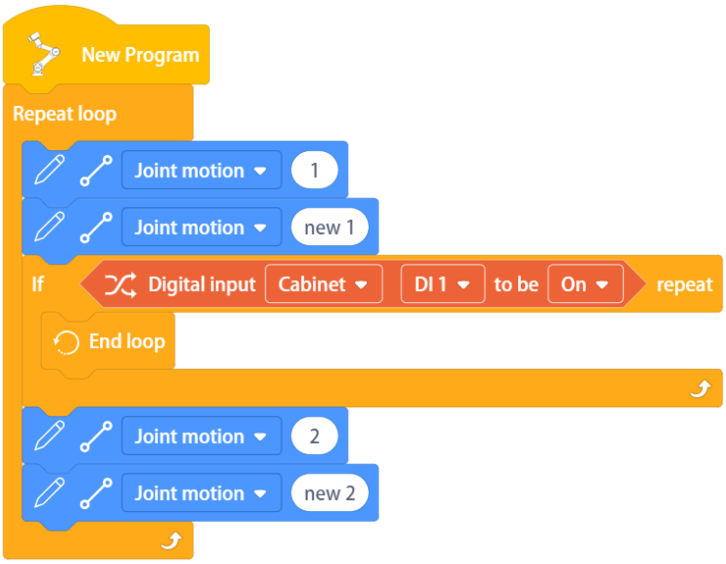


5.1.4.13 Skip this Loop

(1) Meaning: Skip this loop.

(2) Usage: Add a judgment condition before this instruction, and immediately skip the loop if the condition is met.

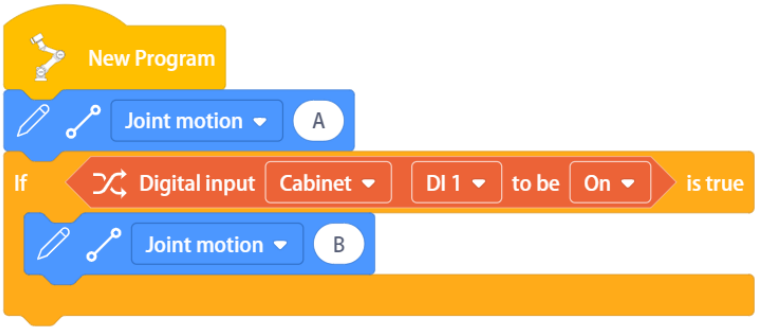
(3) Example: If DI1 is on, the program will not continue "linear motion <2>" . Instead, it will return directly to the beginning of the loop and start running from "joint movement <1>".



5.1.4.14 If () is True

(1) Meaning: If the condition is met, the program inside the frame will be executed.

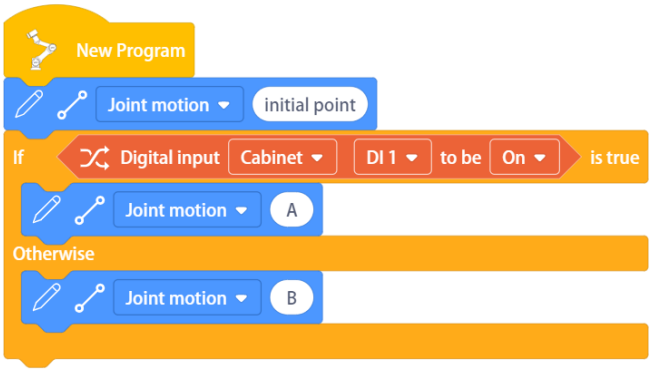
(2) Example: If the current digital input DI1 is on, the robot will move from point A to point B.



5.1.4.15 If () is True, Otherwise

(1) Meaning: If the condition is met, execute the program in upper part of the frame, otherwise execute the program in lower part of the frame.

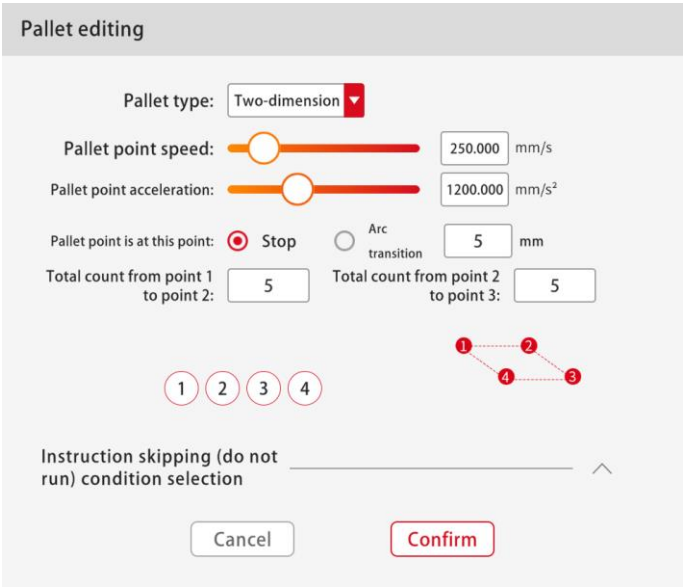
(2) Example: If the current digital input DI1 is on, the robot will move to point A. Otherwise, it will move to point B.



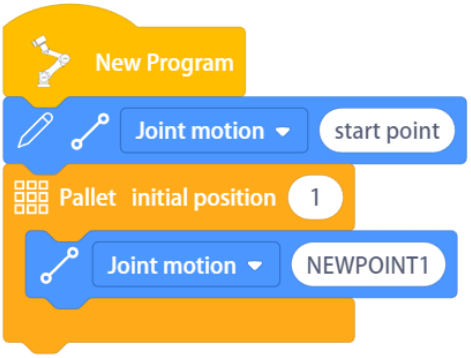
5.1.4.16 Pallet Initial Position ()

(1) Meaning: Set the pallet type, point number and gap number to make the robot execute the pallet program.

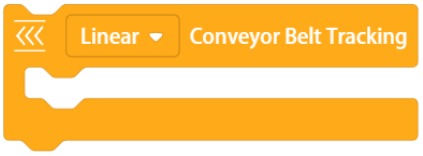
(2) Usage: Click the instruction to open the "Settings" interface, in which you can edit the pallet type, speed and acceleration of this instruction; You can decide whether the robot stops here or performs arc transition, choose the gap number between pallet points, edit the demonstration point of pallet, and choose the stop condition (the program will skip this instruction when the robot meets the state set by the stop condition) (initial position can use variables. If the variable value is less than 0, the program will report an error; If the variable value is more than pallet gap number, the program will execute from the first pallet point).



(3) Example: The robot end moves along a 3X3 square pallet.



5.1.4.17 Conveyor Tracking



(1) Meaning: Make the robot end follow the conveyor belt and make movements in real time.

(2) Usage:

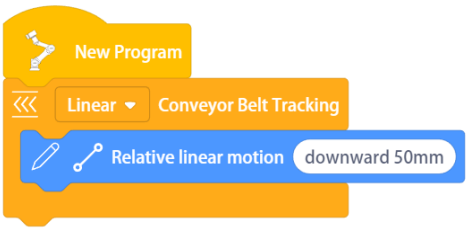
a. Straight line conveyor tracking:

Click the instruction to open the "Settings" interface, in which you can configure the pulse equivalent of the conveyor belt (mm/cnt) and the moving direction of the conveyor belt (you can confirm by demonstration or filling in the direction directly)

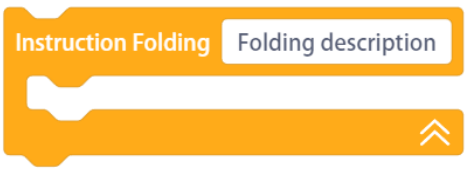
b. Circular conveyor tracking:

Click the instruction to open the "Settings" interface, in which you can configure the pulse equivalent of the conveyor belt (° / cnt), select whether the end position is covariant, and determine the center coordinate system of conveyor belt (you can confirm by demonstrating concyclic three-point position or directly filling in the position)

(3) Example: The robot will move 50mm vertically downward while moving in the conveyor direction with conveyor belt speed.



5.1.4.18 Instruction Folding

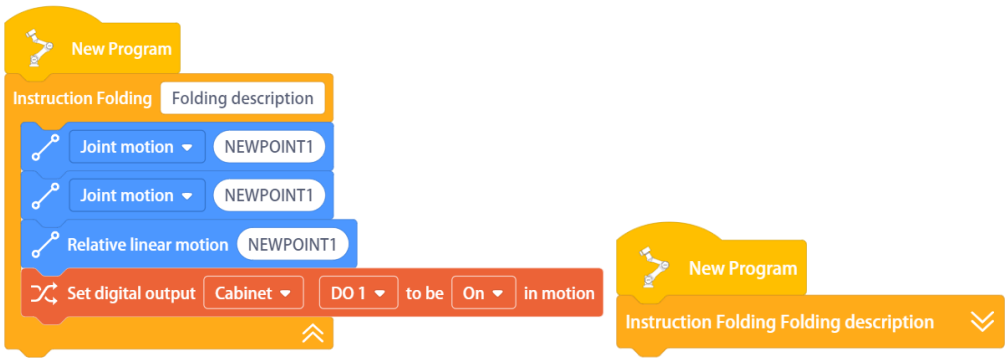


(1) Meaning: Fold the instructions dragged into this instruction into a single instruction to reduce the space.

(2) Usage: You can edit the name of the folded instruction, drag in the instruction to be

folded, and click it to trigger instruction folding.

(3) Example: Fold the following instructions.

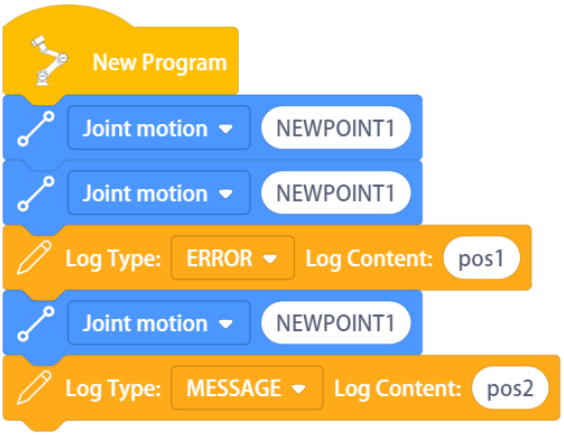


5.1.4.19 Log Type () Log Content ()

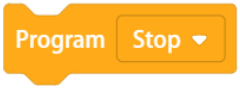
(1) Meaning: User-defined output log content.

(2) Usage: Log types (including message, warning and error) can be selected in the instruction drop-down list. You can input the value at the log content or drag in the string variable, and the output log content can be viewed at the App log information.

