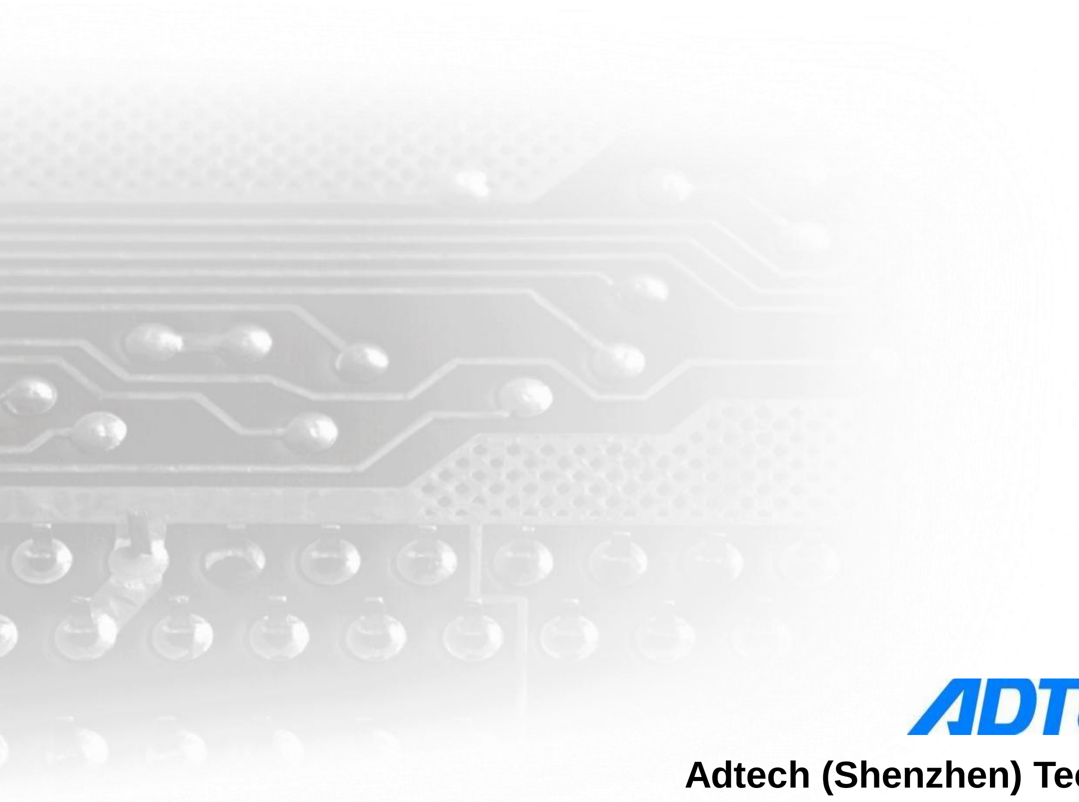
Operating Instructions of Robot

---SD Series Robot



Adtech (Shenzhen) Technology Co., Ltd.

Add: 5th Floor, Tianxia IC Industrial Park, Yiyuan Road,MaJialong,
Nanshan District, Shenzhen City, Guangdong Province, P.R.China.
TEL: +86-755-26722719 FAX: 0755-26722718-816
Email: tech@adtechcn.com Web: http://www.adtechcn.com

Copyright

Adtech (Shenzhen) CNC Technology Co., Ltd.

Contents

Chapter 1 Installation and Maintenance of Robot

Please read these instructions (manual) carefully in order to operate the SD 500 robot safely and effectively, ensure the safety of operators and other staff, as well as to avoid the production accidents such as damage to other equipment. Maximum space: The maximum space that the terminal actuator, workpiece, additional devices and other robot parts can move; this range is only predicted in design. Restricted space: The motion space of the robot that restricted by the limiter and other mechanical devices. Motion space: The motion space restricted with the software. Operating space: The restricted operating space of the robot when the task program is operating the robot.

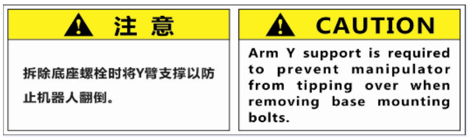
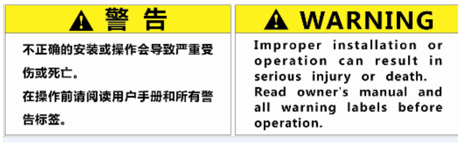
1. Installation precautions

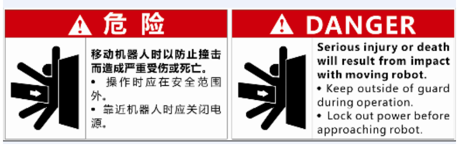
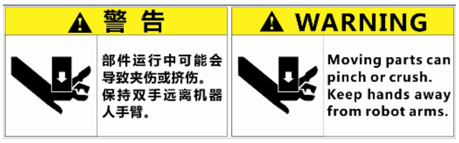
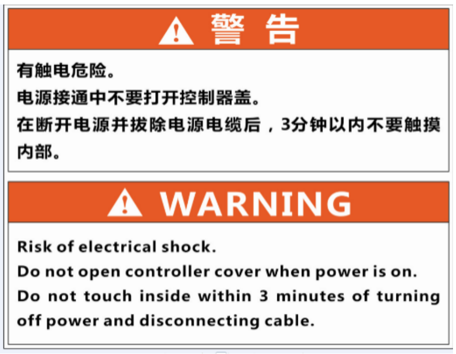
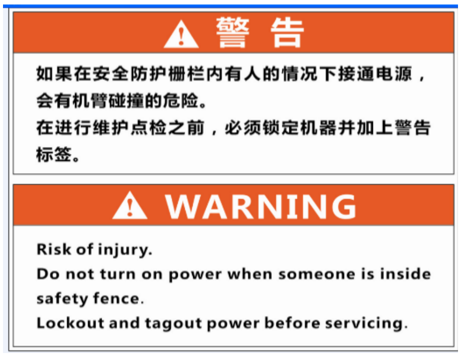
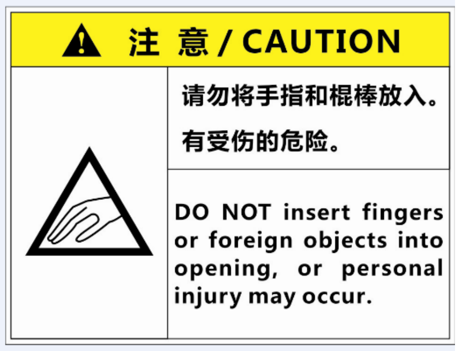
This section is the safety precautions of operating the robot. Please follow the related laws and regulations and ask qualified personnel to install the robot and related devices.

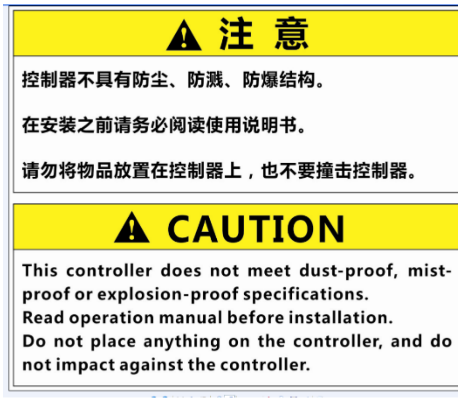
1.1 Warning and caution labels

The warning labels are pasted on the robot and controller. Specific hazard exists near the position pasted with the warning label. Please follow the warning on the label. The relevant labels and meanings are shown in Table 1-1.

Table 1-1 Warnings and Caution

Warning and caution labels	Meaning
 ⚠ 注 意 拆除底座螺栓时将Y臂支撑以防止机器人翻倒。 ⚠ CAUTION Arm Y support is required to prevent manipulator from tipping over when removing base mounting bolts.	Arm Y support is required to prevent robot from tipping over when removing base mounting bolts.
 ⚠ 警 告 不正确的安装或操作会导致严重受伤或死亡。 在操作前请阅读用户手册和所有警告标签。 ⚠ WARNING Improper installation or operation can result in serious injury or death. Read owner's manual and all warning labels before operation.	Improper installation or operation can result in serious injury or death. Read owner’s manual and all warning labels before operation.

 ⚠ 危 险 移动机器人时以防止撞击而造成严重受伤或死亡。 • 操作时应在安全范围外。 • 靠近机器人时应关闭电源。 ⚠ DANGER Serious injury or death will result from impact with moving robot. • Keep outside of guard during operation. • Lock out power before approaching robot.	Serious injury or death will result from impact with moving robot. Keep outside of guard during operation. Lock out power before approaching robot.
 ⚠ 警 告 部件运行中可能会导致夹伤或挤伤。保持双手远离机器人手臂。 ⚠ WARNING Moving parts can pinch or crush. Keep hands away from robot arms.	Moving parts can pinch or crush. Keep hands away from robot arms.
 ⚠ 警 告 有触电危险。 电源接通中不要打开控制器盖。 在断开电源并拔除电源电缆后，3分钟以内不要触摸内部。 ⚠ WARNING Risk of electrical shock. Do not open controller cover when power is on. Do not touch inside within 3 minutes of turning off power and disconnecting cable.	Risk of electrical shock. Do not open controller cover when power is on. Do not touch inside within 3 minutes of turning off power and disconnecting cable.
 ⚠ 警 告 如果在安全防护栅栏内有人的情况下接通电源，会有机臂碰撞的危险。 在进行维护点检之前，必须锁定机器并加上警告标签。 ⚠ WARNING Risk of injury. Do not turn on power when someone is inside safety fence. Lockout and tagout power before servicing.	Risk of injury. Do not turn on power when someone is inside safety fence. Lockout and tagout power before servicing.
 ⚠ 注 意 / CAUTION 请勿将手指和棍棒放入。有受伤的危险。 DO NOT insert fingers or foreign objects into opening, or personal injury may occur.	Do not insert fingers or foreign objects into opening, or personal injury may occur.

 注 意 控制器不具有防尘、防溅、防爆结构。 在安装之前请务必阅读使用说明书。 请勿将物品放置在控制器上，也不要撞击控制器。 CAUTION This controller does not meet dust-proof, mist-proof or explosion-proof specifications. Read operation manual before installation. Do not place anything on the controller, and do not impact against the controller.	This controller does not meet dust-proof, mist-proof or explosion-proof specifications. Read operation manual before installation. Do not place anything on the controller, and do not impact against the controller.
 警 告 / WARNING 有高电压部位。 请注意不要触电。 High-voltage parts. Danger: Electric Shock Risk!	High-voltage parts. Danger: Electric Shock Risk!

1.2 Ensuring appropriate installation environment

The SD500 robot does not meet dust-proof, mist-proof or explosion-proof specifications, and thus cannot be installed in the following environment.

(1) With inflammable gases or liquids;

(2) Scattered with metal chips and other conductive substances;

(3) With acid, alkaline and other corrosive gases;

(4) With mists such as cutting liquid and milling liquid;

(5) With mists such as sulfur cutting liquid and milling liquid;

(6) Near large transformer, large output high frequency oscillator, large electromagnetic contact, electric welder and other electromagnetic interferences.

⚠ **Attention: If oil is prohibited in the environment of certain industries such as electronics, food and medical, please discuss separately. The robot is lubricated but it isn’t airtight, and thus oil may leak. Please select the sealing type according to the environment.**

The ambient temperature when the robot operates should be within the range of 0~40 ℃, and the humidity shall be below 90% with no condensation. In order to ensure the normal operation of the robot, avoid installing it under excessive vibration and shock.

1.3 Ensuring operating space

Please reserve sufficient space for safe teaching and spot inspection before installing the robot and peripheral equipment.

1.4 Installing the controller outside of the restricted space of the robot

The robot controller, multifunctional teaching pendant and miniature teaching pendant should be hung outside the restricted space of the robot, and must be installed in the place where the operating status of the robot can be checked.

1.5 Installing the instruments

The pressure gauge, oil pressure meter and other instruments must be installed in the place that the operators can check at any moment.

1.6 Wiring and protection of air pipes

To avoid damaging the electrical wires and hydraulic air pipes, please protect them with shield or equivalent.

1.7 Ensuring D type grounding

The power grounding of the robot is in D type (grounding resistance is lower than 100Ω).

1.8 Installing emergency switch

To make sure that the robot can be stopped immediately upon emergency, please install the emergency switch at the place easily accessible. Thereinto,

(1) The emergency switch must be red.

(2) When the emergency switch is pressed, the robot cannot be reset automatically even if other personnel operate improperly.

(3) The emergency switch and the power switch should be installed separately.

1.9 Installing running status indicator

Please install the indicator at the place where the operators can check the status (pause, emergency or failure) of the robot.

1.10 Installing protection rail

Install the protection rail to keep the operators and other personnel out of the restricted space of the robot. Follow the precautions below to install the protection rail:

(1) Make sure that the protection rail cannot be moved or removed easily.

(2) The protection rail shouldn’t be damaged or distorted by outside force during running.

(3) Stipulate the entrance and exit of the protection rail, and make sure that the operators and other personnel cannot enter or exit through other passages.

(4) Make sure that the hands, feet or any part of the body cannot enter the protection rail.

(5) Please take the following measures at the entrance/exit of the protection rail.

① Install security door at the entrance/exit of the protection rail, and make sure that the emergency stop device starts automatically when the security door is opened.

② Indicate clearly that “Do not enter when the robot is running” and “Do not start the robot when operating” at the entrance/exit of the protection rail, and the operators should follow the regulations strictly.

If the protection rail hasn’t been installed and the robot should be started for test running, it is necessary to allocate supervisors outside the restricted space to check the status of the robot to avoid operators entering the restricted space and ensure their safety.

1.11 Setting the motion space of robot

The space that the robot requires for operating is the operating space. If the motion space of the robot is larger than the operating space, please set the motion space lower to avoid colliding with other devices.

1.12 Prohibiting refitting the robot

It is prohibited to refit the robot, controller, teaching pendant and other optional parts.

1.13 Cleaning the tools

If the robot has welding torch or painting nozzle in the front, it is recommended to use automatic cleaning rather than manual cleaning.

1.14 Ensuring the illumination

Ensure the illumination required for safe operation.

1.15 Preventing flying off of clamped objects

The objects clamped by the robot may fly off or drop off. To ensure the safety of operators, it is necessary to take protective measures according to the size, weight, temperature, and chemical properties.

1.16 Pasting warning labels

Please paste the “Warning Label” in the robot components at prominent position such as the entrance/exit of the protection rail.

2. Operating precautions

⚠ **Warning: It may cause serious casualties to touch the running robot. Therefore, please follow the precautions below while operating:**

① When the robot is running or the motor is started, do not enter the restricted space of the robot.

② If it is necessary to enter the restricted space to process abnormalities, please start the emergency stop device and turn off the motor of the robot.

③ If it is necessary to operate in the restricted space for teaching and spot inspection, please take the measures in "4.2 Ensuring the safety of operators in restricted space".

2.1 Developing and effectively implementing “Operating Rules”

If it is necessary to operate in the restricted space for teaching and spot inspection, please develop and effectively implement the “Operating Rules” according to the precautions below.

(1) The necessary steps to start the robot;

(2) Robot velocity for teaching;

(3) The contact mode in joint operation state (two or more operators);

(4) The measures taken by the operators when abnormity occurs;

(5) When the robot is stopped with the emergency switch, the necessary measures taken to confirm the release of abnormity and the security to restart the robot;

(6) In addition to above rules, it is also necessary to take the following the measures below to avoid unexpected running of the robot or dangers caused by misoperation.

① The operating position and gesture of the robot should allow checking the status of the robot at any moment and evading immediately when any abnormity occurs.

② Measures to prevent interference.

③ The contact mode used by the operators of related machines.

④ Category and identification of abnormities.

Please develop the “Operating Rules” according to the type, installation location and operation of the robot.

2.2 Ensuring the safety of operators in restricted space

When operating in the restricted space of the robot, please take one of the following measures to ensure stopping the robot immediately once any abnormity occurs.

(1) Start the emergency stop device immediately when abnormity occurs;

(2) Only operators can enter the restricted space of the robot;

(3) The operators in the restricted space should carry the emergency stop switch (or the emergency stop button for the multifunctional teaching pendant or miniature teaching pendant);

(4) Please allocate supervisors outside the restricted space to check the status of the robot.

2.3 Spot inspection before guide and other operations

Before teaching, please check the following points, and take maintenance or other necessary measures when any abnormity occurs.

(1) Whether the coating and packaging of external wires are damaged;

(2) Whether the operating of robot is normal (whether there is abnormal sound or vibration when started);

(3) Whether the emergency stop device is normal;

(4) Whether the pipe leaks gas or oil;

(5) Whether there is barrier in or near the restricted space of the robot.

2.4 Release of residual pressure

Please release the residual pressure in the drive cylinder before decomposing air compression system or replacing parts.

2.5 Confirming the precautions during operating

The operators should move to outside of the restricted space to confirm the operating.

2.6 Precautions during automatic running

(1) Start the robot after confirming the following points:

① There is no person in the restricted space of the robot.

② The multifunctional teaching pendant and tools are in specified position.

③ The indicators of the robot and related devices do not show abnormity.

(2) During automatic running, check whether the AUTO indicator is lighted;

(3) Occur abnormal situation.

When the robot or related devices have problems and it is necessary to enter the restricted space to operate, please stop the robot with the emergency stop device, and indicate “Operating” or similar message on the start switch to prevent other personnel operating the robot.

2.7 Maintenance precautions

(1) Do not repair outside specified range;

(2) Do not disassemble the interlock switch in any cases.

(3) Only use approved parts for maintenance.

2.8 Daily spot inspection and periodic inspection

Please implement daily spot inspection and periodic inspection, and check whether the robot and related devices are normal before operating. If not, please take maintenance or other necessary measures immediately. Please record the details of periodic inspection and maintenance, and save the records for at least three years.

2.9 Safety regulations

Please follow the safety regulations of industrial robot. Please refer to the GB/T20867-2007 *Industrial robot - Safety implementation specification*.

3. SD series robot body

3.1 Basic structure of the robot

SD series robot features compact contour, small size and light weight, it is the most suitable robot for formation of small units and can be used to load and unload materials, sort and assembly with high speed and accuracy. Meanwhile, all cables of SD series robot are so built-in that can ensure flexible movement within a narrow space and installation in either grounded, wall-mounted or ceiling-mounted manner. Among all types of robots, SD500 model has its maximum working radius of 500mm, its wrist can endure a rated load of 1Kg and maximum load of 3Kg. Figure 3-1 is the actual picture of SD500 robot; Figure 3-2 is the figure of structure of SD series robot body; Figure 3-3 is the schematic diagram of each joint movement. SD700 model is such a robot that is elongated the forearm and large arm of SD500 model, its wrist endures same load with SD500 model and the maximum working radius is 700mm. Figure 3-4 and Figure 3-5 are respectively operating space diagram of SD500 and SD700 robot.

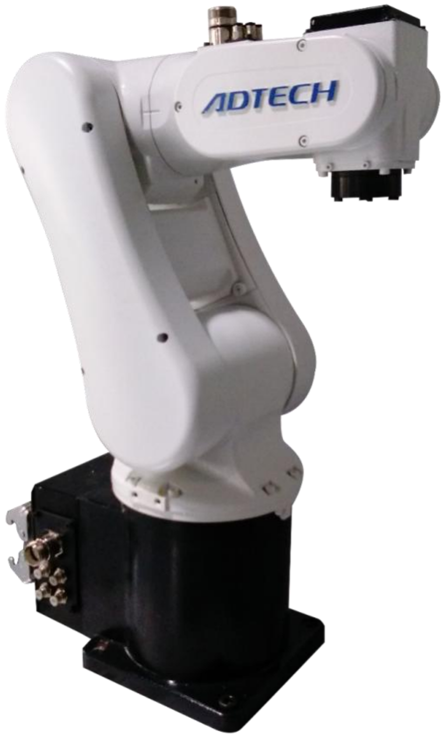
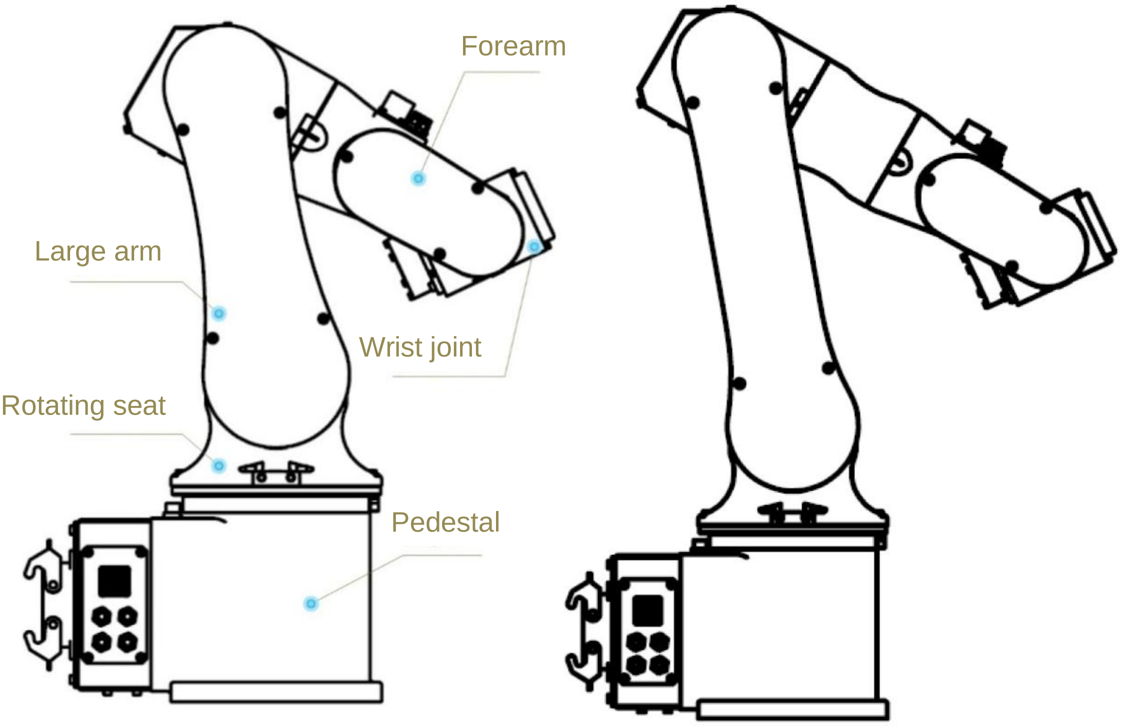


Figure 3-1 Real object of SD500 robot



SD500 model SD700 model

Figure 3-2 Figure of structure of SD series robot body

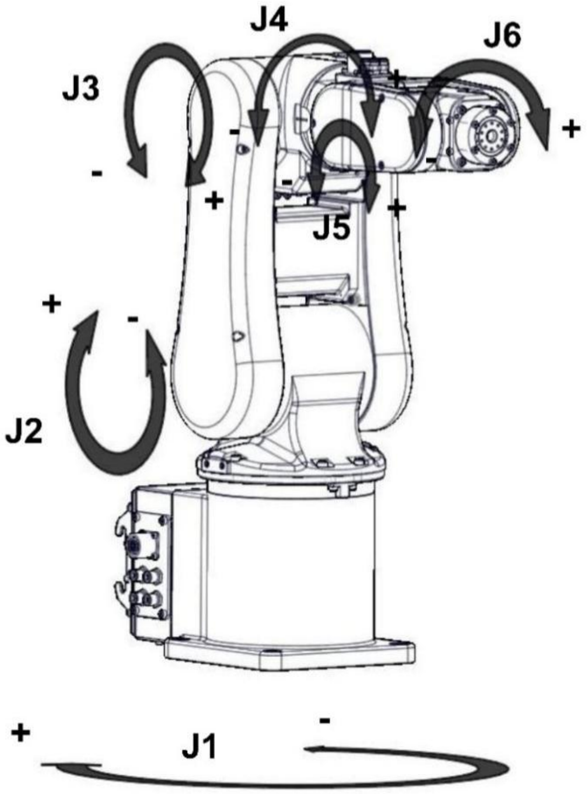


Figure 3-3 Schematic diagram of kinematic axis of SD series robot

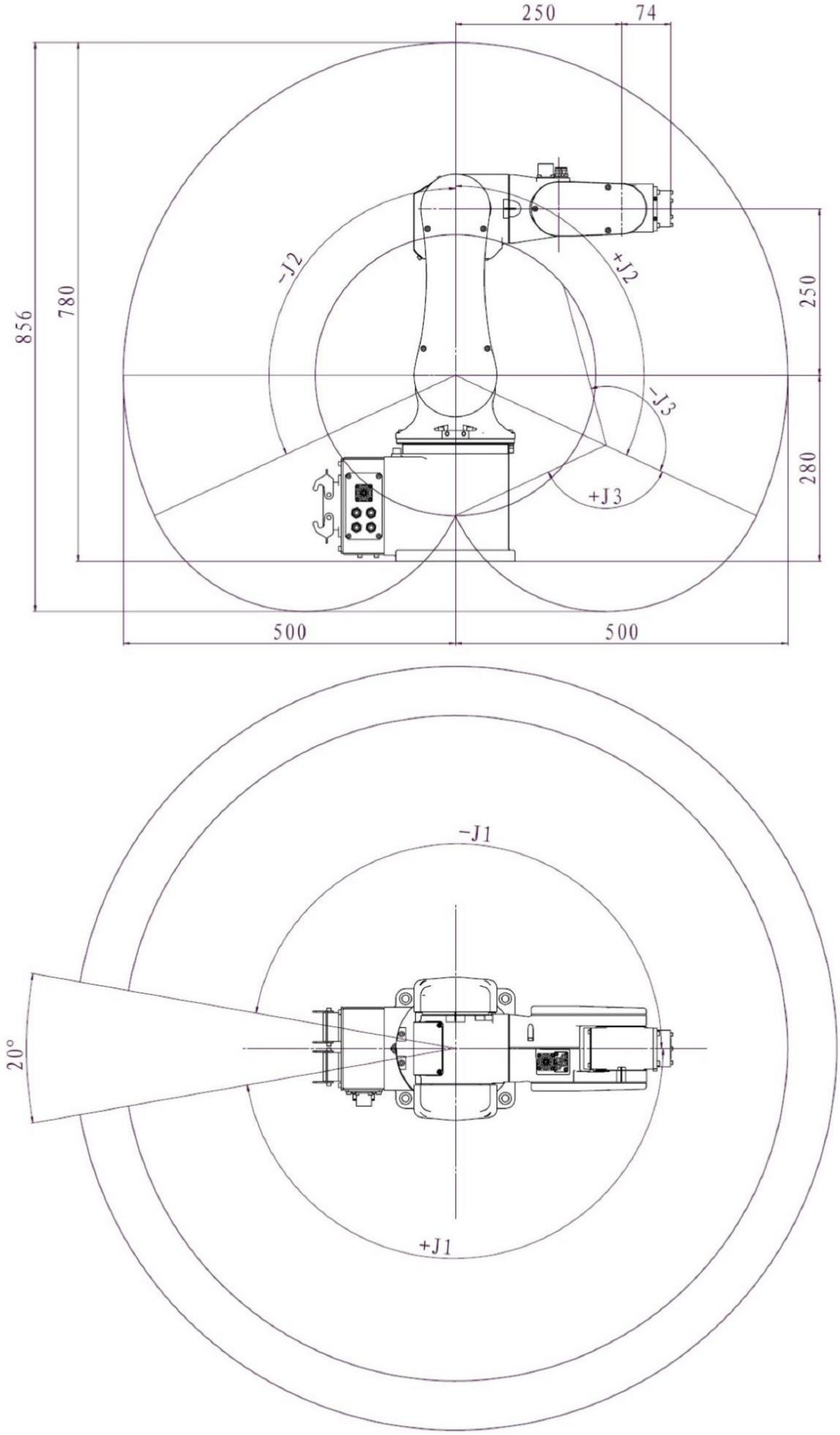


Figure 3-4 Working space of SD500 robot

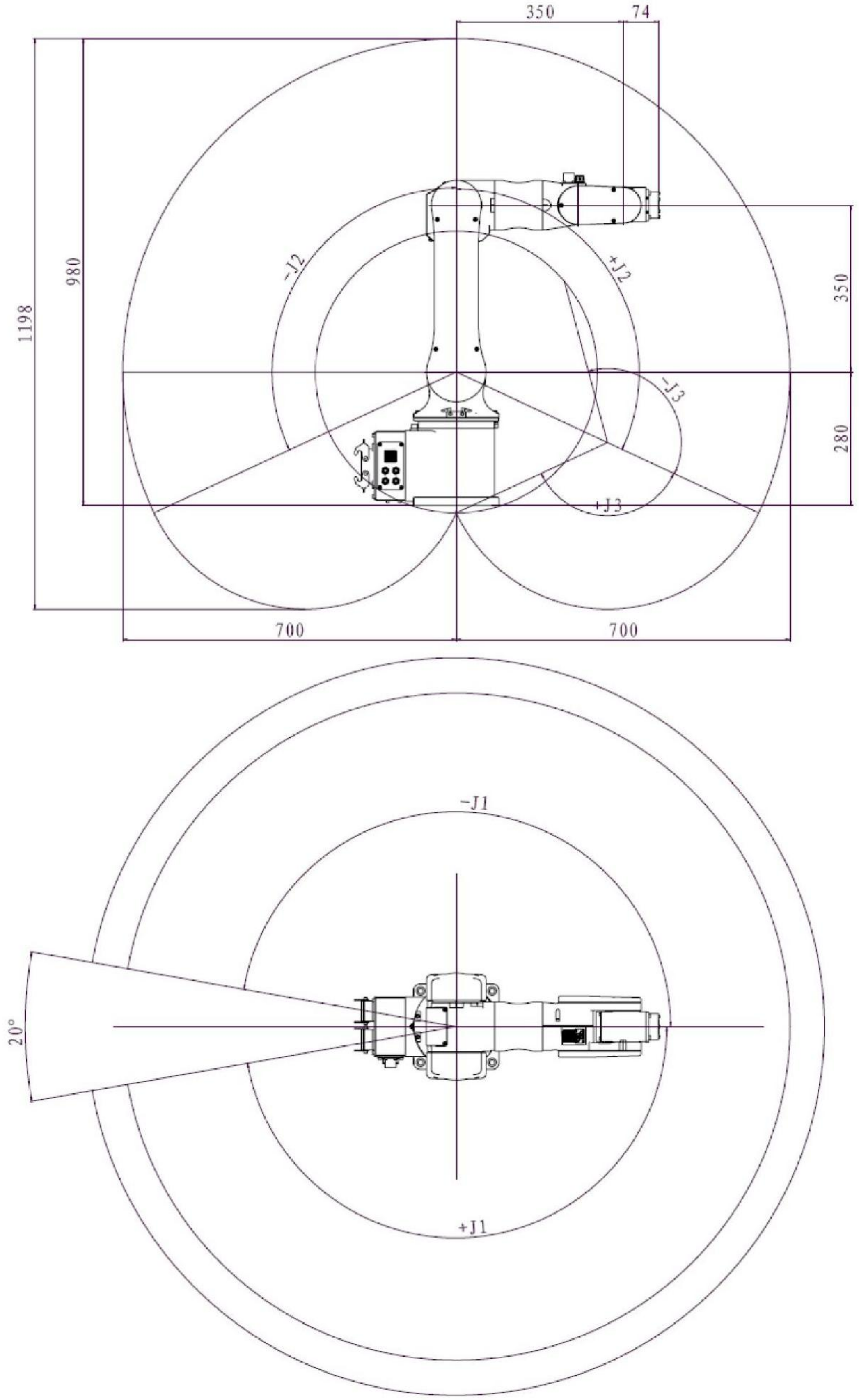


Figure 3-5 Working space of SD700 robot

All cables of SD series robot are so built-in that can ensure flexible movement within a narrow space, the robot can be installed in grounded, wall-mounted or ceiling-mounted way through a connection between its pedestal and mounting face. The structure of pedestal installation can be referred in Figure 3-7 and the installation structure of wrist tool flange is shown in Figure 3-8.

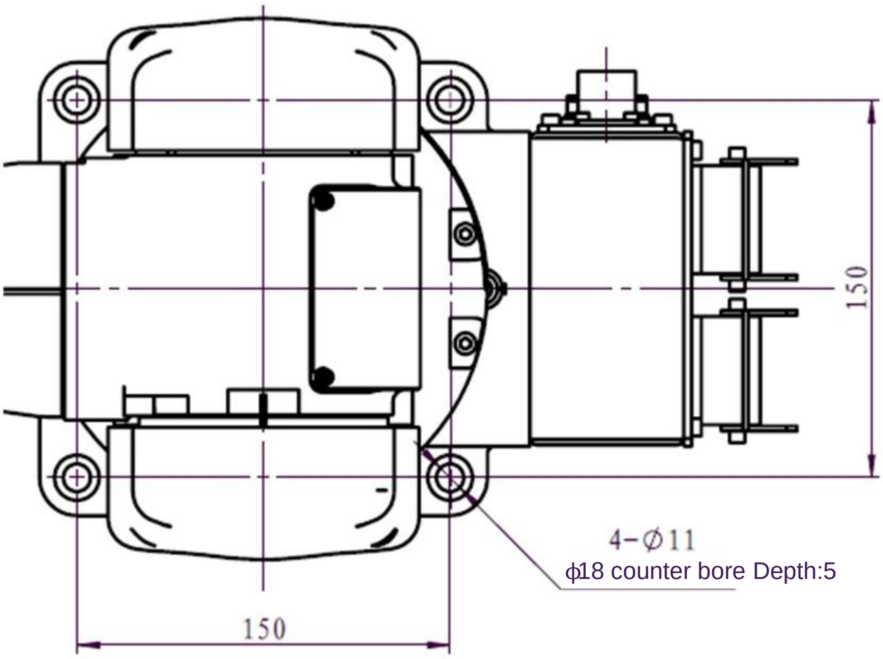


Figure 3-6 Mounting holes on the pedestal

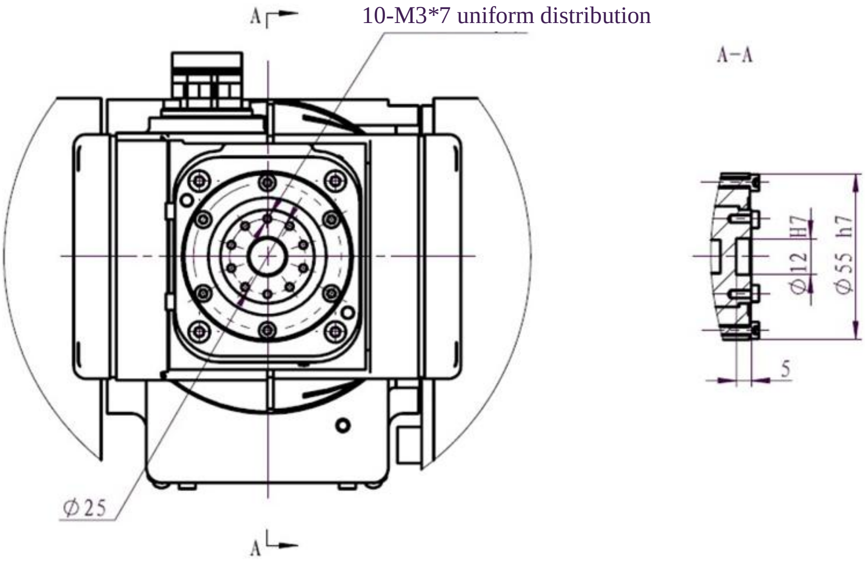


Figure 3-7 Flange mounting hole of wrist tool

3.2 Performance parameter of the robot

SD series robot features compact contour, small size and light weight, it is the most suitable robot for formation of small units and can be used to load and unload materials, sort and assembly with high speed and accuracy. Meanwhile, all cables of SD series robot are so built-in that can ensure flexible movement within a narrow space and installation in grounded, wall-mounted or ceiling-mounted manner. Table 3-1 provides main performance parameters of SD series robot. Among which, the relationship between the quality of wrist load and barycenter is shown in the figure.

Table 3-1 Performance parameter of SD series robot

Model		**SD500**	**SD700**
Type		Vertical multi-joint	Vertical multi-joint
Number of axes		6	6
Rated/Maximum load		1/3 kg	1/3 kg
Maximum turning radius		500 mm	700 mm
Range of movement (°)	J1	±170	±170
	J2	±115	±115
	J3	+40 ~ -220	+40 ~ -220
	J4	±180	±180
	J5	±125	±125
	J6	±360	±360
Maximum speed (°/s)	J1	375	250
	J2	375	185
	J3	430	290
	J4	460	460
	J5	460	460
	J6	600	600
Instantaneous maximum allowable torque	J5	35 Nm	35 Nm
	J6	24 Nm	24 Nm
Normal cycle time		0.6s	0.6s
Repeat positioning accuracy		±0.015 mm	±0.015 mm
Working temperature		0 ~ 45℃	0 ~ 45℃
Signal wires used by the users		0.14sqX16 line	0.14sqX16 line
Position limitation protection		1. Software limit 2. Mechanical limit	1. Software limit 2. Mechanical limit
Weight		26 Kg	28 Kg

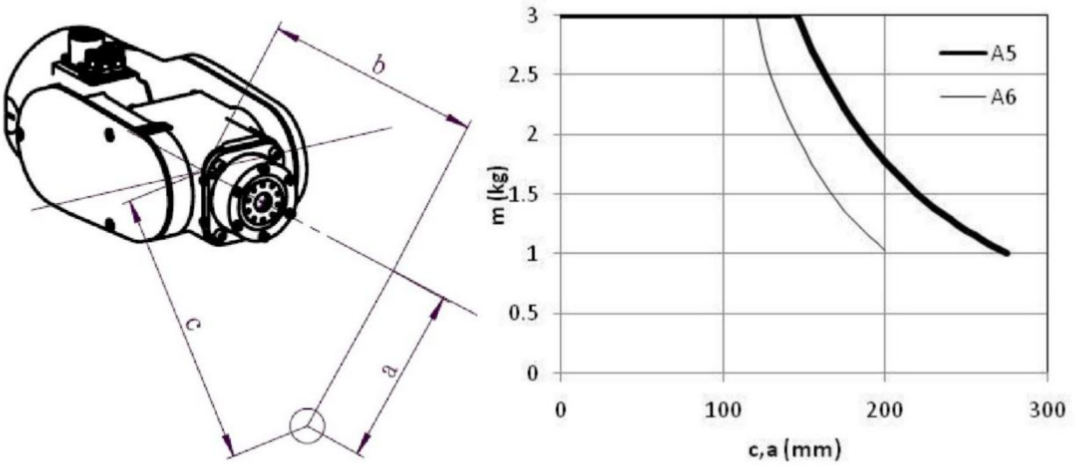


Figure 3-8 The relationship between wrist load quality and load barycenter
(A5 and A6 denote respectively the 5th and 6th axis, c and a represent respectively the distance of load barycenter to the 5th and 6th axis)

3.3 Electrical configuration of the robot body

SD series 6-axes robot is equipped with 6 motors and their corresponding encoders, which are connected respectively via 32-core gravity overload connector and 42-core gravity overload connector to the controller, two end interfaces on the robot body are shown in Figure 3-9. The definitions of connector pin can be referred respectively in Table 3-2 and Table 3-3.

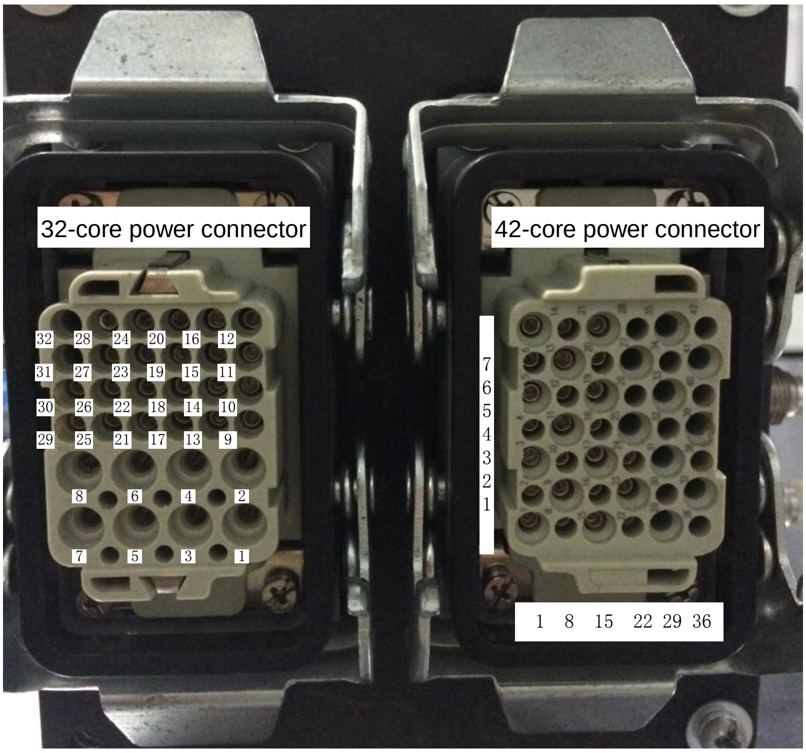


Fig 3-9 Electric interface of the robot body

Table 3-2 Definition of 32-core power connector interface

Power cable			
Use of cable	Motor shaft	Definition of cable	Body interface 32-core power connector
Power wire	M1	U	1
		V	2
		W	3
	M2	U	4
		V	5
		W	6
	M3	U	7
		V	8
		W	9
	M4	U	10
		V	11
		W	12
	M5	U	13
		V	14
		W	15
	M6	U	16
		V	17
		W	18
Internal contracting brake	M1	BK+	19
		BK-	20
	M2	BK+	21
		BK-	22
	M3	BK+	23
		BK-	24
	M4	BK+	25
		BK-	26
	M5	BK+	27
		BK-	28
	M6	BK+	29
		BK-	30
Note: The motor on each shaft shall be provided with 4 cables: U, V, W and PE, PE cables of the motor are all connected on the housing of overload connector, which are only required to connect with UVW3 cable.			

Table 3-3 Definition of 42-core power connector interface

Encoder cable			
Application of the cable	Motor shaft	Definition of the cable	Body interface 42-core power connector
Encoder cable	M1	SD+	1
		SD-	2
		5V	3
		GND	4
	M2	SD+	5
		SD-	6
		5V	7
		GND	8
	M3	SD+	9
		SD-	10
		5V	11
		GND	12
	M4	SD+	13
		SD-	14
		5V	15
		GND	16
	M5	SD+	17
		SD-	18
		5V	19
		GND	20
	M6	SD+	21
		SD-	22
		5V	23
		GND	24

Note:

1.Both ends of all shielded cables among encoder cables are connected on the housing of overload connector;

2.The encoder is provided with 6 cables; Besides 4 listed in the table, the rest 2 are battery cables,SD500 model is equipped with built-in batteries.

On the lateral side of robot pedestal air pressure and electric interface is reserved for hand-grasp tools, as shown in Fig 3-10, to facilitate wiring of hand-grasp tool cable from inside of the robot and prevent dangers caused by robotic winding of hand-grasp accessory cables.

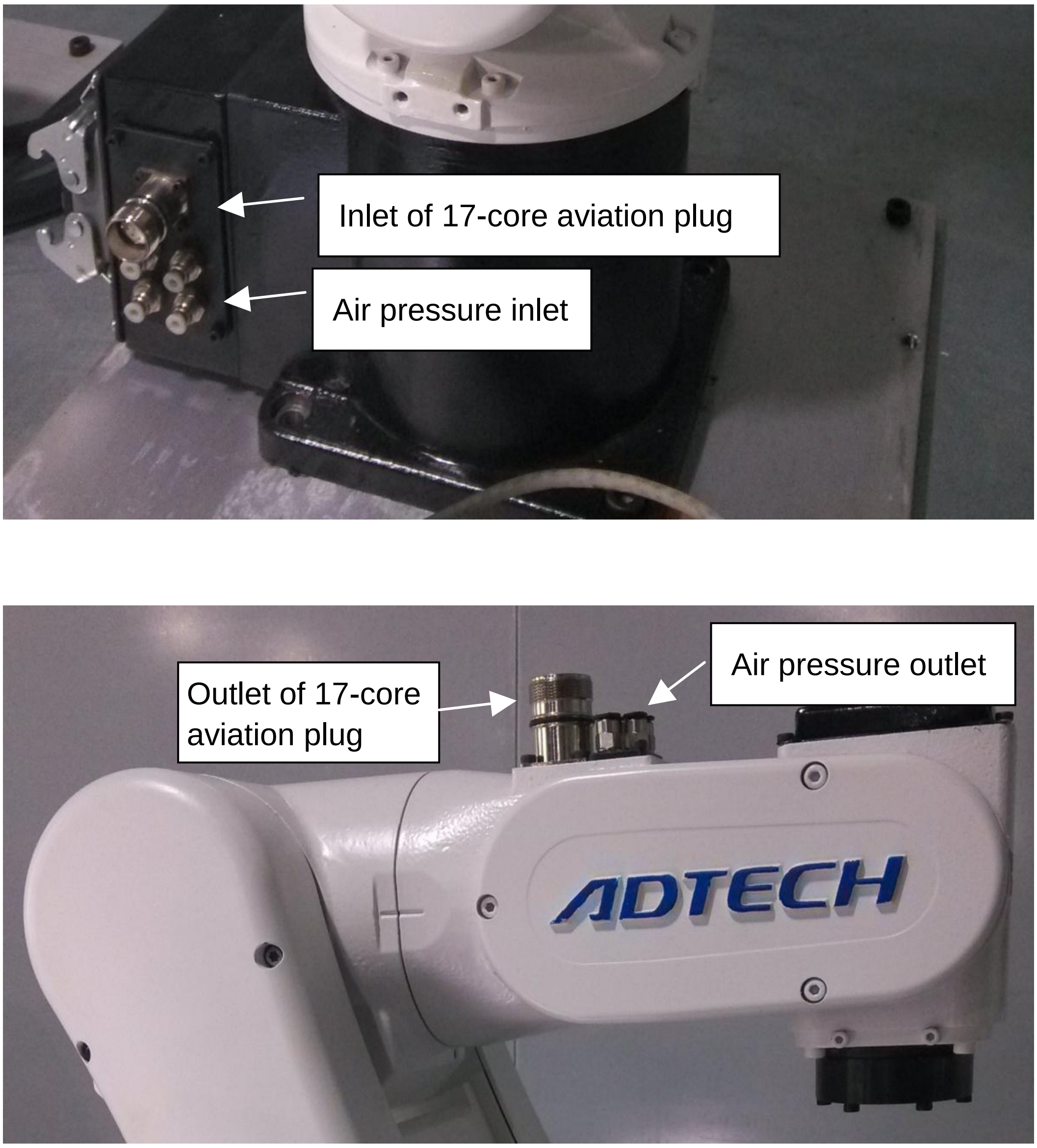
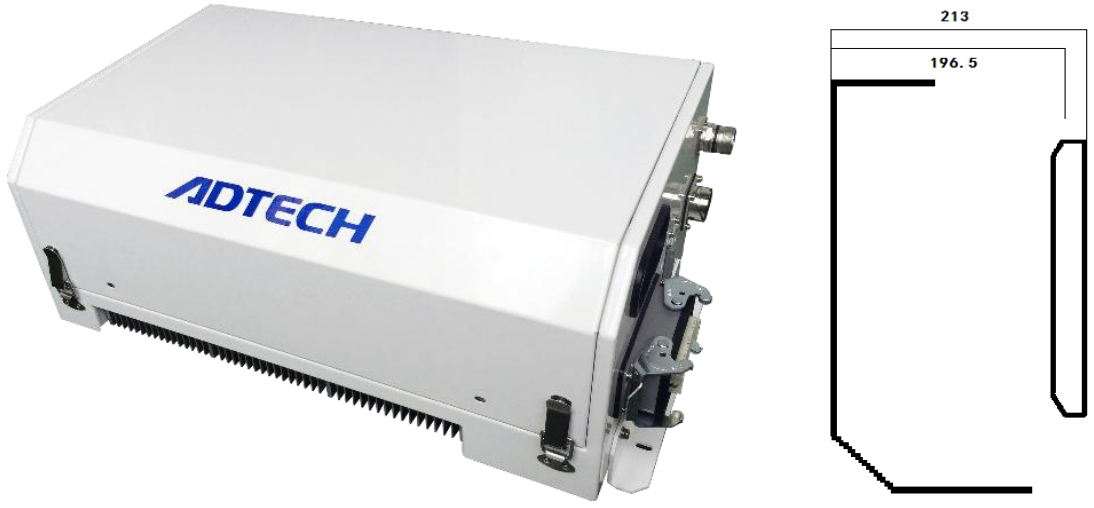

Inlet of 17-core aviation plug
Air pressure inlet
Outlet of 17-core aviation plug
Air pressure outlet
ADTECH

Figure 3-10 Air pressure and cable interface reserved for hand-grasp tools

4. Controller of SD series robot

4.1 Introduction of the controller structure

SD series robot adopts 6-axes drive-control integrated controller QC600A newly developed by ADTECH, which is a tailor-made drive-control integrated system developed specially by this company for 6-axes robot. By revolutionary adoption of integrated design technology of motion control system and servo drive system, this product achieves real full-closed loop control between control system and servo drive system. QC600A integrates ADTECH's new generation of QX servo control technology and DSP motion control card technology and adopts advanced design means like distributed CPU and high-speed internal bus (IB) that demonstrates design achievements of high integration. Figure 4-1 is the controller cabinet appearance and machine body dimensional drawing; Figure 4-2 shows the external terminal plate. The list of QC600A controller configuration is shown in Table 4-1; the user is required to check whether or not all items configured are complete.



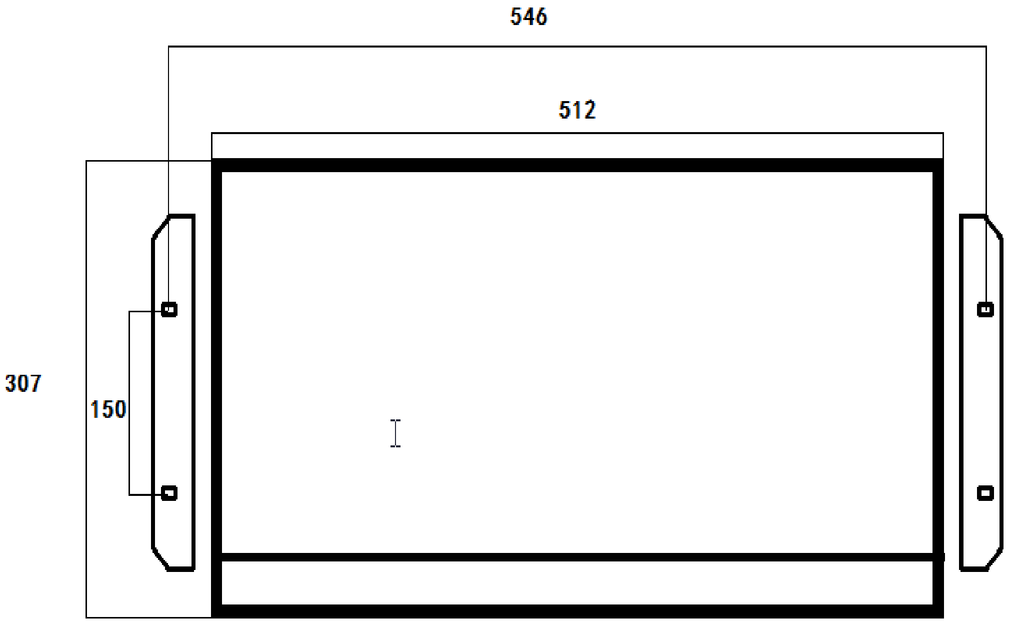


Figure 4-1 Controller cabinet appearance and machine body dimensional drawing

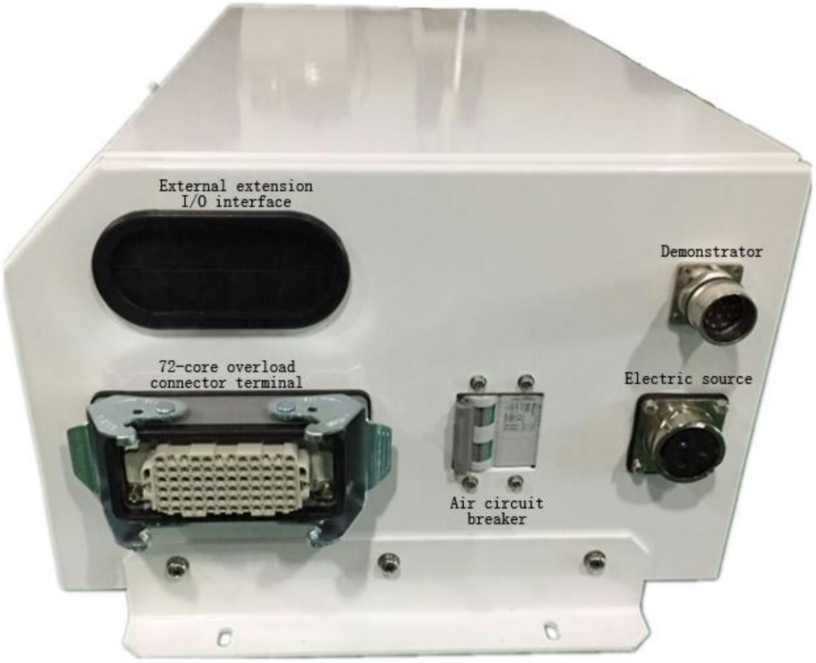


Figure 4-2 External terminal plate interface of the controller

Table 4-1 List of controller configuration

Item Name	Illustration	Quantity
QC600A	Drive and control integrated	1
Power plug	Power plug of main engine	1
Connecting line of the motor	72-core overload connecting line	1
SRPT1000 demonstrator	Multi-functional and independent hand-held demonstrator	1

QC600A features line-saving field mounted experience and intuitive human-machine interaction; In terms of teaching programming and operation, ADTECH and its parent company (Shanghai STEP Electric Corporation) form powerful alliance to co-develop operating software based on WinCE system, it is stable, reliable and has rich, powerful operation control statement, supports point-to-point, linear and circular motion, loop-jump uniform flow control, tool coordinate system and user coordinate system settings and I/O signal reading and output, the programming language is popular and understandable. Meanwhile, as to two types of operations: welding and palletizing, QC600A develops special instruction set to improve significantly the efficiency of programming and ensure safe, reliable operation. This product has many advanced functions, such as 6-axes linear interpolation, circular interpolation in 3D space, trajectory tracking, stable acceleration and deceleration and speed self-adaption of continuous motion trajectory and be applied in various occasions of six-axes robot. QC600A is compatible with a variety of brand absolute and incremental motors developed domestically and internationally, including motors branded SANYO, Panasonic and TAMAGAWA, the model of motor is optional and can drive directly 750W AC servo motor. Therefore, QC600A is suitable both for matching drive and control of SD series robot and control of other servo motors. Table 4-2 lists specification parameters of QC600A drive-control integrated controller.

Table 4-2 List of specification parameters of QC600A drive-control integrated controller

Item				Model
Model				QC600A
Demonstrator	Display			8 inch
	Teaching method			Manual data input(coordinate value input) Teaching of orthogonal and joint coordinate
	Input of trajectory programming language			SR language
	Data wire			5M
	Size			400 mm× 250 mm ×70 mm(Demonstrator)
	Weight			Approximate 2Kg
	Electric source			24V DC
	Data exchange			USB
Drive control integrative machine	Axial-controlled function	Number of control axes		6 axes
		Type of servo motor		AC servo motor (SANYO, Panasonic, TAMAGAWA and ADTECH servo motors)
		Supporting encoder		Protocol servo encoder(incremental/absolute), AB dual phase encoder
		Position control cycle		8K/16K
		Motion/servo control function		Linear interpolation, circular interpolation, continuous trajectory, trajectory smoothness, acceleration and deceleration of various kinds, etc.
		Coordinate system		Joint coordinate system, rectangular coordinate system
		Unit setting of the position		Degree, mm
		Programming environment		Teaching page with robotic movements, object-oriented programming statement
		Storage means		Memory on board
	External input/output	Standard IO	IO input	48 lines
			IO output	48 lines
		External communication	RS-232	3 pieces(dual line and DSP, single line and X86 core board)
			RS-485	1

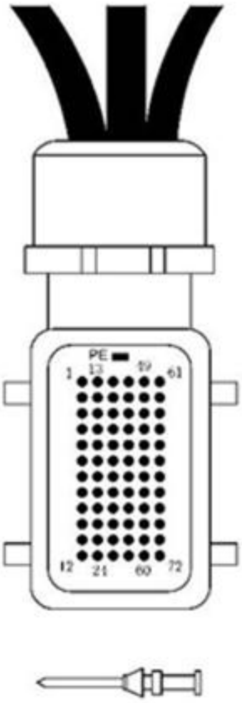
			Ethernet	3
			USB(2.0)	2
	Basic size	External dimension		550mm ×307mm × 213mm(controller)
		Weight		17 Kg
		Power cable		1.5M
		Electric source used		220V AC/50~60 Hz
		Capacity factor		3KW(single-axis output≤750W)
		Atmospheric pressure		86Kpa～106Kpa
		Operating temperature		0~40℃
		Storage and transport temperature		-10~65℃
		Operating temperature		20% ~ 90% RH
		Storage and transport humidity		10% ~ 95% RH

4.2 Extended port of the controller

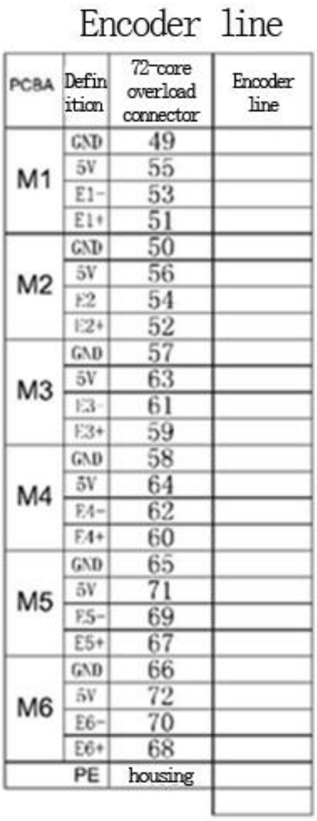
QC600A drive and control integrative machine is equipped with 48 lines of I/O inputs and I/O outputs. As an old saying goes that “keep one’s eyes and ears open”, this machine can be used to improve substantially external cooperative ability. Open the front panel of QC600A machine, I/O panel is placed on the topside of electrical cabinet, the customer may use extended port as required.

4.3 Port of controller over-load connector

QC600A is connected to robot body through 72-core overload connector, which includes power wire powering servo motor, encoder wire used to connect motor encoder and brake wire used for switch locking. As QC600A is compatible with multiple servo motor brands, the definition of 72-core overload connector port in QC600A machine is provided here in order to facilitate electric connection as per the brand of motor used by the users. Here the definition of 72-core overload connector in QC600A is given, as shown in Figure 4-4.

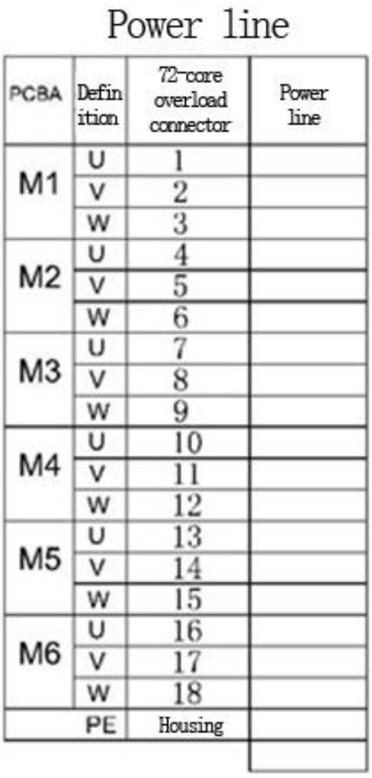


The above is the schematic diagram of overload connector and metal contact pin



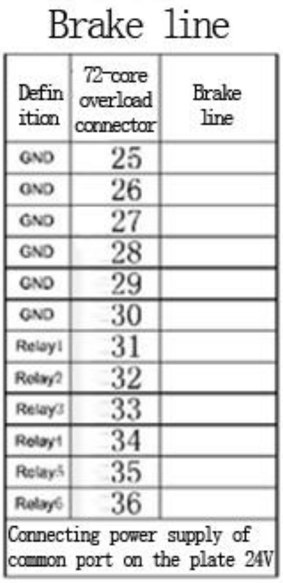
Encoder line

PCBA	Definition	72-core overload connector	Encoder line
M1	GND	49	
	5V	55	
	E1-	53	
	E1+	51	
M2	GND	50	
	5V	56	
	E2	54	
	E2+	52	
M3	GND	57	
	5V	63	
	E3-	61	
	E3+	59	
M4	GND	58	
	5V	64	
	E4-	62	
	E4+	60	
M5	GND	65	
	5V	71	
	E5-	69	
	E5+	67	
M6	GND	66	
	5V	72	
	E6-	70	
	E6+	68	
	PE	housing	



Power line

PCBA	Definition	72-core overload connector	Power line
M1	U	1	
	V	2	
	W	3	
M2	U	4	
	V	5	
	W	6	
M3	U	7	
	V	8	
	W	9	
M4	U	10	
	V	11	
	W	12	
M5	U	13	
	V	14	
	W	15	
M6	U	16	
	V	17	
	W	18	
	PE	Housing	



Brake line

Definition	72-core overload connector	Brake line
GND	25	
GND	26	
GND	27	
GND	28	
GND	29	
GND	30	
Relay1	31	
Relay2	32	
Relay3	33	
Relay4	34	
Relay5	35	
Relay6	36	
Connecting power supply of common port on the plate 24V		

Metal contact pin used in the power line: CDMA-0.7
Metal contact pin used in the encoder line and brake line: CDMA-0.3

Figure 4-4 Definition of 72-core overload connector

5. Demonstrator of SD series robot

Usually SD series robot finishes predetermined motions through its controlling of robot body. The hand-held box consists of keys and buttons used for manually operating robot body and preparatory working procedures.

5.1 Illustration of front panel

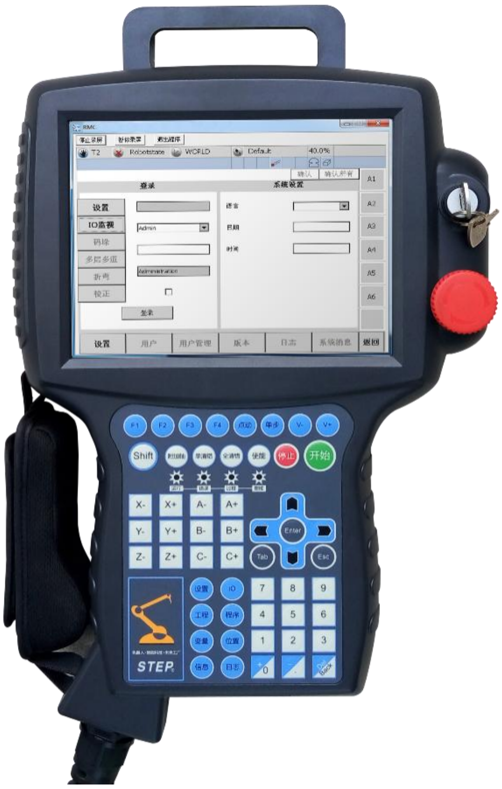
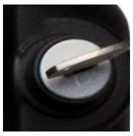
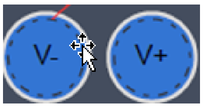
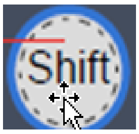


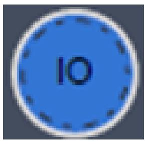
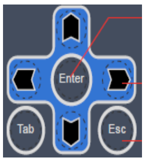
Figure 5-1 Front panel of the demonstrator

5.1.1 Illustration of hand-held press key and display

Press key	Illustration
	**【A/Lock/M】**:Key Switch, it is used for mode conversion among 3 modes: A(Auto), Lock(Lock) and M(Manual), PC can control system only in Lock mode.
	【STOP】:Emergency Stop Key, it is used in emergencies caused by possible risk of collision between the robot and peripheral equipment during operation, and can also be used to switch off “Motor Enable” and turn off [FN] indicator lamp.

	【Reserve Functional Key】: This key can be re-defined on the spot as required.
	【JOG】: This key is used to switch the robot’s jogging coordinate system.
	【SINGLE STEP】: This key is used to switch the robot’s program running mode and achieve loop interchange among 3 modes: CONTINUOUS, SINGLE-STEP and MOVING SINGLE STEP.
	**【V-/V+】**:This key is used to adjust the robot’s operating speed.
	**【Shift】**: This key is used to shift compound key. If this key is not pressed, the line under the compound key is valid; If this key and compound key are pressed simultaneously, the line above this key is valid.
	【ADDITIONAL AXIS】: When the robot is equipped with additional axes, press this key and ADDITIONAL AXIS 47 is displayed in the jogging status area on the right of the demonstrator, the content of display has something to do with the number of additional axes.
	【SINGLE ERROR CLEARING】: Clear one error.
	【All ERROR CLEARING】: Clear all errors.
	【ENABLE】: This key is used to enable the robot in AUTO/EXTERNAL AUTO mode.

<table>
<tr><td>STOP</td><td colspan="4">【<b>STOP</b>】: This key is used to stop robotic operation.</td></tr>
<tr><td>START</td><td colspan="4">【<b>START</b>】: This key is used to start robotic procedure.</td></tr>
<tr><td rowspan="2">RUN ERR PRO ENB</td><td>RUN</td><td>ERR</td><td>PRO</td><td>ENB</td></tr>
<tr><td>【OPERATION】:This indicator lamp is green when the robot is being operated.</td><td>【ERROR】: This indicator lamp is red when the robot reports errors.</td><td>【PROCESS】: Deactivated</td><td>【ENABLE】: This indicator lamp is green after the robot is enabled.</td></tr>
<tr><td>USER SETTING</td><td colspan="4">【<b>USER SETTING</b>】: User management (user’s log-in, obtaining and granting of control right), user edit (user edit, create and delete operations).</td></tr>
<tr><td>ENG</td><td colspan="4">【<b>ENGINEERING MANAGEMENT</b>】<b>:</b> This key is used to conduct editing operation on the engineering (create, delete and copy, etc.) and administrative operation (load, close and open), editing operation (create, delete and copy, etc.) on the procedures; Display current procedures being loaded or opened.</td></tr>
<tr><td>VARIABLE</td><td colspan="4">【<b>VARIABLE MANAGEMENT</b>】<b>:</b> This key is used to maintain variables of demonstrator programs and provide functions such as variable display, correction, creation, copying, cutting, pasting, deleting and renaming as well as demonstration of type, variables of the robot’s positions.</td></tr>
</table>

	【INFORMATION】: This key provides quick entry into current error check page of the robot.
	**【IO】:**This key provides quick entry into IO status check and compulsory page.
	【PROGRAM MANAGEMENT】: This key is used to complete operating functions like program display and edit.
	【POSITION】:This key provides quick entry into robot position check page.
	【LOG】:This key provides quick entry into existing error check page of the robot.
	**【Up/Down/Left/Right key】**: This key is used to move cursor both vertically and horizontally during input.
	【Tab】: Deactivated
	【Esc Key】:Deactivated
	【Enter】: It is used to confirm input.

5.2 Illustration of rear panel

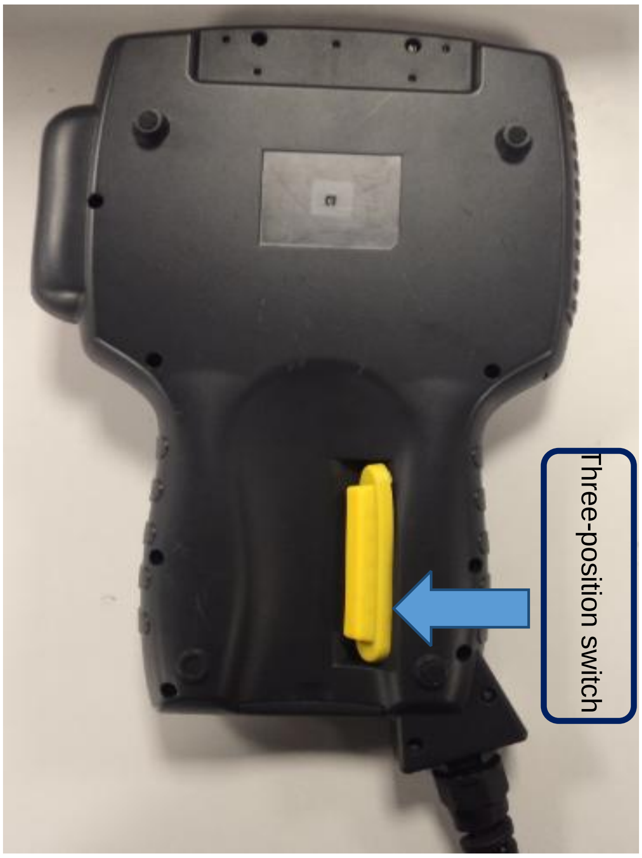


Figure 5-2 Rear panel of hand-held box

5.2.1 Explanation of three-position switch

This switch refers to the buttons at both side of the demonstrator body, as shown in Figure 5-2. This is an enable switch in the manual mode; the demonstrator is equipped with three sections of ON/OFF switch key. If the key is not pressed, the robot’s brake is closed and in OFF status; Press this key gently to enable the robot and open the brake (with clicking sound), in such circumstance manual operations may be conducted to switch to ON status. Keep pressing this key until it is power-off, at this moment the brake is closed and in OFF status.

⚠**Notes:**

1. The robot is not permitted to operate if the enable switch is not in ON status.

2. If the enable switch is in OFF status (by loosening the hand or compulsory pressing) during hand operation, the motor is closed and in OFF status.

6. Robot Installation

6.1 Ensuring appropriate installation environment

When installing the robot body and controller, make sure that the use environment is identical to each item in the “1.Install Considerations" in Chapter 1 “Robot Installation and Maintenance”. If the installation environment isn’t appropriate, it is not possible to exert the performance of the robot, and also will shorten the service life of the machine, or even cause serious accidents. Table 6-1 summarizes the installation environment and conditions parameters of the robot body. For the installation of the robot body, please refer to Figure 6-1, and prepare the mounting rack with sufficient rigidity.

Table 6-1 Installation environment and condition of the robot

Item	Environment & condition
Flatness of installation bench	0.1 / 500mm
Rigidity of installation bench	Steel material
Installation direction	Ground installation, hanging mounting, lifting
Ambient temperature	Operating: 0~40℃; storage & transportation: -10~60℃
Humidity	Operating: <90% (no condensing); Storage & transportation: <75% (no condensing)
Vibration	Operating: <4.9 m/s2 (0.5G); Storage & transportation: <29.4 m/s2 (3G)
Safe installation environment	No inflammable gases or liquids ·No acid, alkaline and other corrosive gases ·No mists such as sulfur cutting liquid and milling liquid ·No large transformer, large output high frequency oscillator, large contact, electric welder and other electromagnetic interferences.
	·No metal chips and other conductive substances. ·No mists such as cutting liquid and milling liquid. ·No water, oil, or scattered chips.
Operating space	·Ensure sufficient spot inspection and disassembly space. ·Reserve wiring space in the rear of the robot (standard: >190mm; dust-proof, mist-proof and cleaning > 230mm); do not apply the weight of the cable on the connector directly; fix the wiring on the installation surface or beam.
Installation condition	D type grounding (resistance <100Ω)

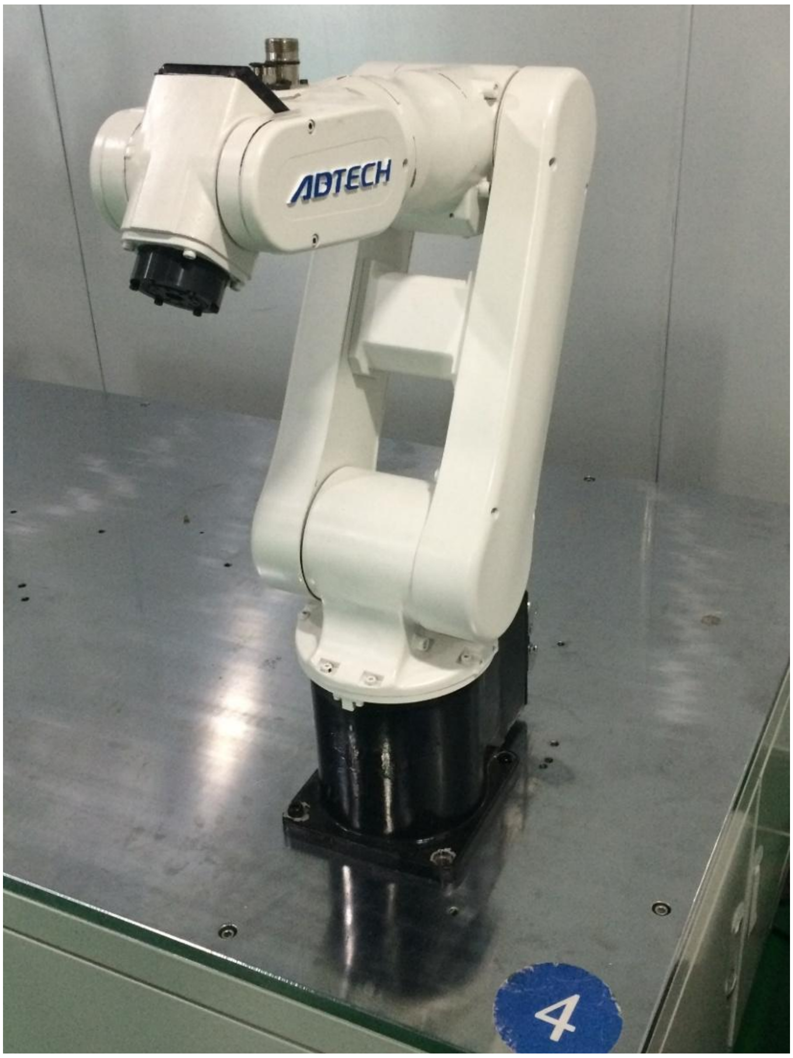


Figure 6-1 Mounting rack of typical robot body

⚠ **Attention:**

① Do not weld the electrical devices (including the robot). The current passing through the motor encoder and the controller is too high, and may cause failure. If the welding is required, please disassemble the robot and the controller first.

② The robot will apply large counterforce on the installation bench when it is operating in high velocity. To avoid the bench vibrating or deviating due to counterforce, the bench should have sufficient rigidity. It is also an effective method to connect other heavy devices to the robot bench.

③ When the robot is operating, the bench may send resonant sound (hoot). When the resonant sound is increased, please improve the rigidity of the bench or decrease the velocity of the robot.

6.2 Installation method of the robot body

[1] **Installation of the robot body**

The installation operations of the robot body can be referred to steps listed as follows, we hereby explain on the condition that the mounting seat of the robot is prepared well and on which screw holes have been bored for fixation of the robot.

Step 1 Check if the package is in good condition, open the packing box to ensure whether or not the robot is intact;

Step 2 Unscrew clamping screw to take down the robot body from bracket mount of the wooden case;

Step 3 Move the robot onto the bench of the robot, align each mounting hole to fix temporarily with 4 screws;

Step 4 Fix the robot with 4 screws(M10×25 mm) and flat washer;

Step 5 Fastening torque of the screw: 50 ± 5N·m.

⚠**Notes:**

(1) Safety Precautions must be read carefully when the robot is moved and installed;

(2)The arm of the robot shall be supported to prevent tilting when fix screws are unscrewed.

(3) Installation must be operated by at least 2 people. The weight of SD series robot body is about 28kg;

(4)The operators must wear safety helmet, shoes, protective goggles and gloves.

[2] **Ground connection of the robot body**

Over 5.5 mm^2 wiring is required for the earth terminal of robot body in ground connection.

⚠**Note: The earth wire and electrode shall use special products, which shall not be combined to use with other electric, powered and welding machines.**

6.3 Installing method of robot controller

Move the robot controller to the proper position of mounting table, and plug the heavy-duty connector male of cables connecting between the robot body and controller to the corresponding female on the controller.

6.4 Connection between robot body and controller

Robot body and controller will be delivered upon matching debugging, and there is a corresponding number on the nameplate. In case of purchasing more than one robot, do not exchange the combinations of robot body and controller.

⚠**Note: The serial number of robot body and controller shall be the same combination.**

6.5 Electrical wiring and air tubing methods of robot body

For the electrical wiring and air tubing of jigs and tools installed in the robot wrist, refer to the empty area reserved on the robot body for installation.

6.6 Installing method of work fixture

SD series robot wrist joints are arranged with tool flange mounting holes. Users are required designing the flanges of working fixture. It is to recommend M3 fixing screws to mount the grippers, suction cups or other tool on the tool mounting holes. There’s air inlet, outlet and corresponding cable head reserved for the grippers in the SD series robot, as shown in Figure 3-10, but these shall be done for the convenience of internal alignment, while avoiding danger due to the wound of tool accessory with the robot, so users are required making connections according to their needs.

7. Robot maintenance

7.1 Category and Objective of Maintenance Spot Inspection

Please perform the maintenance spot inspection described in the table below.

Table 7-1 Category and Objective of Maintenance Spot Inspection

No.	Category	Objective
1	Daily	To use the robot properly, please perform spot inspection every day before operating.
2	Three months	To maintain the precision of the robot and avoid failures caused by overheating of the controller, perform the spot inspection every three months.
3	Six months	To avoid burning and damage that may cause major accidents, perform spot inspection for the rotary and sliding parts of the robot every six months.
4	Two years	Spot inspection of synchronous belts

⚠ **Attention**

The maintenance spot inspection is performed in the motion space of the robot mostly. Since the probability of accidents is high, please follow local laws and regulations and ask qualified personnel to operate. Before operating, please read “1. Safety Precautions” and “2. Operating Precautions” in chapter I and this chapter carefully.

7.2 Operation matters of spot inspections

The spot inspections can be conducted in daily, 3-month, 6-month and 2-year manners. Please complete the spot inspections based on the particulars as listed in tables below.

Table 7-2 Daily Spot Inspection

No.	Spot inspection position or running	Power status of the controller	Spot inspection method	Standard	Treatment of problem
1	Connectors and connected objects	OFF	Visual	No loose, pulling off or dirt	Insert normally and clear

2	Cables and external cable of the robot	OFF	Visual	No damage or crack	Repair, replace
3	Multifunction demonstrator	ON	Visual	Display	Repair, replace
4	Emergency stop button of multifunction demonstrator	ON	Press the emergency stop button	Emergency stop	Repair, replace
5	Motor ON LED in robot body	ON	Visual inspection	LED lights up when the motor is ON	Spot inspection and repair

Table 7-3 3-month spot inspection

No.	Spot inspection position or running	Power status of the controller	Spot inspection method	Standard	Treatment of problem
1	Fixation of base screw of robot	OFF	Determine the tightening torque with torque wrench	No specified torque loosened: 35 ± 8N·m	Fastening with the predetermined torque

Table 7-4 Yearly spot inspection

No.	Item of spot inspection	Power status of the controller	Spot inspection method	Standard	Treatment of problem
1	Encoder batteries	OFF	Measure battery voltage with multimeter	The battery voltage is not lower than 3.2V	Replace the 3.6V battery
2	Robot structure	OFF	Visual inspection	No damage to the structural surface, and no significant gap between motion axle	Contact our customer service department for maintenance

Chapter 2 Basic Operation

1. Changes in specification of robot

1.1 Meaning of changing robot specifications

The software that controls the manipulator uses the motion space of the manipulator as the top limit. In this space, the motion limit can be determined freely. The motion limit in software is called software limit, and changing standard settings is called changing manipulator specifications. To avoid interference with other device and entangling of wiring and pipes of the fixtures, it is required to set appropriate motion limit.

1.2 Software limit

1.2.1 Meaning of software limit

Software limit refers to determining the robot movement range limit by software. The robot is active only after entering the range set by the software limit. The mechanical movement limit is called manipulator end which is set by the mechanical stop. All A1, A2, A3, A5 and A6 axes of SD series of robots exist with mechanical terminal limit with buffer. Software limit is set at the slightly front position of the manipulator end as shown in the figure below during factory leaving, so as to prevent conflict with the mechanical stop. During manual operation and automatic movement of the robot, when it reaches the software limit, there will be certain error message displayed (see the software operation description for the details), with the running stopped. Software limit is separately set on the positive direction side and negative direction side of the movement range of all axes. Soft limit on the positive direction side is called positive-direction software limit; and soft limit on the negative direction side is called negative-direction software limit.

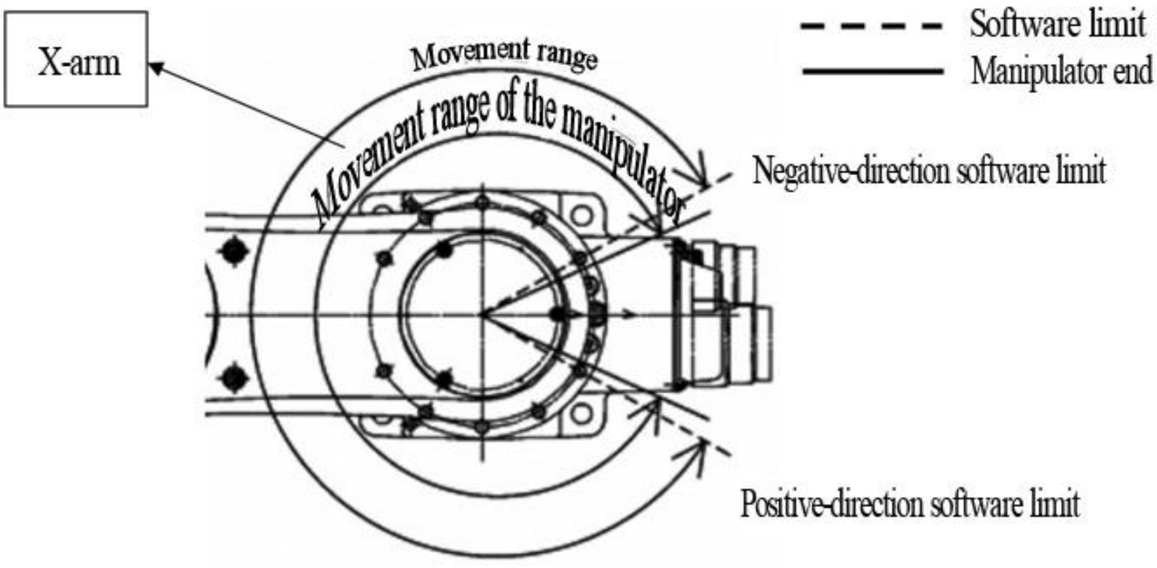

X-arm
Movement range
Movement range of the manipulator
Software limit
Manipulator end
Negative-direction software limit
Positive-direction software limit

Fig. 1-1 Schematic diagram of movement range of software limit of the robot

1.2.2 Setting value of the software limit during factory leaving

Currently the setting value of the software limit during factory leaving of SD series is the movement range value of each axis.

1.2.3 Change of the software limit

Please change the software limit and narrow the movement range when the robot interferes with other device, and also when the air piping and wring for fixture jib is stretched due to robot movement.

⚠**Attention: During changing software limit, the movement range of the robot to set must be the range within the initial value.**

1.2.4 Notes during changing software limit

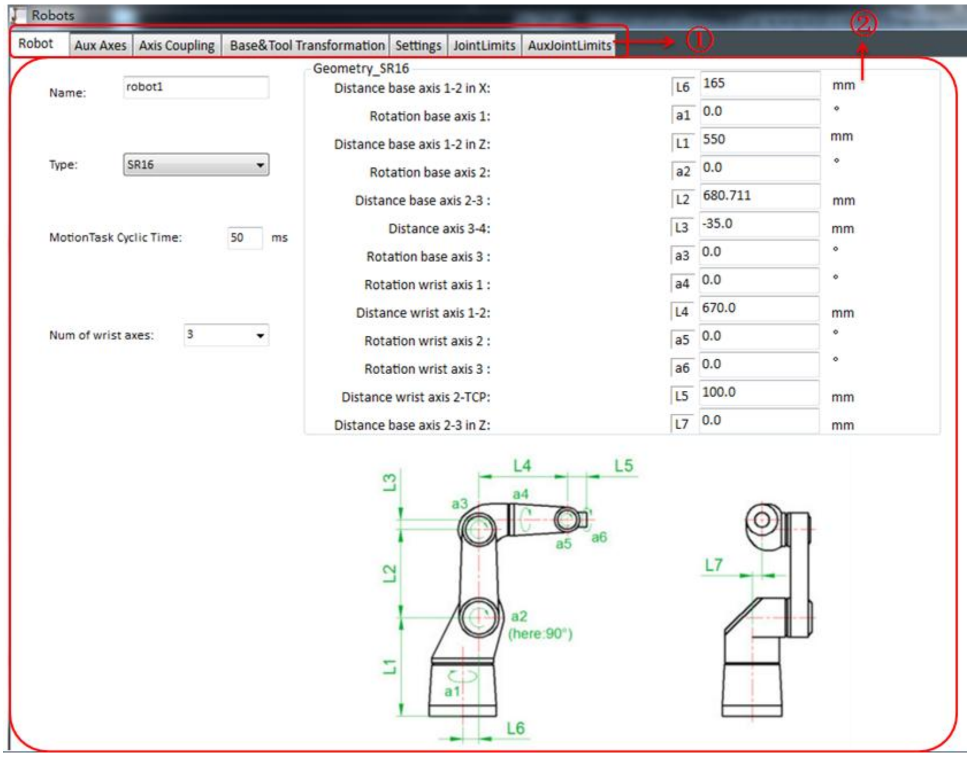
1) Please confirm the movement range of the robot in the actual operating environment.

2) Please note not to mistake the unit.

3) Situation of robot not running will occur if wrongly narrow the movement range to be too small.

1.2.5 Steps of changing the software limit

Change of the software limit of SD series of robots is completed via the special configuration software RobotConfig through which the configuration file (.xml) is created to finally cover the old file in the controller. Double click RobotConfigure.exe to open the robot configuration software as shown in Fig. 1-2.



Name: robot1

Type: SR16

MotionTask Cyclic Time: 50 ms

Num of wrist axes: 3

Geometry_SR16

Distance base axis 1-2 in X:	L6	165	mm
Rotation base axis 1:	a1	0.0	°
Distance base axis 1-2 in Z:	L1	550	mm
Rotation base axis 2:	a2	0.0	°
Distance base axis 2-3 :	L2	680.711	mm
Distance axis 3-4:	L3	-35.0	mm
Rotation base axis 3 :	a3	0.0	°
Rotation wrist axis 1 :	a4	0.0	°
Distance wrist axis 1-2:	L4	670.0	mm
Rotation wrist axis 2 :	a5	0.0	°
Rotation wrist axis 3 :	a6	0.0	°
Distance wrist axis 2-TCP:	L5	100.0	mm
Distance base axis 2-3 in Z:	L7	0.0	mm

Fig. 1-2 Robot configuration main interface

Area 1 in the figure is the menu bar, including Robot, Aux Axes, Axis Coupling, Base&ToolTransfromation, Settings, JointLimits and AuxJointLimits, wherein, JointLimits is used to configure the software limit of the robot axes, invert direction or not and the maximum dynamic information (including maximum velocity, limited velocity, maximum acceleration, maximum deceleration and maximum jerk). Area 2 in the figure is the interface area which displays various configuration information and will be detailed successively below. Each menu page in Area 1 in the figure is described as below:

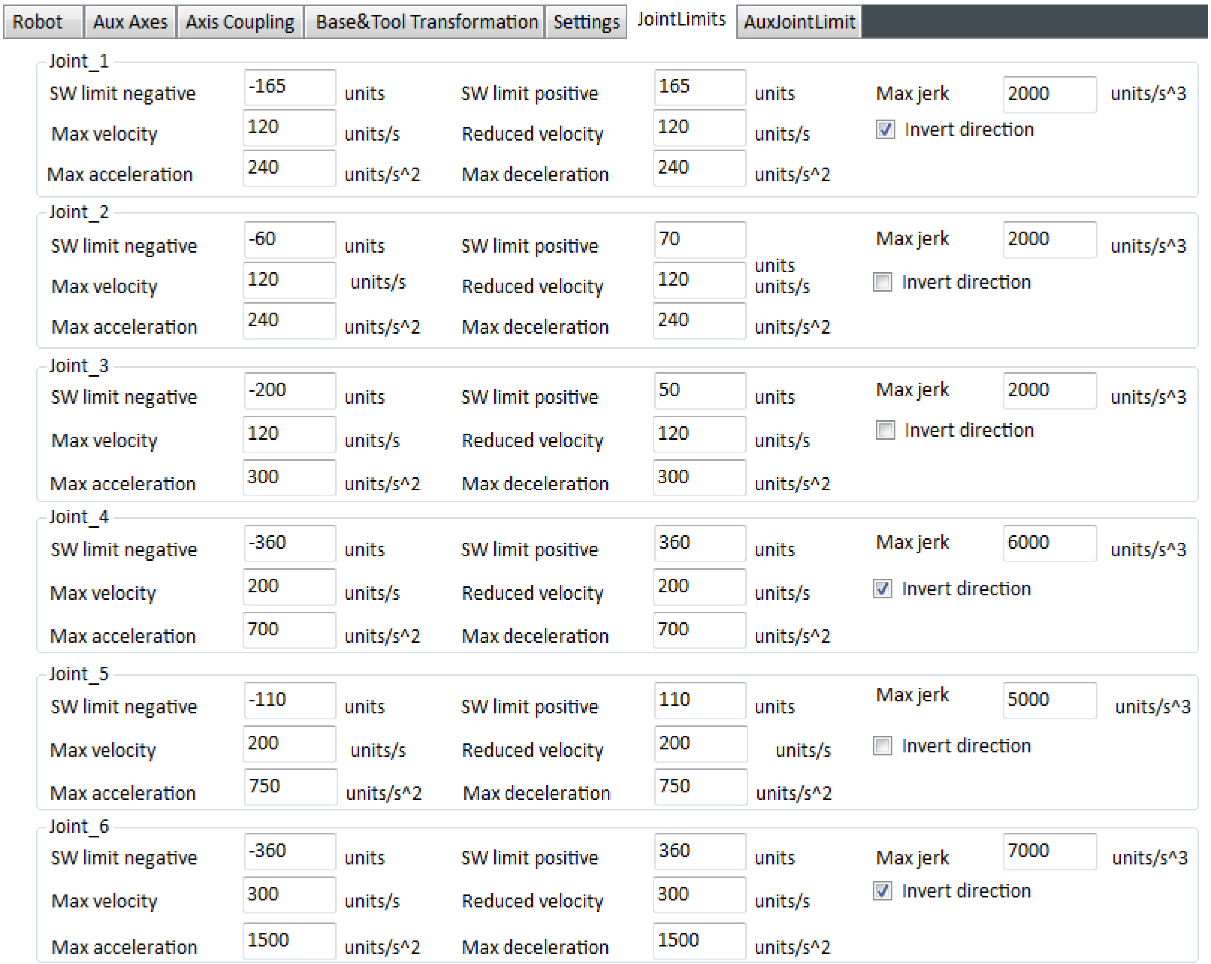
- Robot: for configuring basic parameters of the robot
- AuxAxes: for configuring the additional axes system of the robot
- Axis Coupling: for configuring the coupling parameters of the robot axes
- Base&ToolTransfromation: for configuring the robot base coordinate system and tool coordinate system
- Settings: for configuring the robot movement parameters
- JointLimits: for configuring the robot axis limit
- AuxJointLimits: for configuring the additional axis limit

⚠**Attention:**

1. **If only partial information is completed configuration and saved during the configuration, and the software is closed, then reconfiguration is required during next configuration. The software temporarily does not feature the file loading function, thus reconfiguration is required per time;**
2. **The Save button is in lower right corner of "AuxJointLimit" interface. After click of Save, the software automatically creates XML file of the configured information at a time. Note that the file name to save must be the default name;**
3. **As there are many configuration parameters of the robot and partial parameters involve the robot structure and running performance, such shall not be modified without authorization. Please contact the customer service personnel for robot of our company for conducting configuration if needed.**

There are many configuration parameters of the robot, and before factory leaving of the product, all parameters have been completely configured and imported to the controller without requiring users for configuration. Only software limit configuration is introduced here. Please contact our company for configuring other parameters and having professionals conduct the operation. Configuration operation steps of software limit are as below:

1) Click "JointLimit" on the menu bar to pop up the interface for configuring robot axis limit as shown in Fig. 1-3; meaning of each physical quantity is as below:
   - ➢ SW limit negative: negative limit of the axis;
   - ➢ SW limit positive: positive limit of the axis;
   - ➢ Max velocity: maximum velocity, unit: units/s;
   - ➢ Reduced velocity: maximum limited velocity, unit: units/s;
   - ➢ Max acceleration: maximum acceleration, unit: units/s^2;
   - ➢ Max deceleration: maximum deceleration, unit: units/s^2;
   - ➢ Max jerk: maximum jerk, unit: units/s^3;
   - ➢ Invert direction: invert direction or not. Invert direction means that actual rotation of the robot axis is opposite to the axis direction in "Robot" configuration interface. If the actual rotation of the robot axis is opposite to the axis direction in "Robot" configuration interface, then this check box shall be selected.



Joint_1

SW limit negative	-165	units	SW limit positive	165	units	Max jerk	2000	units/s^3
Max velocity	120	units/s	Reduced velocity	120	units/s	☑ Invert direction		
Max acceleration	240	units/s^2	Max deceleration	240	units/s^2			

Joint_2

SW limit negative	-60	units	SW limit positive	70	units	Max jerk	2000	units/s^3
Max velocity	120	units/s	Reduced velocity	120	units/s	□ Invert direction		
Max acceleration	240	units/s^2	Max deceleration	240	units/s^2			

Joint_3

SW limit negative	-200	units	SW limit positive	50	units	Max jerk	2000	units/s^3
Max velocity	120	units/s	Reduced velocity	120	units/s	□ Invert direction		
Max acceleration	300	units/s^2	Max deceleration	300	units/s^2			

Joint_4

SW limit negative	-360	units	SW limit positive	360	units	Max jerk	6000	units/s^3
Max velocity	200	units/s	Reduced velocity	200	units/s	☑ Invert direction		
Max acceleration	700	units/s^2	Max deceleration	700	units/s^2			

Joint_5

SW limit negative	-110	units	SW limit positive	110	units	Max jerk	5000	units/s^3
Max velocity	200	units/s	Reduced velocity	200	units/s	□ Invert direction		
Max acceleration	750	units/s^2	Max deceleration	750	units/s^2			

Joint_6

SW limit negative	-360	units	SW limit positive	360	units	Max jerk	7000	units/s^3
Max velocity	300	units/s	Reduced velocity	300	units/s	☑ Invert direction		
Max acceleration	1500	units/s^2	Max deceleration	1500	units/s^2			

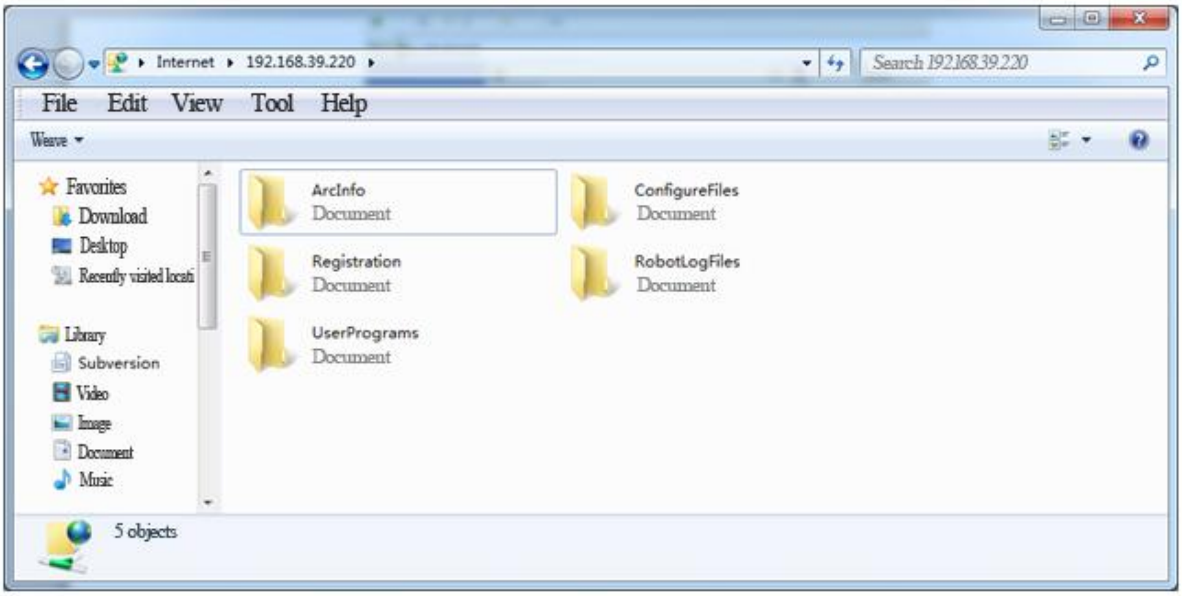
Fig. 1-3 Configuration interface of robot axis limit

2) Modify the values of SW limit negative and SW limit positive of each axis as required, and click the Save button in lower right corner of “AuxJointLimit” interface to create RobotParameter.xml file;

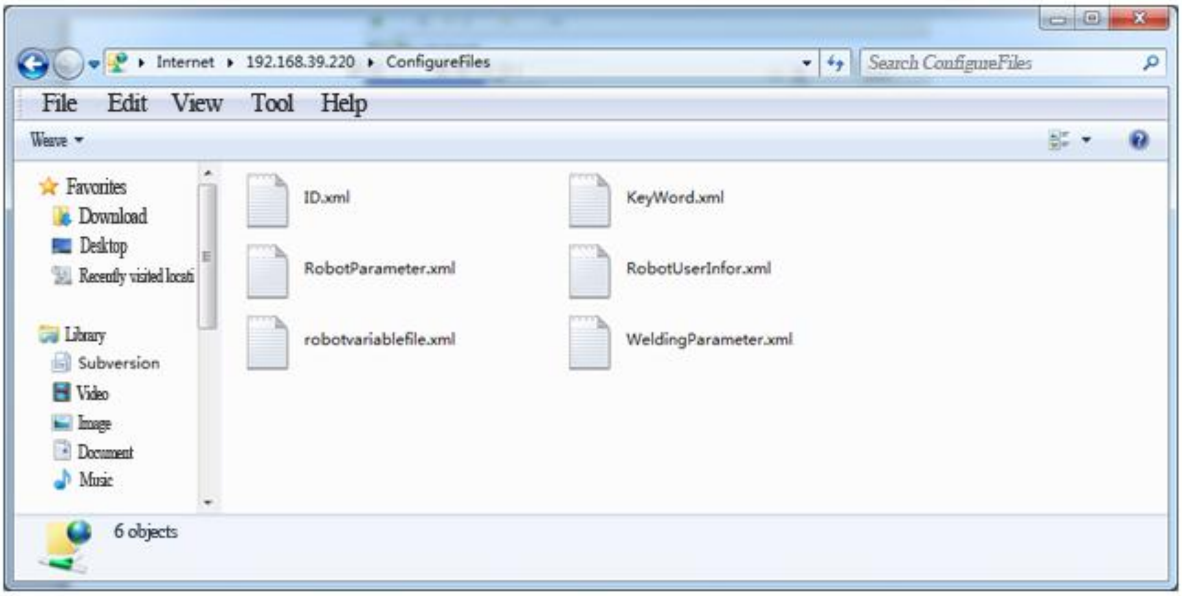
3) After click of Save, the software automatically creates XML file of the configured information at a time. Note that the file name to save must be the default name;

4) Connect local computer with the network interface 1 (the interface in lower left corner of the controller) on the controller;

5) Enter the following content on the computer address bar: **ftp://192.168.39.220/**, and click “Enter” on the keyboard to pop up the interface as shown in the figure below;



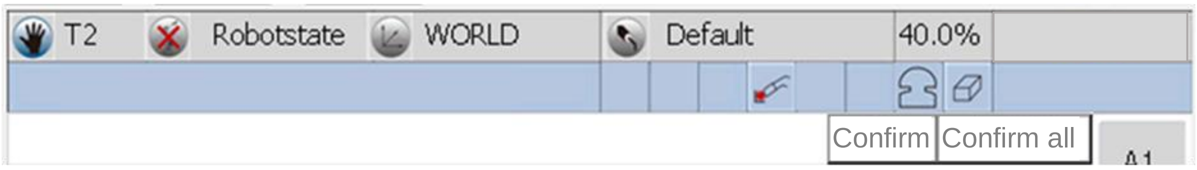
6) Double click ConfigureFiles folder in ftp to pop up the figure as shown in the figure below



7) Then replace the RobotParameter.xml file therein with the new file.

2. Status Bar of the Teaching Box

The status bar of the teaching box is mainly used to intuitively display some current internal statuses of the robot.



Schematic diagram of the status bar

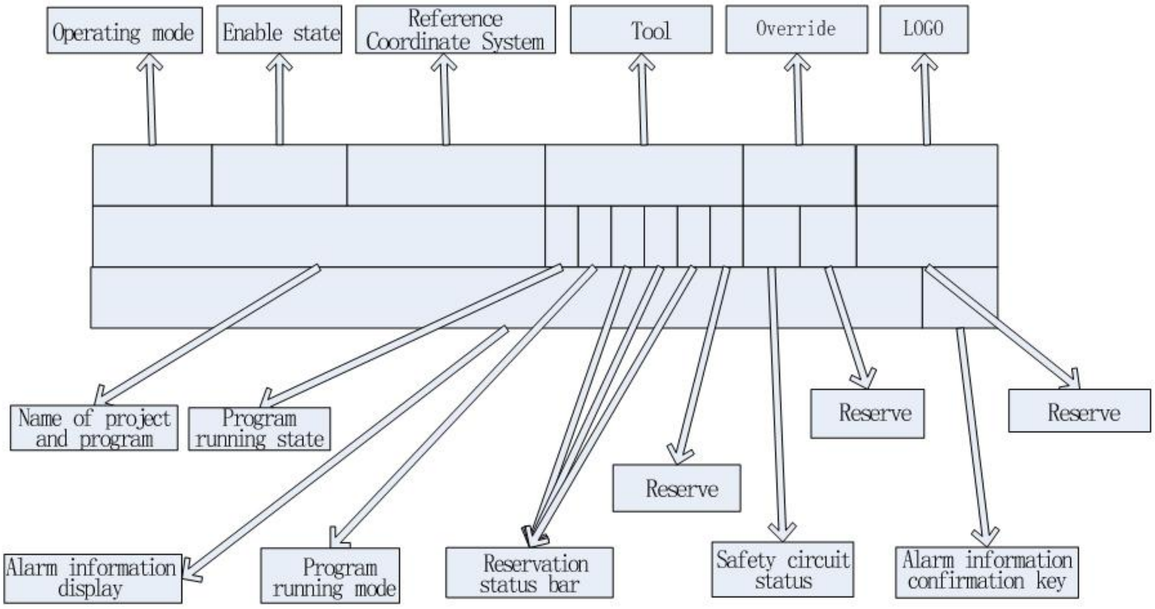
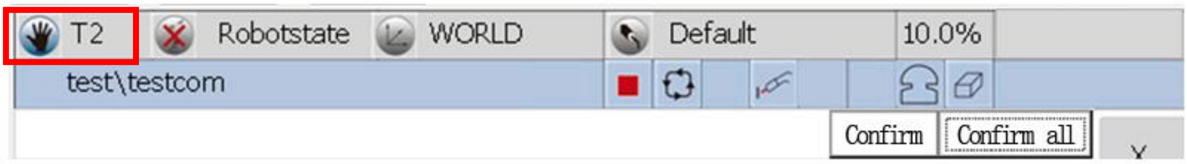


Fig. 2-1 Layout of the status bar

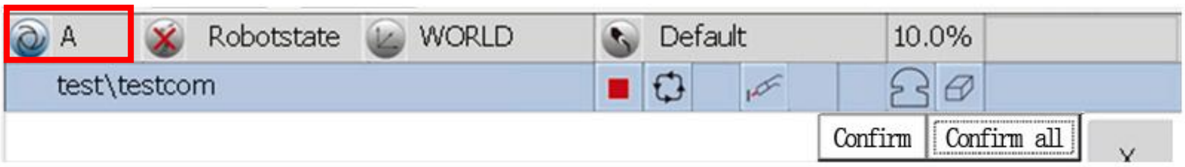
2.1 Running mode

Operation mode: three in total, controlled via key switch.

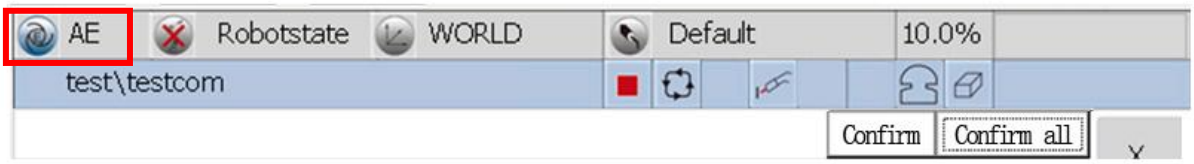
Manual mode: the robot can be controlled to move by hand, in a relatively slow velocity, as shown in the figure below



Automatic mode: the robot automatically runs in a relatively fast velocity, as shown in the figure



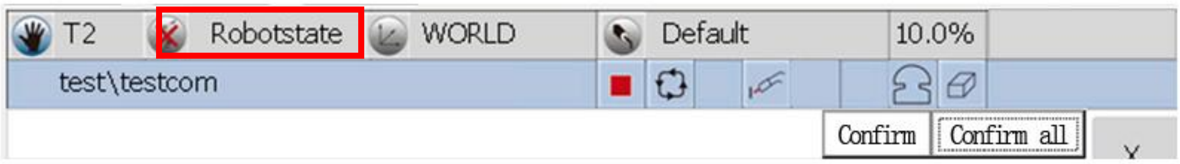
External automatic: the robot is controlled to move by external equipment, which is suitable for engineering project, as shown in the figure



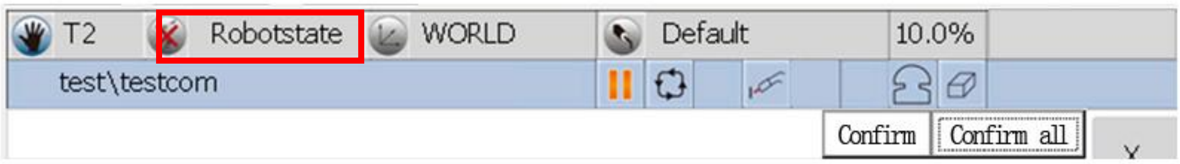
2.2 Enabling status

Function: for viewing if the robot is enabled.

Icon ☒ means the robot enabling is off.



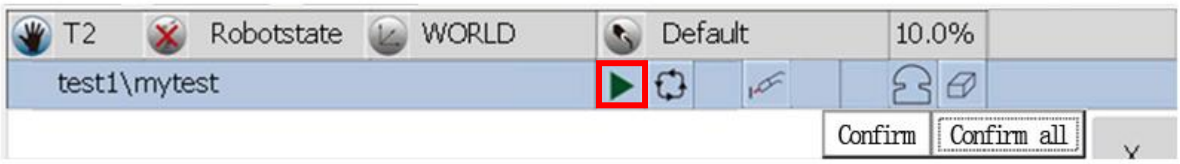
Icon ✓ means the robot enabling is on.



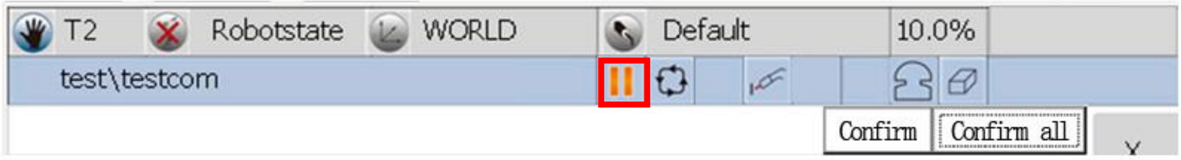
2.3 Program running status

The program running status includes three statuses (Run, Pause and Stop).

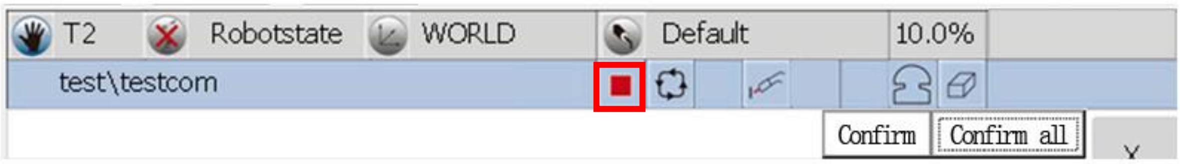
Icon ▶ means the robot is in Run status, as shown in the figure



Icon ⏸ means the robot is in Pause status, as shown in the figure



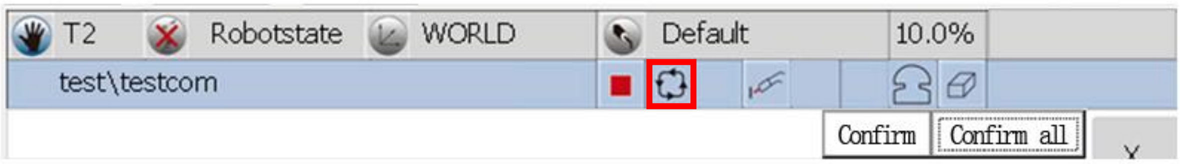
Icon 🟥 means the robot is in Stop status, as shown in the figure



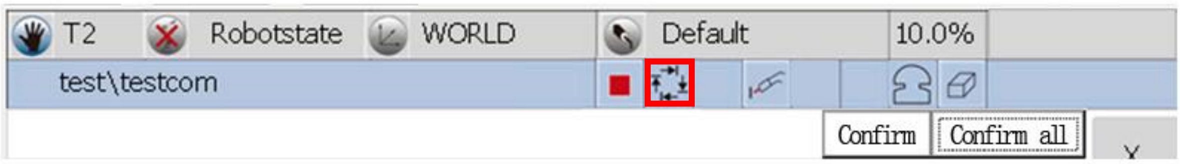
2.4 Program running mode

The program running mode includes three modes (Continuous, Single-step, and Movement single-step)

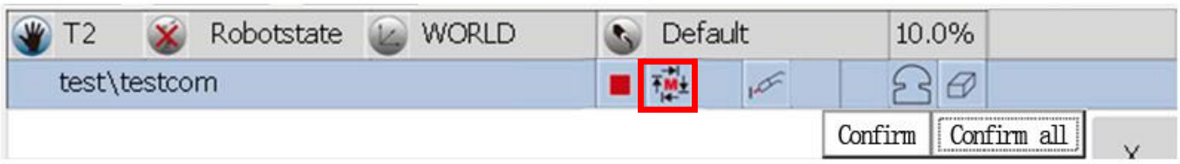
Icon means the robot running mode is Continuous, as shown in the figure



Icon means the robot running mode is Single-step, as shown in the figure

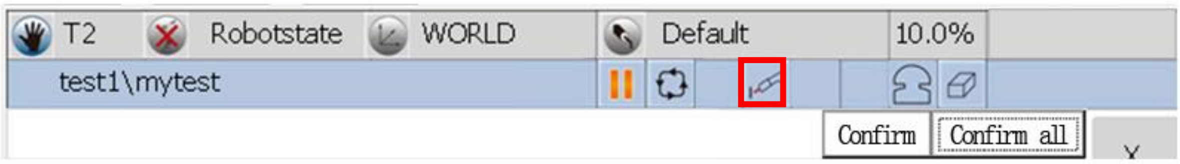


Icon M means the robot running mode is Movement single-step, as shown in the figure

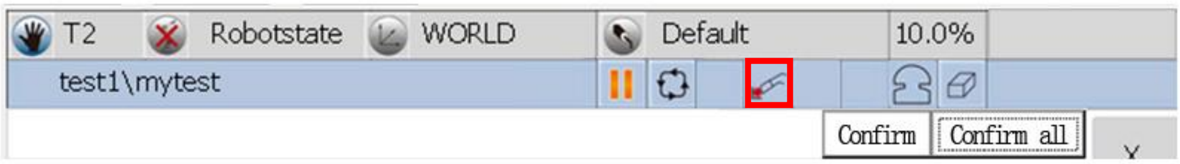


2.5 Welding simulation

Icon means the robot conducts welding simulation, as shown in the figure

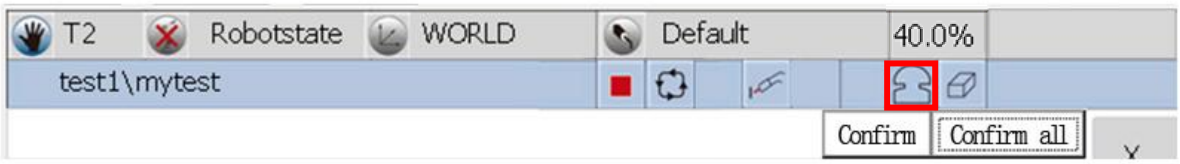


Icon means the robot conducts welding operation, as shown in the figure

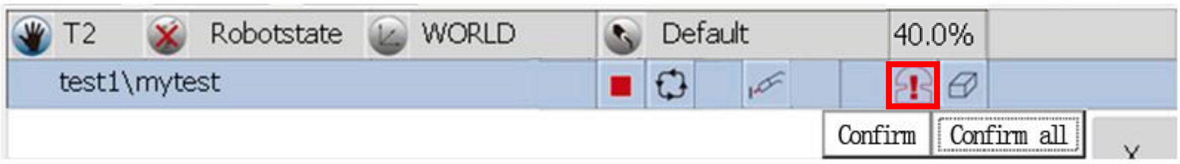


2.6 Emergency stop button status

Icon means the emergency stop button is not pressed, as shown in the figure



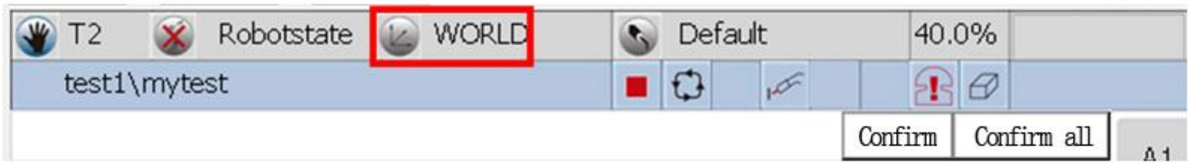
Icon means the emergency stop button is pressed, as shown in the figure



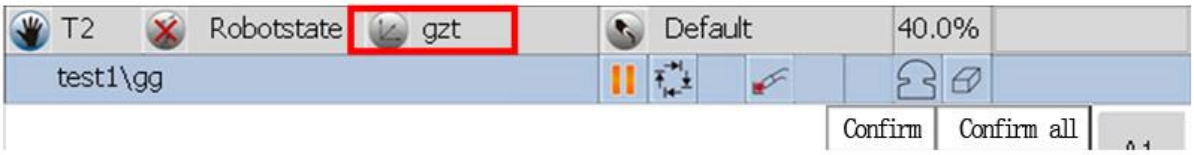
2.7 Reference coordinate system

It is used to display the reference coordinate system of the current robot system. Switching of the reference coordinate system is required to be set in the program. The figure below shows switching of different coordinate systems.

World coordinate system:



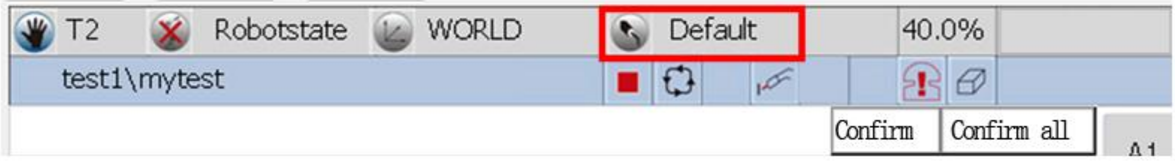
Custom coordinate system:



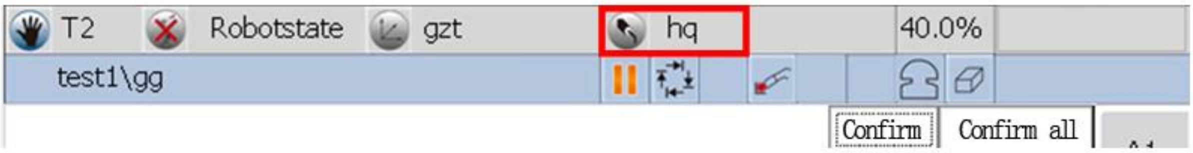
2.8 Tool

It is used to display the internal tool of the current robot system. There are two ways to set the robot system tool: set in the robot program or set through the tool setting drop-down box in the Robot Position interface. The figure below shows switching of different tools.

Status of no tool:

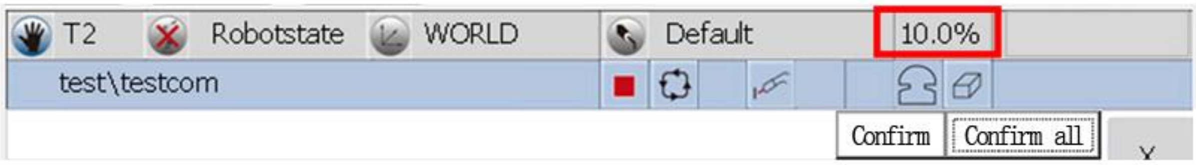


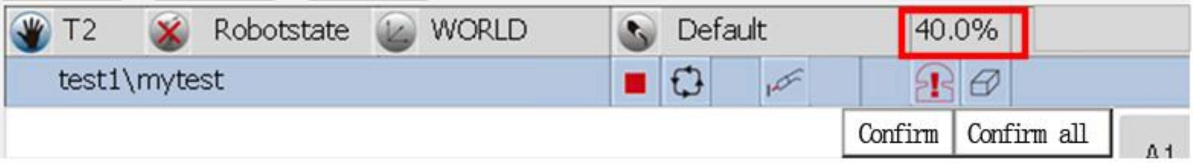
Custom tool:



2.9 Velocity

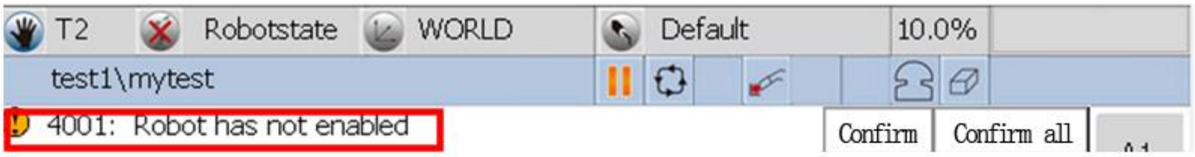
It is used to show the current robot running velocity parameter. There are three ways to set the velocity: set in the program, set through the "Velocity setting" in the Robot Position interface, or set through the panel buttons "V+" and "V-". The figure below shows different velocities:





2.10 Alarm information display

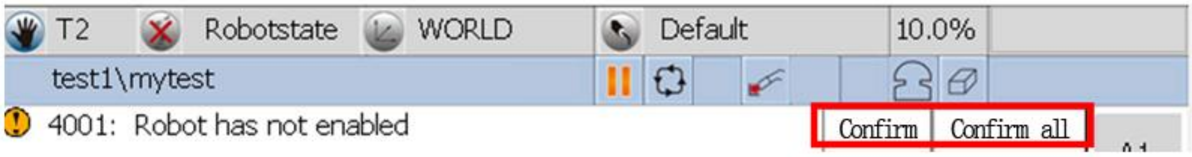
It is used to display the internal alarm information (three kinds: error, alarm and message) of current robot system, as shown in the figure



4001: Robot has not enabled

2.11 Alarm information confirmation button

It is used to confirm the internal alarm information of the robot system. The robot can be enabled again only after all internal alarm information of the system is confirmed.



4001: Robot has not enabled

3. Operation of the Function Interface

3.1 Setting interface

3.1.1 Description of the user authority

User groups are specified here as four groups: Low Level, Middle Level, High Level and Administration, corresponding to 1, 2, 3 and 4 in the Group column in the following table. 1 in Control authority column means requiring control authority, and 0 means requires no control authority. User authority is described as below:

- ➢ Administration>High Level > Middle Level > Low Level
- ➢ The higher level authority can conduct operation of the lower level authority.

- ● Description of authority of Low level users:

Level 1 module	**Level 2 module**	**Group**	**Control authority**
1. Status bar			
	1.1 Status bar display	1	0
2. Error, alarm and message display	2.1 Page display	1	0
	2.2 Information screening display	1	0
3.Report interface display	3.1 Page can be displayed or not	1	0
	3.2 Information screening display	1	0
4. Robot Position interface	4.1 Page display	1	0
	4.2 TCP displays selection of reference coordinate system	1	0
6.IO display	6.1 Page display	1	0
7. System management setting screen	7.1 User login screen display	1	0
	7.2 User login	1	0
	7.3 Control authority be released	1	0
	7.4 Page display of current login user	1	0

	7.5Interface display of version information	1	0
	7.6 Interface display of Reset/Report	1	0
	7.7 Interface display of Sys screen	1	0
8. Variable monitoring and editing screen	8.1 Page display	1	0
	8.9 Variable data display	1	0
12. Project interface	12.13 Program directory display	1	0
	12.14 Display of loaded program	1	0
	12.15 Display of opened program	1	0
	12.16 Page display	1	0
14. Peripheral button	14.1 Interface switching button	1	0

- Description of authority of Middle Level users

Level 1 module	**Level 2 module**	**Group**	**Control authority**
1. Status bar	1.1 Confirm error, alarm and message	2	1
2. Error, alarm and message display			
	2.2 Confirm single piece	2	1
	2.4 Confirm all information	2	1
4. Robot Position interface			
	4.2 Set jog coordinate system	2	1
	4.3 Set override	2	1

12. Project interface	12.9 Load	2	1
	12.10 Open	2	1
	12.11 Kill program	2	1
	12.12 Close program	2	1
13. Program editing interface	13.1 Page display	2	0
	13.5 Program running mode	2	1
14. Peripheral button			
	14.2 Operation mode	2	1
	14.3 MOT	2	1
	14.4 jog	2	1
	14.5 step	2	1
	14.6 v- v+	2	1
	14.7 start	2	1
	14.8 stop	2	1
	14.9 Axis+, and axis-	2	1
	14.10 2nd	2	1

- Description of authority of High Level users

Level 1 module	**Level 2 module**	**Group**	**Control authority**
4. Robot Position interface	4.4 Tool setting	3	1
5. Tool Manual Alignment interface	5.1 Page can be displayed or not	3	1
	5.2 Tool alignment	3	1
	5.3 Enable tool alignment	3	1
	5.4 Cancel tool alignment	3	1
6.I/O display			

	6.2 DI (analog input)	3	1
	6.3 Remove DI	3	1
	6.4 D0 (forced output)	3	1
	6.5 Remove DO	3	1
	6.6 AI (analog input)	3	1
	6.7 Remove AI	3	1
	6.8 A0 (forced output)	3	1
	6.9 Remove AO	3	1
7. System management setting screen	7.11 Restart teaching box	3	1
	7.12 Restart controller	3	1
8. Variable monitoring and editing screen			
	8.2 New variable	3	1
	8.3 Rename variable	3	1
	8.4 Cut variable	3	1
	8.5 Copy variable	3	1
	8.6 Paste variable	3	1
	8.7 Delete variable	3	1
	8.8 Teach variable	3	1
	8.10 Modify variable	3	1
9.Positions interface	9.1 Page display	3	1
	9.2 Select variable	3	1
	9.3 Teach variable	3	1
	9.4 Select jog to position mode	3	1
	9.5 Enable jog to position	3	1
	9.6 Cancel jog to position	3	1
10. Custom coordinate system	10.1 Page display	3	1
	10.2 Select variable of	3	1

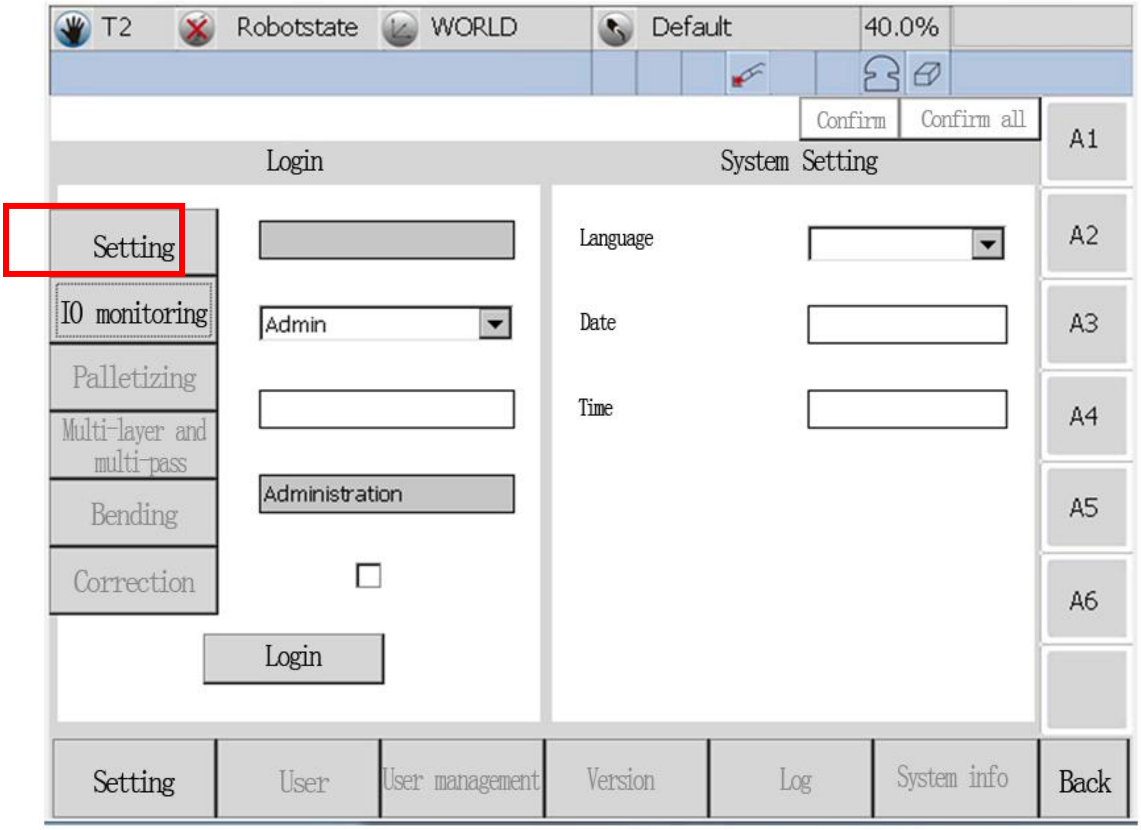
	reference coordinate system		
	10.3 Select reference coordinate system	3	1
	10.4 SETUP button	3	1
11. Tool interface	11.1 Page display	3	1
	11.2 Select tool variable	3	1
	11.3 SETUP button	3	1
12. Project interface	12.1 Rename project program	3	1
	12.2 Delete project program	3	1
	12.3 Paste project program	3	1
	12.4 Copy project program	3	1
	12.5 New program	3	1
	12.6 New project	3	1
	12.7 Import	3	1
	12.8 Export	3	1
13. Program editing interface			
	13.2 MODIFY	3	1
	13.3 New	3	1
	13.4 Setpc	3	1
	13.6 copy	3	1
	13.7 paste	3	1
	13.8 cut	3	1
	13.9 remove	3	1

- Description of authority of Administration users

Level 1 module	Level 2 module	Group	Control authority
7. System management setting screen	7.5 User management page display	4	0
	7.6 Edit user	4	1
	7.7 New user	4	1
	7.8 Delete user	4	1
	7.13 Create (create report)	4	1
	7.14 export (export report)	4	1
	7.16 Modify and set IP connection	4	1

3.1.2 User login

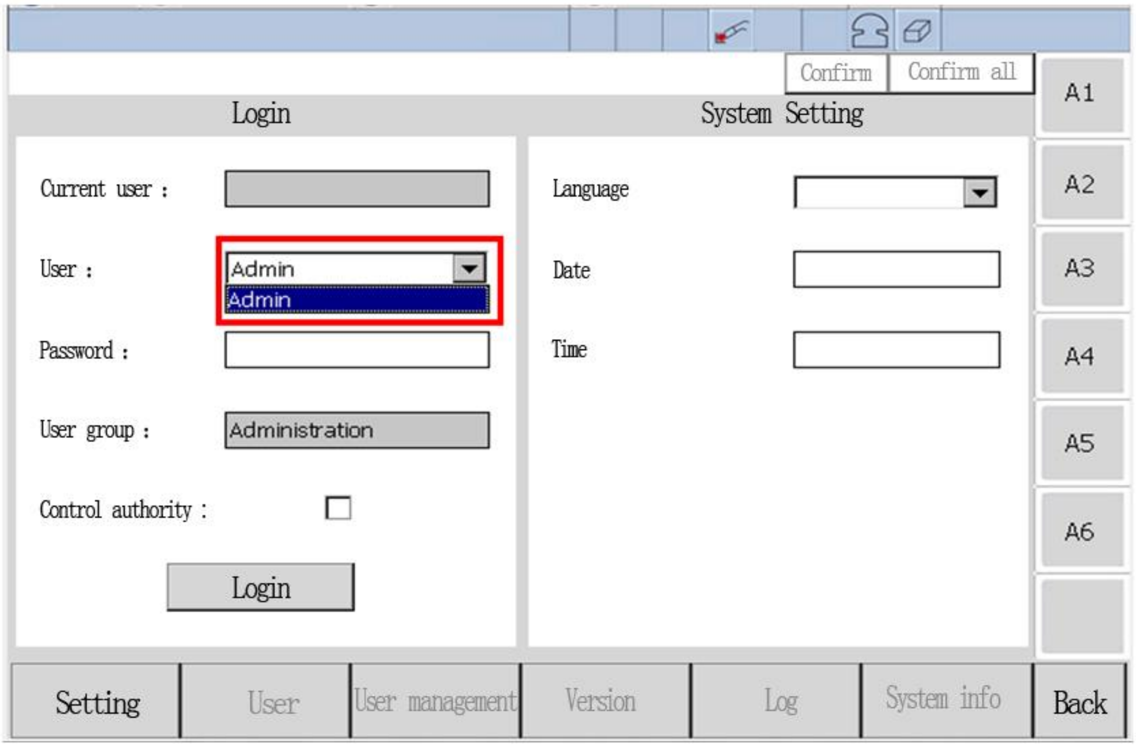
- Click 🔧 button and click "Setting" to enter the user management interface as shown in the figure



Admin

Administration

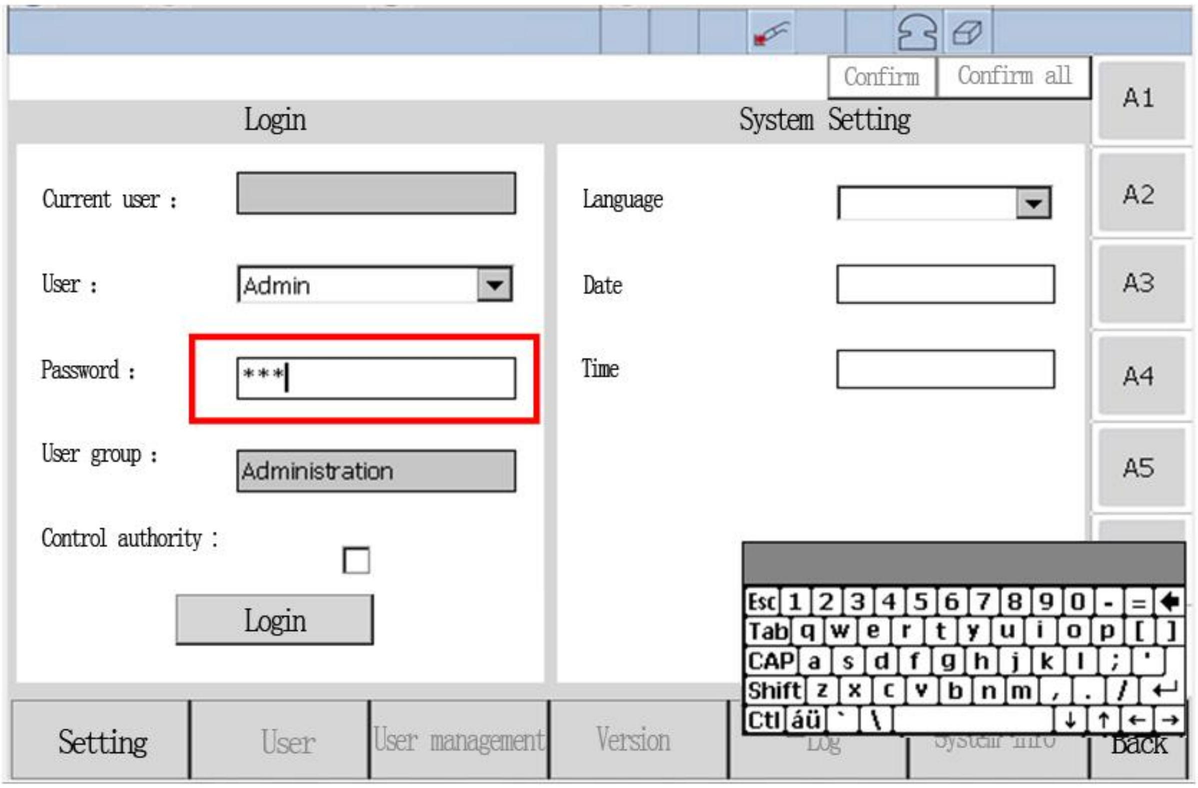
- Select the user name as shown in the figure



User : Admin

User group : Administration

- Enter the password as shown in the figure

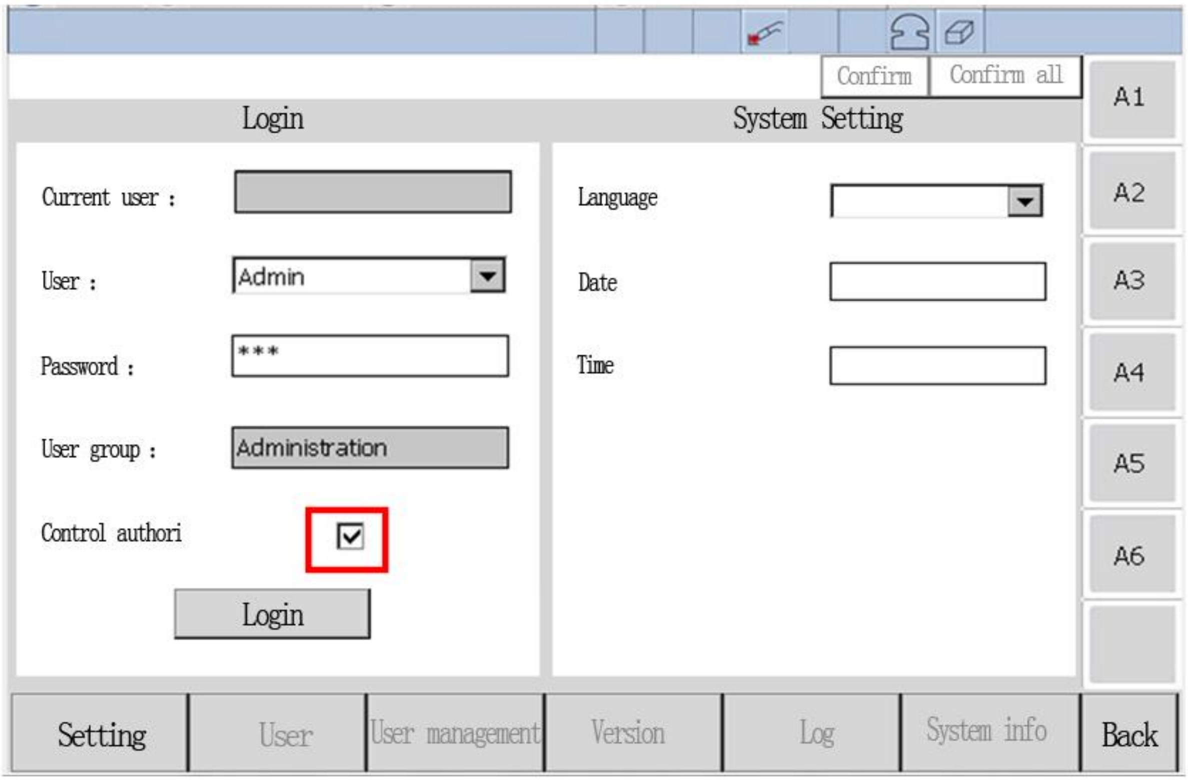


User : Admin

Password : ***

User group : Administration

- Check the control authority to obtain the authority, as shown in the figure



Current user :

User : Admin

Password : ***

User group : Administration

Control authori ☑

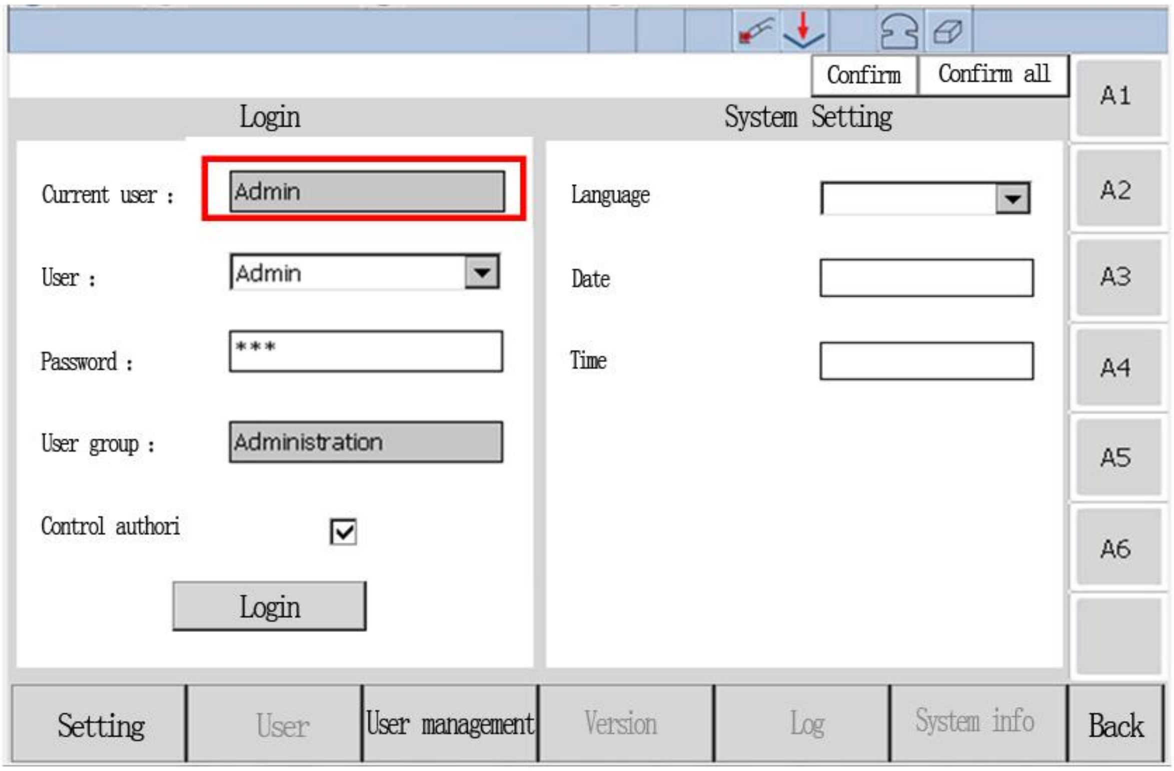
Language

Date

Time

Attention: "Control Authority" is used to obtain and release control authority: when checked, it means that the control authority is obtained; when unchecked, it means that the control authority is released. If the control authority of current controller is obtained by other PLC, the "Control Authority" checking will be unavailable.

- Click Login to obtain the authority for operation. The interface after login is as shown in the following figure. When the username is displayed in "Current user", it means that the login is successful.



Login

Current user： Admin

User： Admin

Password： ***

User group： Administration

Control authori ☑

Login

System Setting

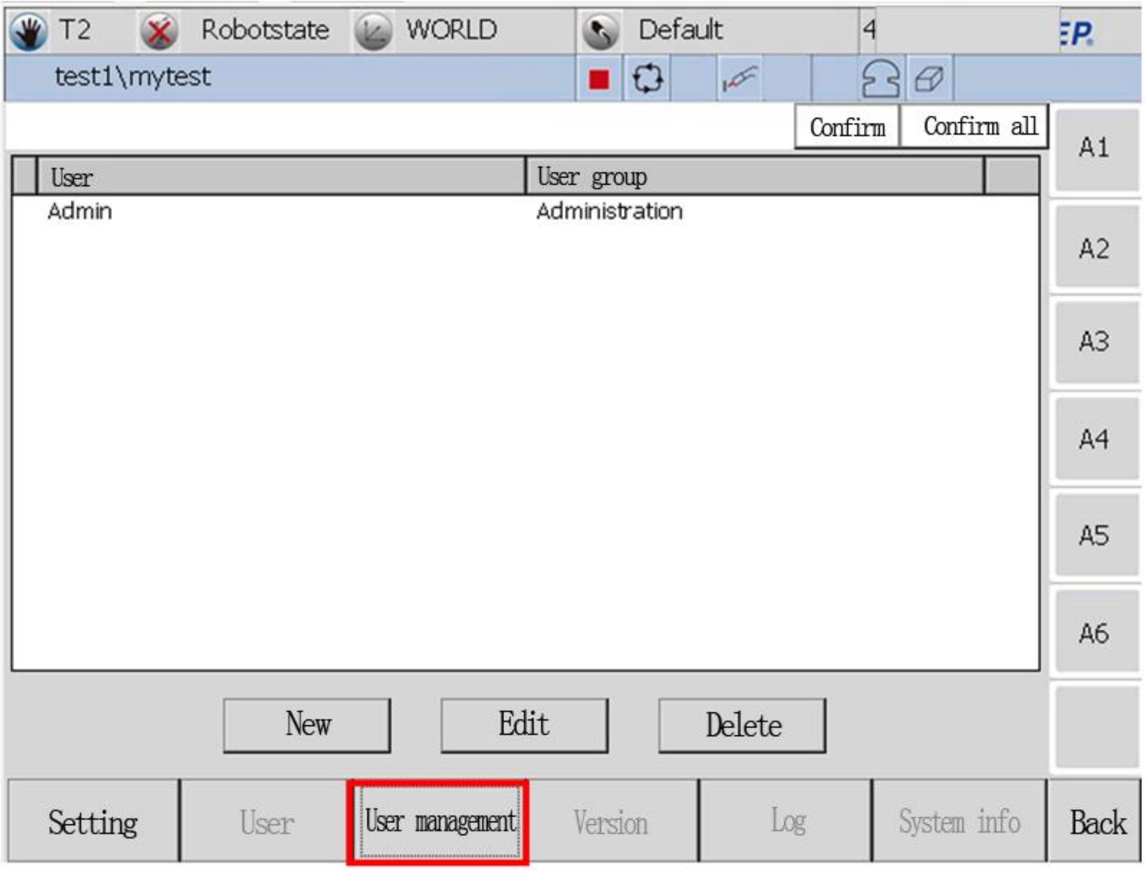
Language

Date

Time

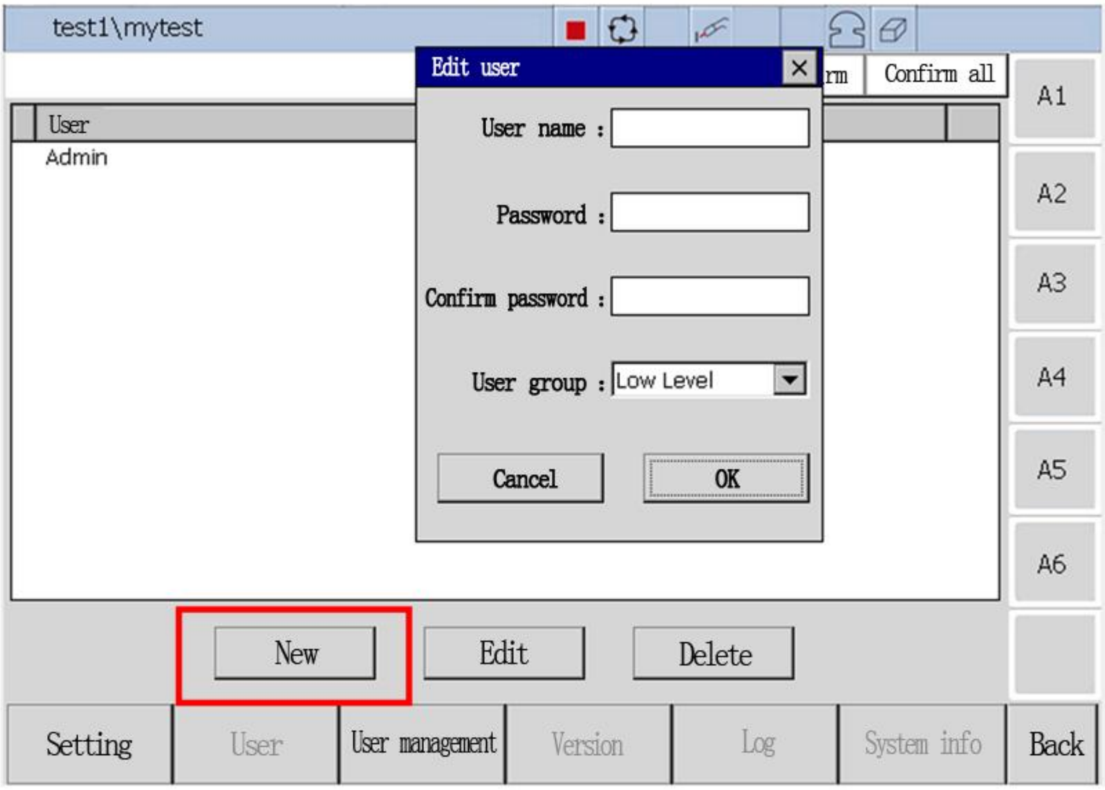
3.1.3 Adding user

- Click User management, as shown in the figure

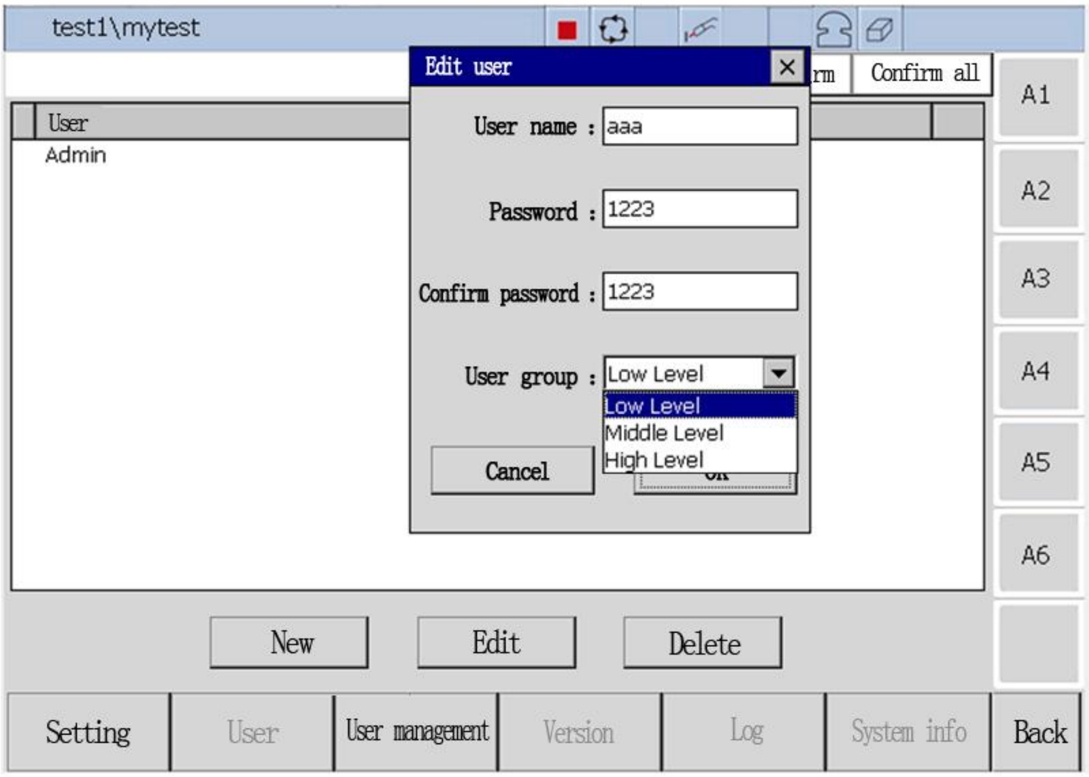


User	User group
Admin	Administration

- Select New as shown in the figure



- Enter the username, password, confirm password, and select the user group information, as shown in the figure

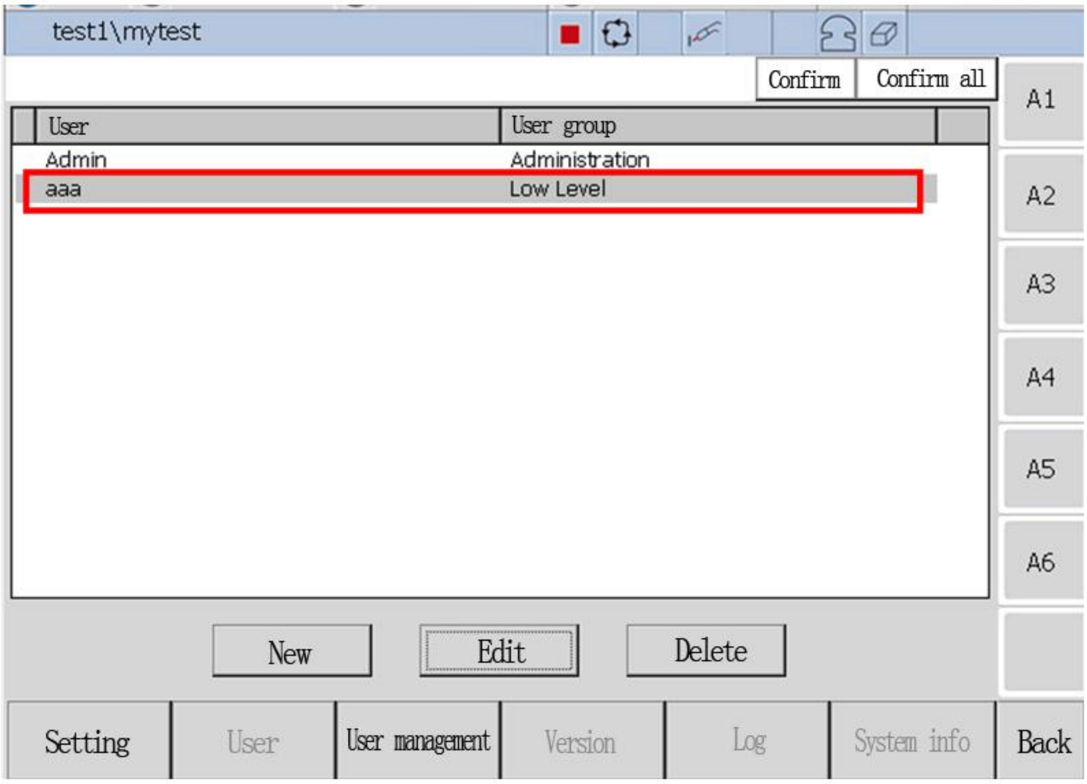

User name : aaa

Password : 1223

Confirm password : 1223

User group : Low Level

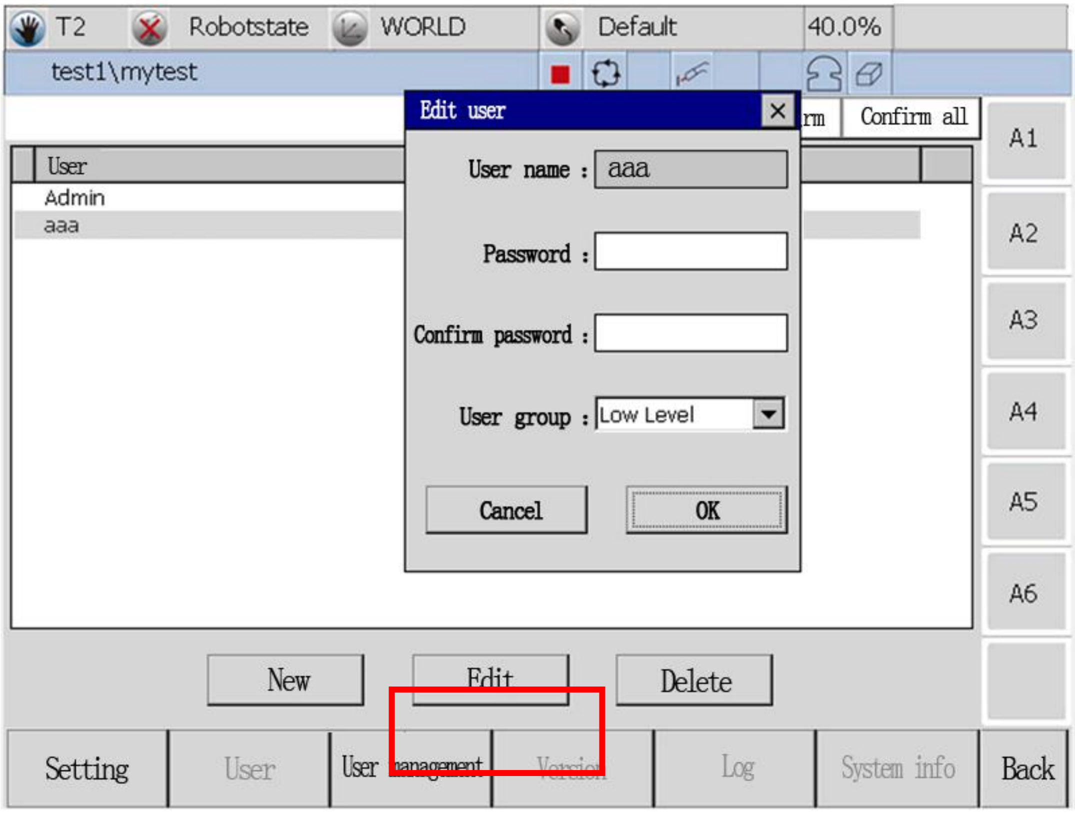
- Click OK to add new user, as shown in the figure



User	User group
Admin	Administration
aaa	Low Level

3.1.4 Editing user

- To edit user, select the user to edit and click Edit, as shown in the figure



User name : aaa

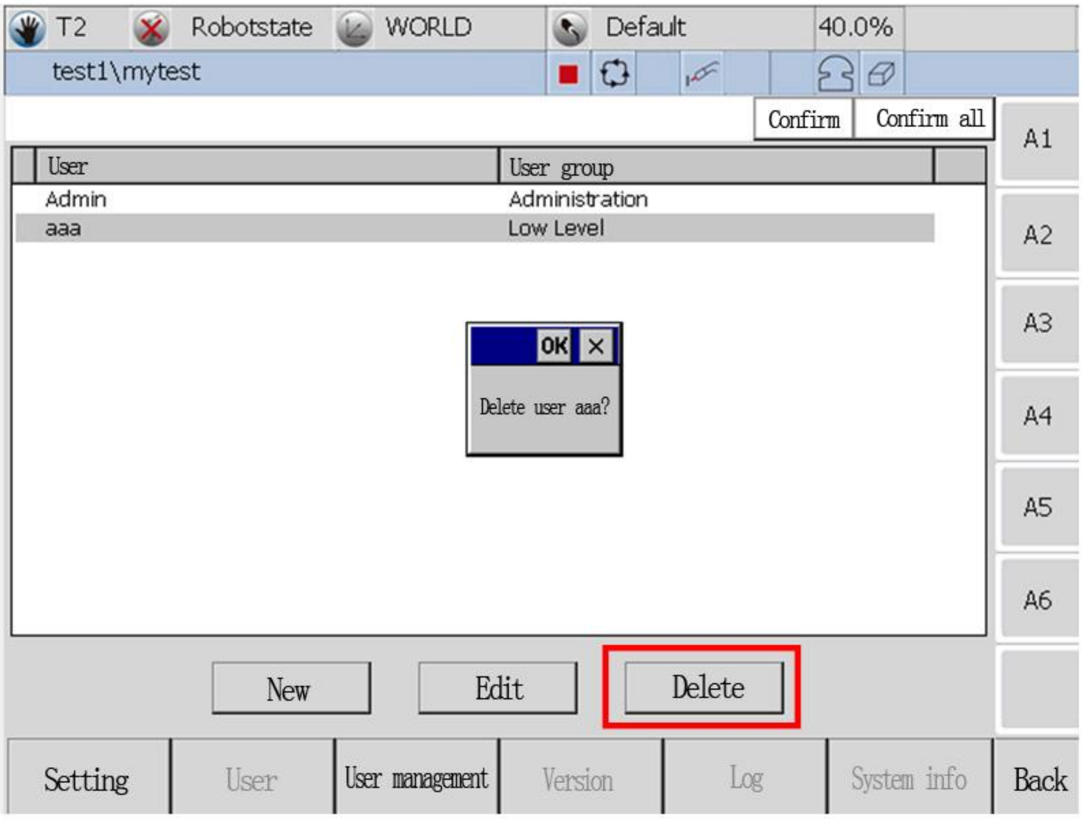
Password :

Confirm password :

User group : Low Level

3.1.5 Deleting user

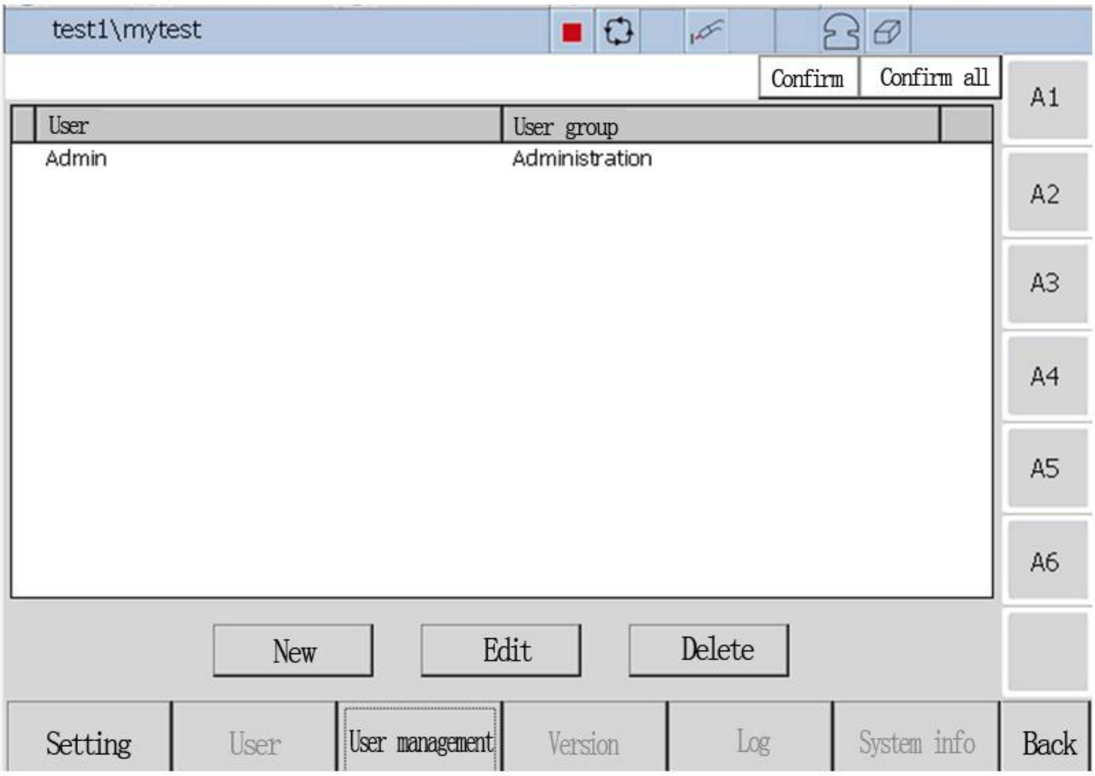
- To delete user, click Delete, as shown in the figure



User	User group
Admin	Administration
aaa	Low Level

Delete user aaa?

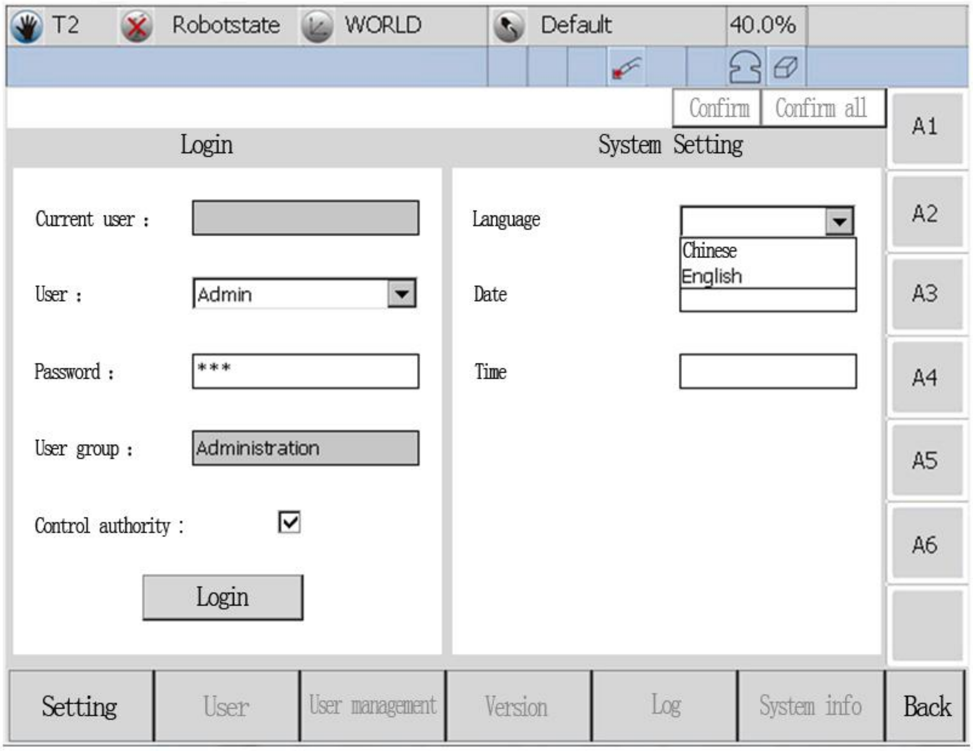
- Click OK to successfully delete the user, as shown in the figure



User	User group
Admin	Administration

3.1.6 Choosing language

- Choose the language, as shown in the figure



T2 Robotstate WORLD Default 40.0%

Confirm Confirm all

Login

Current user :

User : Admin

Password : ***

User group : Administration

Control authority : ☑

Login

System Setting

Language

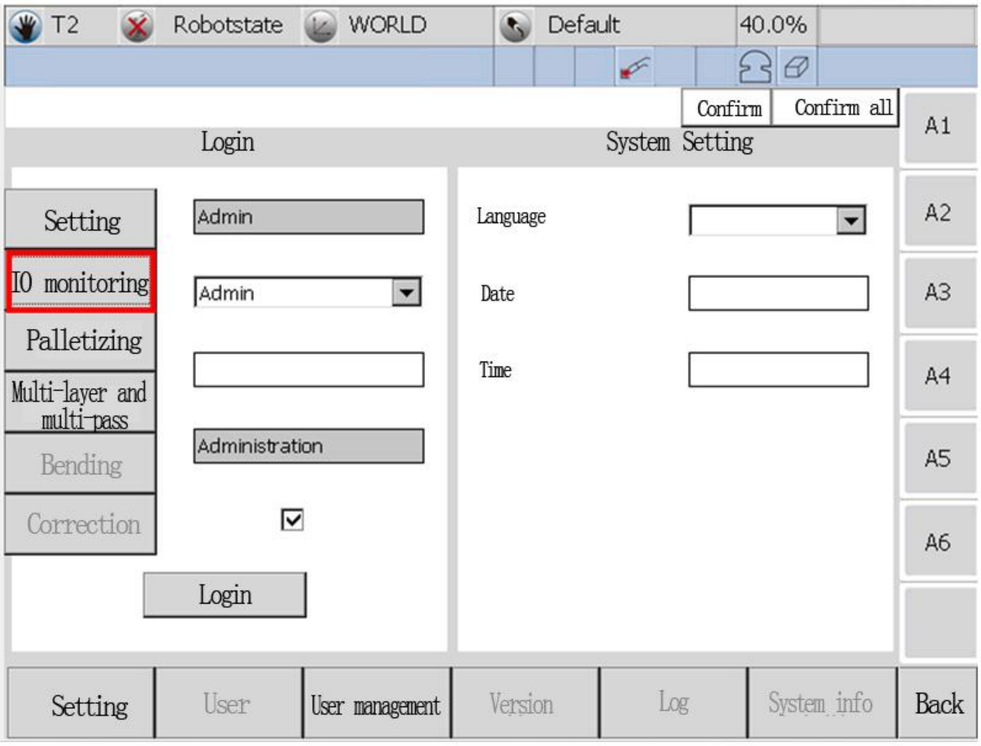
Chinese
English

Date

Time

3.2 IO monitoring

Click button, as shown in the figure



T2 Robotstate WORLD Default 40.0%

Confirm Confirm all

Login

Setting

IO monitoring

Palletizing

Multi-layer and multi-pass

Bending

Correction

Admin

Admin

Administration

☑

Login

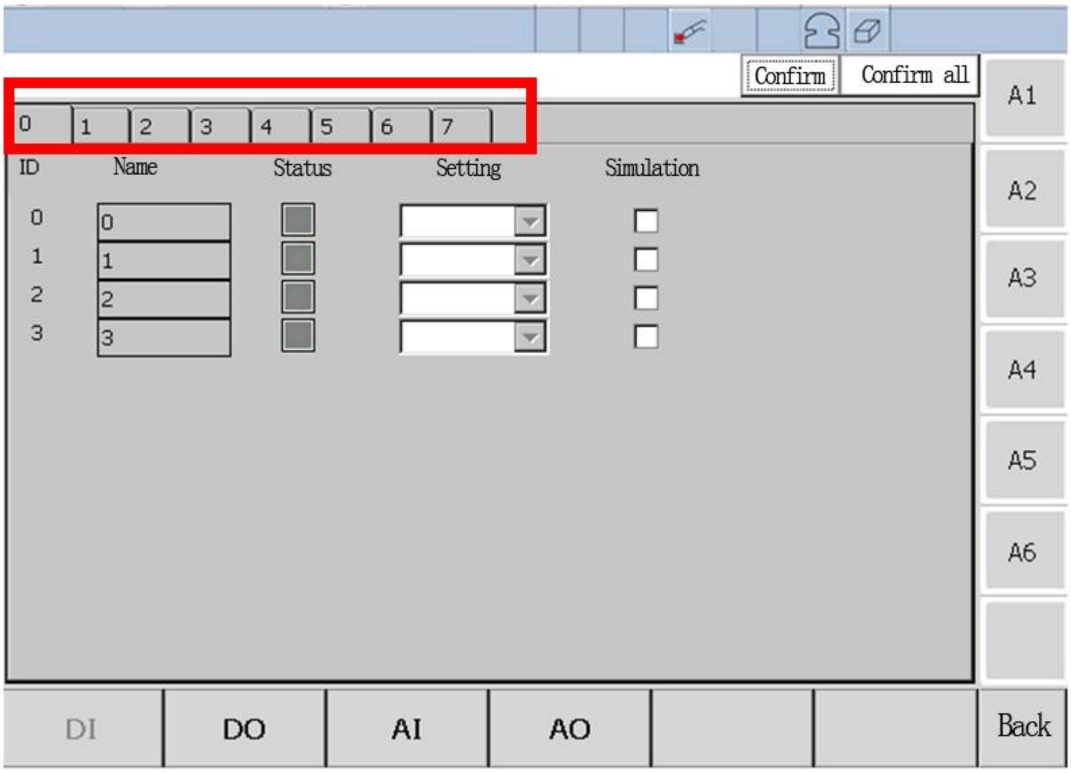
System Setting

Language

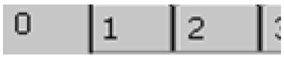
Date

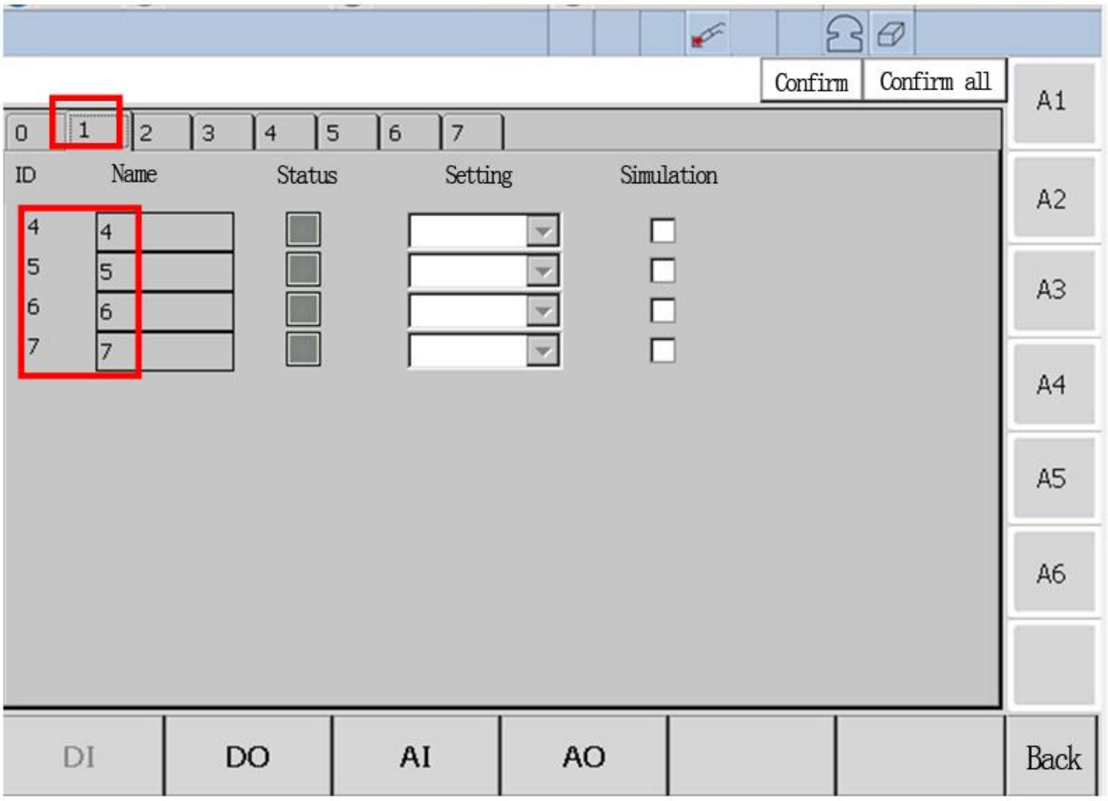
Time

Click “IO monitoring” to enter the initialization interface, as shown in the figure

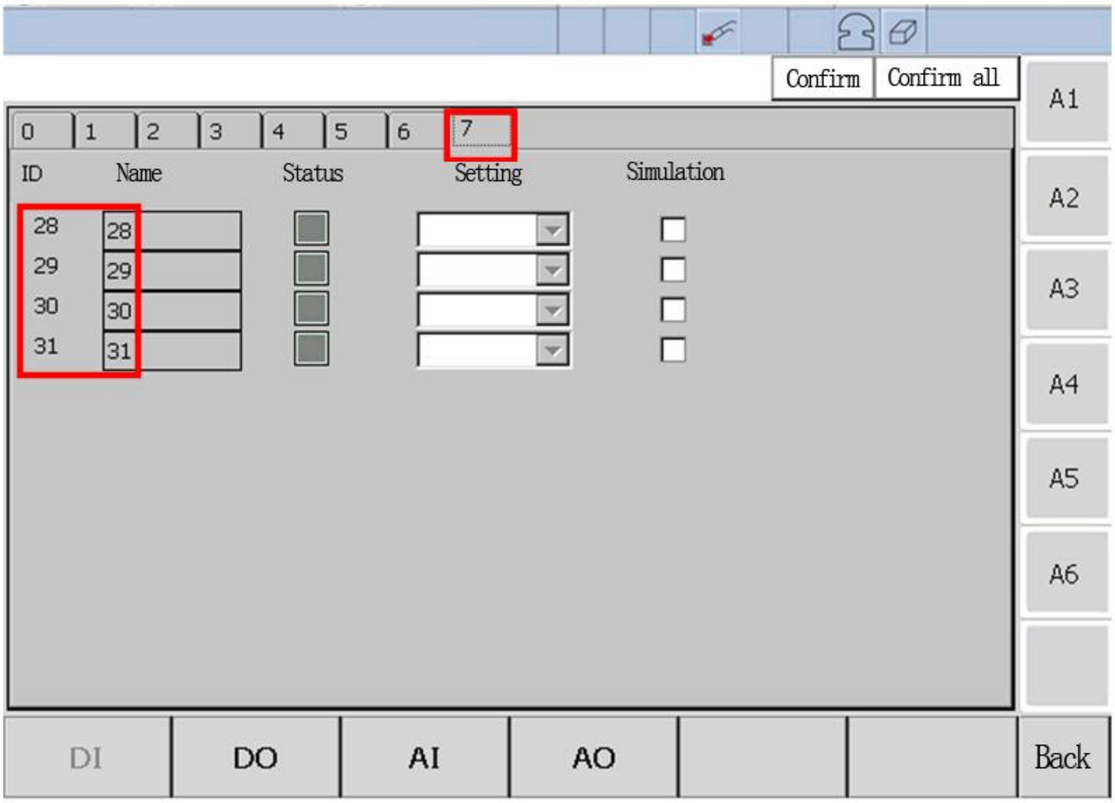


ID	Name	Status	Setting	Simulation
0	0	▨		□
1	1	▨		□
2	2	▨		□
3	3	▨		□

Click the upper tabs in the interface  to enter different IO point interfaces, as shown in the figure



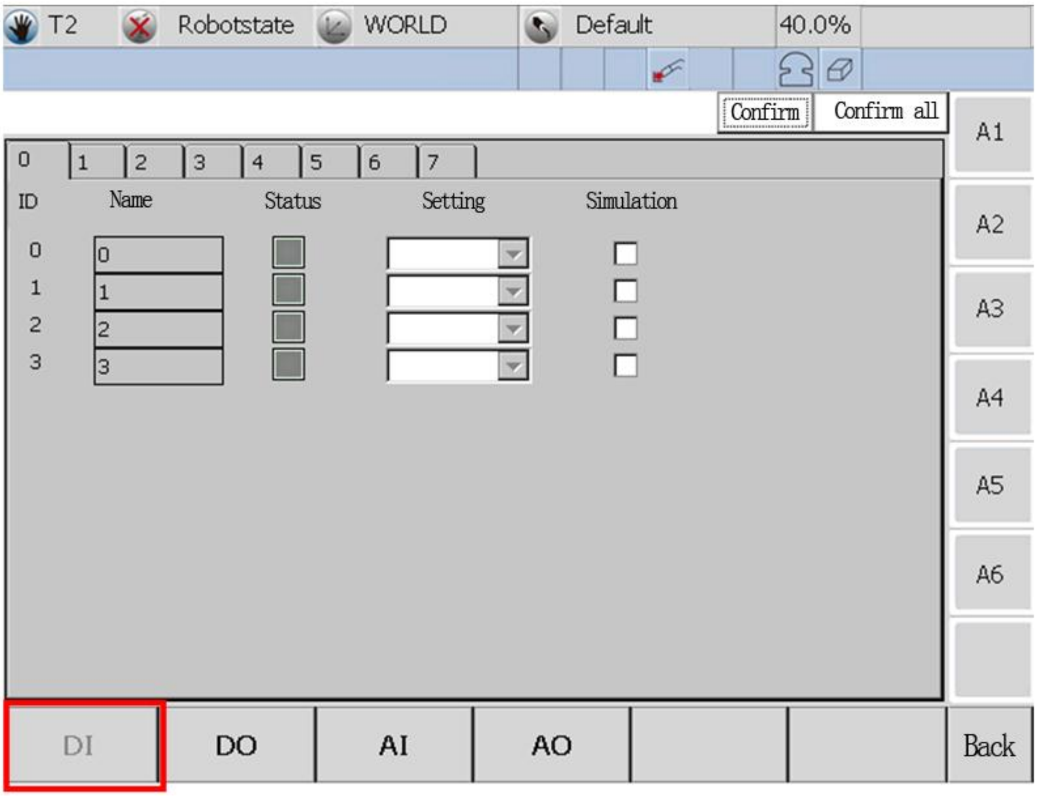
ID	Name	Status	Setting	Simulation
4	4	▨		□
5	5	▨		□
6	6	▨		□
7	7	▨		□



ID	Name	Status	Setting	Simulation
28	28	▨		□
29	29	▨		□
30	30	▨		□
31	31	▨		□

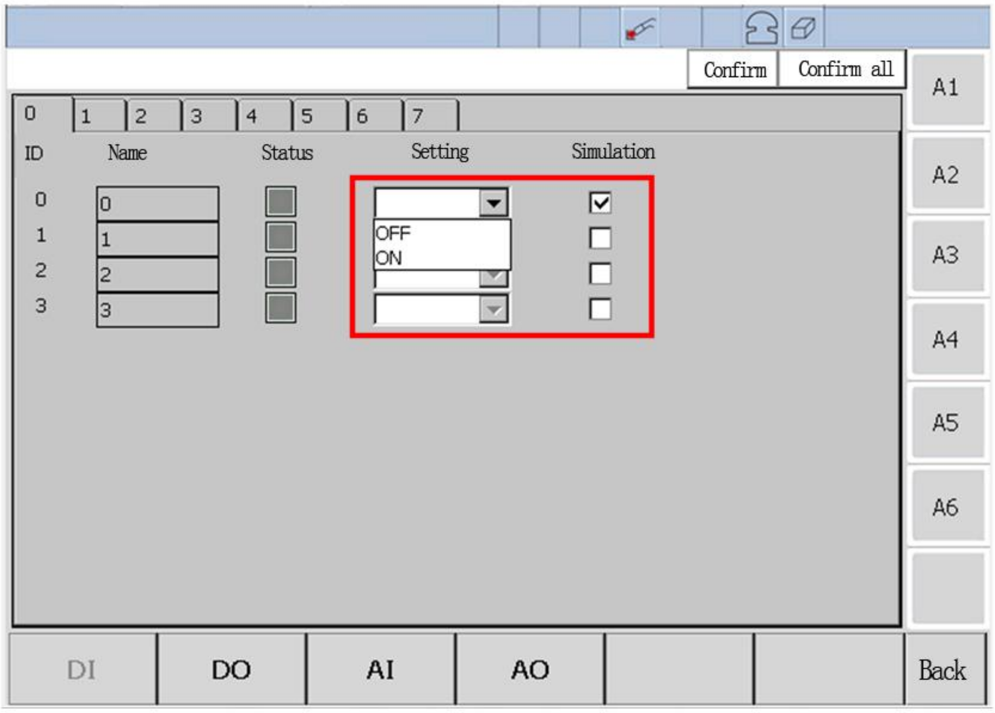
3.2.1 DI (digital input)

- When IO monitoring is selected, DI is selected by default, as shown in the figure



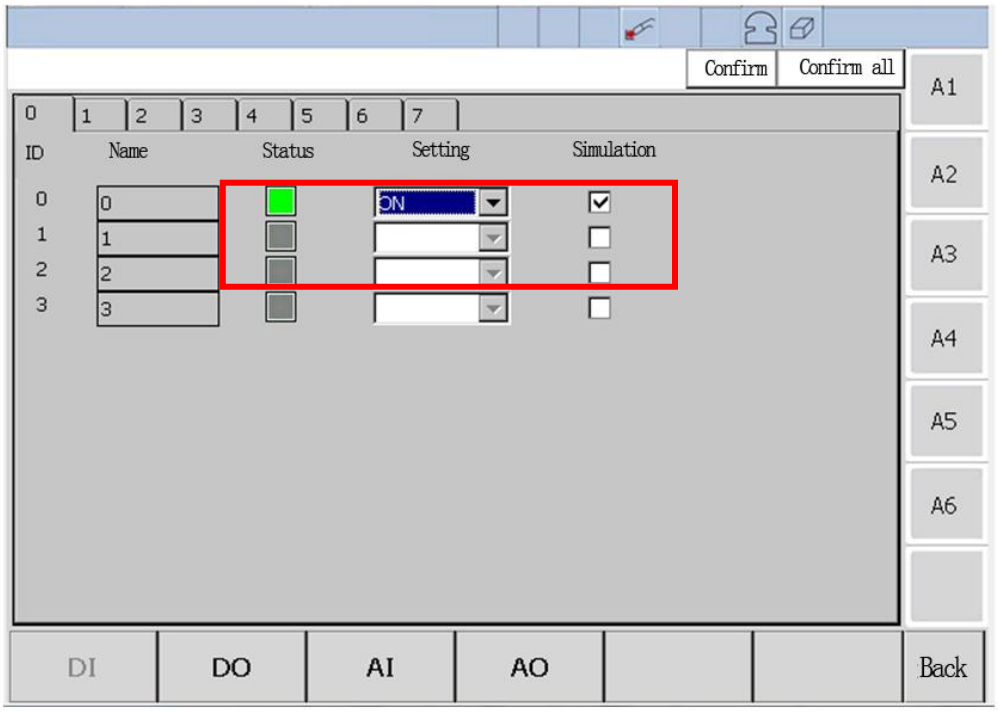
ID	Name	Status	Setting	Simulation
0	0	▨		□
1	1	▨		□
2	2	▨		□
3	3	▨		□

- Check Simulation in the DI window, and select the “OFF” or “ON” in the drop-down box of Setting, as shown in the figure



ID	Name	Status	Setting	Simulation
0	0	▨		☑
1	1	▨		☐
2	2	▨		☐
3	3	▨		☐

- Input simulation will be switched on if “ON” is selected, as shown in the figure



ID	Name	Status	Setting	Simulation
0	0	🟩	ON	☑
1	1	▨		☐
2	2	▨		☐
3	3	▨		☐

⚠**Attention**

① **The ID, Name, Status, Setting and Simulation in the DI interface separately**

means the ID No., name, status, input simulation setting and whether to switch on the input simulation of the DI.

② **The range of DI is 0-31 which can be extended according to the actual situation. The DI can be designated a name according to the actual requirement.**

③ **DI input simulation setting**

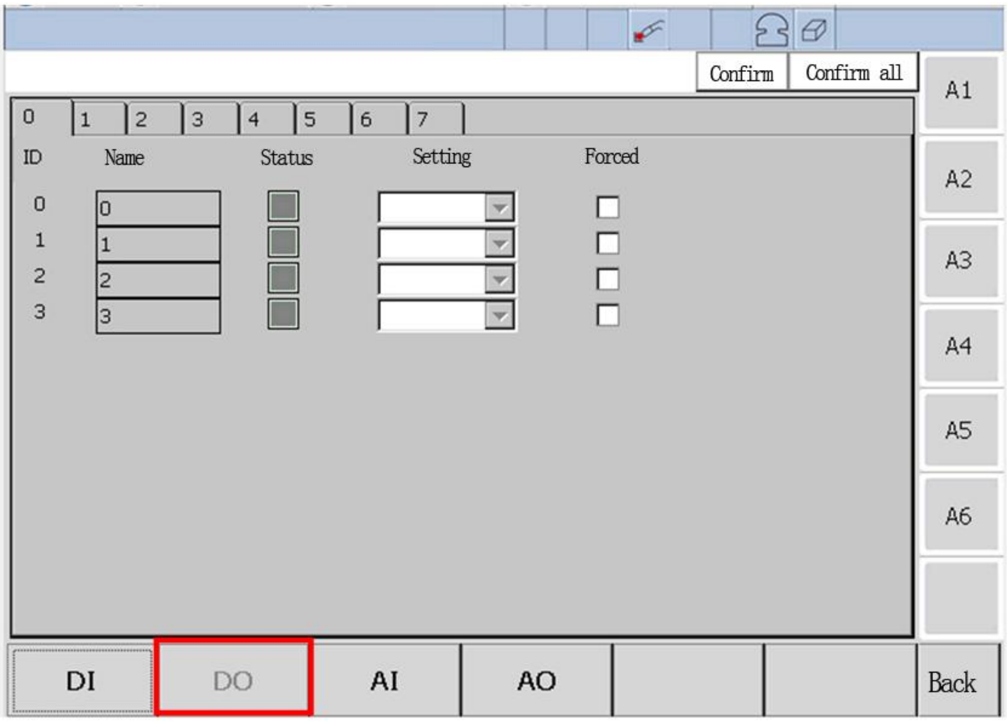
Status displays current DI status, 🟩 means that current DI is 1, and ■ means that current DI is 0.

Two options “ON” and “OFF” will be popped up when the drop-down box of Setting is clicked, separately corresponding to the “1” and “0” of DI status. E.g., when “ON” is clicked, it means that the input simulation status is set as “1”.

④ **Simulation column is used to control whether to conduct DI simulation: when the check box is checked, it means the input simulation will be conducted, and vice versa.**

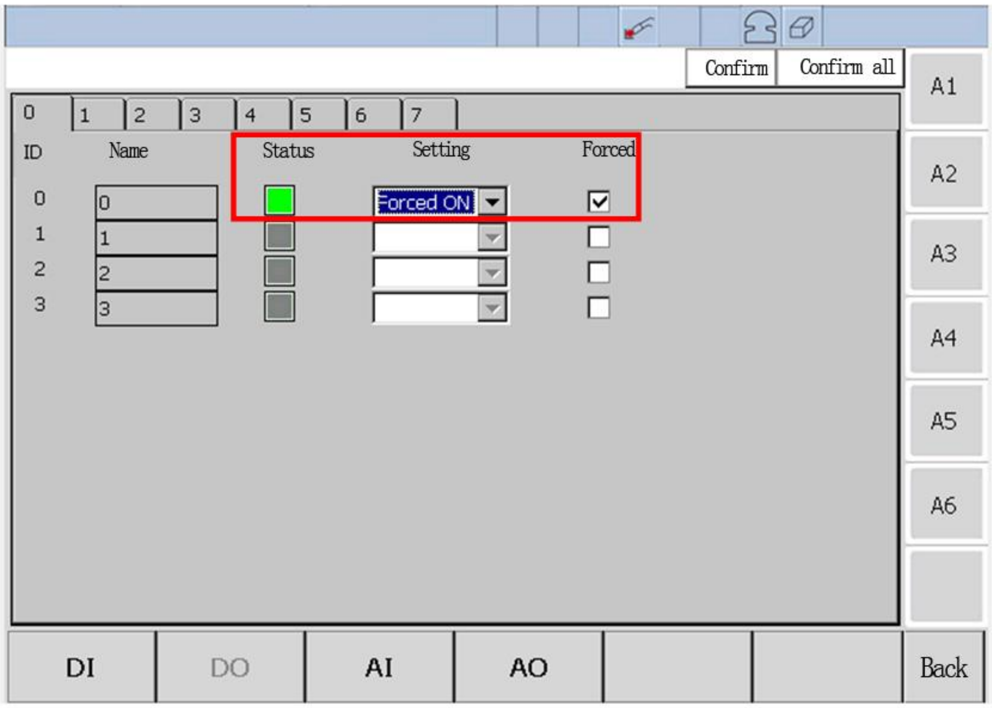
3.2.2 DO (digital output)

- DO window is as shown in the figure



ID	Name	Status	Setting	Forced
0	0	■		□
1	1	■		□
2	2	■		□
3	3	■		□

- Forced DO is as shown in the figure



ID	Name	Status	Setting	Forced
0	0	🟩	Forced ON	☑
1	1	◼		☐
2	2	◼		☐
3	3	◼		☐

Attention:

① **The ID, Name, Status, Setting and Forced in the DO interface separately means the ID No., name, status, forced output setting and whether to switch on the forced output of the DO.**

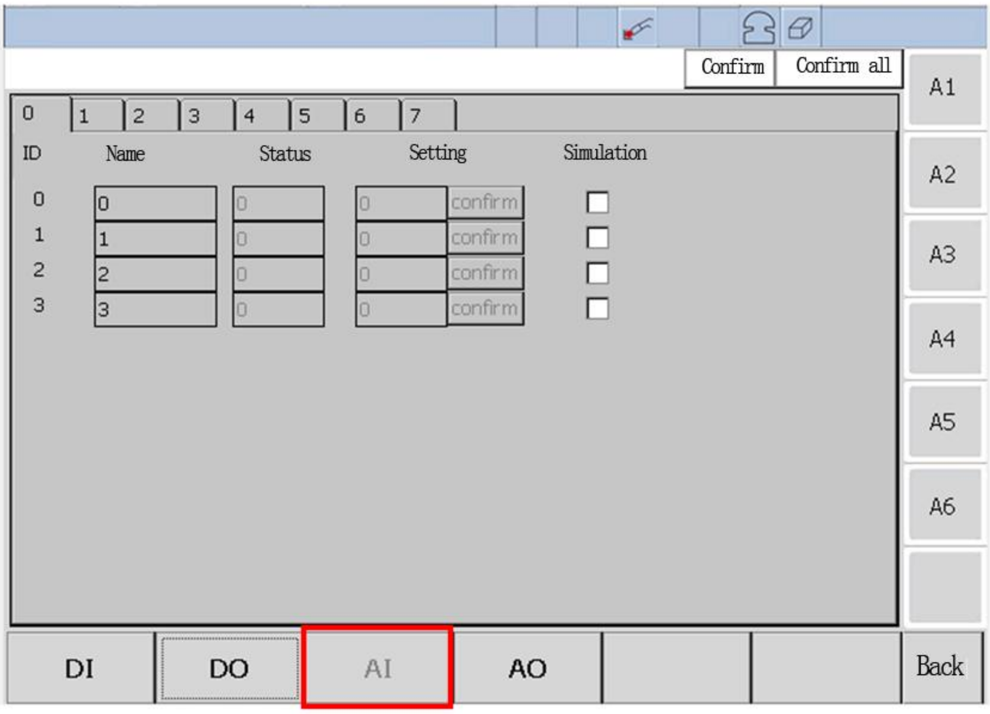
② **The range of DO is 0-31 which can be extended according to the actual situation. The DO can be designated a name according to the actual requirement.**

③ **Status displays current DO status: 🟩 means that current DO is 1, and ◼ means that current DO is 0. Two options "Forced ON" and "Forced OFF" will be popped up when the drop-down box of Setting is clicked, separately corresponding to the "1" and "0" of DO status. E.g., when "ON" is clicked, it means that the input simulation status is set as "1".**

④ **Forced column is used to control whether to conduct forced digital output: when the check box is checked, it means the forced output will be conducted, and vice versa.**

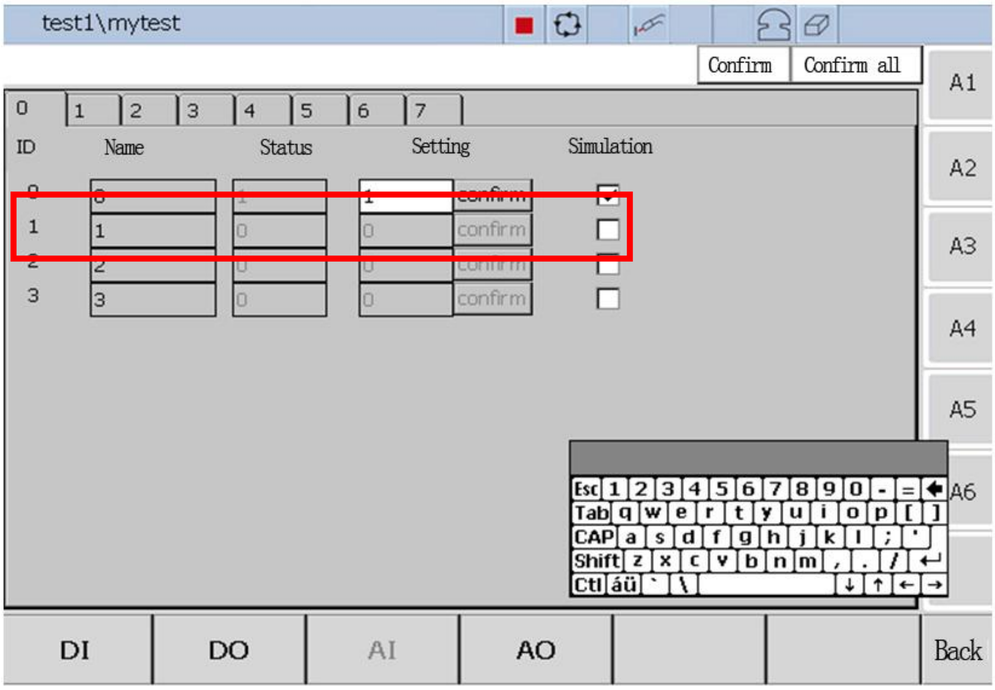
3.2.3 AI (analog input)

- AI is as shown in the figure



ID	Name	Status	Setting		Simulation
0	0	0	0	confirm	☐
1	1	0	0	confirm	☐
2	2	0	0	confirm	☐
3	3	0	0	confirm	☐

- Simulation AI is as shown in the figure


test1\mytest
Confirm
Confirm all
0
1
2
3
4
5
6
7
A1
A2
A3
A4
A5
A6
Esc 1 2 3 4 5 6 7 8 9 0 - =
Tab q w e r t y u i o p [ ]
CAP a s d f g h j k l ; '
Shift z x c v b n m , . / ↵
Ctl áü ` \ ↓ ↑ ← →
DI
DO
AI
AO
Back

ID	Name	Status	Setting		Simulation
0	0	1	1	confirm	☑
1	1	0	0	confirm	☐
2	2	0	0	confirm	☐
3	3	0	0	confirm	☐

⚠**Attention**

① **The ID, Name, Status, Setting and Simulation in the AI interface separately means the ID No., name, status, analog input simulation and whether to switch on the analog input simulation of the AI.**

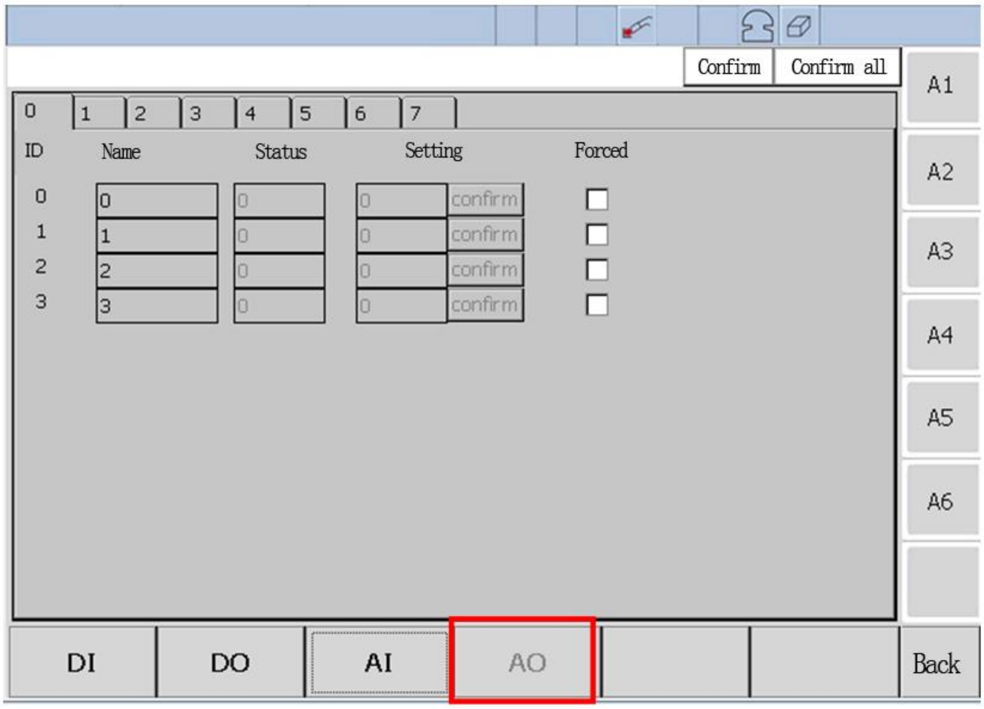
② **The range of AI is 0-31 which can be extended according to the actual situation. The AI can be designated a name according to the actual requirement.**

③ **Status displays current AI data. Enter the data in the text box, and click "Confirm" button after the text box to complete simulation of the AI.**

④ **Simulation column is used to control whether to conduct AI simulation: when the check box is checked, it means the AI simulation will be conducted, and vice versa.**

3.2.4 AO (analog output)

AO is as shown in the figure

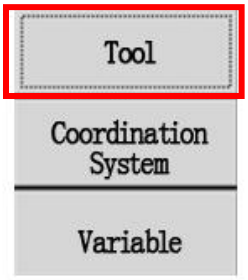


ID	Name	Status	Setting		Forced
0	0	0	0	confirm	□
1	1	0	0	confirm	□
2	2	0	0	confirm	□
3	3	0	0	confirm	□

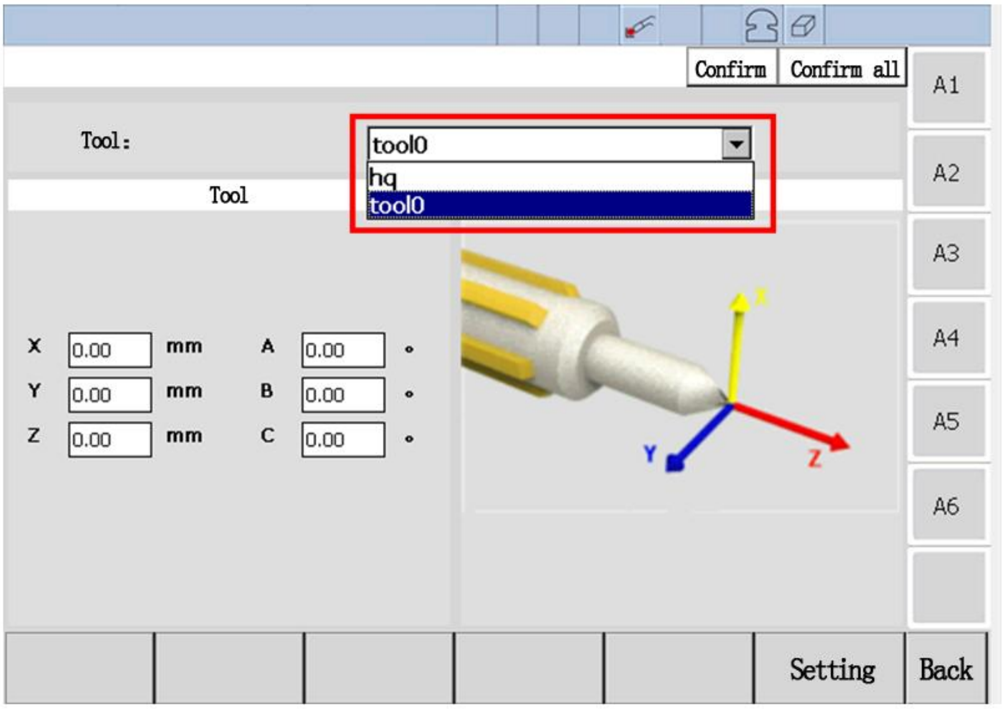
- Forced AO is as shown in the figure

3.3 Tool

- Click X= key to pop up options of “Tool”, “Coordinate system”, and “Variable”, as shown in the figure



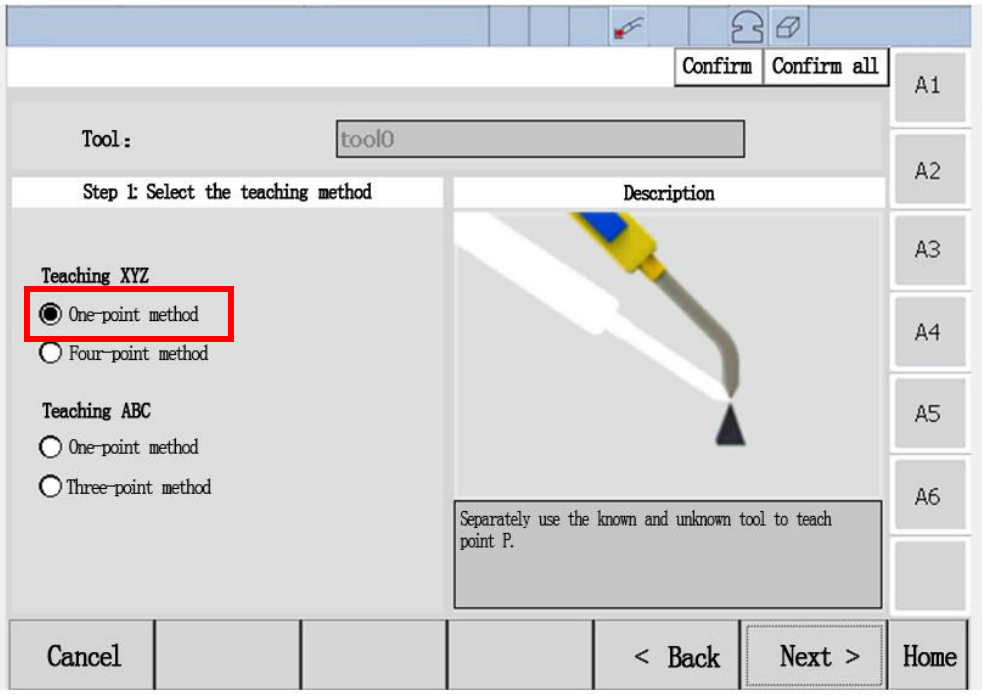
- Select the “Tool” for teaching, to enter the initialization interface, as shown in the figure



Tool: tool0

X 0.00 mm A 0.00 °
Y 0.00 mm B 0.00 °
Z 0.00 mm C 0.00 °

- First select the tool variable needed teaching in the drop-down box of “Tool”, and the numerical values of the tool will be displayed on left side of the interface; click “Setting” key to enter the interface for selecting the tool teaching method, as shown in the following figure:



Tool: tool0

Step 1: Select the teaching method

Teaching XYZ
● One-point method
○ Four-point method

Teaching ABC
○ One-point method
○ Three-point method

Description

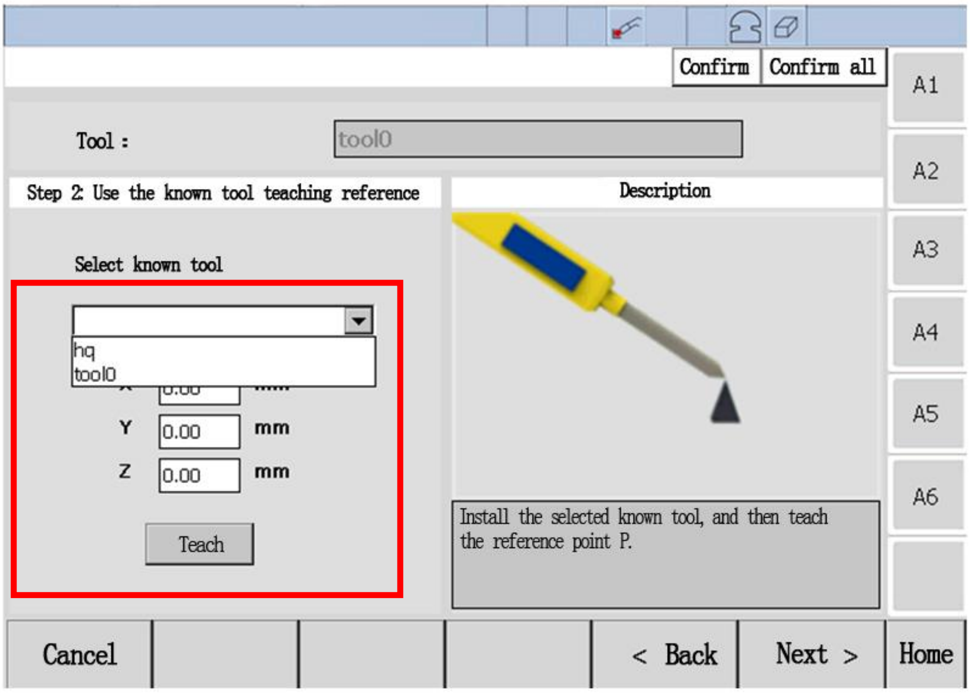
Separately use the known and unknown tool to teach point P.