(3) Example: Output the log content pos1 and pos2 when it reaches the specified position.



5.1.4.20 Program Stop or Pause



(1) Meaning: Make the specified program pause or stop at a certain location.

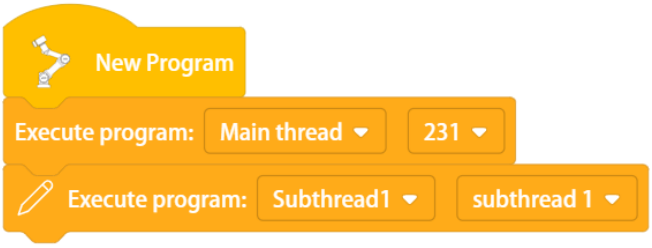
(2) Usage: It is possible to select "pause" or "stop" in the instruction drop-down list, and the instruction can be added anywhere in the program.

5.1.4.21 Execute Program ()

(1) Meaning: Additional programs stored in the robot can be executed.

(2) Usage: The command drop-down list allows you to select another job file in the programming list. The user can specify whether the selected instruction is to be executed in the main program or in a new sub-thread(executed separately).

(3) Example: For example, the first instruction executes the program file named 231 as the main thread, and the second instruction executes the program file named sub-thread 1 as a sub-thread. Sample programs are as follows:

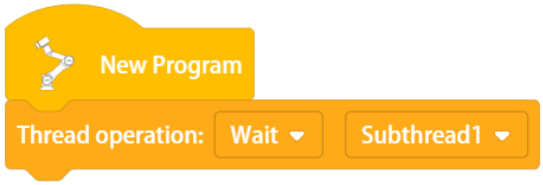


5.1.4.22 Thread Operation () ()

(1) Meaning: Can wait or destroy sub-threads.

(2) Usage: The user can select whether to wait or destroy sub-threads via the instruction drop-down list.

(3) Example: For example, waiting for subthread 1, the sample program is as follows:

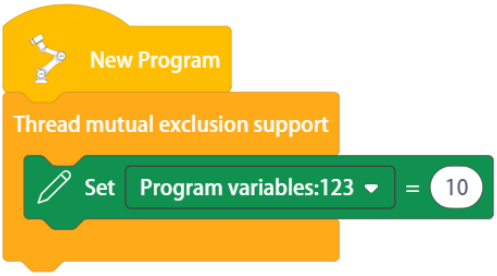


5.1.4.23 Thread Mutual Exclusion Support

(1) Meaning: Critical areas can be created for multiple threads sharing resources.

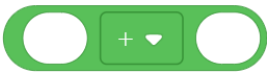
(2) Usage: Put the needed shared resources into critical areas.

(3) Example: For example, if the program variable 123 is placed in the critical area, the sample program is as follows:



5.1.5 Calculation Instructions

5.1.5.1 Operational Symbol ()

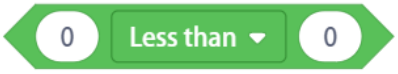


(1) Meaning: Six calculation methods are provided: Add (+), subtract (-), multiply (*), divide (/), find the remainder (%), and find the power (**).

5.1.5.2 Mathematical Function Calculation ()

(1) Meaning: Mathematical function calculation modes are provided, including sine, cosine, tangent, inverse sine, inverse cosine, inverse tangent, natural exponent, natural logarithm, rounding up, rounding down, rounding off, absolute value, and square root.

5.1.5.3 () Comparison Operator ()



(1) Meaning: Comparison operators are provided: <, =, >, ≠, ≤ and ≥

5.1.5.4 () AND ()

(1) Meaning: The conditions at both sides are met.

5.1.5.5 () OR ()

(1) Meaning: The conditions at both sides are met or one of them is met.

5.1.5.6 () XOR ()

(1) Meaning: Only one condition is met.

5.1.5.7 Not ()

(1) Meaning: None of the conditions is met.

5.1.5.8 Position Calculation () Addition (Subtraction/Inverse Transformation/Position Transformation) ()

① **Position Calculation () Addition ()**

(1) Meaning: Calculate the accumulation of two positions.

(2) Usage: You can drag in position variables or demonstration points, and the instruction

return value is the calculation result.

② **Position Calculation () Subtraction ()**

(1) Meaning: Calculate the subtraction of two positions.

(2) Usage: You can drag in position variables or demonstration points, and the instruction return value is the calculation result.

③ **Position Calculation Inversion ()**

(1) Meaning: Calculate the inversion of Cartesian space position.

(2) Usage: You can drag in position variables or demonstration points, and the instruction return value is the calculation result.

④ **Position Calculation () Position Transformation ()**

(1) Meaning: Calculate the positional transformation of the robot.

(2) Usage: You can drag in position variables or demonstration points (representing incremental transformations of the robot initial position and relative initial position respectively), and the instruction return value is the transformation result.

5.1.5.9 Calculate the Position Distance () ()

(1) Meaning: Calculate the distance between two positions.

(2) Usage: You can drag in position variables or demonstration points, and the instruction return value is the calculation result.

5.1.5.10 Interpolation Point () () Coefficient ()

(1) Meaning: The interpolation points between two points are calculated according to the given coefficients, whose range is [0,1].

(2) Usage: You can drag in position variables or demonstration points, and the instruction return value is the calculation result.

5.1.5.11 Plane Transformation XY (YZ/ZX) Base Point () Z (X/Y) Axis Rotation () X (Y/Z) Axis Translation () Y (Z/X) Axis Translation

(1) Meaning: The transformation exists in the XY(YZ/ZX) plane, with the base point first rotated around the Z(X/Y) axis, then translated along the X(Y/Z) axis, and translated along the Y(Z/X) axis.

(2) Usage: The base point can be dragged into position variables, array variables or taught points, and the instruction return value is the transformed position.

5.1.5.12 Kinematic Inverse Solution (Positive Solution)

① Kinematic Inverse Solution　Cartesian Space Values

(1) Meaning: The joint space position corresponding to the input Cartesian space position in the same space is calculated based on the selected reference angle value.

(2) You can drag in position variables or demonstration points, and the instruction return value is the calculation result.

② Kinematic Positive Solution of Joint Angle Values

(1) Meaning: Calculate the Cartesian space position of the input joint angle.

(2) Usage: You can drag in position variables or demonstration points, and the instruction return value is the calculation result.

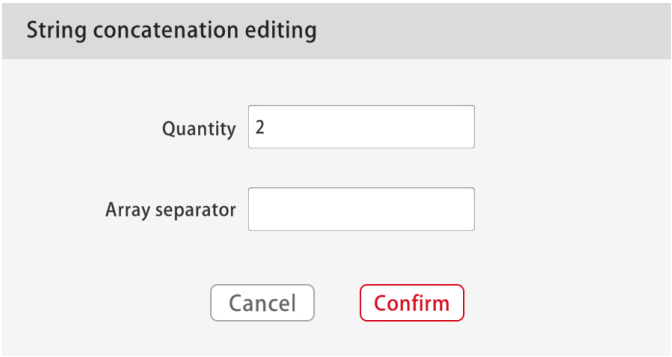
5.1.6 Character Instructions

Note: The escaped characters supported in the string instruction are \\, \', \", \n, \t, \r, corresponding to the backslash symbol, single quote, double quote, line feed, horizontal tab, and carriage return, respectively.

5.1.6.1 String Splicing () ()

(1) Meaning: Splice the dragged-in variables or input strings in a certain order.

(2) Usage: Click this instruction to edit the number of strings to be spliced (1~8) and the desired array separator. It is possible to drag in variables or input strings, and this instruction will return the spliced strings, the value of which is string variable type.



(3) Example:

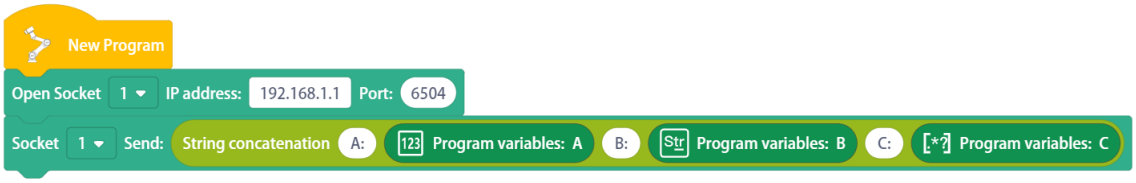
For example, if the variables to be spliced are A=8888, B="Hello", C=[1,2,3,4,5,6], the return result is:

A:8888

B:Hello

C:1,2,3,4,5,6

The sample program is as follows:

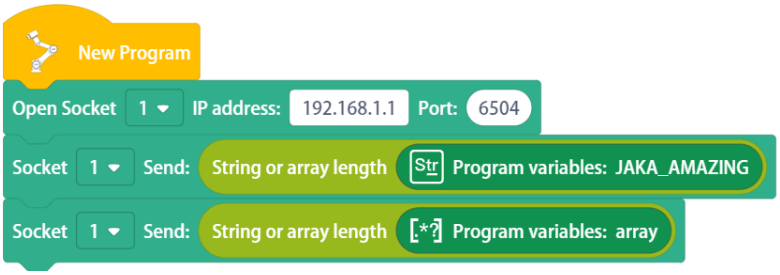


5.1.6.2 Length of String or Array ()

(1) Meaning: Calculate the length of a string or an array and returns the length value.

(2) Usage: You can drag in string variable, array variable, or input string, and the return value is the length.

(3) Example: If the upper unit needs to obtain the length of the string variable "JAKA_AMAZING" (12), and the array variable [1,2,3,4,5,6] (6), the sample process is as follows:

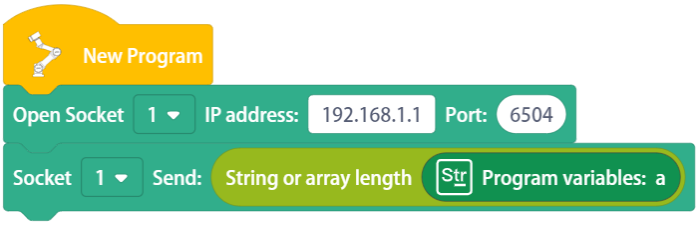


5.1.6.3 String Comparison () ()

(1) Meaning: Compare the size of two strings according to ACSII code.

(2) Usage: You can drag in string variables or input string values str1, str2, if str1=str2, then return 0; if str1str2, then return negative; if str1>str2, then return positive.