Attention:

Tool variable teaching needs teaching in two parts: positions X, Y and Z of teaching tool; postures A, B and C of teaching tool. There are two methods of tool position teaching: one-point method and four-point method. There are two methods of tool posture teaching: one-point method and three-point method. Click “Cancel” or “Back” to return to the initial interface of coordinate system teaching; click “Next” to enter the specific coordinate system teaching interface.

3.3.1 Positions XYZ (one-point method) of teaching tool

- First select the coordinate teaching method (“Teaching XYZ One-point method”) in the above figure, and click “Next” to enter the following interface:



Tool : tool0

Step 2: Use the known tool teaching reference

Select known tool

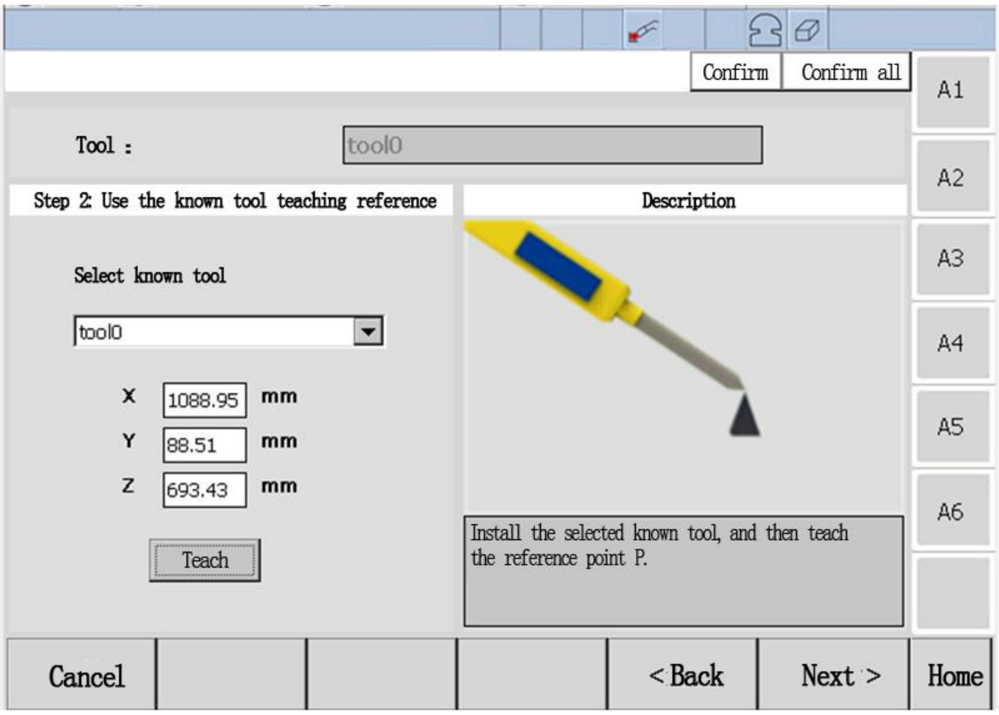
hq
tool0

Y 0.00 mm

Z 0.00 mm

Install the selected known tool, and then teach the reference point P.

- Select one tool that is known the posture, then move the robot to the expected position, and click “Teach” to complete the teaching; click “Cancel” and it will return to the initial interface of tool teaching; click “Back” and it will return to previous step; click “Next” and it will enter to the next teaching interface as shown in the figure below:



Tool : tool0

Step 2: Use the known tool teaching reference

Select known tool

tool0

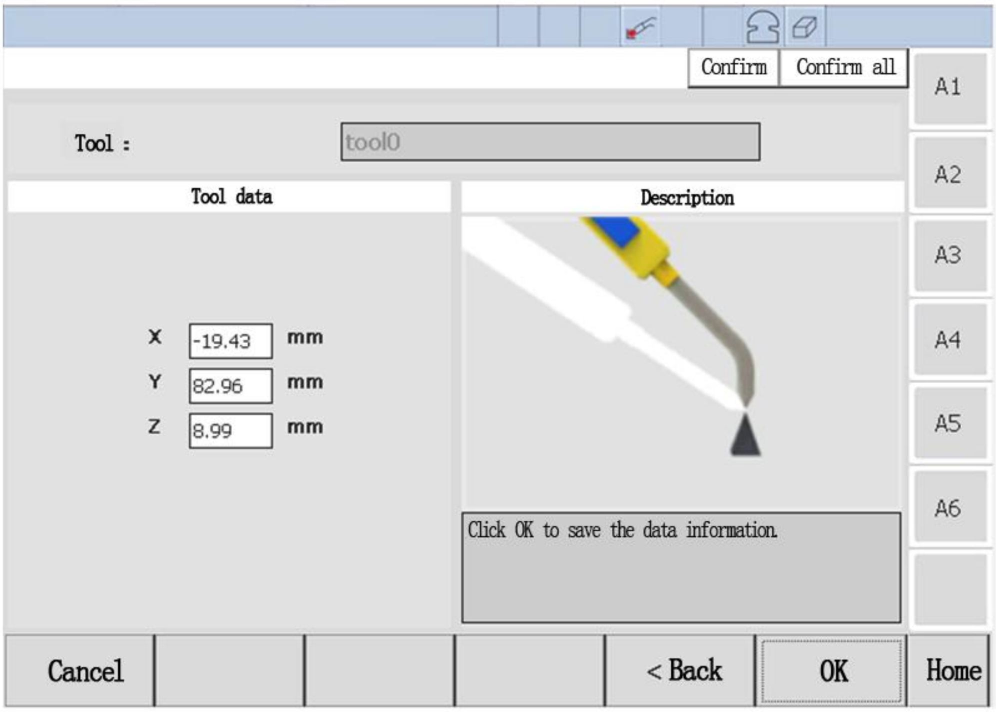
X 1088.95 mm

Y 88.51 mm

Z 693.43 mm

Install the selected known tool, and then teach the reference point P.

- Take down the known tool, replace with the tool needed teaching, then move the robot to the teaching position in Step 1, and click “Teach” to complete the position teaching; click “Cancel” and it will return to the initial interface of tool teaching; click “Back” and it will return to the interface for selecting tool teaching method; click “Next” and it will enter to the next teaching interface as shown in the figure below:


Tool : tool0

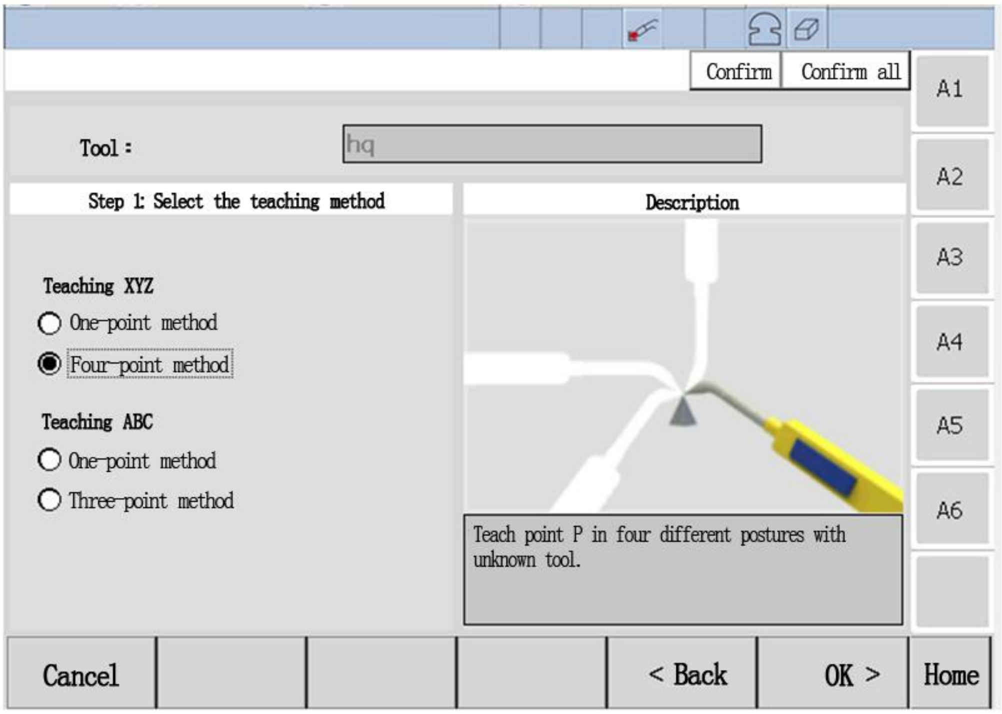
X -19.43 mm
Y 82.96 mm
Z 8.99 mm

Click OK to save the data information.

- The left in the interface displays the coordinate system data after teaching; click “Cancel” and it will return to the initial interface of tool teaching; click “Back” and it will return to the Step 2 interface of obtaining XYZ (one-point method) teaching; click “OK” and it will save the data and complete the tool teaching.

3.3.2 Positions XYZ (four-point method) of teaching tool

- Select the coordinate system teaching method (“Teaching XYZ Four-point method”) as shown in the figure



Tool： hq

Step 1: Select the teaching method

Teaching XYZ

○ One-point method

● Four-point method

Teaching ABC

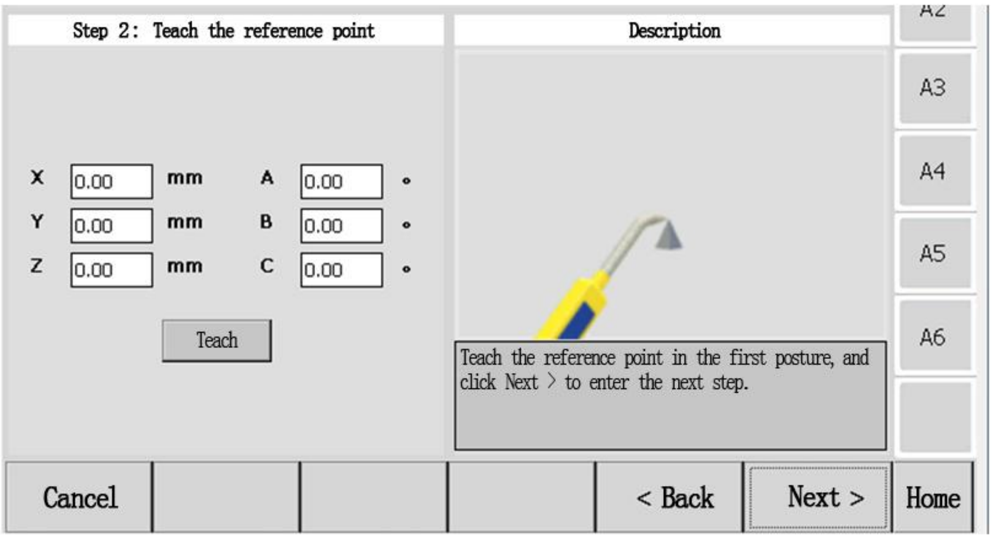
○ One-point method

○ Three-point method

Description

Teach point P in four different postures with unknown tool.

- Click “Next” to enter the following interface, then move the robot to the expected position, and click “Teach” to complete the teaching of first posture; click “Cancel” and it will return to the initial interface of tool teaching; click “Back” and it will return to the previous step as shown in the figure below:



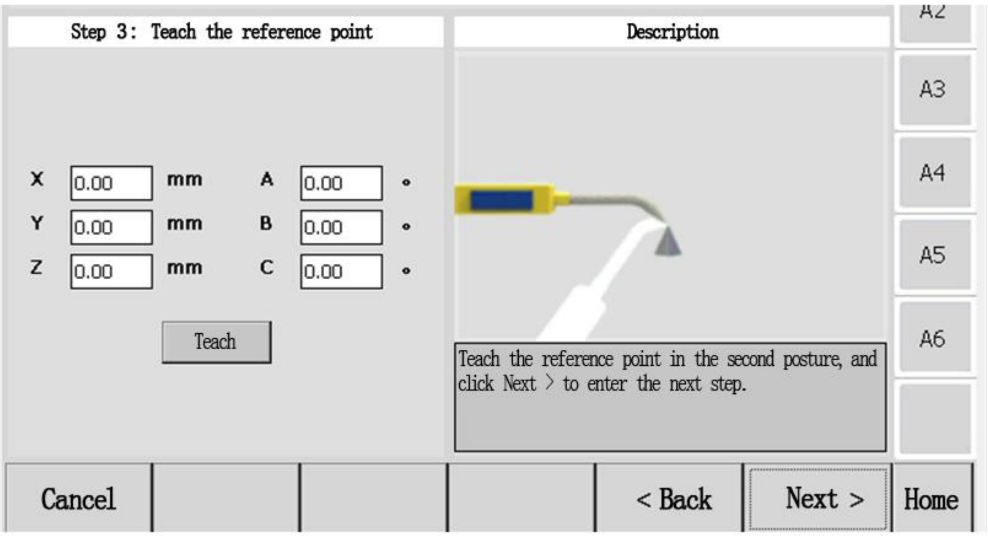
Step 2: Teach the reference point

X	0.00	mm	A	0.00	°
Y	0.00	mm	B	0.00	°
Z	0.00	mm	C	0.00	°

Description

Teach the reference point in the first posture, and click Next > to enter the next step.

- Click “Next” to enter the next teaching interface as shown in the figure below:



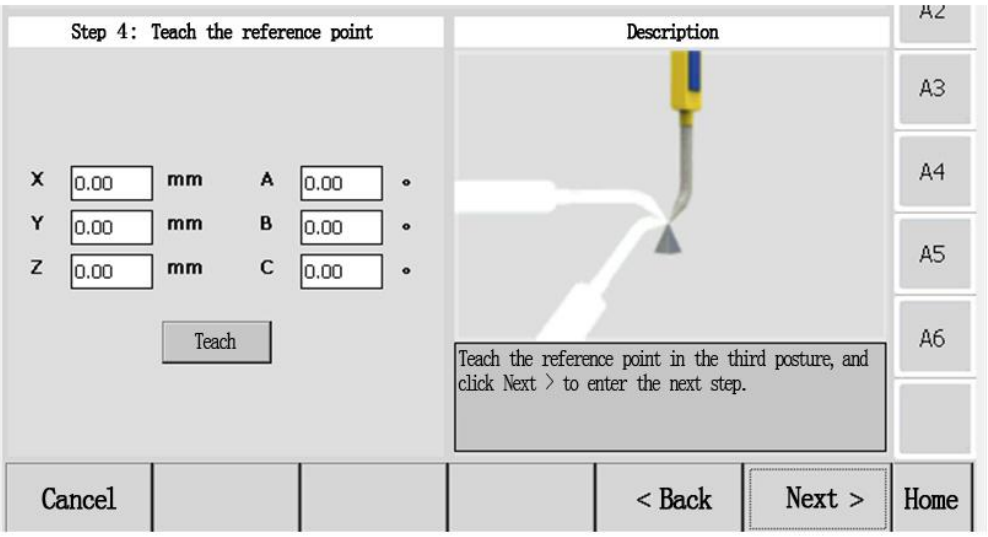
Step 3: Teach the reference point

X 0.00 mm A 0.00 °
Y 0.00 mm B 0.00 °
Z 0.00 mm C 0.00 °

Description

Teach the reference point in the second posture, and click Next > to enter the next step.

- Approach the robot to the expected point in another posture, and click “Teach” to complete the teaching of second posture point. As shown in the figure, click “Cancel” and it will return to the initial interface of tool teaching; click “Back” and it will return to the previous step.
- Click “Next” to enter the next teaching interface as shown in the figure below:



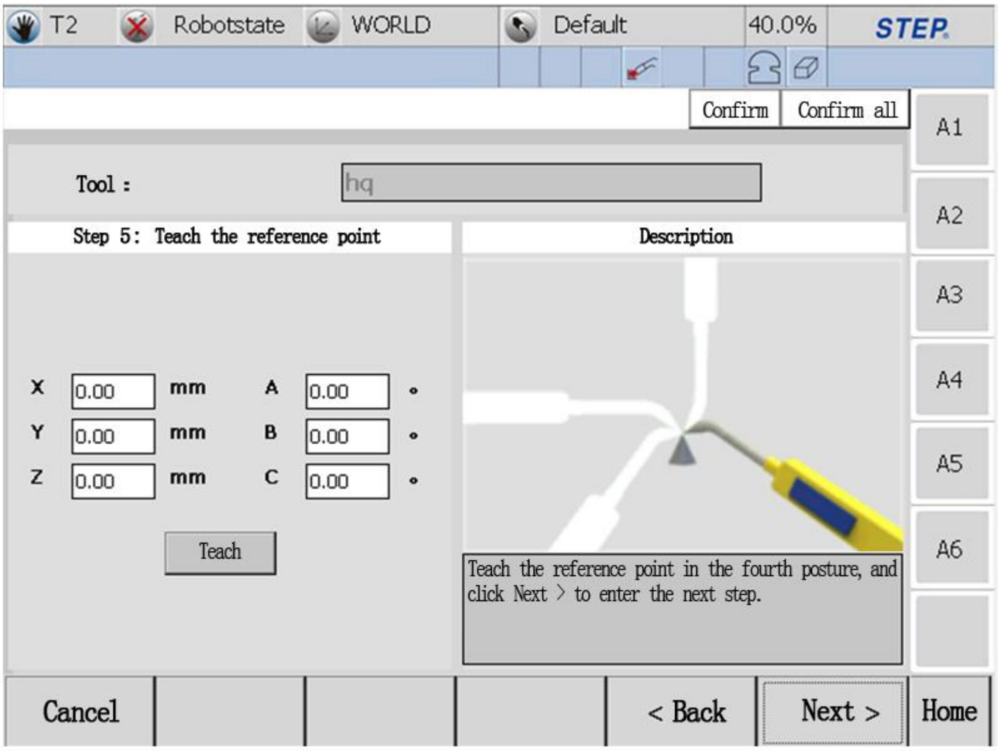
Step 4: Teach the reference point

X 0.00 mm A 0.00 °
Y 0.00 mm B 0.00 °
Z 0.00 mm C 0.00 °

Description

Teach the reference point in the third posture, and click Next > to enter the next step.

- Approach the robot to the expected point in another posture, and click “Teach” to complete the teaching of third posture point. As shown in the figure, click “Cancel” and it will return to the initial interface of tool teaching; click “Back” and it will return to the previous step.
- Click “Next” to enter the next teaching interface.



Tool : hq

Step 5: Teach the reference point

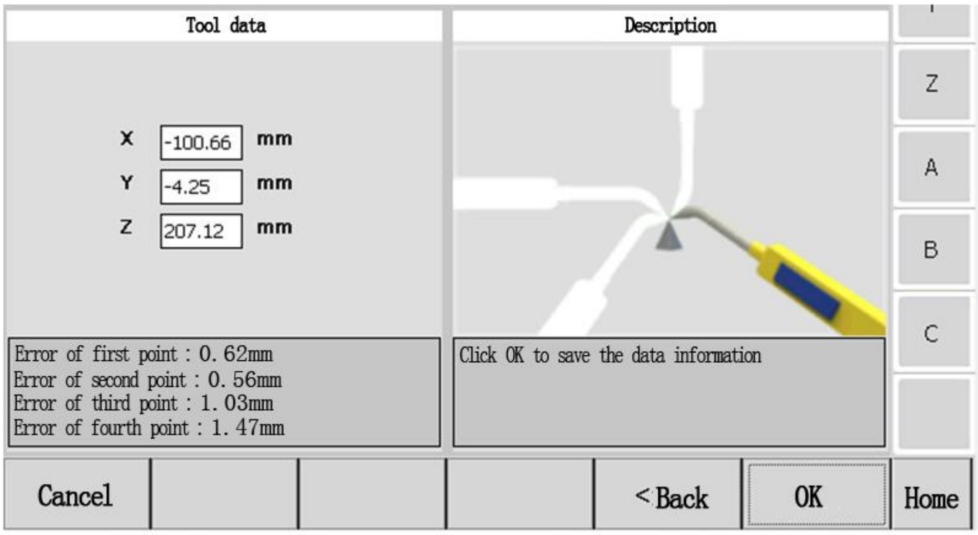
X	0.00	mm	A	0.00	°
Y	0.00	mm	B	0.00	°
Z	0.00	mm	C	0.00	°

Teach

Description

Teach the reference point in the fourth posture, and click Next > to enter the next step.

- Approach the robot to the expected point in another posture, and click “Teach” to complete the teaching of fourth posture point. As shown in the figure, click “Cancel” and it will return to the initial interface of tool teaching; click “Back” and it will return to the previous step.
- Click Next to enter the next interface as shown in the figure



Tool data

X	-100.66	mm
Y	-4.25	mm
Z	207.12	mm

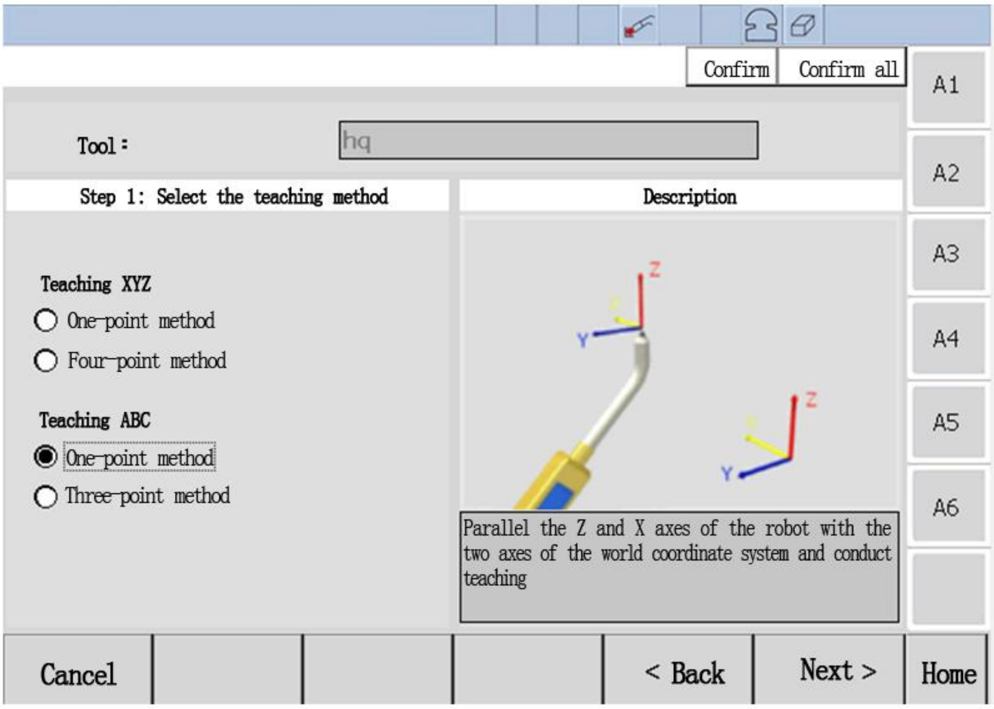
Error of first point : 0. 62mm
Error of second point : 0. 56mm
Error of third point : 1. 03mm
Error of fourth point : 1. 47mm

Description

Click OK to save the data information

- The left in the figure displays the coordinate system data after teaching, and the lower part displays the absolute error between the four teaching points and the actual points; click “Cancel” and it will return to the initial interface of tool teaching; click “Back” and it will return to the previous step; click “OK” and it will save the data and complete the tool teaching.

3.3.3 Postures ABC (one-point method) of teaching tool



Tool: hq

Step 1: Select the teaching method

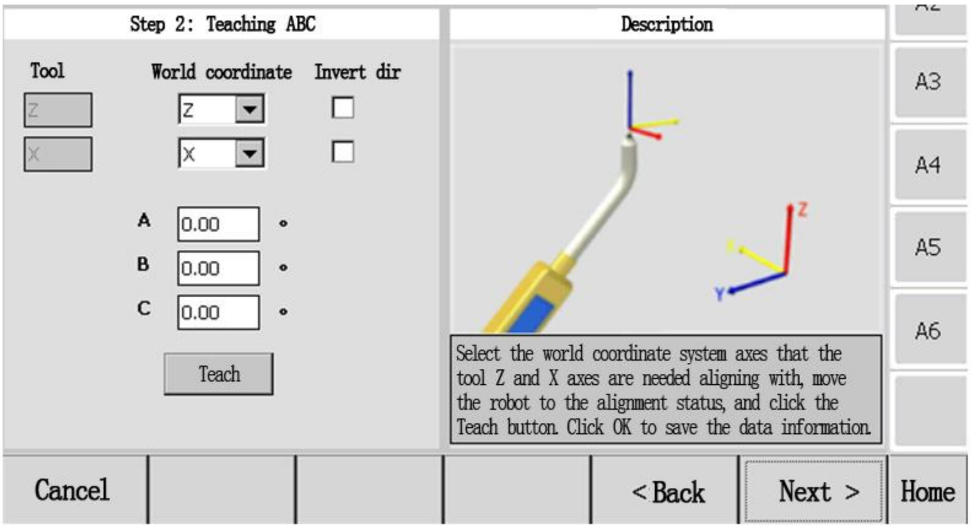
Teaching XYZ
○ One-point method
○ Four-point method

Teaching ABC
● One-point method
○ Three-point method

Description

Parallel the Z and X axes of the robot with the two axes of the world coordinate system and conduct teaching

- Select the coordinate system teaching method (“Teaching ABC One-point method”) and click “Next” to enter the interface as below:



Step 2: Teaching ABC

Tool	World coordinate	Invert dir
Z	Z	□
X	X	□

A 0.00 °
B 0.00 °
C 0.00 °

Description

Select the world coordinate system axes that the tool Z and X axes are needed aligning with, move the robot to the alignment status, and click the Teach button. Click OK to save the data information.

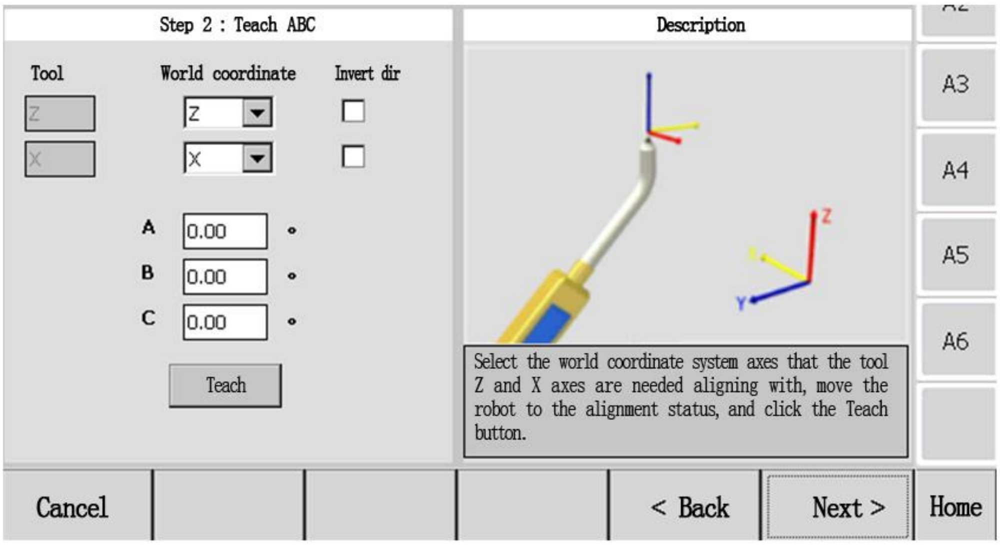
Attention:

① "Tool" column indicates the positive directions of "Z" and "X" axes of the coordinate system of the tool.

② "World coordinate" column is used to confirm the alignment between the tool axes and World axes.

③ "Invert direction" column is used to confirm whether the axes are aligned with the positive axes or negative axes of the selected world coordinate system.

- The teaching result is as shown in the figure below



Step 2 : Teach ABC

Tool	World coordinate	Invert dir
Z	Z	□
X	X	□

A 0.00 °

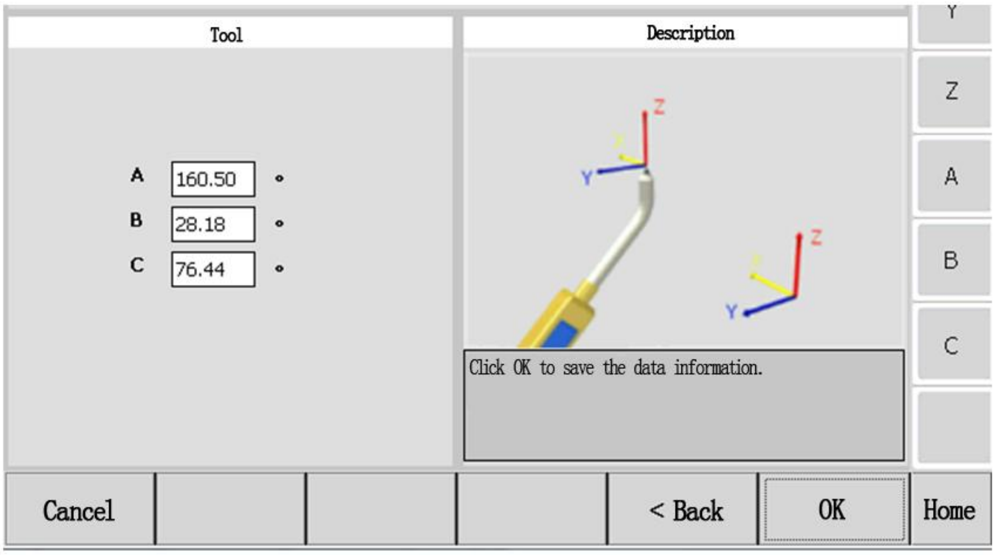
B 0.00 °

C 0.00 °

Description

Select the world coordinate system axes that the tool Z and X axes are needed aligning with, move the robot to the alignment status, and click the Teach button.

- Based on the alignment relationship setting between the tool axes and world coordinate axes, move the robot to the alignment posture, and click "Teach" to complete the teaching of the point; click "Cancel" and it will return to the initial interface of tool teaching; click "Back" and it will return to the interface for selecting the tool teaching method; click "Next" and it will enter the next teaching interface as shown in the figure below:



Tool

A 160.50 °

B 28.18 °

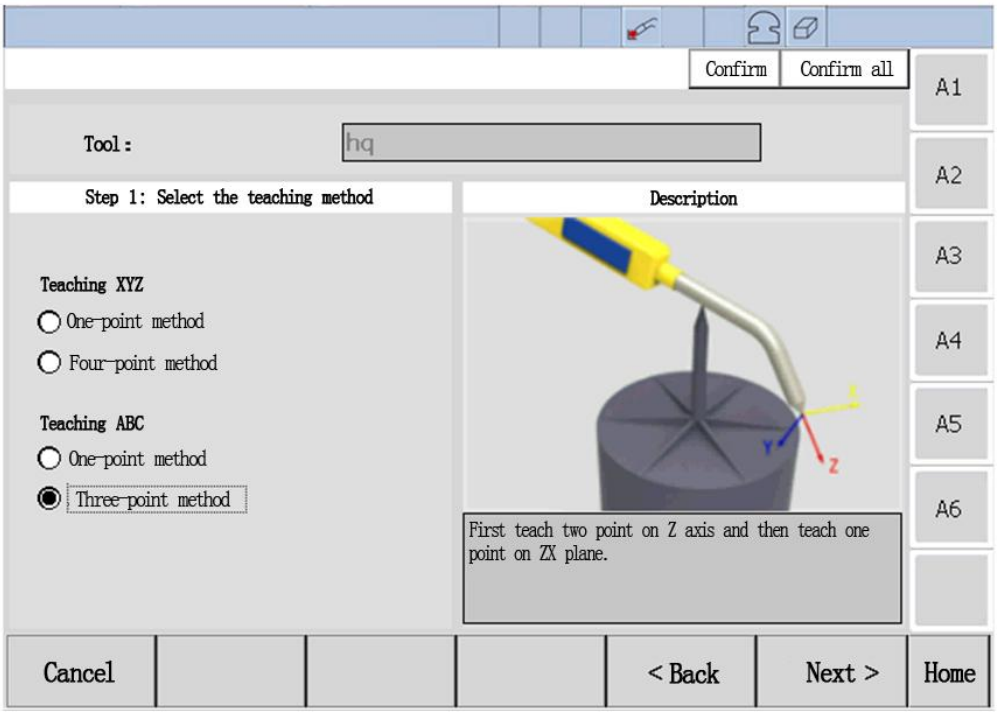
C 76.44 °

Description

Click OK to save the data information.

- The left in the interface displays the coordinate system data after teaching; click "Cancel" and it will return to the initial interface of tool teaching; click "Back" and it will return to the Step 1 interface of obtaining ABC (One-point method) teaching; click "OK" to complete the tool teaching.

3.3.4 Postures ABC (three-point method) of teaching tool



Tool： hq

Step 1: Select the teaching method

Teaching XYZ

○ One-point method

○ Four-point method

Teaching ABC

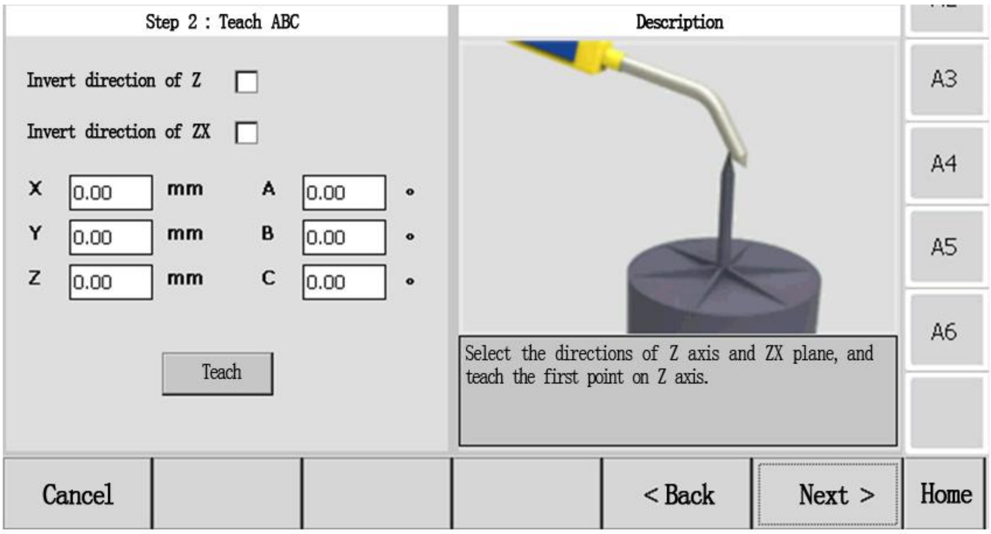
○ One-point method

● Three-point method

Description

First teach two point on Z axis and then teach one point on ZX plane.

- Select the coordinate system teaching method ("Teaching ABC Three-point method") and click "Next" to enter the following interface:



Step 2 : Teach ABC

Invert direction of Z □

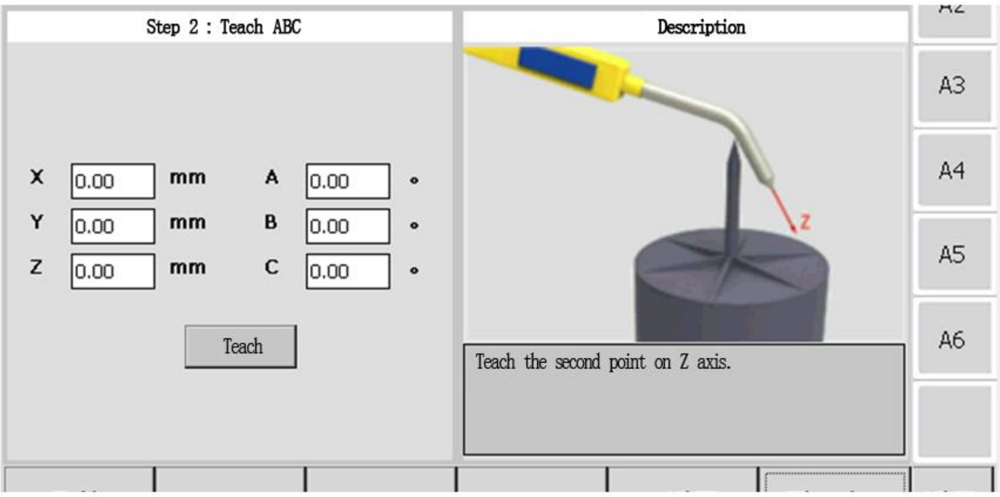
Invert direction of ZX □

X	0.00	mm	A	0.00	°
Y	0.00	mm	B	0.00	°
Z	0.00	mm	C	0.00	°

Description

Select the directions of Z axis and ZX plane, and teach the first point on Z axis.

- First select the teaching direction: forward or invert direction of Z axis, forward or invert direction of ZX plane. Move the robot to certain reference point in the space, and then click “Teach” button to complete teaching of the first point, as shown in the figure above.
- Click “Cancel” to return to the initial interface of tool teaching; click “Back” to return to the previous step; click “Next” to enter the next teaching interface as shown in the figure below:



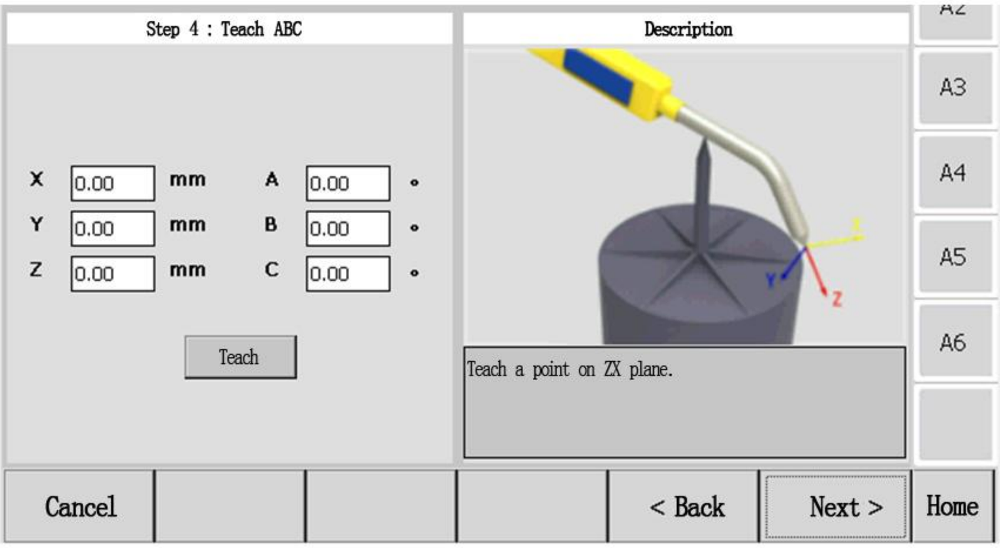
Step 2 : Teach ABC

X	0.00	mm	A	0.00	°
Y	0.00	mm	B	0.00	°
Z	0.00	mm	C	0.00	°

Description

Teach the second point on Z axis.

- Move the robot to certain reference point in the space, and then click “Teach” button to complete teaching of the second point, as shown in the figure above.
- Click “Cancel” to return to the initial interface of tool teaching; click “Back” to return to the interface for selecting the tool teaching method; click “Next” to enter the next teaching interface as shown in the figure below:



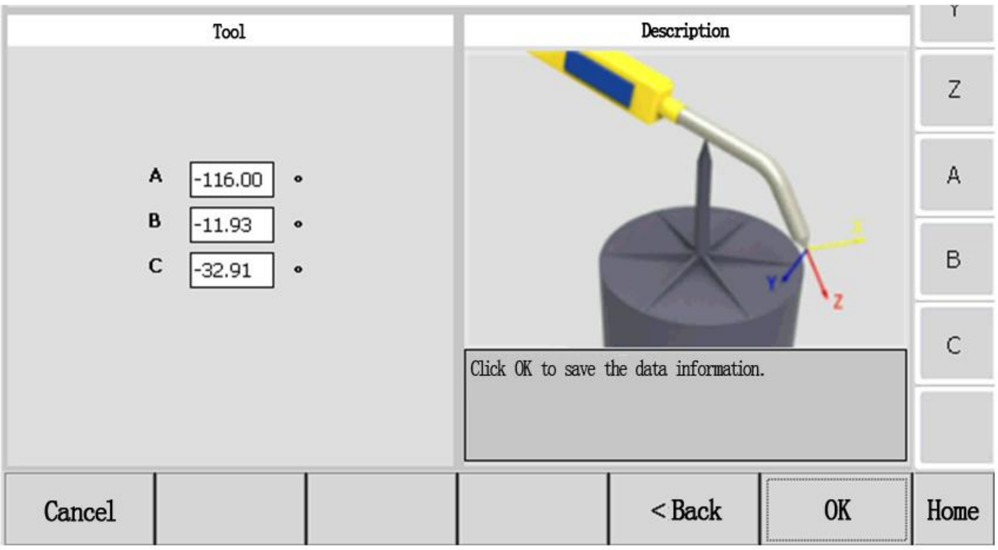
Step 4 : Teach ABC

X 0.00 mm A 0.00 °
Y 0.00 mm B 0.00 °
Z 0.00 mm C 0.00 °

Description

Teach a point on ZX plane.

- Move the robot to certain reference point in the space, and then click “Teach” button to complete teaching of the third point, as shown in the figure above.
- Click “Cancel” to return to the initial interface of tool teaching; click “Back” to return to the interface for selecting the tool teaching method; click “Next” to enter the next teaching interface as shown in the figure below:



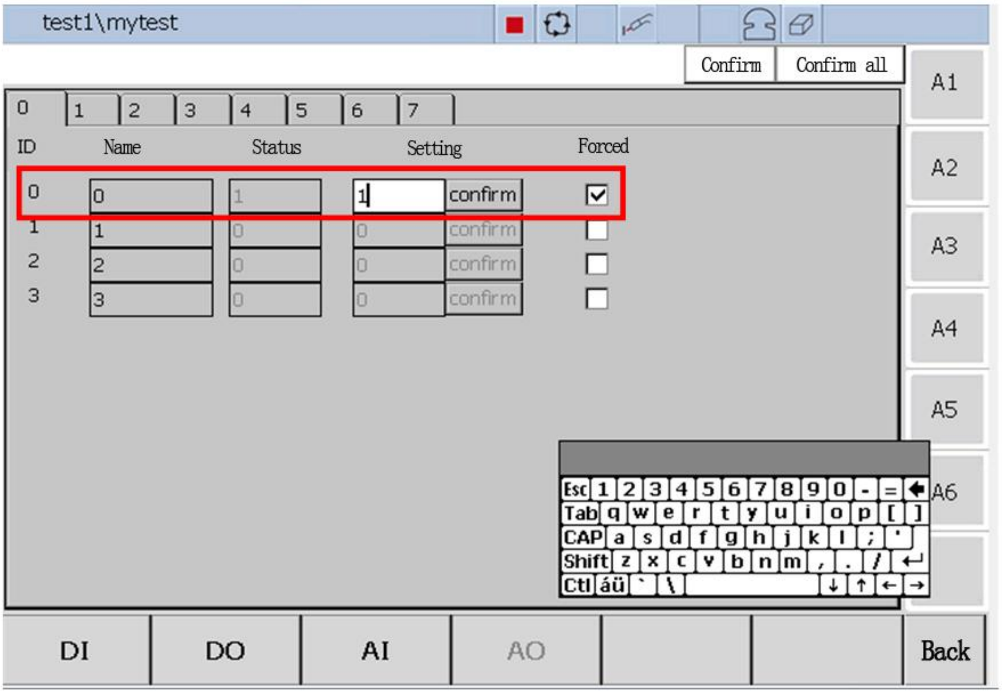
Tool

A -116.00 °
B -11.93 °
C -32.91 °

Description

Click OK to save the data information.

- The left in the interface displays the coordinate system data after teaching; click “Cancel” and it will return to the initial interface of tool teaching; click “Back” and it will return to previous step; click “OK” to save the data information and complete the tool teaching.


test1\mytest
Confirm
Confirm all
A1
A2
A3
A4
A5
A6
0
1
2
3
4
5
6
7
Esc 1 2 3 4 5 6 7 8 9 0 - =
Tab q w e r t y u i o p [ ]
CAP a s d f g h j k l ; '
Shift z x c v b n m , . /
Ctl áü ` \ ↓ ↑ ← →
DI
DO
AI
AO
Back

ID	Name	Status	Setting		Forced
0	0	1	1	confirm	☑
1	1	0	0	confirm	□
2	2	0	0	confirm	□
3	3	0	0	confirm	□

⚠**Attention**

① **The ID, Name, Status, Setting and Forced in the analog output interface separately means the ID No., name, status, forced analog output and whether to switch on the forced analog output of the AO.**

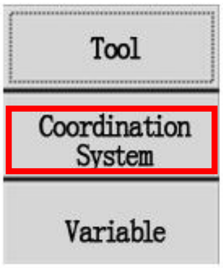
② **The range of AO is 0-31 which can be extended according to the actual situation. The AO can be designated a name according to the actual requirement.**

③ **Status displays current AO data. Enter the data in the text box, and click "confirm" button after the text box to complete forced AO.**

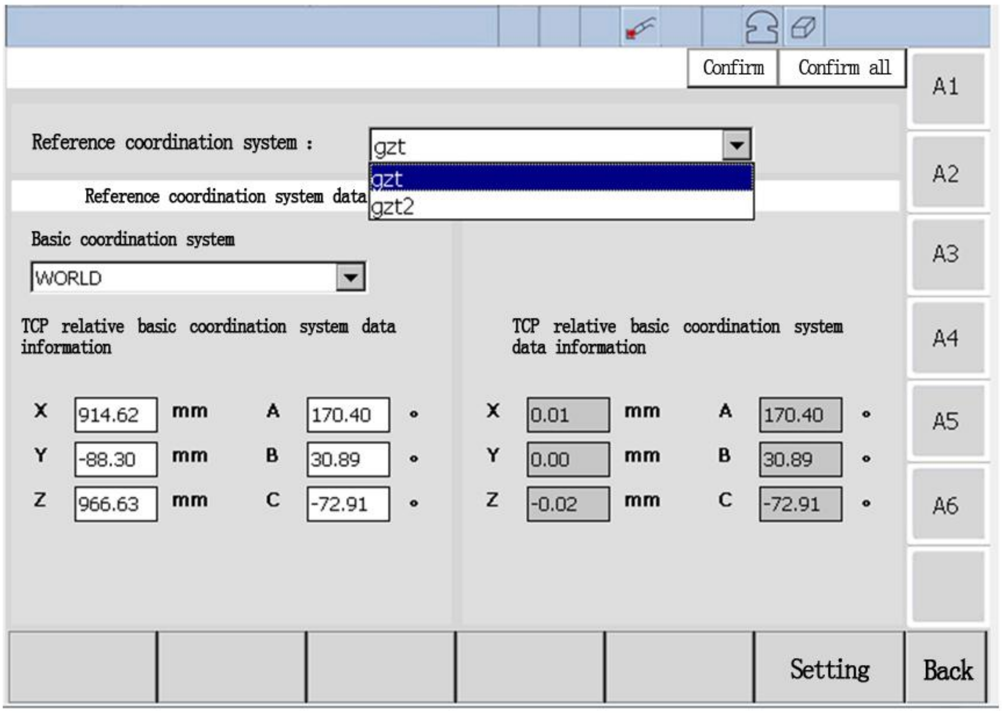
④ **Forced column is used to control whether to conduct forced AO: when the check box is checked, it means the forced AO will be conducted, and vice versa.**

3.4 Coordination system

Press X= to pop up the options of "Tool", "Coordination system" and "Variable" as shown in the figure



Select coordination system to enter into the initial interface of the coordination system as shown in the following figure



Reference coordination system : gzt

gzt
gzt2

Reference coordination system data

Basic coordination system

WORLD

TCP relative basic coordination system data information

X	914.62	mm	A	170.40	°
Y	-88.30	mm	B	30.89	°
Z	966.63	mm	C	-72.91	°

TCP relative basic coordination system data information

X	0.01	mm	A	170.40	°
Y	0.00	mm	B	30.89	°
Z	-0.02	mm	C	-72.91	°

⚠ **Note:**

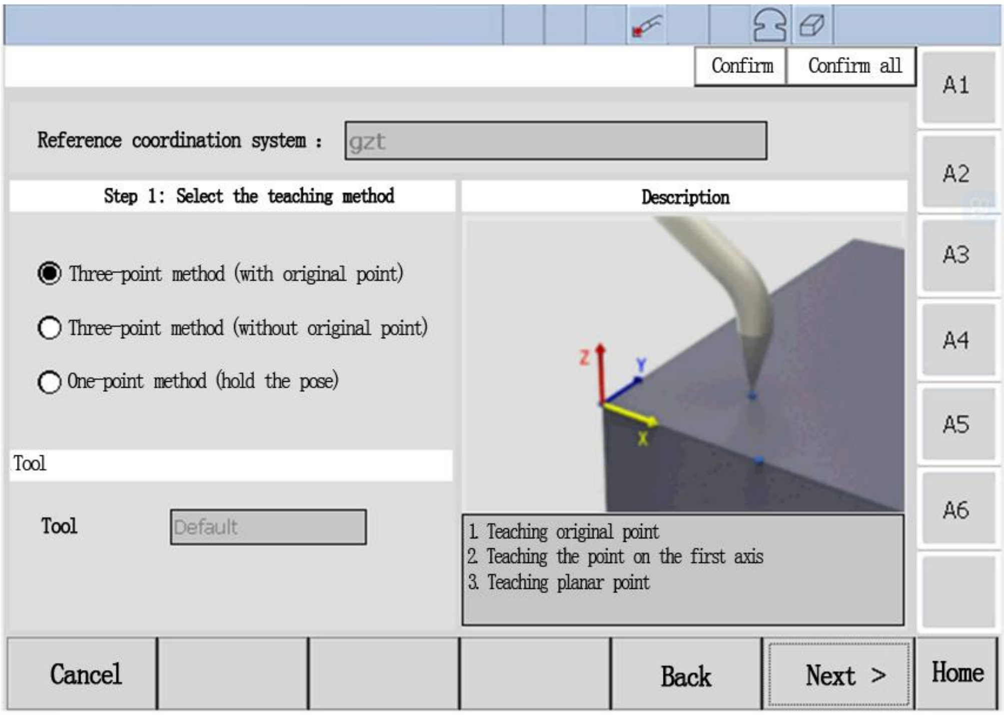
① **Selection of user’s coordination system: select the coordination system for teaching;**

① **Selection of reference coordination system: select reference coordination system for teaching coordination system. The type of the coordination system here include general user coordination system, “WORLD” coordination system and “ROBOTBASE” coordination system;**

③ **The display of TCP point on the user coordination system;**

④ **The display of TCP point on the reference coordination system.**

- Click on “Settings” to enter into the user coordination system for teaching to select the interface as shown in the following figure:



Reference coordination system : gzt

Step 1: Select the teaching method

● Three-point method (with original point)

○ Three-point method (without original point)

○ One-point method (hold the pose)

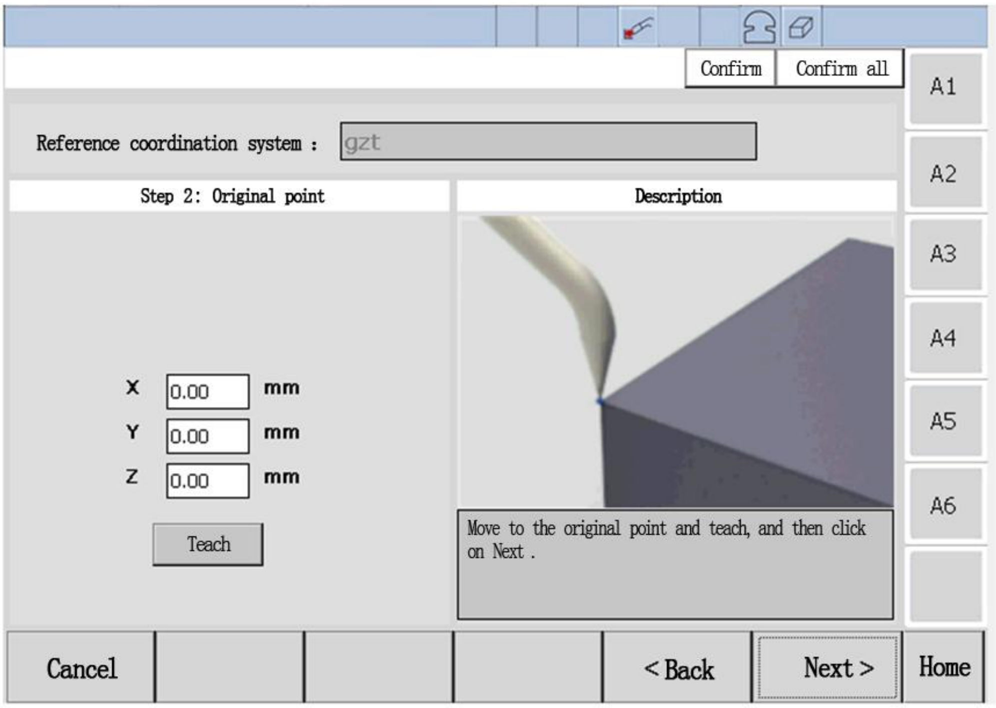
Tool

Tool: Default

1. Teaching original point
2. Teaching the point on the first axis
3. Teaching planar point

3.4.1 Three-point method (with original point)

- Select the coordination system teaching method “Three-point method(with original point) and click on “Next” to enter into the interface as shown as below:



Reference coordination system : gzt

Step 2: Original point

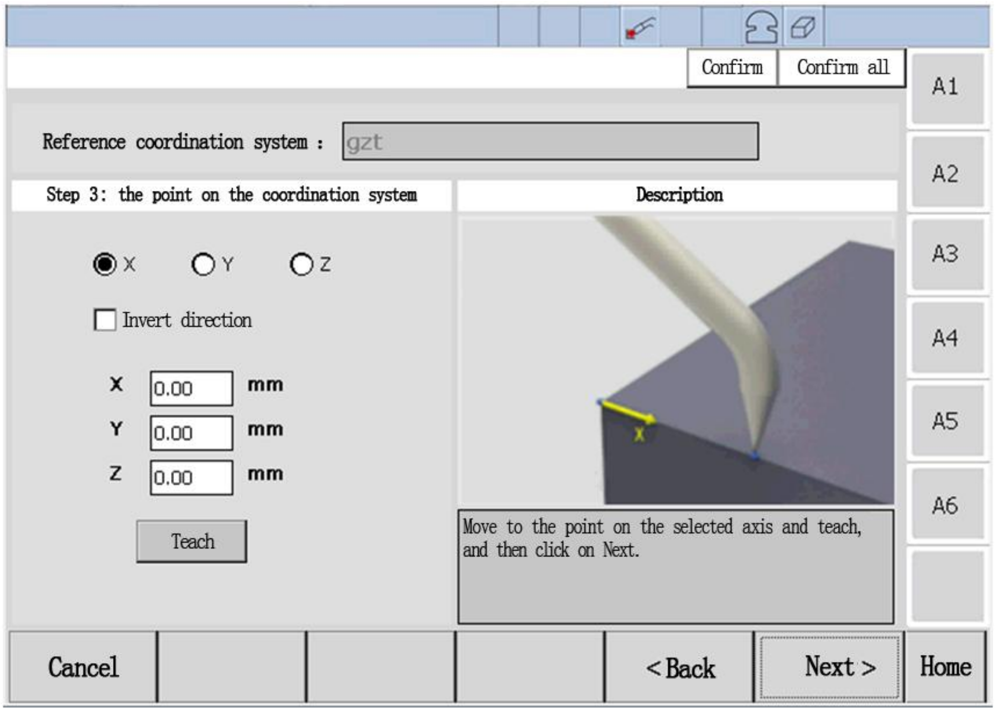
X 0.00 mm

Y 0.00 mm

Z 0.00 mm

Move to the original point and teach, and then click on Next .

- Click "Teaching" button to complete the original point teaching of coordination system; click on "Cancel" to return to the initial interface of coordination system teaching; click on "Return" to return to the previous interface (coordination system teaching method selecting interface).
- Click on "Next" to enter into the next teaching interface as shown in the following figure:



Reference coordination system : gzt

Step 3: the point on the coordination system

● X ○ Y ○ Z

□ Invert direction

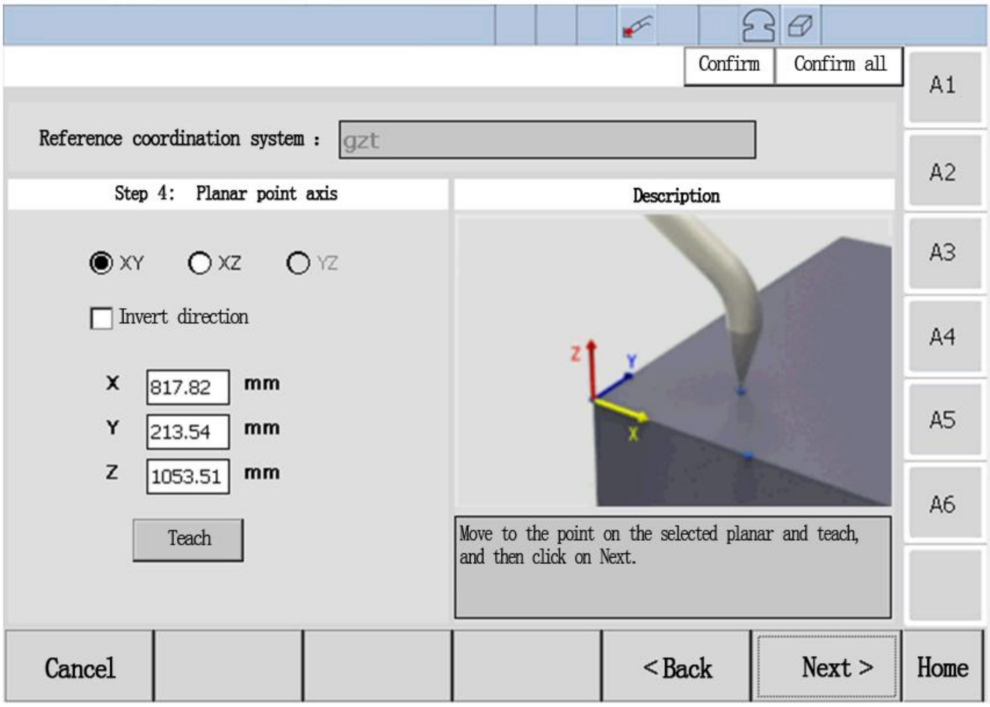
X 0.00 mm

Y 0.00 mm

Z 0.00 mm

Move to the point on the selected axis and teach, and then click on Next.

- Firstly, select the axis (X, Y, Z) where the point for teaching locates, select the direction of the teaching axis (positive or negative), move the robot to the desired position and finally click on "Teaching" to complete the teaching of the second point of the Three-point method. Click on "Cancel" to return to the initial interface of coordination system teaching; click on "Return" to return to the previous step.
- Click on "Next" to enter into the next teaching interface as shown in the following figure:



Reference coordination system : gzt

Step 4: Planar point axis

● XY ○ XZ ○ YZ

□ Invert direction

X 817.82 mm

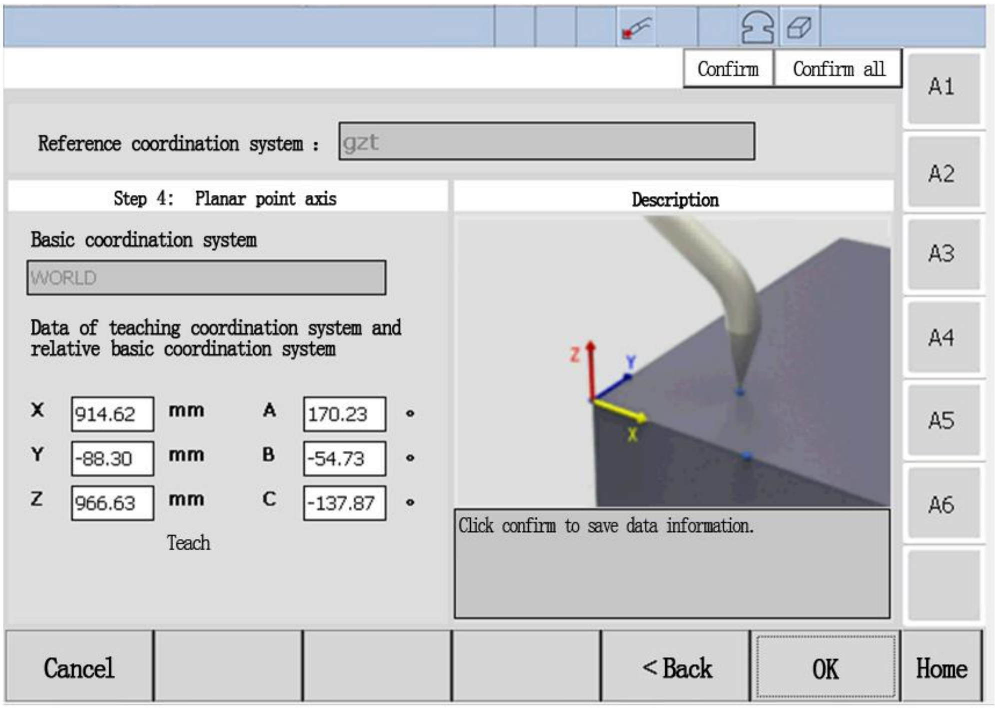
Y 213.54 mm

Z 1053.51 mm

Description

Move to the point on the selected planar and teach, and then click on Next.

- Firstly, select the axis (XY, XZ, YZ) where the point for teaching locates, select the direction of the teaching axis (positive or negative), move the robot to the desired position and finally click on “Teaching” to complete the teaching of the third point of the Three-point method. Click on “Cancel” to return to the initial interface of coordination system teaching; click on “Return” to return to the previous step.
- Click on “Next” to enter into the next teaching interface as shown in the following figure:



Reference coordination system : gzt

Step 4: Planar point axis

Basic coordination system: WORLD

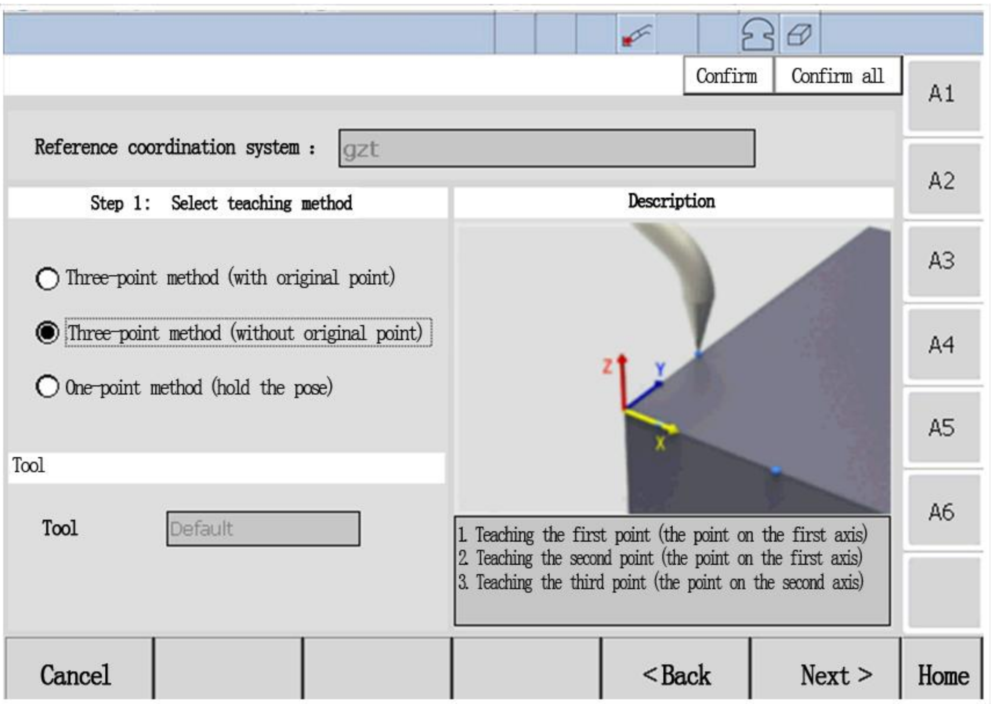
Data of teaching coordination system and relative basic coordination system

X	914.62	mm	A	170.23	°
Y	-88.30	mm	B	-54.73	°
Z	966.63	mm	C	-137.87	°

Click confirm to save data information.

- What displays on the left side of the interface is the coordination system data XYZ after teaching; click on “Cancel” to return to the initial interface of coordination system teaching; click on “Return” to return to the previous step; click on “CONFIRM” to save data and complete the coordination system teaching.

3.4.2 Three-point method (without original point)



Reference coordination system : gzt

Step 1: Select teaching method

○ Three-point method (with original point)

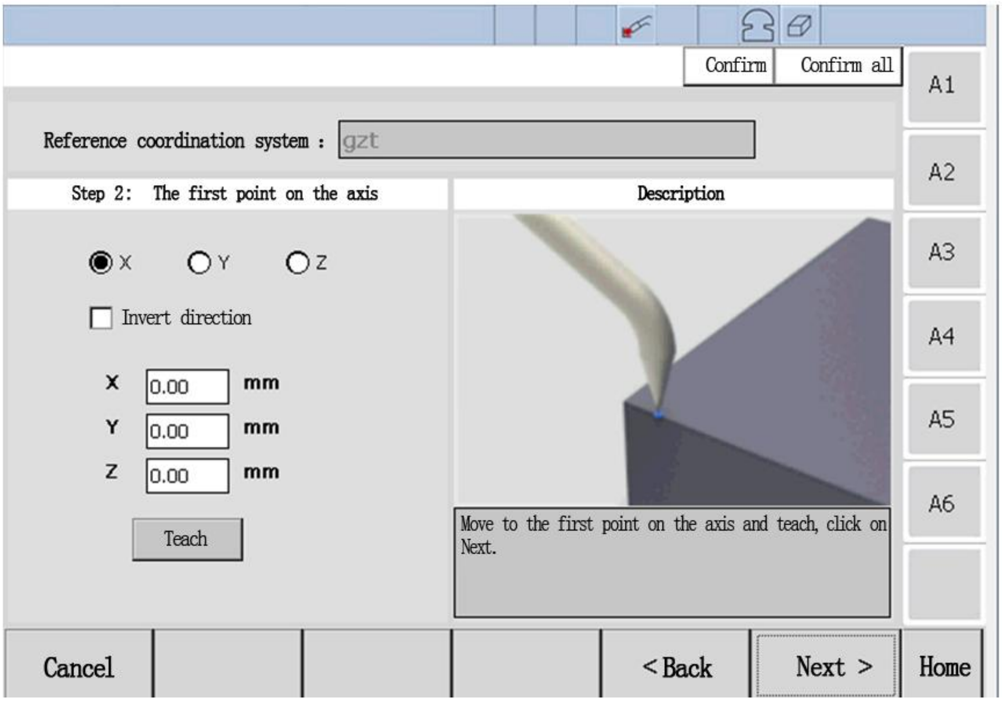
● Three-point method (without original point)

○ One-point method (hold the pose)

Tool: Default

1. Teaching the first point (the point on the first axis)
2. Teaching the second point (the point on the first axis)
3. Teaching the third point (the point on the second axis)

- Select the coordination system teaching method “Three-point method (without original point) and click on “Next” to enter into the interface as shown as below:



Reference coordination system : gzt

Step 2: The first point on the axis

● X ○ Y ○ Z

□ Invert direction

X 0.00 mm

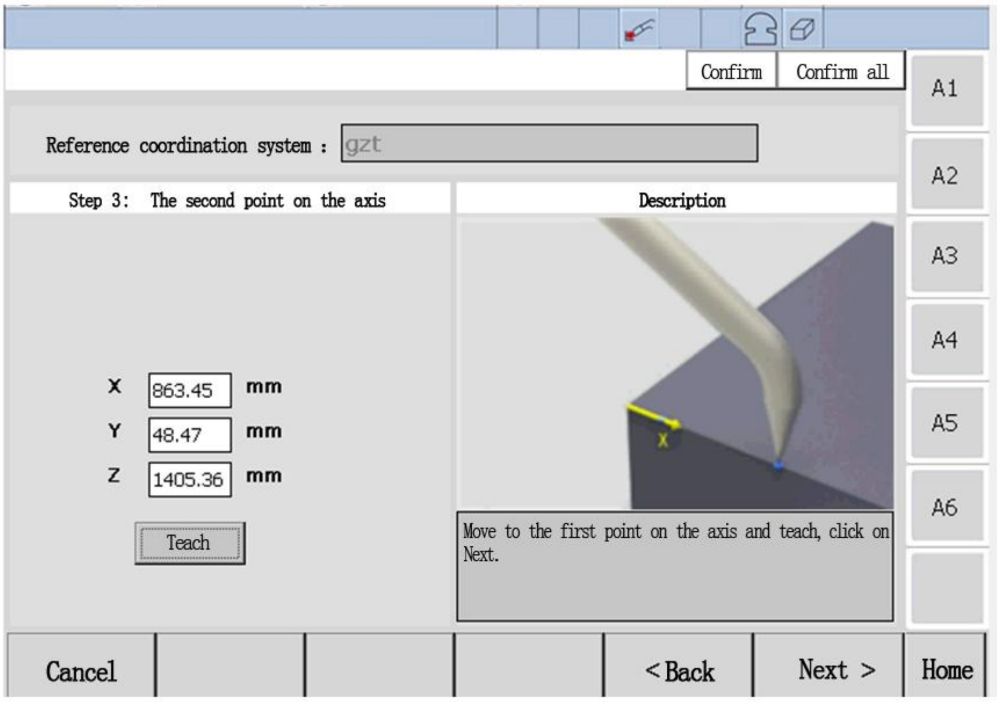
Y 0.00 mm

Z 0.00 mm

Description

Move to the first point on the axis and teach, click on Next.

- Firstly, select the axis (X, Y, Z) where the point for teaching locates, select the direction of the teaching axis (positive or negative), move the robot to the desired position and finally click on “Teach” to complete the teaching of the first point of the Three-point method. Click on “Cancel” to return to the initial interface of coordination system teaching; click on “Return” to return to the previous step (coordination system teaching method selecting interface).
- Click on “Next” to enter into the next teaching interface as shown in the following figure:



Reference coordination system : gzt

Step 3: The second point on the axis

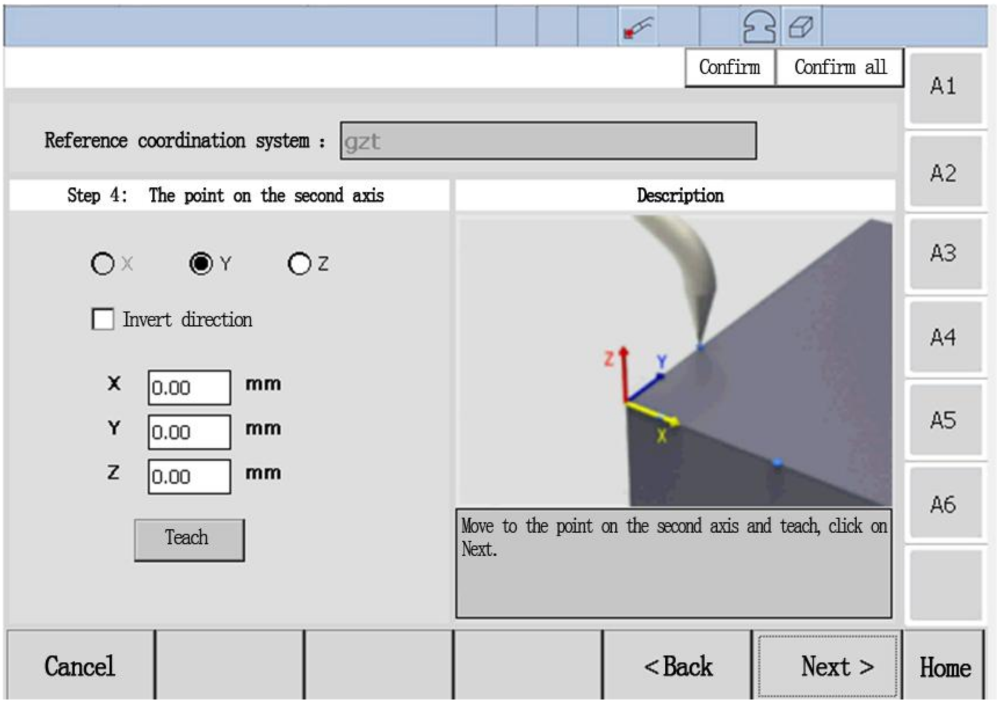
X 863.45 mm

Y 48.47 mm

Z 1405.36 mm

Move to the first point on the axis and teach, click on Next.

- Firstly move the robot to the desired position and click on "Teach" to complete the teaching of the second point. In this way, an axis of the coordination system can be determined through the points determined in this and last steps. Click on "Cancel" to return to the initial interface of coordination system teaching; click on "Return" to return to the previous step.
- Click on "Next" to enter into the next teaching interface as shown in the following figure:



Reference coordination system : gzt

Step 4: The point on the second axis

○ X ● Y ○ Z

□ Invert direction

X 0.00 mm

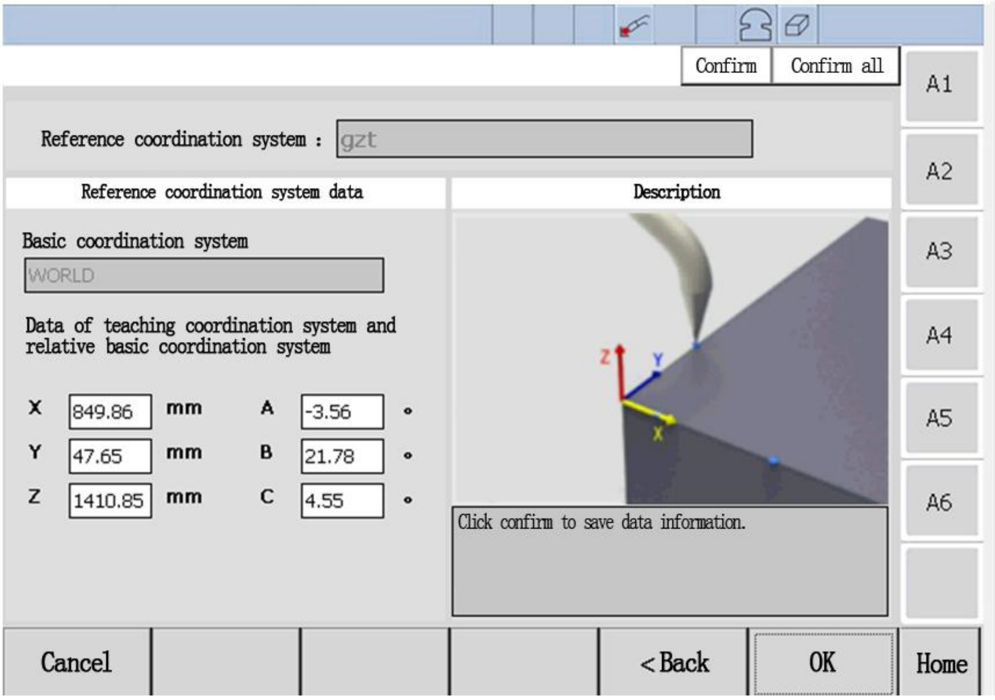
Y 0.00 mm

Z 0.00 mm

Move to the point on the second axis and teach, click on Next.

- Firstly, select the axis (X, Y, Z) where the point for teaching locates, select the direction of the teaching axis (positive or negative), move the robot to the desired position and finally click on “Teach” to complete the teaching of the third point of the Three-point method. Click on “Cancel” to return to the initial interface of coordination system teaching; click on “Return” to return to the second step interface of the three-point method teaching without original point.

- Click on “Next” to enter into the next teaching interface as shown in the following figure:



Reference coordination system : gzt

Basic coordination system: WORLD

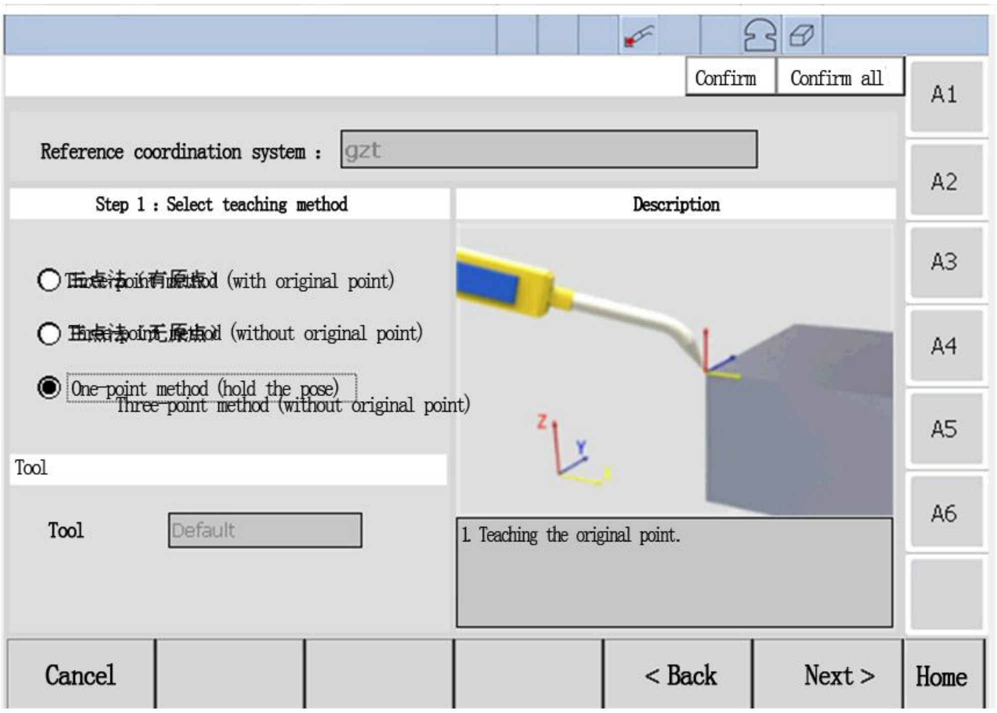
Data of teaching coordination system and relative basic coordination system

X	849.86	mm	A	-3.56	°
Y	47.65	mm	B	21.78	°
Z	1410.85	mm	C	4.55	°

Click confirm to save data information.

- What displays on the left side of the interface is the coordination system data after teaching; click on “Cancel” to return to the initial interface of coordination system teaching; click on “Return” to return to the previous step; click on “CONFIRM” to save data and complete the coordination system teaching.

3.4.3 One-point method (hold the pose)



Reference coordination system : gzt

Step 1 : Select teaching method

○ Three-point method (with original point)

○ Three-point method (without original point)

● One-point method (hold the pose)

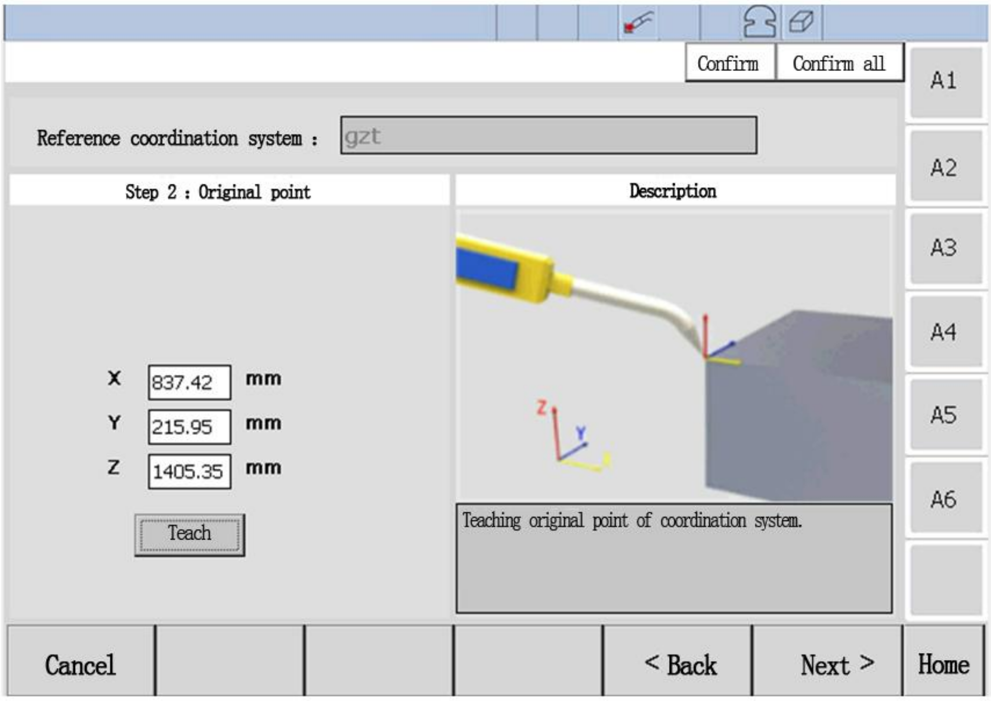
Tool

Tool Default

Description

1. Teaching the original point.

- Select coordination system teaching method "one-point method (hold the pose)" and click on "Next" to enter into the following interface as shown as below:



Reference coordination system : gzt

Step 2 : Original point

X 837.42 mm

Y 215.95 mm

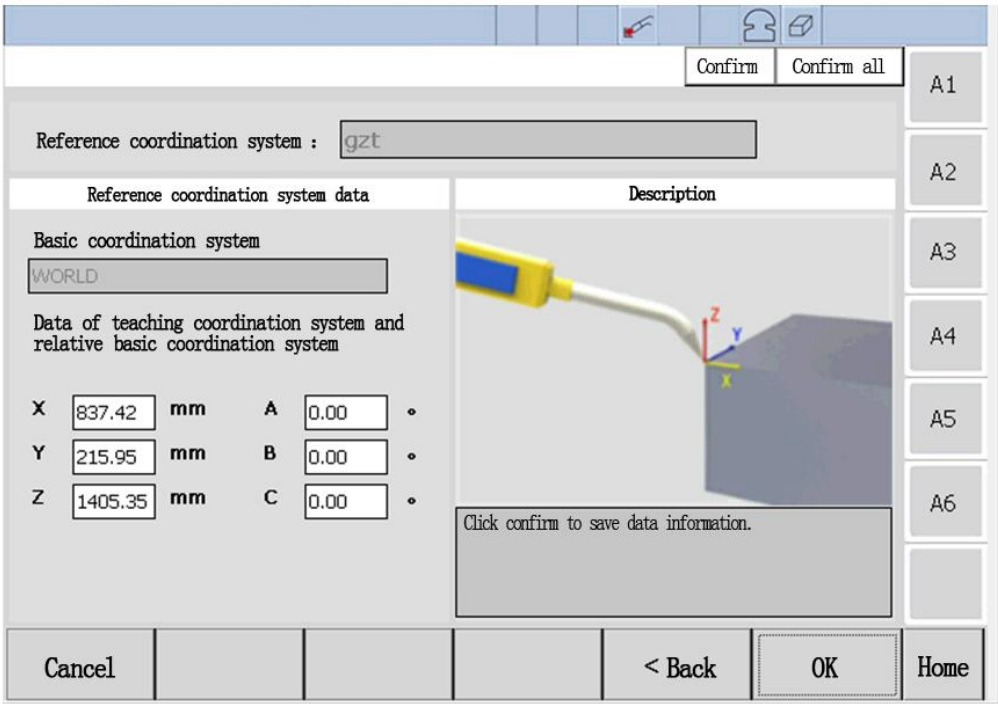
Z 1405.35 mm

Description

Teaching original point of coordination system.

- Move the robot to the desired position, click on "Teach" button to complete the teaching of the original point position of coordination system and the coordination system pose remains unchanged; click on "Cancel" to return to the initial interface of coordination system teaching; click on “Return” to return to the previous step (coordination system teaching method selecting interface).

- Click on “Next” to enter into the next teaching interface as shown in the following figure:



Reference coordination system : gzt

Reference coordination system data

Basic coordination system: WORLD

Data of teaching coordination system and relative basic coordination system

X	837.42	mm	A	0.00	°
Y	215.95	mm	B	0.00	°
Z	1405.35	mm	C	0.00	°

Description

Click confirm to save data information.

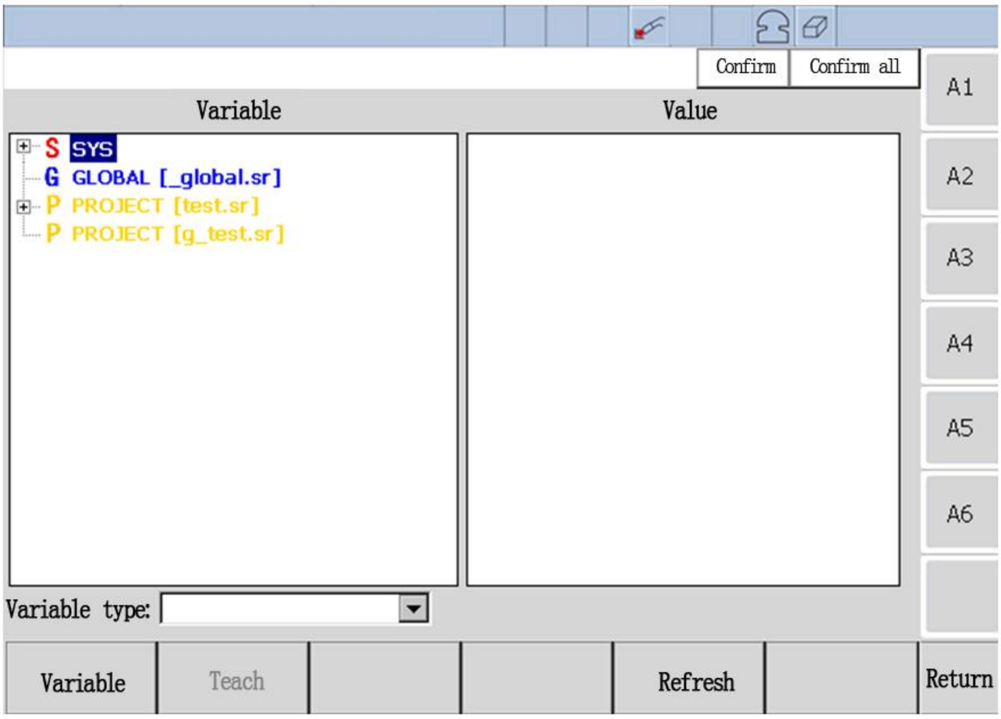
- What displays on the left side of the interface is the coordination system data after teaching; click on “Cancel” to return to the initial interface of coordination system teaching; click on “Return” to return to the previous step; click on “CONFIRM” to save data and complete the coordination system teaching.

3.5 Variable

Press X= to pop up the options of “Tool”, “Coordination system” and “Variable” as shown in the figure

3.5.1 Variable classification and notices

The variable is used to maintain the program variable data of demonstrator. It is as shown in the figure when selecting a variable



Note:

① **The variable in the robot system is divided into four types: system variable, global variable, engineering variable and program variable.**

② **The program variables between different programs can be the same name.**

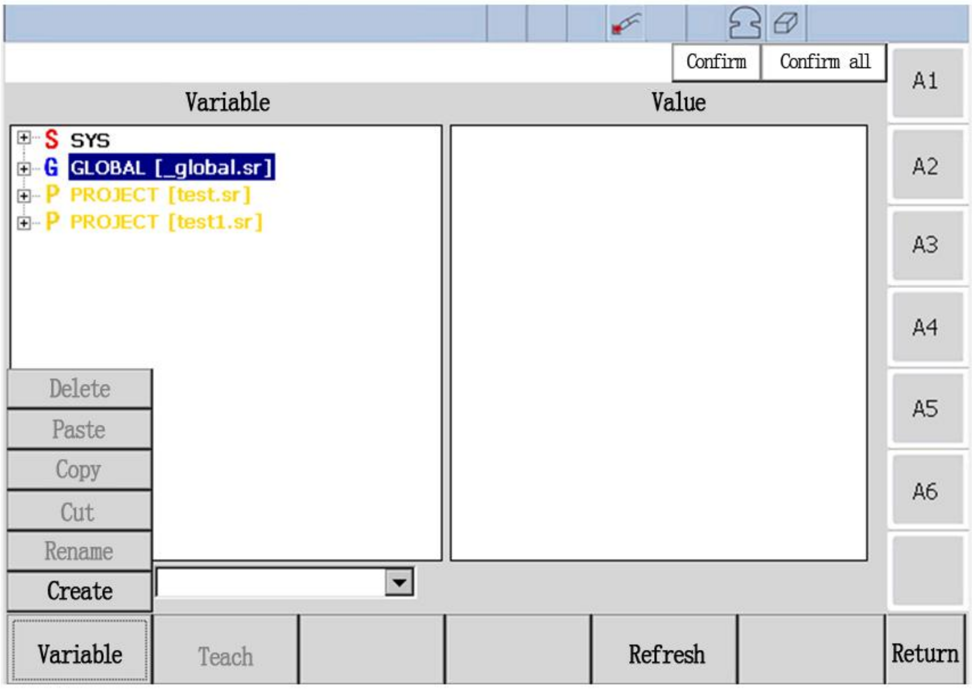
③ **The program variable under certain engineering cannot be named the same as the engineering variable of the engineering.**

④ **Any variable cannot be named the same as the global variable.**

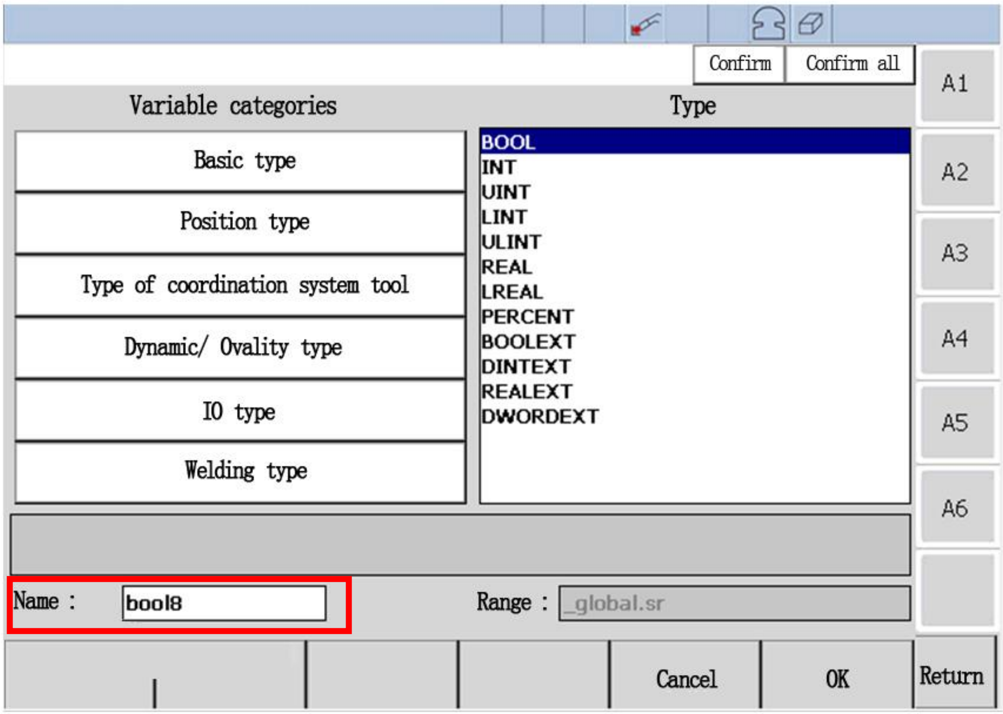
⑤ **The name of the system variable is fixed, for example, “WORLD”, “ROBOTBASE” AND “EASYS”. And the system variable cannot be renamed or deleted.**

3.5.2 Variable operation

- Creation of variable. Select the desired variable type and click on “Create” as shown in the following figure:



- Click on “Create” to pop up variable creation interface as shown in the following figure:



Name : bool8

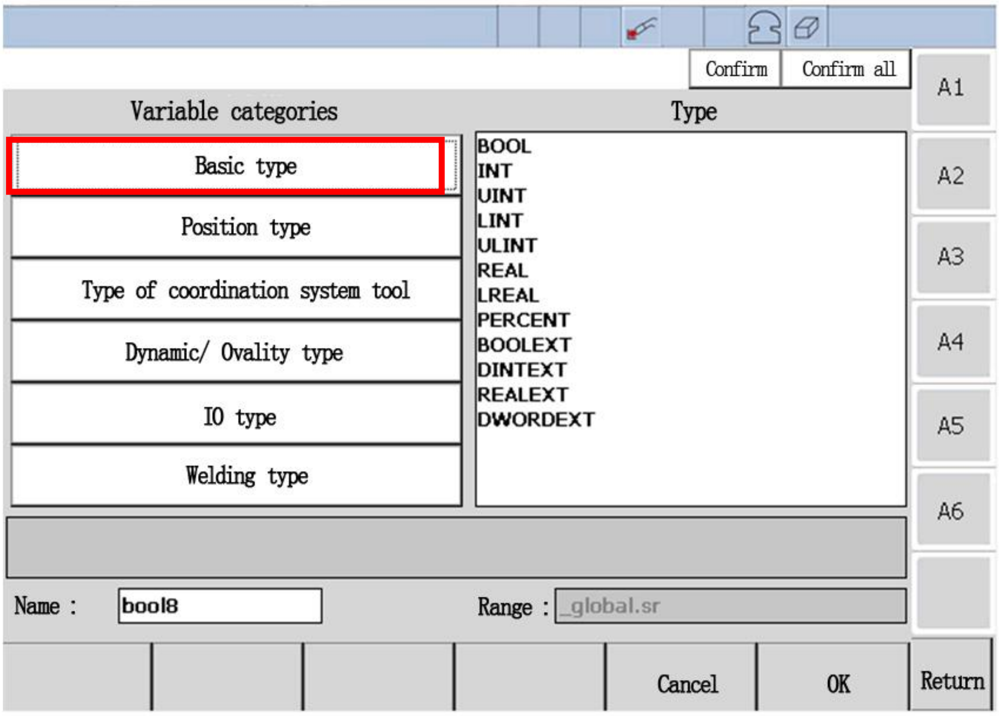
Range : _global.sr

⚠ **Note:**

① **The variable can be renamed but it is still subject to the rules of variable** naming mentioned in the last section;

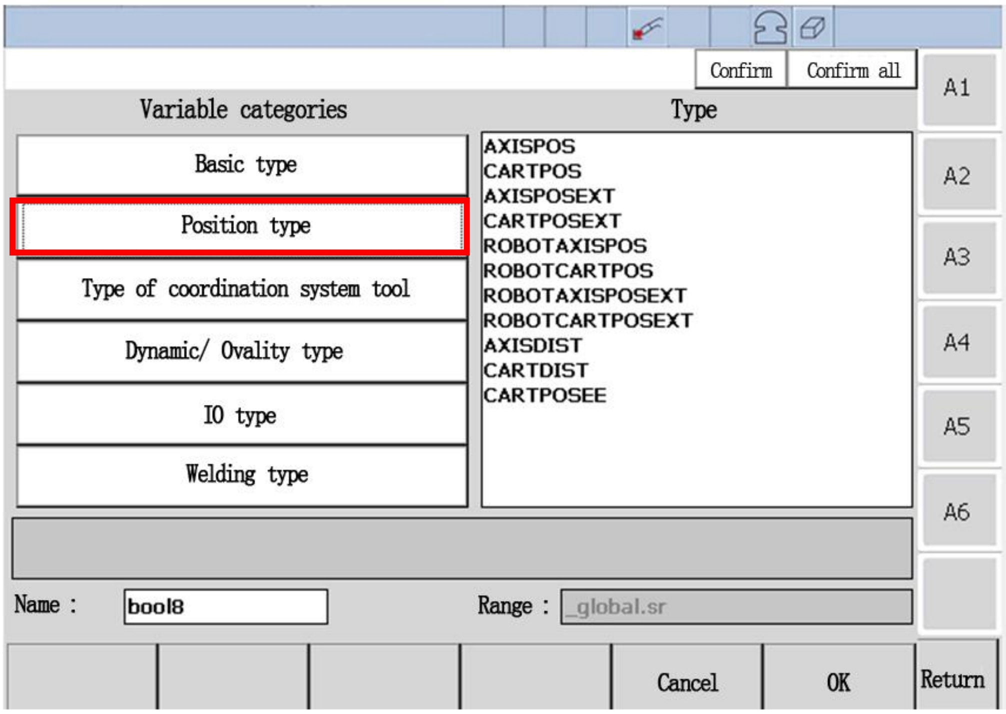
② The left button in the above figure is used to select the large type of variable and the right side of the interface lists the specific variable types under the large type. What is shown in the above figure is the variable types under the corresponding “basic type”. Input variable name and click on “CONFIRM” to complete the creation of variable. Click “Cancel” to cancel the creation and variable and quit the variable creation interface.

③ The variable type is divided into basic type, position type, coordination system tool type, dynamic smooth type, IO type and welding type which are as shown in the following figure separately



Name : bool8

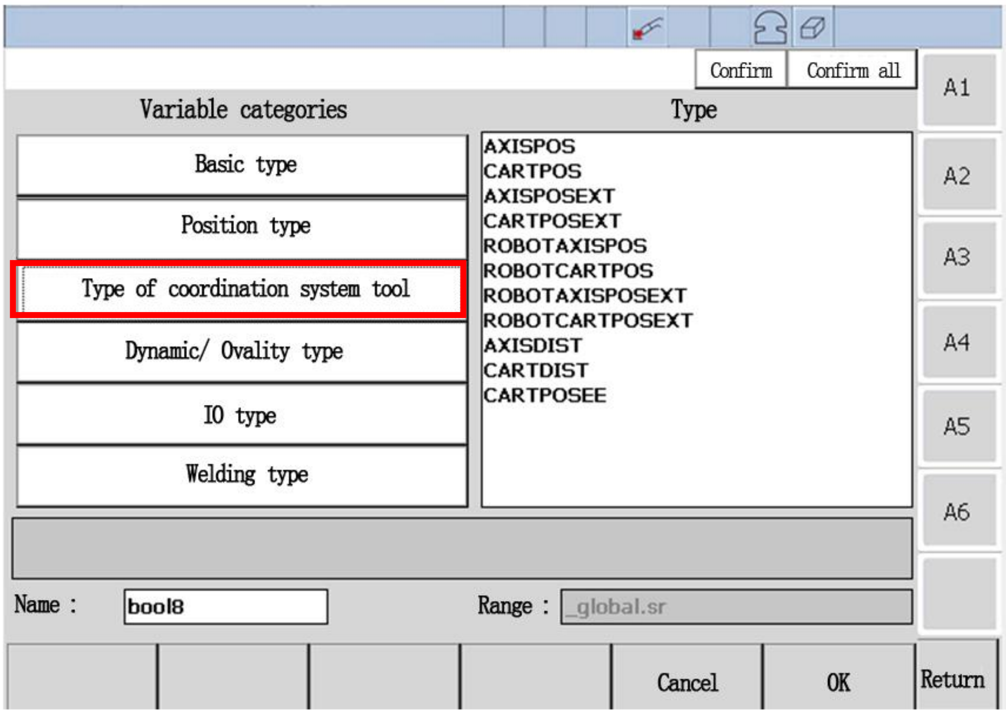
Range : _global.sr



AXISPOS
CARTPOS
AXISPOSEXT
CARTPOSEXT
ROBOTAXISPOS
ROBOTCARTPOS
ROBOTAXISPOSEXT
ROBOTCARTPOSEXT
AXISDIST
CARTDIST
CARTPOSEE

Name : bool8

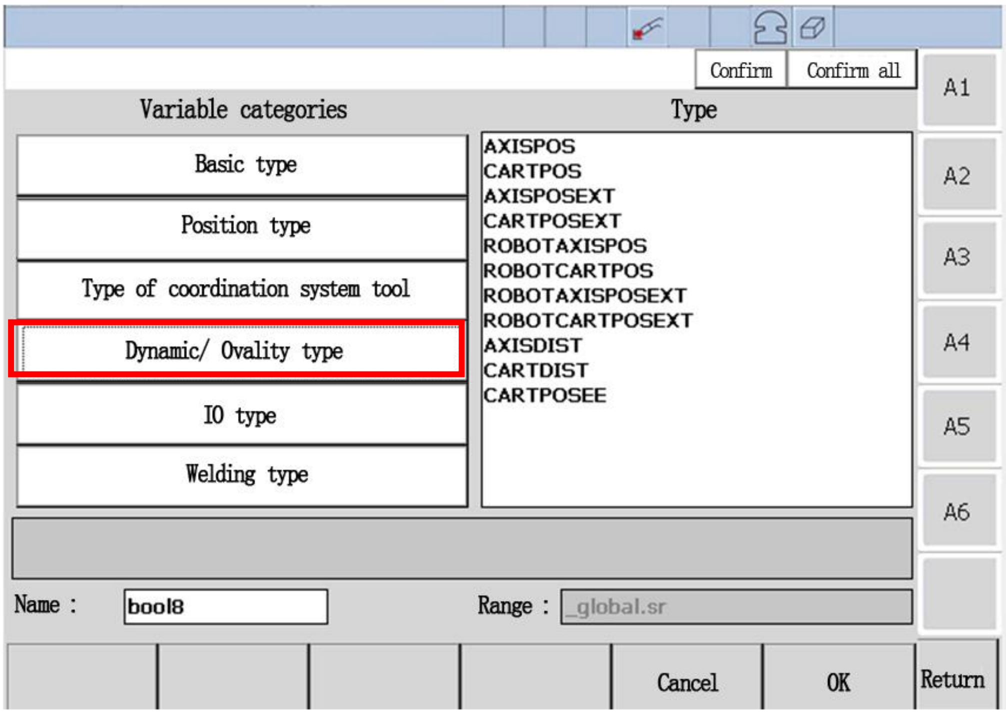
Range : _global.sr



AXISPOS
CARTPOS
AXISPOSEXT
CARTPOSEXT
ROBOTAXISPOS
ROBOTCARTPOS
ROBOTAXISPOSEXT
ROBOTCARTPOSEXT
AXISDIST
CARTDIST
CARTPOSEE

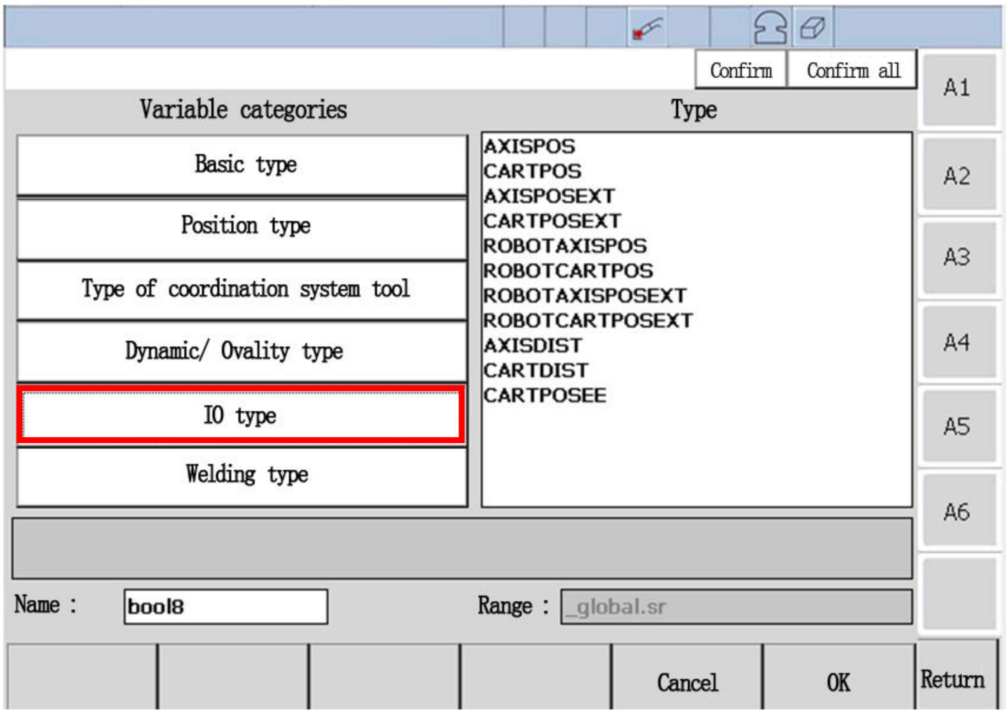
Name : bool8

Range : _global.sr



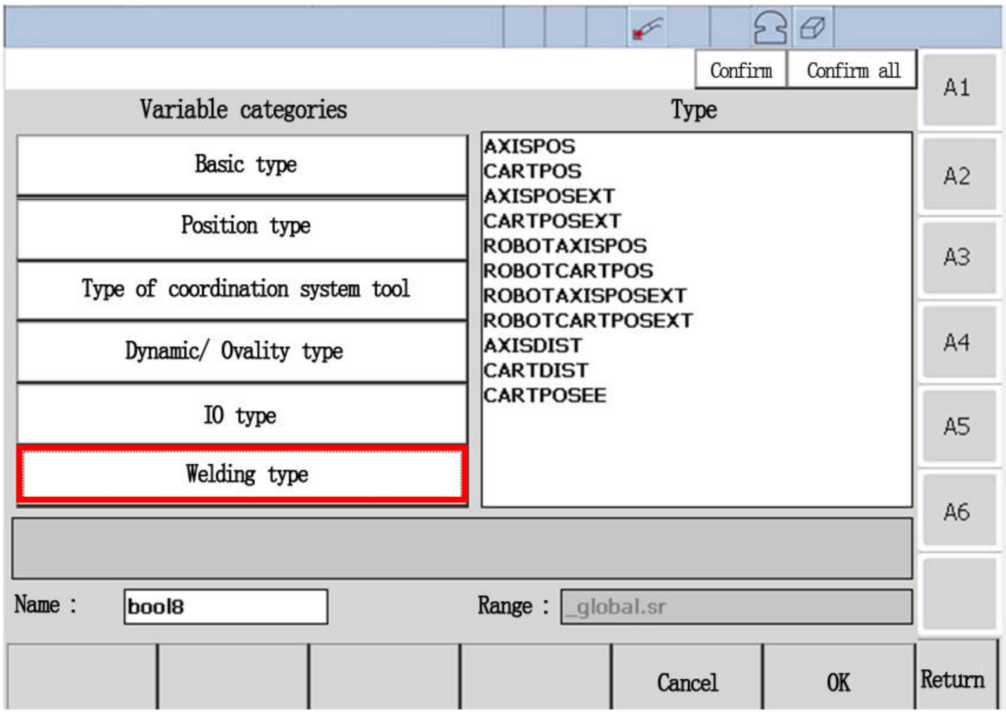
AXISPOS
CARTPOS
AXISPOSEXT
CARTPOSEXT
ROBOTAXISPOS
ROBOTCARTPOS
ROBOTAXISPOSEXT
ROBOTCARTPOSEXT
AXISDIST
CARTDIST
CARTPOSEE

Name : bool8 Range : _global.sr



AXISPOS
CARTPOS
AXISPOSEXT
CARTPOSEXT
ROBOTAXISPOS
ROBOTCARTPOS
ROBOTAXISPOSEXT
ROBOTCARTPOSEXT
AXISDIST
CARTDIST
CARTPOSEE

Name : bool8 Range : _global.sr

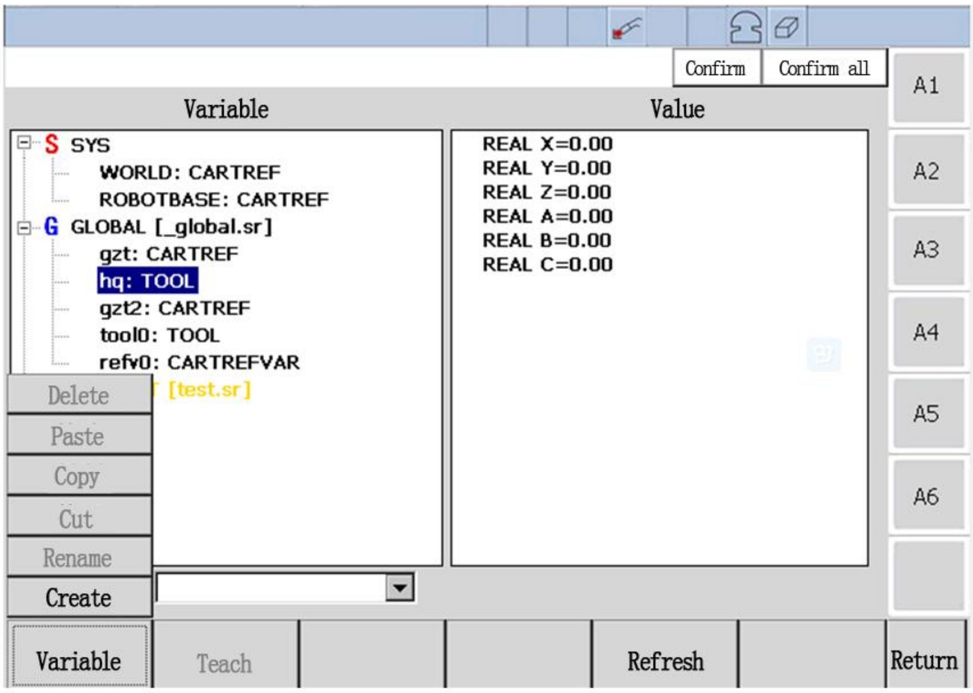


AXISPOS
CARTPOS
AXISPOSEXT
CARTPOSEXT
ROBOTAXISPOS
ROBOTCARTPOS
ROBOTAXISPOSEXT
ROBOTCARTPOSEXT
AXISDIST
CARTDIST
CARTPOSEE

Name : bool8

Range : _global.sr

- Copy of variable: select a variable, click on “Copy” button and the variable will be copied into the clipboard as shown in the figure

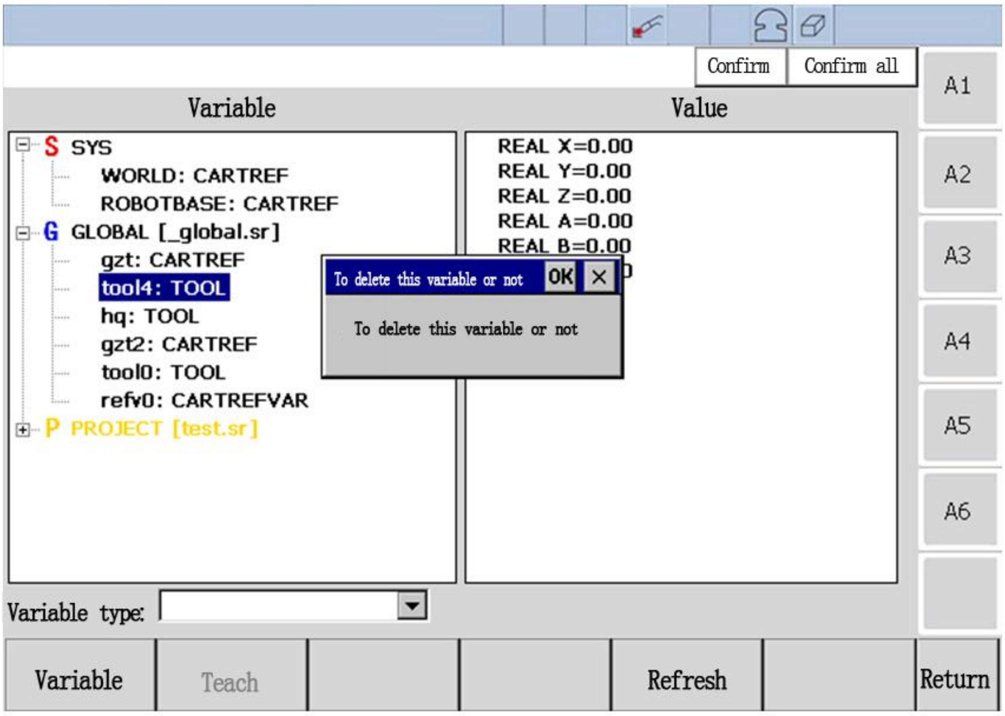


S SYS
WORLD: CARTREF
ROBOTBASE: CARTREF
G GLOBAL [_global.sr]
gzt: CARTREF
hq: TOOL
gzt2: CARTREF
tool0: TOOL
refv0: CARTREFVAR
[test.sr]

REAL X=0.00
REAL Y=0.00
REAL Z=0.00
REAL A=0.00
REAL B=0.00
REAL C=0.00

- Cut of variable: select the variable which is not occupied by the program and this variable can be cut and pasted; the variable which is occupied by the program cannot be cut but it will be judged if the paste is valid, if not, the paste will not be operated; the rule for cut is: the program variable can only be cut into under the engineering or the global situation. The engineering variable can only be cut into the global situation.

- Paste of variable: select a name of engineering or program and then it can be pasted from the clipboard under the engineering or program. A dialog box for change of variable name will appear when pasting.
- Delete of variable: the variable which is occupied by the program cannot be deleted as shown in the figure



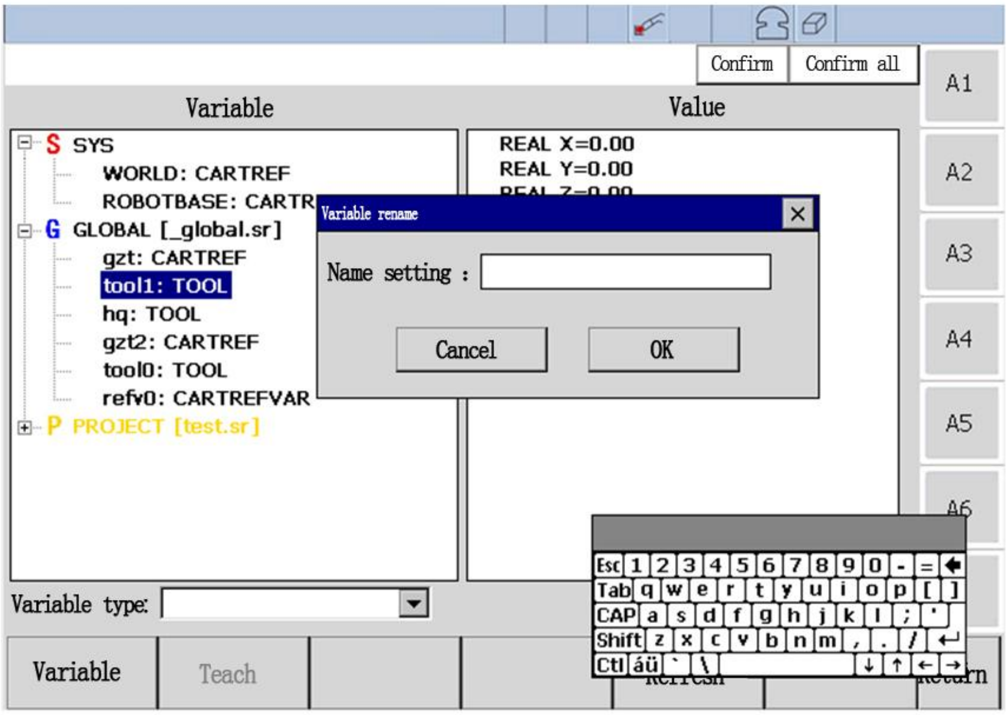
```
S SYS
    WORLD: CARTREF
    ROBOTBASE: CARTREF
G GLOBAL [_global.sr]
    gzt: CARTREF
    tool4: TOOL
    hq: TOOL
    gzt2: CARTREF
    tool0: TOOL
    refv0: CARTREFVAR
P PROJECT [test.sr]
```

```
REAL X=0.00
REAL Y=0.00
REAL Z=0.00
REAL A=0.00
REAL B=0.00
```

To delete this variable or not

To delete this variable or not

- Rename of variable: a selected variable can be renamed as shown in the figure:


Variable

- S SYS
  - WORLD: CARTREF
  - ROBOTBASE: CARTR
- G GLOBAL [_global.sr]
  - gzt: CARTREF
  - tool1: TOOL
  - hq: TOOL
  - gzt2: CARTREF
  - tool0: TOOL
  - refv0: CARTREFVAR
- P PROJECT [test.sr]

Value

```
REAL X=0.00
REAL Y=0.00
REAL Z=0.00
```

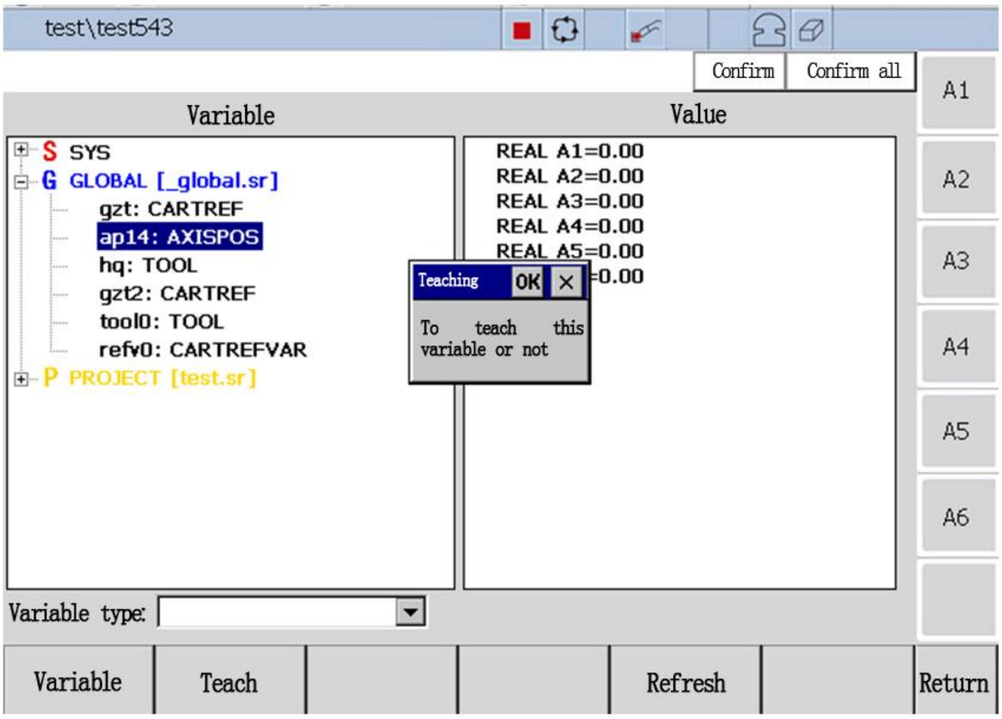
Variable rename

Name setting :

Variable type:

Confirm
Confirm all
Cancel
OK
A1
A2
A3
A4
A5
A6
Variable
Teach
Refresh
Return
Esc 1 2 3 4 5 6 7 8 9 0 - =
Tab q w e r t y u i o p [ ]
CAP a s d f g h j k l ; '
Shift z x c v b n m , . /
Ctl áü ` \ ↓ ↑ ← →

- Position variable teaching: select a position variable type (for example: ROBOTAXISPOS), move the robot to the desired position and then click on “Teaching” button to complete the variable teaching function, teaching the current position as shown in the figure:


test\test543

Variable

- S SYS
- G GLOBAL [_global.sr]
  - gzt: CARTREF
  - ap14: AXISPOS
  - hq: TOOL
  - gzt2: CARTREF
  - tool0: TOOL
  - refv0: CARTREFVAR
- P PROJECT [test.sr]

Value

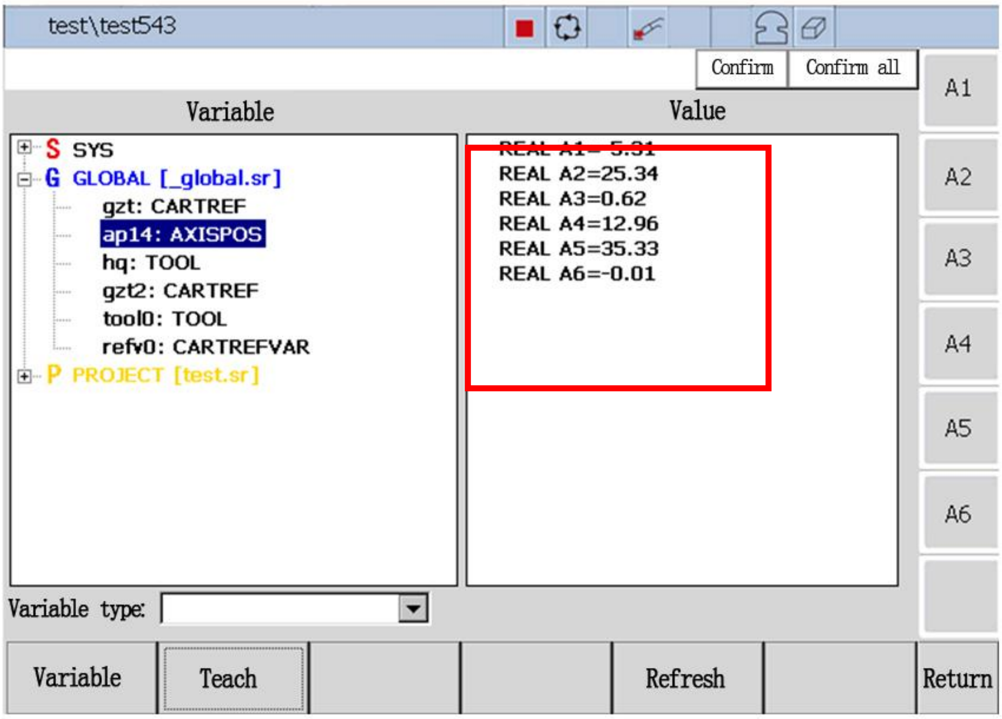
```
REAL A1=0.00
REAL A2=0.00
REAL A3=0.00
REAL A4=0.00
REAL A5=0.00
REAL A6=0.00
```

Teaching

To teach this variable or not

Variable type:

- Teaching finishes as shown in the figure:

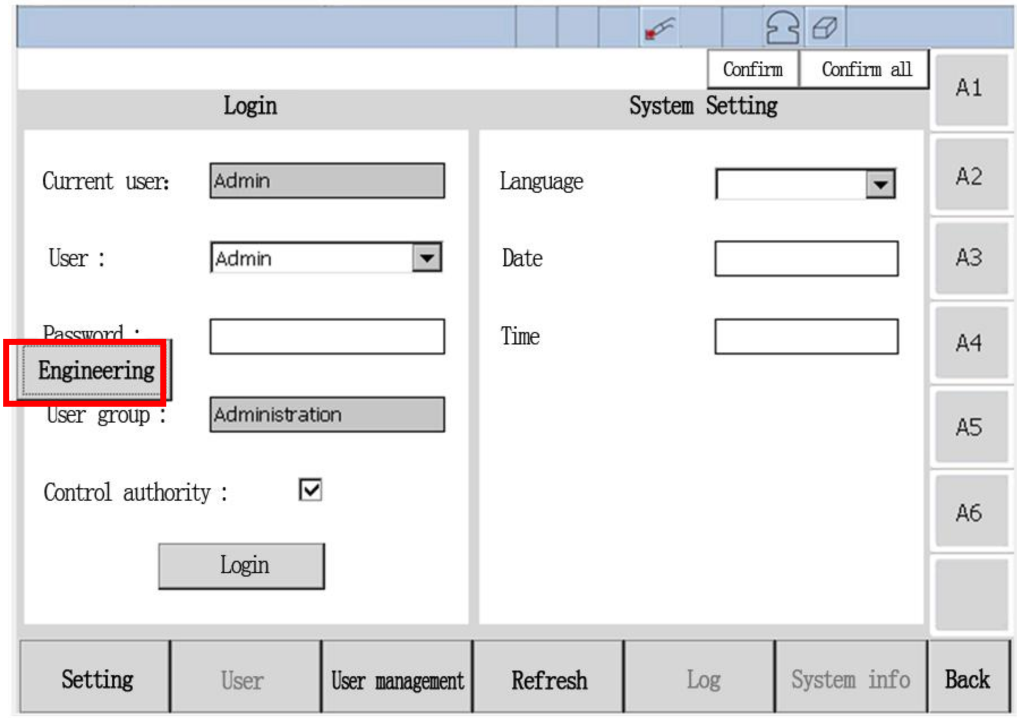


test\test543

```
REAL A1=5.31
REAL A2=25.34
REAL A3=0.62
REAL A4=12.96
REAL A5=35.33
REAL A6=-0.01
```

3.6 Engineering

- Press the button on the left side of the demonstrator as shown in the figure



Current user: Admin

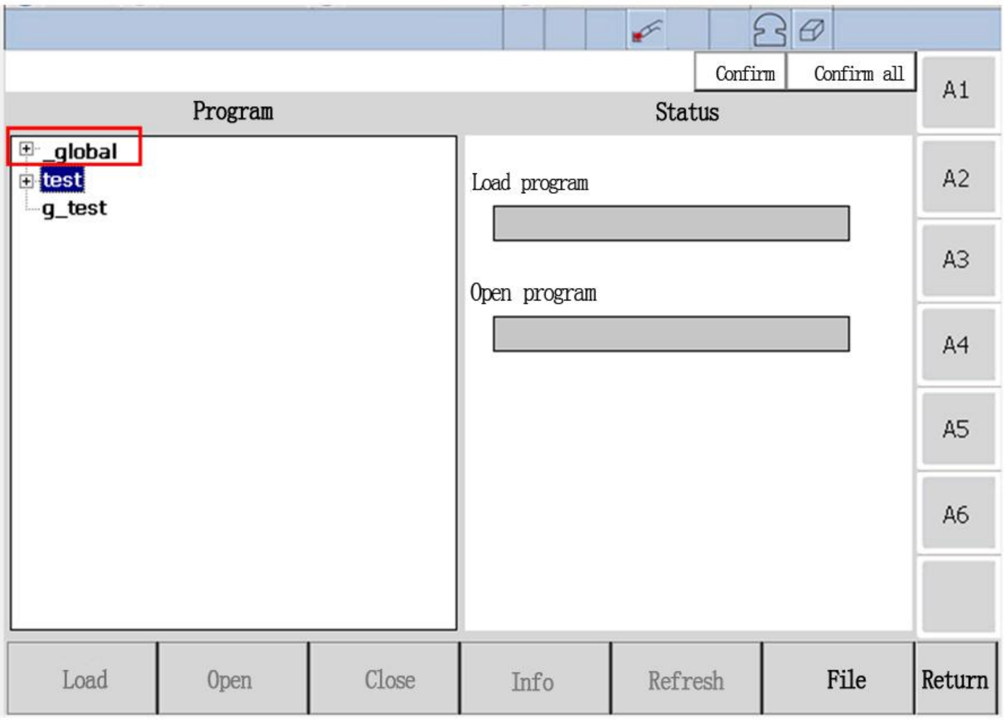
User : Admin

Password :

User group : Administration

Control authority : ☑

- Single click on “Engineering” to enter into the engineering interface as shown in the figure.

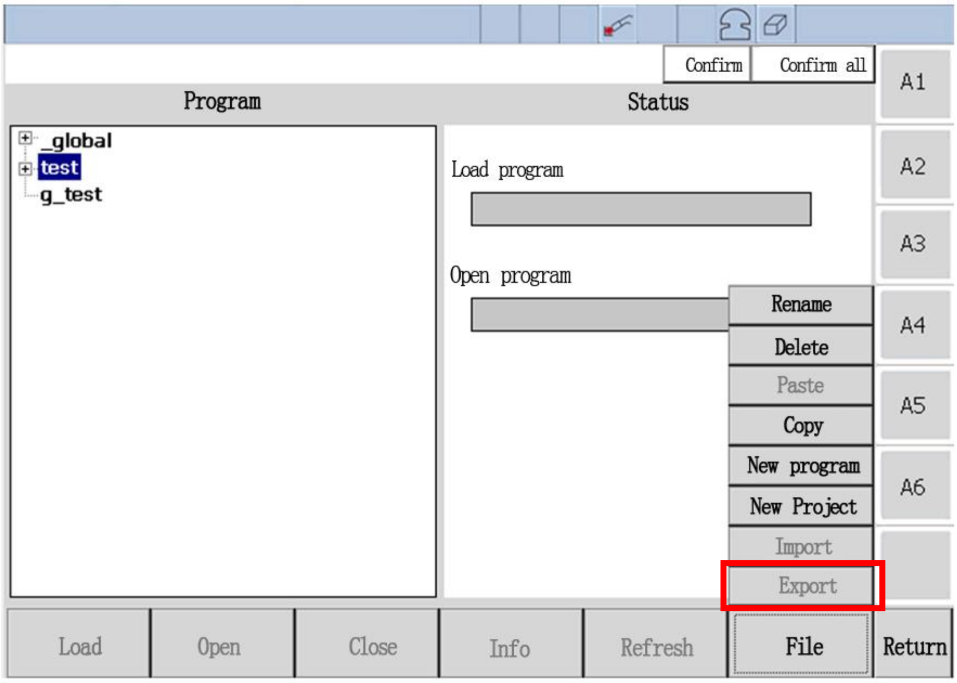


Note:

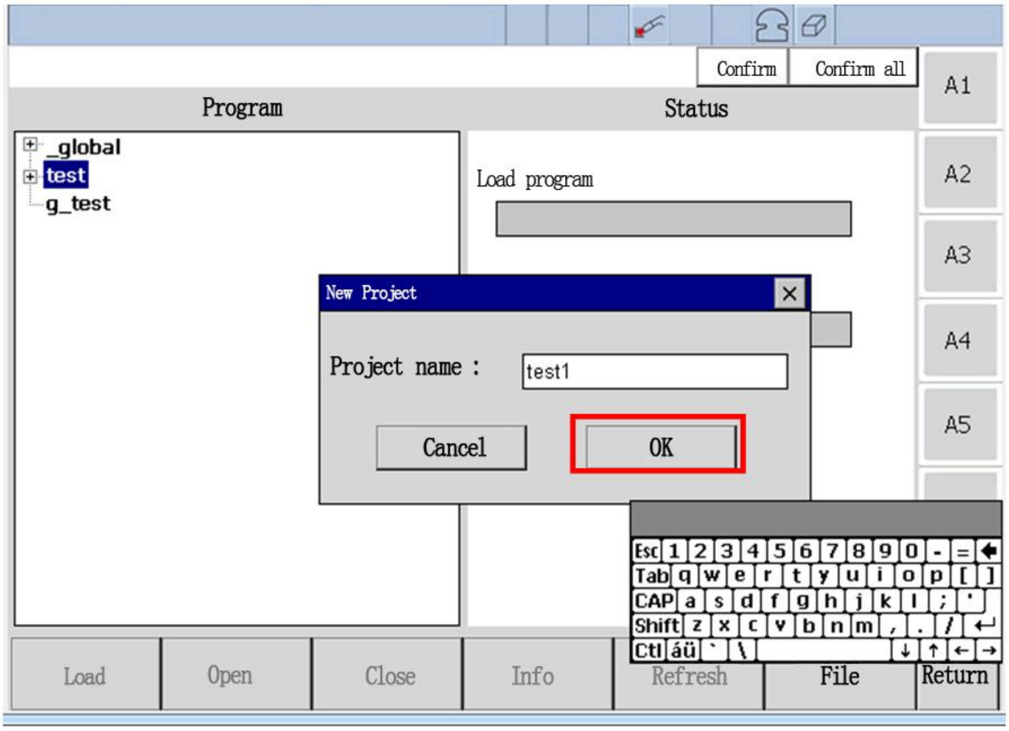
“_global” means the global engineering. This engineering always exists and user cannot delete or rename it.

3.6.1 New Project

- Click the "New Project" in the "File", as shown below.

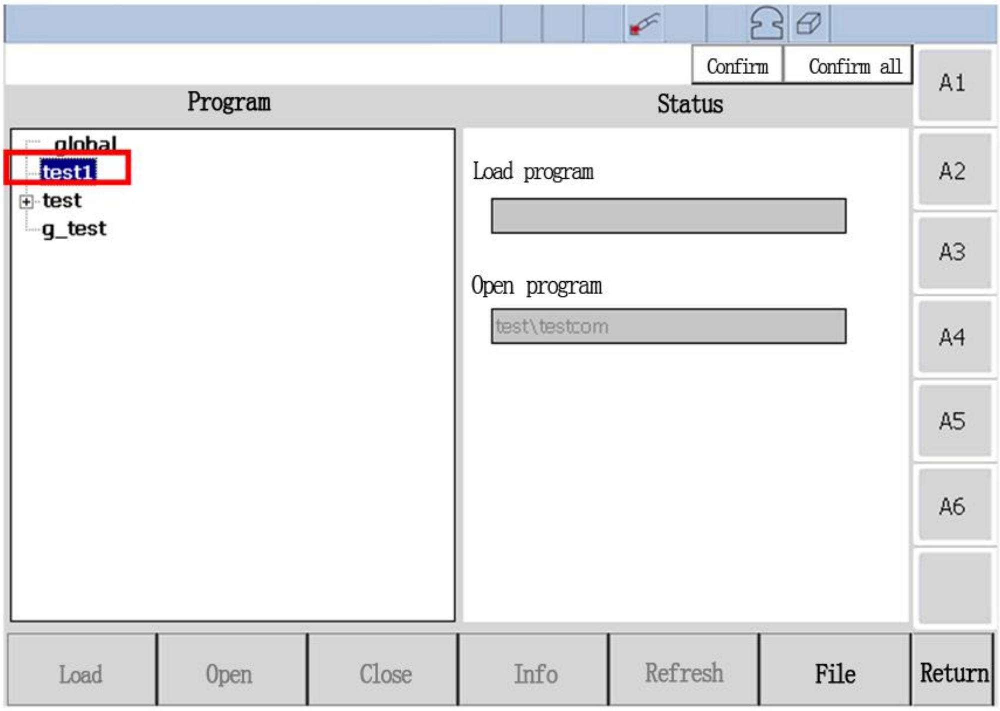


- Select "New Project" and the window pops up, as shown in figure below.



Project name : test1

- Enter the project name and click "OK", the new project is completed, as shown in Figure.

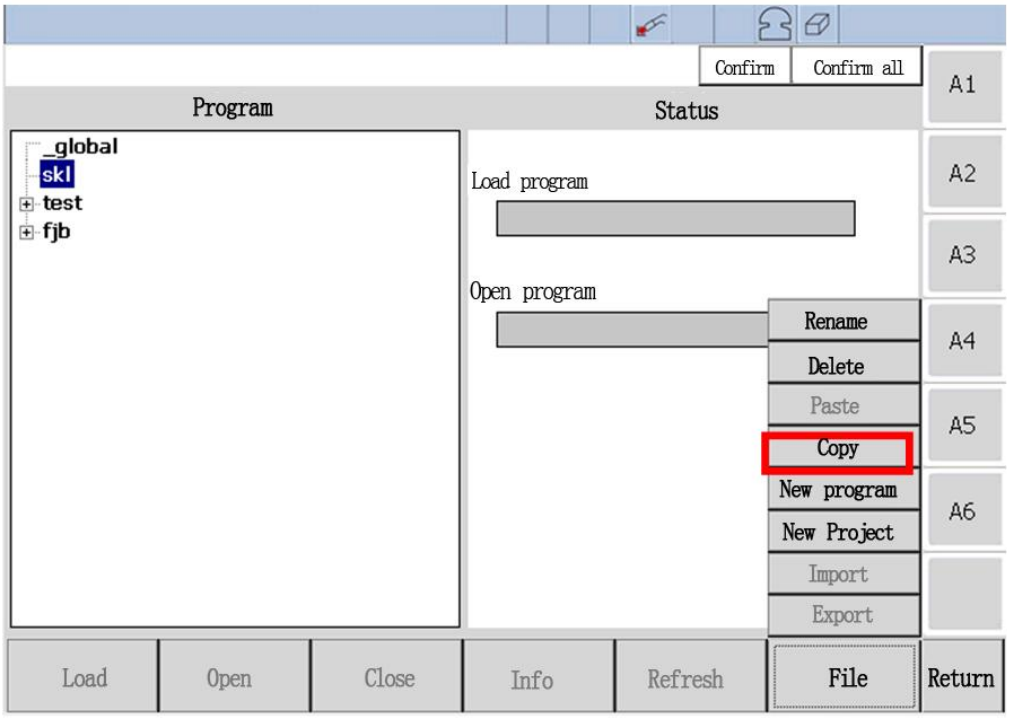


Load program

Open program test\test.com

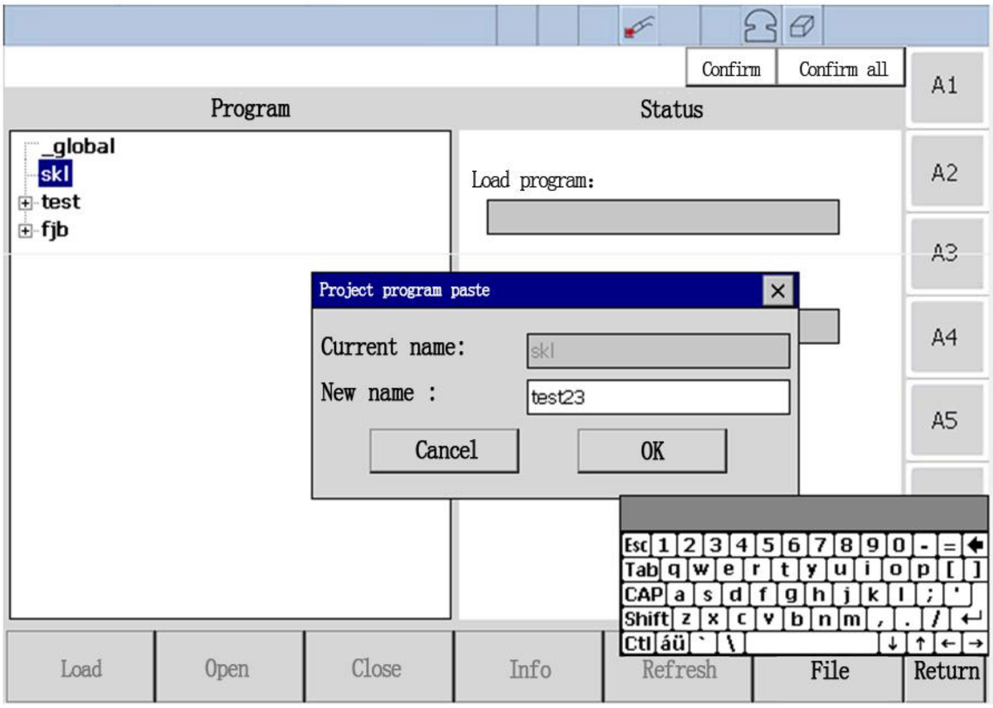
3.6.2 Copy Project

- Select the project you want to copy, such as the “skl” in figure below, click the "Copy" in the "File", and then the project is copied to the clipboard, as the figure below shows.



3.6.3 Paste Project

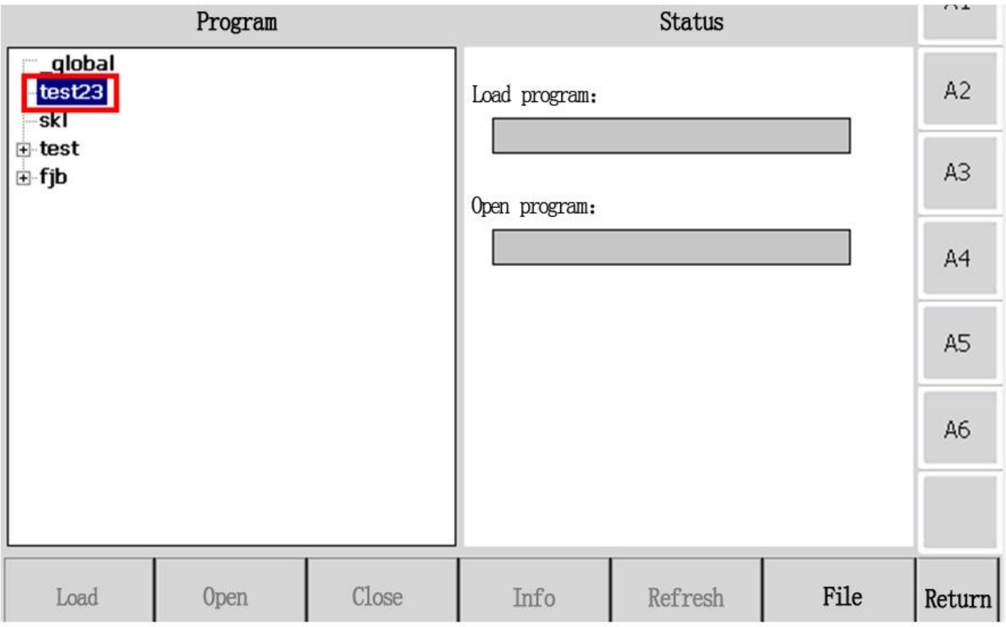
- When the project is copied to the clipboard, choose the place where you want to paste, select "Paste" in the "File", and the following dialog box pops up prompting you to enter a new name. Click "OK" to complete the paste. As shown in figure below:


Project program paste

Current name: skl

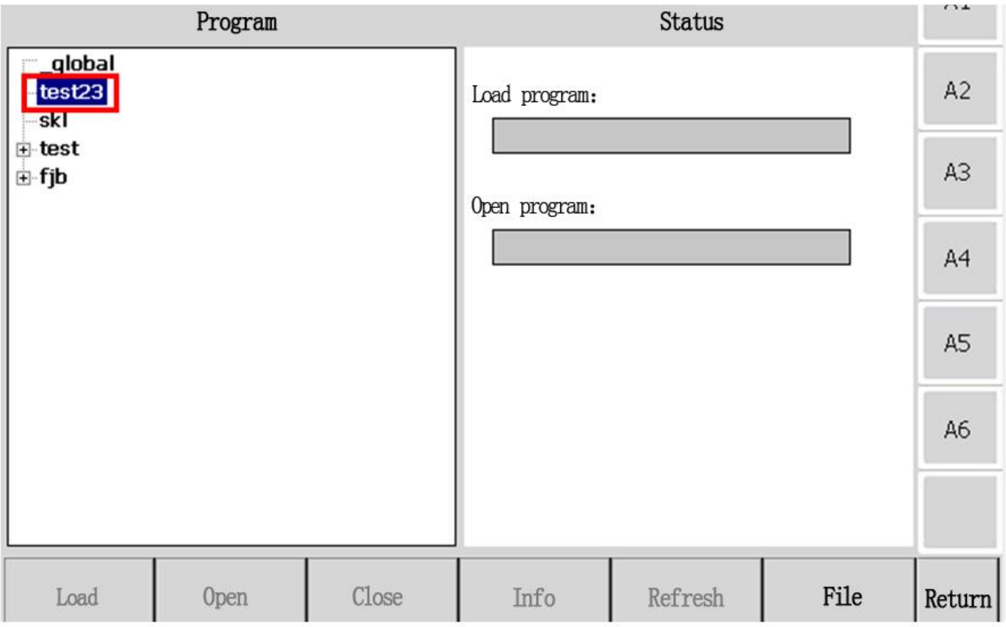
New name : test23

- The figure below shows the status after copy and paste.

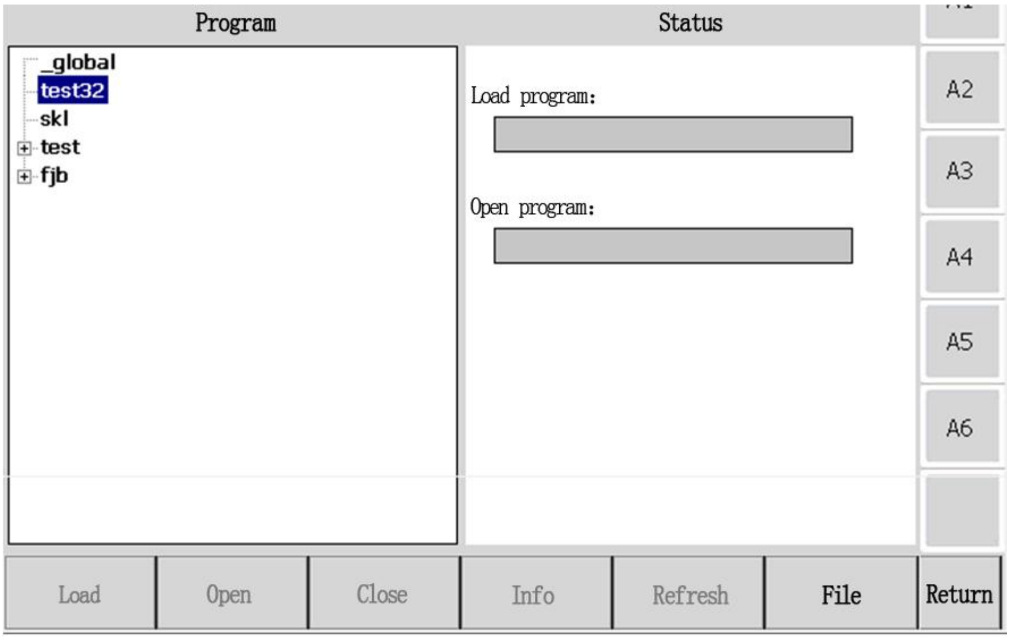


3.6.4 Rename Project

- Select the project you want to rename, click "Rename" in the "File", and the dialog box as the figure below shows appears, at this time you need to modify the new name, and click "OK."

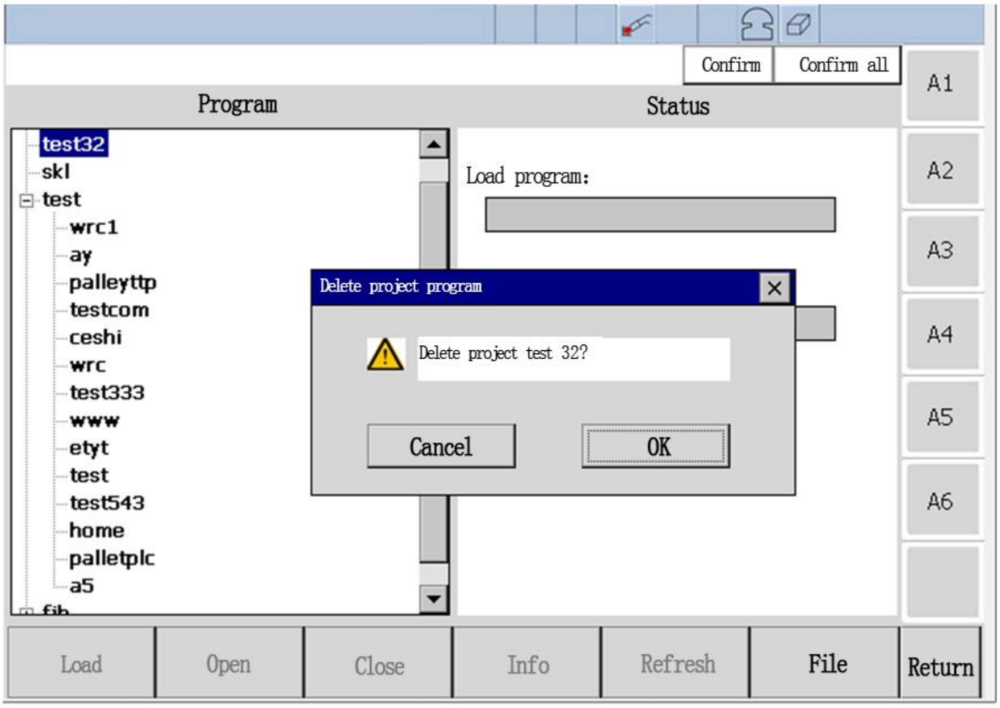


- Following modifications appears for the project directory upon renaming, and "test23" is modified as "test32".



3.6.5 Delete Project

- Select the project you want to delete, click on the "Delete" in the "File". When the following dialog box appears, click "OK" and confirm to delete the project.



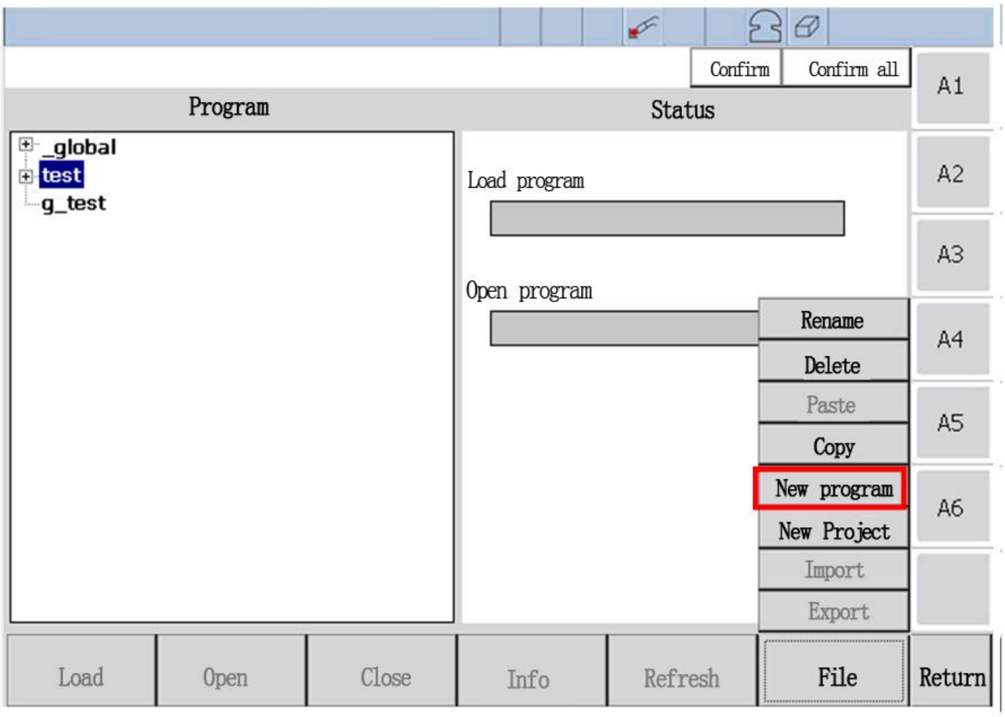
Delete project program

Delete project test 32?

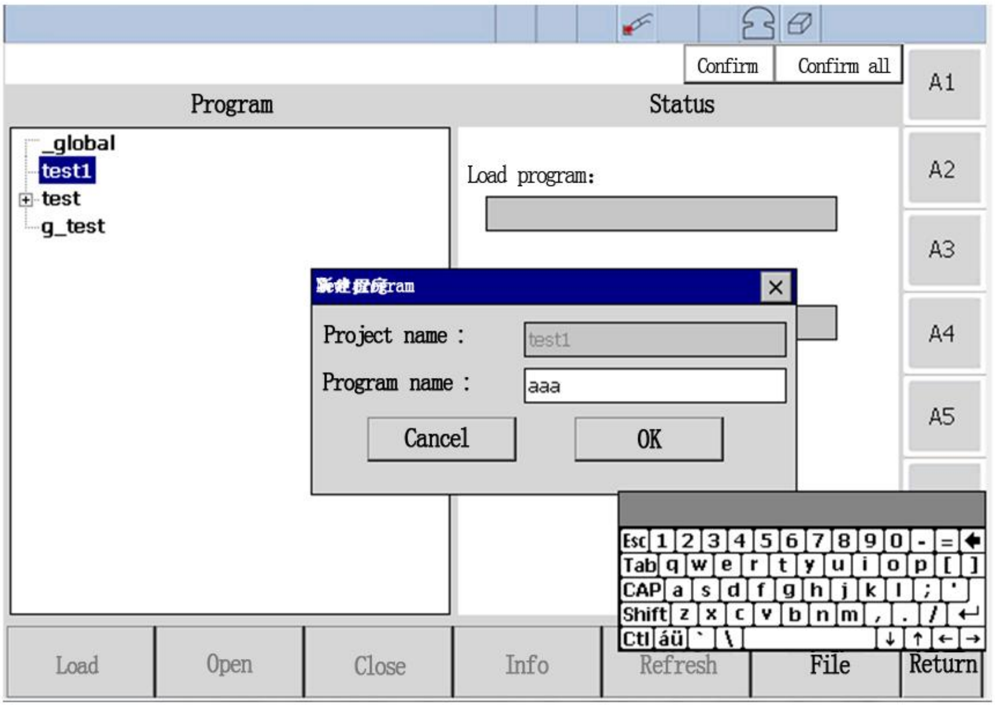
3.7 Program

3.7.1 New Program

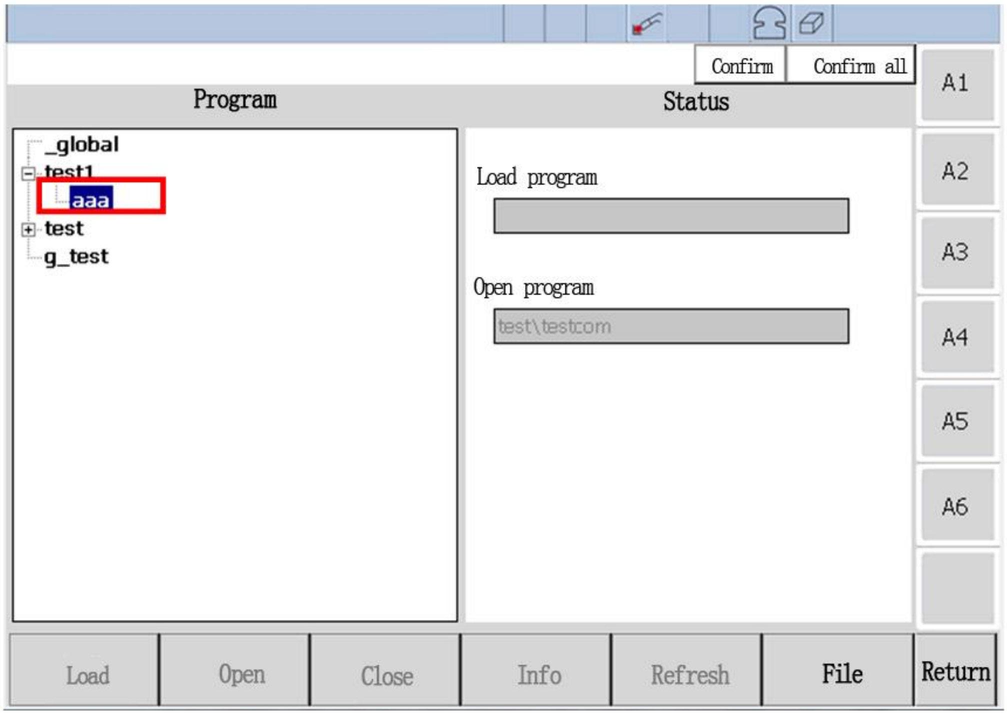
- Select the new project directory, and click the "New Program" in the "File", as shown in figure below.



- Dialog box as the figure below shows pops up, then enter the program name, as shown in the following figure.



- Click "OK", and new program is completed, as shown in figure below.

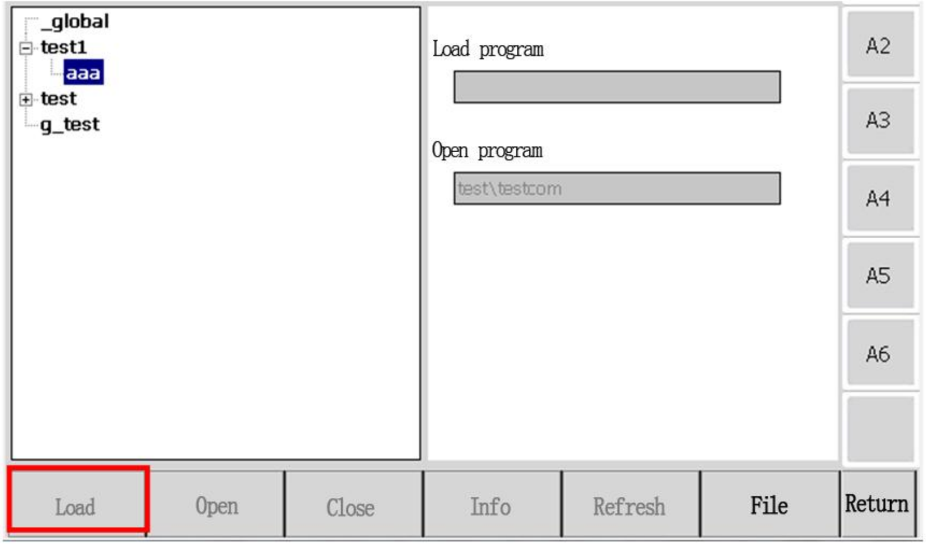


Note:

Editing operation is also available for the program (new, rename, delete, copy, etc.), which is similar with the edit operations of the project, so no detailed introduction will be given.

3.7.2 Load Program

First, you need to select the project that has been loaded in the project directory, and then click the "Load". If the program has been modified, and there will be prompted to save the program, as the figure shows.

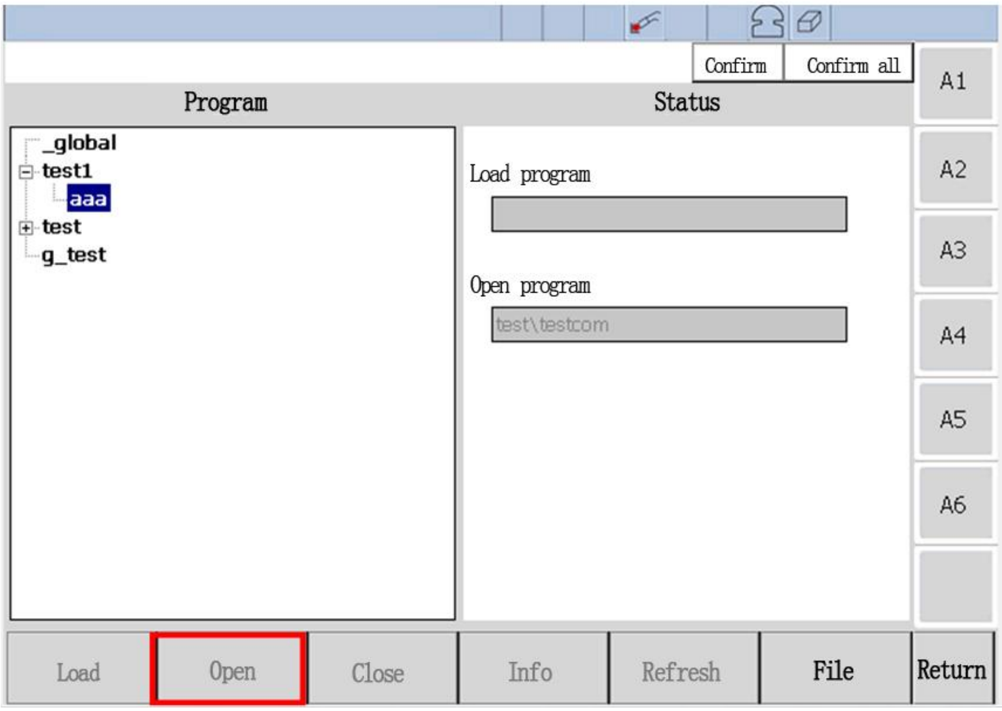


- ➢ Notes:

There’s only one program being loaded at the same time. If the program A has been loaded, and if you want to load the program B, then you may load program B after closing the program A; in case of syntax error existed to the program to be loaded, there will be an error dialog box popping up to prompt the line where the statements are located, and enter into the open state interface of the program.

3.7.3 Open Program

First, you need to select in the program (not being loaded) in the project directory tree, and then click on the "Open", as shown in Figure:

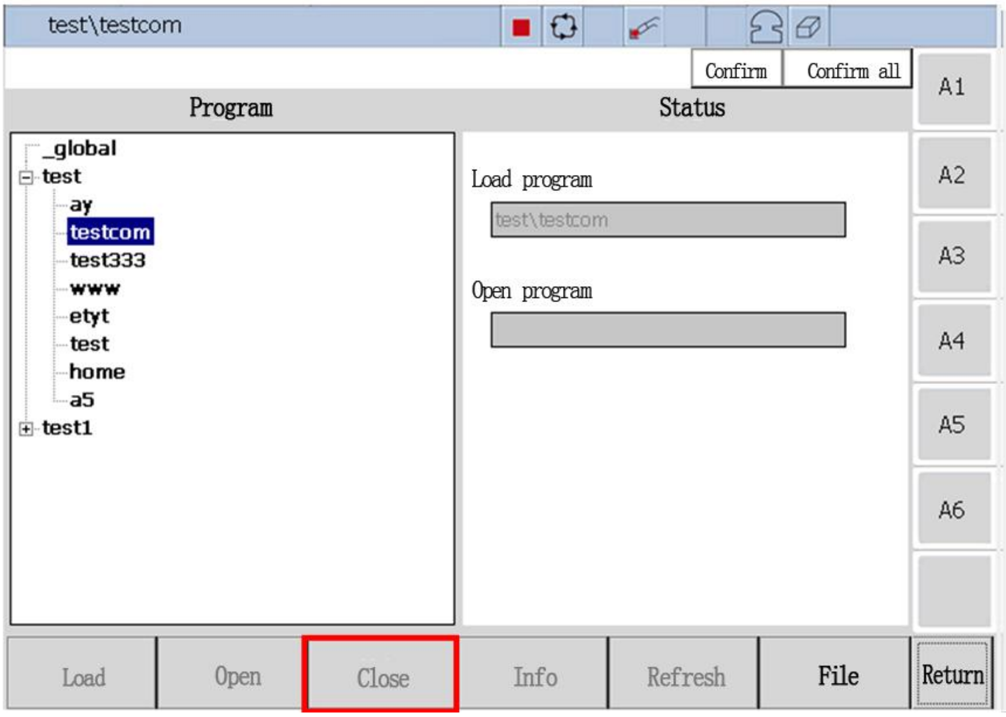


⚠**Attention :**

"Open Program" allows only the program opening and viewing, but cannot run the program. If you run such program, the program shall be loaded.

3.7.4 Close Program

First, you need to select the opened programs in the project directory tree, and then click "Close" to close the program. As the figure shows


test\testcom

Confirm | Confirm all

Program | Status

- _global
- test
  - ay
  - testcom
  - test333
  - www
  - etyt
  - test
  - home
  - a5
- test1

Load program: test\testcom

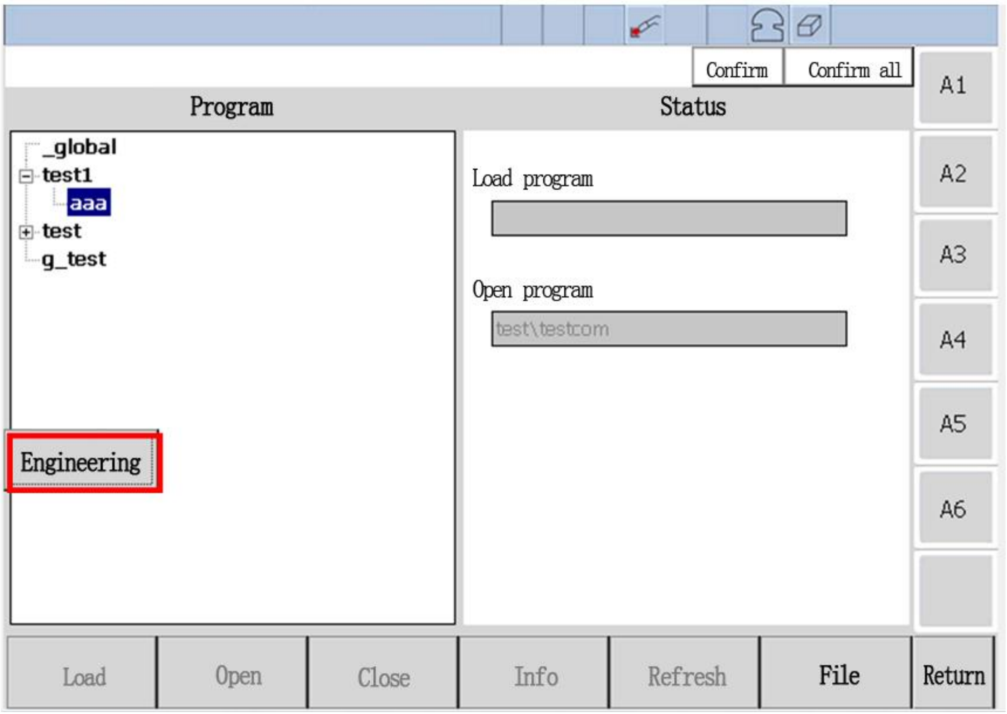
Open program:

A1 A2 A3 A4 A5 A6

Load | Open | Close | Info | Refresh | File | Return

3.7.5 View Current Program

- Press the ☰ button at the left of demonstrator, as shown in figure,


Confirm | Confirm all

Program | Status

- _global
- test1
  - aaa
- test
- g_test

Engineering

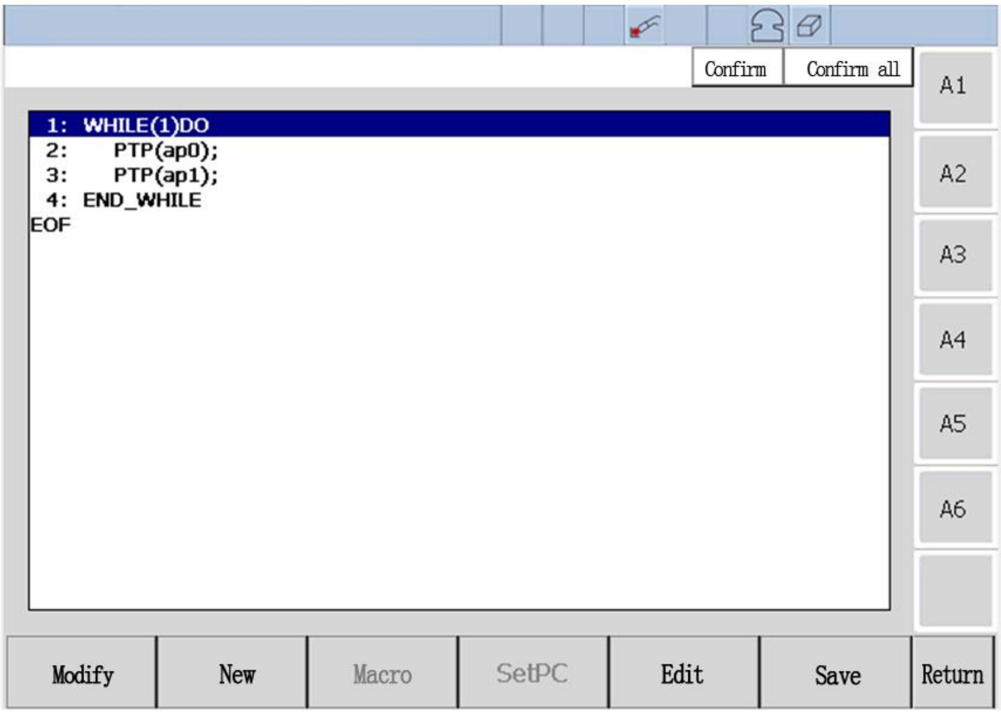
Load program:

Open program: test\testcom

A1 A2 A3 A4 A5 A6

Load | Open | Close | Info | Refresh | File | Return

- Upon clicking the "Program", you can quickly access to the program interface that has been recently opened or loaded; as shown in the figure below.

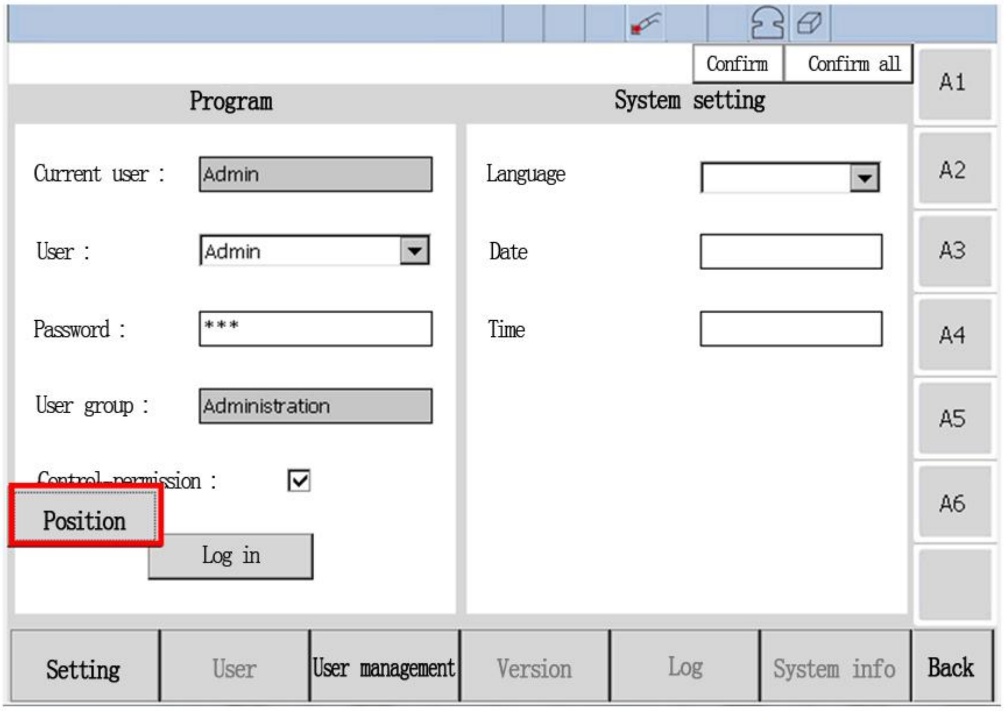


```
WHILE(1)DO
    PTP(ap0);
    PTP(ap1);
END_WHILE
EOF
```

3.8 Position

3.8.1 Position display

Press the button at the left of demonstrator, as shown in the figure below.



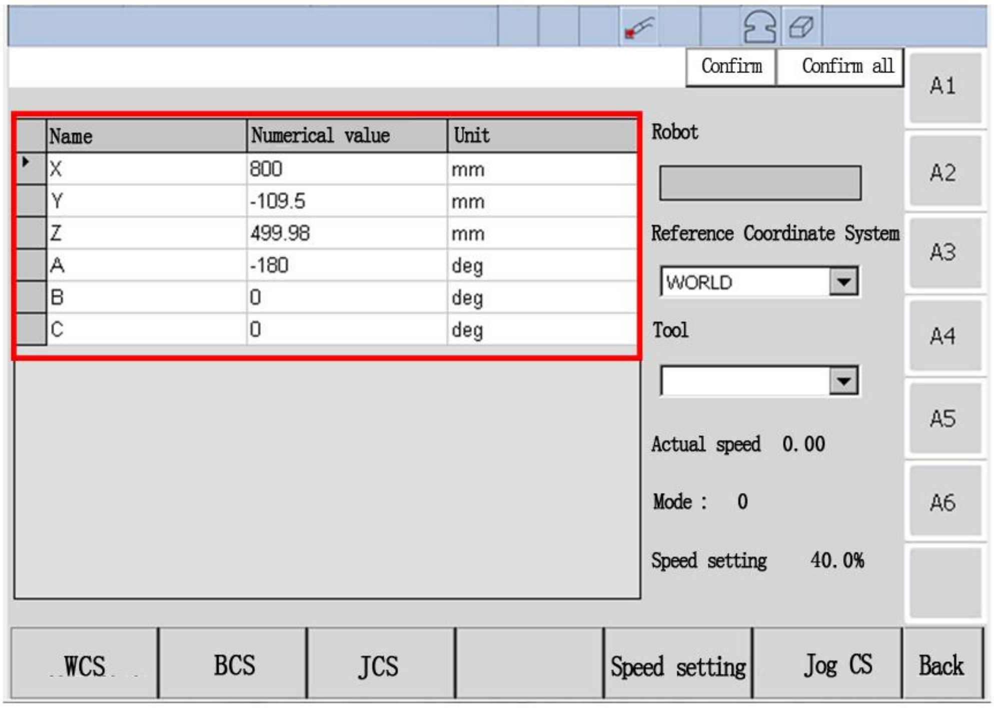
Program

Current user :	Admin
User :	Admin
Password :	***
User group :	Administration
Control permission :	☑

System setting

Language	
Date	
Time	

Click the "Position" and you can get the position information in different coordinate systems. TCP position display can be switched between two spaces, the joint space and Cartesian space, as shown in the figures.



Name	Numerical value	Unit
X	800	mm
Y	-109.5	mm
Z	499.98	mm
A	-180	deg
B	0	deg
C	0	deg

Robot

Reference Coordinate System WORLD

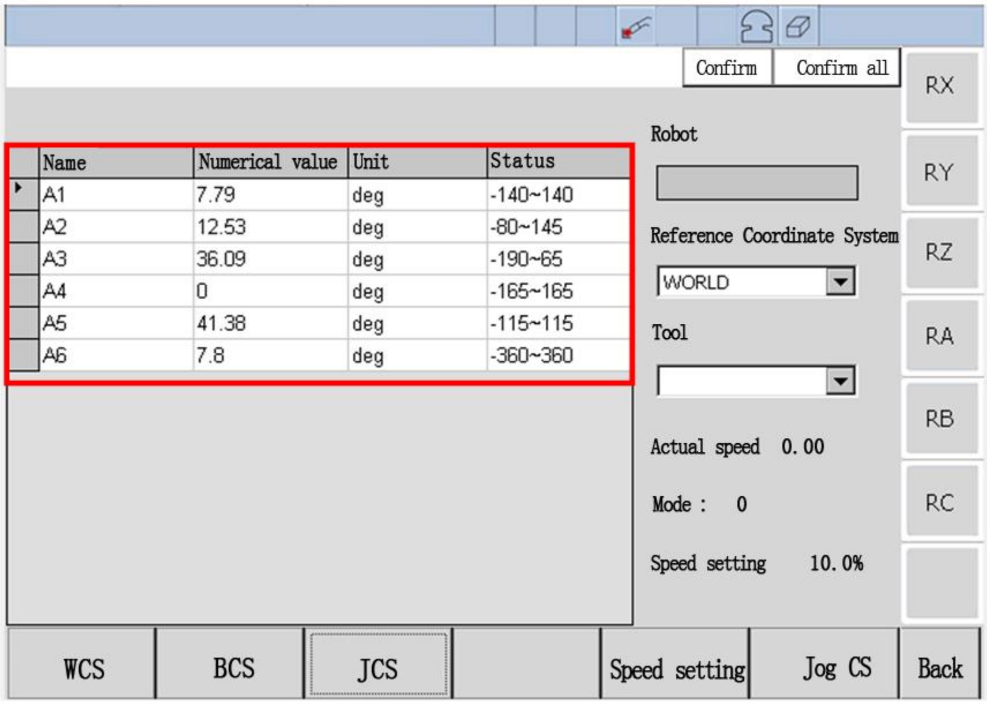
Tool

Actual speed 0.00

Mode : 0

Speed setting 40.0%

Cartesian position display of robot



Name	Numerical value	Unit	Status
A1	7.79	deg	-140~140
A2	12.53	deg	-80~145
A3	36.09	deg	-190~65
A4	0	deg	-165~165
A5	41.38	deg	-115~115
A6	7.8	deg	-360~360

Robot

Reference Coordinate System WORLD

Tool

Actual speed 0.00

Mode : 0

Speed setting 10.0%

Joint space position display of robot

⚠**Attention:**

① **Reference coordinate system switching of TCP position: There're three**

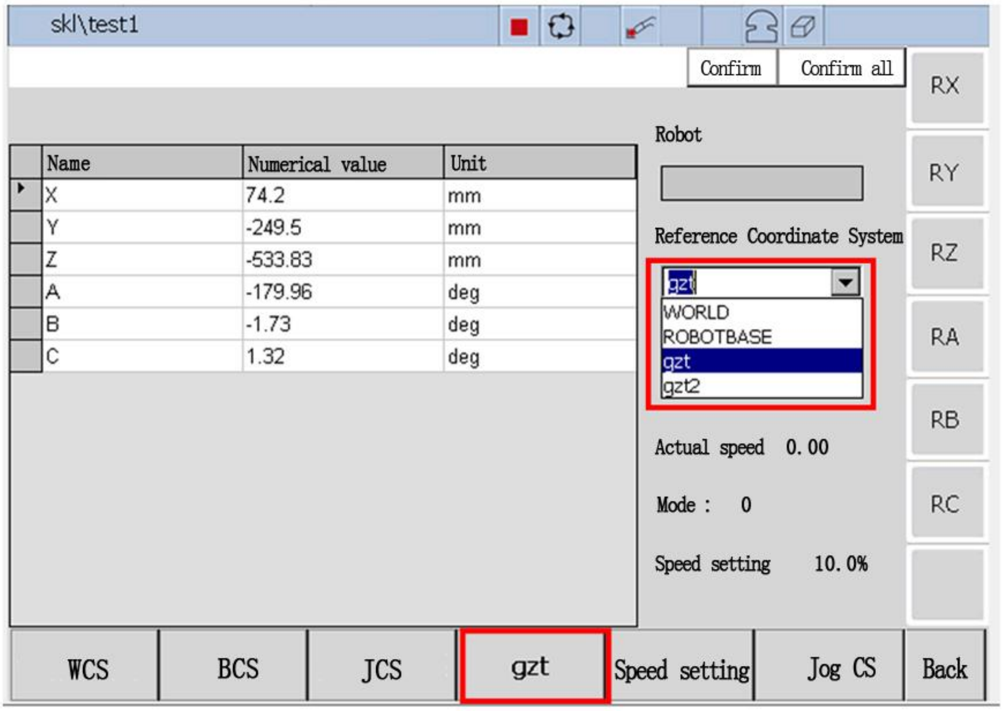
buttons located at the lower of interface, they are "World Coordinate System", "Base Coordinate System" and "Joint Coordinate System". Only click on the "Joint Coordinate System" button, then it will switch to joint space position. If you click on the "World Coordinate System" and "Base Coordinate System" button, it will switch to the Cartesian space position.

② **The "Reference Coordinate System" drop-down box in the right of the interface is used to customize the reference coordinate system selection. When the custom coordinate system as "gzt" is selected in the drop-down box, the button of the corresponding custom reference coordinate system in the lower of interface is displayed as "gzt ", which means that if the additional axis is configured, there will be additional axis coordinate system option here.**

③ **TCP is displayed in three columns in Cartesian space, respectively the name, value, units; and the names are indicated as X, Y, Z, A, B and C; the numerical display accurate to two decimal places; the unit is mm (millimeters, corresponding to the X, Y, Z) and deg (degrees, corresponding to A, B, C).**

④ **TCP is displayed in three columns in the joint space position, they are name, value, unit and status, respectively; and the names can be A1, A2, A3, A4, A5 and A6; the numerical display accurate to two decimal places; the unit is deg (degrees); the status display shows the joint limit, measured in degrees.**

⑤ **If the robot is configured with additional axes, A7, A8, A9 ... shall be added to the above figure showing the additional axis of robot, and the robot can be equipped with up to six additional axes.**



skl\test1

Name	Numerical value	Unit
X	74.2	mm
Y	-249.5	mm
Z	-533.83	mm
A	-179.96	deg
B	-1.73	deg
C	1.32	deg

Robot

Reference Coordinate System: gzt

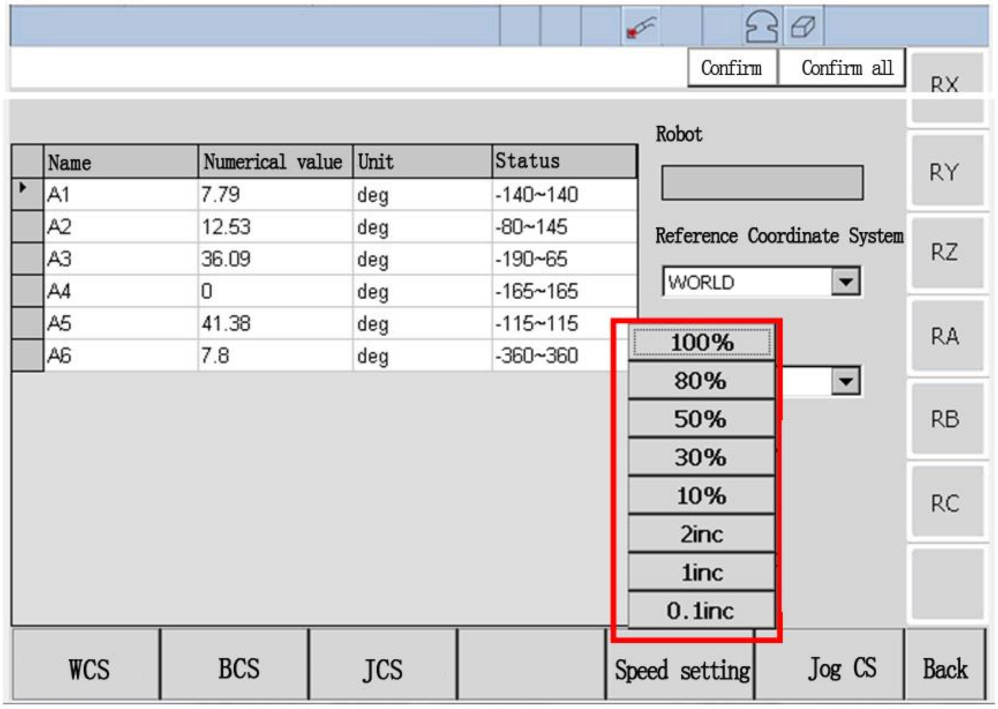
Actual speed 0.00

Mode : 0

Speed setting 10.0%

3.8.2 Velocity Settings

Click "Velocity Settings", and you can adjust the running velocity of the robot. The velocity settings are divided into 8 levels: 100%, 80%, 50%, 30%, 10%, 2inc, 1inc, 0.1inc. After selecting an appropriate velocity here (e.g. 10%), the velocity value display in status bar will changes accordingly. as the figure shows.



Name	Numerical value	Unit	Status
A1	7.79	deg	-140~140
A2	12.53	deg	-80~145
A3	36.09	deg	-190~65
A4	0	deg	-165~165
A5	41.38	deg	-115~115
A6	7.8	deg	-360~360

Robot

Reference Coordinate System: WORLD

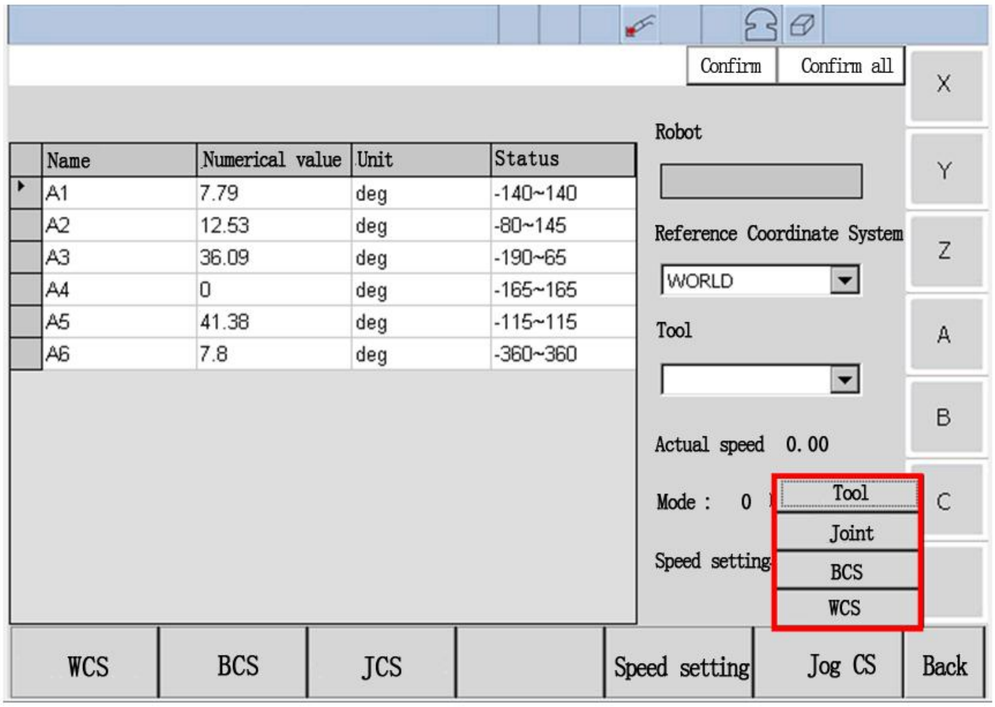
⚠**Attention:**

① **Velocity setting can also be done by pressing the v +, v- buttons on the panel.**

② **The jog type corresponding to the velocity ending in "inc" is the incremental jog, and others are the continuous jog.**

3.8.3 Settings of moving coordinate system

Click on the "Jog Coordinate System" button, and the settings dialog of jog reference coordinate system will pop up, respectively corresponding to the tools, joints, base coordinate system and world coordinate system. As the figure below shows:



Name	Numerical value	Unit	Status
A1	7.79	deg	-140~140
A2	12.53	deg	-80~145
A3	36.09	deg	-190~65
A4	0	deg	-165~165
A5	41.38	deg	-115~115
A6	7.8	deg	-360~360

Robot

Reference Coordinate System WORLD

Tool

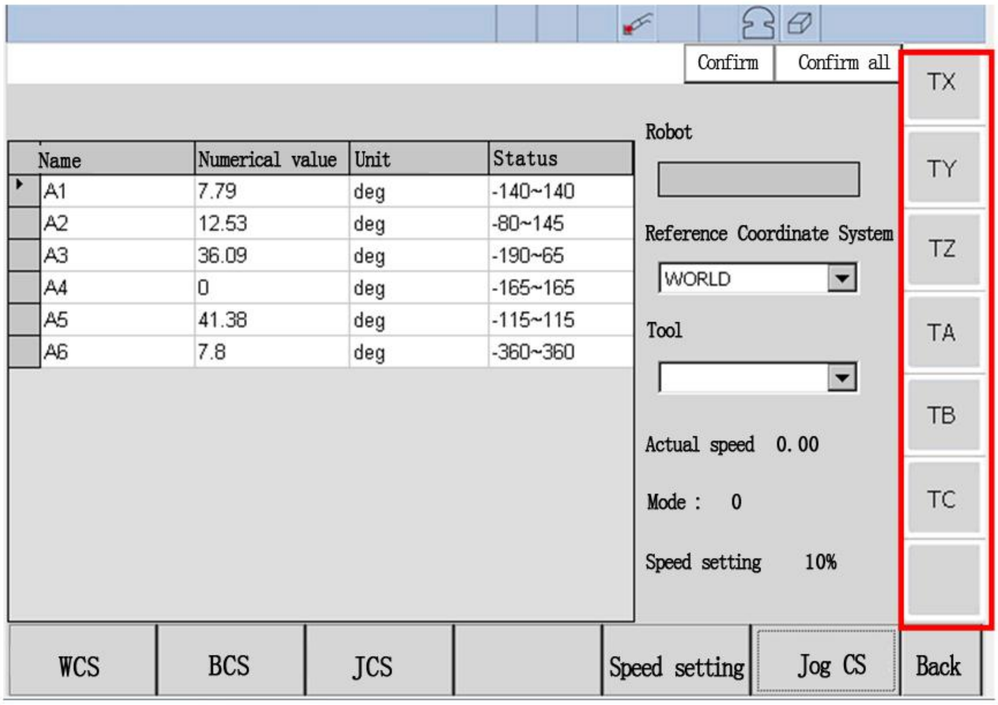
Actual speed 0. 00

Mode : 0

Speed setting

⚠ **Attention:**

① **After the jog reference coordinate system is set to "Tools", the right status bar of interface displays as TX, TY, TZ, TA, TB, TC, as shown in the figure;**



Name	Numerical value	Unit	Status
A1	7.79	deg	-140~140
A2	12.53	deg	-80~145
A3	36.09	deg	-190~65
A4	0	deg	-165~165
A5	41.38	deg	-115~115
A6	7.8	deg	-360~360

Robot

Reference Coordinate System: WORLD

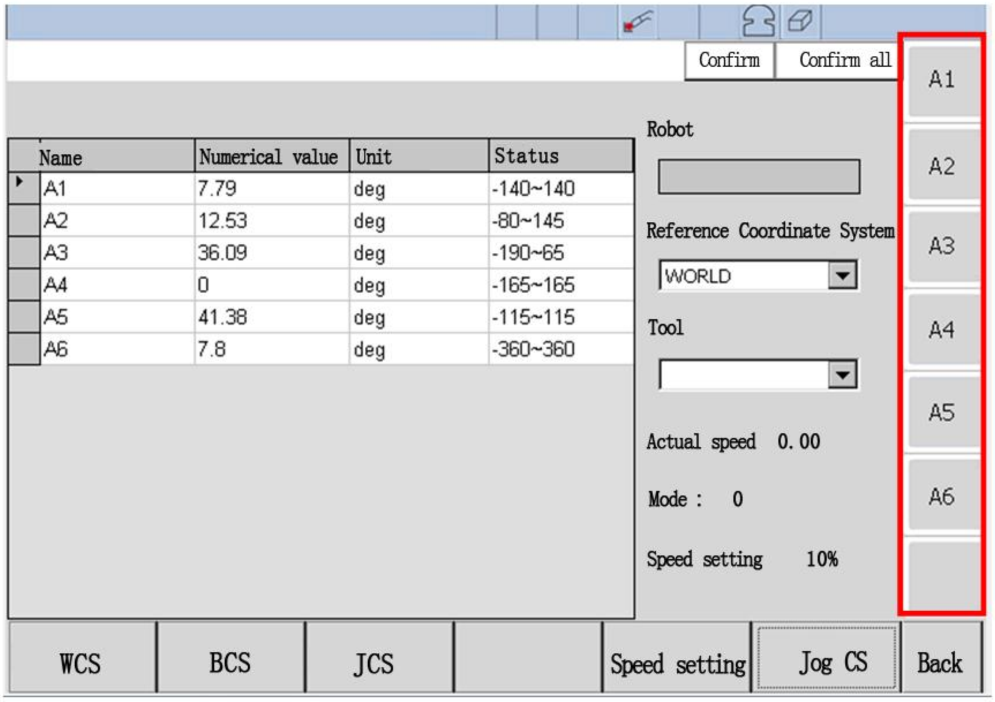
Tool

Actual speed 0.00

Mode : 0

Speed setting 10%

② **Setting the jog reference coordinate system to "Joint", the right status bar of interface displays as A1, A2, A3, A4, A5, A6, as shown in the figure;**



Name	Numerical value	Unit	Status
A1	7.79	deg	-140~140
A2	12.53	deg	-80~145
A3	36.09	deg	-190~65
A4	0	deg	-165~165
A5	41.38	deg	-115~115
A6	7.8	deg	-360~360

Robot

Reference Coordinate System: WORLD

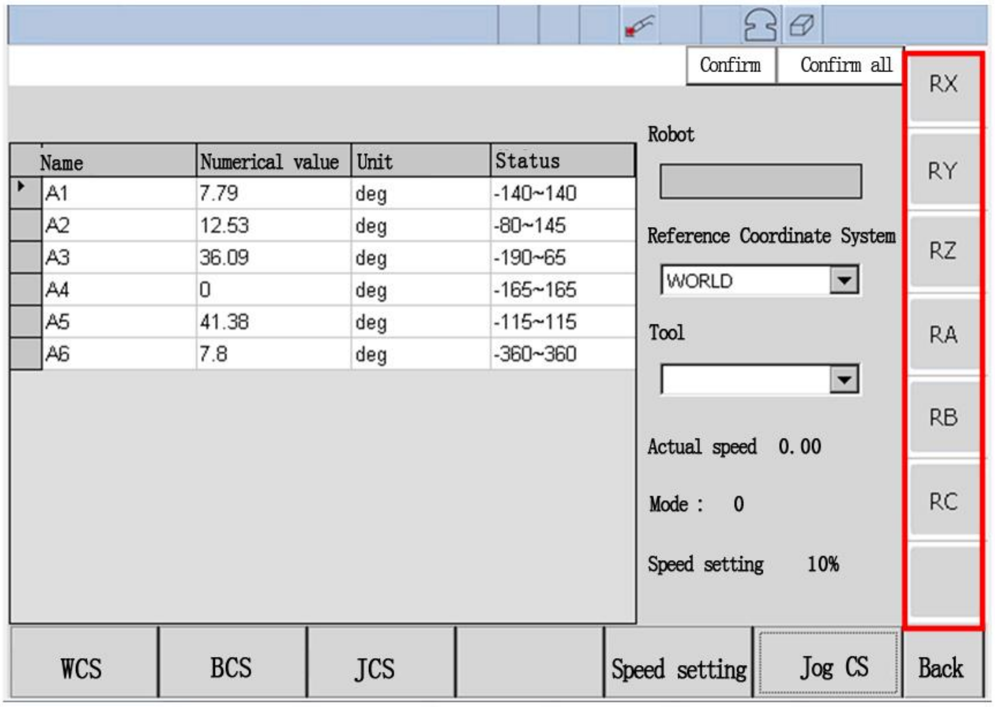
Tool

Actual speed 0.00

Mode : 0

Speed setting 10%

③ **Setting the jog reference coordinate system to the "Base Coordinate System" or custom coordinate system, and the right status bar of interface displays as RX, RY, RZ, RA, RB, RC, as shown in the figure below;**



Name	Numerical value	Unit	Status
A1	7.79	deg	-140~140
A2	12.53	deg	-80~145
A3	36.09	deg	-190~65
A4	0	deg	-165~165
A5	41.38	deg	-115~115
A6	7.8	deg	-360~360

Robot

Reference Coordinate System: WORLD

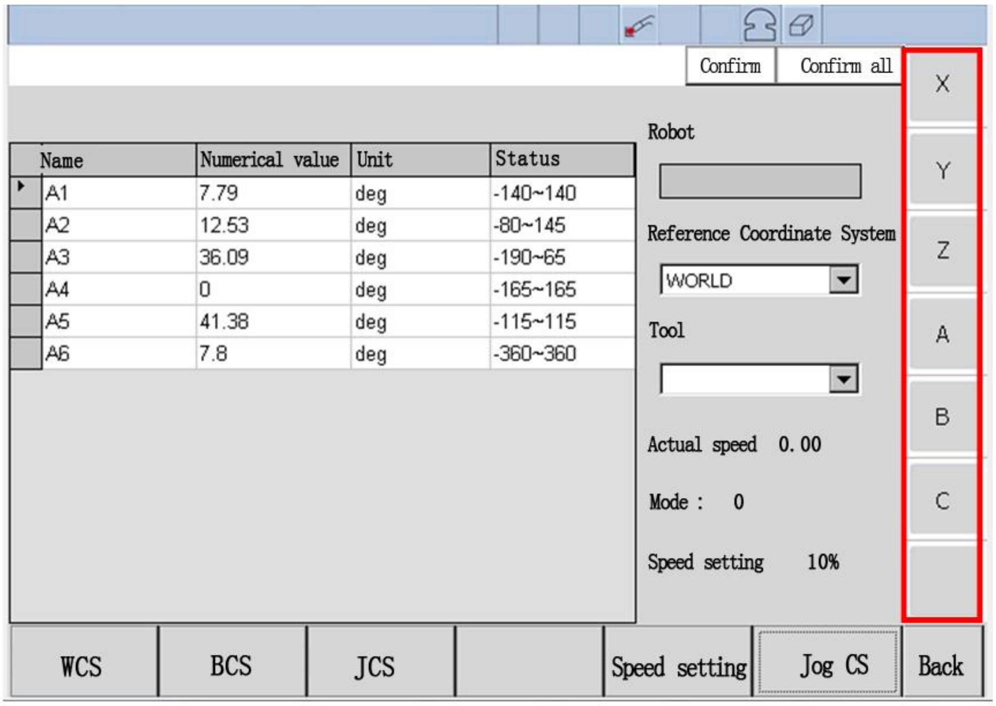
Tool:

Actual speed 0.00

Mode : 0

Speed setting 10%

④ **Setting the jog reference coordinate system to "World Coordinate System", and the right status bar of interface displays as X, Y, Z, A, B, C, as shown in the figure below.**



Name	Numerical value	Unit	Status
A1	7.79	deg	-140~140
A2	12.53	deg	-80~145
A3	36.09	deg	-190~65
A4	0	deg	-165~165
A5	41.38	deg	-115~115
A6	7.8	deg	-360~360

Robot

Reference Coordinate System: WORLD

Tool:

Actual speed 0.00

Mode : 0

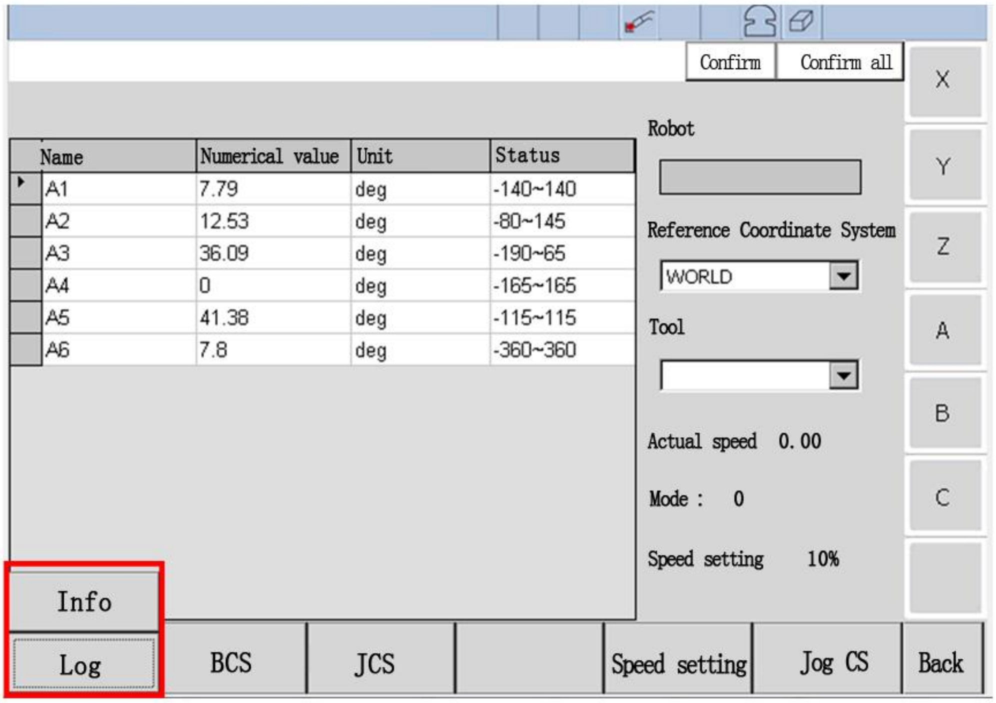
Speed setting 10%

⑤ **If there's additional axis configured for the robot, click on the 2nd and the right status bar of the interface shows the additional information, such as axis A7, A8, A9 ..., and the number as displayed is connected with the**

number of additional axle that are current configured.