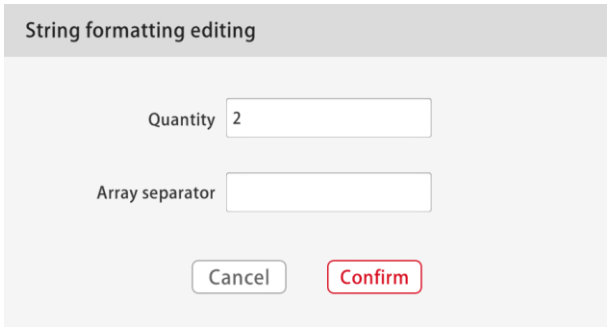
(3) Example: For example, if you need to compare the size of the string variables A and a, the return value should be negative. The sample process is as follows



5.1.6.4 Formatting the Output String Format: () ()

(1) Meaning: Output the specified data as a string in the specified format.

(2) Usage: Click this instruction to edit the number of data to be formatted (1~8) and the desired array separator. It is possible to drag in variables or input values, and this instruction will return the formatted string, the value of which is string variable type.

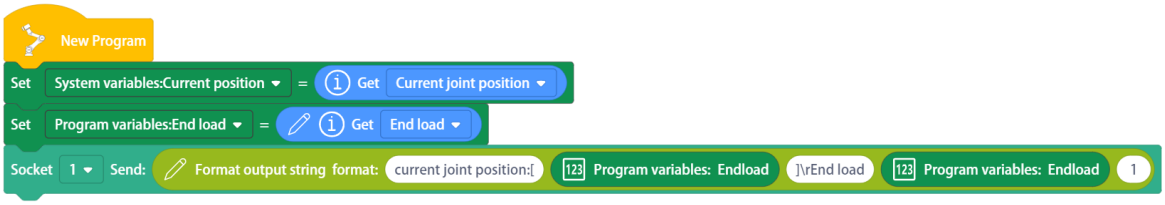


(3) Example: For example, if you need to get the robot information:

Current joint position: [0,90,0,90,180,0]

End load: [0,0,0,0]

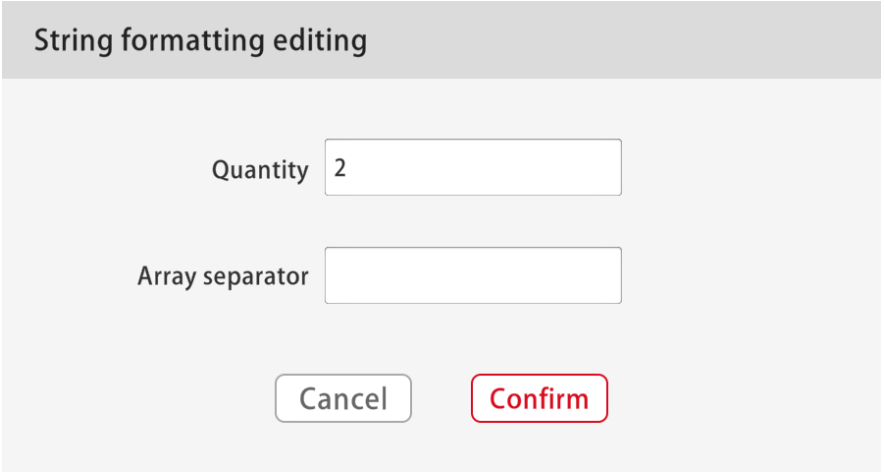
The sample program is as follows:



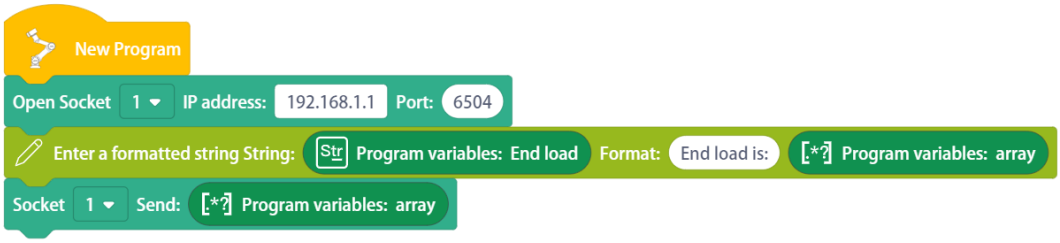
5.1.6.5 Enter the Formatted String String: () Format: () () Results

(1) Meaning: Match a certain format of string and input the matched data into the specified variable and return the result.

(2) Usage: Click this instruction to edit the data number to be formatted (1~8) and the required array separator. You can drag in specified variables or input values. If the formatting is successful, return the number of matched variables, otherwise return -1. Formatted result variables support integer type, floating-point type, and string type variables, or constants.



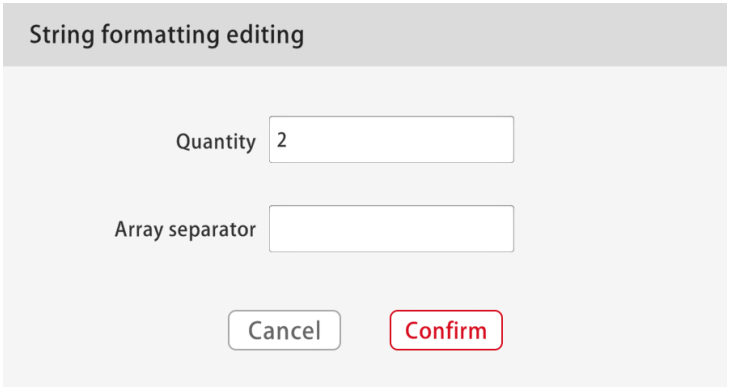
(3) Example: For example, to extract the array [0,0,0] in the string variable "End load is: 0,0,0", the sample program is as follows:



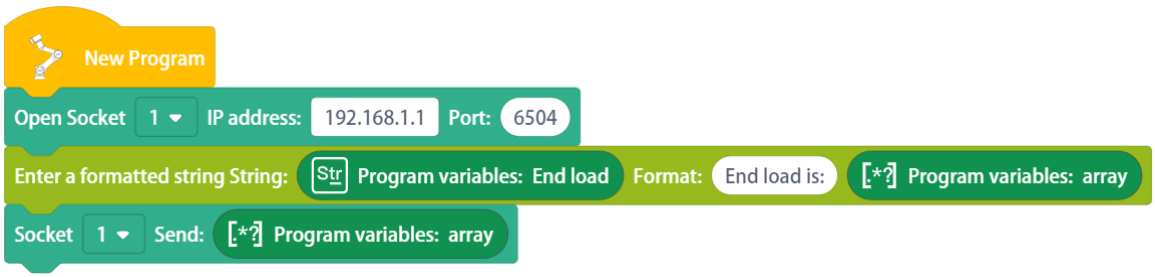
5.1.6.6 Enter the formatted string String: () Format () ()

(1) Meaning: Match a string in a certain format and enter the matched data into the specified variable.

(2) Usage: Click this instruction to edit the data number to be formatted (1~8) and the required array separator. You can drag in specified variables or input values, and the specified variable can be extracted to the desired value. Formatted result variables support integer type, floating-point type, and string type variables, or constants.



(3) Example: For example, to extract the array [0,0,0] in the string variable "End load is : 0,0,0", the sample program is as follows:



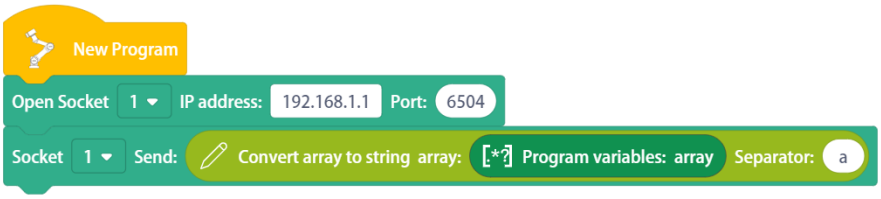
5.1.6.7 Convert Arrays into Strings Arrays: () Separator: ()

(1) Meaning: Convert an array into a string variable in a certain format.

(2) Usage: You can drag in the array type variable to be converted in "Array", and enter the array separator to be converted in "Separator".

(3) Example: For example, to convert the array variable [1,2,3,4,5,6] into "1a2a3a4a5a6",

the sample process is as follows:

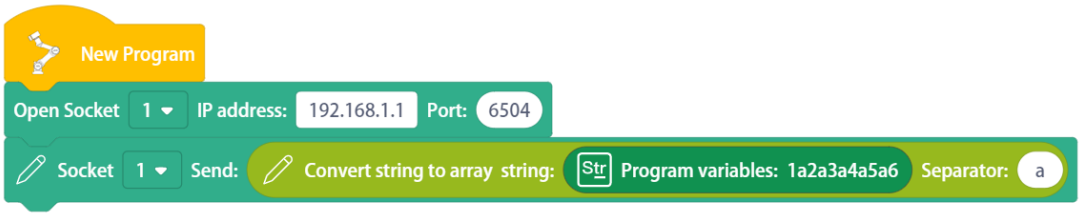


5.1.6.8 String to Array Conversion String: () Separator: ()

(1) Meaning: Convert a string in a certain format into an array variable.

(2) Usage: You can drag in the string variable to be converted into an array type in "String", and enter the array separator to be converted in "Separator".

(3) Example: For example, to convert the string variable "1a2a3a4a5a6" into the array variable [1,2,3,4,5,6], the sample process is as follows:

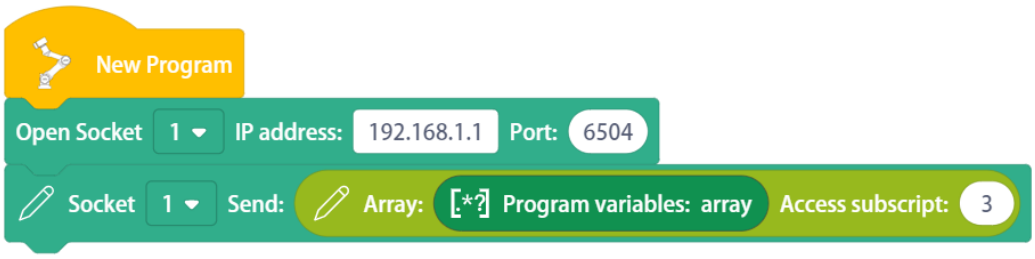


5.1.6.9 Array: () Access the Subscript: ()

(1) Meaning: Access elements in an array variable.

(2) Usage: You can drag in the array variable to be accessed and enter the array subscript index value to be accessed, in which the subscript index starts from 0. The return value of the instruction is the accessed element.