3.9 Message

- Press the ⚠ button at left of the demonstrator, as shown in the figure.



Name	Numerical value	Unit	Status
A1	7.79	deg	-140~140
A2	12.53	deg	-80~145
A3	36.09	deg	-190~65
A4	0	deg	-165~165
A5	41.38	deg	-115~115
A6	7.8	deg	-360~360

Robot

Reference Coordinate System WORLD

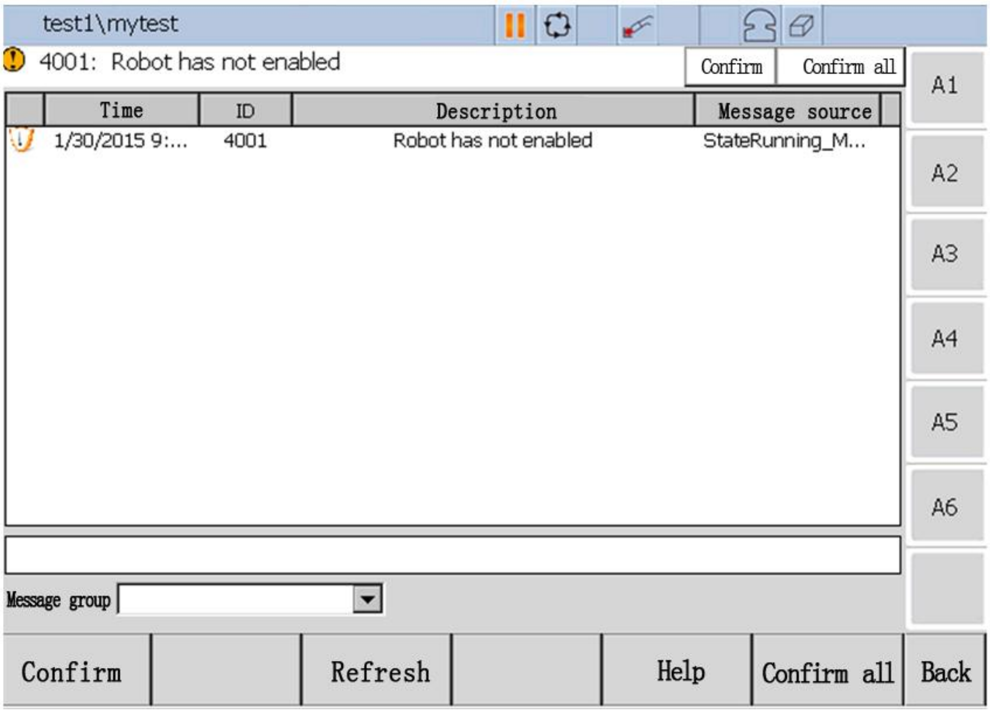
Tool

Actual speed 0.00

Mode : 0

Speed setting 10%

- Click "Info" and there displays the error alarm message of current robot failure. As the figure below shows:



test1\mytest

4001: Robot has not enabled

Time	ID	Description	Message source
1/30/2015 9:...	4001	Robot has not enabled	StateRunning_M...

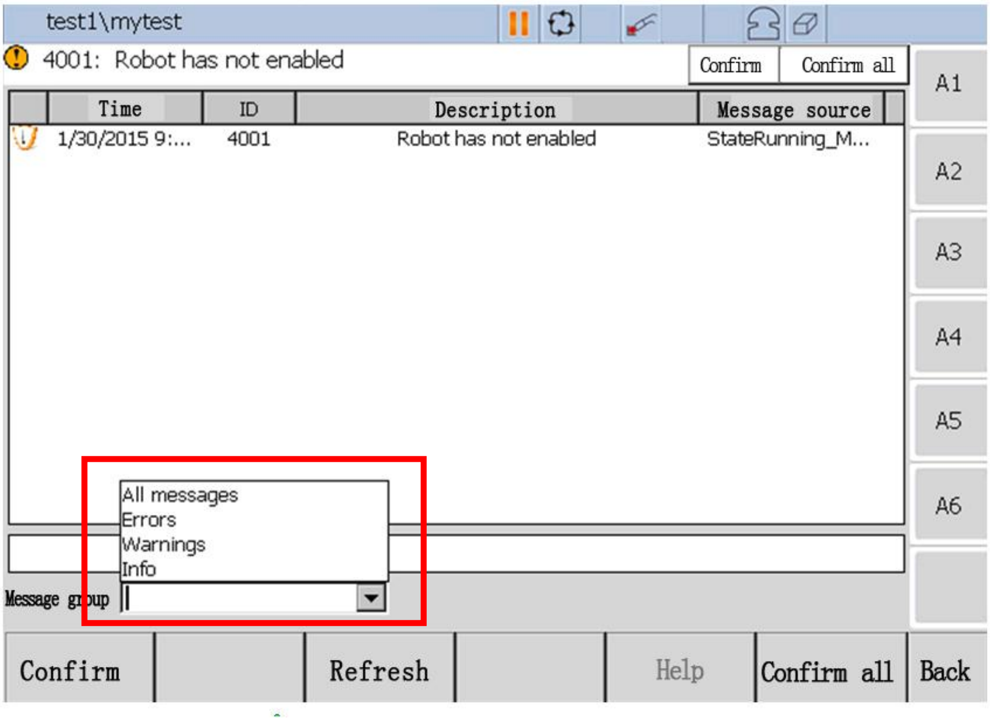
Message group

⚠Attention:

① Message data display: The occurrence time, ID number, description, as well as sources of message;

② Complete message display: Select a message, and the complete description of such message is displayed in the control located at lower of the interface;

③ Message screening display: Select the message type you want to display in the "Message Block " drop-down box at lower of the interface, in which All messages, Errors, Warnings, Info correspond to the message types as follows: all messages, error messages, alarm message and general message. As shown in the figure below:



test1\mymytest

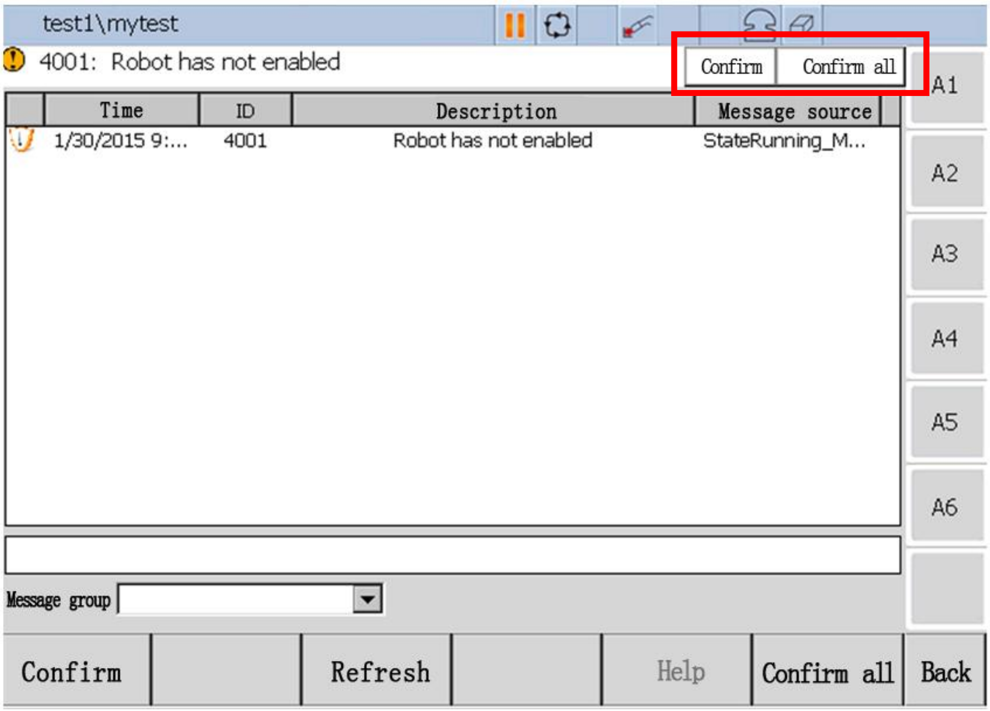
4001: Robot has not enabled

Time	ID	Description	Message source
1/30/2015 9:...	4001	Robot has not enabled	StateRunning_M...

④ **Message confirmation:**

Single Confirmation: First, select a message you want to confirm, and then click on the "OK" button to confirm the single piece of message, and such selected message will be disappeared in the interface.

Overall Confirmation: Click "Confirm all" to confirm all the messages, the all the messages will disappear from the interface. As shown in figure below:



4001: Robot has not enabled

Time	ID	Description	Message source
1/30/2015 9:...	4001	Robot has not enabled	StateRunning_M...

Chapter 3 Advanced Programming

1 Jog operation of robot

1.1 Continuous jog

The continuous jog means that when hold down the jog button "+" and "-", the robot will continuously moving, and will stop upon releasing the button "+" and "-".

For the continuous jog, first you need to set the velocity (the velocity shall not end in "inc" after setting), and then determine whether the status bar on the right side displays the joint jog or Cartesian jog, rather than the jog icon GO of absolute position.

The reference coordinate system of continuous jog includes the joint and Cartesian reference coordinate systems (namely the "World Coordinate System", "Base Coordinate System" and custom reference coordinate system). The continuous jog reference coordinate system can be set in the pop-up box of “Jog” button in the robot interface; velocity setting can be done in three ways: First setting in the program, the second is setting by the "Velocity Setting" on the robot position interface, and the third is setting via the "V +" "V-" buttons of the demonstrator.

1.2 Incremental jog

In the incremental jog mode, the running route length of the robot is independent of the button "+", "-" hold-down time, but the number of pressing the "+", "-" buttons.

For the incremental jog, first you need to set the velocity (the velocity shall not end in "inc" after setting), and then determine whether the status bar on the right side displays the joint jog or Cartesian jog, rather than the jog icon GO of absolute position.

Similar with the continuous jog, the reference coordinate system of incremental jog includes the joint and Cartesian reference coordinate systems (namely the "World Coordinate System", "Base Coordinate System" and custom reference coordinate system). The incremental jog reference coordinate system can be set in the pop-up box of “Jog” button in the robot interface; velocity setting can be done in three ways: First setting in the program, the second is setting by the "Velocity Setting" on the robot position interface, and the third is setting via the "V +" "V-" buttons of the demonstrator.

1.3 Jog of absolute position (not being enabled temporarily)

The absolute position jog is to enable the robot to move to a point in space in some ways ("PTP" or "Lin").

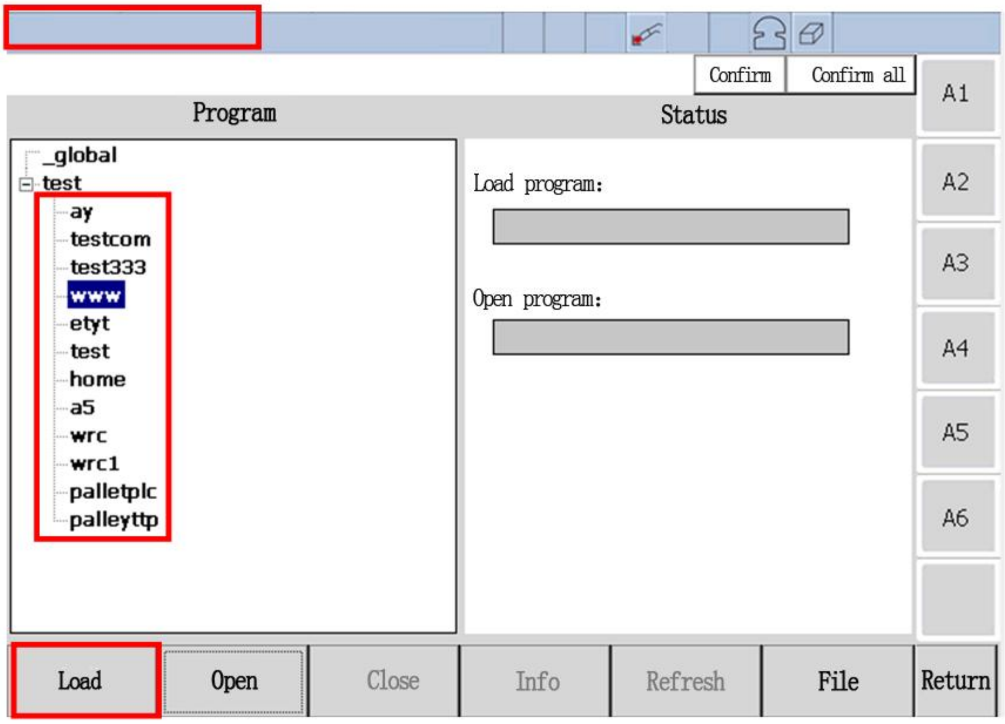
For the absolute position jog, first you need to switch the robot to the absolute position jog mode by pressing "Activate" button in the corresponding interface, and the icon GO will appear on the right status bar at this time. For the other specific settings, please refer to the instructions on the absolute position jog interface.

2 Run program

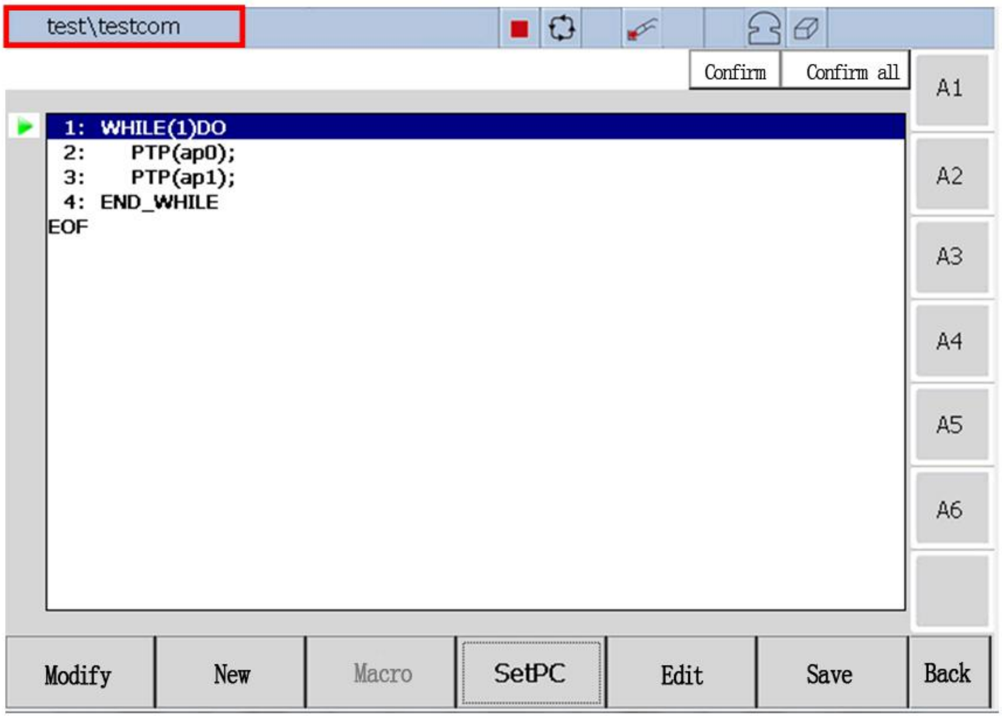
2.1 Manual mode

Under manual mode T2:

- Open the project screen and select the program (it is available if there’s no loaded program in the red box on the upper side). As the figure shows

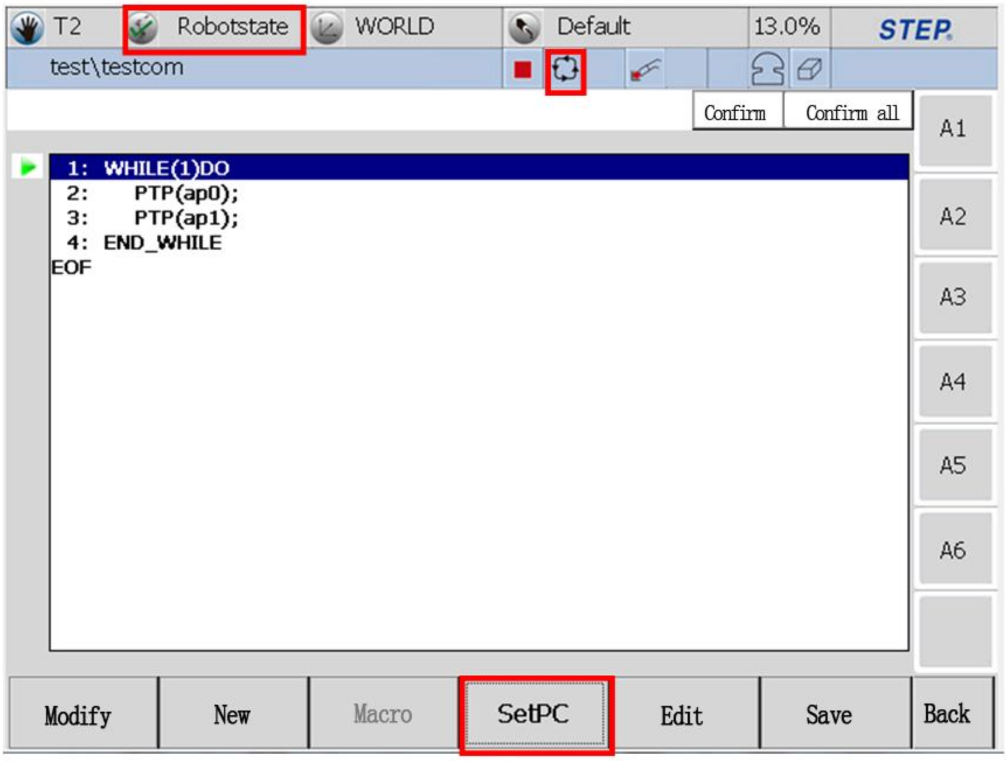


Click the “Load”, as the figure shows.



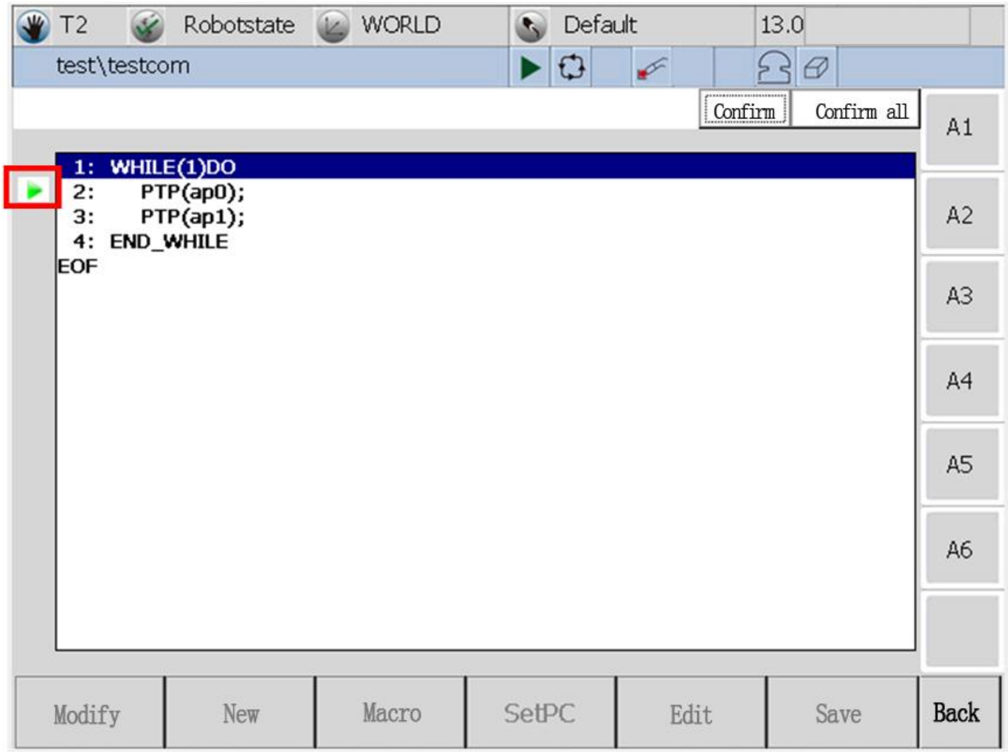
```
WHILE(1)DO
    PTP(ap0);
    PTP(ap1);
END_WHILE
EOF
```

- Select the statement line you want to run, click SetPC, and the program will jump to the line you want to run. Enable on the three-position switch to select the way of movement, and hold down the "Start" button located at the right. As the figure shows



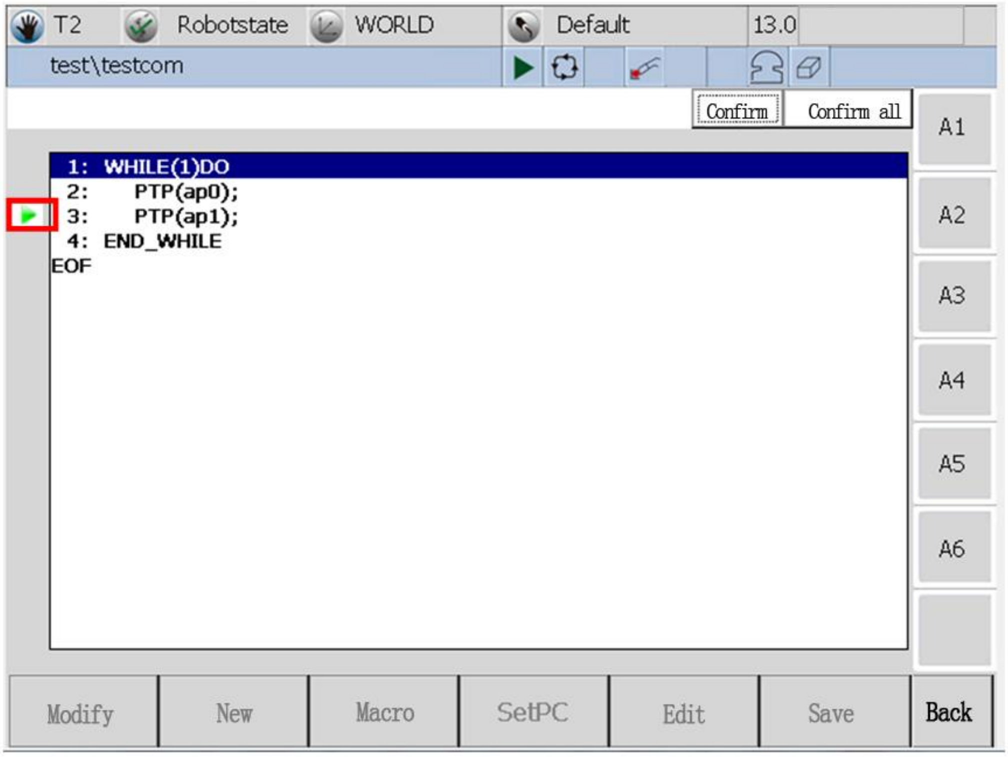
```
WHILE(1)DO
    PTP(ap0);
    PTP(ap1);
END_WHILE
EOF
```

- Run to the second step, as shown in the figure.



```
 1:  WHILE(1)DO
 2:      PTP(ap0);
 3:      PTP(ap1);
 4:  END_WHILE
EOF
```

- Run to the third step, as shown in the figure.



```
 1:  WHILE(1)DO
 2:      PTP(ap0);
 3:      PTP(ap1);
 4:  END_WHILE
EOF
```

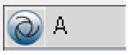
- The program will repeat the second and third steps.

➢ Notes:

① Press and hold the three-position switch in manual mode, and the robot will suspend to move if you release the "Start" button.

② In manual mode, if you press and hold the "Start" button and release the three-position switch, the robot will give out the error alarm.

2.2 Auto mode

Switch the key switch to external auto  in the auto mode, and press down the Mot button to enable, click on "Start" and run the program. Click the "Stop" button to pause the program.

3 Variable types and programming instructions

3.1 Introduction of variable types

3.1.1 Basic types

Type	Meaning	Value range	Corresponded C++ type
BOOL	Boolean type	TRUE / FALSE(1/0)	Bool
INT	Integer	-2^15 ~ 2^15-1	Short
UINT	Unsigned integer	0 ~ 2^16-1	Unsigned short
LINT	Long integer	-2^31 ~ 2^31-1	Long int (the same with the int length)
ULINT	Unsigned long integer	0 ~ 2^32-1	Unsigned long int
FLOAT(REAL)	Single-precision floating point		Float
DOUBLE(LREAL)	Double-precision floating point		Double
STRING	String type		String / char []

Usage of common types:

3.1.1.1 **BOOL**

For example, BOOL b1 := TRUE

3.1.1.2 **INT**

For example, INT int1 := -10

3.1.1.3 **UINT**

For example, UINT uint1 := 100

3.1.1.4 **LINT**

For example, LINT lint1 := -1000

3.1.1.5 **ULINT(DWORD)**

For example, ULINT ulint1 := 1000

3.1.2 Type of position data

The complex data types are defined in reference to the structure in the ROBOT data structure type that has already been defined. They will be provided to users as the options when the users creating variables, and user-defined data types are not allowed.

Data type	Meaning	Corresponding internal type
AXISPOS	Joint + Additional axis position	JointsPos + AuxPos
CARTPOS	Space + Additional axis position	CartPos + AuxPos
AXISPOSEXT	External joint position	Index of JointsAuxEXT
CARTPOSEXT	External space position	Index of CartAuxEXT
ROBOTAXISPOS	Robot joint position	JointsPos
ROBOTCARTPOS	Robot Cartesian position	CartPos
ROBOTAXISPOSEXT	External robot joint position	Index of JointsEXT

ROBOTCARTPOSEXT	External robot Cartesian position	Index of CartEXT
AXISDIST	Relative joint + Additional axis position	RcDistance
CARTDIST	Relative Cartesian + Additional axis position	RcDistance
CARTPOSEE	Send Cartesian coordinate system data to PLC	

Such types can be AXISPOS, CARTPOS, AXISPOSEXT, CARTPOSEXT, ROBOTAXISPOS, AUXAXISPOS, ROBOTCARTPOS, ROBOTAXISPOSEXT, AUXAXISPOSEXT and ROBOTCARTPOSEXT

3.1.2.1 **AXISPOS**

Joint + additional axis position, for which 6 joints + additional axes position at most, the type of each variable is DOUBLE, in degrees (°).

For example, AXISPOS ap1 := {13.22, 45, 21, 22.6, 40.7, 11, 2.4 ,30.5 , 40.11, 61.3, 11.4, 23.1}

3.1.2.2 **CARTPOS**

The spatial position under the Cartesian coordinate system, with additional axes up to 6. Variable is consisted of the X, Y, Z, A, B, C, Aux1 ~ 6 and Mode, wherein X ~ Aux6 is DOUBLE type, and Mode is UINT type.

For example, CARTPOS cp1 := {13.22, 45, 21, 22.6, 40.7, 11, 2.4 ,30.5 , 40.11, 61.3, 11.4, 23.13, 1}

3.1.2.3 **AXISPOSEXT**

External joint + additional axis position, in which only the UINT type value can be assigned. It represents an index of JointsAuxEXT [32].

For example, AXISPOSEXT ape1 := 5

3.1.2.4 **CARTPOSEXT**

External space+ additional axis position, in which only the UINT type value can be assigned. It represents an index of CartAuxEXT [32].

For example, CARTPOSEXT cpe1 := 5

3.1.2.5 **ROBOTAXISPOS**

Robot joint position, up to six, and the type of each variable is DOUBLE, measured in degrees (°).

For example, ROBOTAXISPOS rap1 := {10.1 , 10.1 , 10.1 , 10.1 , 10.1 , 10.1}

For example, AUXAXISPOS raxp1 := {10.1 , 10.1 , 10.1 , 10.1 , 10.1 , 10.1}

3.1.2.6 **ROBOTCARTPOS**

It is the spatial position under the Cartesian coordinate system. Variable is consisted of the X, Y, Z, A, B, C and Mode, wherein X~C is DOUBLE type, and Mode is UINT type.

For example, ROBOTCARTPOS   rcp1 := {10.1 , 10.1 , 10.1 , 10.1 , 10.1 , 10.1 , 1}

3.1.2.7 **ROBOTAXISPOSEXT**

External joint position, in which only the UINT type value can be assigned, which indicates an index of JointsEXT [32].

For example, ROBOTAXISPOSEXT rape1 := 5

3.1.2.8 **ROBOTCARTPOSEXT**

External space position, in which only the UINT type value can be assigned, which indicates an index of CartEXT [32].

For example, ROBOTCARTPOSEXT rcpe1 := 5

3.1.2.9 **AXISDIST**

Relative joint+ additional axis position, for which, there’re 6 joints+6 additional axes at most, and the type of each variable is DOUBLE, measured in degrees (°).

For example, AXISDIST ad1 := {10.1 , 10.1 , 10.1 , 10.1 , 10.1 , 10.1, 10.1, 10.1, 10.1, 10.1, 10.1, 10.1}

3.1.2.10 **CARTDIST**

The spatial position under the Cartesian coordinate system, with additional axes up to 6. Variable is consisted of the X, Y, Z, A, B, C, Aux1 ~ 6 and Mode, wherein X ~ Aux6 is DOUBLE type, and Mode is UINT type.

For example, CARTDIST cd1 := {10.1 , 10.1 , 10.1 , 10.1 , 10.1 , 10.1 , 10.1 , 10.1 , 10.1 , 10.1 , 10.1 , 10.1 , 1}

3.1.3 Coordinate system and tool data types

Data type	Meaning	Corresponded internal variables
CARTREF	Reference coordinate system	CartSys
CARTREFEXT	External reference coordinate system	Index of m_RefExtern, along with the name of reference coordinate system
CARTREFVAR	External tracking coordinate system	Index of m_RefVar, along with the name of reference coordinate system
CARTREFEXSYS	Additional axis coordinate system	Index of m_RefExSys
TOOL	Tools	Tool

3.1.3.1 **CARTREF**

The position of coordinate system against other reference coordinate systems, for which the variables consist of the X, Y, Z, A, B, C, as well as the name coordinate system that is used for reference, and X ~ C are DOUBLE type.

For example, CARTREF cr1 := {1.1 , 1.2 , 1.3 , 1.4 , 1.5 , 1.6, crs0}

3.1.3.2 **CARTREFEXT**

External coordinate system, assigning with a value of UNIT type and indicating an index of m_RefExtern [8]. There is also a value of STRING type that represents the reference coordinate system. The coordinate system used for reference is only the fixed coordinate system.

For example, CARTREFEXT cre1 := {5, crs0}

3.1.3.3 **CARTREFVAR**

External coordinate system, assigning with a value of UNIT type and indicating an index of m_RefVar [8]. There is also a value of STRING type that represents the reference coordinate system. The coordinate system used for reference is only the fixed coordinate system.

For example, CARTREFVAR crv1 := {5, crs0}

3.1.3.4 **CARTREFSYSEXSYS**

Additional axis coordinate system, assigning with a value of UNIT type and indicating an index of m_RefExSys [6].

For example, CARTREFEXSYS crex1 := 0

3.1.3.5 **TOOL**

The transformation matrix representing the tool. It consist of X, Y, Z, A, B and C, all of which are DOUBLE type.

For example, TOOL t1 := {1.1, 1.2, 1.3,1.4,1.5,1.6}

3.1.4 Dynamic and oval characteristics

Data type	Meaning	Corresponded internal variables
DYNAMIC	Dynamic characteristics	DynamicPerc
OVLREL	OVLREL	Overlap
OVLABS	Absolute magnitude of ovality	Overlap

3.1.4.1 **DYNAMIC**

It is composed by the position, rotation and joint velocity, acceleration, deceleration and jerk, totaling 12 quantities. It is based on the path type executed (for Cartesian or joint), and different amounts are used. 12 quantities are:

PathVel	DOUBLE	Position velocity (mm/s)
PathAcc	DOUBLE	Position acceleration (mm/s^2)
PathDec	DOUBLE	Position deceleration (mm/s^2)
PathJerk	DOUBLE	Position jerk (mm/s^3)
OriVel	DOUBLE	Rotation speed (deg/s)
OriAcc	DOUBLE	Rotation acceleration (deg/s^2)
OriDec	DOUBLE	Rotation deceleration

		(deg/s^2)
OriJerk	DOUBLE	Rotation jerk (deg/s^3)
JointVel	DOUBLE	Joint velocity (%)
JointAcc	DOUBLE	Joint acceleration (%)
JointDed	DOUBLE	Joint deceleration (%)
JointJerk	DOUBLE	Joint jerk (%)

For example, DYNAMIC dyn1 := {1000 , 2000 , 2000 , 5000 , 100 , 200 , 200 , 500 , 80 , 80 , 80 , 80}

3.1.4.2 **OVLREL**

It indicates the percentage of ovality, and is an amount of INT type, representing the percentum.

For example, OVERLAPREL ovl1 := 50

3.1.4.3 **OVLABS**

The variable type includes four members

PosDist	DOUBLE	TCP position oval distance (mm)
oriDist	DOUBLE	TCP posture oval angle (°)
linAxDist	DOUBLE	Oval distance of linear axis (mm)
rotAxDist	DOUBLE	Oval angle of rotation axis (°)

For example, OVERLAPABS ovl2 := {1.1, 1.2, 1.3, 1.4}

3.1.4.4 **ORITYPE**

It indicates the posture on the path, belonging to enumerated types. The value is only three, as shown in below:

eVAR: // Make quaternion posture interpolation based on the posture information as demonstrated

eCONST: // Keep the posture of starting position always eCIRCLE: // Keep the fixed posture with the round plane for circle interpolation

For example, ORITYPE oritype := eVAR

3.1.4.5 **RAMPTYPE**

Velocity planning types, consisting of three types: Trap, Shape and Ssine;

3.1.5 IO variable

It is composed of the digital input/output and analog input/output. Each type is equipped with two quantities: the port number and value.

Data type	Meaning	Corresponded internal type
DI	Digital input	DI
DO	Digital output	DO
AI	Analog input	AI[…]
AO	Analog output	AO[…]

3.1.5.1 **DI**

Digital input, composing of the Port and Value. Port represents the DI port corresponded and Value indicates the value read over the port. Port is a figure (0-31), and Value is BOOL type.

For example, DI di1 := {0 ,TRUE }

3.1.5.2 **DO**

Digital output, composing of the Port and Value. Port represents the DO port corresponded and Value indicates the value read over the port. Port is a figure (0-31), and Value is BOOL type.

For example, DO do1 := {0 , TRUE }

3.1.5.3 **AI**

Analog input, consisting of the Port and Value. Port represents the AI port corresponded and Value indicates the value read over the port. Port is a figure (0-31), and Value is INT type.

For example, AI ai1 := {0 , 52}

3.1.5.4 **AO**

Analog output, consisting of the Port and Value. Port represents the AO port corresponded and Value indicates the value read over the port. Port is a figure (0-31), and Value is INT type.

For example, AO ao1 := {0 , 11}

3.1.5.5 **BOOLEXT**

External (PLC) variable type. When it is required interacting with external bool variable, create and use the variable of such type as a parameter. This type of variable consists of the Port and Value, where Port is a figure (0-31), Value is bool type (the BOOL above). The defined data has the following format:

For example, BOOLEXT ext1 := {0 , TRUE}

3.1.5.6 **DINTEXT**

External (PLC) variable type. When it is required interacting with external long integer variable, create and use the variable of such type as a parameter. This type of variable consists of the Port and Value, where Port is a figure (0-31), Value is DINT type (the LINT above). The defined data has the following format:

For example, DINTEXT ext1 := {0 , 12}

3.1.5.7 **REALEXT**

External (PLC) variable type. When it is required interacting with external REAL variable, create and use the variable of such type as a parameter. This type of variable consists of the Port and Value, where Port is a figure (0-31), Value is REAL type (the FLOAT above). The defined data has the following format:

For example, REALEXT ext1 := {0 , 1.1}

3.1.5.8 **DWORDEXT**

External (PLC) variable type. When it is required interacting with external DWORD variable, create and use the variable of such type as a parameter. This type of variable consists of the Port and Value, where Port is a figure (0-31), Value is DWORD type (the ULINT above). The defined data has the following format:

For example, DWORDEXT ext1 := {0 , 222}.

3.2 Motion statement

Create and add in the previous line which you selected when editing the statement.

3.2.1 PTP

- PTP: Point-to-point movement, for which the robot reaches the designated position with the fastest way, and the middle path is uncertain. As the figure shows

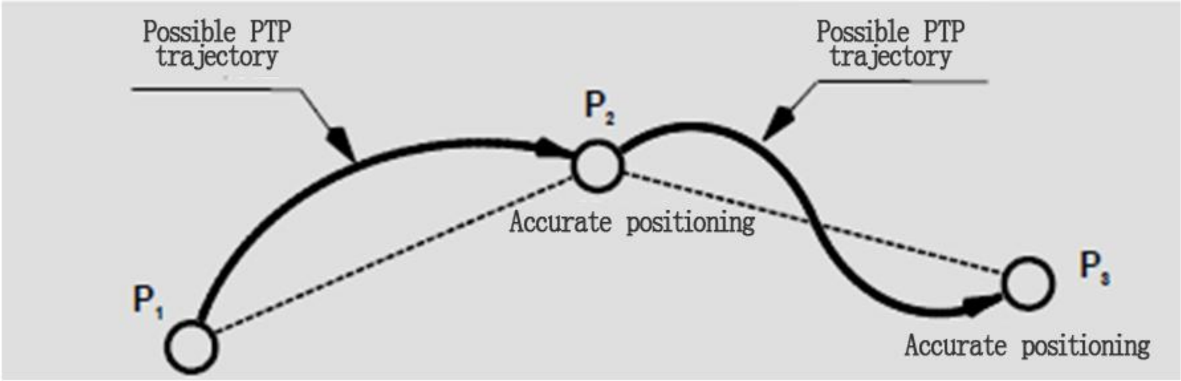
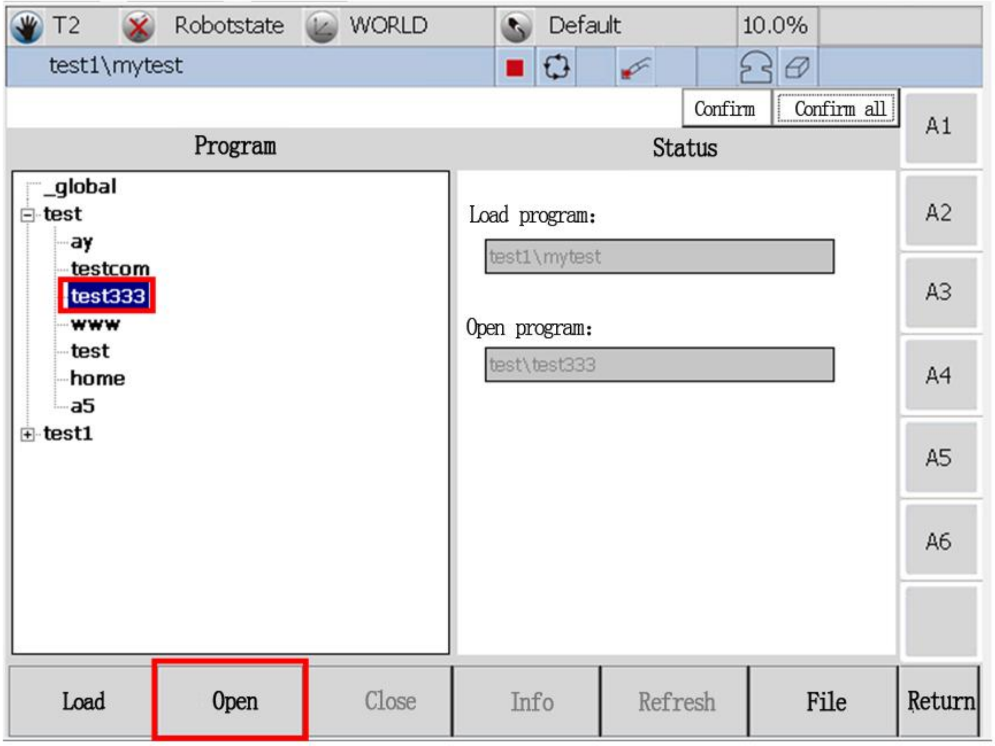


Figure 3-1 PTP schematic diagram

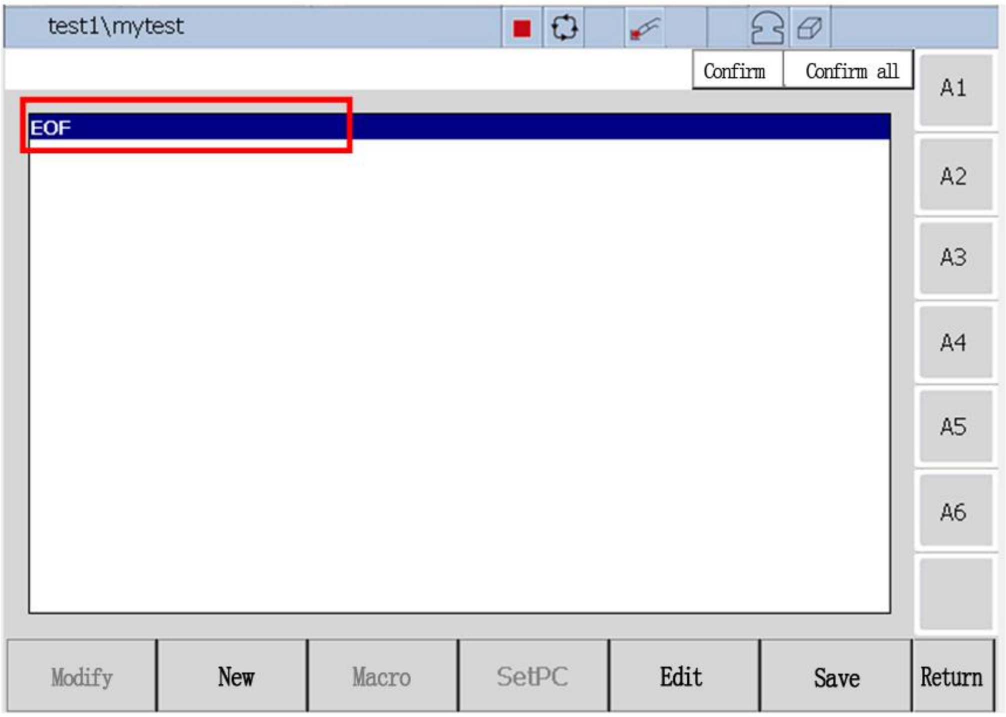
- Select a new program, and open it. As the figure shows



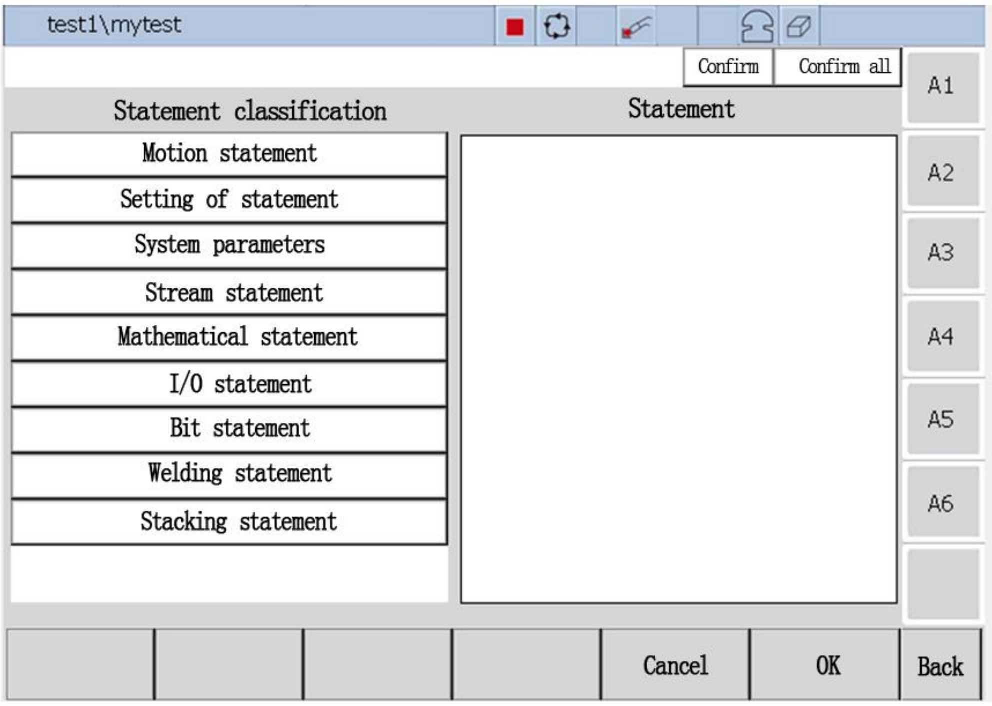
Load program: test1\mytest

Open program: test\test333

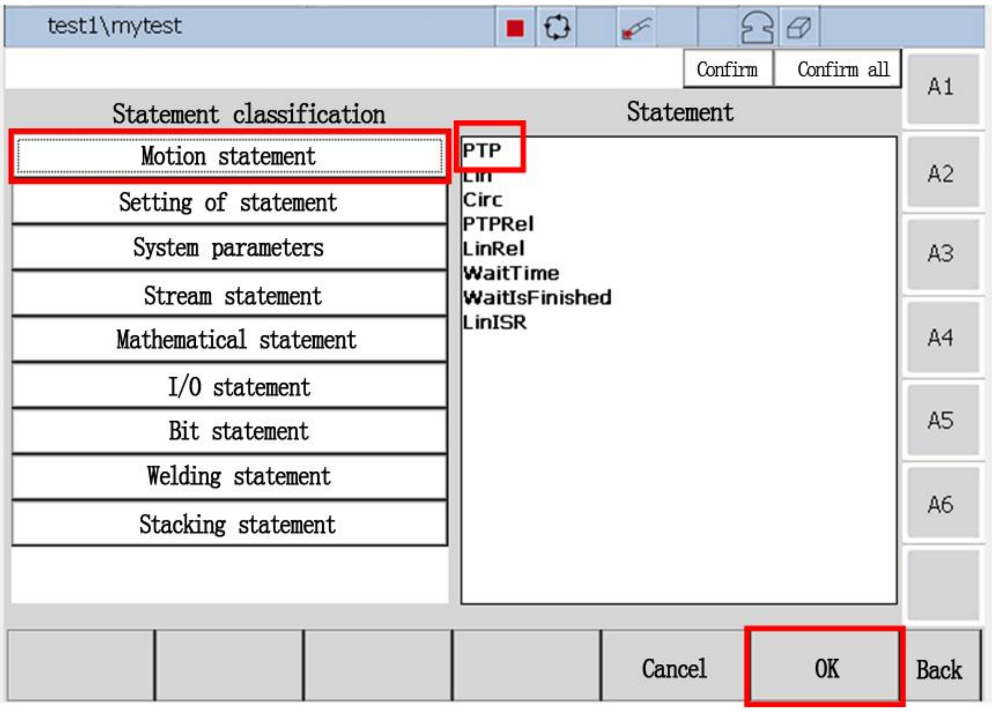
- After opening, EOF is terminator. As the figure shows



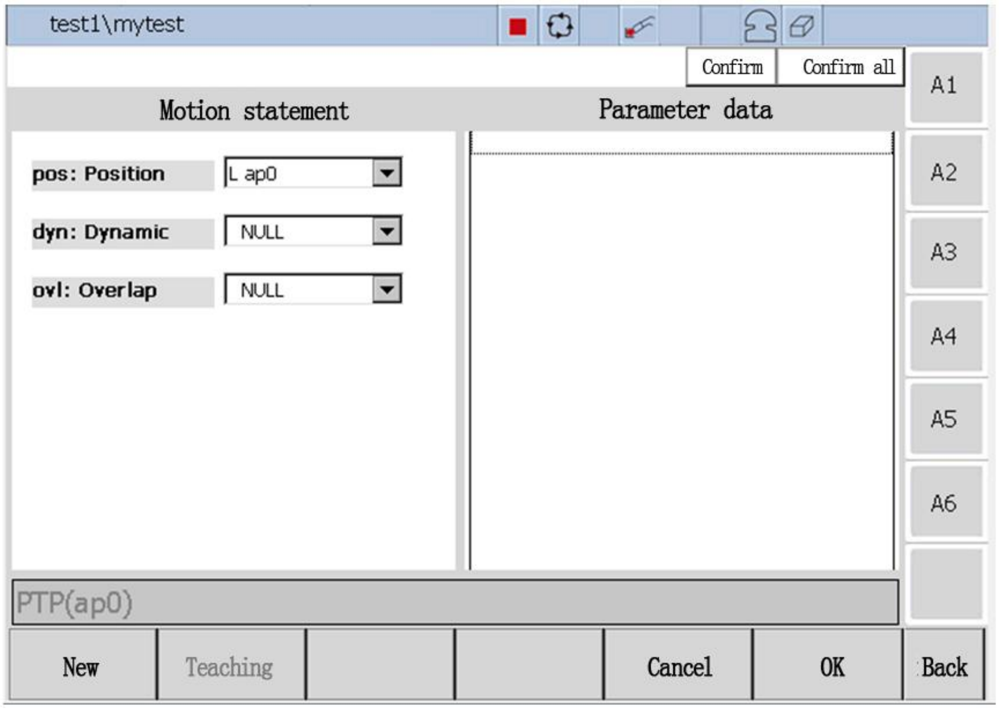
- Choose “New”, and display the statement classification. As shown in figure



- Select the motion statements, and select PPT, and click OK. As the figure shows



- Click OK, the position, dynamic performance and ovality parameters are displayed, as the figure shows.



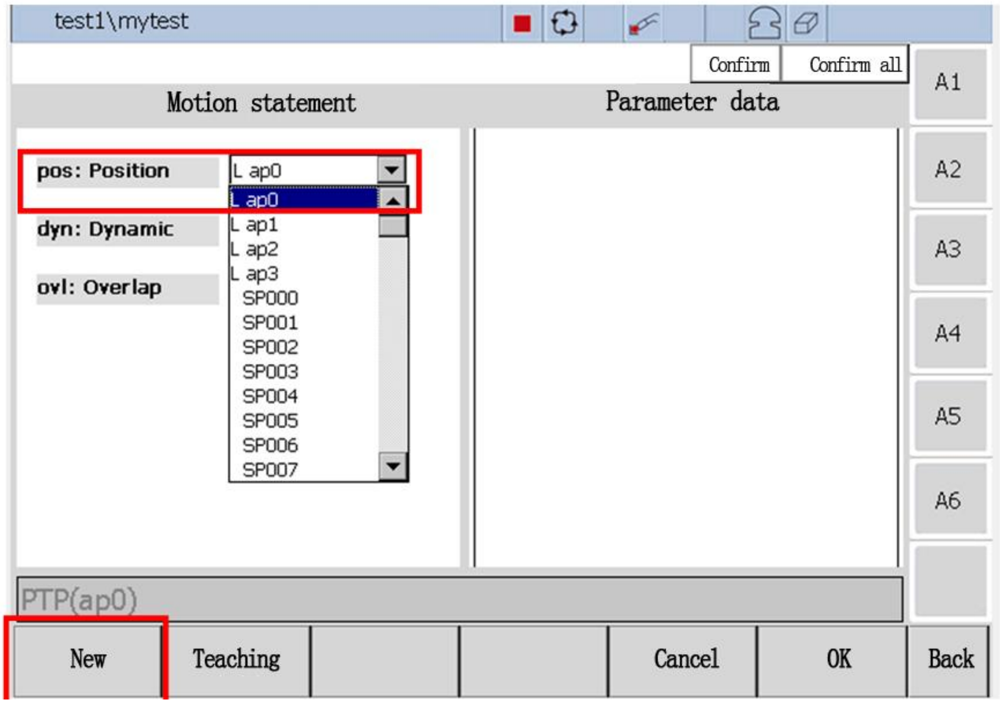
pos: Position L ap0

dyn: Dynamic NULL

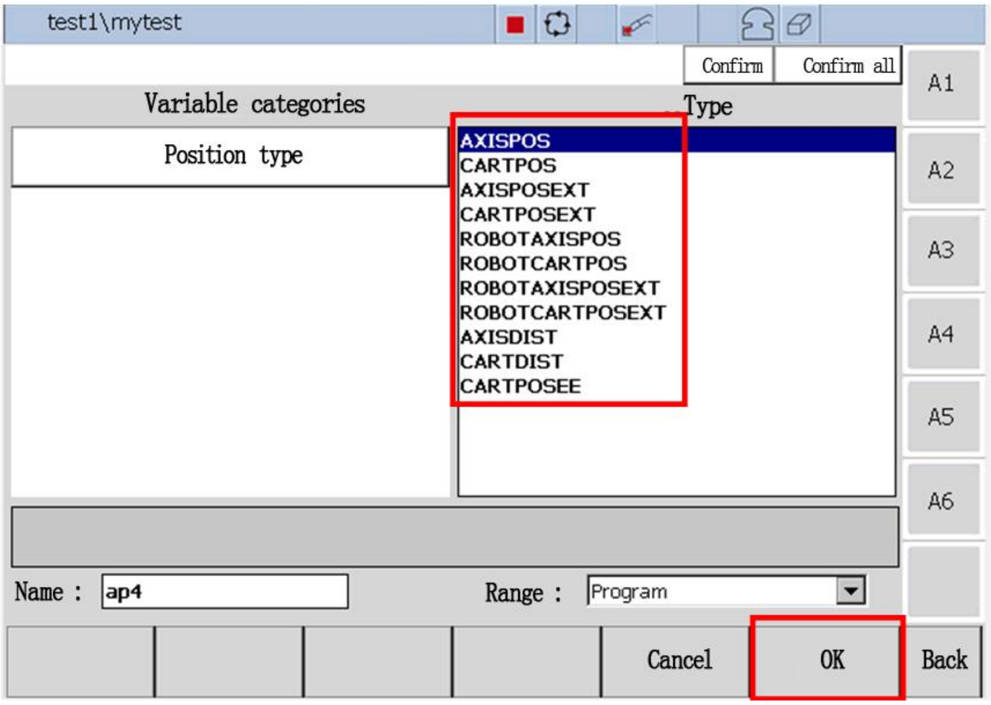
ovl: Overlap NULL

PTP(ap0)

- Select the position variable that has been established before at pos:Position (position), or you may choose to create a new one. As the figure below shows.



- After selecting to create a new one, select the position type, and click OK. As the figure below shows.

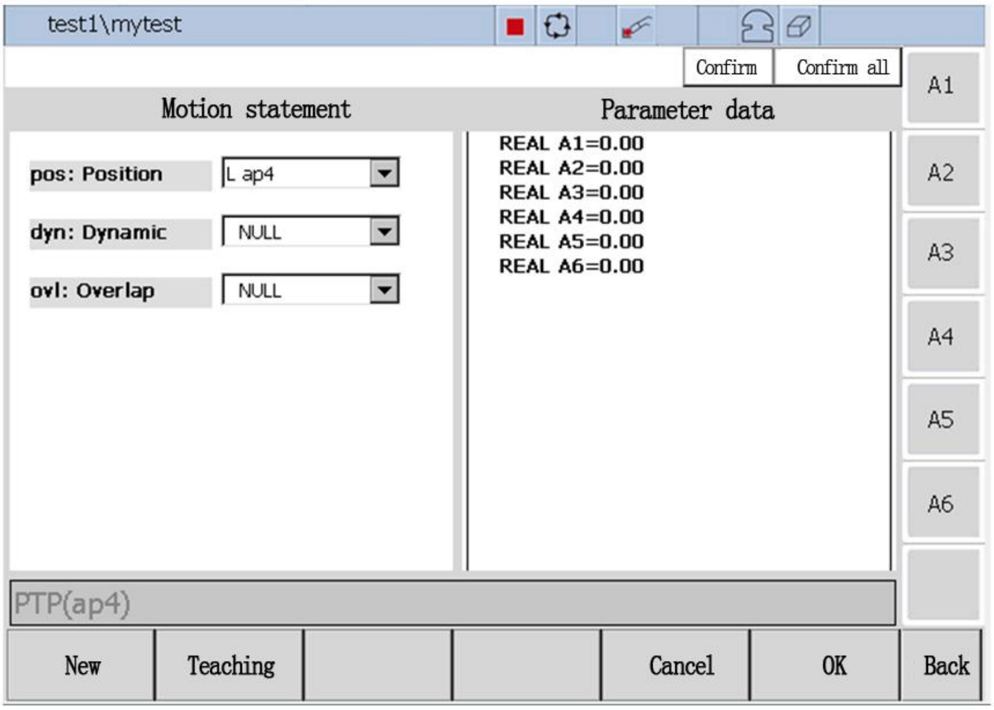


Name : ap4

Range : Program

- Where AXIS POS: joint + additional axis position; CART POS: Space + additional axis position.
- See the position data type in variable introductions for the details of type of right position.

- Following figure shows the situation when selecting the AXISPOS type and click OK.



test1\mytest

Motion statement

pos: Position: L ap4

dyn: Dynamic: NULL

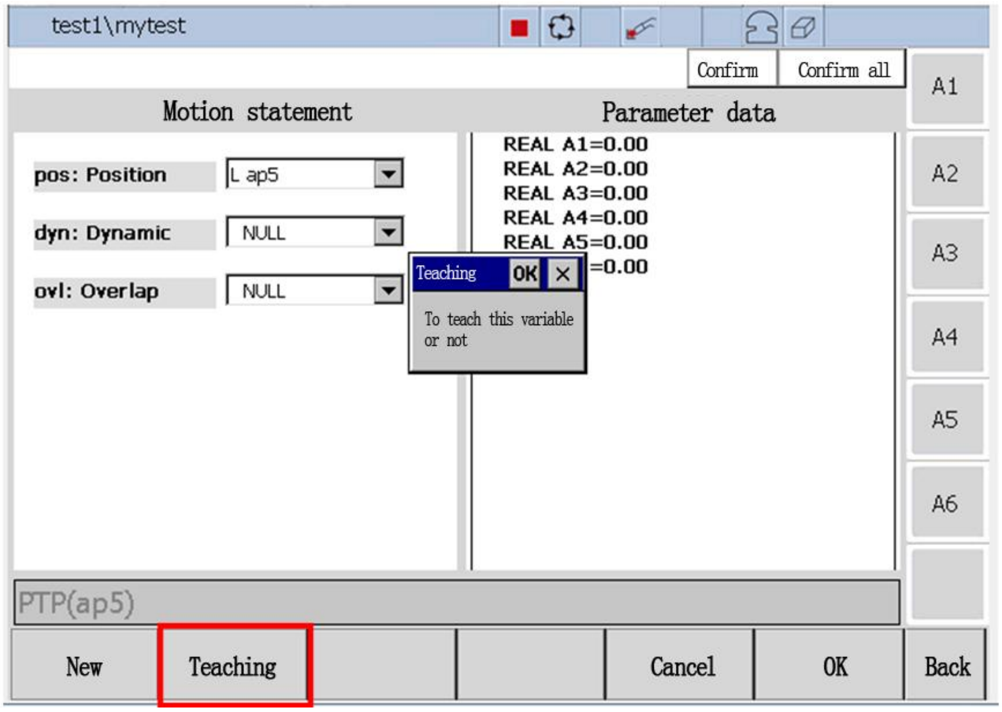
ovl: Overlap: NULL

Parameter data

REAL A1=0.00
REAL A2=0.00
REAL A3=0.00
REAL A4=0.00
REAL A5=0.00
REAL A6=0.00

PTP(ap4)

- Click Teaching after moving the robot to the position desired. As the figure below shows.



test1\mytest

Motion statement

pos: Position: L ap5

dyn: Dynamic: NULL

ovl: Overlap: NULL

Parameter data

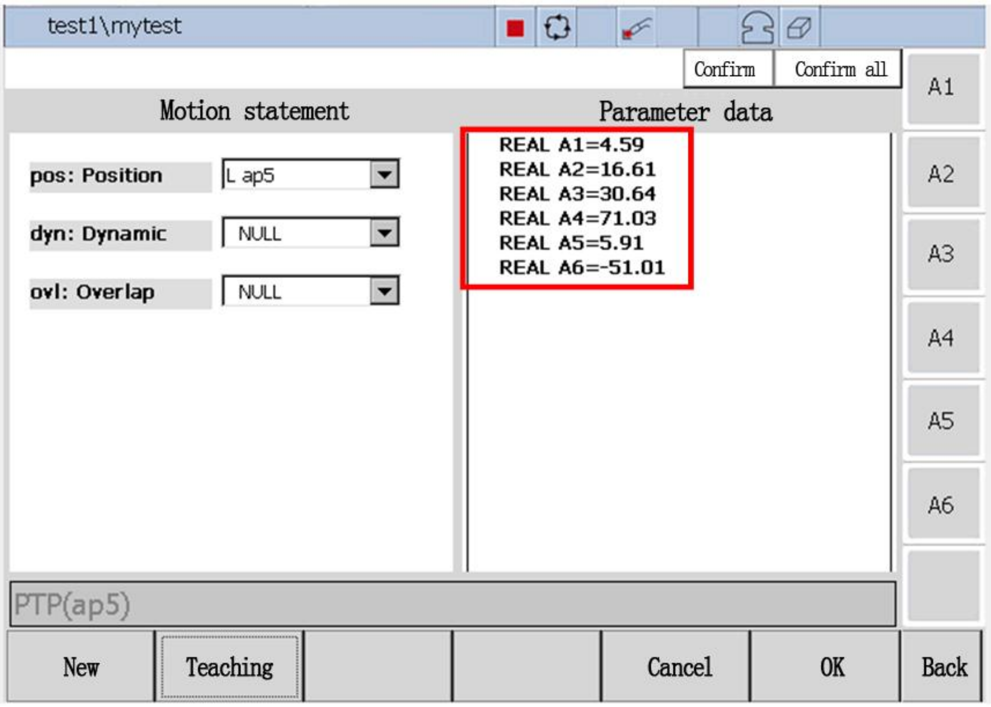
REAL A1=0.00
REAL A2=0.00
REAL A3=0.00
REAL A4=0.00
REAL A5=0.00
=0.00

Teaching

To teach this variable or not

PTP(ap5)

- Click OK and display the position of current point. As the figure shows



test1\mytest

Motion statement

pos: Position L ap5

dyn: Dynamic NULL

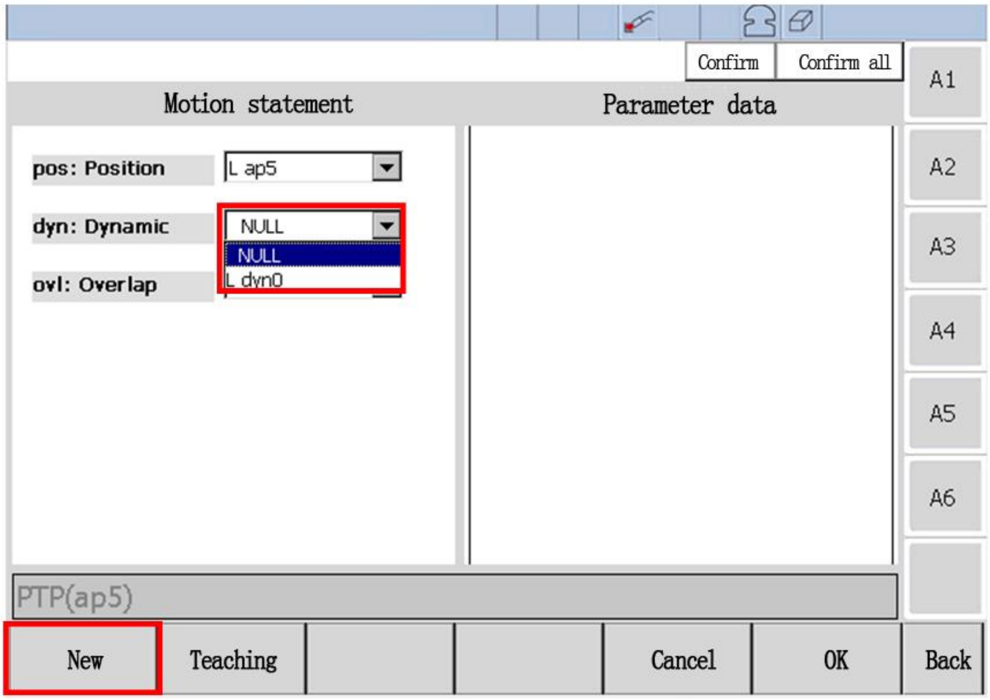
ovl: Overlap NULL

Parameter data

```
REAL A1=4.59
REAL A2=16.61
REAL A3=30.64
REAL A4=71.03
REAL A5=5.91
REAL A6=-51.01
```

PTP(ap5)

- Set the dynamic characteristic data dyn:Dynamic by selecting an existing parameter, or just creating a new parameter. As the figure shows



Motion statement

pos: Position L ap5

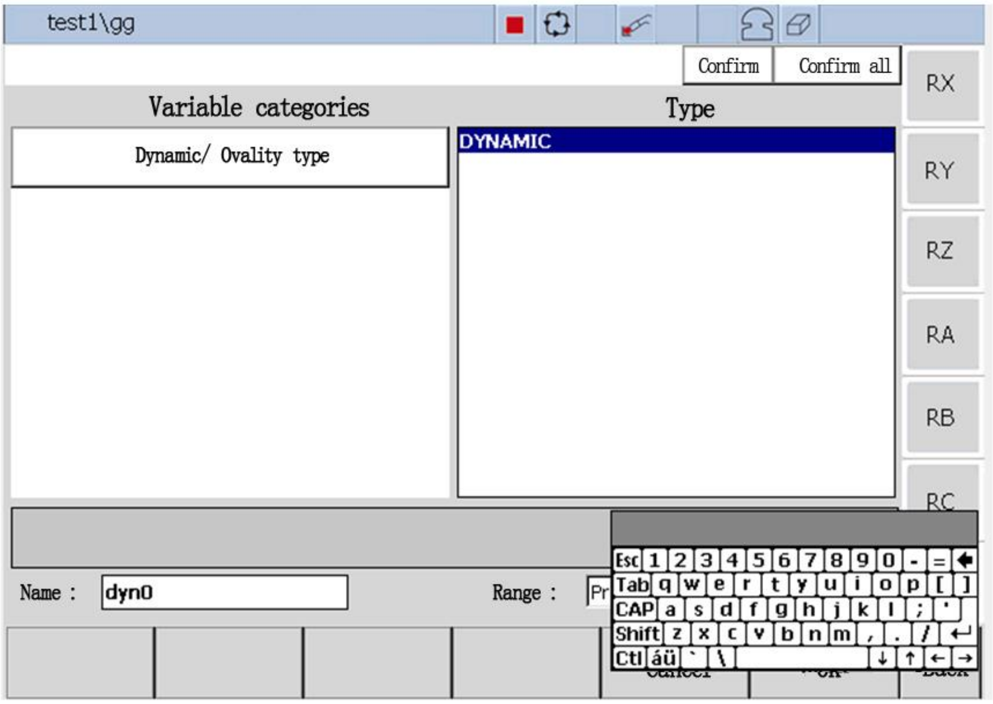
dyn: Dynamic NULL

ovl: Overlap

Parameter data

PTP(ap5)

- After choose to create a new parameter, only the DYNAMIC dynamic characteristics can be selected, or you may rename it. As the figure shows


test1\gg
Confirm
Confirm all
RX
RY
RZ
RA
RB
RC
Esc 1 2 3 4 5 6 7 8 9 0 - =
Tab q w e r t y u i o p [ ]
CAP a s d f g h j k l ; '
Shift z x c v b n m , . /
Ctl áü ` \ ↓ ↑ ← →
Cancel
OK
Back

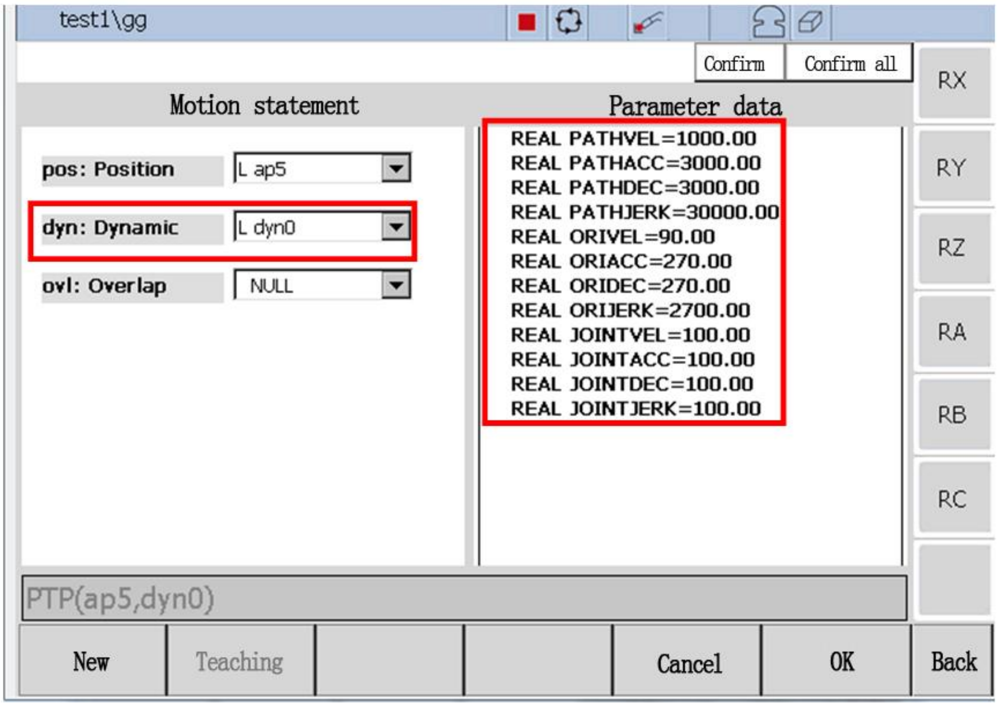
Variable categories: Dynamic/ Ovality type

Type: DYNAMIC

Name : dyn0

Range : Pr

- Following figure shows the parameter data after clicking OK.



Motion statement

pos: Position L ap5

dyn: Dynamic L dyn0

ovl: Overlap NULL

Parameter data

```
REAL PATHVEL=1000.00
REAL PATHACC=3000.00
REAL PATHDEC=3000.00
REAL PATHJERK=30000.00
REAL ORIVEL=90.00
REAL ORIACC=270.00
REAL ORIDEC=270.00
REAL ORIJERK=2700.00
REAL JOINTVEL=100.00
REAL JOINTACC=100.00
REAL JOINTDEC=100.00
REAL JOINTJERK=100.00
```

PTP(ap5,dyn0)

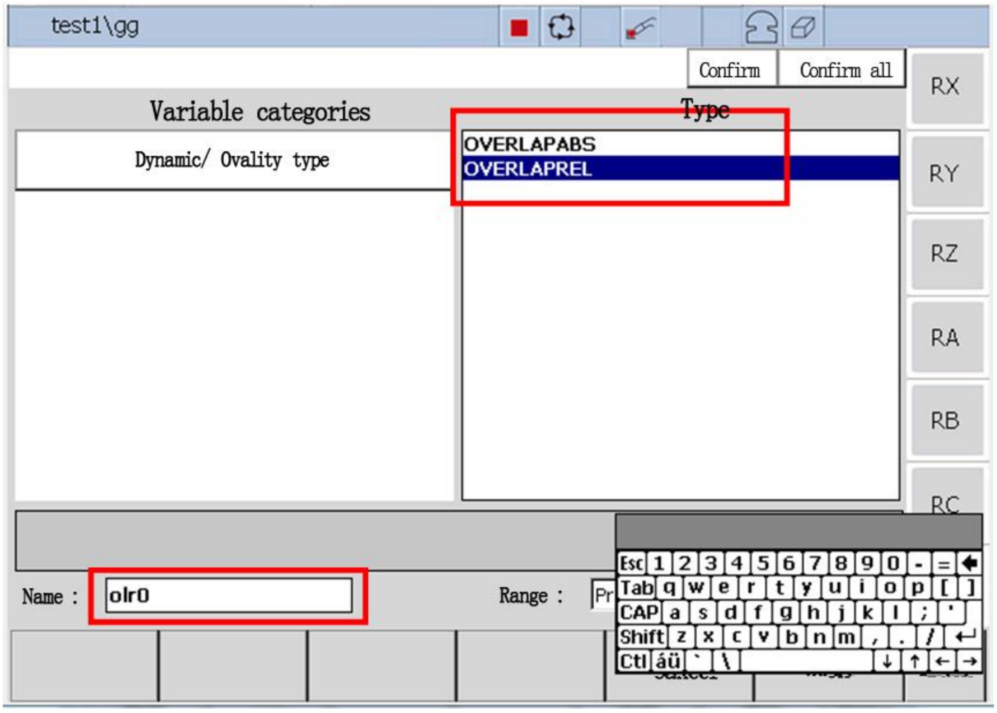
- ➢ Where the REAL PATHVEL position velocity (in MM/S, and 1000.00 is 1M/S)

REAL ORIVEL rotational speed (in deg/s, 90.00 is 90deg/S)

REAL JOINVEL joint speed (depending on the percentage of motor speed, and 100.00 is 100%).

- ➢ See the dynamic/ovality type parameters in variable introductions for the details of type of right position.

- ● Set the ovality parameter ovl: Overlap by selecting an existing parameter, or just creating a new parameter. As the figure shows



test1\gg

Variable categories: Dynamic/ Ovality type

Type:
OVERLAPABS
OVERLAPREL

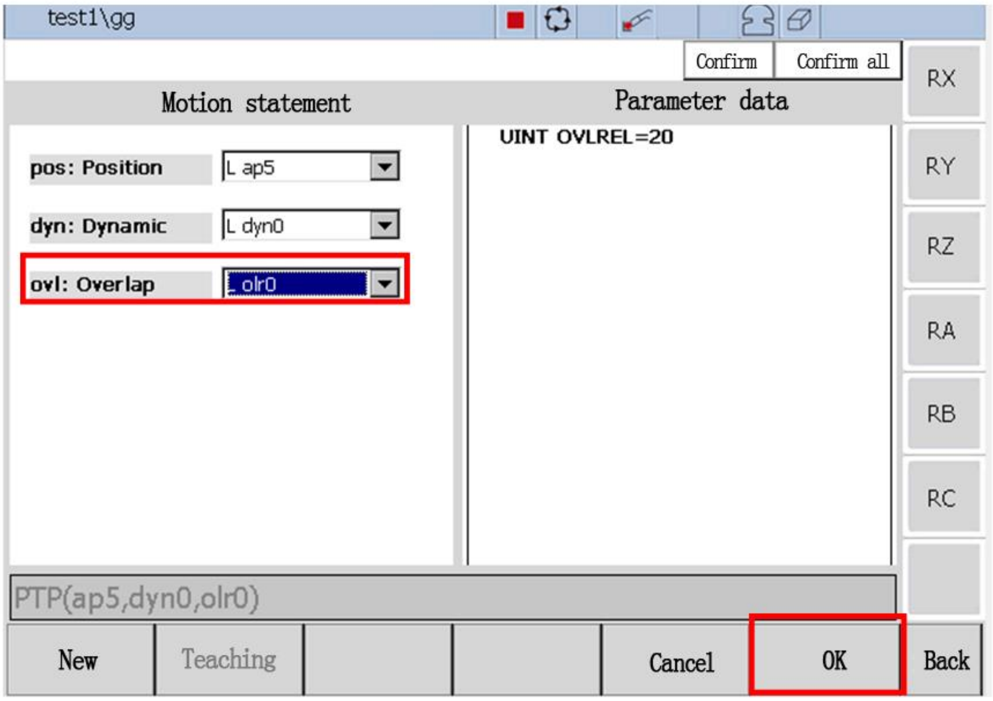
Name : olr0

- ➢ The right data in figure above is as follows:

OVERLAPABS ovality absolute value (Generally not used)

OVERLAPREL ovality percentage (commonly used)

- ● After clicking OK, and it will appear as the figure shows:



pos: Position L ap5

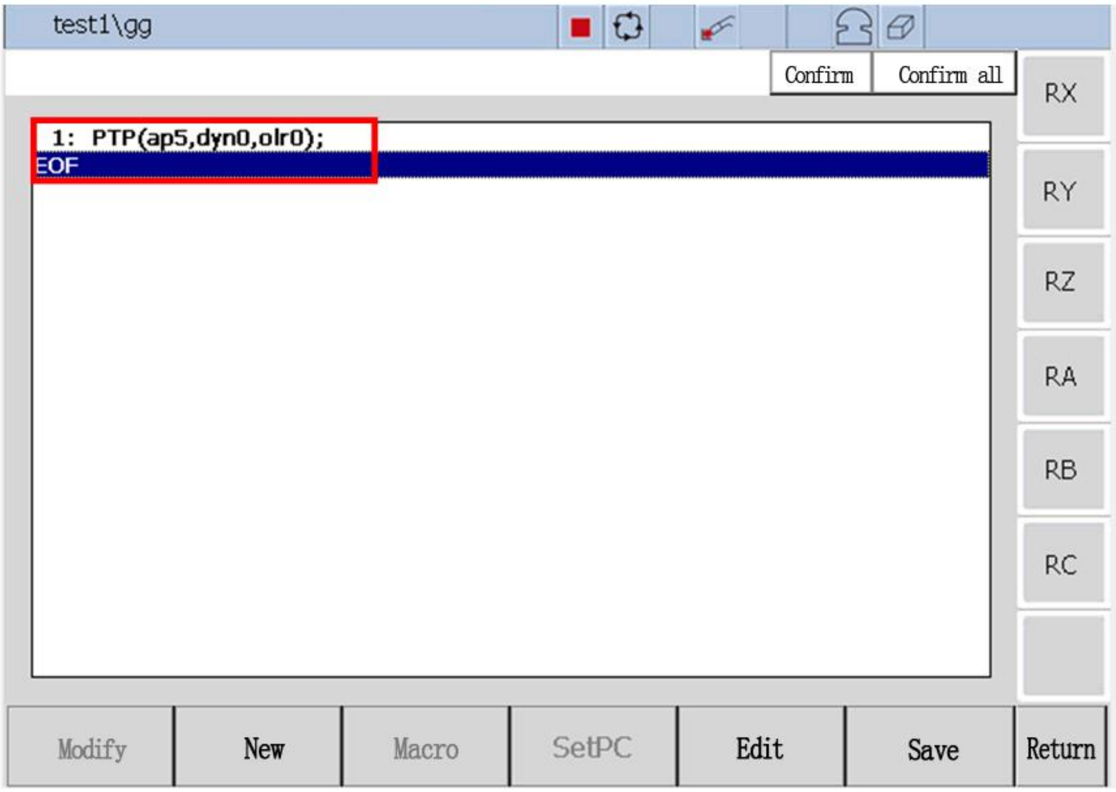
dyn: Dynamic L dyn0

ovl: Overlap L olr0

UINT OVLREL=20

PTP(ap5,dyn0,olr0)

- Click OK and there displays the written PTP point. As the figure shows



```
1:  PTP(ap5,dyn0,olr0);
EOF
```

- PTP usage:

- In case of only one parameter, the latter two parameters will use the default values PTP (param1); (common)

Have the first two user-defined parameters, and the third parameter uses the default value PTP (param1, optionparma2);

- The first and third use the user-defined parameters, but the second parameter uses the default value

PTP(param1 , NULL , optionparma3);

- With three parameters using not the default values, but the user-specified variable

PTP(param1, optionparma2, optionparma3);

3.2.2 LIN

- LIN: Linear motion, for which, the end of robot moves along a straight line to the specified position, and the posture and trajectory are determined. As the figure shows

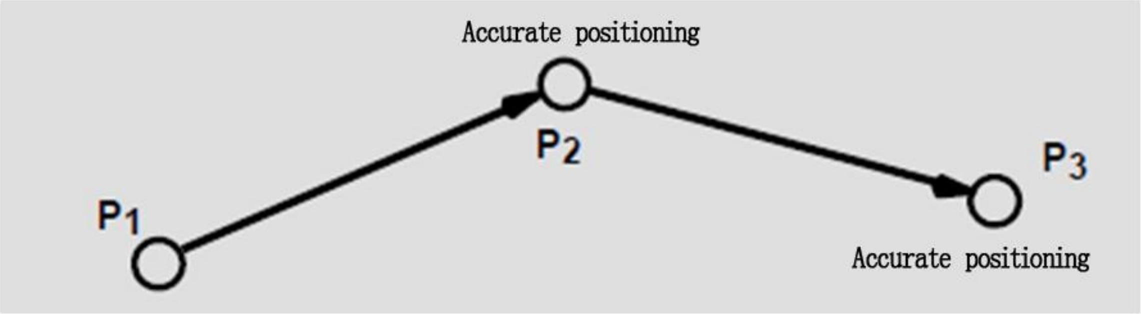
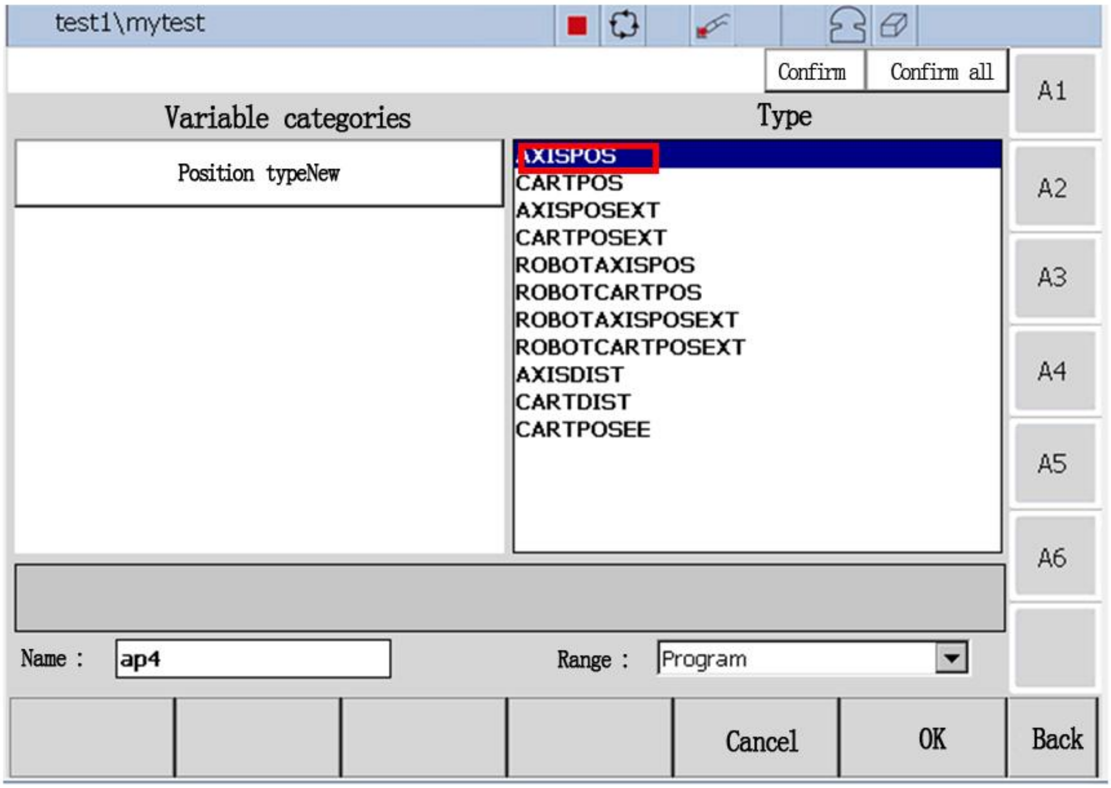


Figure 3-2 LIN schematic diagram

- Choose the motion statement, create a new Lin by referring to PTP. If you selecting CARTPOS type, and it shows in Figure as follows:



Variable categories

Position typeNew

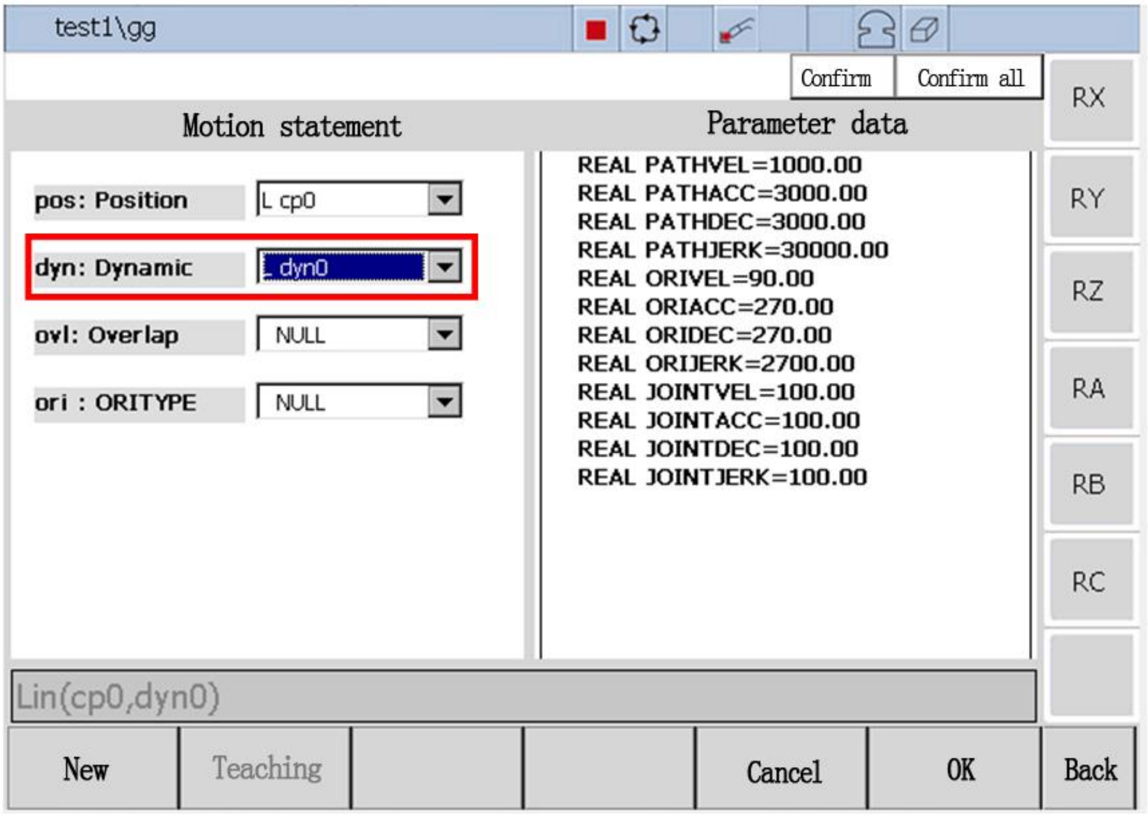
Type

\XISPOS
CARTPOS
AXISPOSEXT
CARTPOSEXT
ROBOTAXISPOS
ROBOTCARTPOS
ROBOTAXISPOSEXT
ROBOTCARTPOSEXT
AXISDIST
CARTDIST
CARTPOSEE

Name： ap4

Range： Program

- Dyn: Dynamic selects L dyn0, or may create a new one. As the figure shows



Motion statement

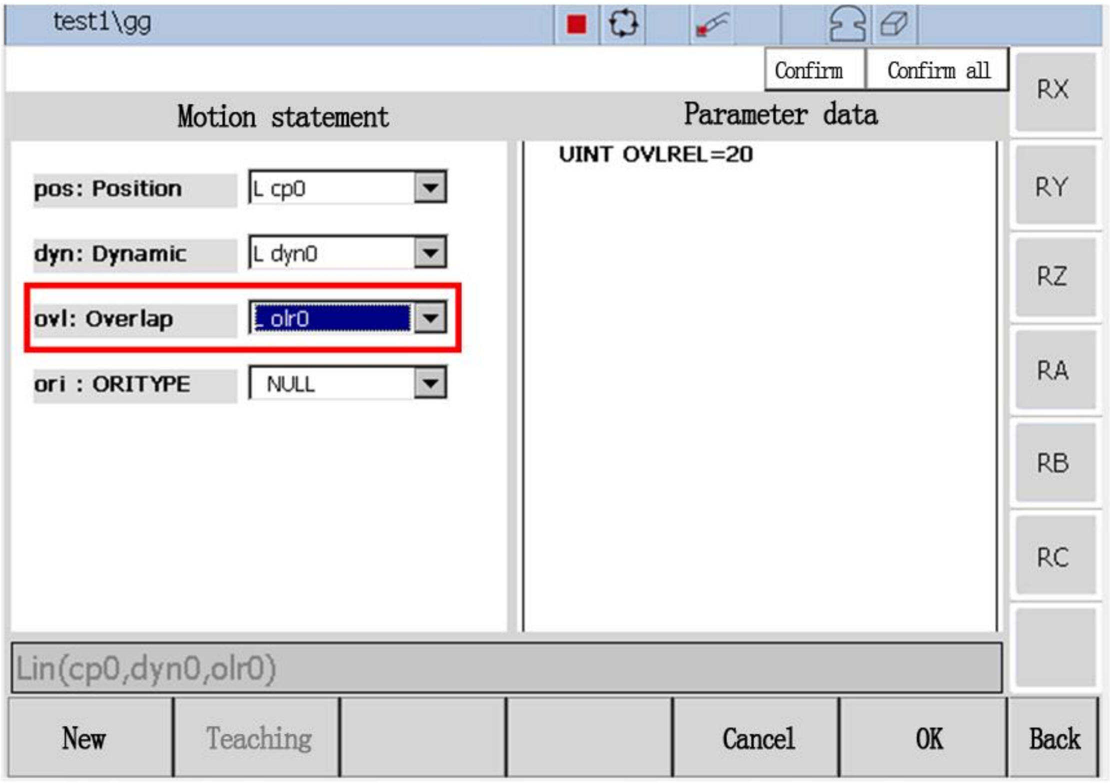
pos: Position L cp0
dyn: Dynamic L dyn0
ovl: Overlap NULL
ori : ORITYPE NULL

Parameter data

```
REAL PATHVEL=1000.00
REAL PATHACC=3000.00
REAL PATHDEC=3000.00
REAL PATHJERK=30000.00
REAL ORIVEL=90.00
REAL ORIACC=270.00
REAL ORIDEC=270.00
REAL ORIJERK=2700.00
REAL JOINTVEL=100.00
REAL JOINTACC=100.00
REAL JOINTDEC=100.00
REAL JOINTJERK=100.00
```

```
Lin(cp0,dyn0)
```

- Ovl:Overlap select L orl0, or may create a new one. As the figure shows



test1\gg

pos: Position L cp0

dyn: Dynamic L dyn0

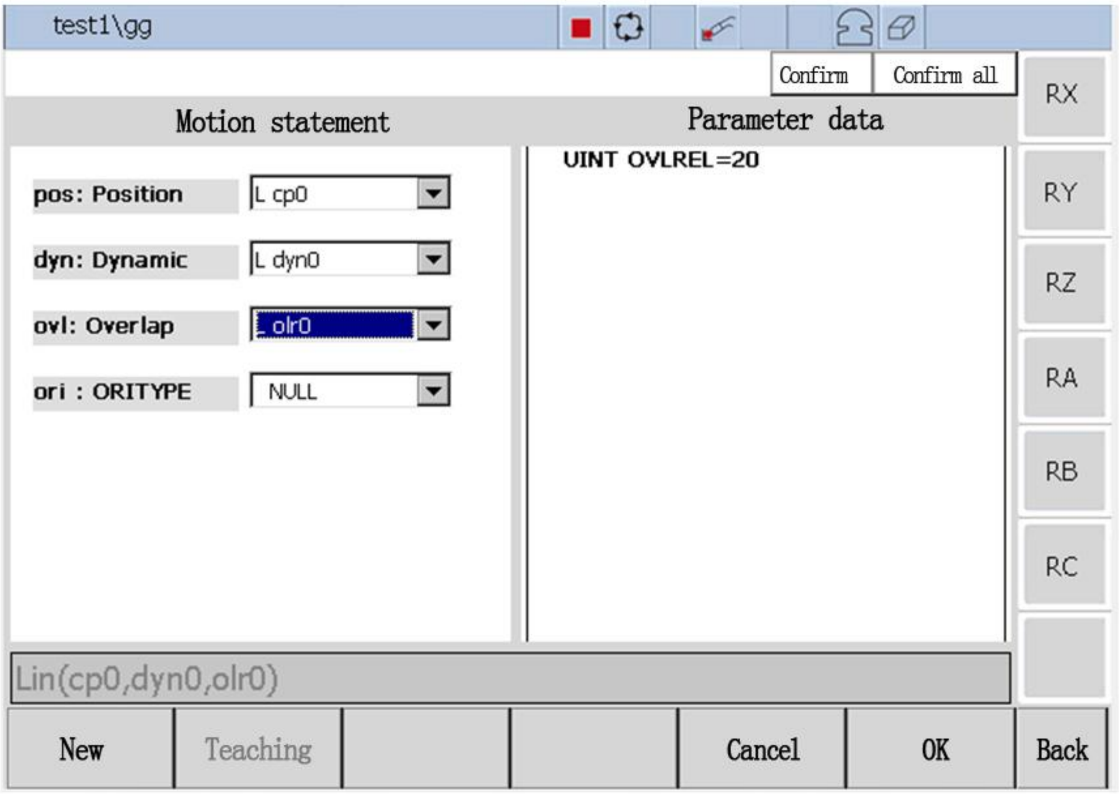
ovl: Overlap L olr0

ori : ORITYPE NULL

UINT OVLREL=20

Lin(cp0,dyn0,olr0)

- Click OK, and create a new ori: ORITYPE (indicate the posture interpolation mode on the path). As the figure shows



test1\gg

pos: Position L cp0

dyn: Dynamic L dyn0

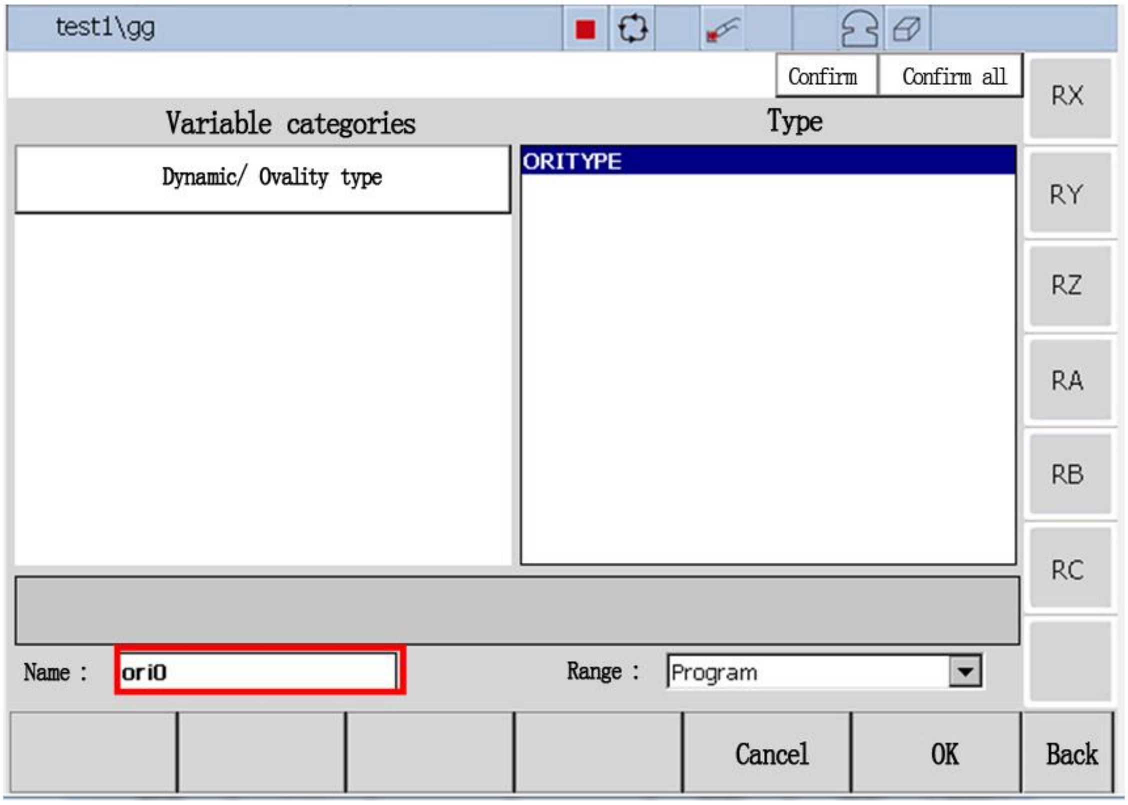
ovl: Overlap L olr0

ori : ORITYPE NULL

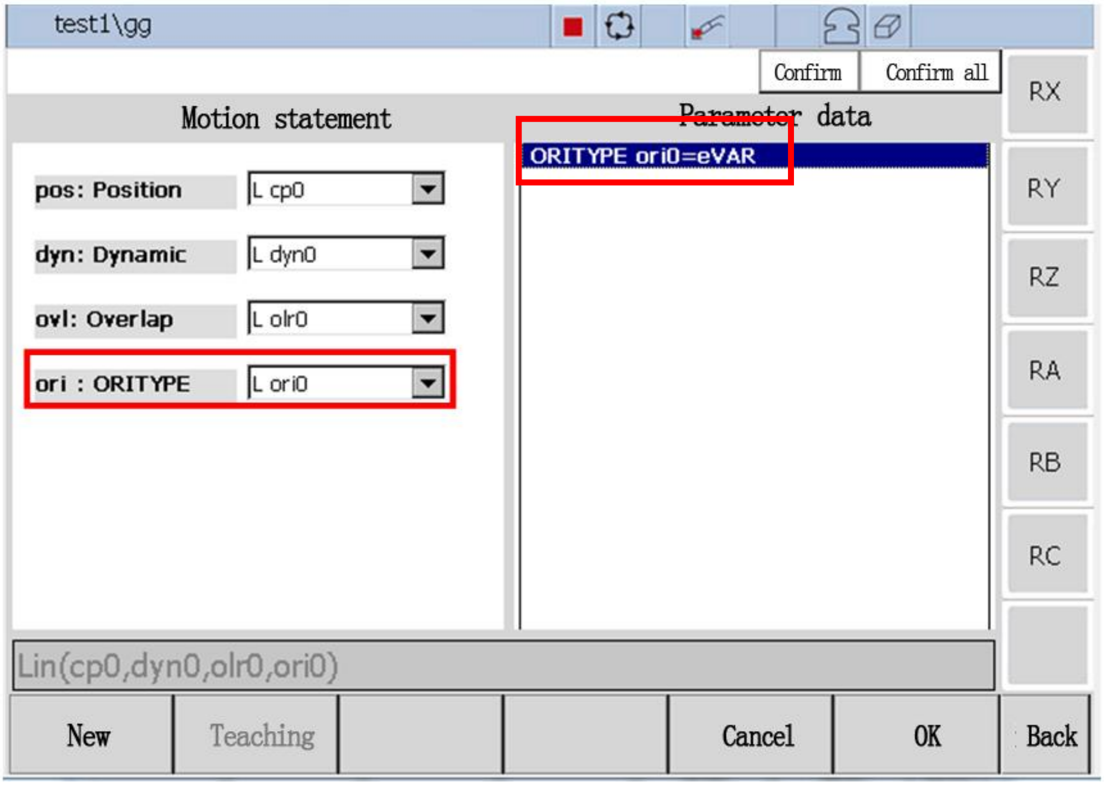
UINT OVLREL=20

Lin(cp0,dyn0,olr0)

- You may rename by clicking the “New”, as the figure shows.



- After clicking OK, it appears as the figure shows



- ORITYPE indicates the posture on the path and enumerated types. The value is only three.