(3) Example: For example, to access the value with subscript 3 in the array variable [1,2,3,4,5,6], i.e. the return value is 4, the sample program is as follows:

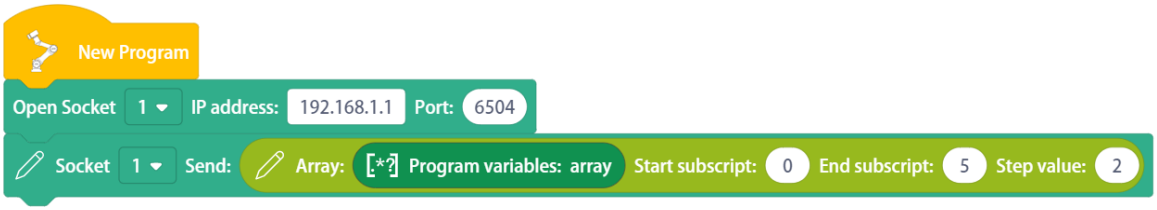


5.1.6.10 Array: () Start Subscript: () End Subscript: () Step Value: ()

(1) Meaning: Access the elements in an array variable according to the required subscript and step value.

(2) Usage: You can drag in the array variable to be accessed and enter the array subscript index value and step value to be accessed, in which the subscript index starts from 0. The return value of the command is the accessed subarray.

(3) Example: For example, to access the values with subscripts 0, 2, 4 in the array variable [1,2,3,4,5,6], i.e. the return value is [1,3,5], the sample program is as follows:

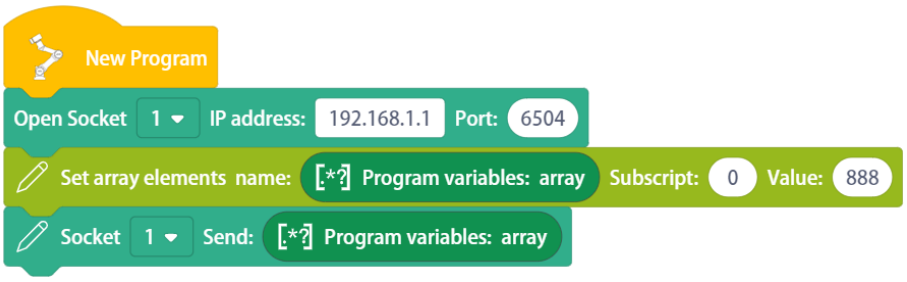


5.1.6.11 Set Array Elements Name: () Subscript: () Values: ()

(1) Meaning: Set the value of an element in the array.

(2) Usage: You can drag in the array variable to be set and enter the array subscript index value to be set, in which the subscript index starts from 0.

(3) Example: Set the value of subscript 0 in the array variable [1,2,3,4,5,6] to 888, i.e. the array variable becomes [888,2,3,4,5,6], the sample program is as follows:



5.1.7 Communication Instructions

5.1.7.1 Open SOCKET () IP address: () Port: ()

(1) Meaning: Create TCP Client and establish communication with TCP server.

(2) Usage: Select the specified SOCKET ID in the drop-down list, enter the IP address and port number of TCP server, and connection between TCP Client (robot) and TCP Server will be established when this instruction is executed.

5.1.7.2 Open SOCKET () IP address: () Port: () Result

(1) Meaning: Create TCP Client and establish communication with TCP server, and return Socket handle.

(2) Usage: Select the specified SOCKET ID in the drop-down list, enter the IP address and port number of TCP Server, and connection between TCP Client (robot) and TCP Server will be established when this instruction is executed; and this instruction will get a return value, which is an integer value greater than 0 if there is a successful connection; or -1 if the connection is not established.

5.1.7.3 Close SOCKET ()

(1) Meaning: Disconnect the specified SOCKET communication.

(2) Usage: Select the specified SOCKET ID in the drop-down list, and the specified SOCKET communication connection will be disconnected when this instruction is executed.

5.1.7.4 SOCKET () Receive Variables: () Waiting Time: () seconds

(1) Meaning: Waiting () seconds until the variable is received. The controller will send a data request string, and then enter the receive waiting state. (array types are not supported)

(2) Usage: Select the specified SOCKET ID from the drop-down list to drag in the variable type to be received. If the variable is received or not received within () seconds, but the waiting time exceeds () seconds, the program will continue to execute (if the waiting time is set to 0s, the robot will wait until the conditions are met), and the variable name containing Unicode characters is not supported.

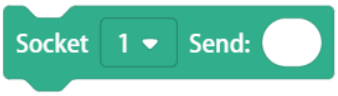
Note: The received variable types include numeric type, string type and array type. The server's data sending format is:

Numeric type: <variable name> <data content>

String type: <Variable Name> <"Data Content">

Array type: <Variable Name> <[Data Content]>

5.1.7.5 SOCKET () Send: ()

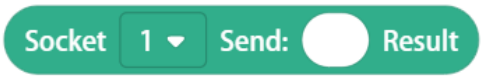


(1) Meaning: Enable the controller to send the variable to TCP Server via SOCKET communication.

(2) Usage: Select the specified SOCKET ID in the drop-down list, drag in the variable or enter the value to execute this instruction, and TCP Server will receive the contents of this variable.

Note: The variable types that support sending include numeric type, string type and array type. There are no special requirements for data format on sending and Unicode characters are not supported.

5.1.7.6 SOCKET () Send: () Result



(1) Meaning: Enable the controller to send the variable to TCP Server via SOCKET communication.

(2) Usage: Select the specified SOCKET ID in the drop-down list, drag in the variable or input value, execute this instruction, and TCP Server will receive the content of this variable; This instruction will get the return value, if the sending is successful, the length of the data sent will be returned; if the sending fails, -1 will be returned.

5.1.7.7 SOCKET() Arrays Received Array Length: () Waiting Time: () seconds

(1) Meaning: The robot stops in () seconds until an array variable of length () is received. The controller sends the data request string, and then enters receive waiting state.

(2) Usage: Select the specified SOCKET ID in the drop-down list. If the variable is received or not received within () seconds but the waiting time exceeds () seconds, the program will continue to execute (if the waiting time is set to 0s, the robot will wait until the condition is met); If the receiving fails or timeout happens, the empty array is returned; If the receiving succeeds, it will be stored in the array.

Note: Server data sending format: [n1,n2,n3...]

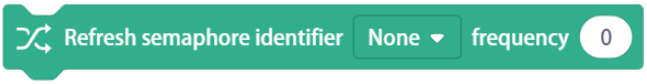
5.1.7.8 SOCKET() Receive Waiting Time: () seconds

(1) Meaning: The robot will stop for () seconds until it receives the data.

(2) Usage: Select the specified SOCKET ID in the drop-down list. If the data is received or not received in () seconds but the waiting time exceeds () seconds, the program will continue to execute (if the waiting time is set to 0s, the robot will wait until the conditions are met); if received normally, the function returns the received string. If the receiving fails or a timeout happens, the empty string is returned.

Note: Since receiving variable is string type, there is no special requirements for server-side data sending format, and Unicode characters are not supported.

5.1.7.9 Refresh Signal Variable Identifier () Frequency ()



(1) Meaning: Refresh signal variables defined in the end IO of "Hardware & Communication".

(2) Usage: Select the signal variables to be refreshed in the instruction drop-down list and enter the frequency to be refreshed in Hz. Refresh frequency: Limited by the bus wideband, the sum of refresh frequency of all signals cannot exceed 125Hz, and the internal system will be automatically

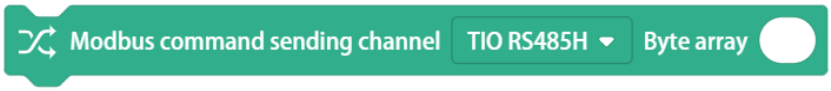
scheduled.

5.1.7.10 Get Signal Variable State Identifier ()

(1) Meaning: Get the value of signal variables.

(2) Usage: Select the desired signal variable in the instruction drop-down list, and the return value is the current value of the signal variable.

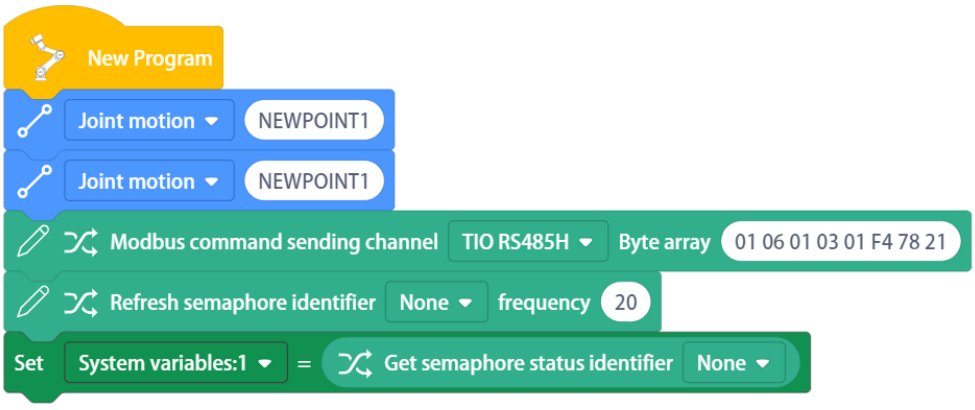
5.1.7.11 Modbus Instruction Issue Channel () Byte Array ()



(1) Meaning: Immediate control instruction issue for TIO external devices.

(2) Usage: Select the TIO channel used in the instruction drop-down list. Users can enter hexadecimal data instructions at the byte array, and is not required to add the check code which can be added by the system automatically.

(3) Example: Set the location of a device to 500 and get the position message after setting. The sample process is as follows:



5.1.8 Subroutine Instructions

5.1.8.1 Subroutines

①Instruction Type Subroutine

(1) Meaning: An instruction sequence is formed into a relatively independent program segment, which is added to the "Program" to execute the instruction sequence in this program segment, and return to the original position to resume program after the execution is completed. This relatively independent program segment is subroutine.

(2) Usage: Add a subroutine of instruction editing type, edit a separate program segment in the subroutine and save it, then you can add this subroutine (reusable) at any time into the main program.

②Script Type Subroutine

(1) Meaning: The subroutine content is the script syntax written in the prescribed format.

(2) Usage: Add a subroutine of script editing type, edit the script syntax in the prescribed format in the subroutine and save it, then you can add this subroutine (reusable) at any time into the main program.

5.1.8.2 Add or delete submodule

① Add submodule

Click the plus button  on the right of the subroutine column title, select the type of subroutine to be added, and click OK. Once created, the programming area will automatically enter the subroutine.

③ Delete submodule

Click the minus button  on the right of the subroutine column title, click the trash icon on the right of the subroutine to delete the submodule, and click OK.

5.1.8.3 Edit subroutine

In the subroutine command column or in the program, click the subroutine command box to be edited to enter subroutine editing interface. After completing subroutine editing, it is required to save the main program again to ensure that the subroutine has been modified correctly in the main program.

5.1.9 Variable Instructions

5.1.9.1 Add Variables

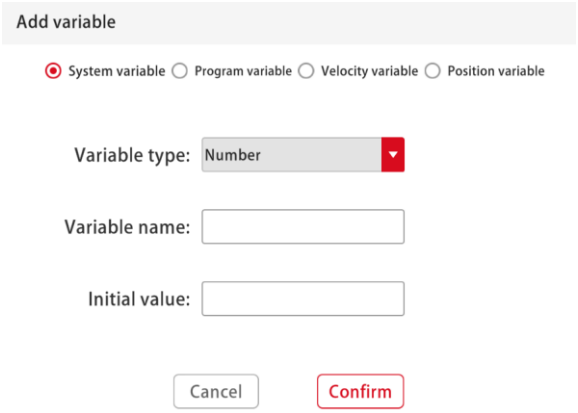
Click the plus button on the right of the variable column title, select the variable type to be added, click OK to enter variable setting interface, fill in the required parameters, and click OK. There are four types of variables to be added: system variables, program variables, speed variables and position variables.

5.1.9.2 System Variables

(1) Meaning: System variables.

(2) Usage: It can be created and used in any program when programming or at system settings, with numeric type supported.

System variables can be created in two ways, either in the "System Variable" of the setting interface or by clicking the "Plus" sign in the "Program Control Variables" instruction column. The system variable exists in the electrical control cabinet and will not be cleared or disappear when the robot is turned off or the program is deleted, and can be called from any program. The system variable type is numeric type.



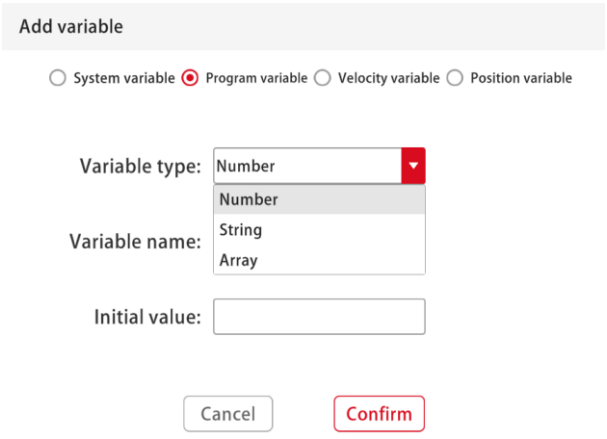
5.1.9.3 Program Variables

(1) Meaning: Program variables.

(2) Usage: It can be created and used when programming, and cannot be used in other programs, with numeric, array, and string type supported.

Program variables are created for use in the currently opened program. This type of variable follows the current program and can only be called in the currently opened program. If the program is deleted, the program variables created in this program will be deleted as well. There are three types of program variables, which are numeric, string and array types.

The program variable names support Chinese format, but it is important to note that garbled characters may exist in Chinese format variable names due to encoding problems when communicating with third parties.



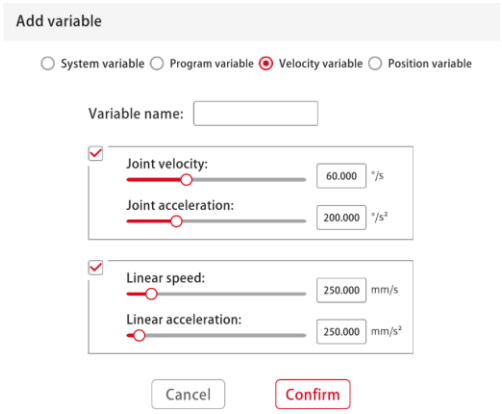
5.1.9.4 Speed Variables

(1) Meaning: Speed variables.

(2) Usage: It can be created and used when programming and cannot be used in other programs, and can be used in the "Set Global Speed" instruction.

Speed variables are special variables that can be on or off in the program by calling the global speed instruction in motion instruction. The robot movement speed in the program can be controlled based on actual working condition. When the speed variable is called by the global speed instruction, the movement speed after this instruction will follow the speed selected in the current speed variable, and the speed settings in the movement instruction will not take effect.

Note: Speed variables can be sent via socket instruction, including joint speed, joint acceleration, linear speed, and linear acceleration.

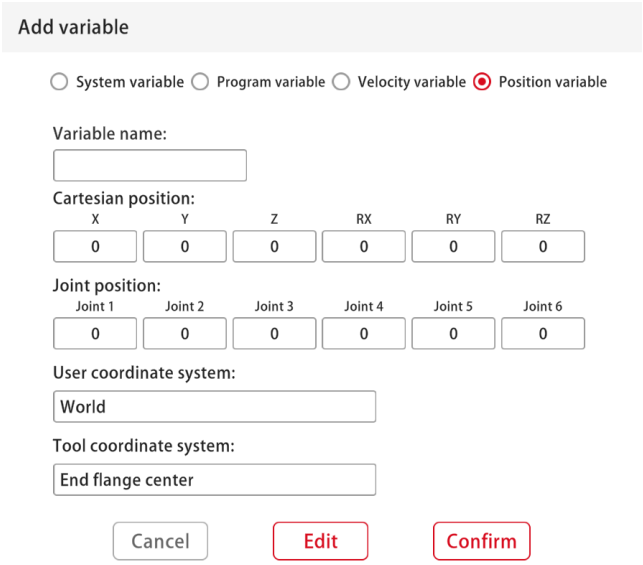


5.1.9.4 Position Variables

(1) Meaning: Position variables.

(2) Usage: It can be created and used when programming and cannot be used in other programs, and can be used in movement instructions or position calculation instructions.

Position variables are specific variables which are applicable to movement instructions and cannot be called in other instructions. Position variables can set the robot Cartesian and joint positions, either by clicking the text box for direct input or by clicking "Edit" to enter robot manual operation interface for positioning the robot. Pay special attention to user coordinate system and tool coordinate system because safety hazard may exist due to potential robot point offset caused by the difference between set coordinate system and actual situation. When a position variable is placed in movement instructions, the robot will make movements according to the selected point information based on the type of motion.



5.1.9.5 Set Variables () Equal to ()

(1) Meaning: Assign a value to each variable.

(2) Usage: You can enter fixed values, variables, values obtained by SOCKET in the input box to assign the data to variables.

Note: To assign values to array variables as a whole, you need to add [] before and after the specific value.

To assign a value to a string, you need to add "" (quotation marks in English) before and after the specific content.

5.1.9.6 Delete Variables

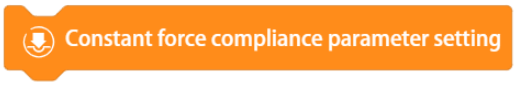
Click "Minus" button on the right of the variable column title, click the trash icon on the right of the variable to delete the variable, and click OK.

5.1.9.7 Edit Variable Parameters

Click the variable box to be edited in the variable instruction column or in the program to enter variable editing interface. After completing variable editing, it is required to save the main program again to ensure that the variables have been modified correctly in the main program.

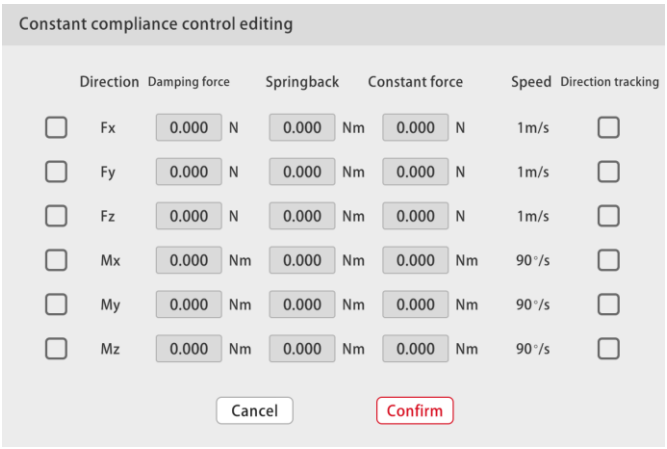
5.1.10 Extended Instructions

5.1.10.1 Constant force compliance parameter settings



(1) Meaning: Set the compliance parameter in constant force mode.

(2) Usage: You can select the direction in which force control sensing function is enabled to set the amount of damping force in each direction (During program execution, the greater the external environment stiffness, the greater the damping force is required; In drag mode, the damping force is recommended to be greater than 10N, M is greater than 0.2Nm, and the setting value cannot be 0), the rebound force (which allows the robot to return to the preset trajectory. The greater the setting value, the harder the robot deviates from the preset trajectory), the constant force (to ensure that the contact force between the robot end and the external environment is within the set constant force value range), and direction tracking function (vertical tracking).



5.1.10.2 Enable Constant Force Softness Control Initialize/Not Initialize

(1) Meaning: Enable constant force mode and select whether to initialize it or not.

(2) Usage:

Initialize: Compensation for sensor bias, load, etc. (to ensure no external contact with the sensor when entering this mode);

Not initialize: Take the previous compensation value.

5.1.10.3 Disable Constant Force Softness Control

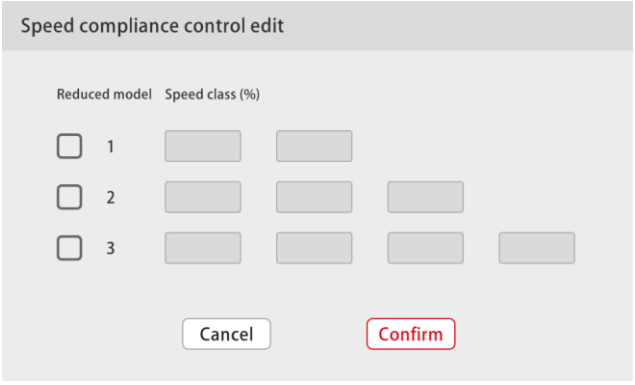
(1) Meaning: Exit constant force mode.

5.1.10.4 Speed Softness Parameter Settings

(1) Meaning: Set softness parameters in speed mode.

(2) Usage: Configure the deceleration level step, with up to three deceleration levels.

When the force at the robot end is greater than the set value of the control force, the robot will decelerate until the value detected by sensor is less than the set control force.



5.1.10.5 Enable Speed Softness Control Initialize/Not Initialize

(1) Meaning: Enable speed mode and select whether to initialize it.

(2) Usage:

Initialize: Compensation for sensor bias, load, etc. (to ensure no external contact with the sensor when entering this mode);

Not initialize: Take the previous compensation value.