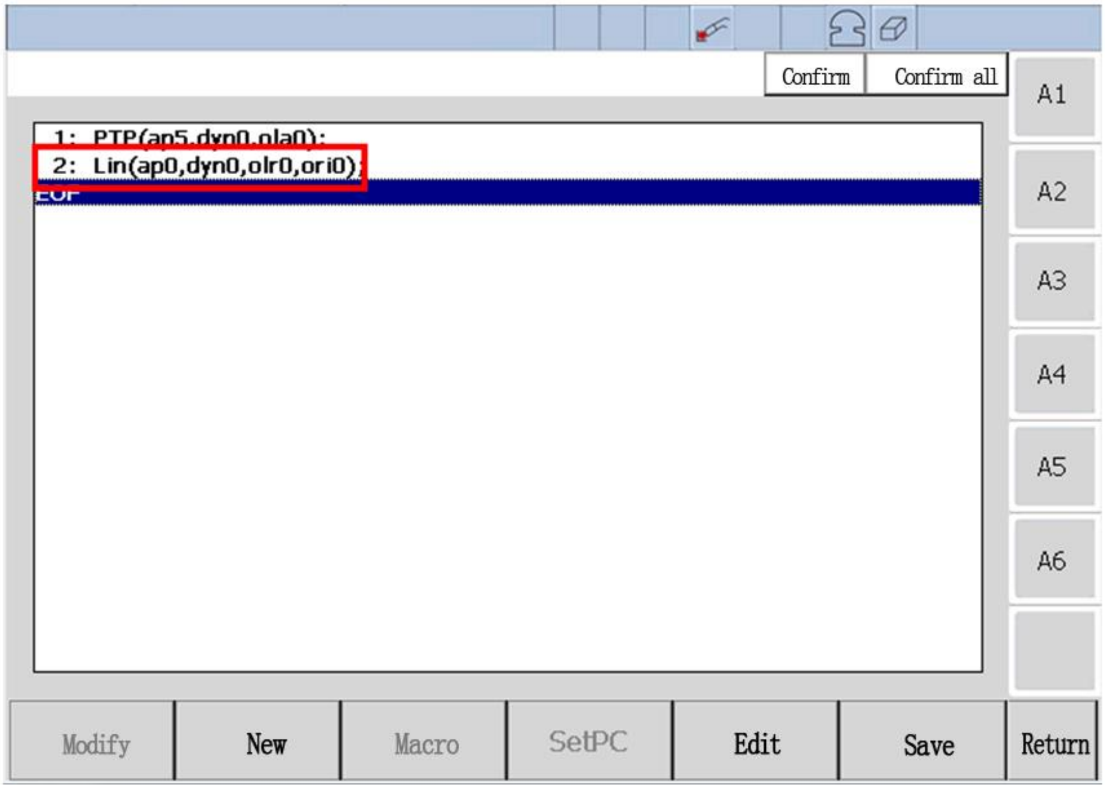
eVAR: // Make quaternion posture interpolation based on the posture information as demonstrated

eCONST: // Keep the posture of starting position always

eCIRCLE: // Keep the fixed posture with the round plane for circle interpolation

For example, ORITYPE oritype := eVAR

- Click OK and there displays the Lin, as shown in the figure.



```
1: PTP(ap5,dyn0,ola0);
2: Lin(ap0,dyn0,olr0,ori0);
EOF
```

➢ Usage of Lin:

The use rule of Lin statement is the same as what is described in PTP. Lin can have up to four parameters, where the first is the required, and others are optional.

3.2.3 Circ

- Circ: Circular motion, for which the end of robot moves to the specified position along a circular arc, and the posture uncertain trajectory are determined. As the figure shows

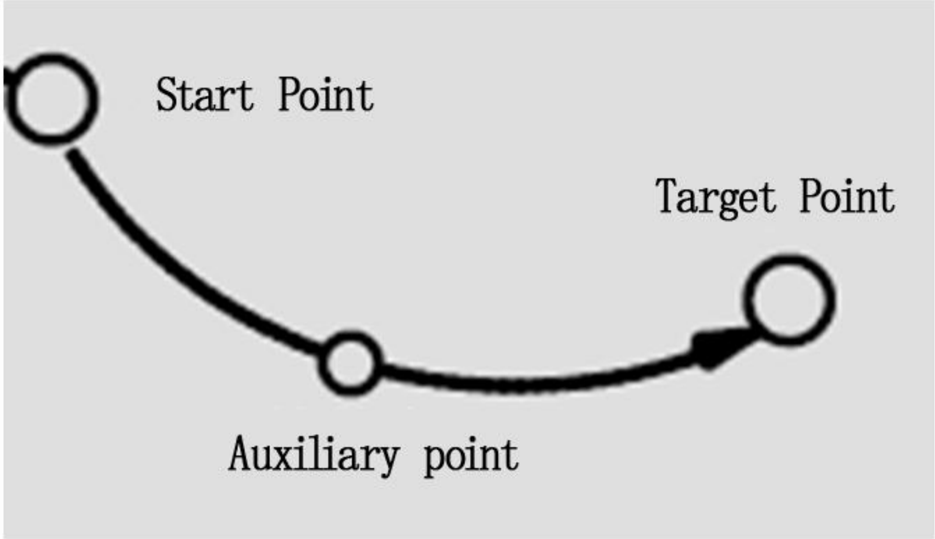
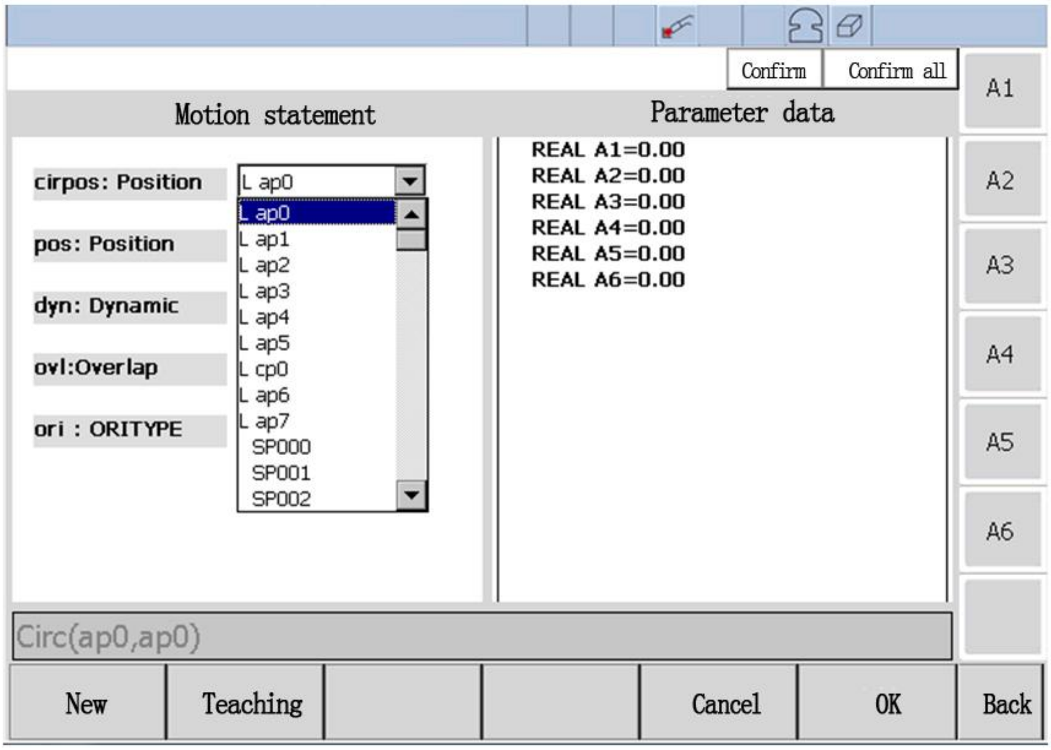


Figure 3-3 Schematic diagram of circle

- Choose the motion statement, and create a new Cric. As the figure shows



Motion statement

cirpos: Position — L ap0

pos: Position

dyn: Dynamic

ovl:Overlap

ori : ORITYPE

L ap0
L ap1
L ap2
L ap3
L ap4
L ap5
L cp0
L ap6
L ap7
SP000
SP001
SP002

Parameter data

REAL A1=0.00
REAL A2=0.00
REAL A3=0.00
REAL A4=0.00
REAL A5=0.00
REAL A6=0.00

Circ(ap0,ap0)

- Parameters:

  Cripos: Position// Auxiliary position on the arc

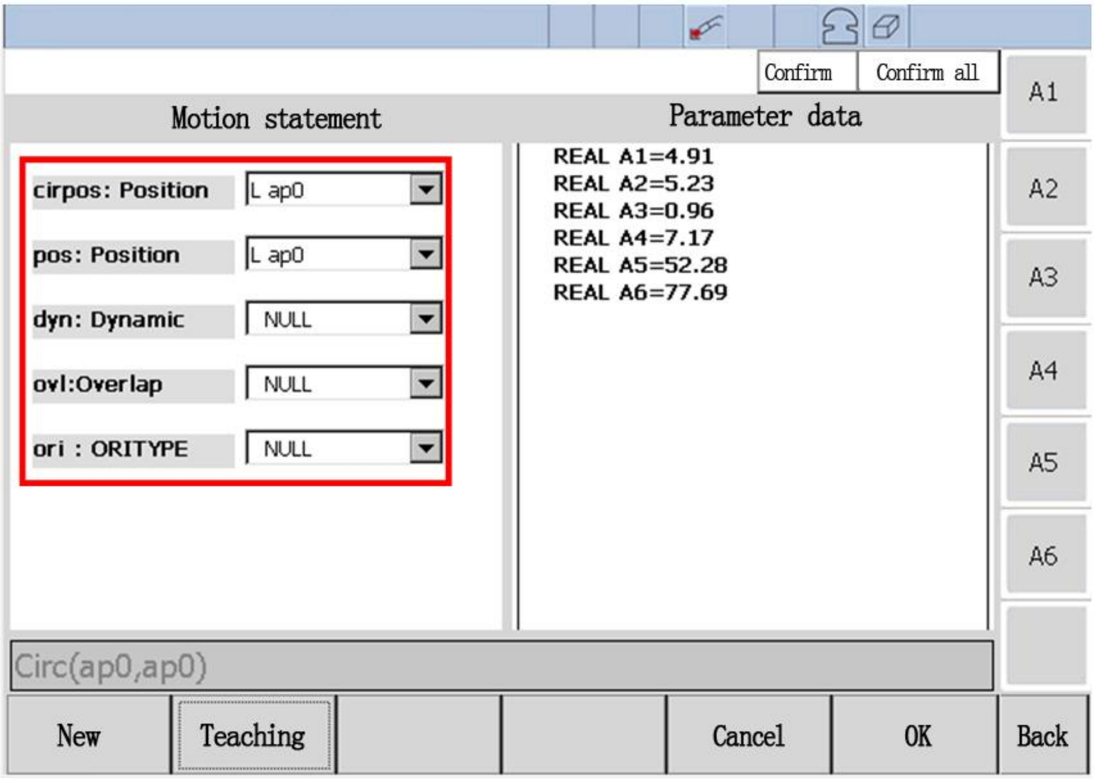
  pos : Position // Destination position

  dyn : Dynamic //Dynamic characteristics

  ovl : Overlap//Ovality data

  ori : ORITYPE //Posture parameters

- Choose the parameter, or create a new one by referring to PTP. As the figure shows



Motion statement

cirpos: Position: L ap0

pos: Position: L ap0

dyn: Dynamic: NULL

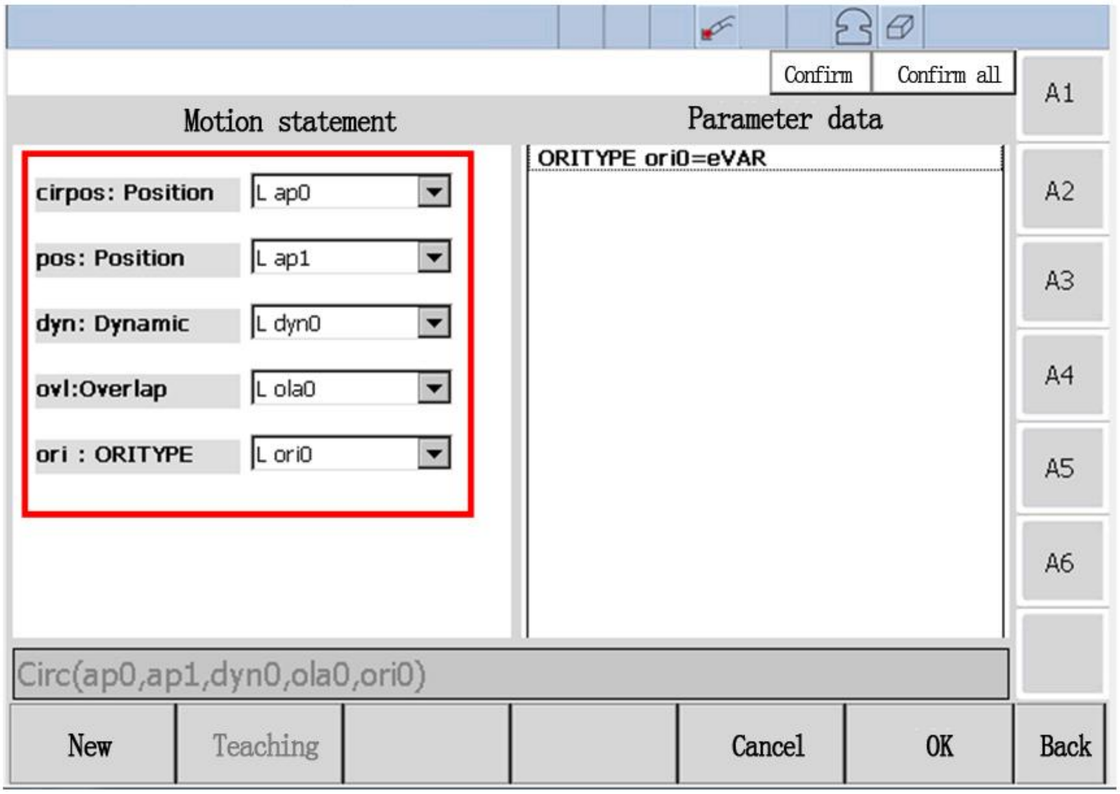
ovl:Overlap: NULL

ori : ORITYPE: NULL

Parameter data

```
REAL A1=4.91
REAL A2=5.23
REAL A3=0.96
REAL A4=7.17
REAL A5=52.28
REAL A6=77.69
```

Circ(ap0,ap0)



Motion statement

cirpos: Position: L ap0

pos: Position: L ap1

dyn: Dynamic: L dyn0

ovl:Overlap: L ola0

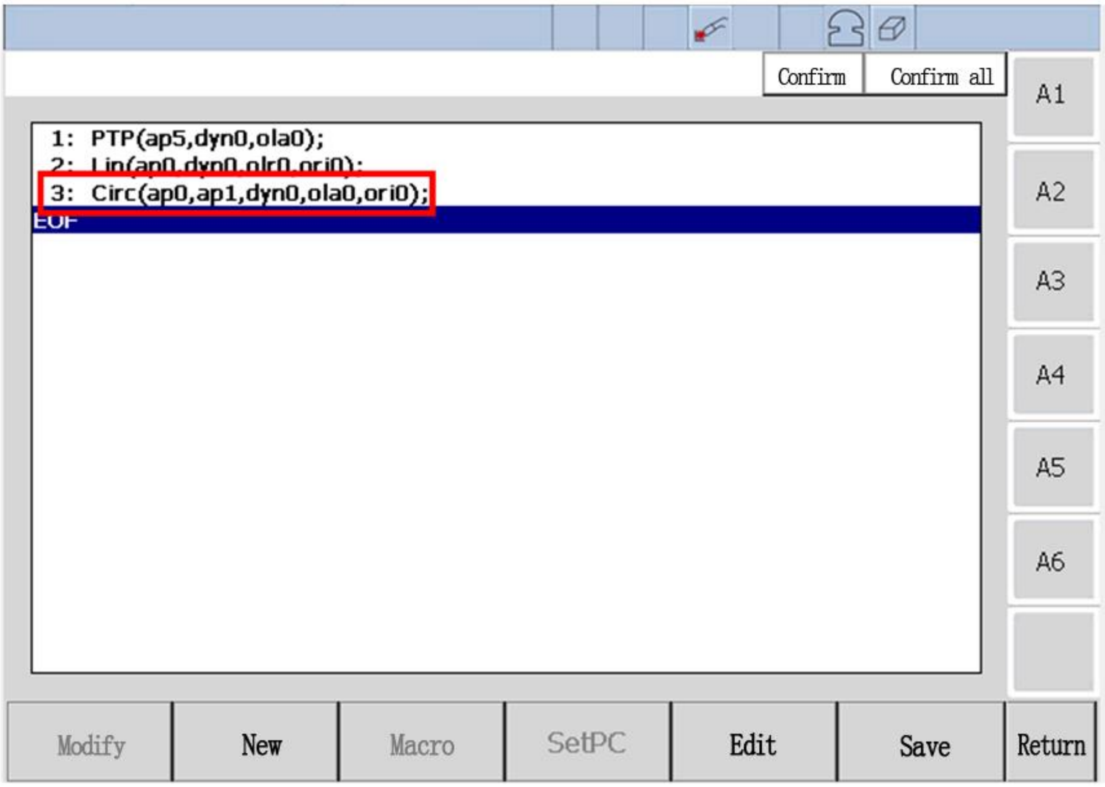
ori : ORITYPE: L ori0

Parameter data

```
ORITYPE ori0=eVAR
```

Circ(ap0,ap1,dyn0,ola0,ori0)

- Click OK and there displays the Circ. As the figure shows



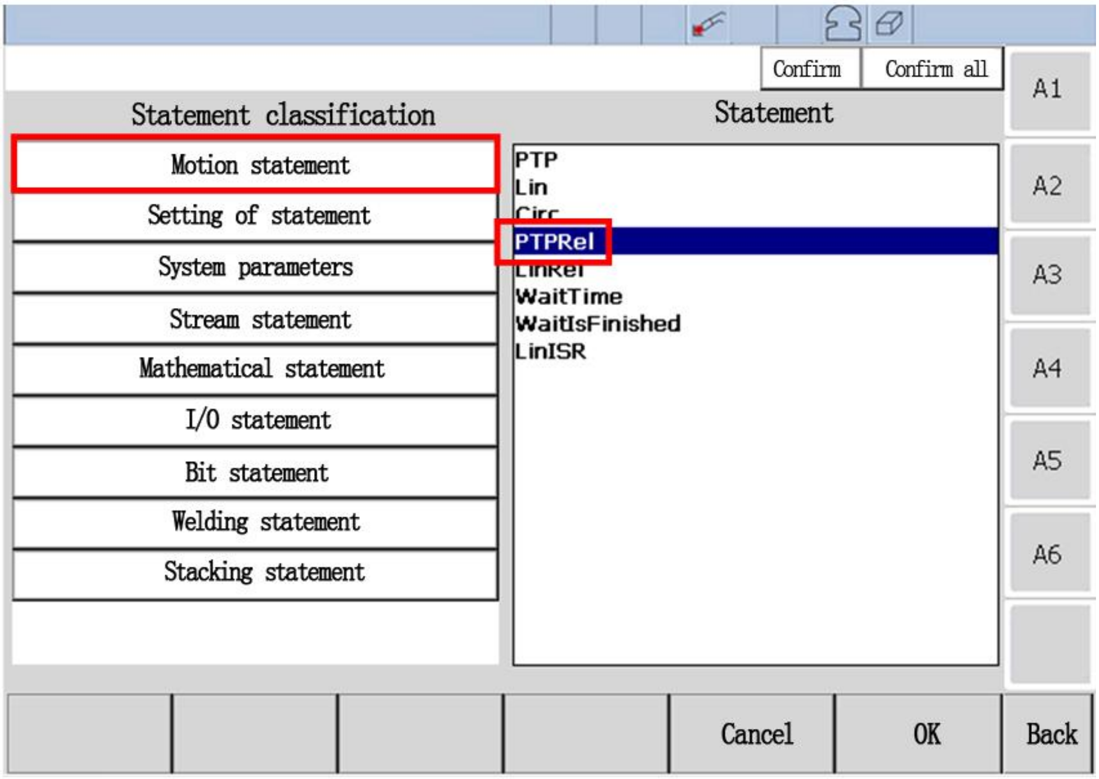
```
1: PTP(ap5,dyn0,ola0);
2: Lin(ap0,dyn0,olr0,ori0);
3: Circ(ap0,ap1,dyn0,ola0,ori0);
EOF
```

➢ Usage of Circ:

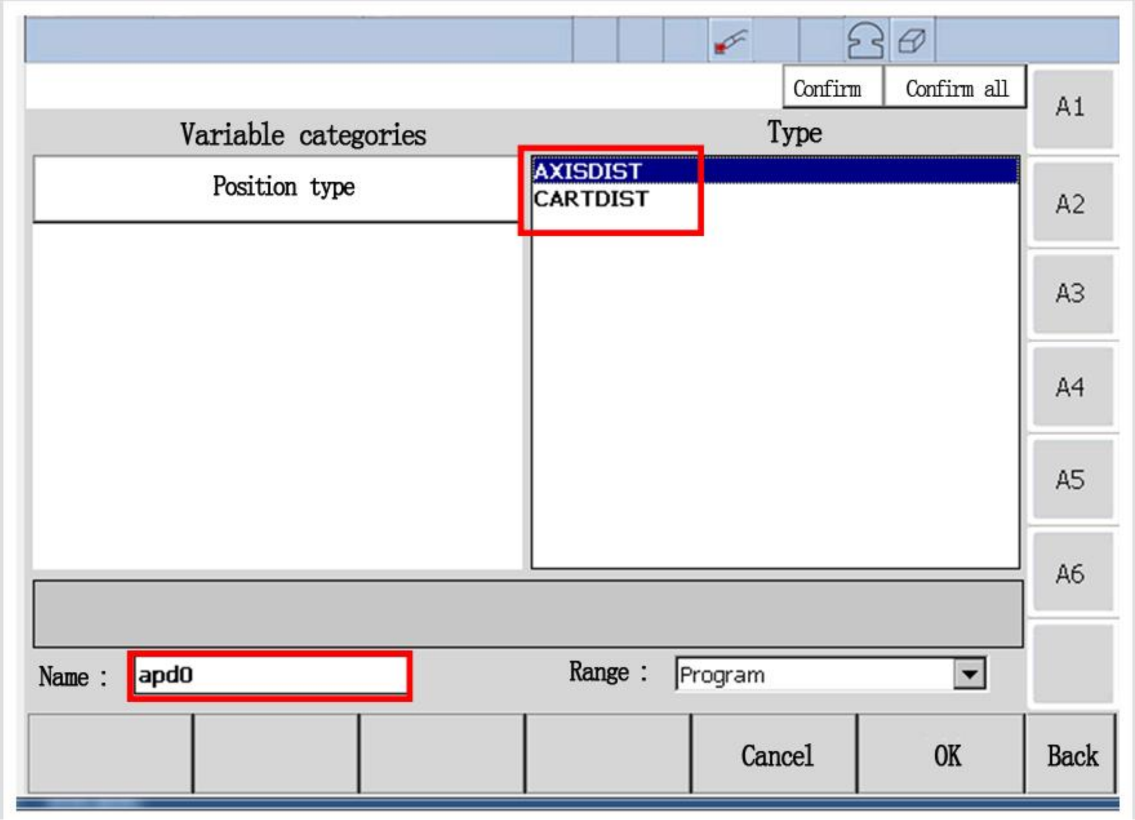
The instruction has two required parameters Circ (param1, parma2);

3.2.4 PTPRel

- PTPRel(dist,dyn,ovl) is a point-to-point relative motion instruction, where the moving distance is against the initial position of the robot (the position depends on the previous motion instruction).
- Choose the motion statement, and create a new PTPRo. As the figure shows



- Click OK and select to create a new apd0. As the figure shows



Name : apd0

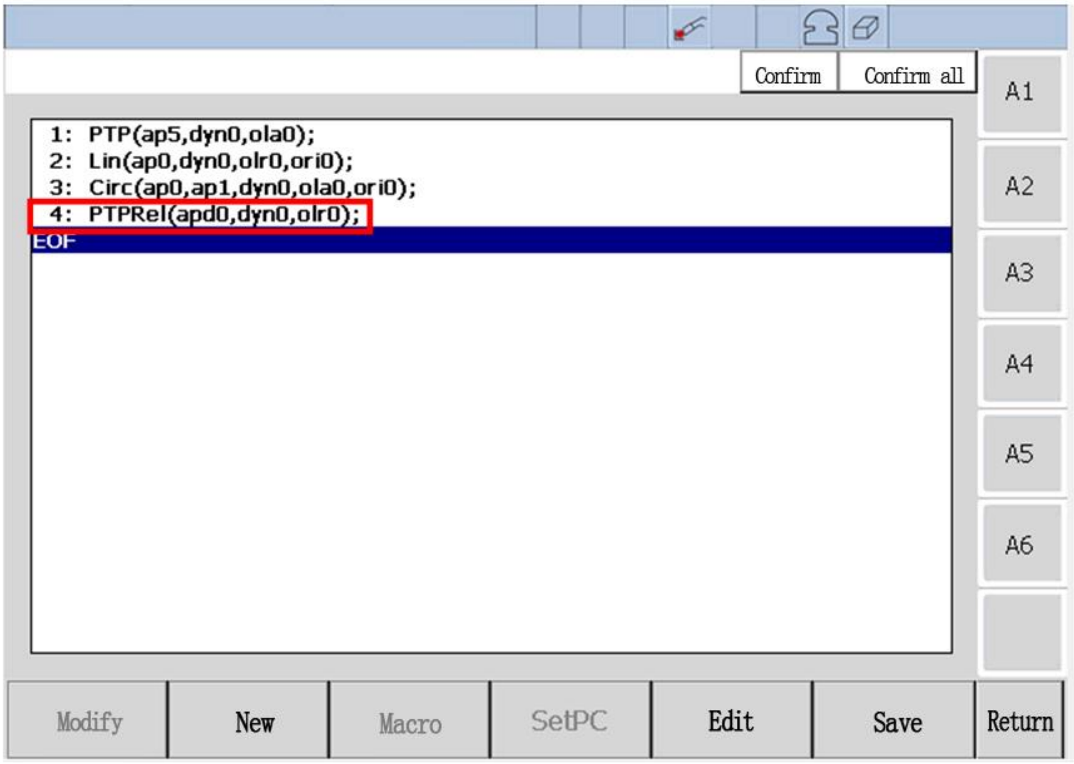
Range : Program

- The right side in figure above shows as follows:

AXISDIST: Relative joint+ additional axis position, for which, there're 6 joints+6 additional axes at most, and the type of each variable is DOUBLE, measured in degrees (°).

CARTDIST: The spatial position under the Cartesian coordinate system, with additional axes up to 6. Variable is consisted of the X, Y, Z, A, B, C, Aux1 ~ 6 and Mode, wherein X ~ Aux6 is DOUBLE type, and Mode is UINT type.

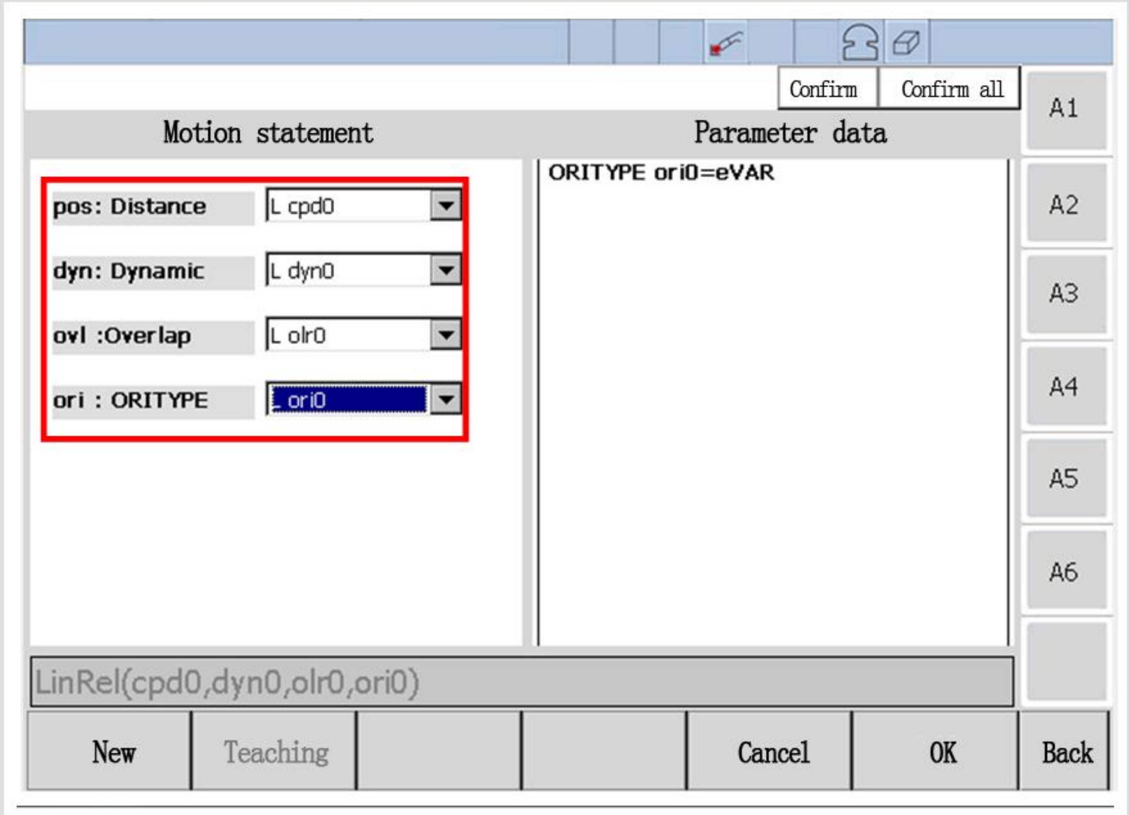
- Click OK and there displays the PTPRel. As the figure shows



```
1: PTP(ap5,dyn0,ola0);
2: Lin(ap0,dyn0,olr0,ori0);
3: Circ(ap0,ap1,dyn0,ola0,ori0);
4: PTPRel(apd0,dyn0,olr0);
EOF
```

3.2.5 LinRel

- LinRel (dist, dyn, ovl, ori) is a relative linear motion instruction, where the moving distance is relative to the initial position of the robot (the position depends on the previous motion instruction).
- Choose the motion statement, create a new LinRel, and set parameters (refer to PTP). As shown in figure:



Motion statement

pos: Distance: L cpd0

dyn: Dynamic: L dyn0

ovl :Overlap: L olr0

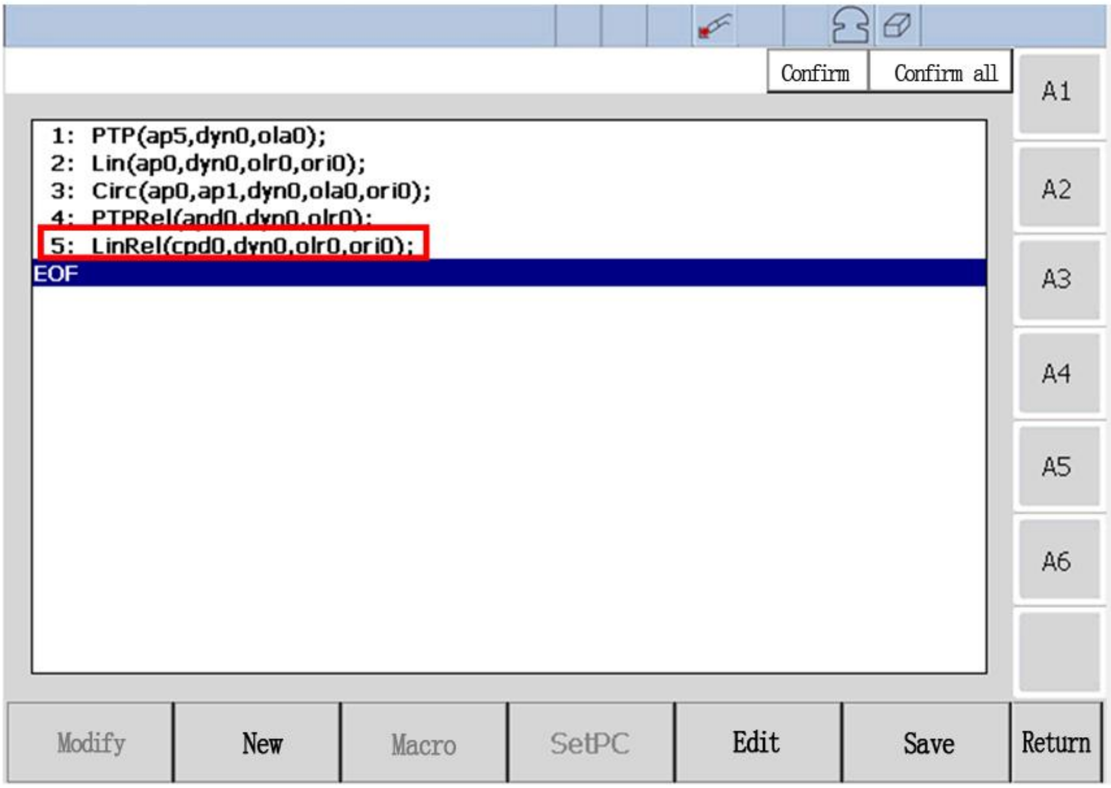
ori : ORITYPE: L ori0

Parameter data

ORITYPE ori0=eVAR

LinRel(cpd0,dyn0,olr0,ori0)

- Click OK and there displays the LinRel. As the figure shows

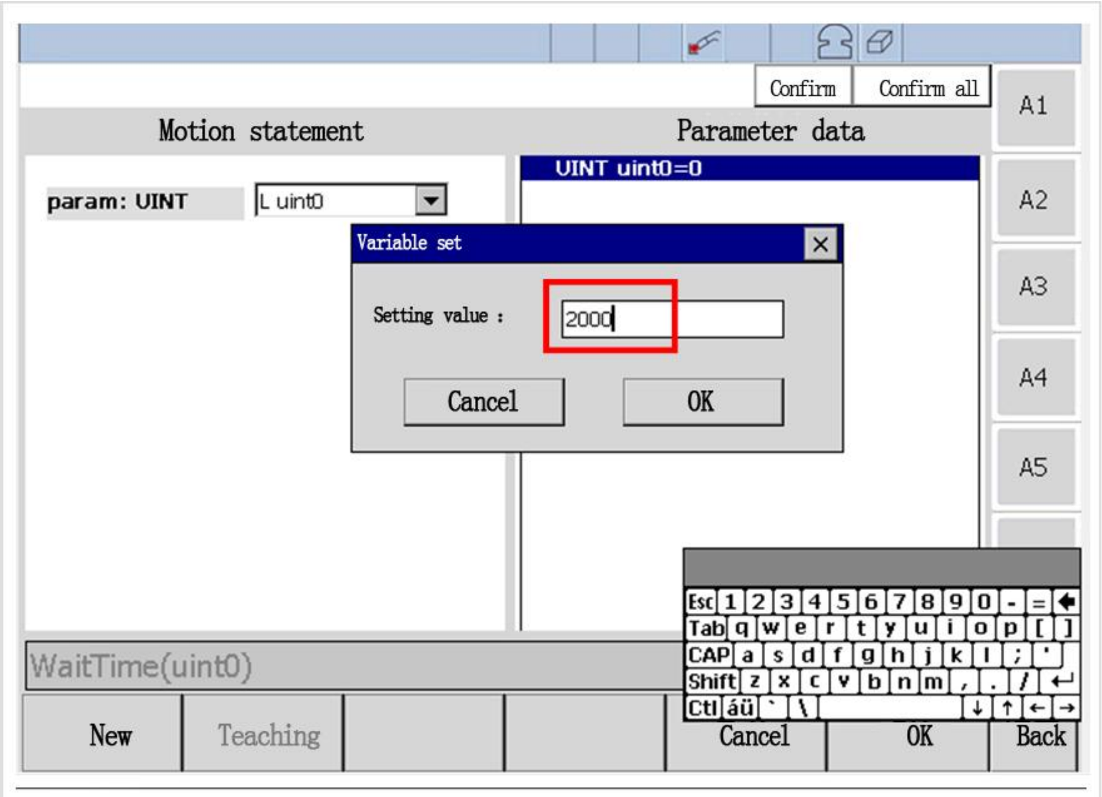


```
1:  PTP(ap5,dyn0,ola0);
2:  Lin(ap0,dyn0,olr0,ori0);
3:  Circ(ap0,ap1,dyn0,ola0,ori0);
4:  PTPRel(apd0,dyn0,olr0);
5:  LinRel(cpd0,dyn0,olr0,ori0);
EOF
```

3.2.6 WaitTime (timeMs)

- The instruction pauses the robot timeMs time.

- Choose the motion statement, create a new WaitTime, while the new variable is uint 0, and also set the pause time. As shown in Figure (2000 indicate that the pause lasts for two seconds).


Confirm
Confirm all
A1
A2
A3
A4
A5
Cancel
OK
Esc 1 2 3 4 5 6 7 8 9 0 - =
Tab q w e r t y u i o p [ ]
CAP a s d f g h j k l ; '
Shift z x c v b n m , . /
Ctl áü ` \ ↓ ↑ ← →
New
Teaching
Cancel
OK
Back

Motion statement

Parameter data

param: UINT L uint0

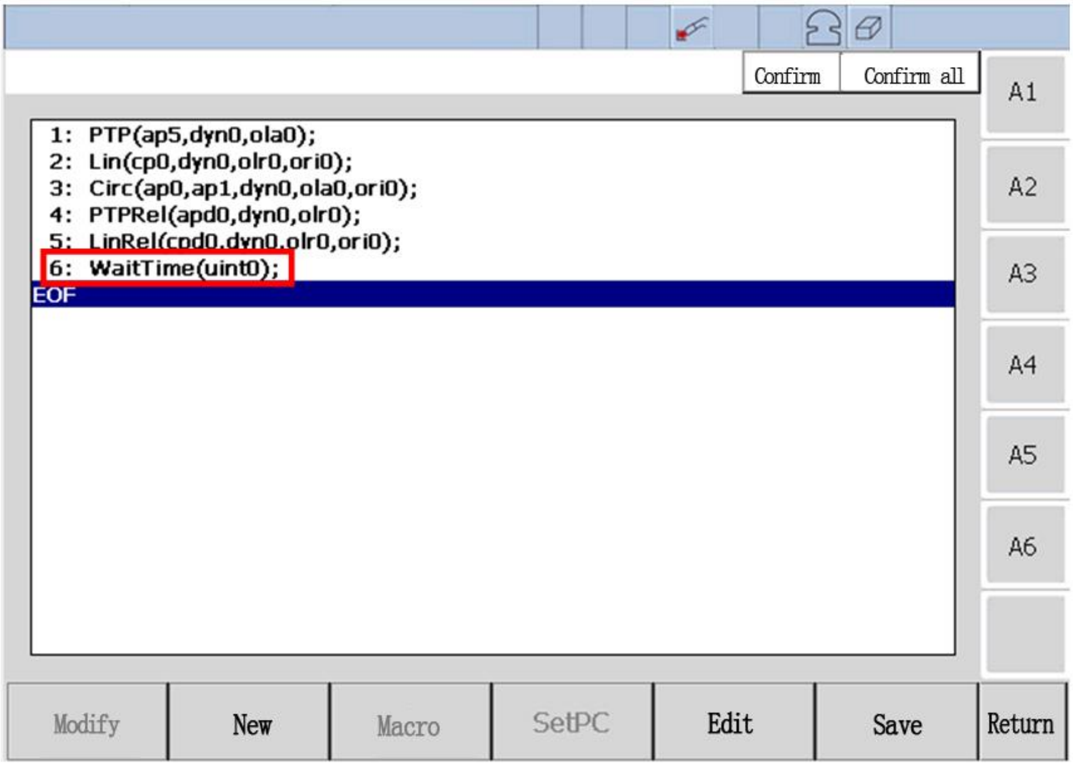
UINT uint0=0

Variable set

Setting value : 2000

WaitTime(uint0)

- Click OK and there displays the WaitTime. As the figure shows

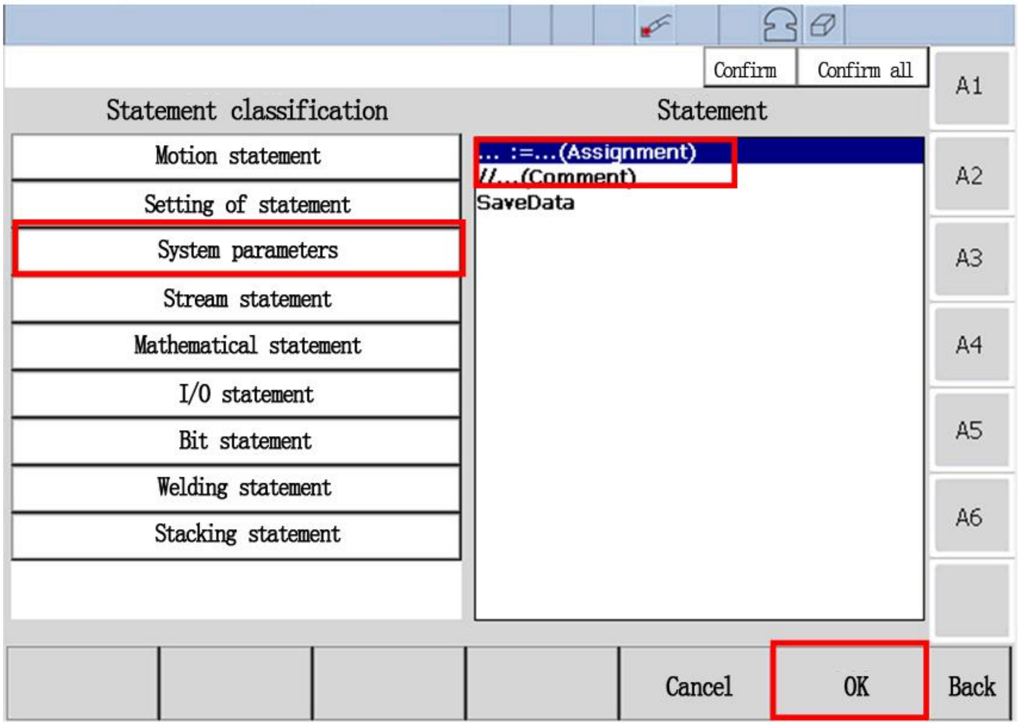


```
1: PTP(ap5,dyn0,ola0);
2: Lin(cp0,dyn0,olr0,ori0);
3: Circ(ap0,ap1,dyn0,ola0,ori0);
4: PTPRel(apd0,dyn0,olr0);
5: LinRel(cpd0,dyn0,olr0,ori0);
6: WaitTime(uint0);
EOF
```

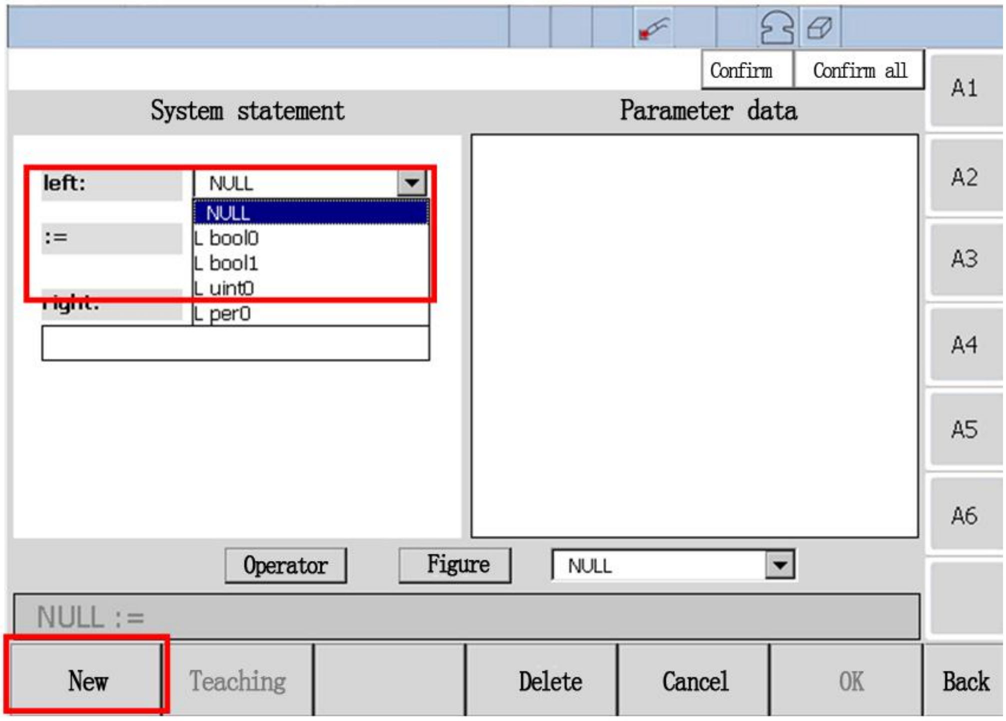
3.3 System parameters

3.3.1 Assignment statement

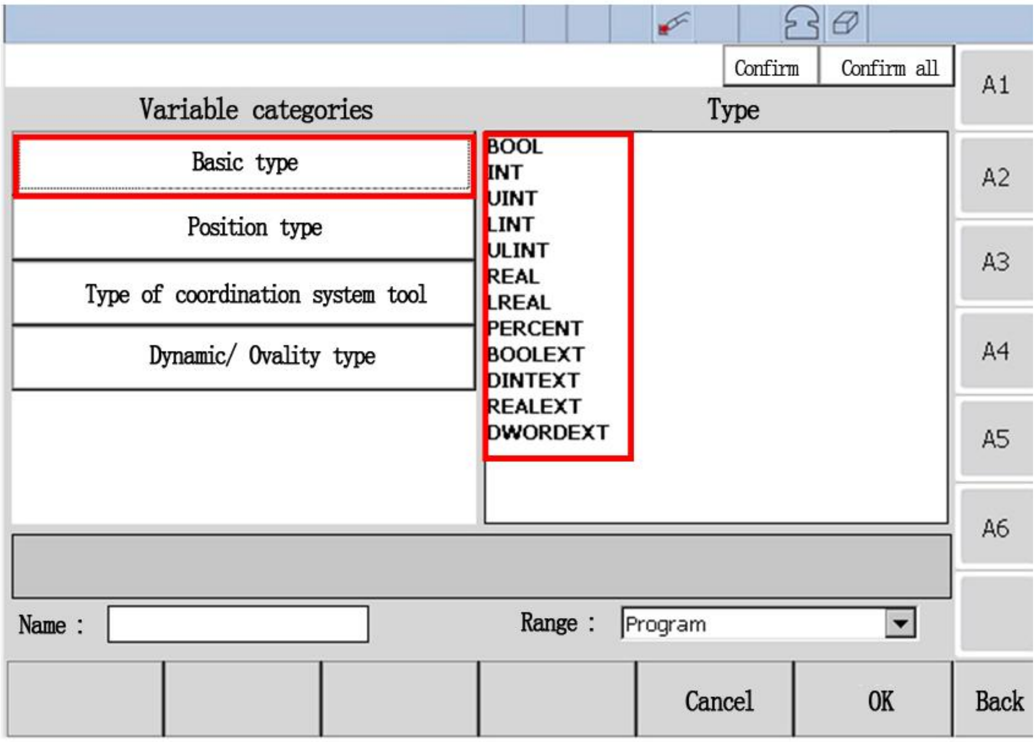
- Select the system function and choose an assignment statement at right...: = ... (Assignment). As the figure shows



- Click OK, select the left variable, and you can also create a new variable. As shown in the figure



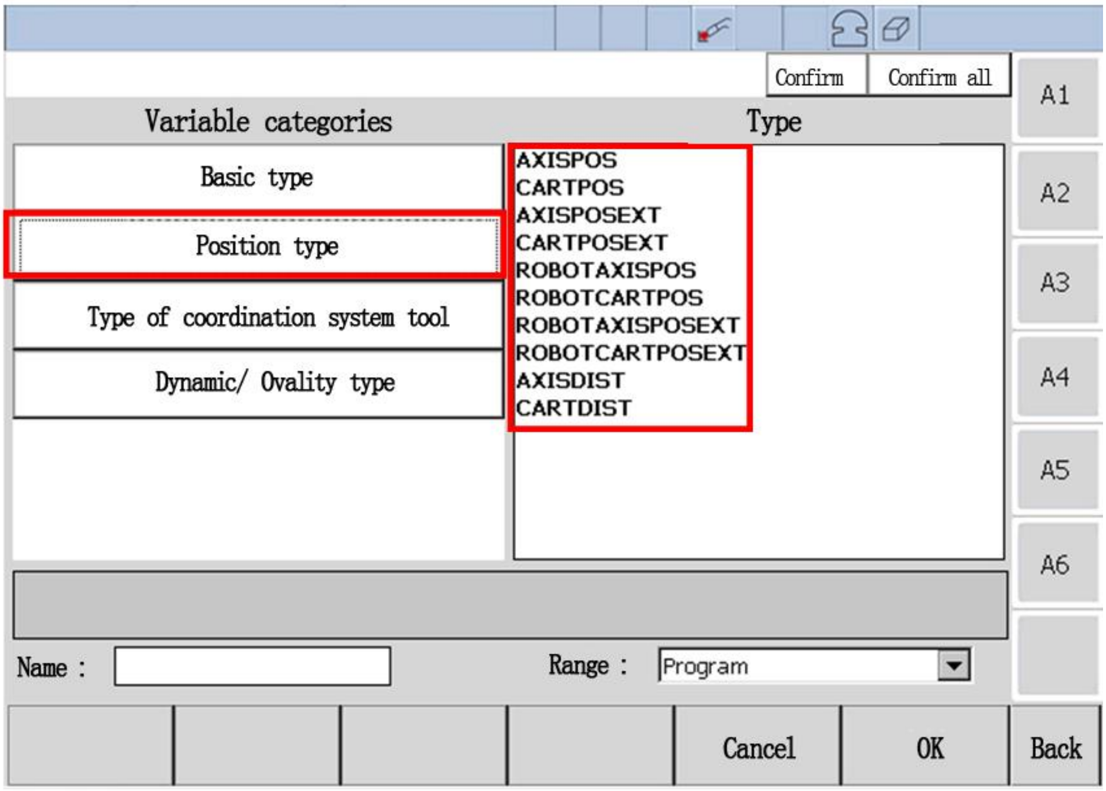
- After selecting to create a new one, there are basic types. As the figure shows



BOOL
INT
UINT
LINT
ULINT
REAL
LREAL
PERCENT
BOOLEXT
DINTEXT
REALEXT
DWORDEXT

➢ See the position data type in variable introductions for the details of type of right position.

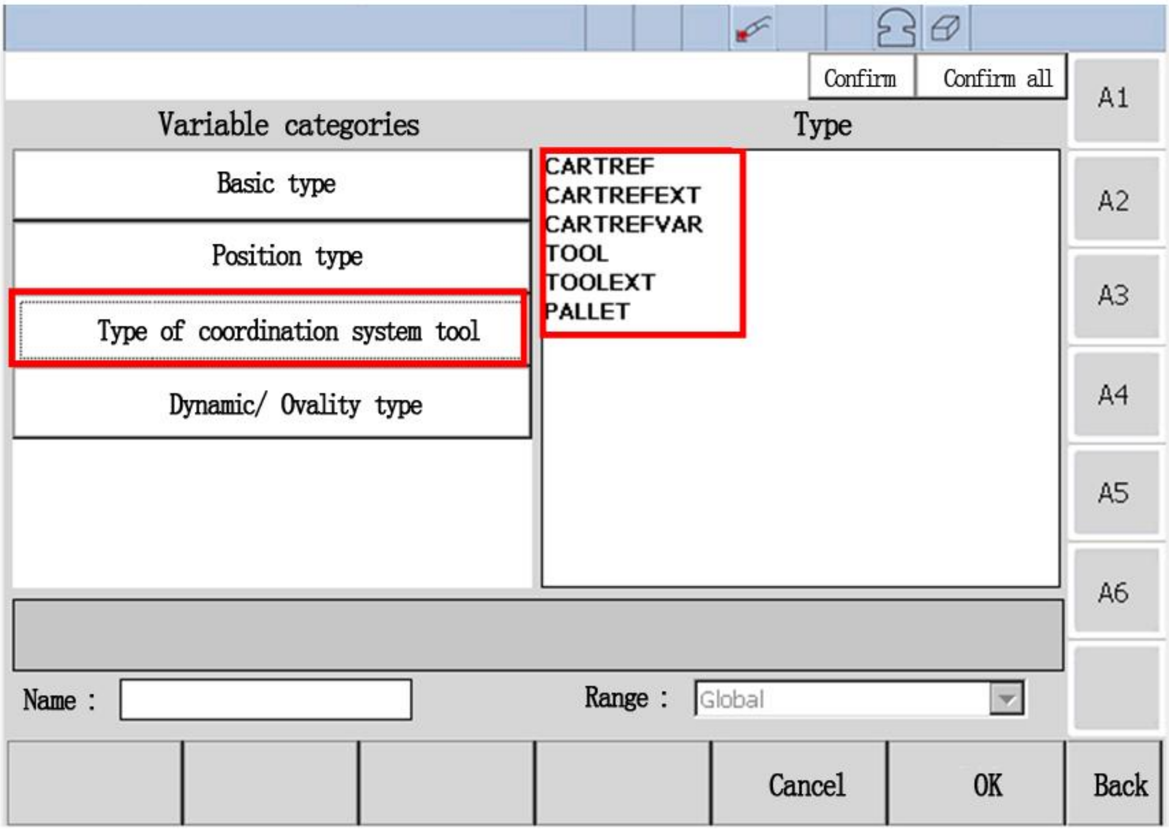
- Position type is shown in figure as:



AXISPOS
CARTPOS
AXISPOSEXT
CARTPOSEXT
ROBOTAXISPOS
ROBOTCARTPOS
ROBOTAXISPOSEXT
ROBOTCARTPOSEXT
AXISDIST
CARTDIST

➢ See the position data type in variable introductions for the details of type of right position.

● Tool types of coordinate system tool, as shown in figure



➢ Introduction of types of common coordinate system tools at right:

◆ CARTREF

The position of coordinate system against other reference coordinate systems, for which the variables consist of the X, Y, Z, A, B, C, as well as the name coordinate system that is used for reference, and X ~ C are DOUBLE type.

For example, CARTREF cr1 := {1.1 , 1.2 , 1.3 , 1.4 , 1.5 , 1.6, crs0}

◆ CARTREFEXT

External coordinate system, assigning with a value of UNIT type and indicating an index of m_RefExtern [8]. There is also a value of STRING type that represents the reference coordinate system. The coordinate system used for reference is only the fixed coordinate system.

For example, CARTREFEXT cre1 := {5, crs0}

◆ CARTREFVAR

External coordinate system, assigning with a value of UNIT type and indicating an index of m_RefVar [8]. There is also a value of STRING type that represents the reference coordinate system. The coordinate system used for reference is only the fixed coordinate system.

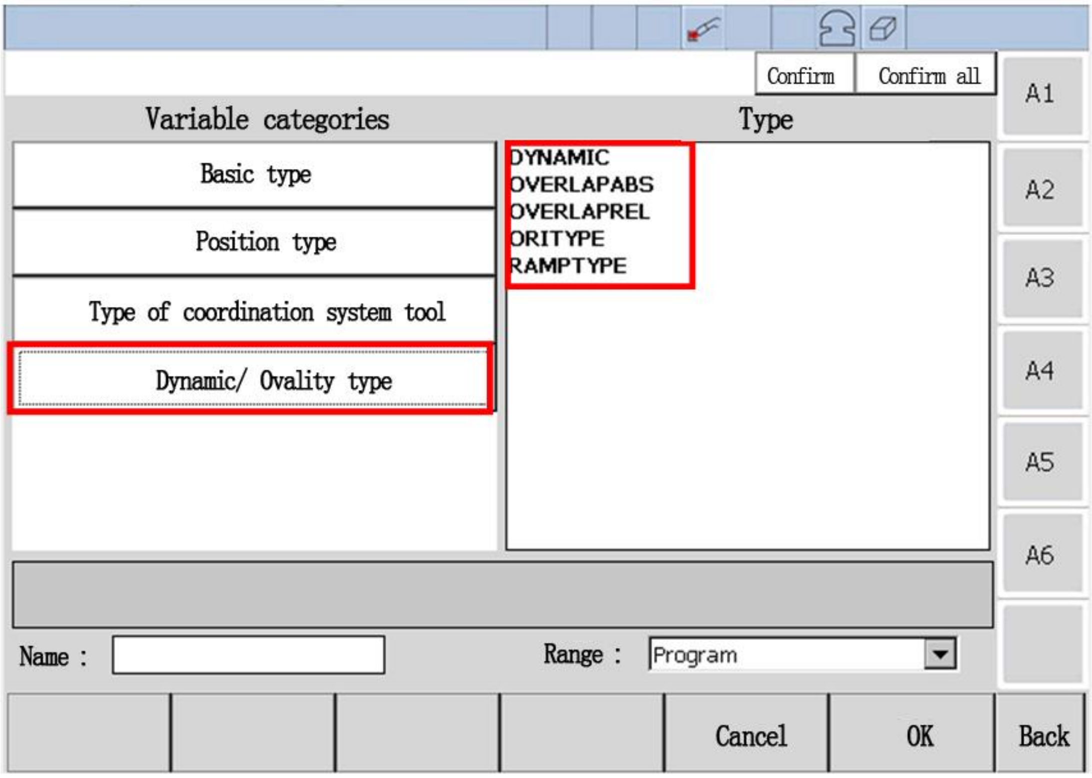
For example, CARTREFVAR crv1 := {5, crs0}

◆ TOOL

The transformation matrix representing the tool. It consist of X, Y, Z, A, B and C, all of which are DOUBLE type.

For example, TOOL t1 := {1.1, 1.2, 1.3,1.4,1.5,1.6}

● Type of DYNAMIC/ovality, as the figure shows.



➢ Introduction of right types

◆ DYNAMIC

It is composed by the position, rotation and joint velocity, acceleration, deceleration and jerk, totaling 12 quantities. It is based on the path type executed (for Cartesian or joint), and different amounts are used. 12 quantities are:

For example, DYNAMIC dyn1 := {1000 , 2000 , 2000 , 5000 , 100 , 200 , 200 , 500 , 80 , 80 , 80 , 80}

◆ OVLREL

It indicates the percentage of ovality, and is an amount of INT type, representing the percentum.

For example, OVERLAPREL ovl1 := 50

◆ OVLABS

The variable type includes four members (one less than the keba)

PosDist	DOUBLE	TCP position oval distance (mm)
oriDist	DOUBLE	TCP posture oval angle (°)
linAxDist	DOUBLE	Oval distance of linear axis (mm)
rotAxDist	DOUBLE	Oval angle of rotation axis (°)

For example, OVERLAPABS ovl2 := {1.1, 1.2, 1.3, 1.4}

◆ ORITYPE

It indicates the posture on the path and enumerated types. The value is only three.

eVAR: // Make quaternion posture interpolation based on the posture information as demonstrated

eCONST: // Keep the posture of starting position always

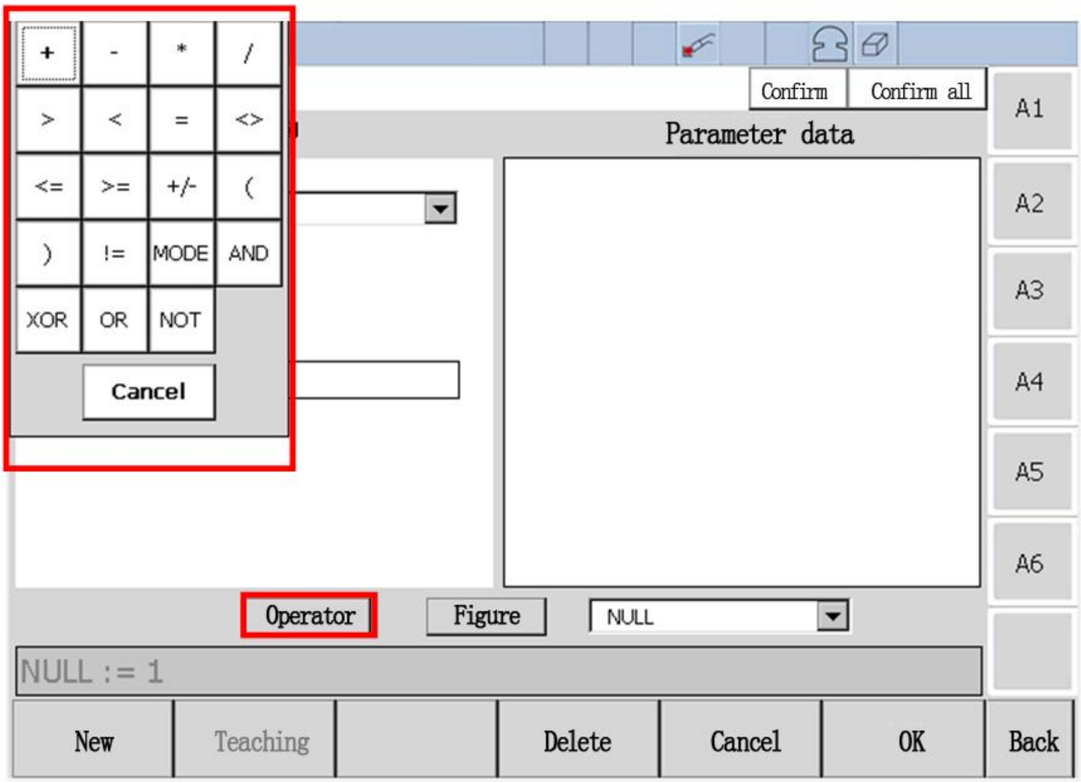
eCIRCLE: // Keep the fixed posture with the round plane for circle interpolation

For example, ORITYPE oritype := eVAR

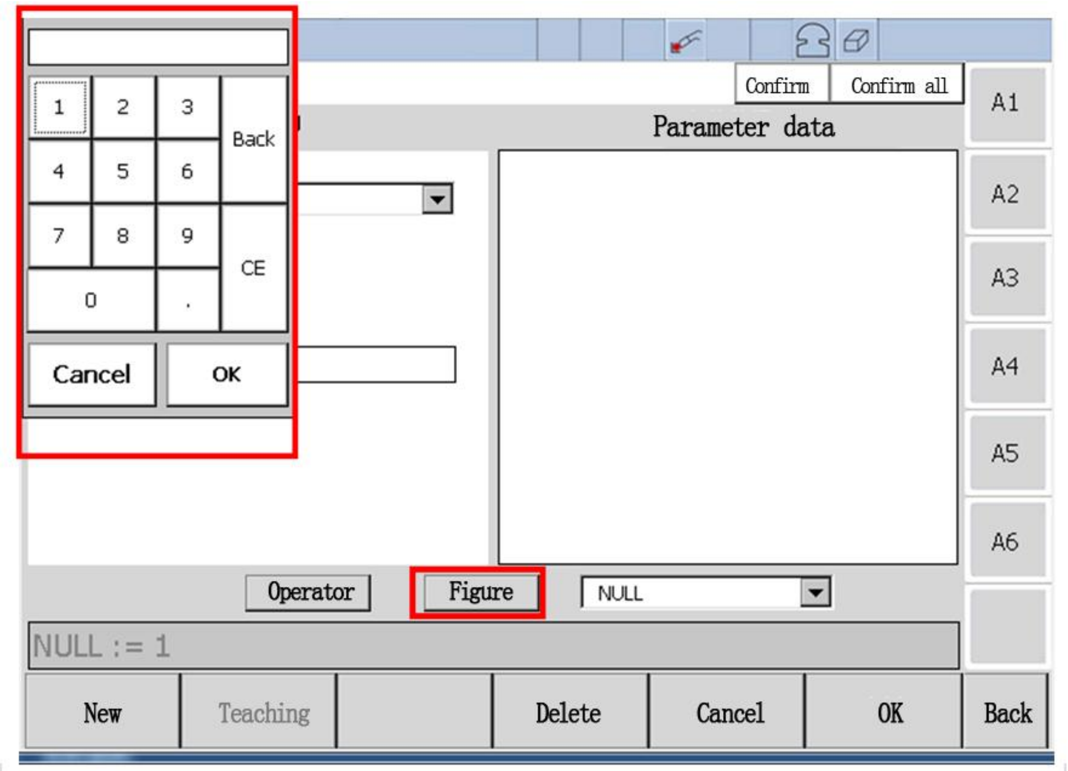
◆ RAMPTYPE

It include three types, the Trap, Sshape and Ssine;

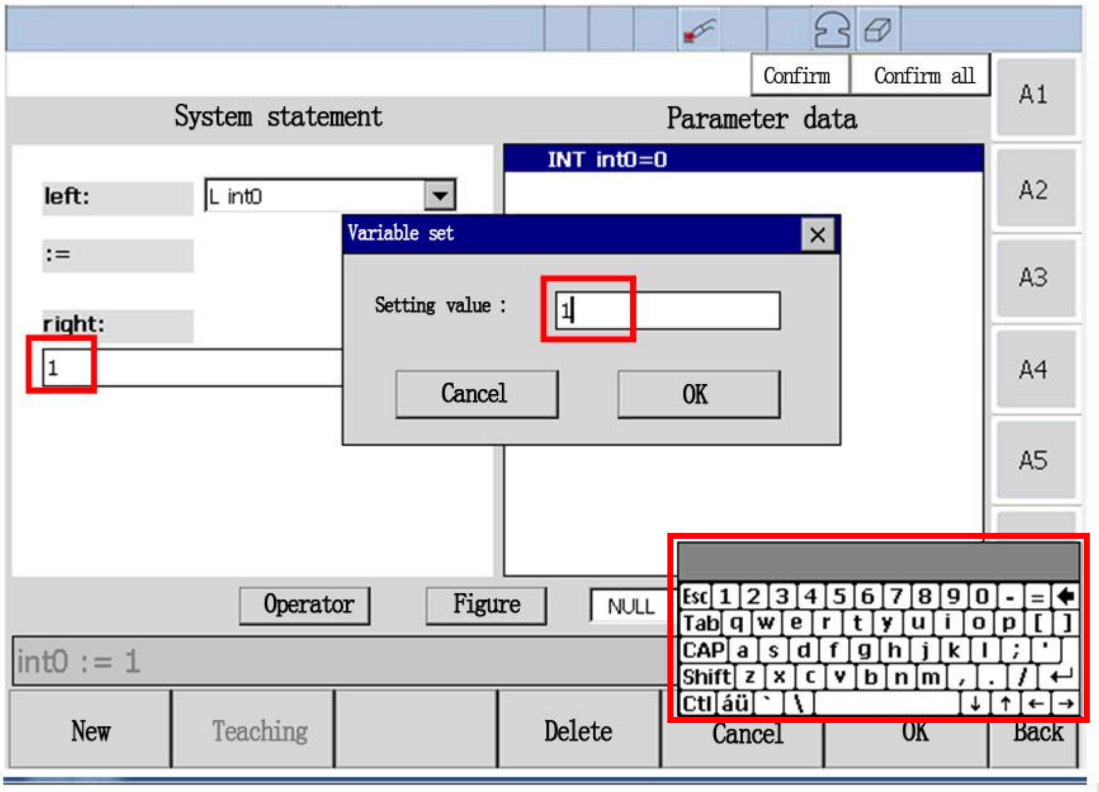
- If you choose the int in basic type, and it appears as the figure shows when clicking the operator.



- Click the figures, and it shown as the figure shows.



- Right variable settings, as the figure shows



INT int0=0

left: L int0

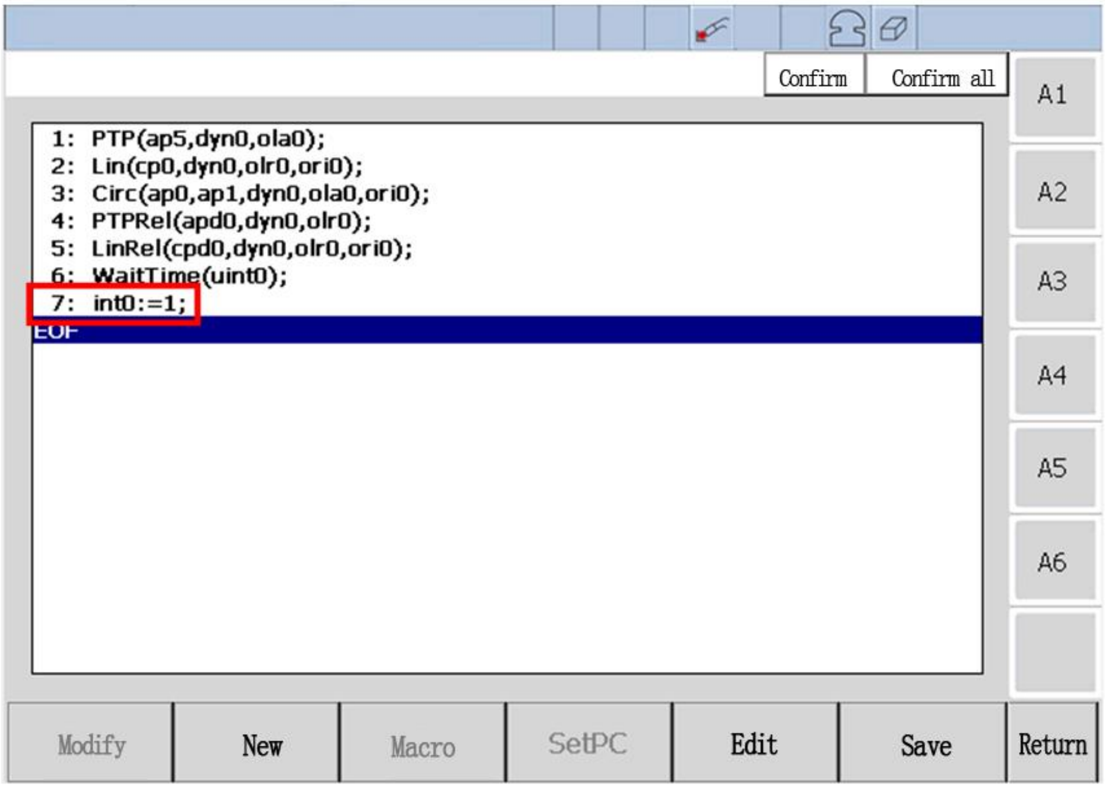
:=

right: 1

Setting value : 1

int0 := 1

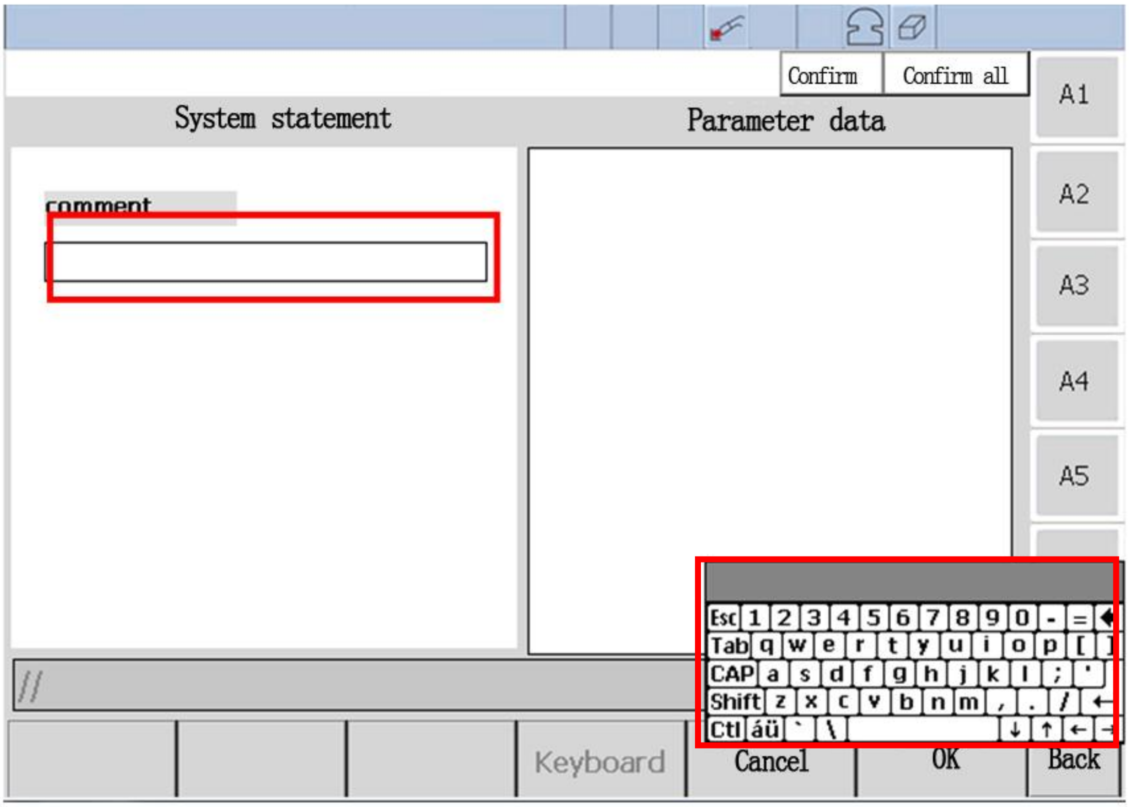
- Click OK, as shown in the figure



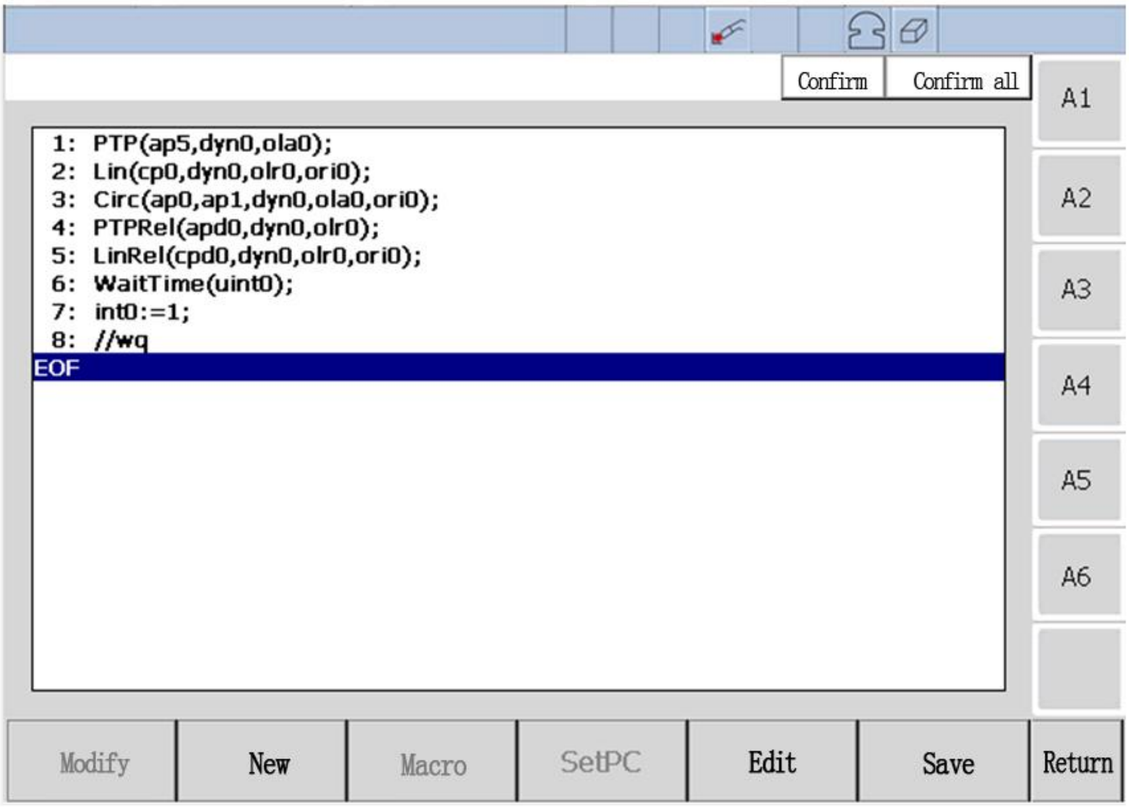
```
PTP(ap5,dyn0,ola0);
Lin(cp0,dyn0,olr0,ori0);
Circ(ap0,ap1,dyn0,ola0,ori0);
PTPRel(apd0,dyn0,olr0);
LinRel(cpd0,dyn0,olr0,ori0);
WaitTime(uint0);
int0:=1;
EOF
```

3.3.2 Comment statement

- In the system function, upon selecting // ... comment statement, a comment is entered from the keyboard, such as enter wq. As the figure shows.

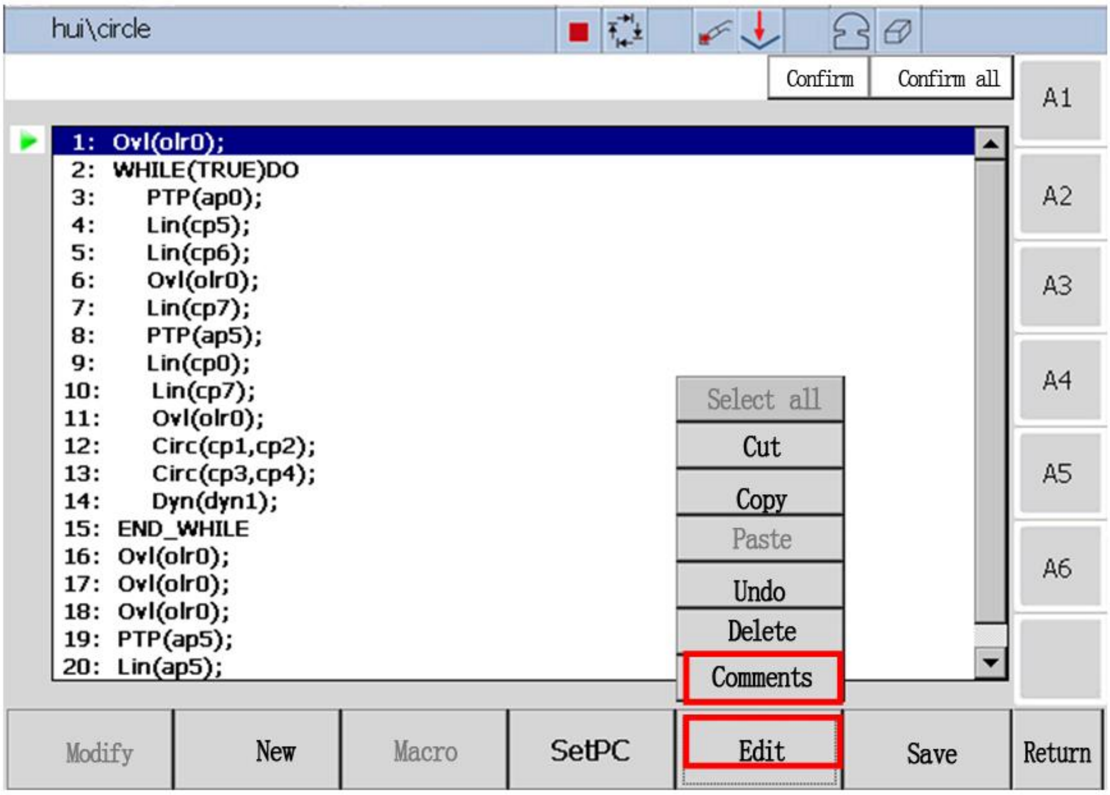


- Click OK and there displays the comment statements. As the figure shows



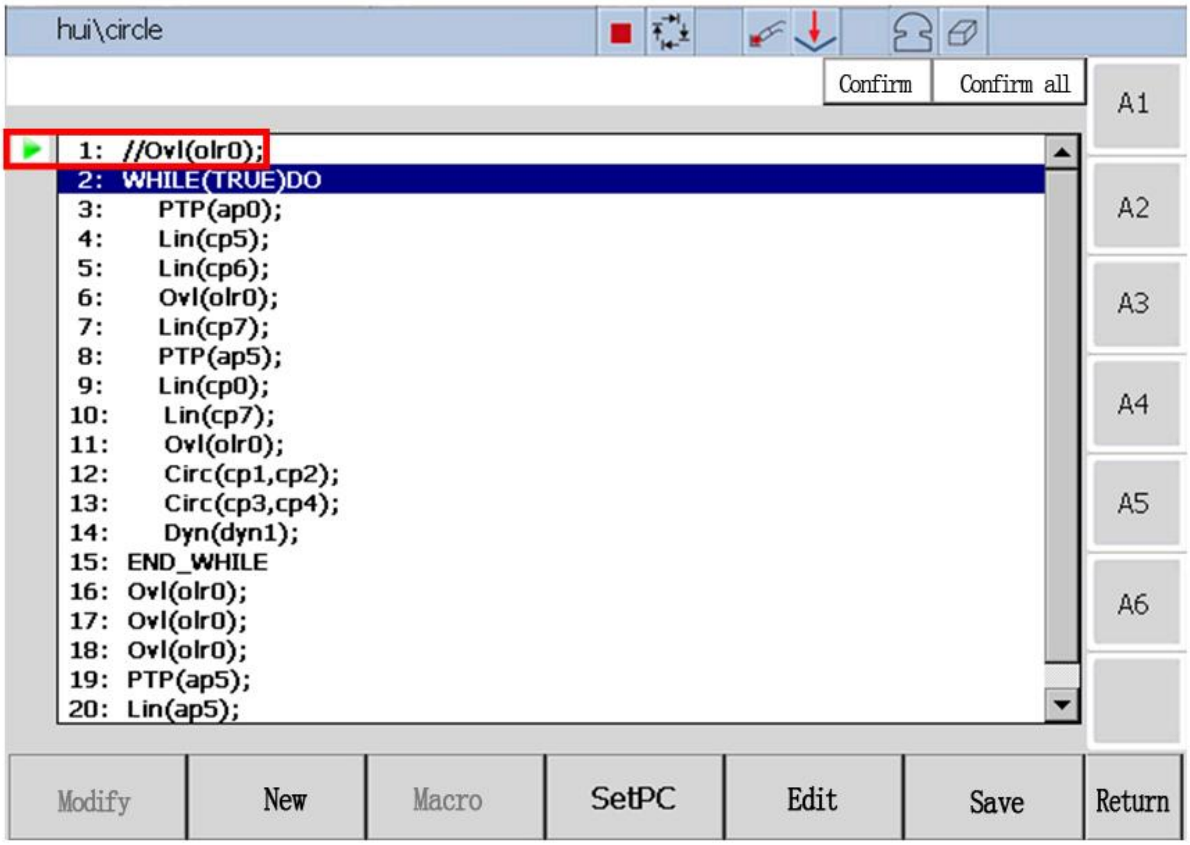
```
1: PTP(ap5,dyn0,ola0);
2: Lin(cp0,dyn0,olr0,ori0);
3: Circ(ap0,ap1,dyn0,ola0,ori0);
4: PTPRel(apd0,dyn0,olr0);
5: LinRel(cpd0,dyn0,olr0,ori0);
6: WaitTime(uint0);
7: int0:=1;
8: //wq
EOF
```

- In addition to the new statement, you may also click on "Comments" in the "Edit" in the program, as shown in figure below



```
Ovl(olr0);
WHILE(TRUE)DO
    PTP(ap0);
    Lin(cp5);
    Lin(cp6);
    Ovl(olr0);
    Lin(cp7);
    PTP(ap5);
    Lin(cp0);
    Lin(cp7);
    Ovl(olr0);
    Circ(cp1,cp2);
    Circ(cp3,cp4);
    Dyn(dyn1);
END_WHILE
Ovl(olr0);
Ovl(olr0);
Ovl(olr0);
PTP(ap5);
Lin(ap5);
```

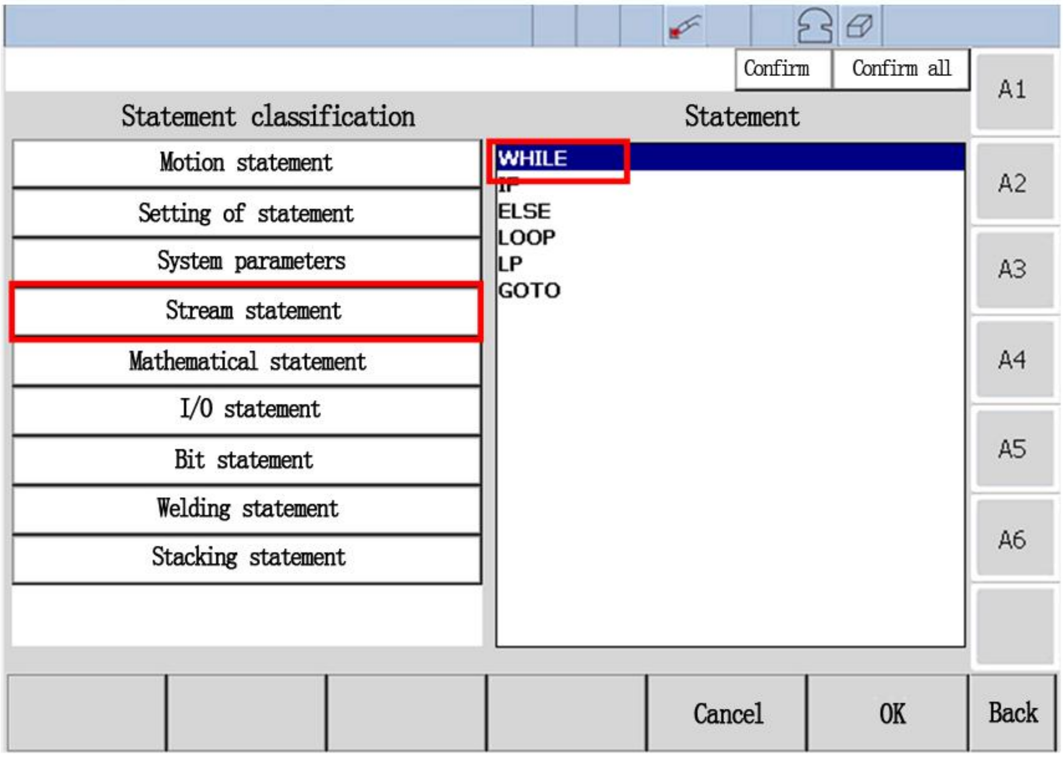
- After making comment to the statement, it appears as figure below shows.



```
//Ovl(olr0);
WHILE(TRUE)DO
    PTP(ap0);
    Lin(cp5);
    Lin(cp6);
    Ovl(olr0);
    Lin(cp7);
    PTP(ap5);
    Lin(cp0);
    Lin(cp7);
    Ovl(olr0);
    Circ(cp1,cp2);
    Circ(cp3,cp4);
    Dyn(dyn1);
END_WHILE
Ovl(olr0);
Ovl(olr0);
Ovl(olr0);
PTP(ap5);
Lin(ap5);
```

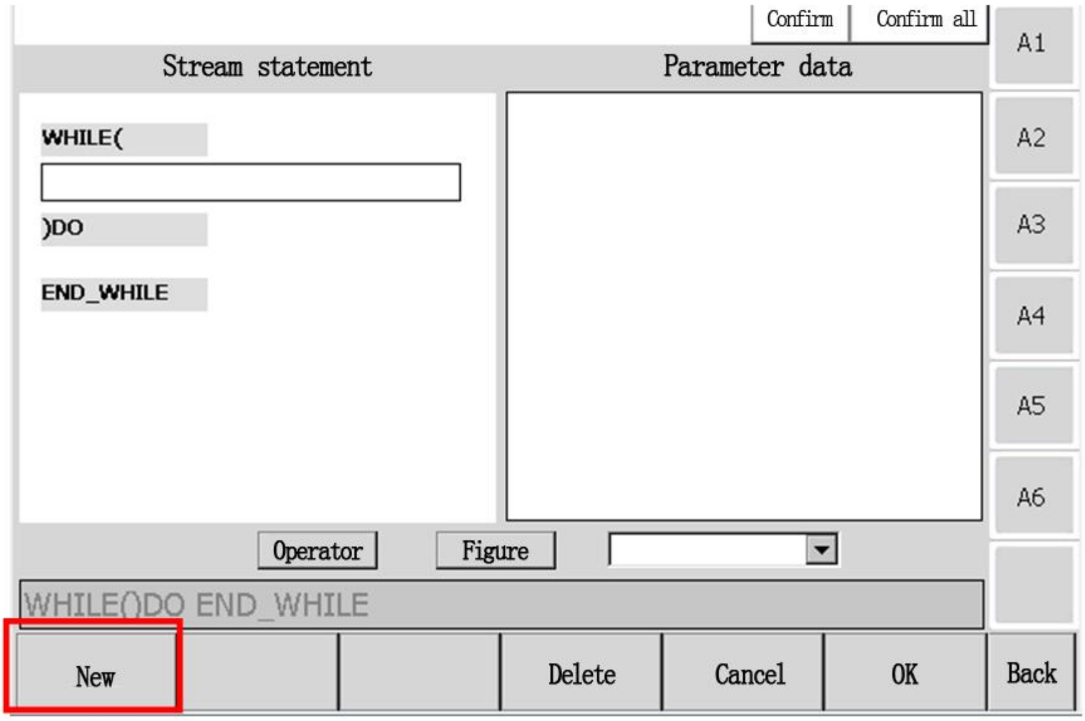
3.4 Stream statement

- Select the stream data, and it appears as the figure shows.

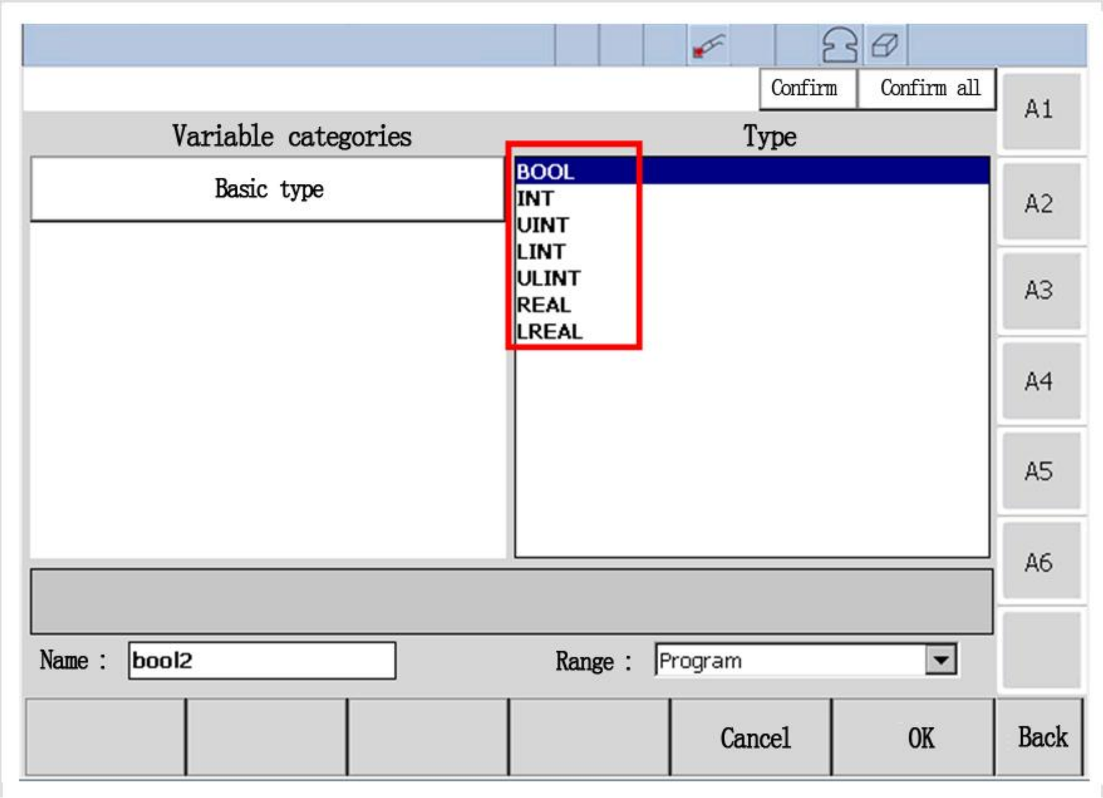


3.4.1 While statement

- When selecting the WHILE, and it appears as the figure shows.



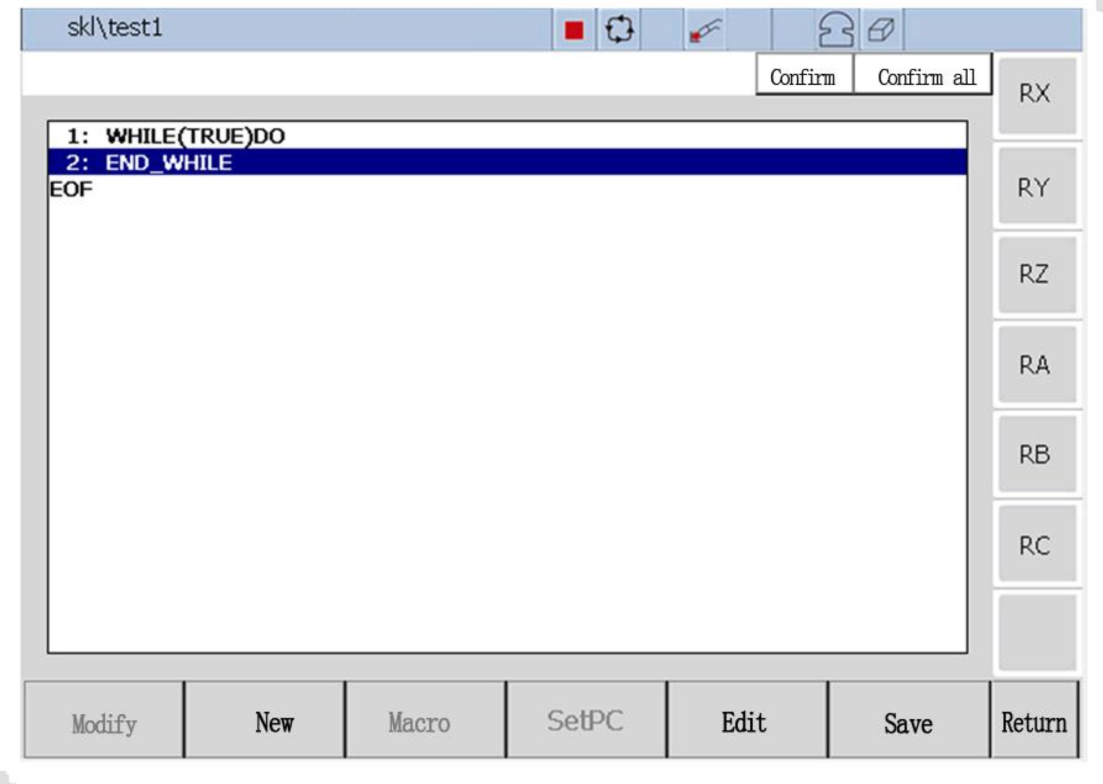
- Create the basic types. As the figure shows



Name : bool2

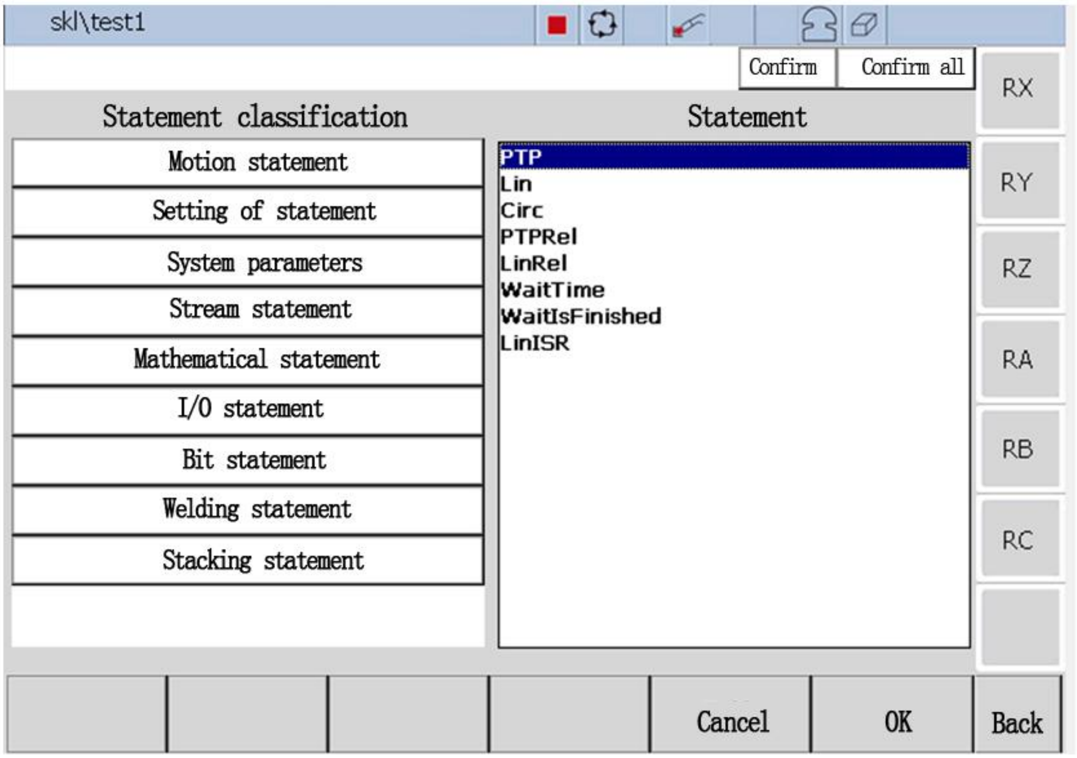
Range : Program

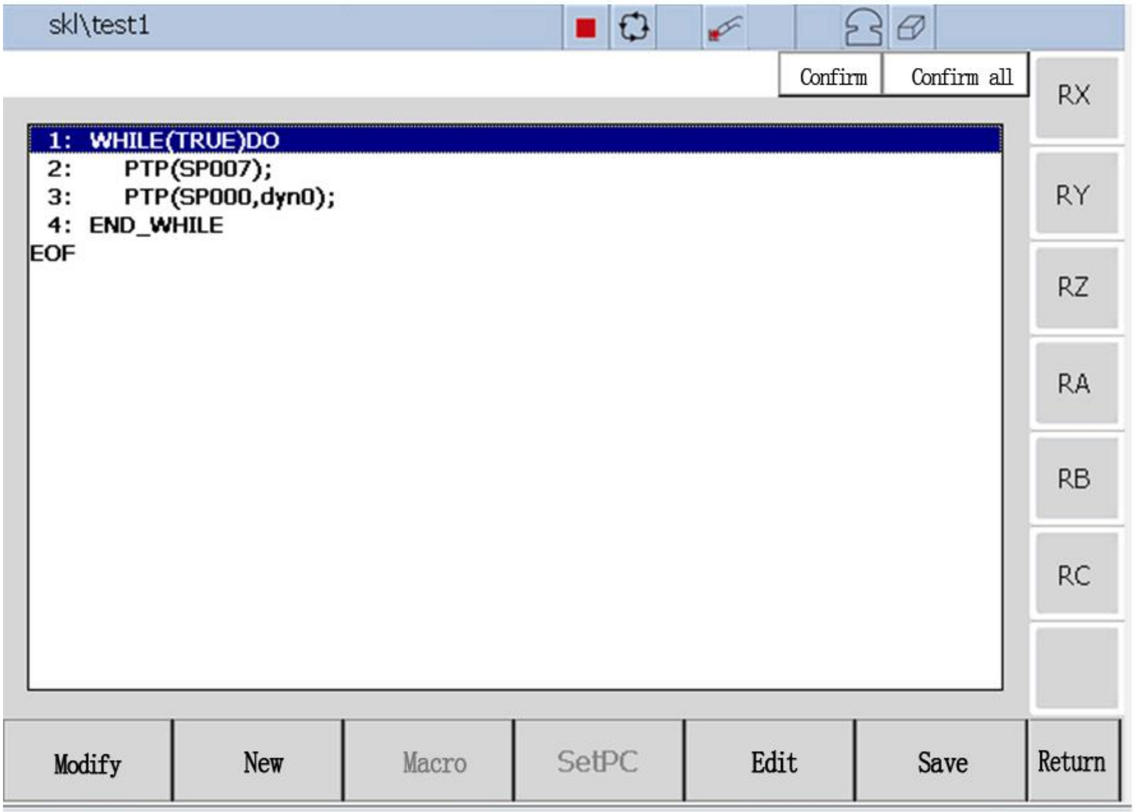
- ➢ Type usage can refer to the assignment sentence in the system parameters.
- ● If you choose TRUE, it appears as the figure below shows



```
1:  WHILE(TRUE)DO
2:  END_WHILE
EOF
```

- Write statement between "while" and "endwhile", and two motion statements will be written here.





```
WHILE(TRUE)DO
    PTP(SP007);
    PTP(SP000,dyn0);
END_WHILE
EOF
```

- For example,
- WHILE ( )DO … END_WHILE

This instruction is used to repeat an instruction sequence when the conditions are satisfied. The usage is as follows:

```
WHILE(TURE)DO
PTP（SP007）;
PTP（SP000，dyn0）;
END_WHILE
```

The program acts as follows: The robot makes reciprocating motion between SP007 and SP000.