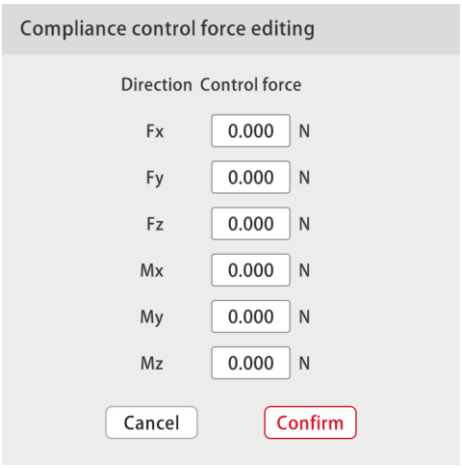
5.1.10.6 Unenable Speed Softness Control

(1) Meaning: Exit speed mode.

5.1.10.7 Softness Control Force

(1) Meaning: Set the direction and magnitude of control force.

(2) Usage: It is used to set the control force in speed softness control mode.



5.1.10.8 Motion Termination Conditions

(1) Meaning: Set conditions for early motion termination.

(2) Usage: Select the direction to be monitored and set the upper limit value or lower limit value; When the external force value is lower than the lower limit value or greater than the upper limit, the motion termination condition will be triggered. This instruction monitors the next movement instruction, and if the motion termination condition is triggered, the robot will immediately move from its current position to the set position of the next movement instruction. It is important to note that all parameters must be set to 0 in "Constant Force Softness Parameter Settings" without selecting any direction.

5.1.10.9 Set Force Control Coordinate System Tool Coordinate System/World Coordinate System

(1) Meaning: Set the coordinate system in force control mode (it is required to ensure that the direction of the coordinate system is consistent with the direction of the sensor).

(2) Usage: Select tool coordinate system, and the robot end position will be described in the system. Select world coordinate system, and the robot end position will be described in the robot base coordinate system.

5.2 Screen Adjustment

5.2.1 Zoom in

Click "Plus" button in the lower right corner of the interface to enlarge the programming area.

5.2.2 Zoom out

Click "Minus" button in the lower right corner of the interface to narrow the programming

area.

5.2.3 Recovery

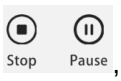
Click "Restore" button  in the lower right corner of the interface to restore the programming area to its default size.

5.3 Program Operation

5.3.1 Program Running

Click "Run" 

, and the robot will operate according to the currently loaded program.

During program runtime, the "Run" button is replaced with "Pause" and "Stop" buttons 

, which are used to pause the program and terminate it, respectively.

5.3.2 Program Operating Rate

Click "Speed" 

, and the speed bar will appear below the speed button. Drag the speed bar to adjust the current operating speed; You can also click the number below the speed bar to enter the percentage of operating speed directly.

5.3.3 New Program

Click "Add Program" button  to create a program, and click the name at the top of the program to change the program name.

5.3.4 Program Saving

Click "Save" button  to save the current program as the latest version.

5.3.5 Save as Program

Click "Save as"  to enter the name of file to be saved, and click OK to save the current program.

5.3.6 Open the Program

Click program file list  to list all the programs saved in the electrical control cabinet, and programs can be sorted in three ways: name, creation time, and file size.

5.3.6.1 Open the Program

(1) Meaning: Open and load the saved program.

(2) Usage: Click the program name to be opened to load it.

5.3.6.2 Import the Program

(1) Meaning: Import JAKA robot program exported from the device via App into the currently connected controller.

(2) Usage: Click "Import" button  in the upper right corner of the program file list window to select the device path in which the program zip is saved; Find the program zip to be imported under

the path and select it; Click OK to complete.

Note: If the source program is exported after backup, it is required to be deleted to ensure successful import; After the import is completed successfully, the backup file will be directly displayed as the source file instead of the backup file.

5.3.6.3 Export the Program

(1) Meaning: Export the program saved in the controller to the current device.

(2) Usage: Click "Export" button  in the upper right corner of the program file list window to select the program to be exported; Click OK to select the path address of the file to be exported, and click OK to complete. Up to 5 operating programs can be exported simultaneously.

5.3.6.4 Delete the Program

(1) Meaning: Delete the robot program.

(2) Usage: Click "Delete" button  in the upper right corner of the program file list window to select the program to be deleted, and click OK. Up to 5 operating programs can be deleted simultaneously.

5.3.6.5 Share the Program

(1) Meaning: Copy the programs in the current controller to other JAKA Zu robot controllers under the same subnet via network.

(2) Usage: Click "Share" button  in the upper right corner of the program file list window to select the program to be shared; Click OK to select robot in the pop-up window, and click OK.

5.3.7 Advanced Program Operation

Click "Advanced Operation", and "Select All", "Copy", "Delete", "Cancel" buttons will appear at the top of the program editing area.

5.3.7.1 Select All

(1) Meaning: Select all instructions in the current editing area.

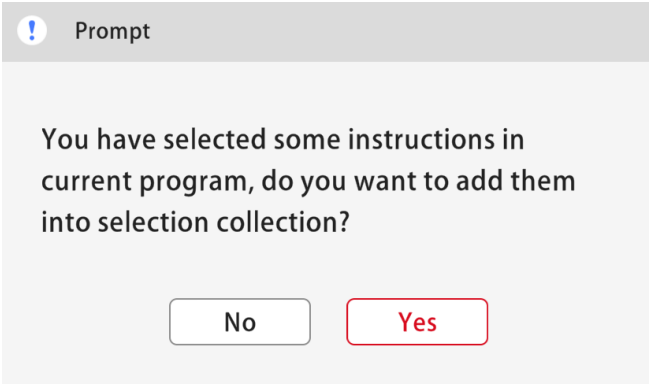
(2) Usage: Click to select all.

5.3.7.2 Copy

(1) Meaning: Copy the selected instructions according to the selected order and paste them into the current editing area.

(2) Usage: Select the instructions to be copied in the desired order and click "Copy".

Note: If you want to copy across programs, open the program or subroutine to be pasted after selecting the instructions to be copied, and a pop-up box will appear as shown in the figure. Click OK. After entering the program, click "Copy Collection" to execute the cross-program copy instruction.



5.3.7.3 Delete

(1) Meaning: Delete the selected instruction.

(2) Usage: Select the instruction to be deleted and click "Delete".

5.3.7.4 Cancel

(1) Meaning: Exit the advanced operation.

(2) Usage: Click to cancel.

5.3.8 Program Single-step Debug

(1) Meaning: Execute the program instruction in single-step.

(2) Usage: Click "Single-step Debug" button, the robot will start running the program, and a red arrow will appear on the left of the instructions to indicate which instruction is currently running. Click "Next", the robot will execute the next instruction, and then wait for users clicking "Next" again. During the debugging process, click "Stop Debugging" or "Stop" to stop the program.

5.3.9 Program Lock

After opening the program lock, the program will not be modified. If you need to modify the program, it is required to close the program lock first.

5.3.10 Variable Observation

Users can observe the variable values in the currently running job program in real time on the App, including four types of variables, i.e. system variables, program variables, speed variables, and position variables.

You can self-define the variables to be observed by the setting button shown in the figure, and add them to the observed variables by clicking the variables under "Variables to be observed", as shown in Figures 5-2 and 5-3.

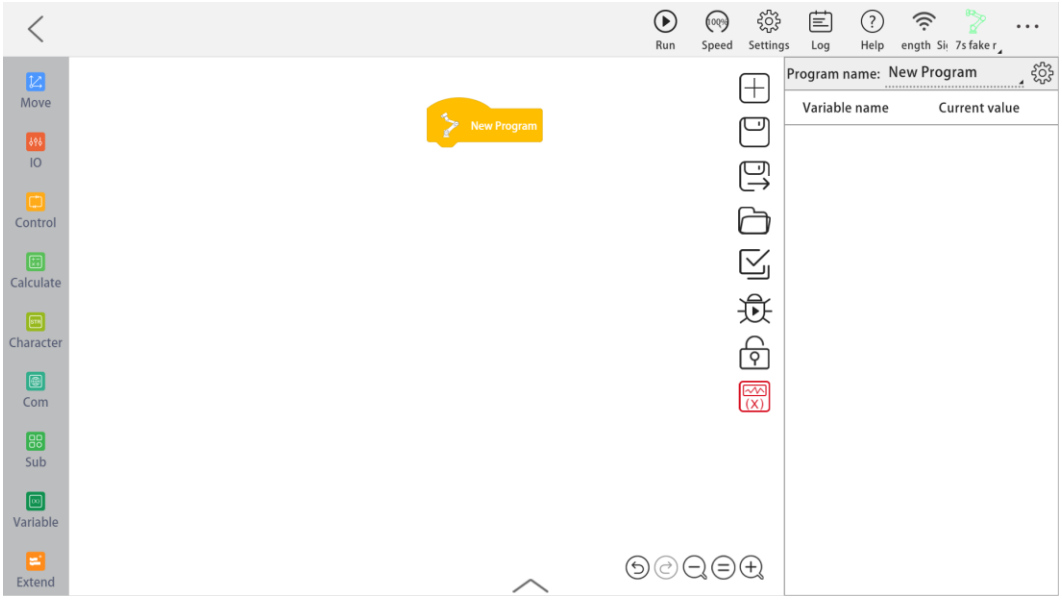


Figure 5-2 Schematic Diagram of Variable Observation

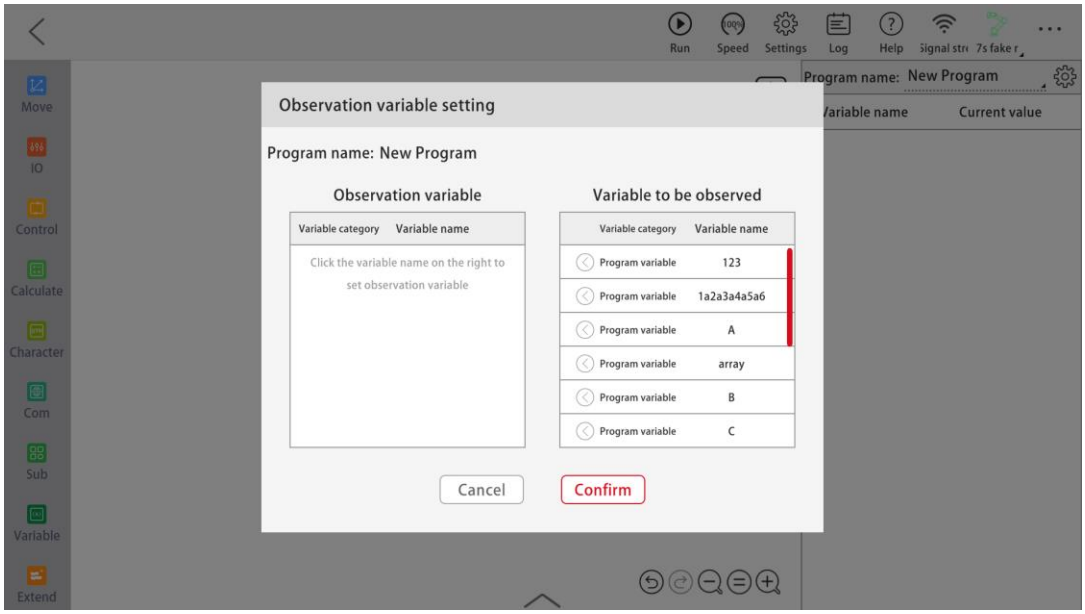


Figure 5-3 Schematic Diagram of Observed Variable Settings

5.4 How to Write a Program

5.4.1 Program Execution Logic

The instruction box that displays the program name is the program header, and all instructions in series with the header form the program body. When the robot executes a program, it will execute sequentially from the program header.