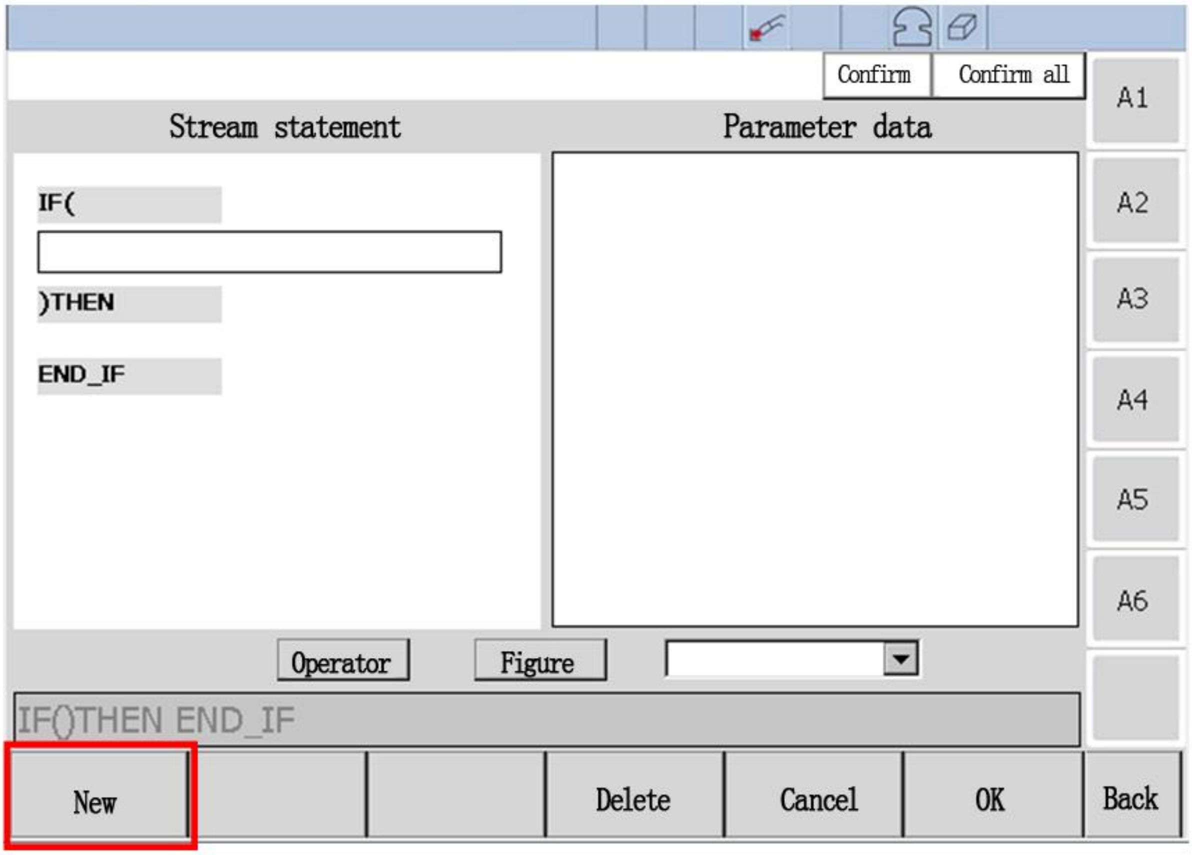
- Usage: The judgment condition behind WHILE need to return the BOOL type. Behind DO there is an instruction sequence, with no number restriction; END_WHILE must be used at the end.

```
//WHILE standard format
WHILE ( EXPRESS1 )DO
STATEMENT;
END_WHILE
```

- Format Description: The three keywords required by the statement are "WHILE" "DO" "END_WHILE". A parentheses is followed behind the WHILE, and the EXPRESS 1 in that parentheses is an expression, and the expression structure consists of three parts: Working equations1+ relation character + working equations 2, in which, the expression can be a single variable or an arithmetic formula, e.g., a, a + b * c and the like. Behind the parentheses is "DO", the content of next line is one or more lines of statements that can either be the assignment statements, or the motion instruction, or other statements. The last line uses "END_WHILE" string as the end of the entire statement.

3.4.2 If statement

- Selecting IF, as the figure shows



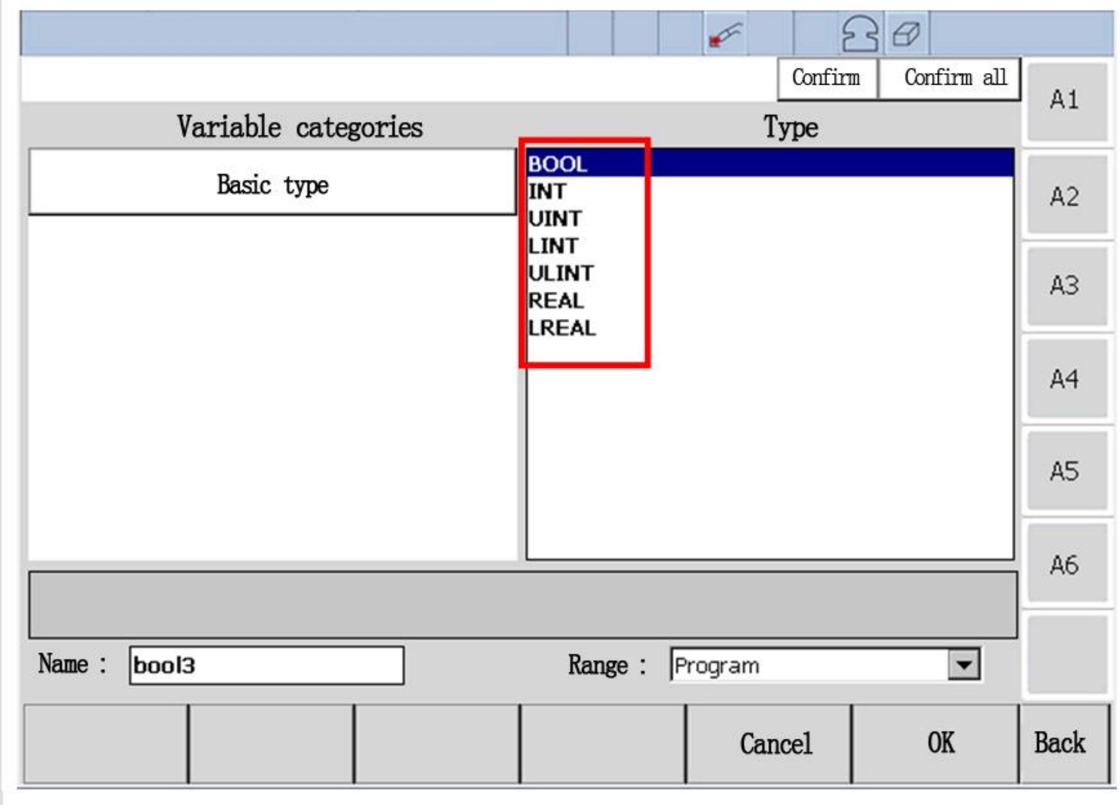
IF(

)THEN

END_IF

IF()THEN END_IF

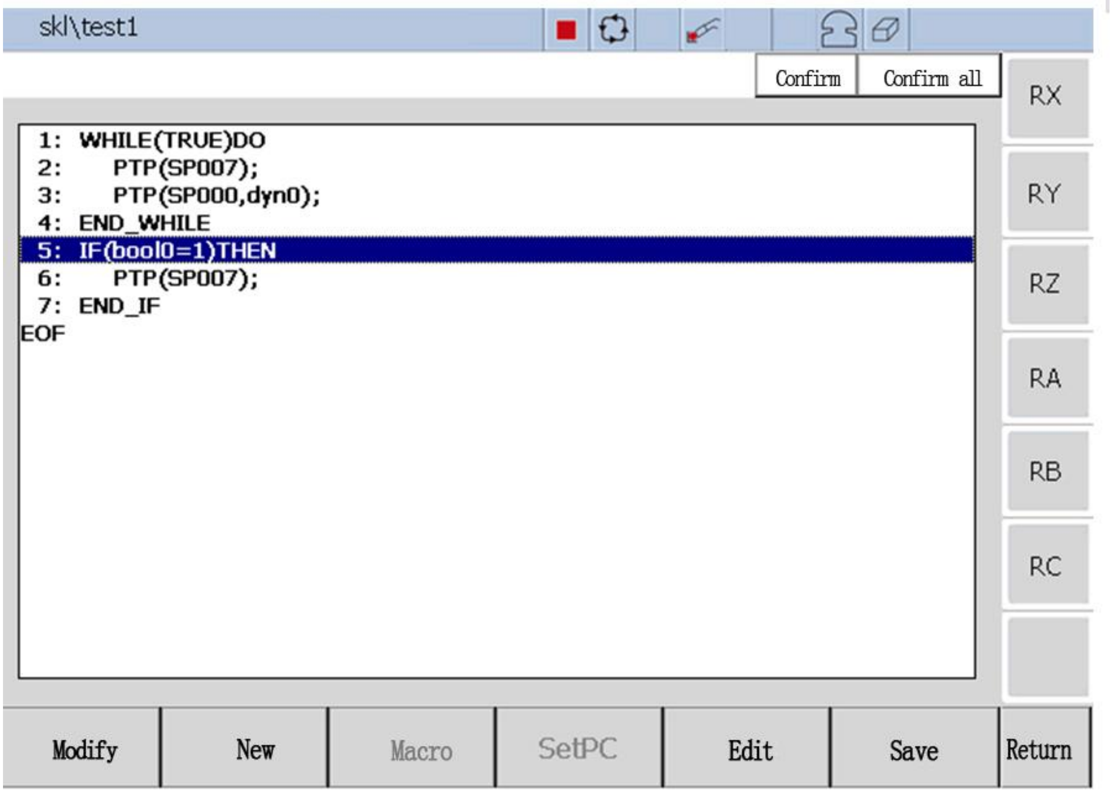
- Create variable UINT, as the figure shows



Name： bool3

Range： Program

- For example, create the bool 0 variable, and click OK. Enter a motion statement, as shown in Figure



```
WHILE(TRUE)DO
    PTP(SP007);
    PTP(SP000,dyn0);
END_WHILE
IF(bool0=1)THEN
    PTP(SP007);
END_IF
EOF
```

➢ The instruction control program flows into the conditional branch, and the example is given as follows:

IF(bool0=1)THEN

PTP(SP007);

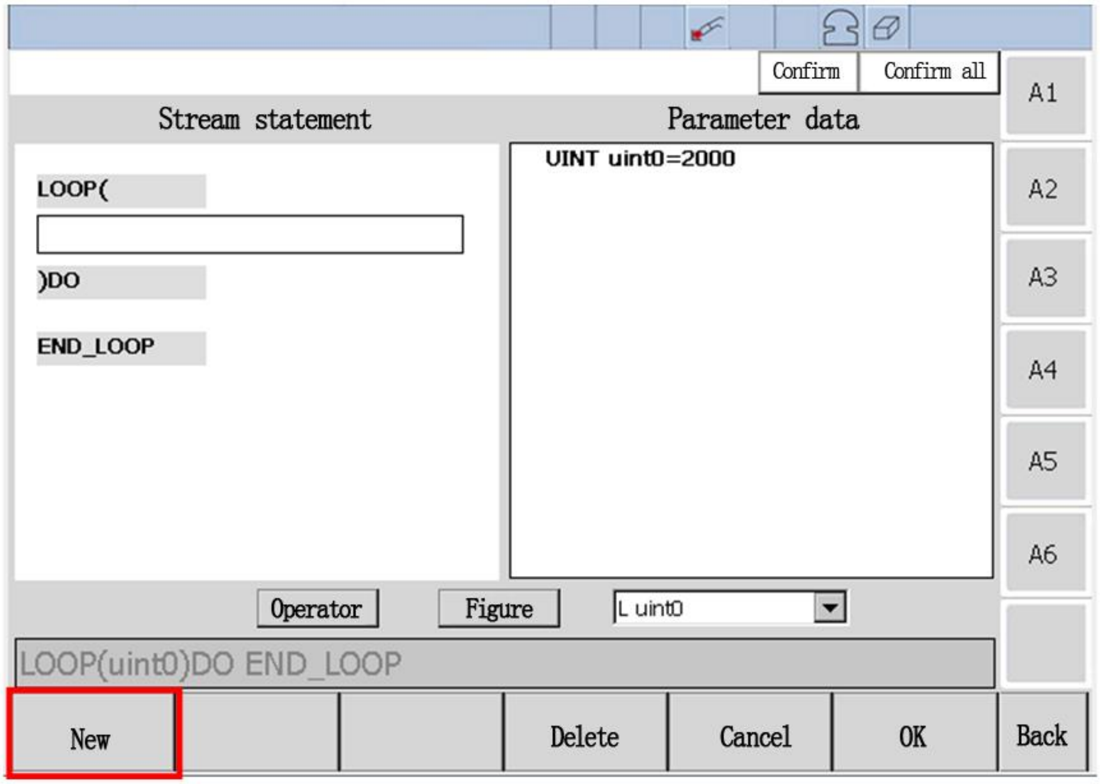
END_IF

The program acts as follows: When bool0 is 1, the robot moves to the SP007.

➢ Usage: The determining conditions behind IF need to return the BOOL type, and there's no limitation for the THEN, ELSEIF, ELSE as well as number of their subsequent instructions; each IF instruction must use the keyword END_IF.

3.4.3 Loop statement

- When selecting LOOP, the particulars is shown in figure



LOOP(

)DO

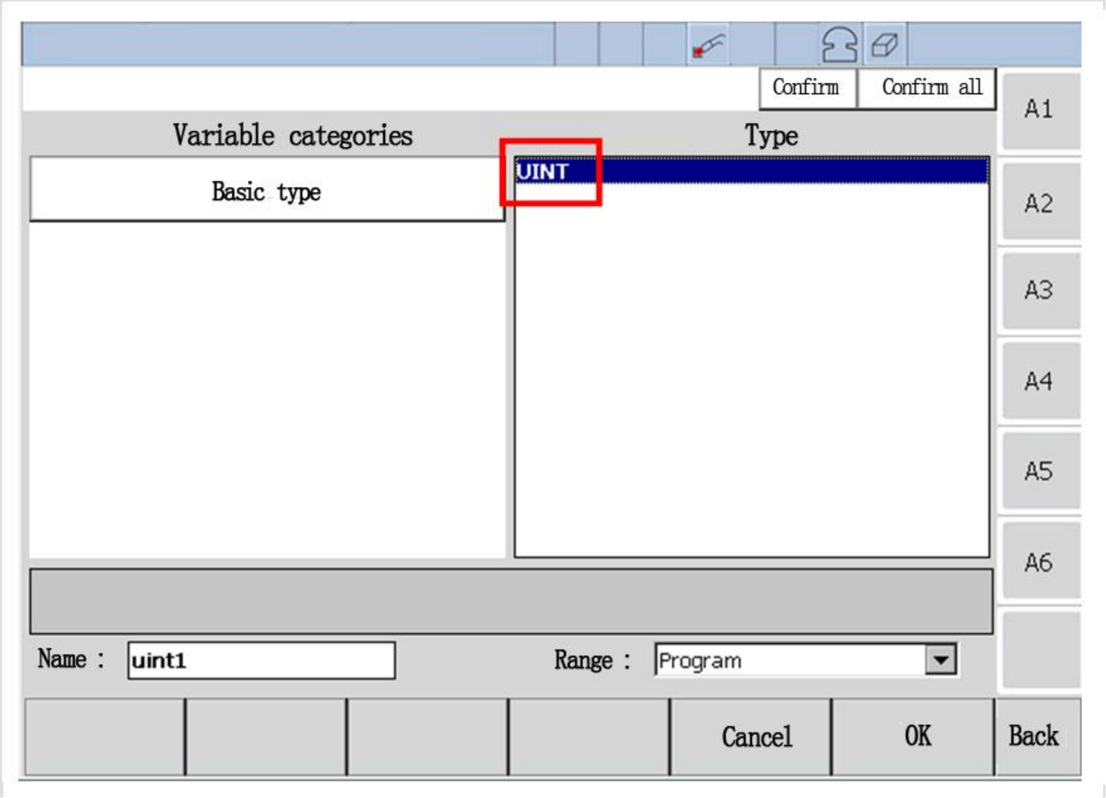
END_LOOP

UINT uint0=2000

L uint0

LOOP(uint0)DO END_LOOP

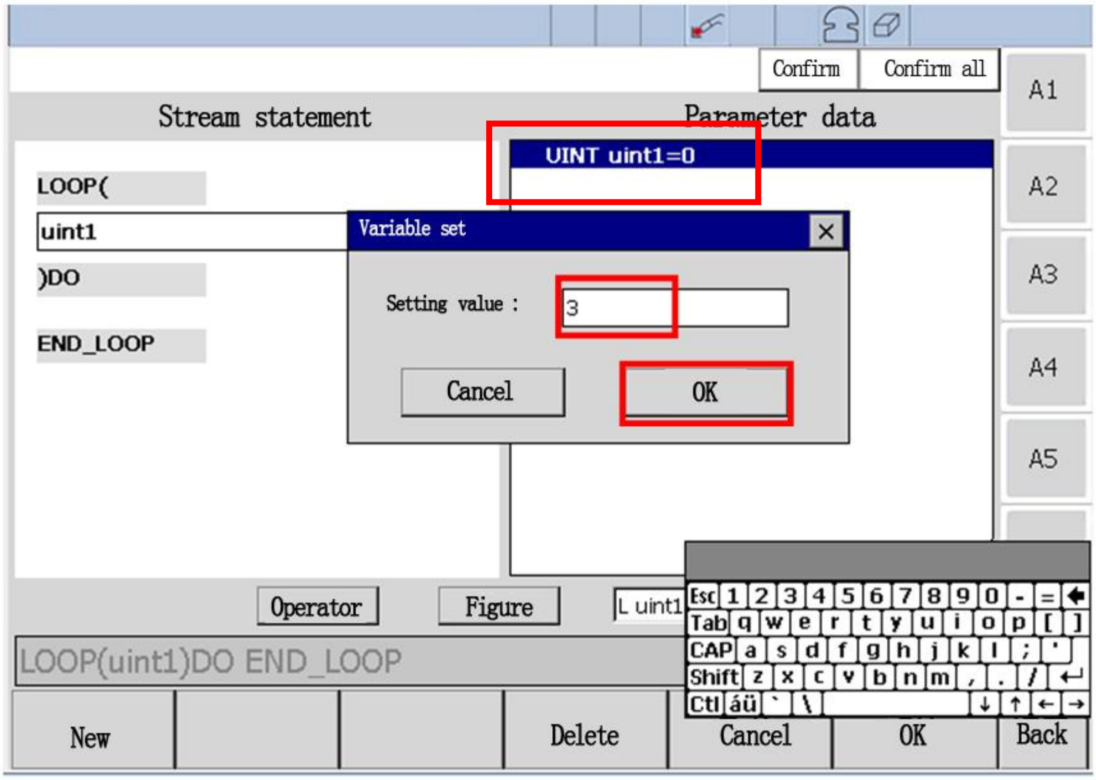
- Create variable UINT, as the figure shows



Name : uint1

Range : Program

- Input the set value by keyboard. As the figure shows



```
LOOP(
uint1
)DO
END_LOOP
```

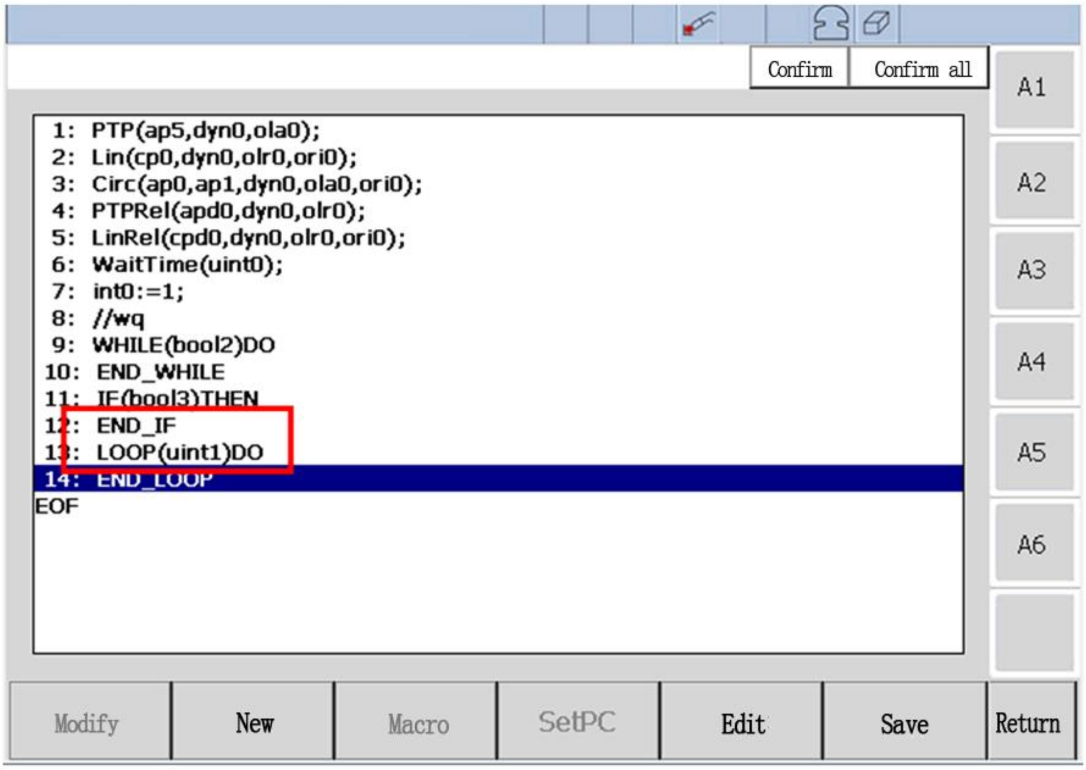
UINT uint1=0

Setting value : 3

L uint1

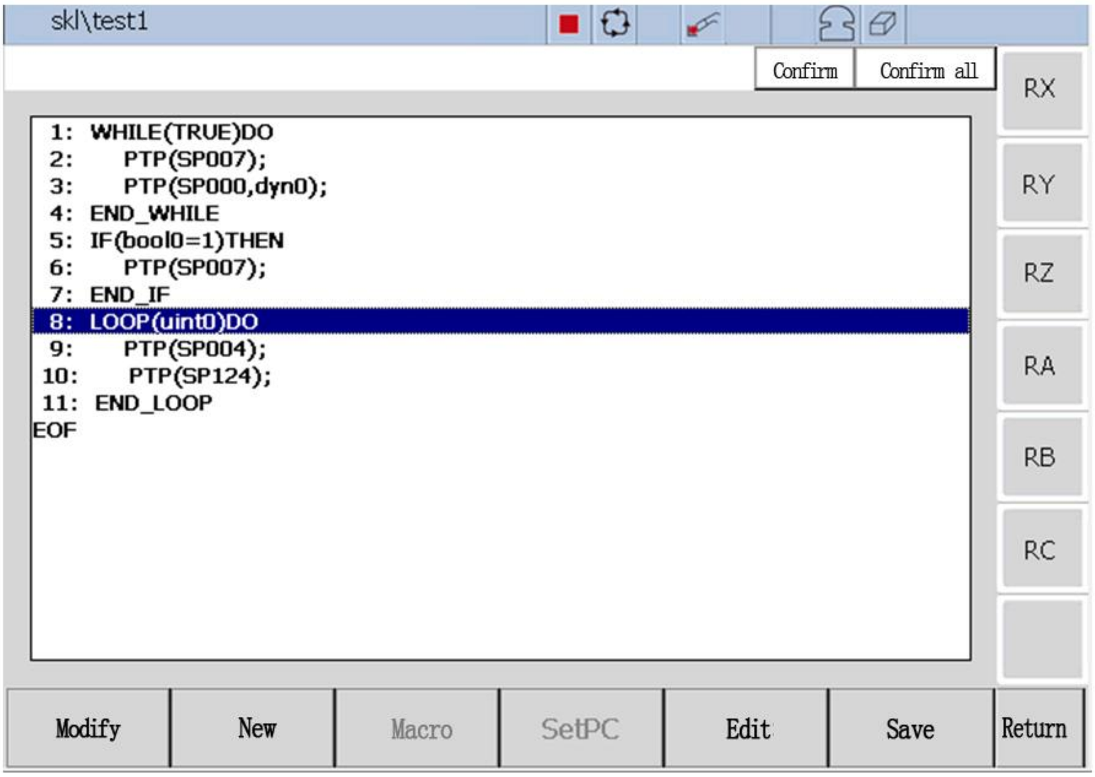
LOOP(uint1)DO END_LOOP

- Click OK. As the figure shows



```
PTP(ap5,dyn0,ola0);
Lin(cp0,dyn0,olr0,ori0);
Circ(ap0,ap1,dyn0,ola0,ori0);
PTPRel(apd0,dyn0,olr0);
LinRel(cpd0,dyn0,olr0,ori0);
WaitTime(uint0);
int0:=1;
//wq
WHILE(bool2)DO
END_WHILE
IF(bool3)THEN
END_IF
LOOP(uint1)DO
END_LOOP
EOF
```

- For example, as the following procedures describe:



```
WHILE(TRUE)DO
    PTP(SP007);
    PTP(SP000,dyn0);
END_WHILE
IF(bool0=1)THEN
    PTP(SP007);
END_IF
LOOP(uint0)DO
    PTP(SP004);
    PTP(SP124);
END_LOOP
EOF
```

- LOOP ( )DO … END_LOOP The instruction is used to execute the given number for the instruction sequence, and the usage is as follows:

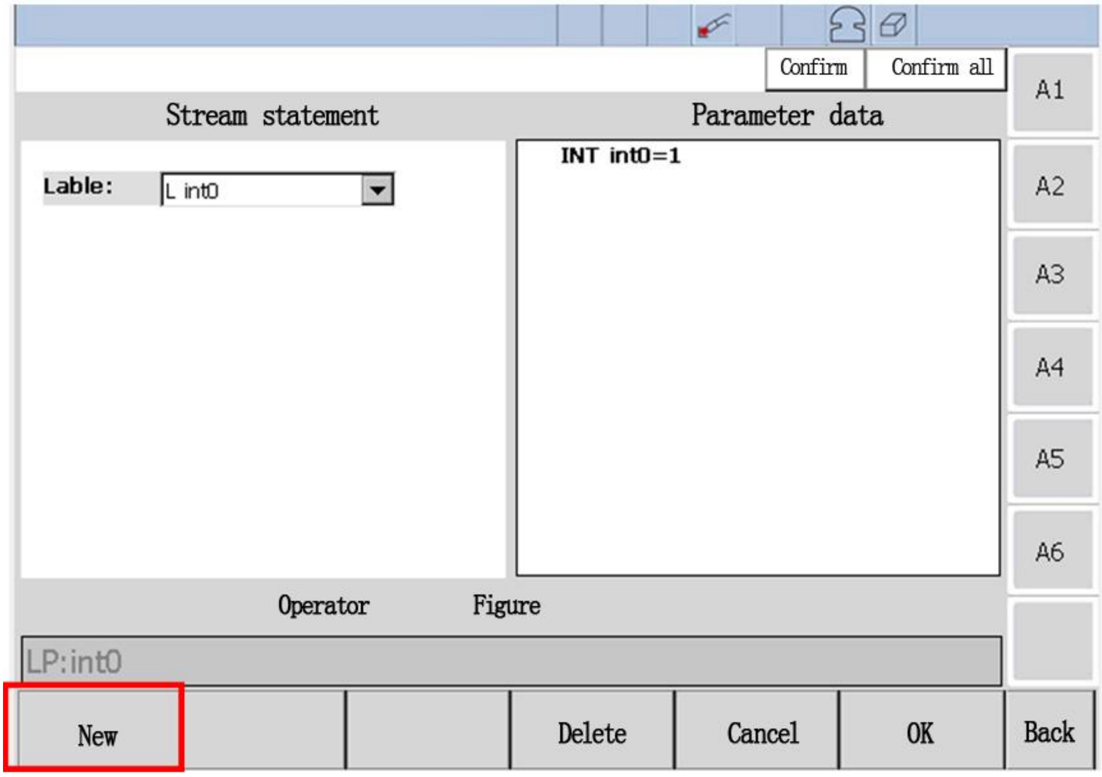
```
LOOP(uint0)DO
PTP（SP004）；
PTP（SP124）；
END_LOOP
```

The program acts as follows: Run the two statements in the LOOP for uint0 times.

- Format Description: The number of cycles can only be replaced with the variables, but cannot directly enter the figures.

3.4.4 Lp statement

- Select LP, as the figure shows



Stream statement

Parameter data

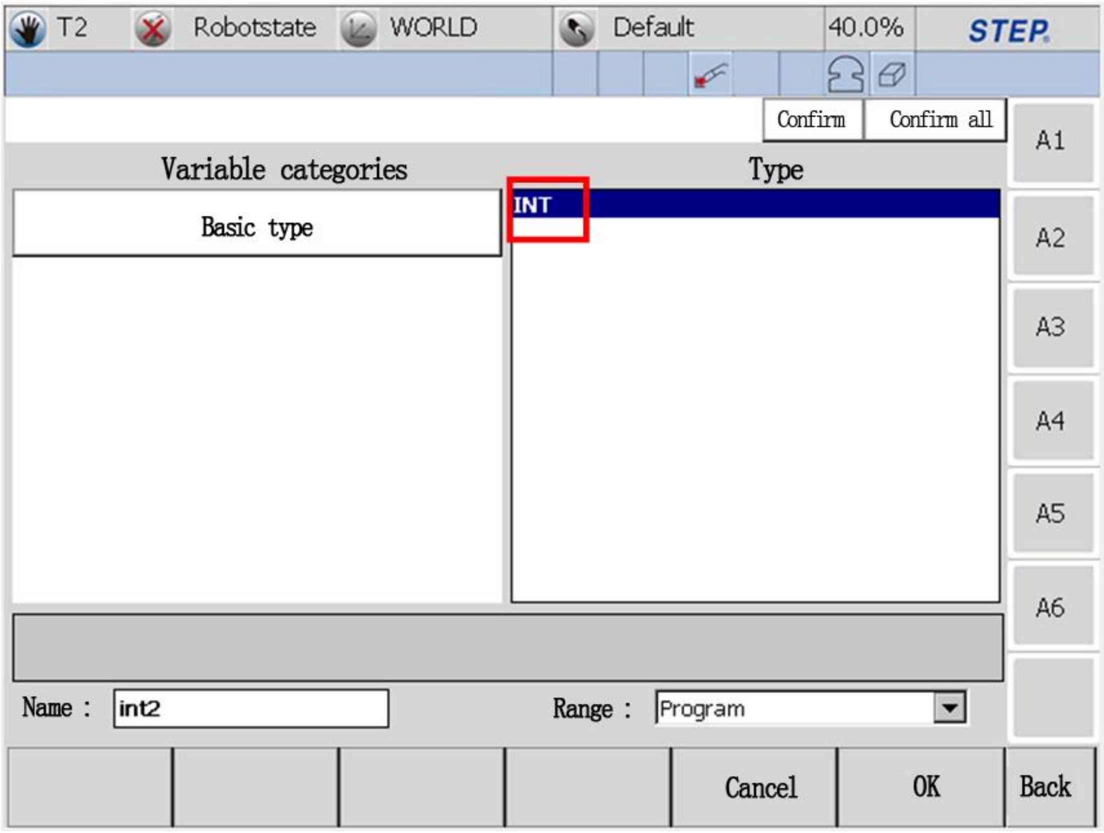
Lable: L int0

INT int0=1

Operator Figure

LP:int0

- Create new variable INT type, as the figure shows



T2 Robotstate WORLD Default 40.0% STEP

Variable categories

Type

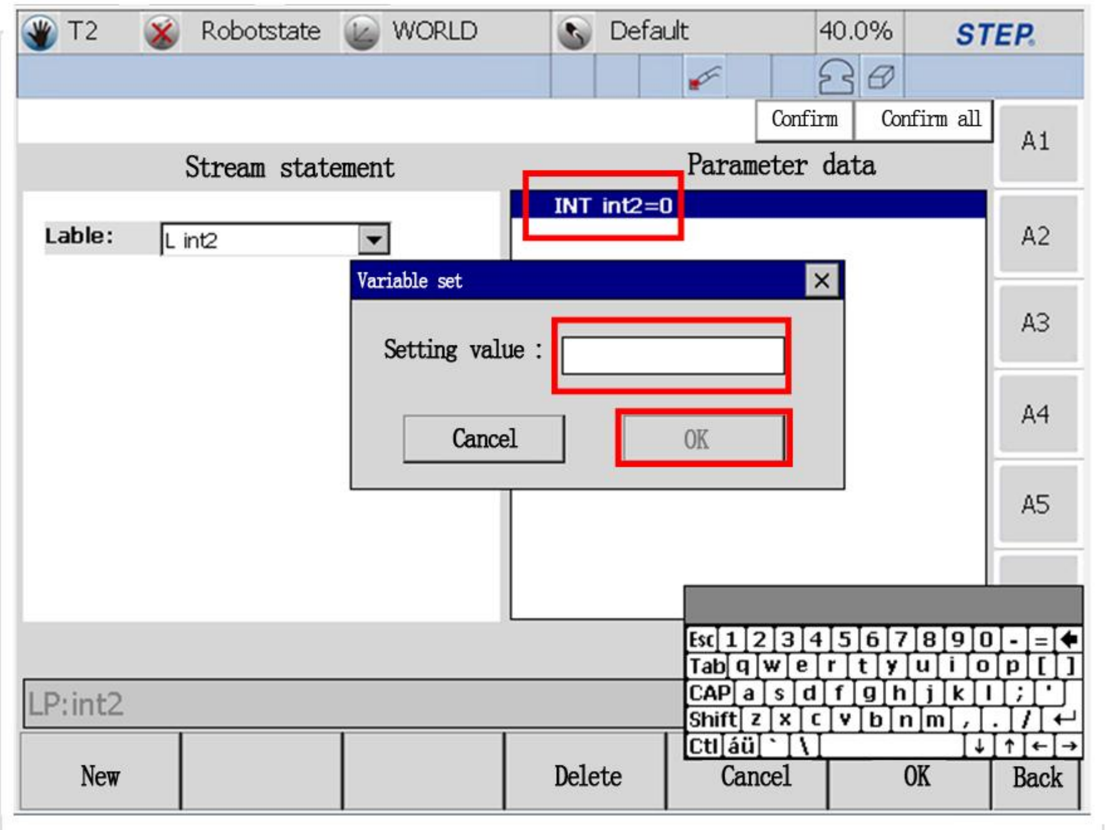
Basic type

INT

Name : int2

Range : Program

- Set the value, as the figure shows



Stream statement

Lable: L int2

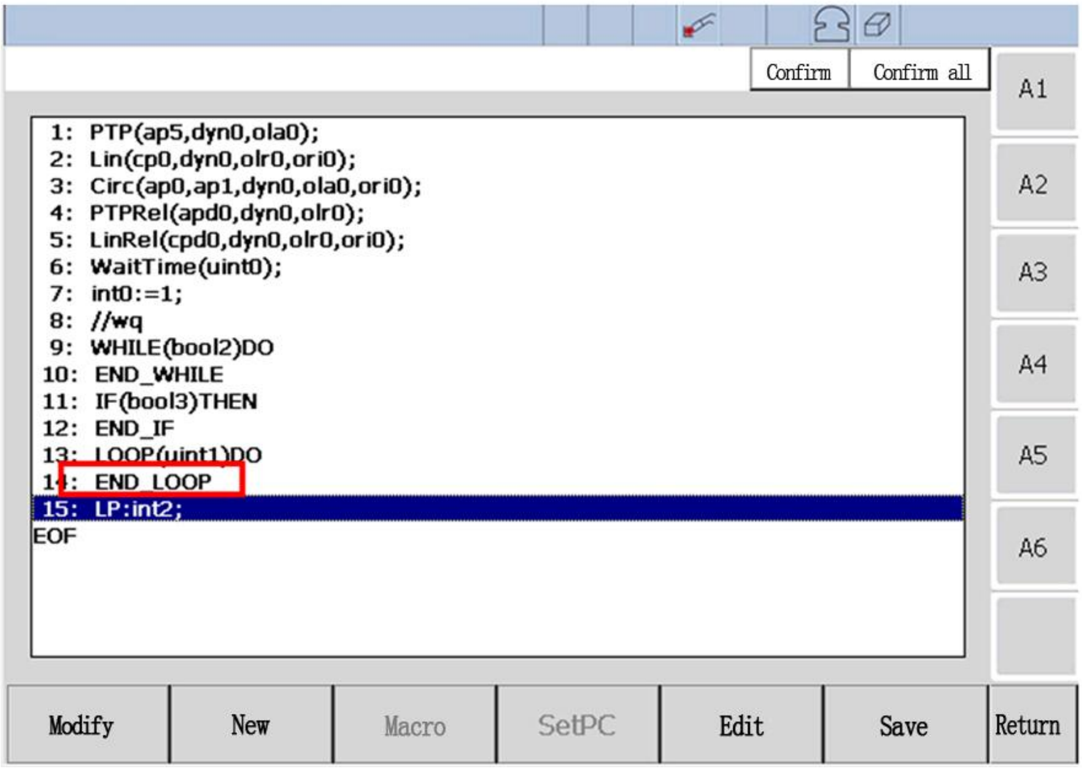
Parameter data

INT int2=0

Setting value :

LP:int2

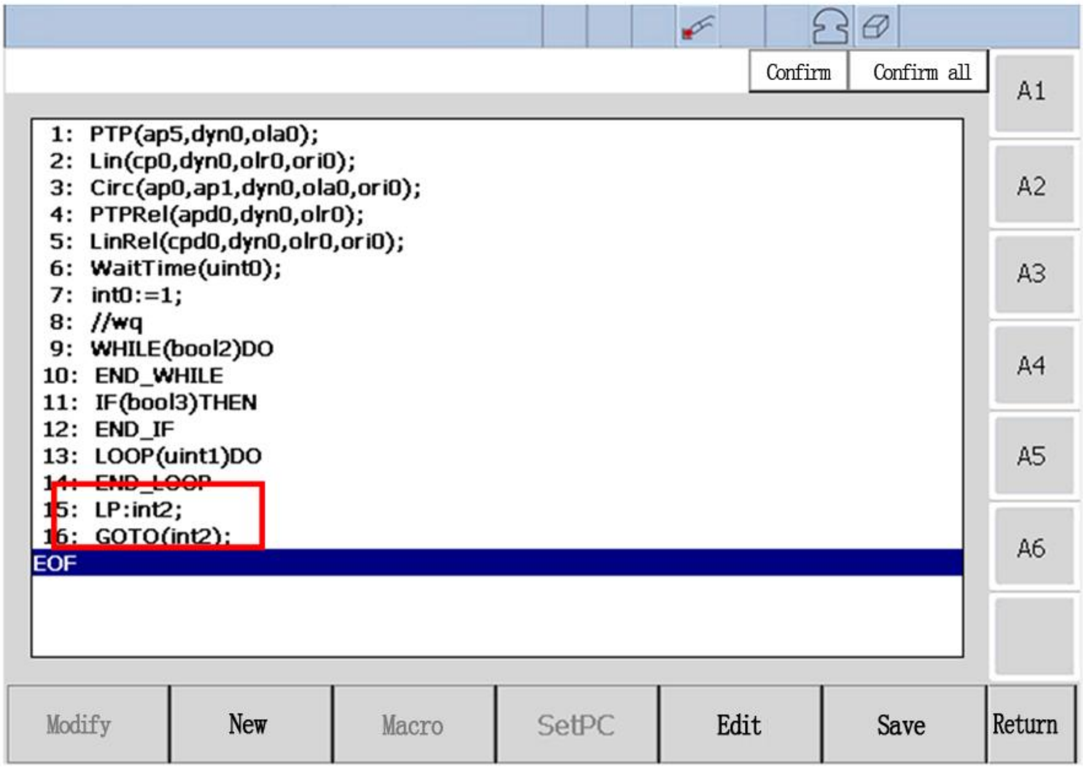
- Click OK, as the figure shows



```
PTP(ap5,dyn0,ola0);
Lin(cp0,dyn0,olr0,ori0);
Circ(ap0,ap1,dyn0,ola0,ori0);
PTPRel(apd0,dyn0,olr0);
LinRel(cpd0,dyn0,olr0,ori0);
WaitTime(uint0);
int0:=1;
//wq
WHILE(bool2)DO
END_WHILE
IF(bool3)THEN
END_IF
LOOP(uint1)DO
END_LOOP
LP:int2;
EOF
```

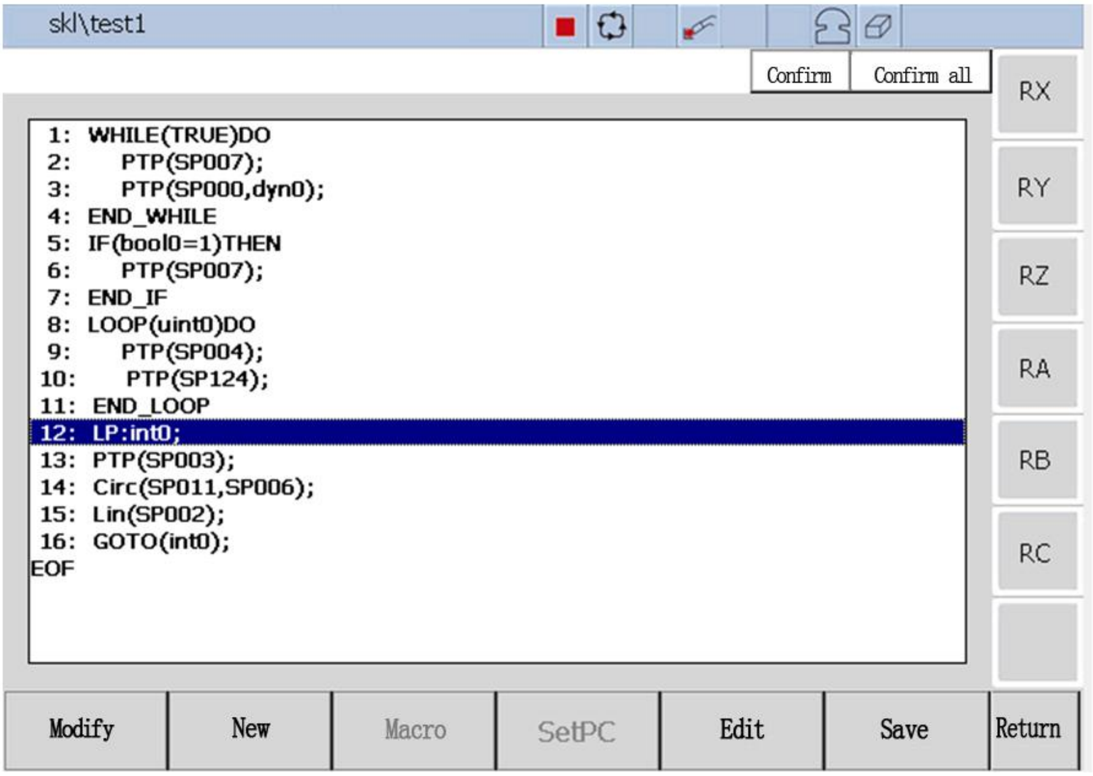
3.4.5 Goto and lp statements

- Select GOTO (conjunction with LP) and LP variables, click OK. As shown in the figure



```
PTP(ap5,dyn0,ola0);
Lin(cp0,dyn0,olr0,ori0);
Circ(ap0,ap1,dyn0,ola0,ori0);
PTPRel(apd0,dyn0,olr0);
LinRel(cpd0,dyn0,olr0,ori0);
WaitTime(uint0);
int0:=1;
//wq
WHILE(bool2)DO
END_WHILE
IF(bool3)THEN
END_IF
LOOP(uint1)DO
END_LOOP
LP:int2;
GOTO(int2);
EOF
```

- As being explained by the following program:



skl\test1

```
WHILE(TRUE)DO
    PTP(SP007);
    PTP(SP000,dyn0);
END_WHILE
IF(bool0=1)THEN
    PTP(SP007);
END_IF
LOOP(uint0)DO
    PTP(SP004);
    PTP(SP124);
END_LOOP
LP:int0;
PTP(SP003);
Circ(SP011,SP006);
Lin(SP002);
GOTO(int0);
EOF
```

- LP:int0;

  PTP(SP003);

  Circ(SP011,SP006);

  Lin(SP002);

  GOTO(int0);

The program acts as follows: Define the jump tag point int0, and moves to SP003 point in PTP, move to SP006 point in arc, move to SP002 point in straight, and then jump to the mode moving to SP003 in PTP. These can be done in cycles.

- LP Function Description: This instruction is used to define the jump tag point.

  Usage: Followed by the tag name (you can choose any one of the variable name).

- GOTO Function Description: used to jump in different parts of the program.

  Usage: Followed by the jump tag (the tag must be consistent with that defined in the LP, or else the function fails).

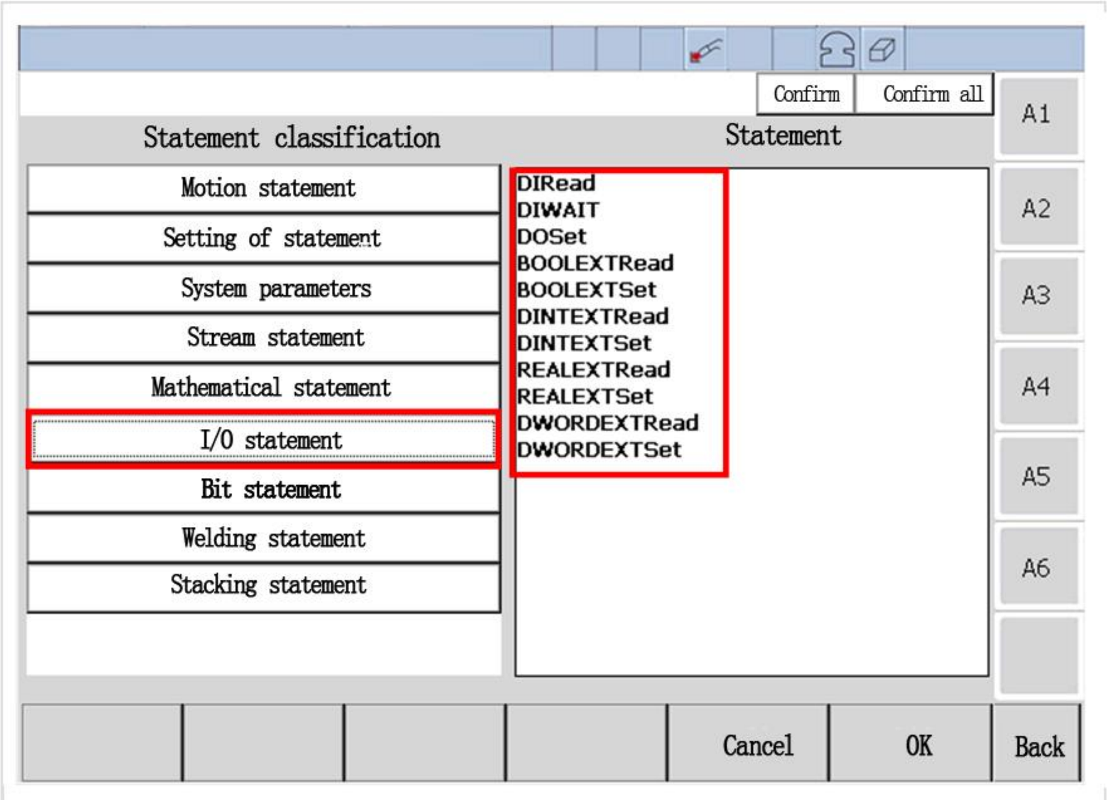
- LP:variable;

  GOTO (variable);

  Format Description: LP is followed by the jump variable name as set by user (only a variable name is used as a jump identifier, but unconcern with the type of such variable), and there's variable name in the parentheses behind the GOTO, for which the variable must be defined after the LP (LP definition statement can be positioned in front or behind the GOTO).

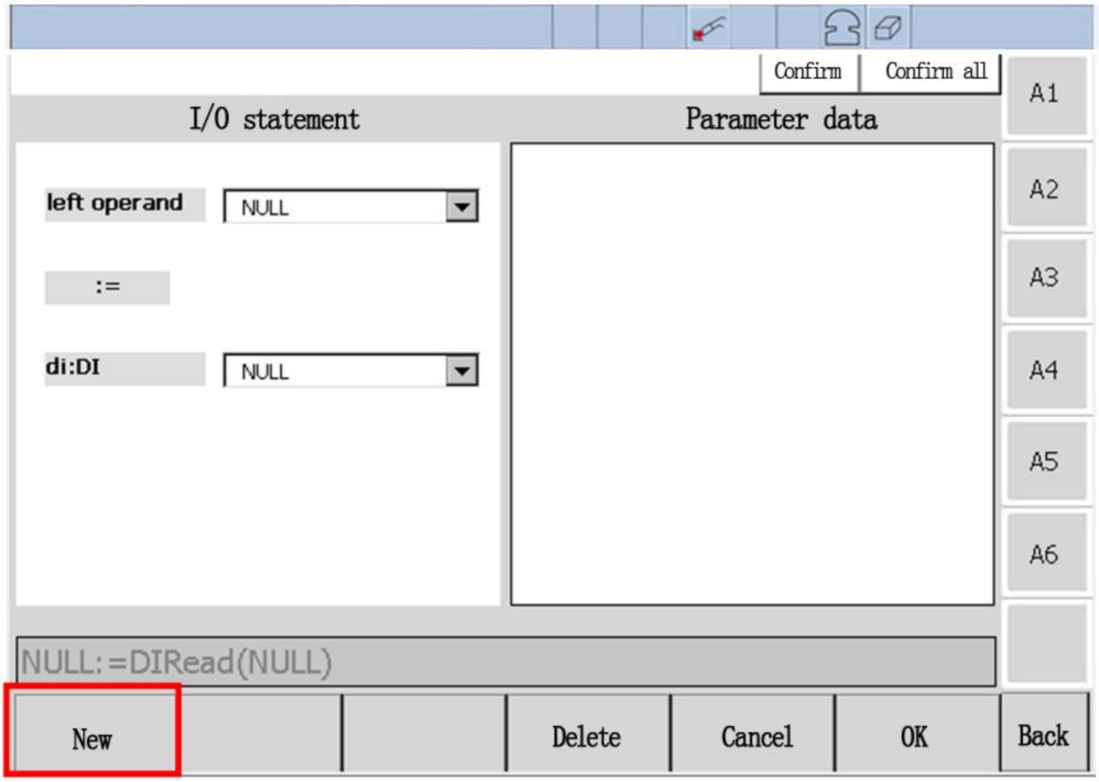
3.5 I/O instruction

- Choose I/O statement, as shown in the figure



3.5.1 DIRead instruction

- Select DIRead, as shown in figure



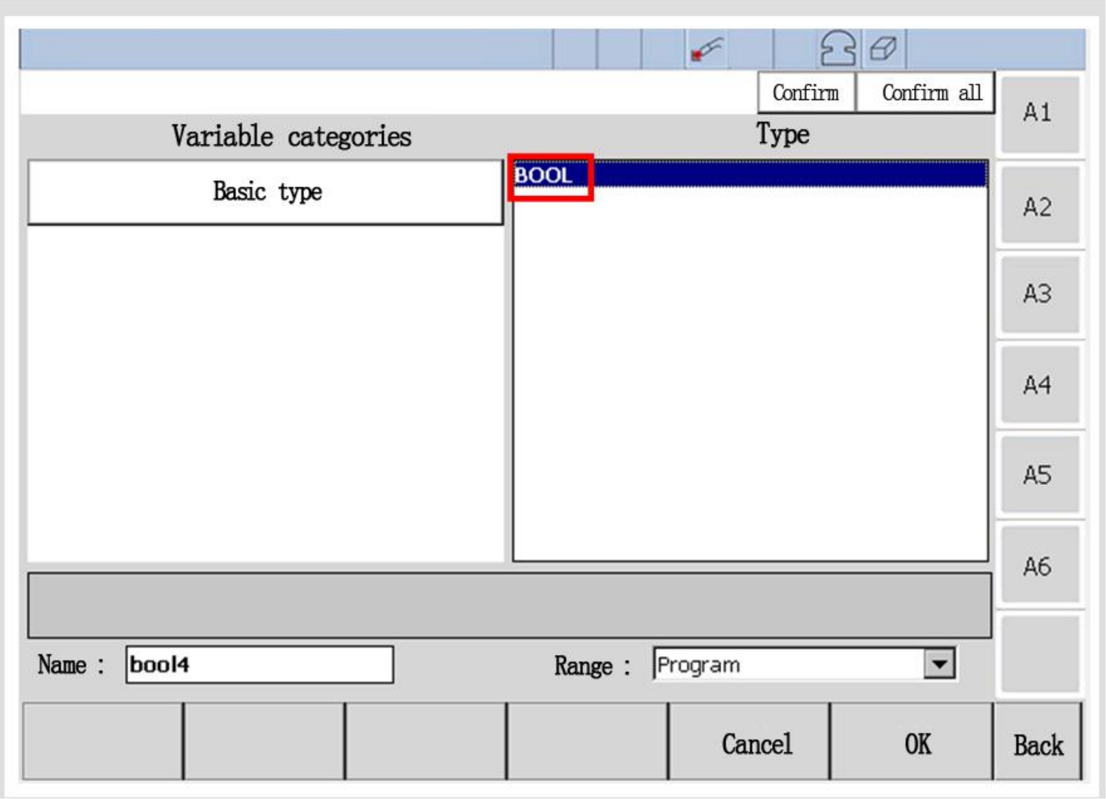
left operand NULL

:=

di:DI NULL

NULL:=DIRead(NULL)

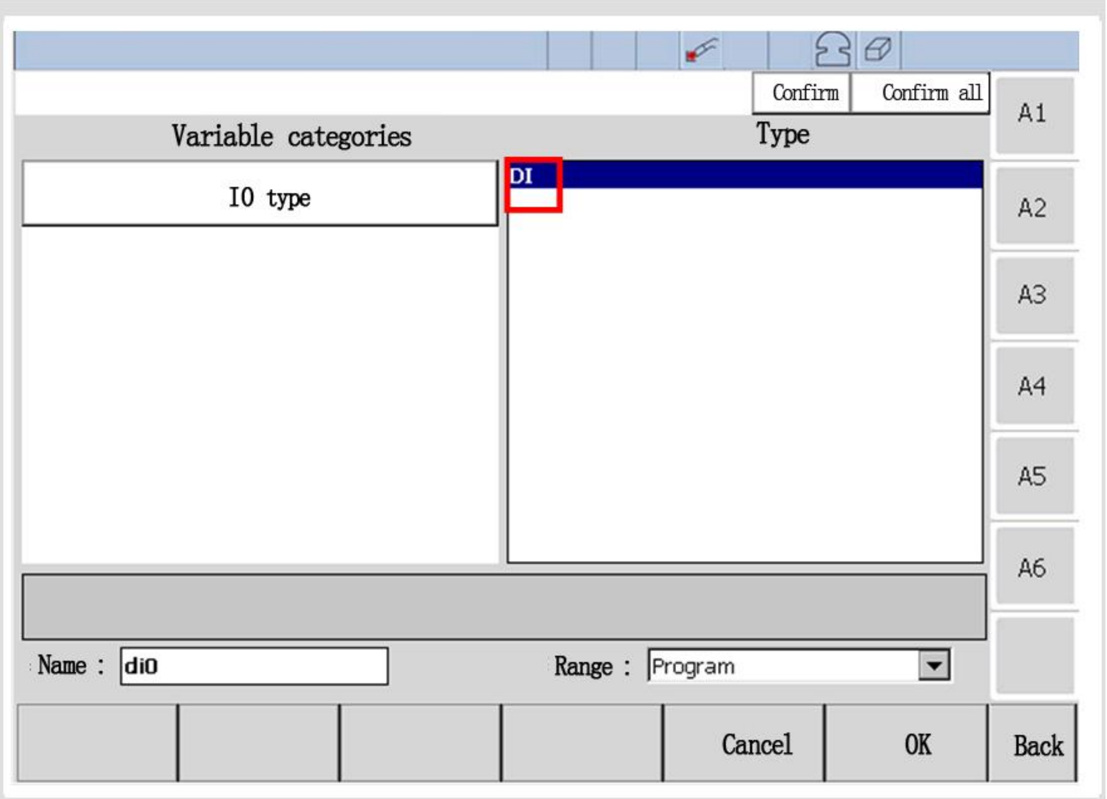
- Create the left operand specified bool type, as shown in figure



Name： bool4

Range： Program

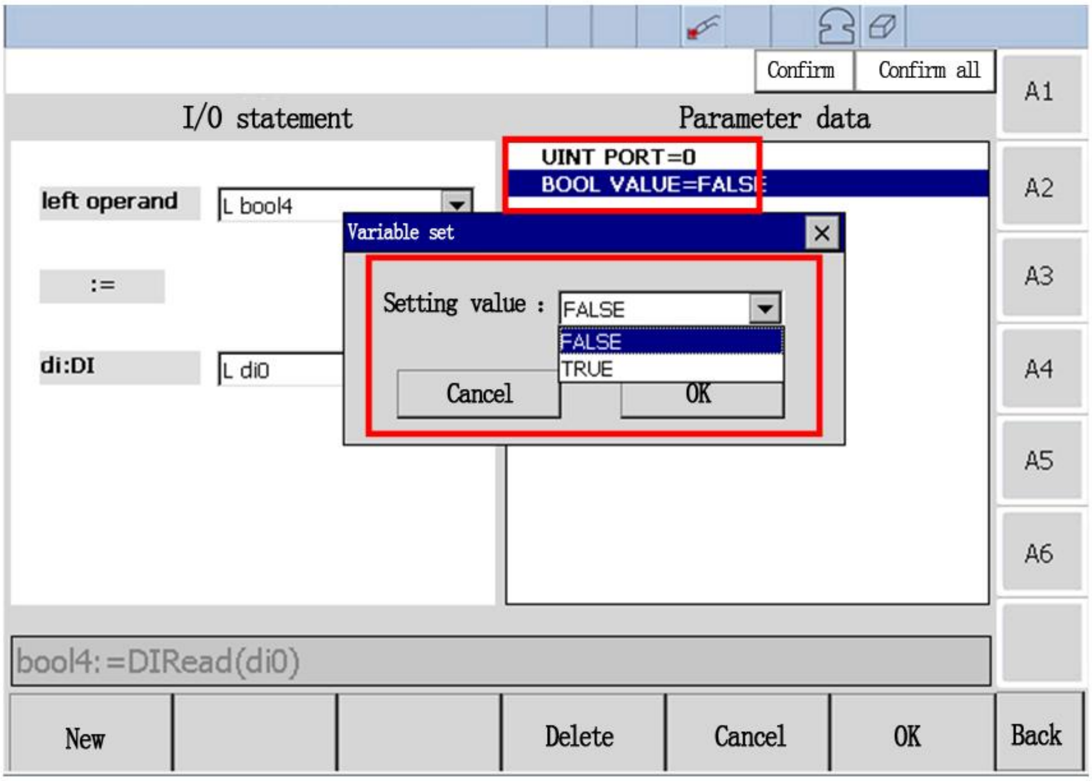
- Click OK, and create the di: DI, as shown in the figure.



Name： di0

Range： Program

- Click OK, and you can modify BOOL VALUE = TRUE / FALSE Click OK, you can modify BOOL VALUE = TRUE / FALSE. as the figure shows



left operand: L bool4

:=

di:DI: L di0

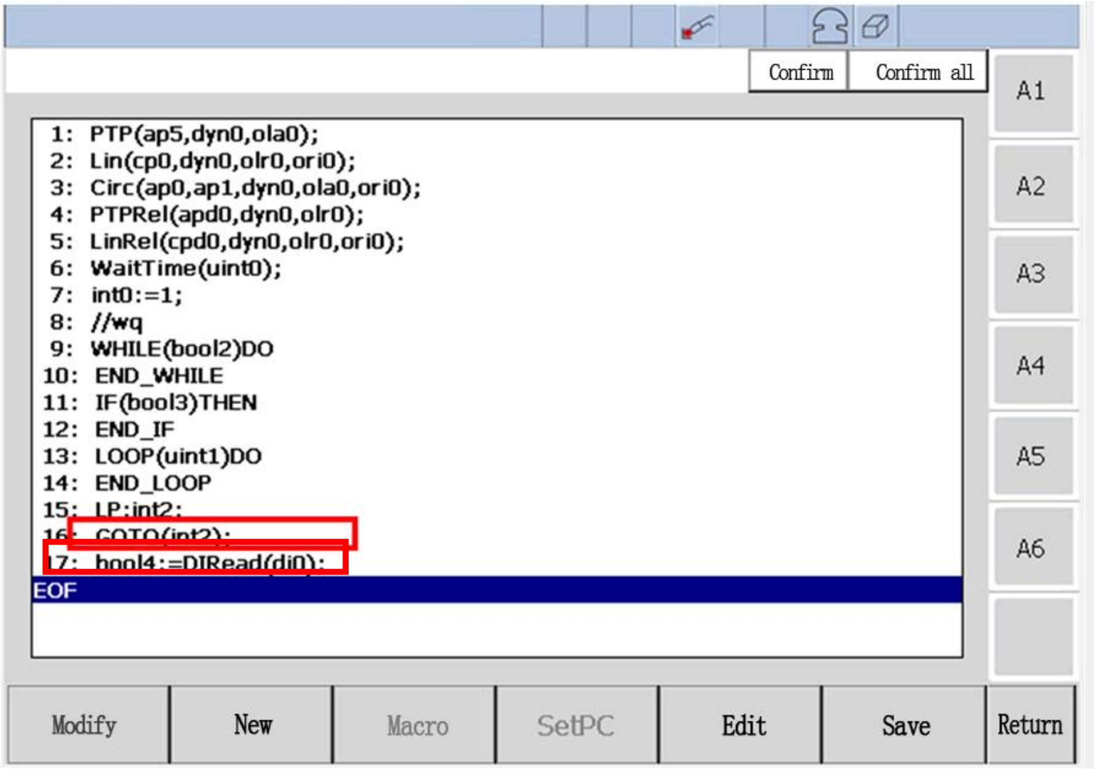
UINT PORT=0
BOOL VALUE=FALSE

Variable set

Setting value : FALSE

bool4:=DIRead(di0)

- Click OK, as shown in the figure.



```
PTP(ap5,dyn0,ola0);
Lin(cp0,dyn0,olr0,ori0);
Circ(ap0,ap1,dyn0,ola0,ori0);
PTPRel(apd0,dyn0,olr0);
LinRel(cpd0,dyn0,olr0,ori0);
WaitTime(uint0);
int0:=1;
//wq
WHILE(bool2)DO
END_WHILE
IF(bool3)THEN
END_IF
LOOP(uint1)DO
END_LOOP
LP:int2;
GOTO(int2);
bool4:=DIRead(di0);
EOF
```

- **Sentence format**

```
BOOLVAR=DIRead(di);
```

- **Parameters**

  di : DI / /structure, including digital input signal port number and value (BOOL)

- **Functional description**

  This instruction is used to read the digit input from the external data port, and the return value is bool type.

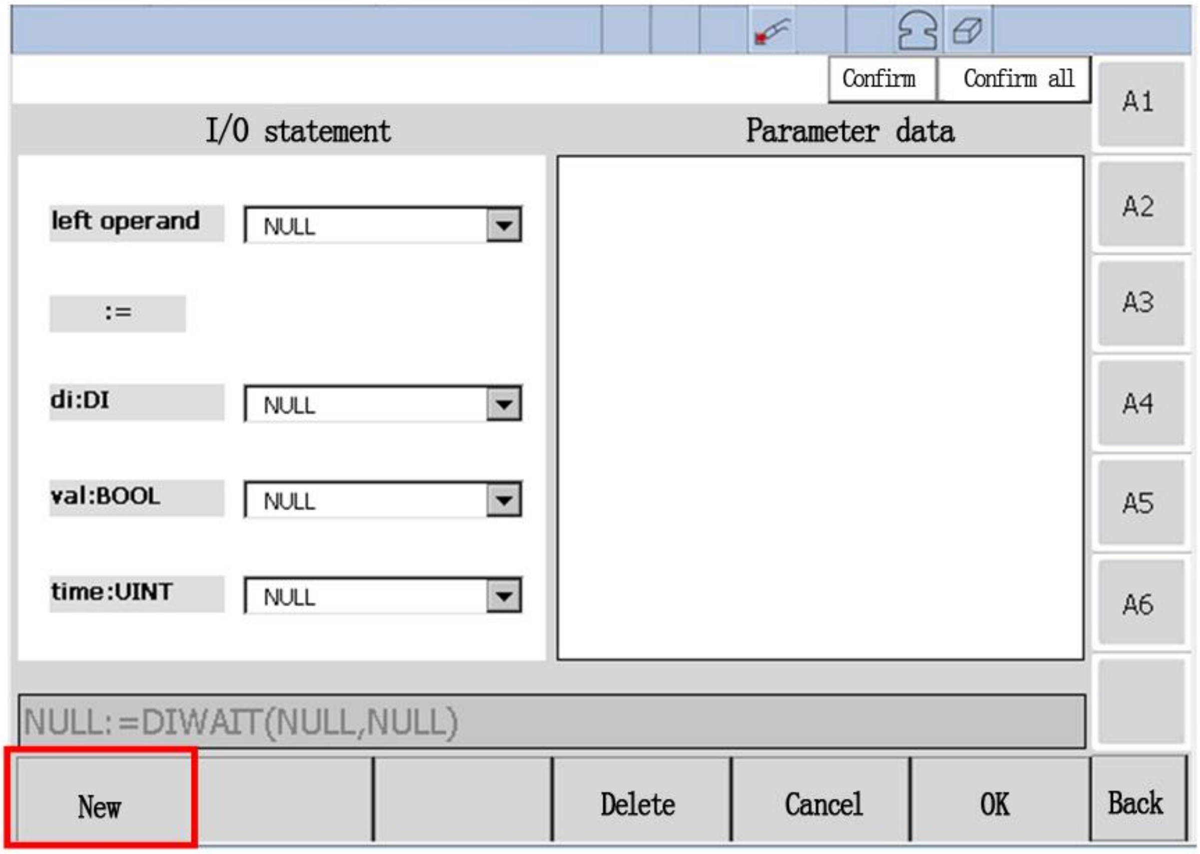
- **Writing example**

  Bool4:=DIRead(di0);

  Program Description: Assign the data in digital quantity structure di0 to the variable bool4.

3.5.2 Diwait instruction

- Select DIWAIT as shown in the diagram.



left operand: NULL

:=

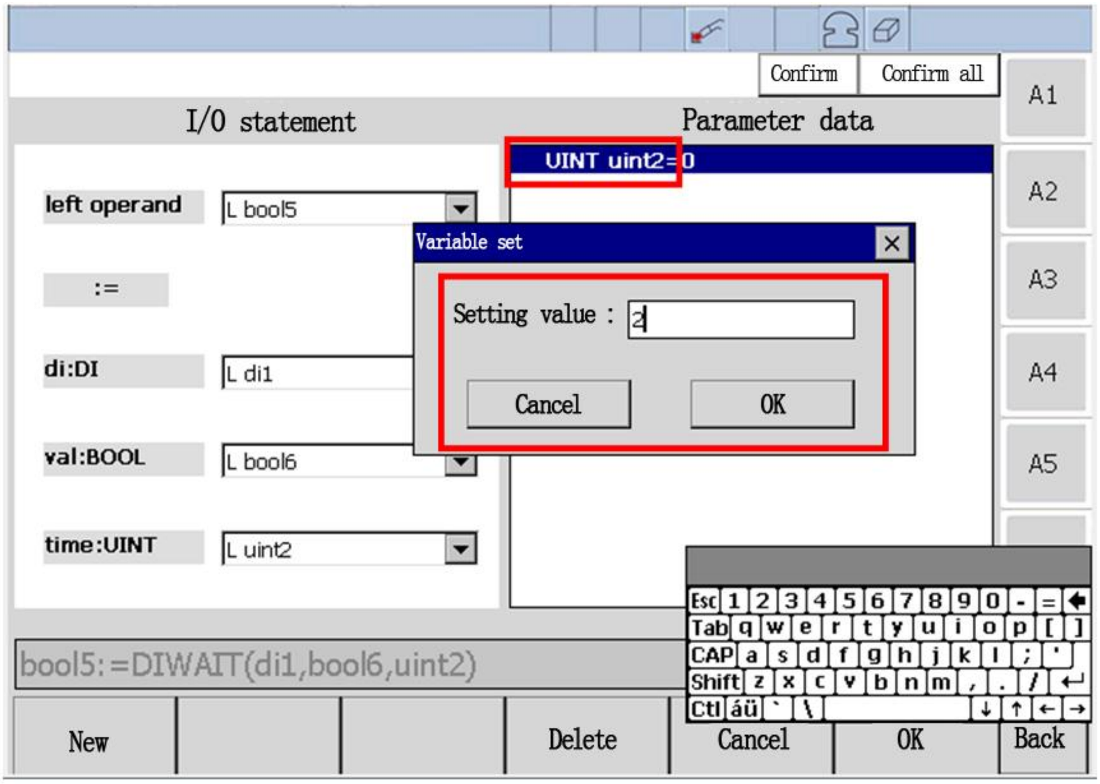
di:DI: NULL

val:BOOL: NULL

time:UINT: NULL

NULL:=DIWAIT(NULL,NULL)

- Select or create a new variable as shown in the diagram


Confirm
Confirm all
A1
A2
A3
A4
A5
Variable set
Cancel
OK
Esc 1 2 3 4 5 6 7 8 9 0 - =
Tab q w e r t y u i o p [ ]
CAP a s d f g h j k l ; '
Shift z x c v b n m , . /
Ctl áü ` \ ↓ ↑ ← →
New
Delete
Cancel
OK
Back

I/O statement

left operand: L bool5

:=

di:DI: L di1

val:BOOL: L bool6

time:UINT: L uint2

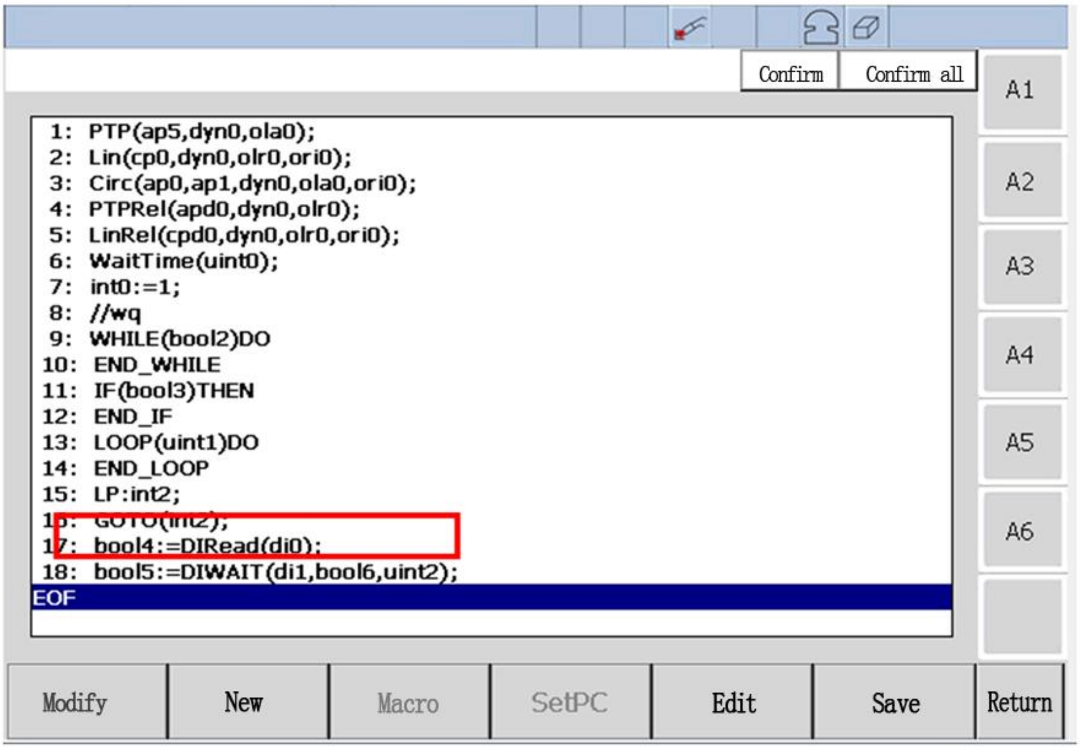
Parameter data

UINT uint2=0

Setting value : 2

```
bool5:=DIWAIT(di1,bool6,uint2)
```

- Click OK as shown in the diagram



```
PTP(ap5,dyn0,ola0);
Lin(cp0,dyn0,olr0,ori0);
Circ(ap0,ap1,dyn0,ola0,ori0);
PTPRel(apd0,dyn0,olr0);
LinRel(cpd0,dyn0,olr0,ori0);
WaitTime(uint0);
int0:=1;
//wq
WHILE(bool2)DO
 END_WHILE
 IF(bool3)THEN
 END_IF
 LOOP(uint1)DO
 END_LOOP
 LP:int2;
 GOTO(int2);
 bool4:=DIRead(di0);
 bool5:=DIWAIT(di1,bool6,uint2);
EOF
```

- **Statement format**

```
BOOLVAR=DIWAIT(di,val,time);
```

- **Parameter**

Di: DI

Val: BOOL

Time: UINT (ms) is optional

- **Function description**

This instruction is used to wait for the trigger signal within a specified waiting time and assign a value. The return value is boolean.

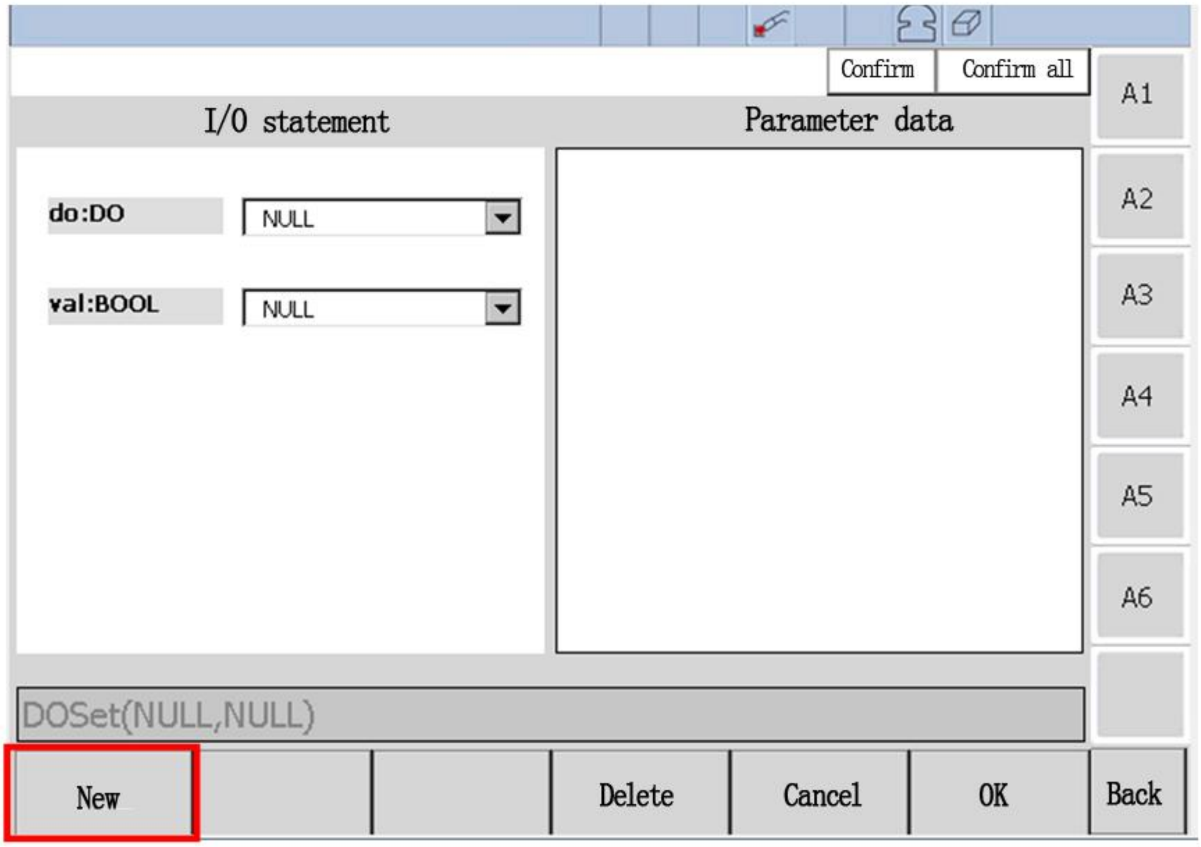
- **Sample**

Bool5=DIWAIT (di1,bool6,uint2);

The program demonstrates that within uint2, if val signal is received from the dil digital input terminal, bool5 is set to true.

3.5.3 Doset instruction

- Select or create DoSet as shown in the diagram

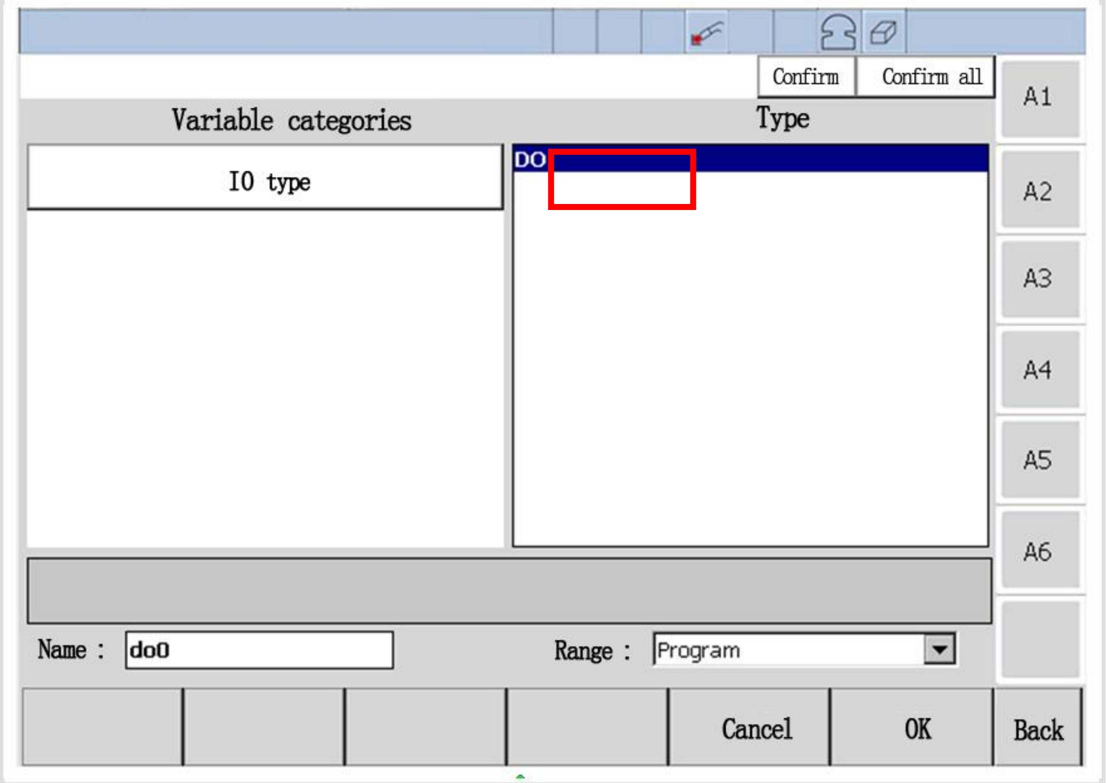


do:DO NULL

val:BOOL NULL

DOSet(NULL,NULL)

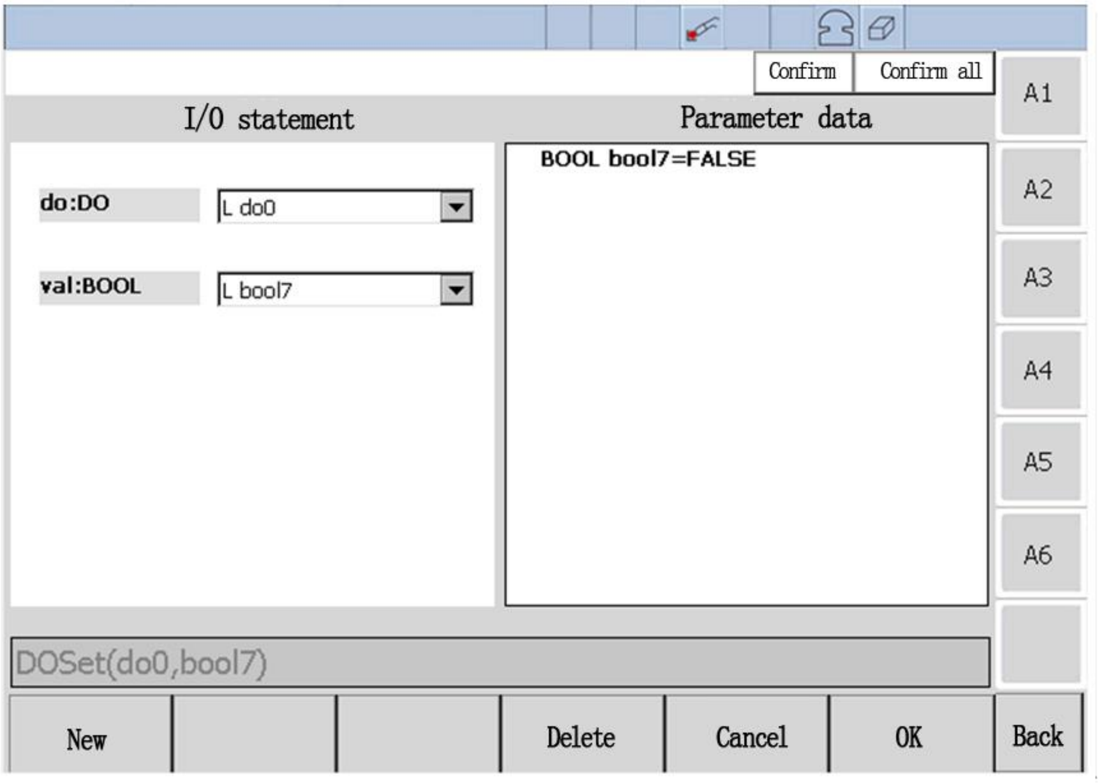
- Create do: DO, which has been specified as DO type as shown in the diagram.



Name : do0

Range : Program

- Set val: BOOL parameter data FALSE/TRUE as shown in the diagram.



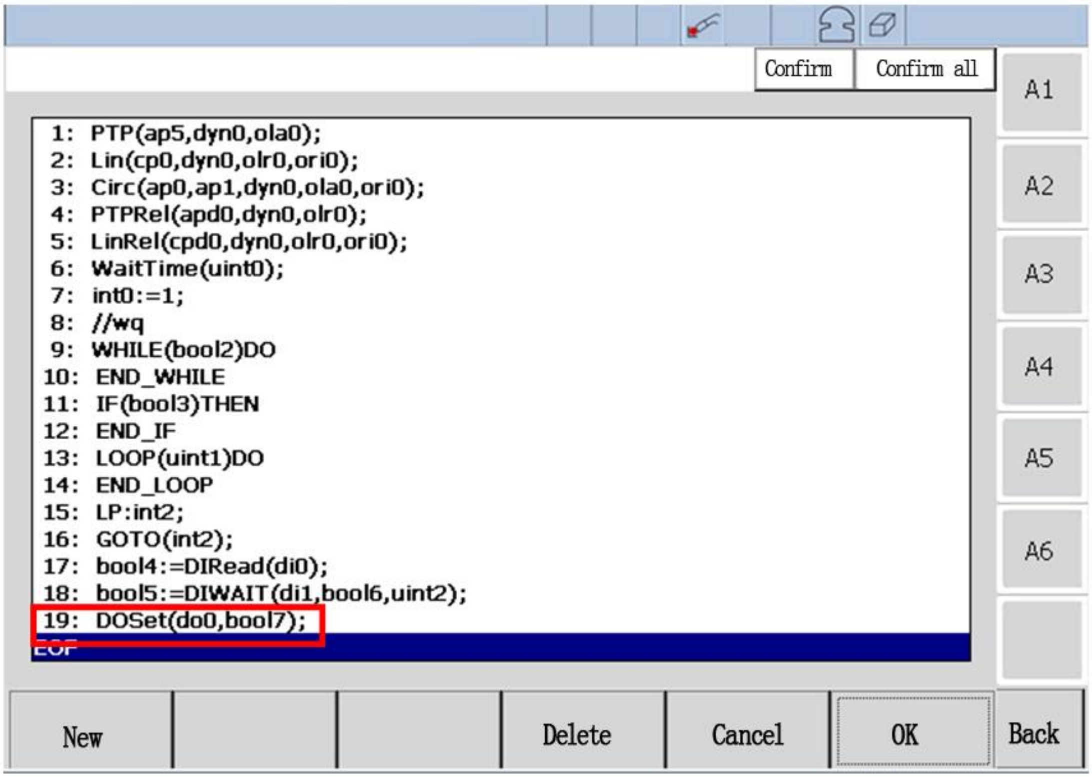
do:DO L do0

val:BOOL L bool7

BOOL bool7=FALSE

DOSet(do0,bool7)

- Click OK as shown in the diagram.



```
1:  PTP(ap5,dyn0,ola0);
2:  Lin(cp0,dyn0,olr0,ori0);
3:  Circ(ap0,ap1,dyn0,ola0,ori0);
4:  PTPRel(apd0,dyn0,olr0);
5:  LinRel(cpd0,dyn0,olr0,ori0);
6:  WaitTime(uint0);
7:  int0:=1;
8:  //wq
9:  WHILE(bool2)DO
10:  END_WHILE
11:  IF(bool3)THEN
12:  END_IF
13:  LOOP(uint1)DO
14:  END_LOOP
15:  LP:int2;
16:  GOTO(int2);
17:  bool4:=DIRead(di0);
18:  bool5:=DIWAIT(di1,bool6,uint2);
19:  DOSet(do0,bool7);
EOF
```

➢ **Statement format**

DOSet(do,val);

➢ **Parameter**

do ： DO

val：BOOL

➢ **Function description**

This instruction is used to read the digital input from the external data port.

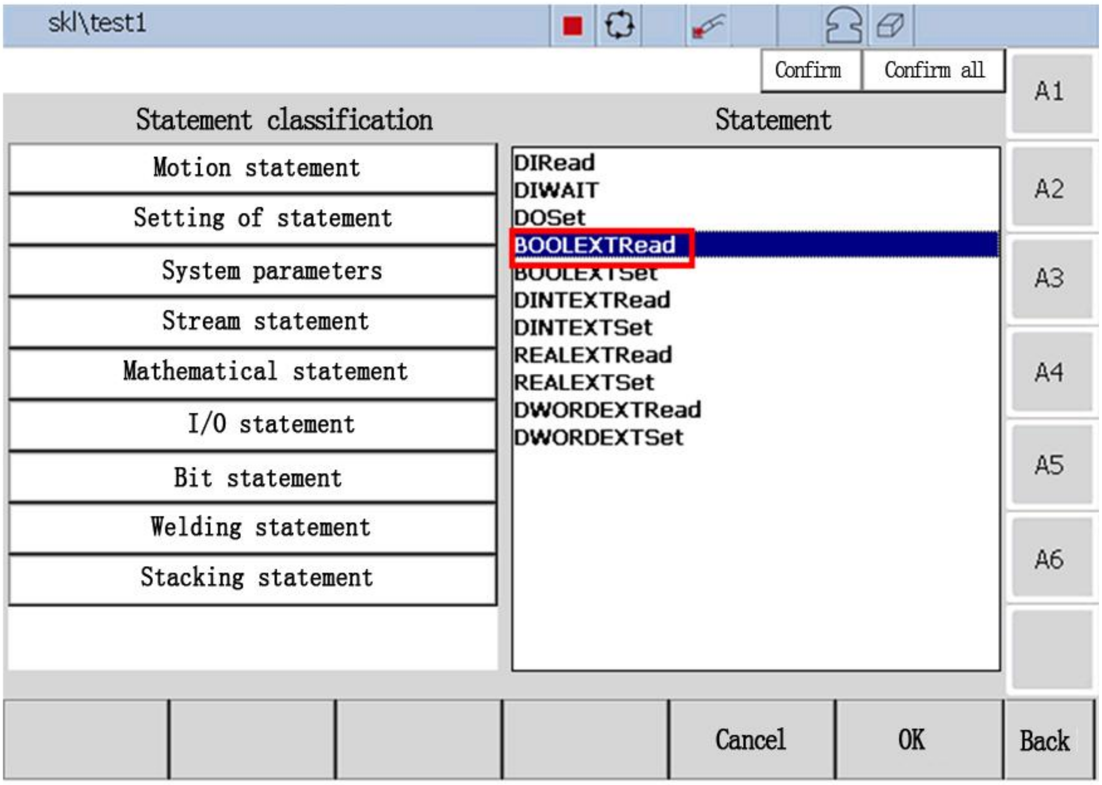
➢ **Sample**

DOSet(do0,bool7);

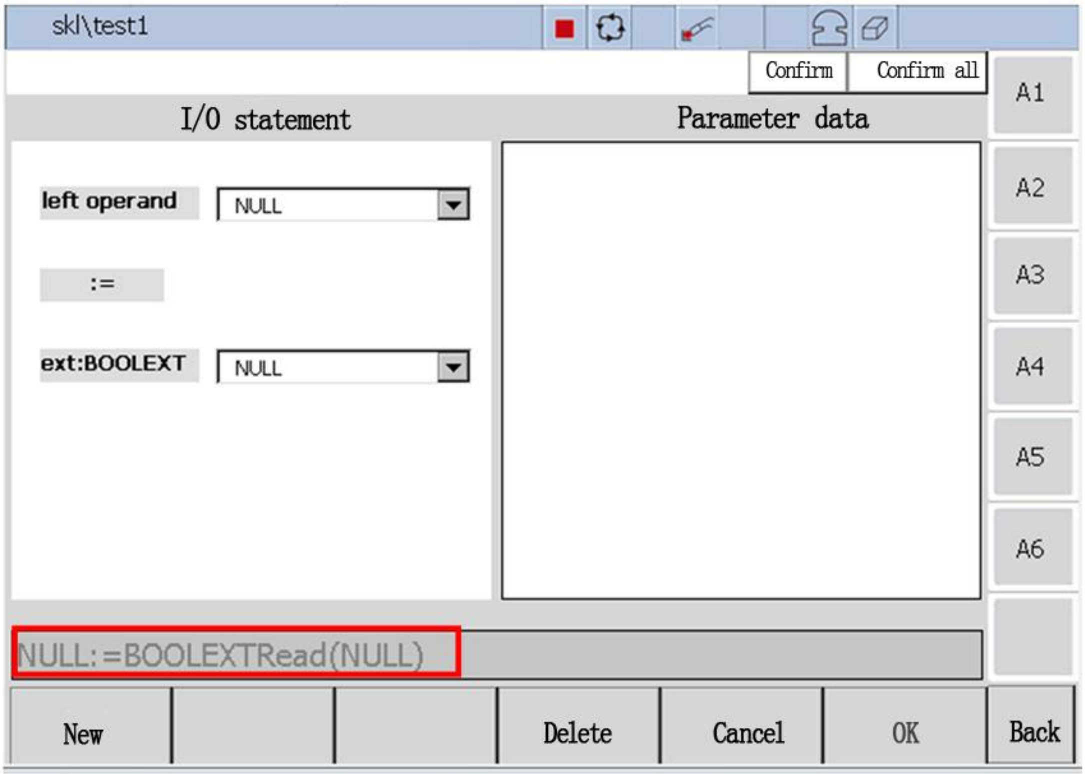
Program function: assign the value of variable bool7 to do0.

3.5.4 BOOLEXTRead instruction

- Create BOOLEXTRead as shown in the following diagram



- Click OK as shown in the following diagram



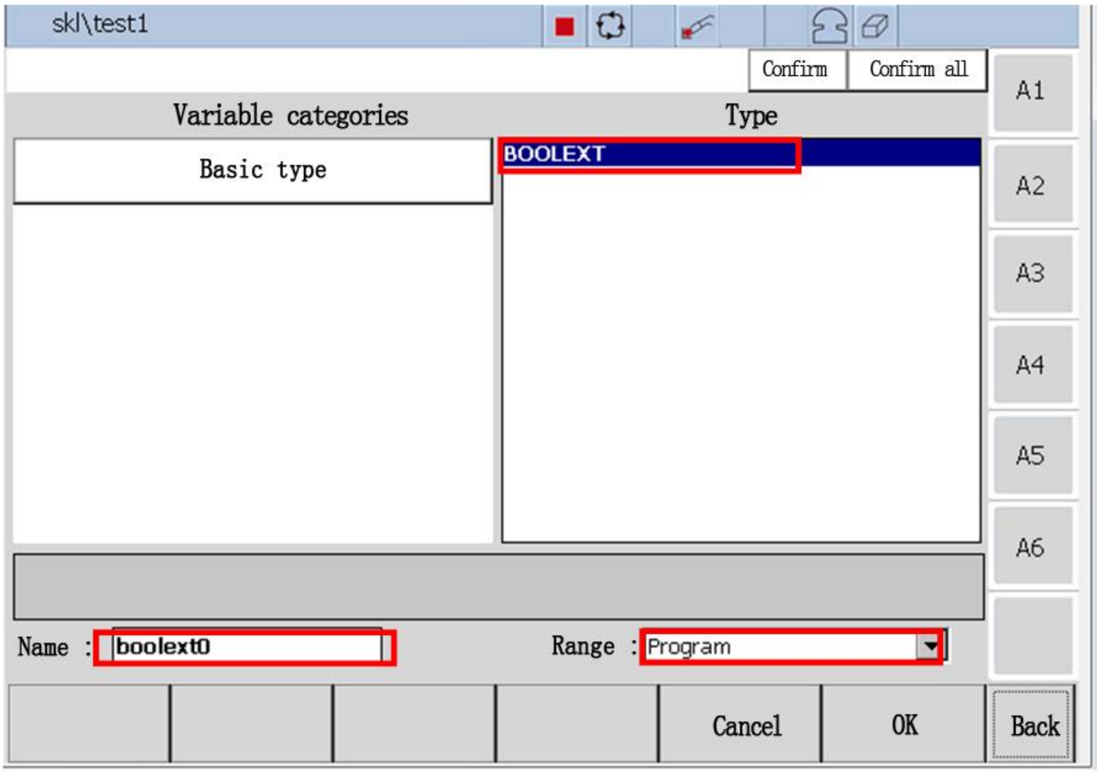
left operand NULL

:=

ext:BOOLEXT NULL

NULL:=BOOLEXTRead(NULL)

- It can select the defined variable or create a new variable as the variable. Select the new variable and click NEW as shown in the following diagram.



skl\test1

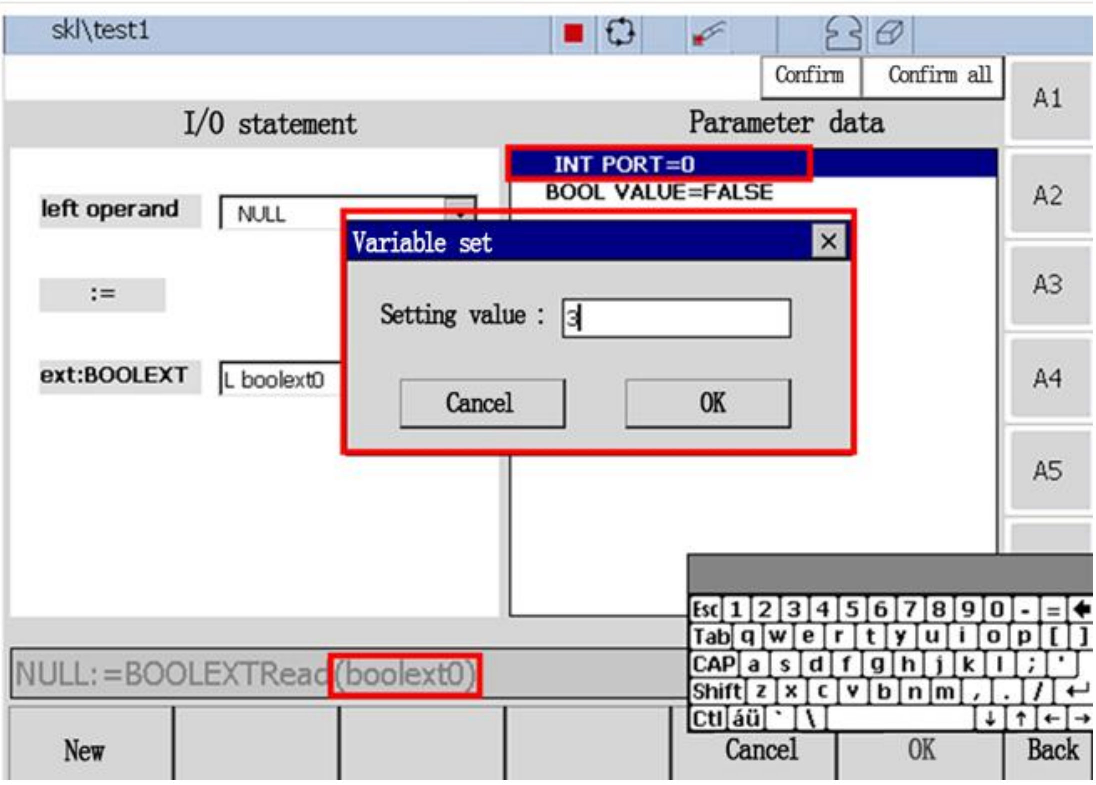
Variable categories: Basic type

Type: BOOLEXT

Name : boolext0

Range : Program

- Modify its name and scope and click OK, and then turn to the interface shown in the following diagram.



skl\test1

I/O statement

left operand NULL

:=

ext:BOOLEXT L boolext0

Parameter data

INT PORT=0

BOOL VALUE=FALSE

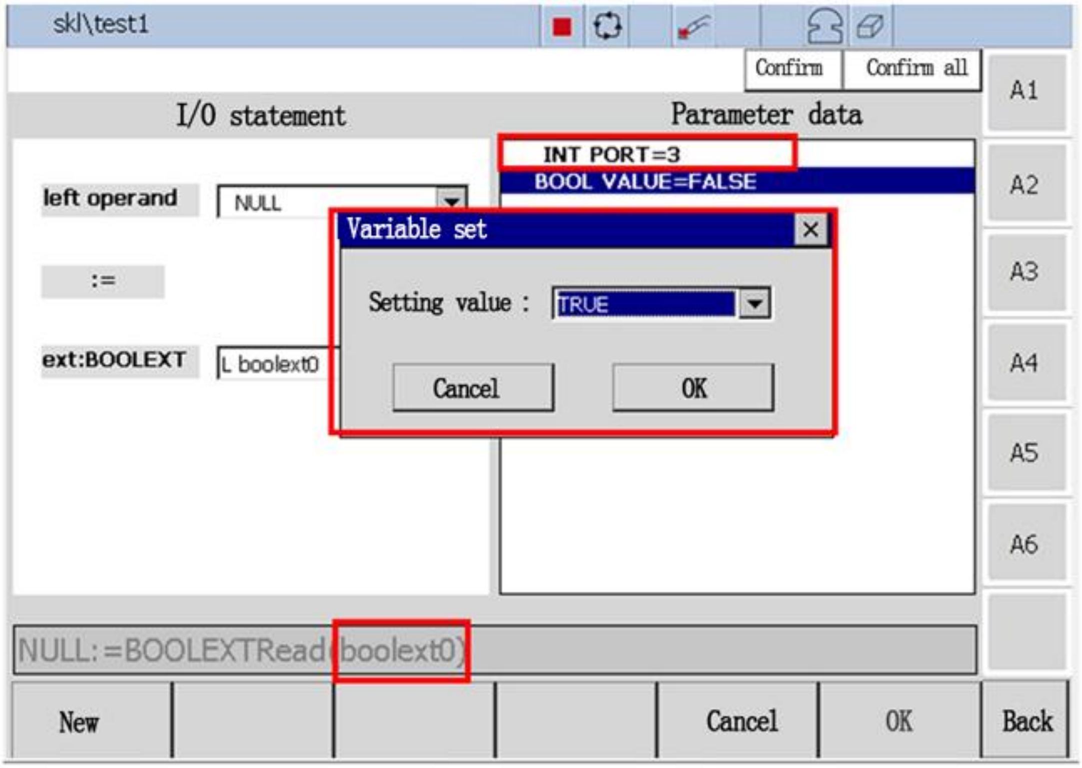
Variable set

Setting value : 3

NULL:=BOOLEXTRead(boolext0)

Confirm
Confirm all
A1
A2
A3
A4
A5
Cancel
OK
New
Cancel
OK
Back
Esc 1 2 3 4 5 6 7 8 9 0 - =
Tab q w e r t y u i o p [ ]
CAP a s d f g h j k l ; '
Shift z x c v b n m , . /
Ctl áü ` \

- Click the two parameter settings on the right side of parameter data and set the parameter value. The parameter setting in the red box is finished after clicking OK.



skl\test1

I/O statement

left operand: NULL

:=

ext:BOOLEXT: L boolext0

Parameter data

INT PORT=3

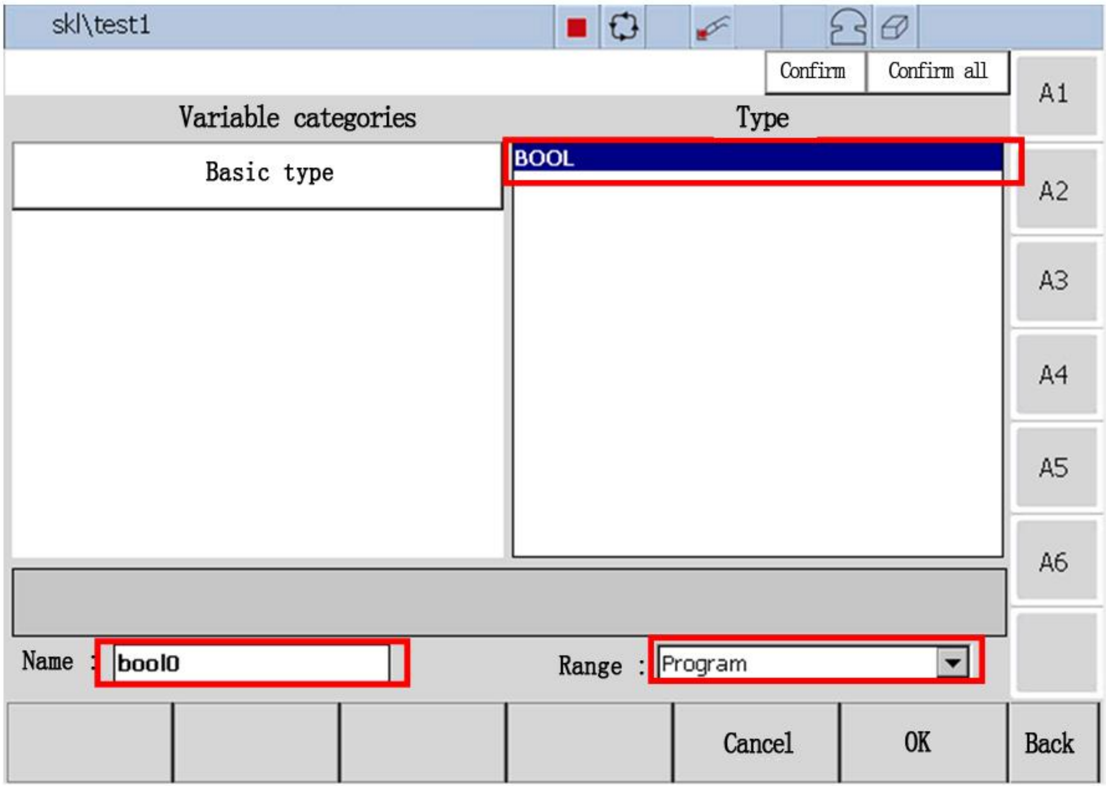
BOOL VALUE=FALSE

Variable set

Setting value : TRUE

NULL:=BOOLEXTRead(boolext0)

- Then set the variables in left operand as shown in the following diagram.



skl\test1

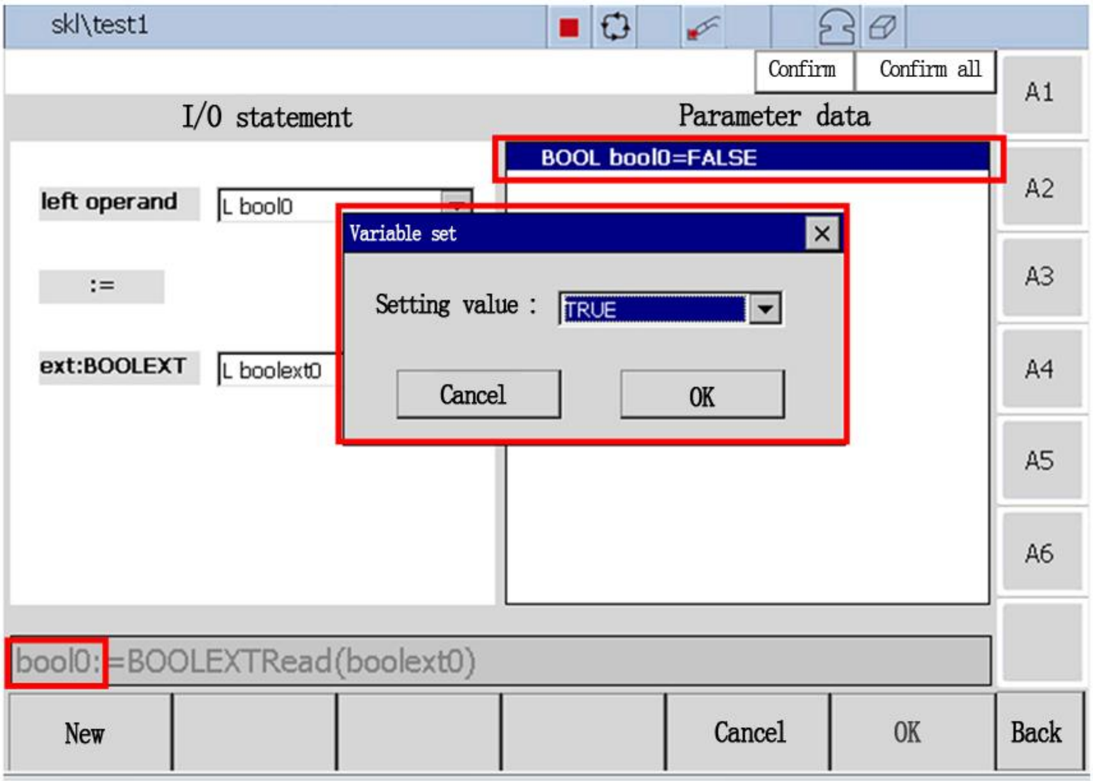
Variable categories: Basic type

Type: BOOL

Name : bool0

Range : Program

- Select the data type, change its name and scope, and then click OK as shown in the following diagram.