5.4.2 How to Use Instructions

JAKA robot is programmed graphically, so that users can directly drag the required instructions from the instruction column to the appropriate location, and edit the parameters in each instruction to write the program.

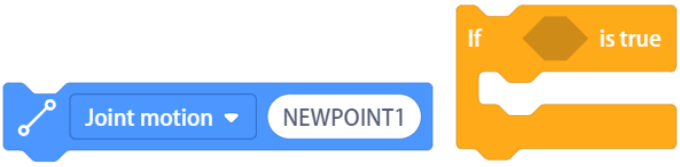
At the same time, users can initially determine the instruction role and location in the program via the

shape of instruction box, facilitating comprehension of logical relationship between instructions.

5.4.2.1 Action Instructions

Instructions with instruction boxes shaped like "MoveJ or MoveL " and "if () is true" are called action instructions. When such instructions are executed, the robot or controller will make corresponding movements.

Action instructions can be directly linked together to form a completion program.



5.4.2.2 Judgment Instructions

Instructions with instruction boxes shaped like "comparison instruction" or "digital input () () is (on/off)" are called judgment instructions. Judgment instructions are used as judgment conditions, and are placed in judgment boxes comprising such instructions as "if () is true", "if () is true, otherwise", "wait until ()", etc.

5.4.2.3 Data Instructions

Command boxes with the shape similar to that of "Get Analog Input () ()", operational symbol instructions and variables, are called data instructions.

Data instructions are generally used to obtain or store data, and is placed in data boxes with the shape similar to data instructions.

Appendix

Appendix I Data Types of Robot Parameters

1 Get the current joint position:

Data type: Arrays

Length: 6

Meaning: The six elements in the array represent the angle values (in °) of joint one to six respectively.

2 Get the current tool-side center position:

Data type: Arrays

Length: 6

Meaning: The six elements in the array represent the spatial position of the origin of the currently used tool coordinate system under the current user coordinate system. Elements 0-5 represent X, Y, Z (in mm), RX, RY and RZ (in °) respectively.

3 Get the current flange center position:

Data type: Arrays

Length: 6

Meaning: The spatial position of the robot end flange center in the current user coordinate system. Elements 0-5 represent X, Y, Z (in mm), RX, RY and RZ (in °) respectively.

4 Get current end load:

Data type: Arrays

Length: 4

Meaning: The stored robot end load information. Elements 0-3 represent the load mass (in kg), X, Y, and Z distances (in mm) of the load centroid relative to the flange center respectively.

5 Get current end force:

Data type: Arrays

Length: 6

Meaning: Get the net torque value (in N.m) of the current end torque sensor after compensated by the sensor end load.

6 Get current sensitivity:

Data type: Digital

Meaning: The collision sensitivity of the current robot settings.

7.System time:

Data type: Digital

Meaning: Get the current system time.

Appendix II Modbus IO Address Table

ID	Type	Name	Data Type	Function Code	Register Type
8	General digital input	DO1	BOOL	02	Discrete inputs are readable but not writable
9		DO2			
10		DO3			
⋮		⋮			
133		DO125			
134		DO126			
135		DO128			
40	General digital output	DI1	BOOL	01/05/15	Coil state
41		DI2			
42		DI3			
⋮		⋮			
165		DI125			
166		DI126			
167		DI128			
96	Analog Inputs	AO01	UINT16	04	Input registers are readable but not writable
97		AO02			
98		AO03			
99		AO04			
⋮		⋮			
109		AO13			
110		AO14			
111		AO15			
112		AO16	INT16		
113		AO17			
114		AO18			
⋮		⋮			
125		AO29			
126		AO30			
127		AO31			
128		AO32	FLOAT32 (big end display)		
129					
130		AO33			
131					
132		AO34			
133					
⋮		⋮			
⋮					
186		AO62			
187					
188		AO63			
189					
190		AO64			
191					

100	Analog Outputs	AI01	UINT16	03/06	Keep registers readable and writable
101		AI02			
102		AI03			
103		AI04			
104		AI05			
⋮		⋮			
111		AI11			
112		AI12			
113		AI13			
114		AI14			
115		AI15			
116		AI16	INT16		
117		AI17			
118		AI18			
119		AI19			
120		AI20			
⋮		⋮			
127		AI27			
128		AI28			
129		AI29			
130		AI30			
131		AI31			
132		AI32	FLOAT32 (big end display)		
133					
134		AI33			
135					
136		AI34			
137					
138		AI35			
139					
140		AI36			
141					
⋮		⋮			
⋮					
186		AI60			
187					
188		AI61			
189					
190		AI62			
191					
192		AI63			
193					
194		AI64			
195					

ID	Type	Name	Data Type	Function Code	Description	Unit	Register Type
300	Machine data	Servo version number	INT32	04	/	/	Input registers are readable but not writable
302		Robot serial number					
304		Joint 1 voltage Joint 2 voltage Joint 3 voltage Joint 4 voltage Joint 5 voltage Joint 6 voltage	INT32		Voltages of each joint	V	
316		Joint 1 temperature Joint 2 temperature Joint 3 temperature Joint 4 temperature Joint 5 temperature Joint 6 temperature			Temperatures of each joint	°C	
328		Joint 1 Servo error code Joint 2 servo error code Joint 3 servo error code Joint 4 servo error code Joint 5 servo error code Joint 6 servo error code	INT32		Servo serial number of each joint	/	
340		Joint 1 error state Joint 2 error state Joint 3 error state Joint 4 error state Joint 5 error state Joint 6 error state	UINT16		Current servo error state 0 means no error 1 means error		
346		Joint 1 enable state Joint 2 enable state Joint 3 enable state Joint 4 enable state Joint 5 enable state Joint 6 enable state			Current servo enable state 0 means not enabled 1 means enabled		
352		Joint 1 collision state Joint 2 collision state Joint 3 collision state Joint 4 collision state Joint 5 collision state Joint 6 collision state			Current servo collision detection state 0 means no collision 1 means collision		
358		Joint 1 current Joint 2 current Joint 3 current Joint 4 current Joint 5 current Joint 6 current	Float32		Current of each joint	A	
370		Sensor torque x Sensor torque y Sensor torque z			Torque of each joint	N	
376		Sensor torque rx Sensor torque ry Sensor torque rz				Nm	
382		Joint 1 position Joint 2 position			Position of each joint	°	
386		Joint 3 position					

388		Joint 4 position	Float32	04	Speed of each joint	°/s	Input registers are readable but not writable
390		Joint 5 position					
392		Joint 6 position					
394		Joint 1 speed					
396		Joint 2 speed					
398		Joint 3 speed					
400		Joint 4 speed					
402		Joint 5 speed					
404		Joint 6 speed					
406		TCP position X			TCP	mm	
408		TCP position Y					
410		TCP position Z					
412		TCP position RX				°	
414		TCP position RY					
416		TCP position RZ					
418		TCP speed X			TCP speed	mm/s	
420		TCP speed Y					
422		TCP Speed Z					
424		TCP Speed RX				°/s	
426		TCP Speed RY					
428		TCP Speed RZ					
430		TCP_OFFSET_X			Tool coordinate system	mm	
432		TCP_OFFSET_Y					
434		TCP_OFFSET_Z					
436		TCP_OFFSET_RX				°	
438		TCP_OFFSET_RY					
440		TCP_OFFSET_RZ					
442		BASE_OFFSET_X			User coordinate system	mm	
444		BASE_OFFSET_Y					
446		BASE_OFFSET_Z					
448		BASE_OFFSET_RX				°	
450		BASE_OFFSET_RY					
452		BASE_OFFSET_RZ					
454		PROTECTIVE_STOP	UINT16		Robot collision: 1 No robot collision: 0	/	
455		EMERGENCY_STOP			Emergency stop		
456		POWER_ON			Power up		
457		ROBOT_ENABLE			Enable		
458		ON_SOFT_LIMIT			Soft limit		
459		INPOS			Reaching the target position		
460		Move mode			Servo position mode: 4 Admittance control mode: 2 Drag mode 1 Other modes (Jog and other operations): 0		
461		Reduction mode level			Reduction mode level: Level 1 reduction: 1 Level 2 reduction: 2 Protective stop: 3		
462		Speed magnification	FLOAT32		Speed magnification		
464		MOTION_ERRCODE	INT32		Error code		
466		CAB_TEMPERATURE	FLOAT32		Electrical control cabinet temperature		
468		CAB_AVERAGEPOWER			Electrical control cabinet power		
470		CAB_AVERAGECURRENT			Electrical control cabinet current		
472		UHI_PULES	FLOAT32		Conveyor belt pulse		

474		UHI_SPEED			Conveyor belt movement speed		
476		UHI_DIR	UINT16		Conveyor belt movement direction		
477		UHI_ORIGIN_PULES	INT32		Conveyor belt original pulse		
479		Retention	UINT16		/		

Appendix III Profinet IO Address Table

Transmission type R->P (Robot->PLC)										
Bit	0	1	2~7	8~15	16	17	18	19~23	24~31	Unit Modules
0	Robot serial number (int32)									Robot state, safety settings 1_R->P_Robot_Safety 32+4 Bytes
32	Servo version number (int32)									
64	CAB_AVERAGECURRENT (float) [A]									
96	CAB_AVERAGEPOWER (float) [W]									
128	CAB_TEMPERATURE (float) [℃]									
160	Power state	Enable state	Retention	Retention						
192	MOTION_ERRCODE (int32)									
224	Move mode (uint8)			Reduction mode level	Emergency Stop	Protective stop	Soft limit state	Retention	Retention	
256	Reserved (int) 4 bytes									
288	Joint 1 voltage (float) [V]									Joint parameters 2_R->P_Joints 172+48 Bytes
320	Joint 2 voltage (float) [V]									
352	Joint 3 voltage (float) [V]									
384	Joint 4 voltage (float) [V]									
416	Joint 5 voltage (float) [V]									
448	Joint 6 voltage (float) [V]									
480	Joint 1 current (float) [A]									
512	Joint 2 current (float) [A]									
544	Joint 3 current (float) [A]									
576	Joint 4 current (float) [A]									
608	Joint 5 current (float) [A]									
640	Joint 6 current (float) [A]									
672	Joint 1 position (float)[°]									
704	Joint 2 position (float) [°]									
736	Joint 3 position (float) [°]									
768	Joint 4 position (float) [°]									
800	Joint 5 position (float) [°]									
832	Joint 6 position (float) [°]									
864	Joint 1 speed (float) [°/s]									
896	Joint 2 speed (float) [°/s]									
928	Joint 3 speed (float) [°/s]									
960	Joint 4 speed (float) [°/s]									
992	Joint 5 speed (float) [°/s]									
1024	Joint 6 speed (float) [°/s]									
1088	Joint 1 temperature (float) [°C]									
1120	Joint 2 temperature (float) [°C]									
1152	Joint 3 temperature (float) [°C]									
1184	Joint 4 temperature (float) [°C]									
1216	Joint 5 temperature (float) [°C]									
1248	Joint 6 temperature (float) [°C]									
1280	Joint 1 torque (float) [Nm]									
1312	Joint 2 torque (float) [Nm]									
1344	Joint 3 torque (float) [Nm]									
1376	Joint 4 torque (float) [Nm]									
1408	Joint 5 torque (float) [Nm]									

1440	Joint 6 torque (float) [Nm]				
1472	Joint 1 servo error code (int32)				
1504	Joint 2 servo error code (int32)				
1536	Joint 3 servo error code (int32)				
1568	Joint 4 servo error code (int32)				
1600	Joint 5 servo error code (int32)				
1632	Joint 6 servo error code (int32)				
1664	Joint error state (uint8)	Joint enable state	Joint collision state (uint8)	Retention	
1696	Reserved (float) 48 Bytes				
2048	TCP position X (float) [mm]				TCP and BASE parameters 3_R->P_TCP_BASE 96+48 Bytes
2080	TCP position Y (float) [mm]				
2112	TCP position Z (float) [mm]				
2144	TCP position RX (float) [mm]				
2176	TCP position RY (float) [mm]				
2208	TCP position RZ (float) [mm]				
2240	TCP speed X (float) [mm/s]				
2272	TCP speed Y (float) [mm/s]				
2304	TCP speed Z (float) [mm/s]				
2336	TCP speed RX (float) [mm/s]				
2368	TCP speed RY (float) [mm/s]				
2400	TCP speed RZ (float) [mm/s]				
2432	TCP_OFFSET_X (float)[mm]				
2464	TCP_OFFSET_Y (float)[mm]				
2496	TCP_OFFSET_Z (float)[mm]				
2528	TCP_OFFSET_RX (float)[mm]				
2560	TCP_OFFSET_RY (float)[mm]				
2592	TCP_OFFSET_RZ (float)[mm]				
2624	BASE_OFFSET_X (float)[mm]				
2656	BASE_OFFSET_Y (float)[mm]				
2688	BASE_OFFSET_Z (float)[mm]				
2720	BASE_OFFSET_RX (float)[mm]				
2752	BASE_OFFSET_RY (float)[mm]				
2784	BASE_OFFSET_RZ (float)[mm]				
2816	Reserved 48 Bytes				
3200	Boolean register 0-31				Boolean output register DO 0~63 4_R->P_DO 8+4 Bytes
3232	Boolean register 32-63				
3264	Reserved (4 Bytes)				
3296	Integer register 0				Integer output register AO 0~31 5_R->P_AO_INT 128 Bytes
3328	Integer register 1				
3360	Integer register 2				
3392	Integer register 3				
3424	Integer register 4				
3456	Integer register 5				
3488	Integer register 6				
3520	Integer register 7				
3552	Integer register 8				
3584	Integer register 9				
3616	Integer register 10				

3648	Integer register 11	
3680	Integer register 12	
3712	Integer register 13	
3744	Integer register 14	
3776	Integer register 15	
3808	Integer register 16	
3840	Integer register 17	
3872	Integer register 18	
3904	Integer register 19	
3936	Integer register 20	
3968	Integer register 21	
4000	Integer register 22	
4032	Integer register 23	
4064	Integer register 24	
4096	Integer register 25	
4128	Integer register 26	
4160	Integer register 27	
4192	Integer register 28	
4224	Integer register 29	
4256	Integer register 30	
4288	Integer register 31	
4320	Floating point number register 0	Floating point number output register AO 0~31 6_R->P_AO_FLOAT 128 Bytes
4352	Floating point number register 1	
4384	Floating point number register 2	
4416	Floating point number register 3	
4448	Floating point number register 4	
4480	Floating point number register 5	
4512	Floating point number register 6	
4544	Floating point number register 7	
4576	Floating point number register 8	
4608	Floating point number register 9	
4640	Floating point number register 10	
4672	Floating point number register 11	
4704	Floating point number register 12	
4736	Floating point number register 13	
4768	Floating point number register 14	
4800	Floating point number register 15	
4832	Floating point number register 16	
4864	Floating point number register 17	
4896	Floating point number register 18	
4928	Floating point number register 19	
4960	Floating point number register 20	
4992	Floating point number register 21	
5024	Floating point number register 22	
5056	Floating point number register 23	
5088	Floating point number register 24	
5120	Floating point number register 25	
5152	Floating point number register 26	

5184	Floating point number register 27	
5216	Floating point number register 28	
5248	Floating point number register 29	
5280	Floating point number register 30	
5312	Floating point number register 31	

Appendix IV Ethernet/IP IO Address Table

Transmission type R->P (Robot->PLC)

Bit	0	1	2~7	8~15	16	17	18	19~23	24~31	Unit Modules
0	Robot serial number (int32)									Robot state, safety settings 1_R->P_Robot_Safety 20 Bytes
32	Servo version number (int32)									
64	Power state	Enable state	Retention	Retention						
96	MOTION_ERRCODE (int32)									
128	Move mode (uint8)			Reduction mode level	Emergency Stop	Protective stop	Soft limit state	Retention	Retention	
160	Joint 1 current (float) [A]									Joint parameters 2_R->P_Joints 124 Bytes + 20 Bytes
192	Joint 2 current (float) [A]									
224	Joint 3 current (float) [A]									
256	Joint 4 current (float) [A]									
288	Joint 5 current (float) [A]									
320	Joint 6 current (float) [A]									
352	Joint 1 position (float)[°]									
384	Joint 2 position (float) [°]									
416	Joint 3 position (float) [°]									
448	Joint 4 position (float) [°]									
480	Joint 5 position (float) [°]									
512	Joint 6 position (float) [°]									
544	Joint 1 speed (float) [°/s]									
576	Joint 2 speed (float) [°/s]									
608	Joint 3 speed (float) [°/s]									
640	Joint 4 speed (float) [°/s]									
672	Joint 5 speed (float) [°/s]									
704	Joint 6 speed (float) [°/s]									
736	Joint 1 torque (float) [Nm]									
768	Joint 2 torque (float) [Nm]									
800	Joint 3 torque (float) [Nm]									
832	Joint 4 torque (float) [Nm]									
864	Joint 5 torque (float) [Nm]									
896	Joint 6 torque (float) [Nm]									
928	Joint 1 servo error code (int32)									
960	Joint 2 servo error code (int32)									
992	Joint 3 servo error code (int32)									
1024	Joint 4 servo error code (int32)									
1056	Joint 5 servo error code (int32)									
1088	Joint 6 servo error code (int32)									
1120	Joint error state (uint8)			Joint enable	Joint collision state (uint8)				Retention	
1152~1280	reserved 20 Bytes									
1312	Sensor torque x (float) [Nm]									TCP parameters 3_R->P_TCP 76 Bytes + 48 Bytes
1344	Sensor torque y (float) [Nm]									
1376	Sensor torque z (float) [Nm]									
1408	Sensor torque rx (float) [Nm]									
1440	Sensor torque ry (float) [Nm]									
1472	Sensor torque rz (float) [Nm]									
1504	TCP position X (float) [mm]									

1536	TCP position Y (float) [mm]	
1568	TCP position Z (float) [mm]	
1600	TCP position RX (float) [mm]	
1632	TCP position RY (float) [mm]	
1664	TCP position RZ (float) [mm]	
1696	TCP_OFFSET_X (float)[mm]	
1728	TCP_OFFSET_Y (float)[mm]	
1760	TCP_OFFSET_Z (float)[mm]	
1792	TCP_OFFSET_RX (float)[mm]	
1824	TCP_OFFSET_RY (float)[mm]	
1856	TCP_OFFSET_RZ (float)[mm]	
1888	TCP linear speed V (float) [mm/s]	
1920~2272	reserved 48 Bytes	
2304	Boolean register 0-31	Boolean output register DO 0~63 4_R->P_DO 8+4 Bytes
2336	Boolean register 32-63	
2368	Reserved (4 Bytes)	
2400	Integer register 0	Integer output register AO 0~23 5_R->P_AO_INT 96 Bytes
2432	Integer register 1	
2464	Integer register 2	
2496	Integer register 3	
2528	Integer register 4	
2560	Integer register 5	
2592	Integer register 6	
2624	Integer register 7	
2656	Integer register 8	
2688	Integer register 9	
2720	Integer register 10	
2752	Integer register 11	
2784	Integer register 12	
2816	Integer register 13	
2848	Integer register 14	
2880	Integer register 15	
2912	Integer register 16	
2944	Integer register 17	
2976	Integer register 18	
3008	Integer register 19	
3040	Integer register 20	
3072	Integer register 21	
3104	Integer register 22	
3136	Integer register 23	
3168	Floating point number register 0	Floating point number output register AO 0~23 6_R->P_AO_FLOAT 96 Bytes
3200	Floating point number register 1	
3232	Floating point number register 2	
3264	Floating point number register 3	
3296	Floating point number register 4	
3328	Floating point number register 5	
3360	Floating point number register 6	
3392	Floating point number register 7	
3424	Floating point number register 8	

3456	Floating point number register 9	
3488	Floating point number register 10	
3520	Floating point number register 11	
3552	Floating point number register 12	
3584	Floating point number register 13	
3616	Floating point number register 14	
3648	Floating point number register 15	
3680	Floating point number register 16	
3712	Floating point number register 17	
3744	Floating point number register 18	
3776	Floating point number register 19	
3808	Floating point number register 20	
3840	Floating point number register 21	
3872	Floating point number register 22	
3904	Floating point number register 23	