BOOL bool0=FALSE

left operand: L bool0

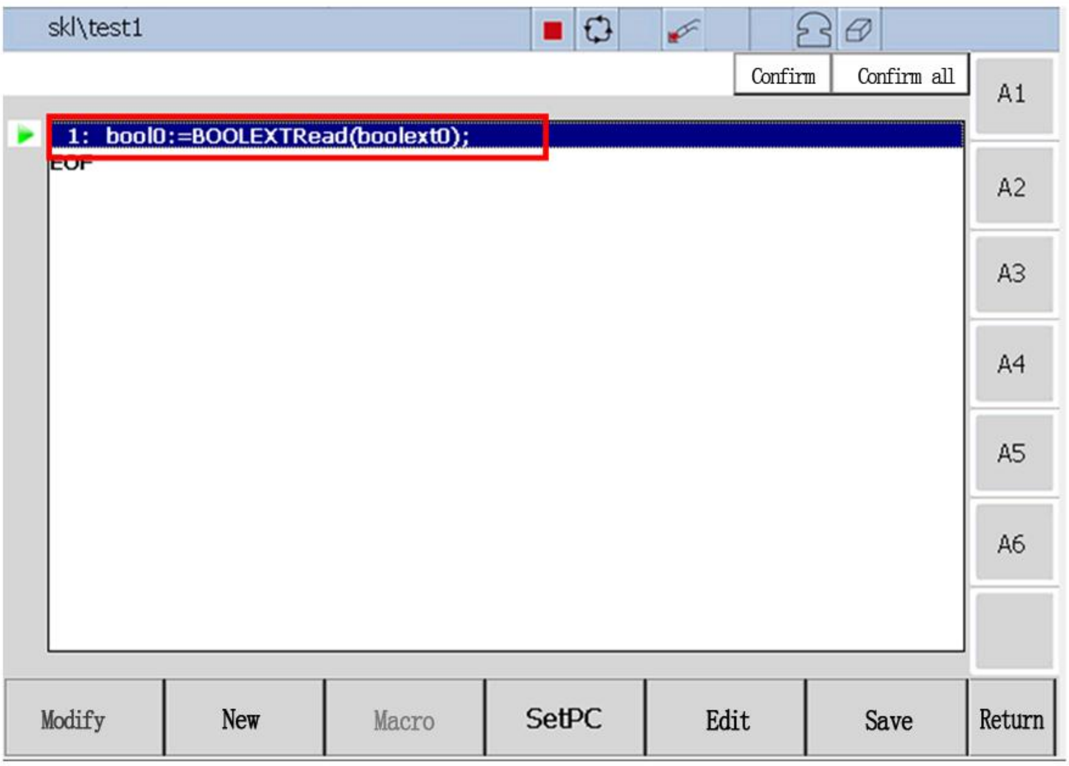
:=

ext:BOOLEXT: L boolext0

Setting value : TRUE

bool0:=BOOLEXTRead(boolext0)

- Change the parameter value, click OK and complete.



1: bool0:=BOOLEXTRead(boolext0);
EOF

- **Statement format**

```
BOOLVAR= BOOLEXTRead(ext);
```

- **Parameter**

ext : BOOLEXT / /the structure, including the number and the value of the digital input signal port

- **Function description**

This instruction is used to read the specified variable in PLC. The return value is boolean.

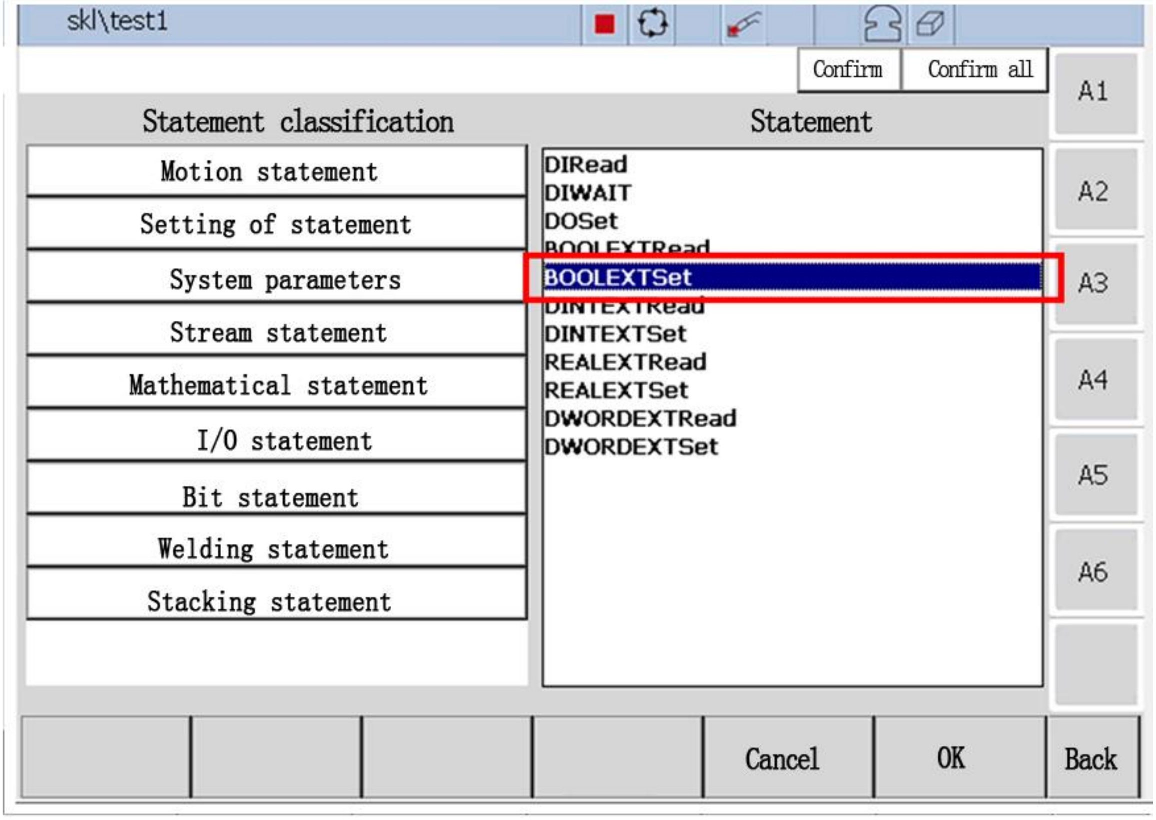
- **Sample**

bool0= BOOLEXTRead(boolext0);

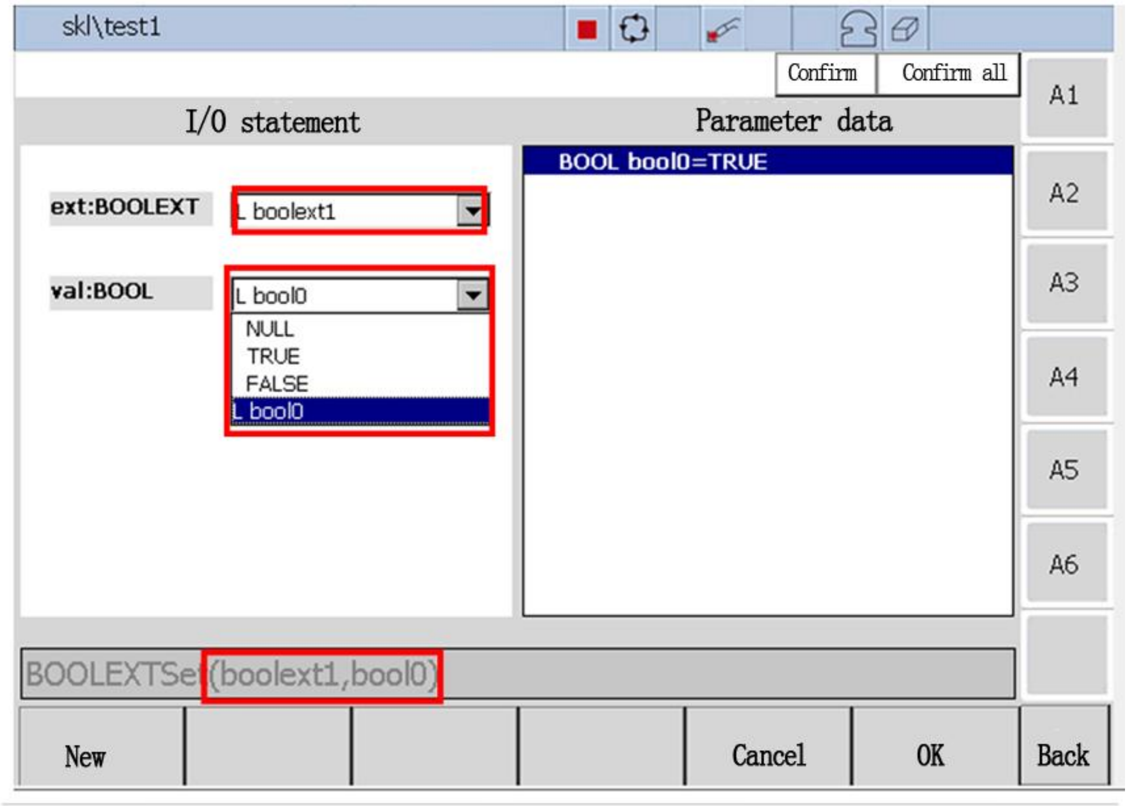
The program demonstrates: assign the data in the external data structure boolext0 to the variable bool0.

3.5.5 BOOLEXTSet instruction

- Create BOOLEXTSet as shown in the following diagram



- Click OK and set New and the parameter data of ext and val variables.



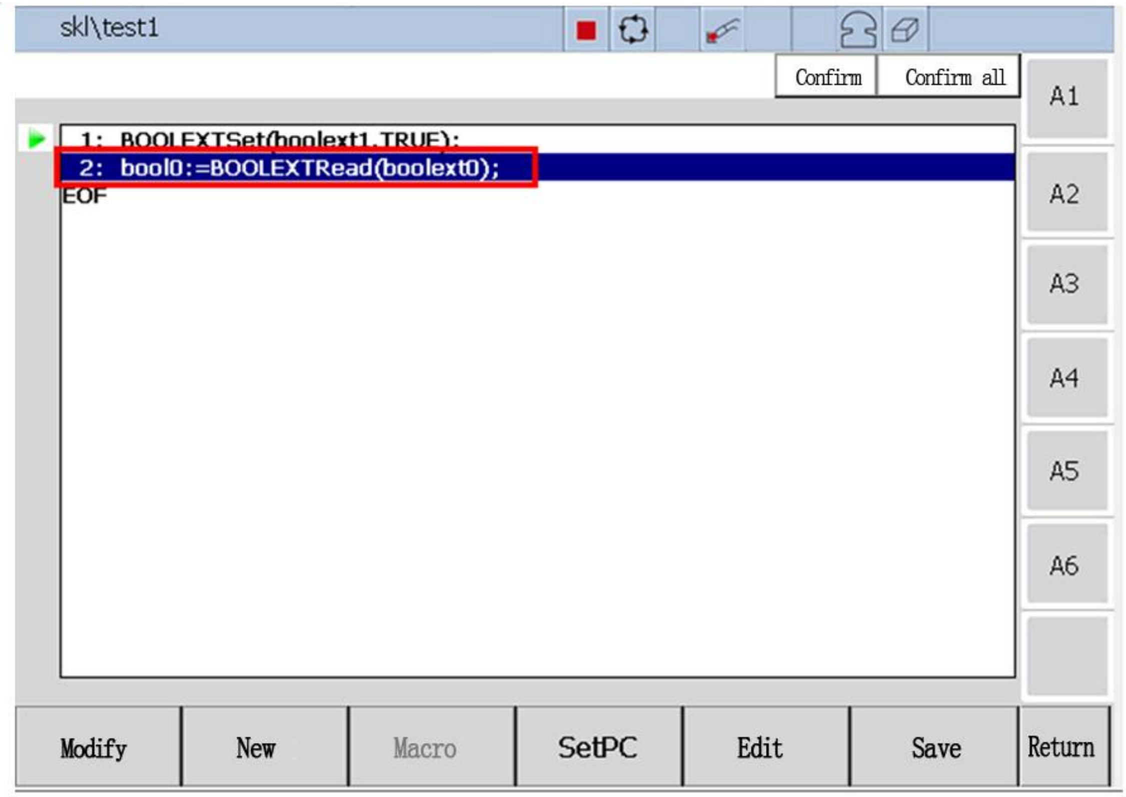
ext:BOOLEXT: L boolext1

val:BOOL: L bool0

BOOL bool0=TRUE

BOOLEXTSet(boolext1,bool0)

- Click OK and complete BOOLEXTSet statements shown in the following diagram.



```
1:  BOOLEXTSet(boolext1,TRUE);
2:  bool0:=BOOLEXTRead(boolext0);
EOF
```

- **Statement format**

```
BOOLEXTSet(ext, val);
```

- **Parameter**

ext : BOOLEXT / /the structure, including the number and the value of the digital input signal port

val：BOOL

- **Function description**

This instruction is used to set the specified variable in PLC.

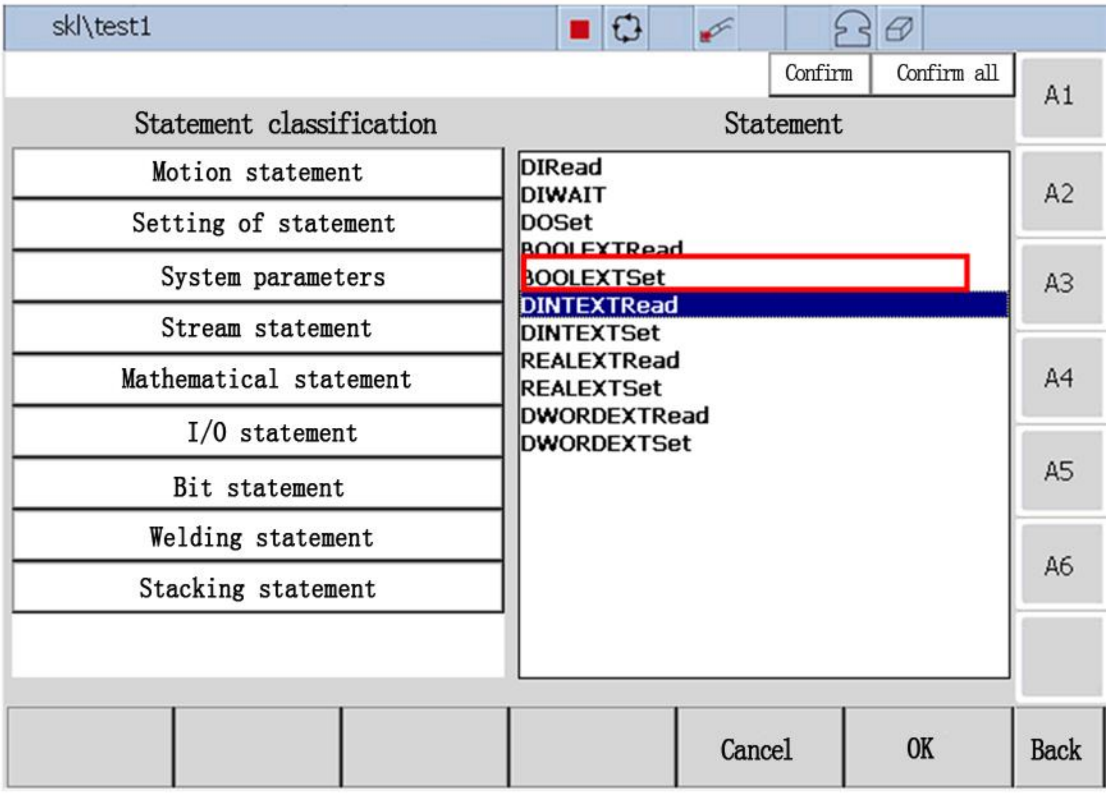
- **Sample**

BOOLEXTSet(boolext1, TRUE);

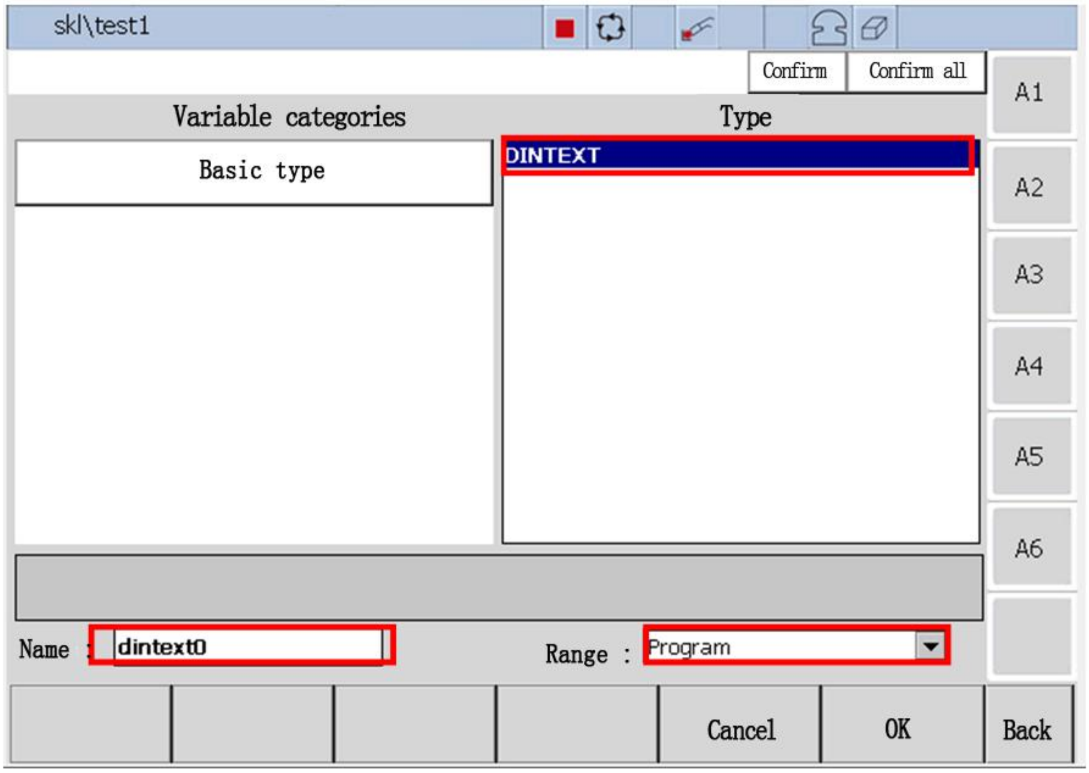
The program function: assign the value of TRUE to the external data structure boolext1.

3.5.6 DINTEXTRead instruction

- Create DINTEXTRead as shown in the following diagram



- Click OK and complete it. The following three diagrams shows how to create the variables and set their parameters.



Variable categories

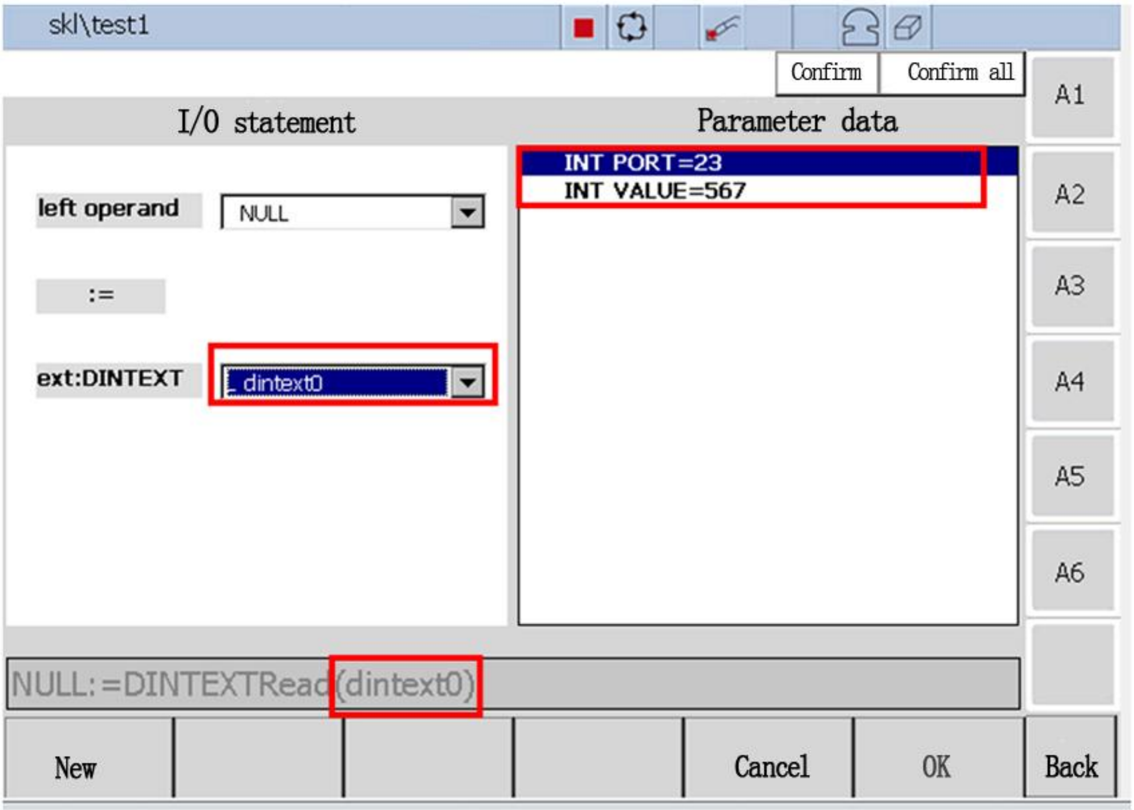
Basic type

Type

DINTEXT

Name : dintext0

Range : Program



I/O statement

left operand NULL

:=

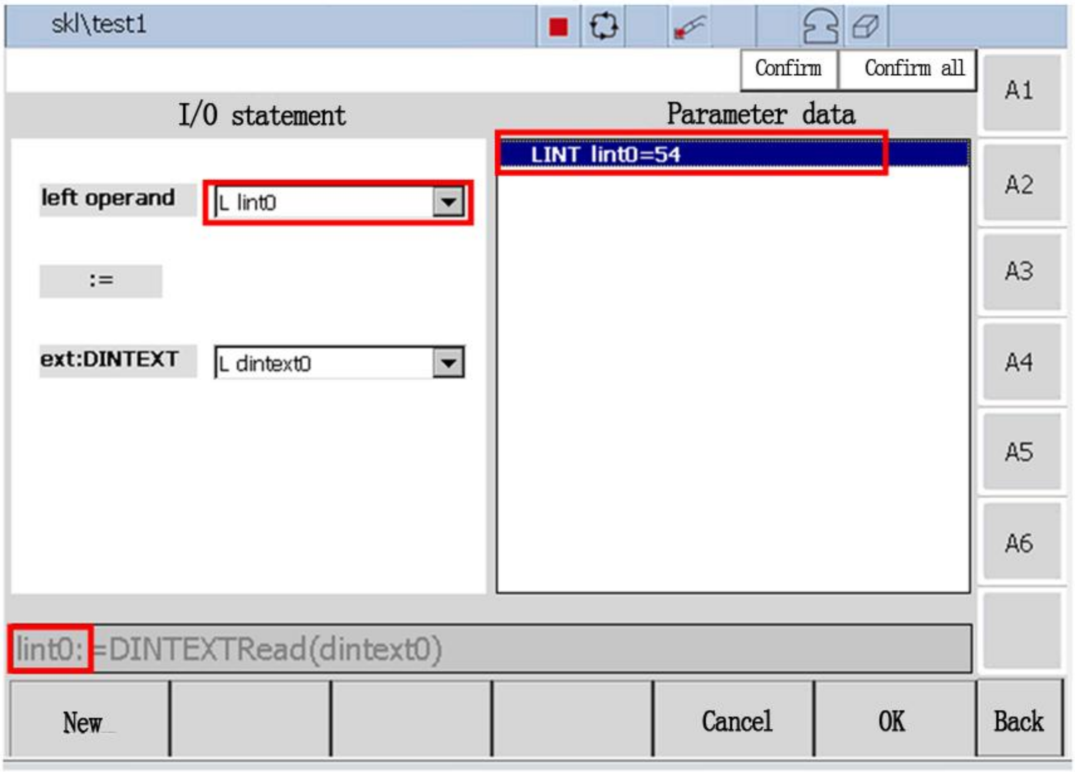
ext:DINTEXT dintext0

Parameter data

INT PORT=23

INT VALUE=567

NULL:=DINTEXTRead(dintext0)



LINT lint0=54

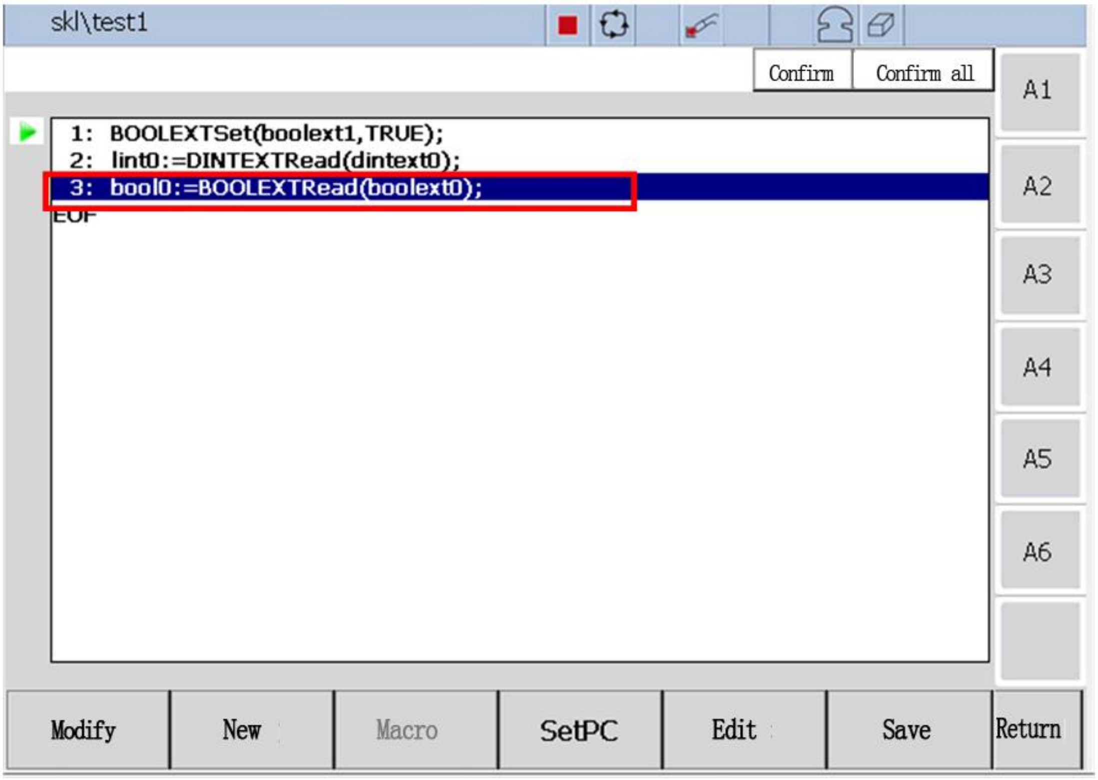
left operand: L lint0

:=

ext:DINTEXT: L dintext0

lint0:=DINTEXTRead(dintext0)

- Click OK and complete the statements shown in the following diagram.



```
1:  BOOLEXTSet(boolext1,TRUE);
2:  lint0:=DINTEXTRead(dintext0);
3:  bool0:=BOOLEXTRead(boolext0);
EOF
```

- **Statement format**

```
DINTVAR= DINTEXTRead(ext);
```

- **Parameter**

ext : BOOLEXT / /the structure, including the number and the value of the digital input signal port

- **Function description**

This instruction is used to read the specified variable in PLC. The return value is DINT.

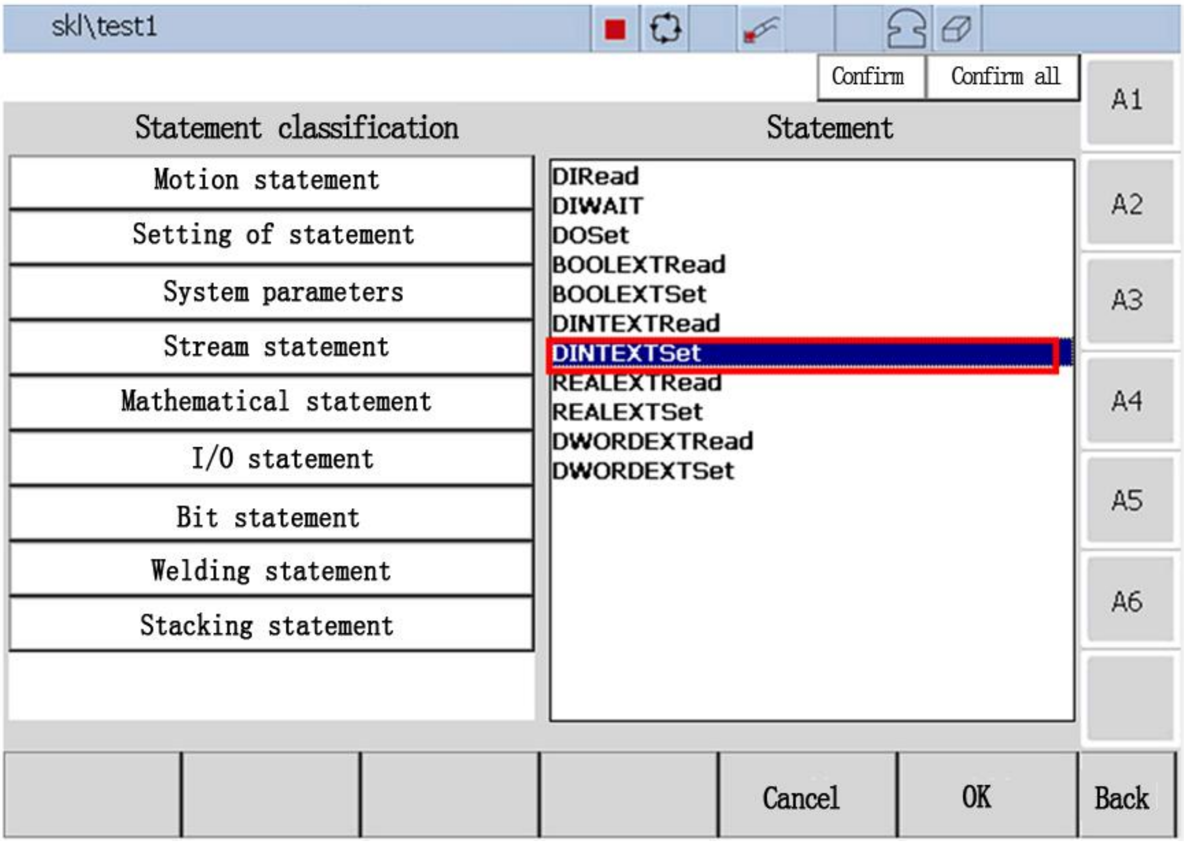
- **Sample**

bool0=DINTEXTRead(boolext0);

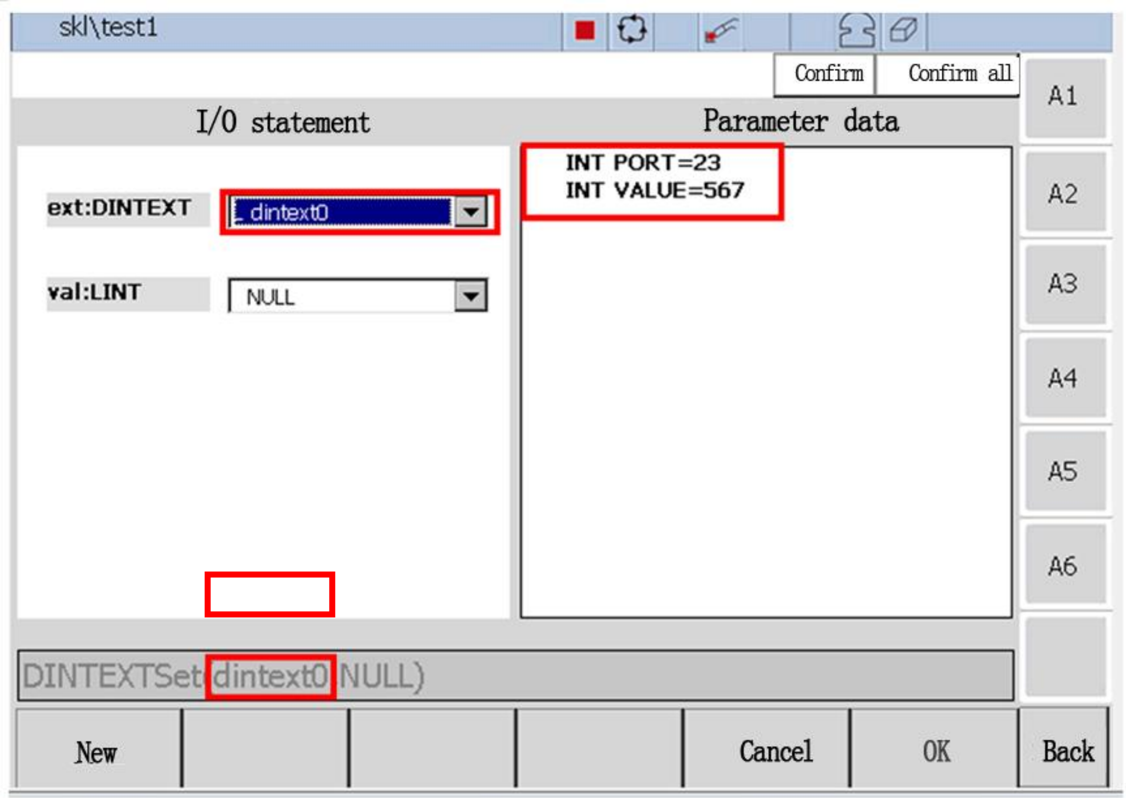
The program function is to assign the data in boolext0 to the variable bool0.

3.5.7 DINTEXTSet instruction

- Create DINTEXTSet as shown in the following diagram



- Click OK and enter the following interface to create variables and parameter setting.

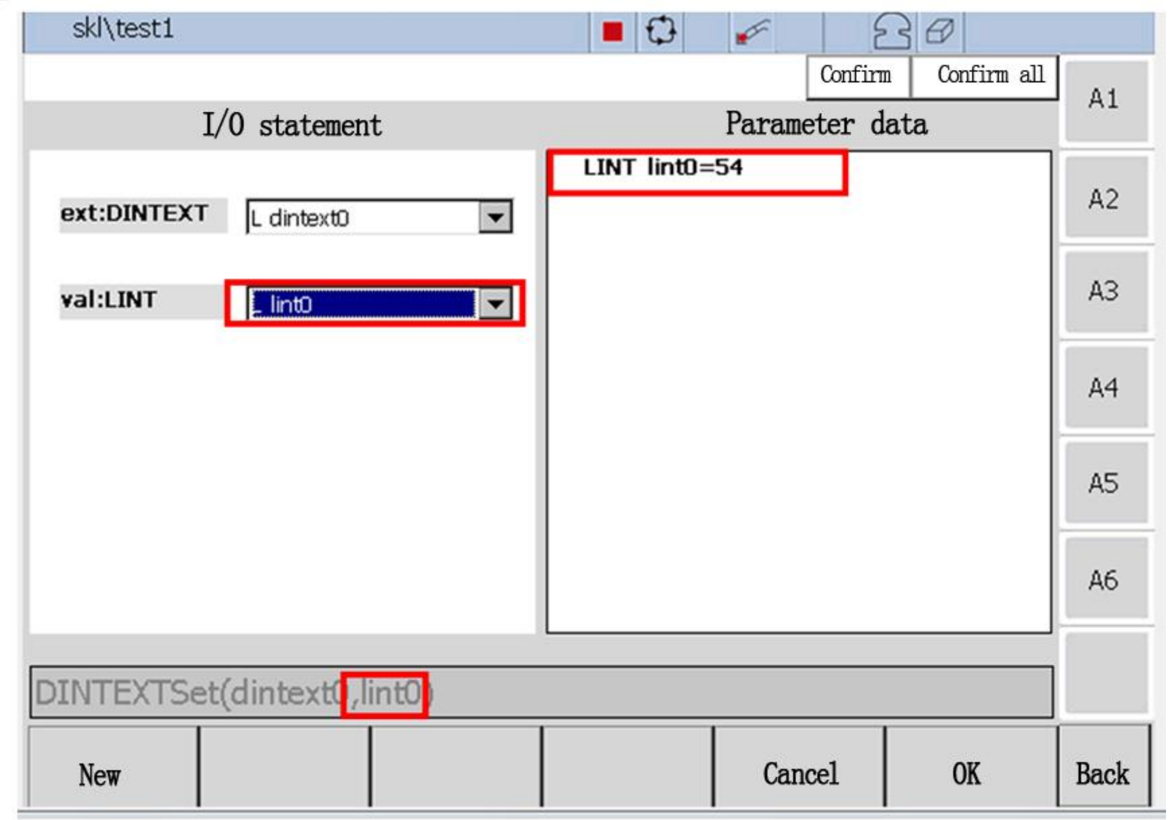


ext:DINTEXT L dintext0

val:LINT NULL

INT PORT=23
INT VALUE=567

DINTEXTSet(dintext0,NULL)



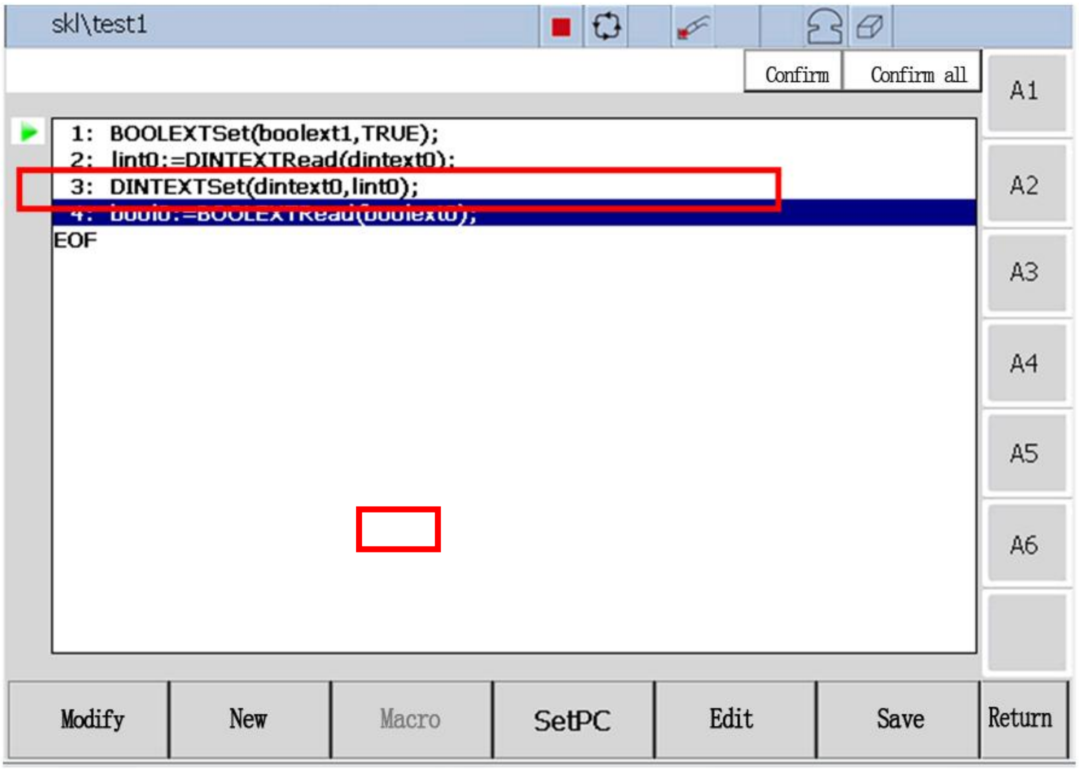
ext:DINTEXT L dintext0

val:LINT L lint0

LINT lint0=54

DINTEXTSet(dintext0,lint0)

- Click OK and complete the statements shown in the third line of the diagram.



```
1: BOOLEXTSet(boolext1,TRUE);
2: lint0:=DINTEXTRead(dintext0);
3: DINTEXTSet(dintext0,lint0);
4: bool0:=BOOLEXTRead(boolext0);
EOF
```

- **Statement format**

```
DINTEXTSet(ext, val);
```

- **Parameter**

ext : BOOLEXT / /the structure, including the number and the value of the digital input signal port

val：DINT(LINT)

- **Function description**

This instruction is used to set the value of the specified variable in PLC, in which val0 is DINT.

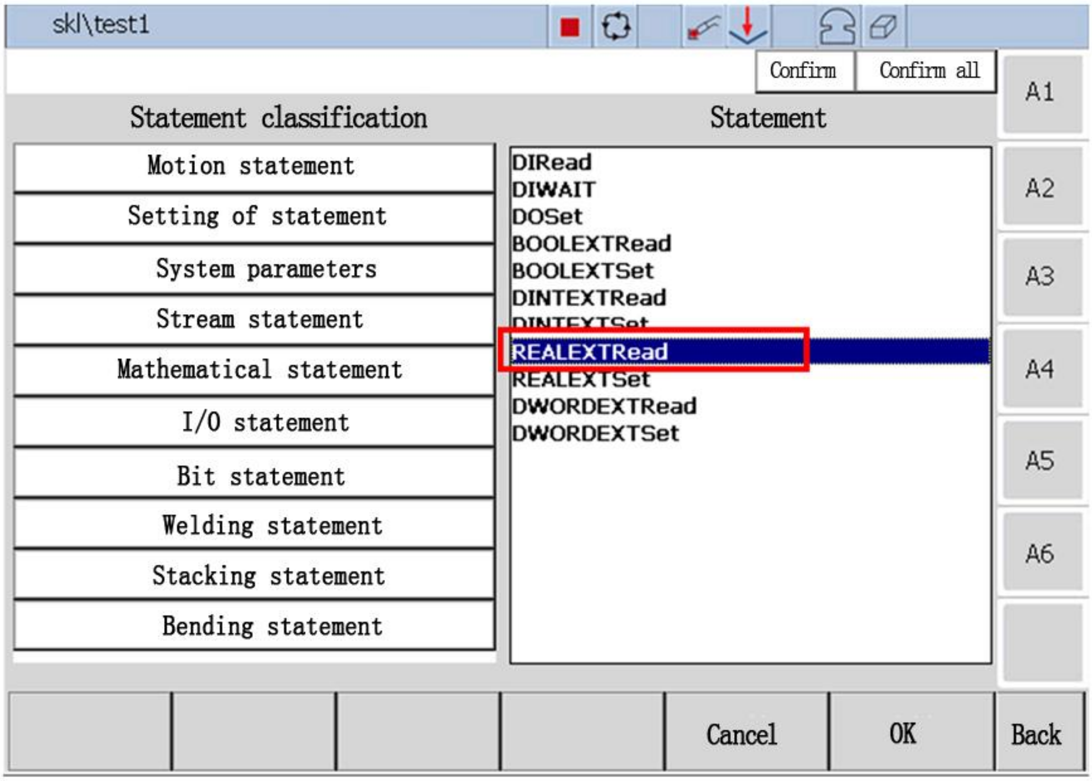
- **Sample**

DINTEXTSet(dintext0, lint0);

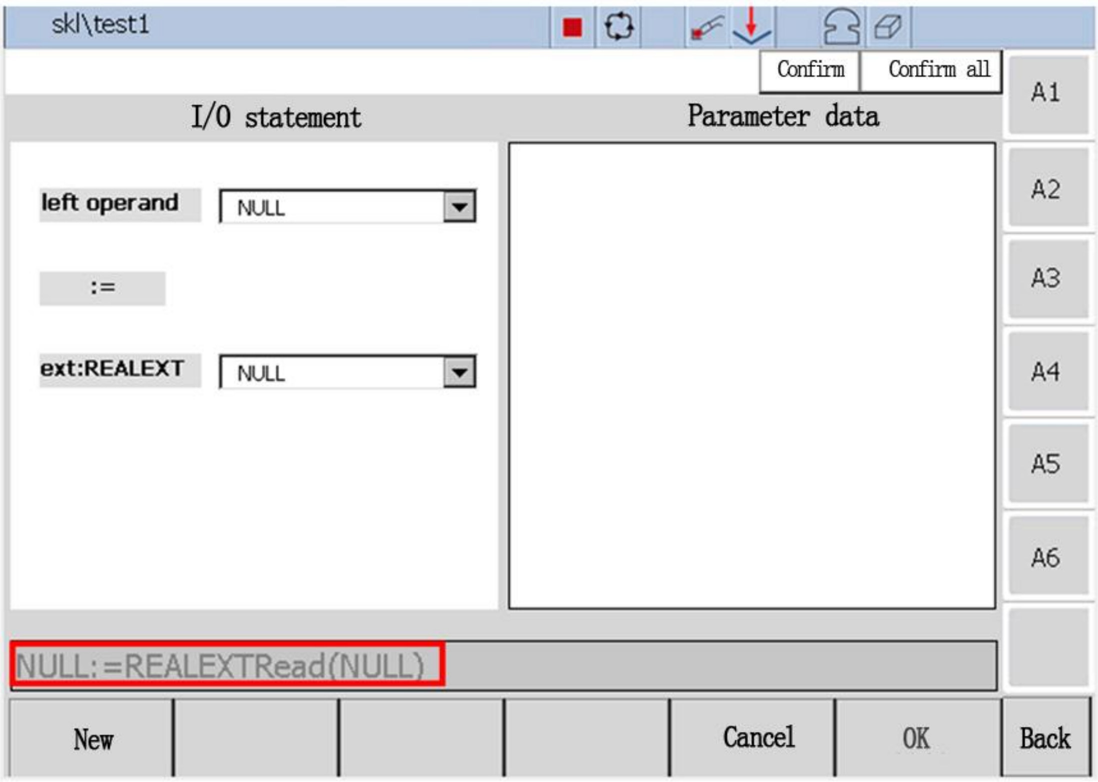
The program function is to assign the data in lint0 to the variable dintext0.

3.5.8 REALEXTRead instruction

- Create REALEXTRead as shown in the following diagram



- Click OK and enter the following interface



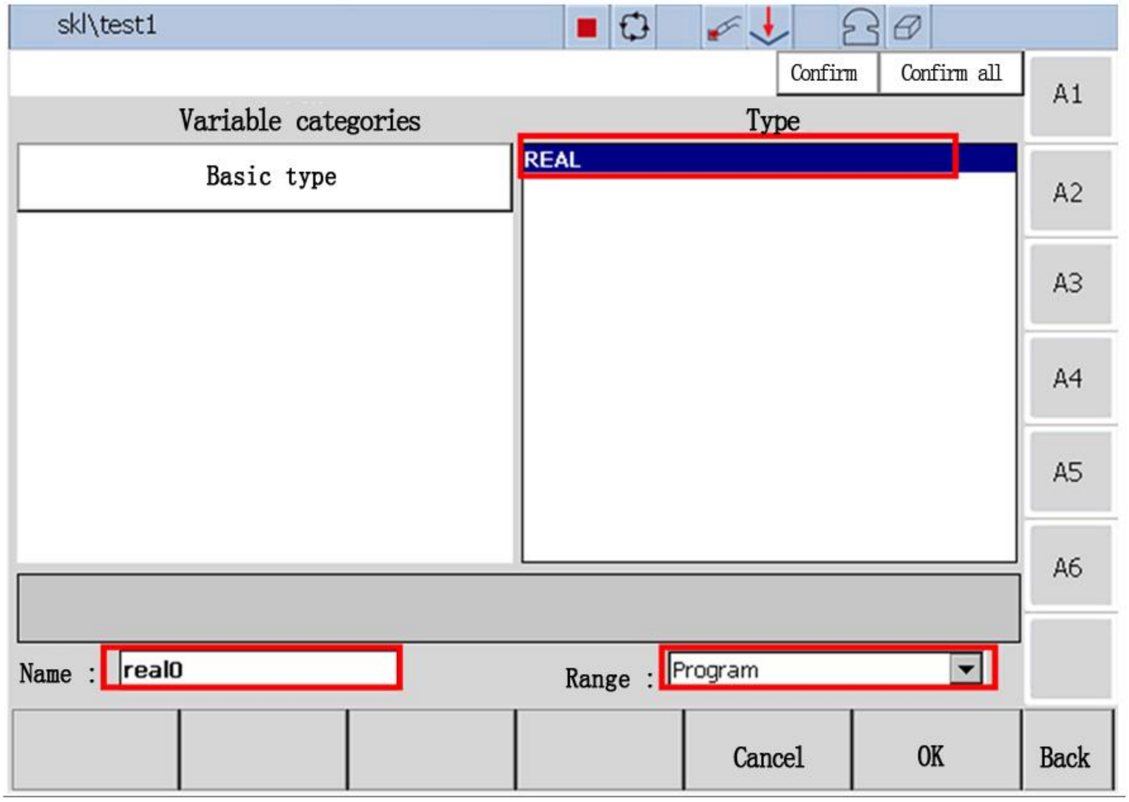
left operand NULL

:=

ext:REALEXT NULL

NULL:=REALEXTRead(NULL)

- Create the left side of variables and set the parameters.

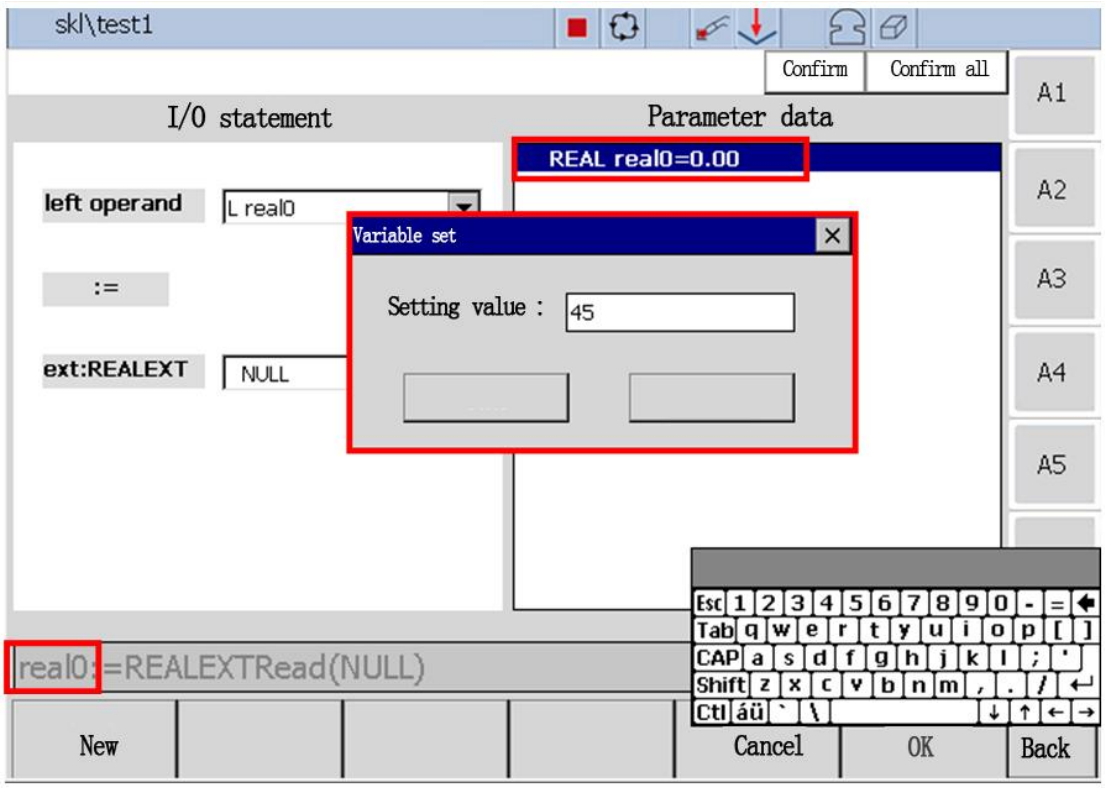


Variable categories: Basic type

Type: REAL

Name : real0

Range : Program


skl\test1
Confirm
Confirm all
A1
A2
A3
A4
A5
Variable set
Esc 1 2 3 4 5 6 7 8 9 0 - =
Tab q w e r t y u i o p [ ]
CAP a s d f g h j k l ; '
Shift z x c v b n m , . /
Ctl áü ` \ ↓ ↑ ← →
New
Cancel
OK
Back

I/O statement

left operand: L real0

:=

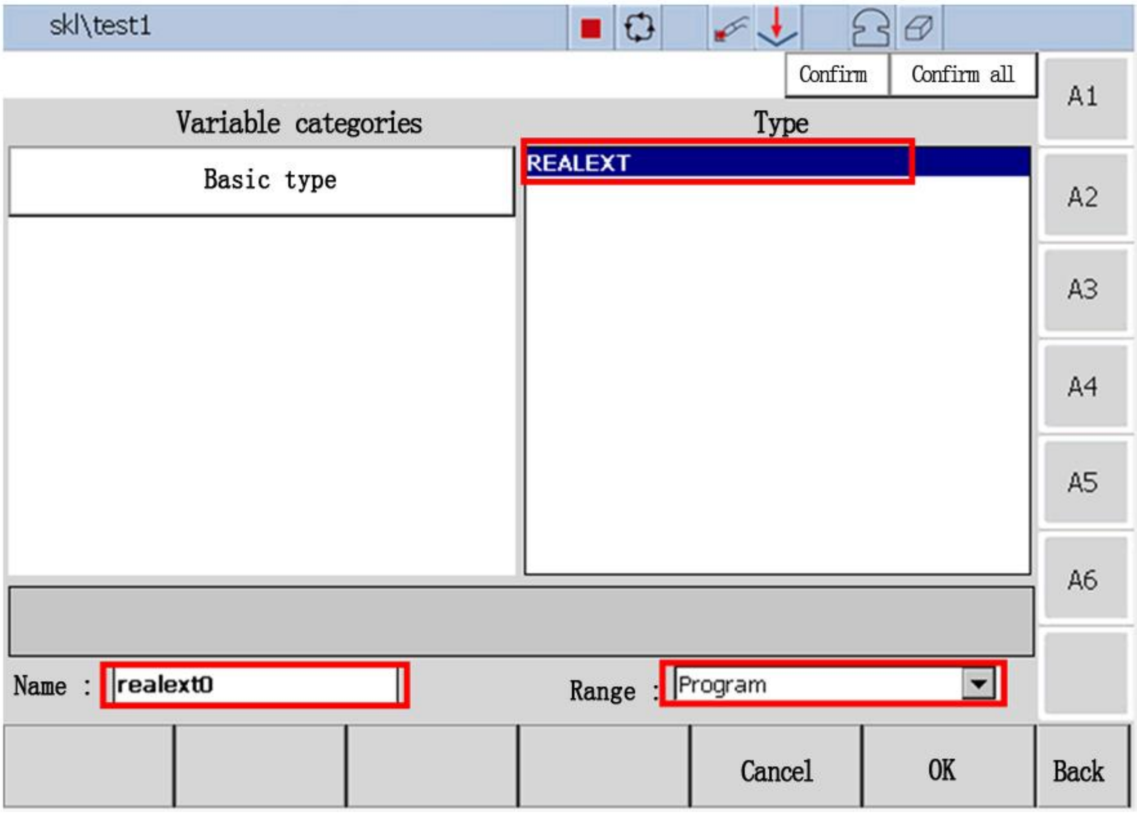
ext:REALEXT: NULL

Parameter data: REAL real0=0.00

Setting value : 45

real0:=REALEXTRead(NULL)

- Create the right side of variables and set the parameters.

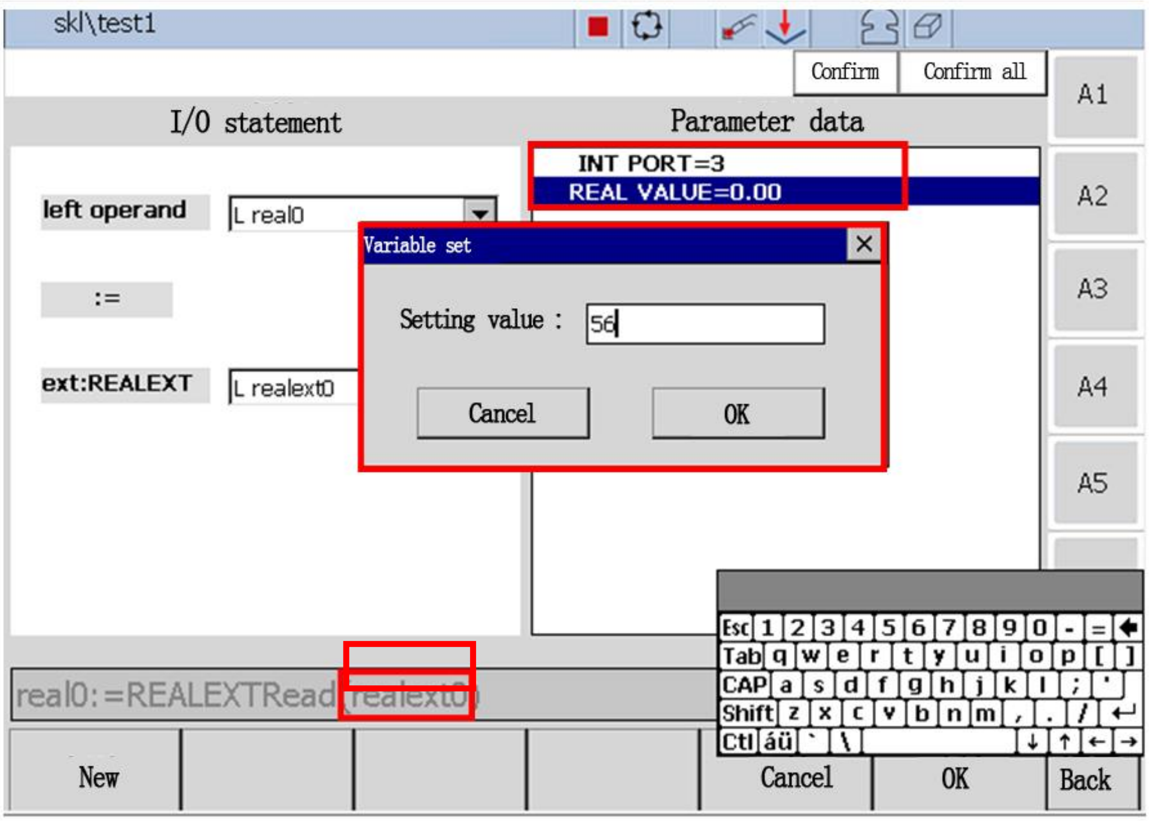


Variable categories: Basic type

Type: REALEXT

Name : realext0

Range : Program


skl\test1
Confirm
Confirm all
A1
A2
A3
A4
A5
Esc 1 2 3 4 5 6 7 8 9 0 - =
Tab q w e r t y u i o p [ ]
CAP a s d f g h j k l ; '
Shift z x c v b n m , . /
Ctl áü ` \ ↓ ↑ ← →
New
Cancel
OK
Back

I/O statement

left operand: L real0

:=

ext:REALEXT: L realext0

Parameter data:

INT PORT=3

REAL VALUE=0.00

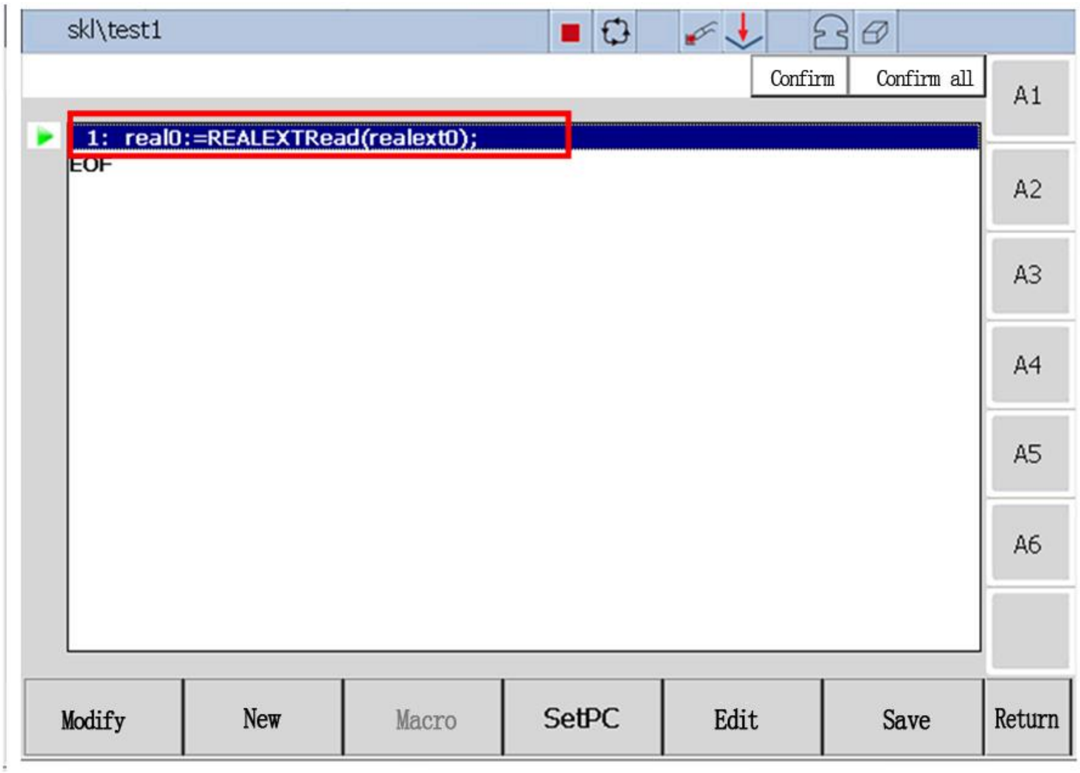
Variable set

Setting value : 56

Cancel OK

real0:=REALEXTRead(realext0)

- Click OK and complete the statement as shown in the following diagram.



```
1:  real0:=REALEXTRead(realext0);
EOF
```

- ➢ **Statement format**

```
REALVAR= REALEXTRead(ext);
```

- ➢ **Parameter**

ext : BOOLEXT / /the structure, including the number and the value of the digital input signal port

- ➢ **Function description**

This instruction is used to read the specified variable in PLC. The return value is REAL.

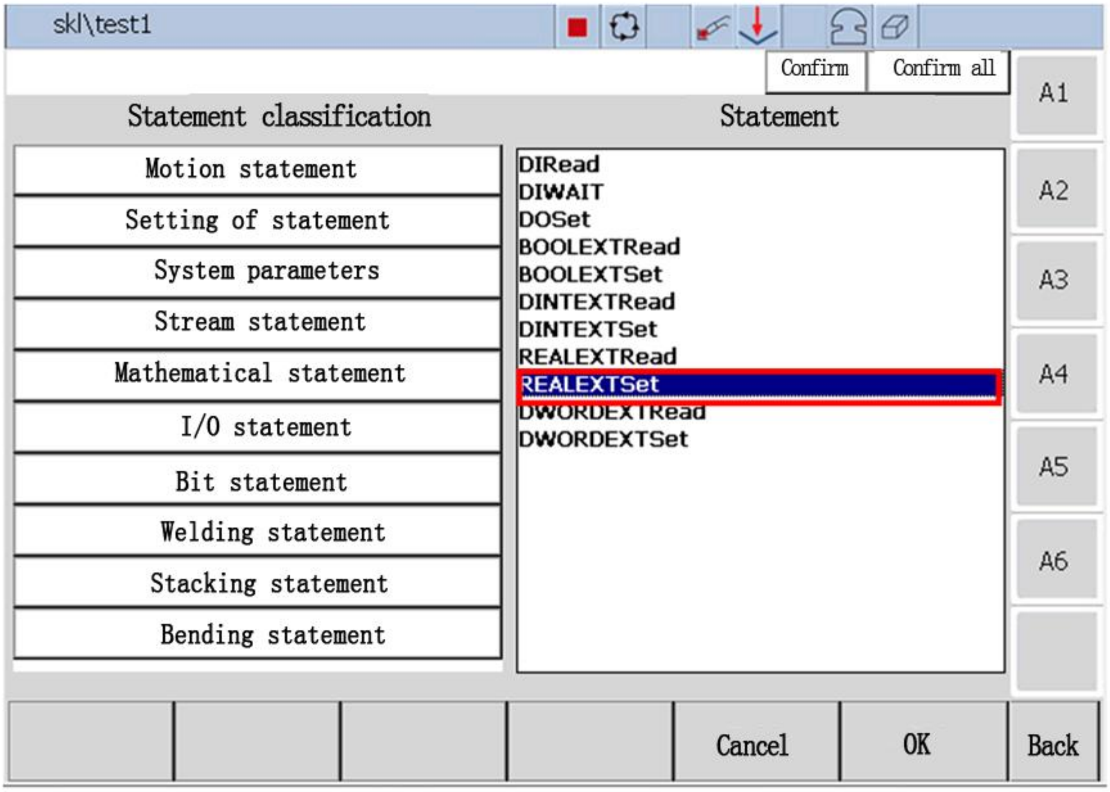
- ➢ **Sample**

real0= REALEXTRead(realext0);

The program function is to assign the data in realext0 to real0.

3.5.9 REALEXTSet instruction

- ● Create REALEXTSet as shown in the following diagram


skl\test1

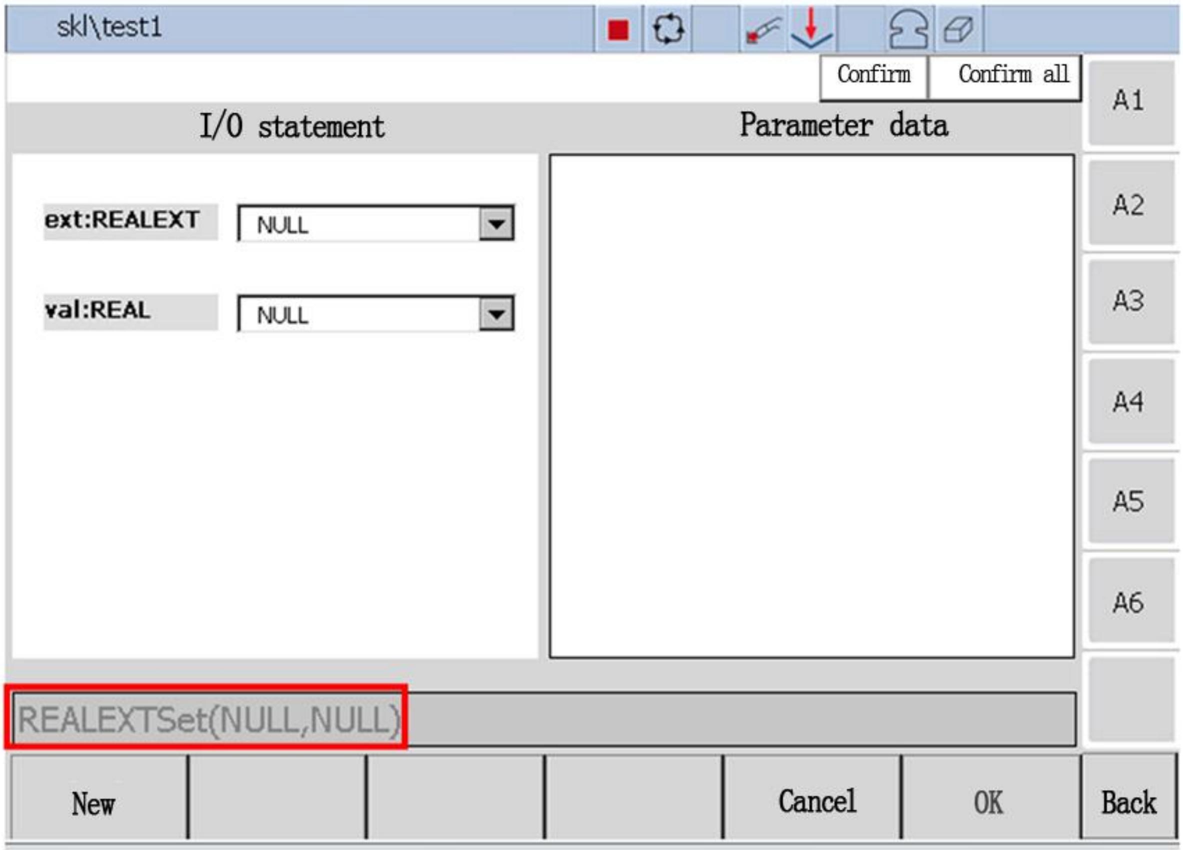
Statement classification

- Motion statement
- Setting of statement
- System parameters
- Stream statement
- Mathematical statement
- I/O statement
- Bit statement
- Welding statement
- Stacking statement
- Bending statement

Statement

- DIRead
- DIWAIT
- DOSet
- BOOLEXTRead
- BOOLEXTSet
- DINTEXTRead
- DINTEXTSet
- REALEXTRead
- REALEXTSet
- DWORDEXTRead
- DWORDEXTSet

- Click OK and enter the following interface


skl\test1

I/O statement

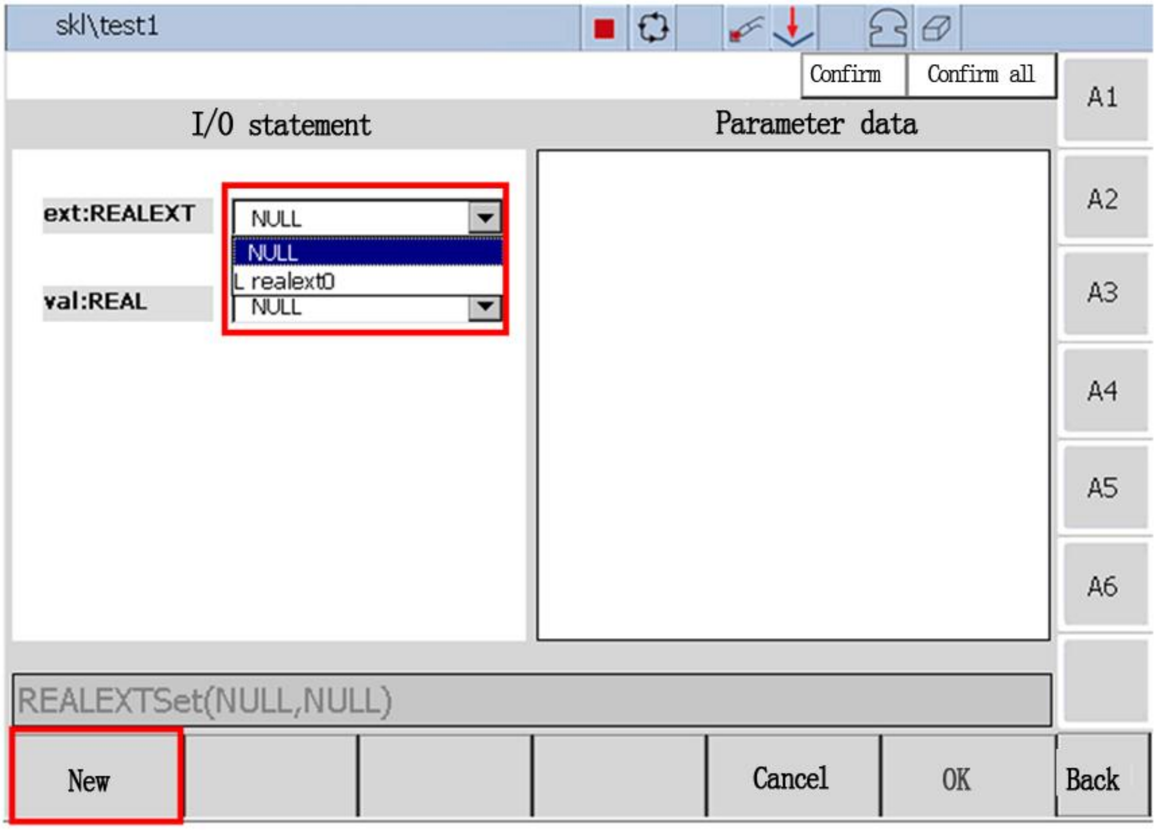
ext:REALEXT NULL

val:REAL NULL

Parameter data

REALEXTSet(NULL,NULL)

- Create the first variable and set the parameter

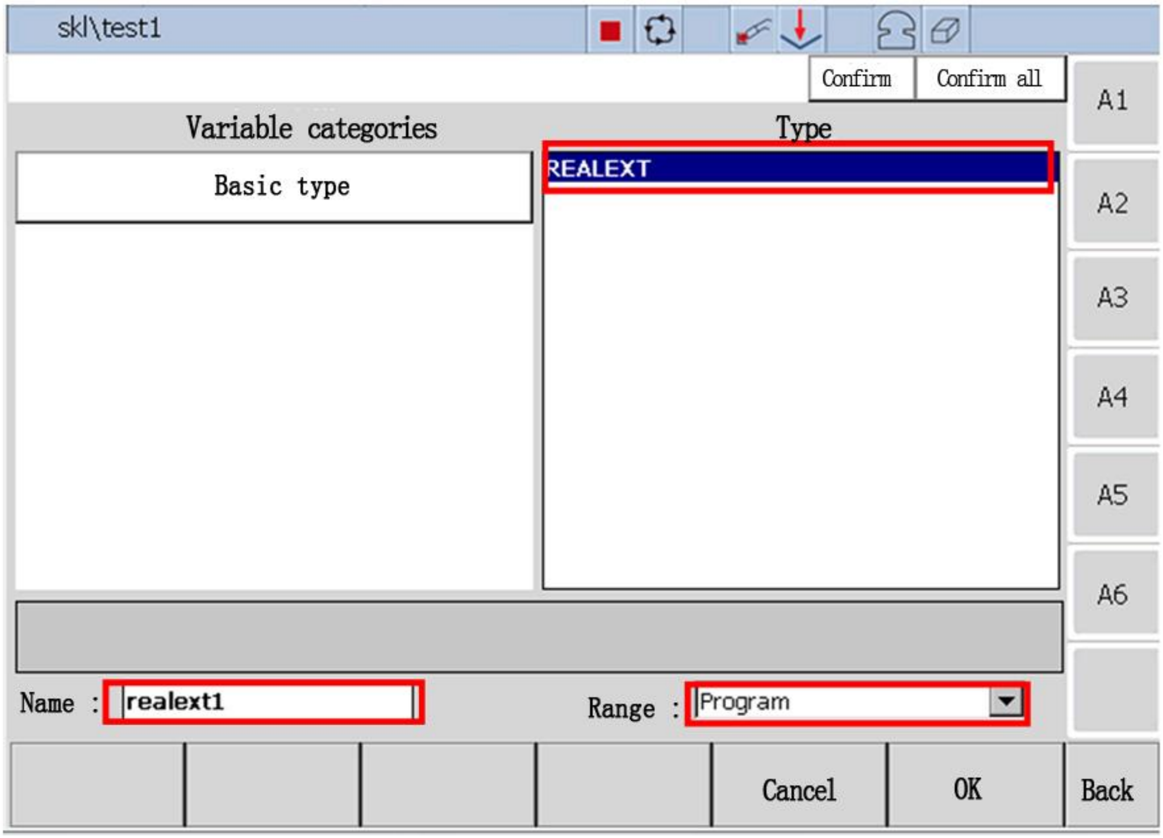


ext:REALEXT NULL

NULL
L realext0

val:REAL NULL

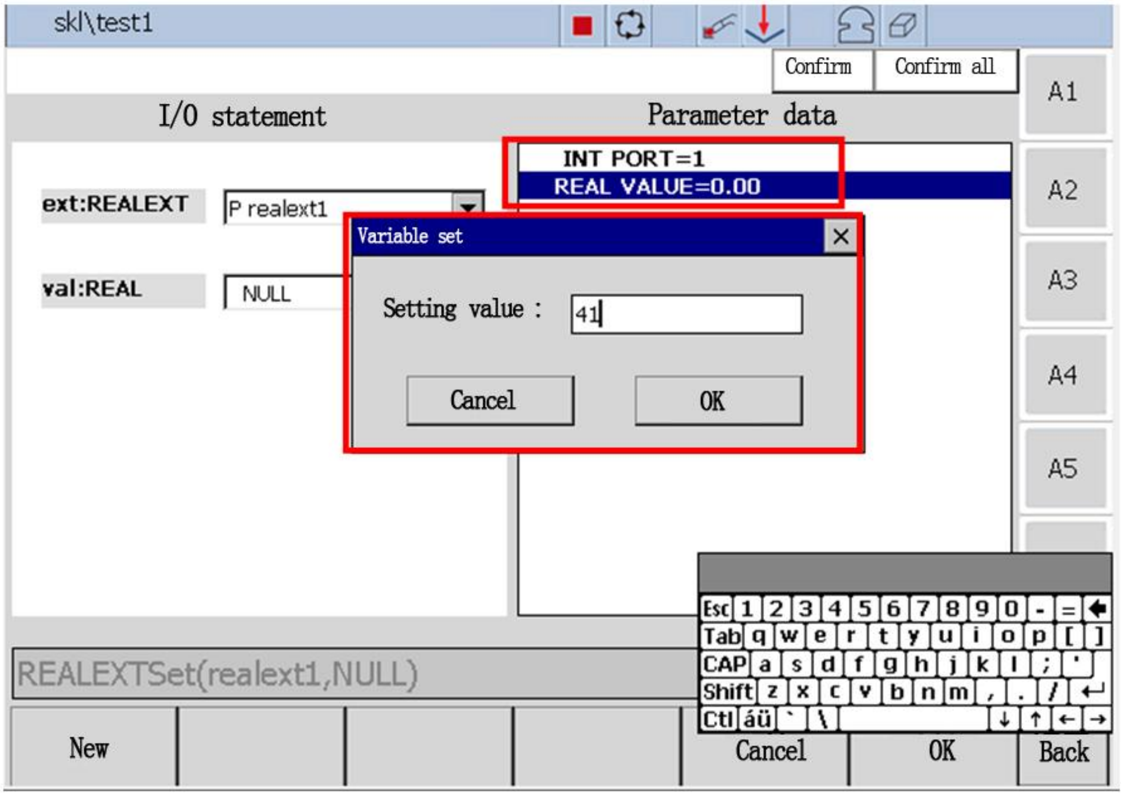
REALEXTSet(NULL,NULL)



REALEXT

Name : realext1

Range : Program



skl\test1

I/O statement

Parameter data

INT PORT=1
REAL VALUE=0.00

ext:REALEXT P realext1

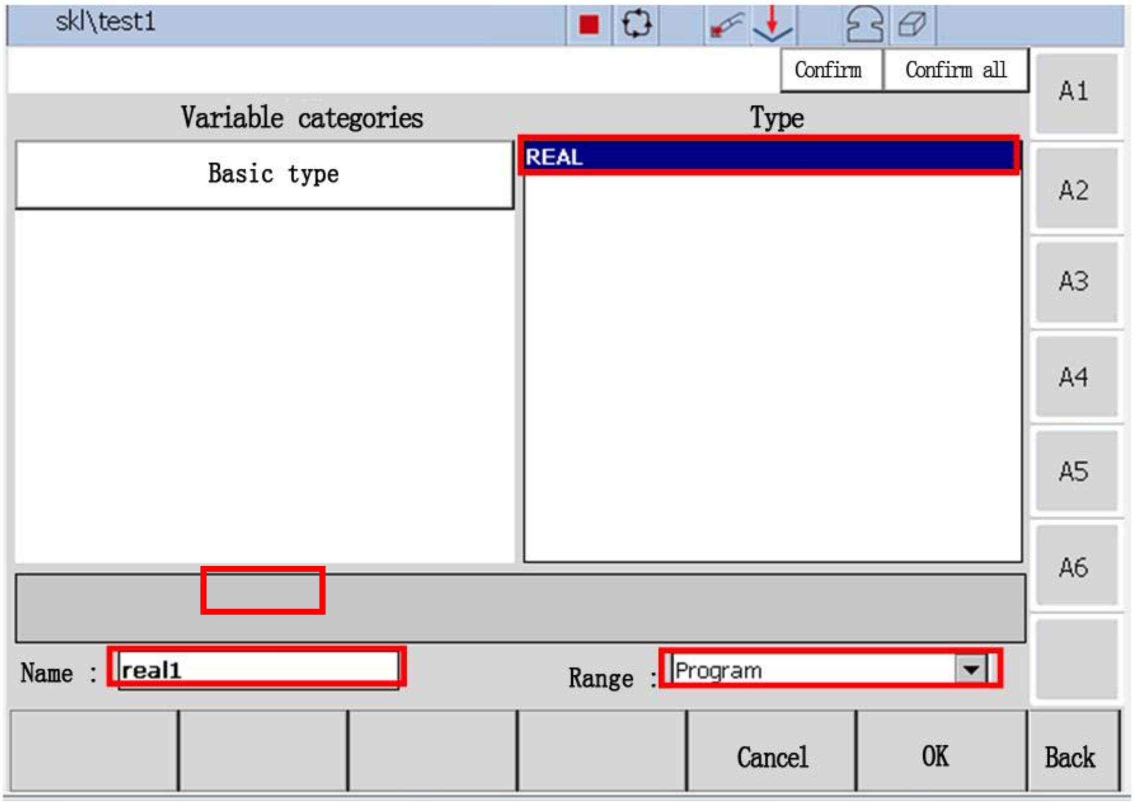
val:REAL NULL

Variable set

Setting value： 41

REALEXTSet(realext1,NULL)

- Create the second variable and set the parameter



skl\test1

Variable categories

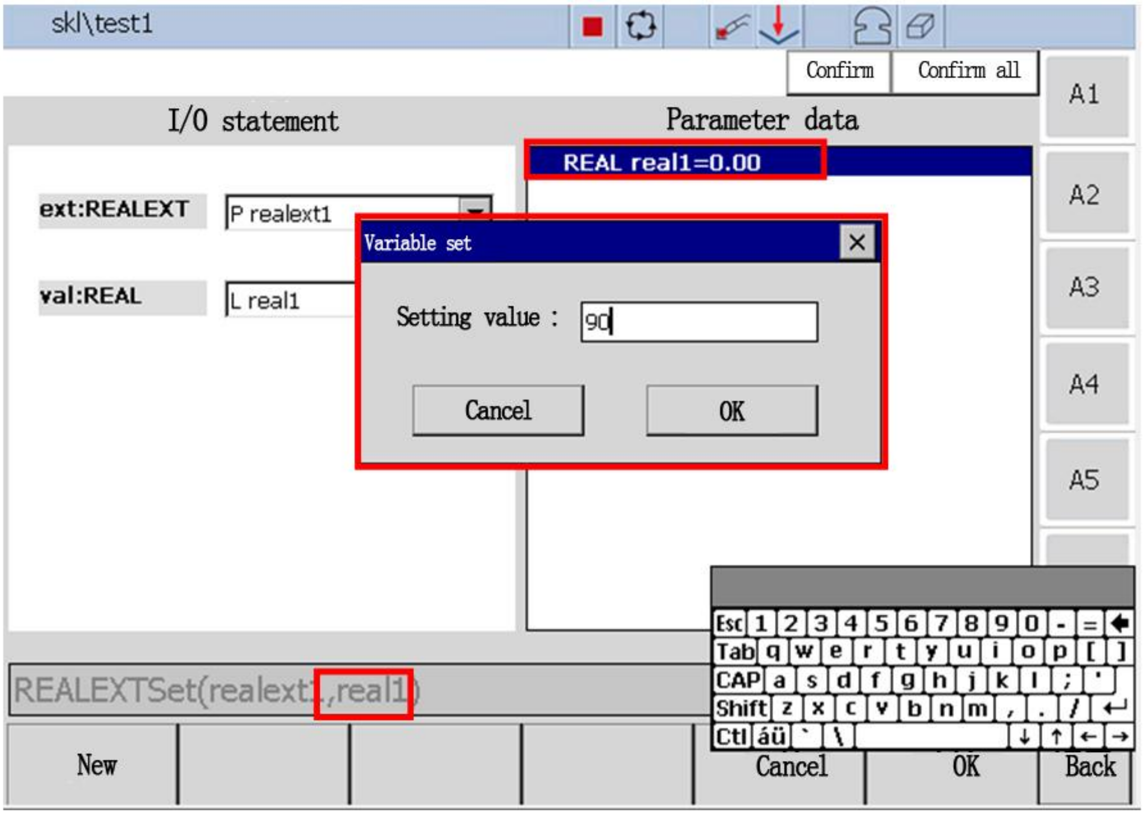
Type

Basic type

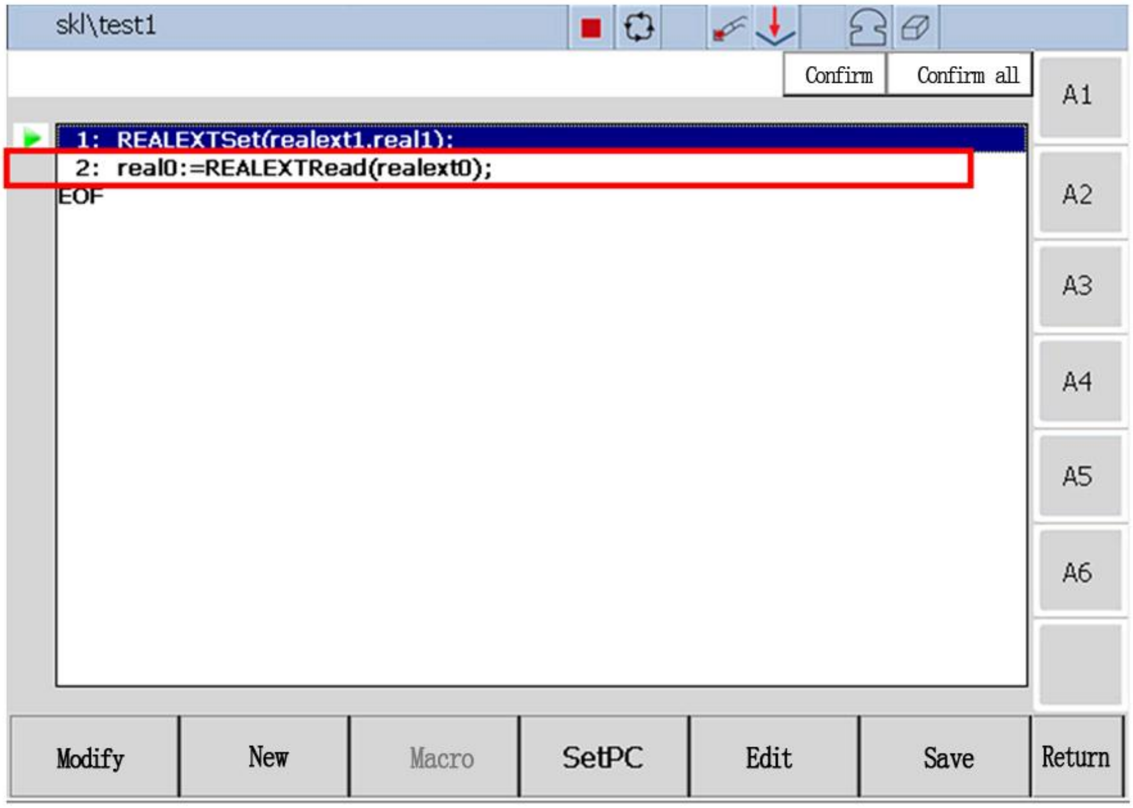
REAL

Name ： real1

Range ： Program



- Click OK and complete the statement



```
1:  REALEXTSet(realext1,real1);
2:  real0:=REALEXTRead(realext0);
EOF
```

➢ **Statement format**

```
REALEXTSet(ext, val);
```

- **Parameter**

ext : BOOLEXT / /the structure, including the number and the value of the digital input signal port

val：REAL

- **Function description**

This instruction is used to set the specified variable in PLC.

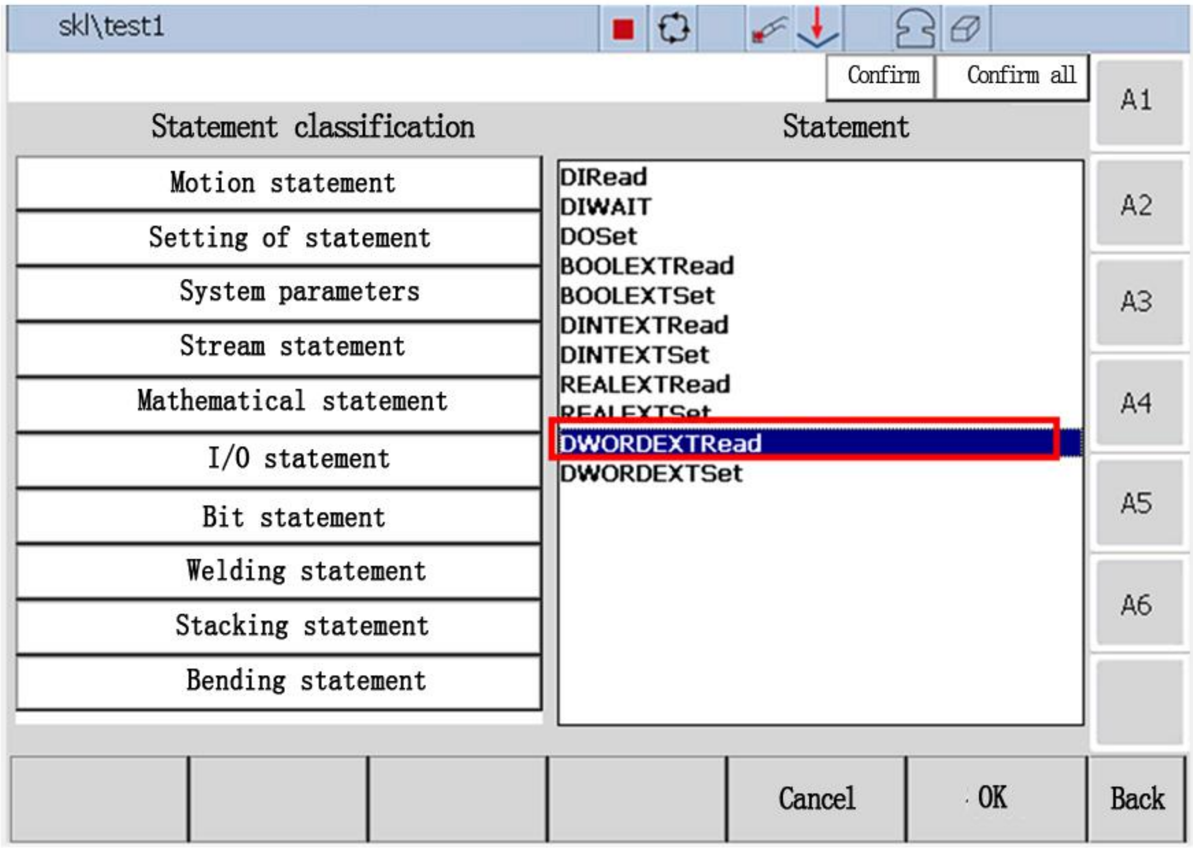
- **Sample**

REALEXTSet(realext1, real1);

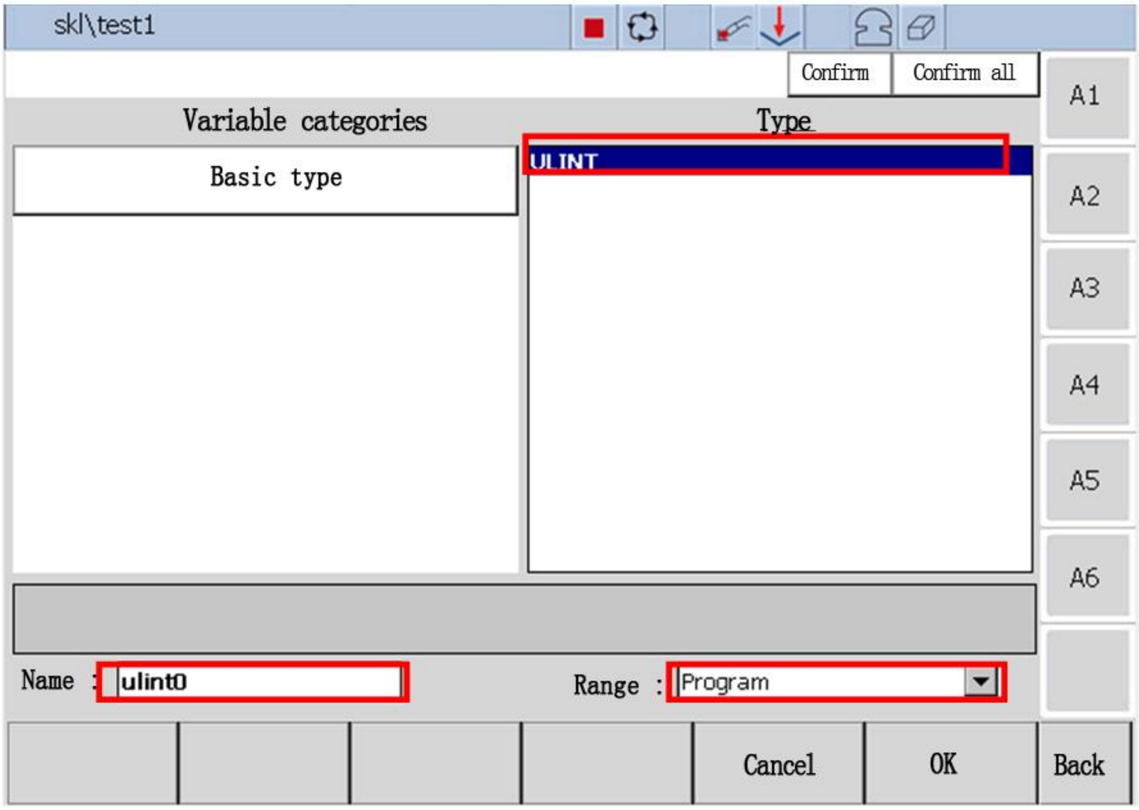
The program function is to assign the value of real1 to realext1.

3.5.10 DWORDEXTRead instruction

- Create DWORDEXTRead



- Click OK. And firstly create the first variable and parameter setting.

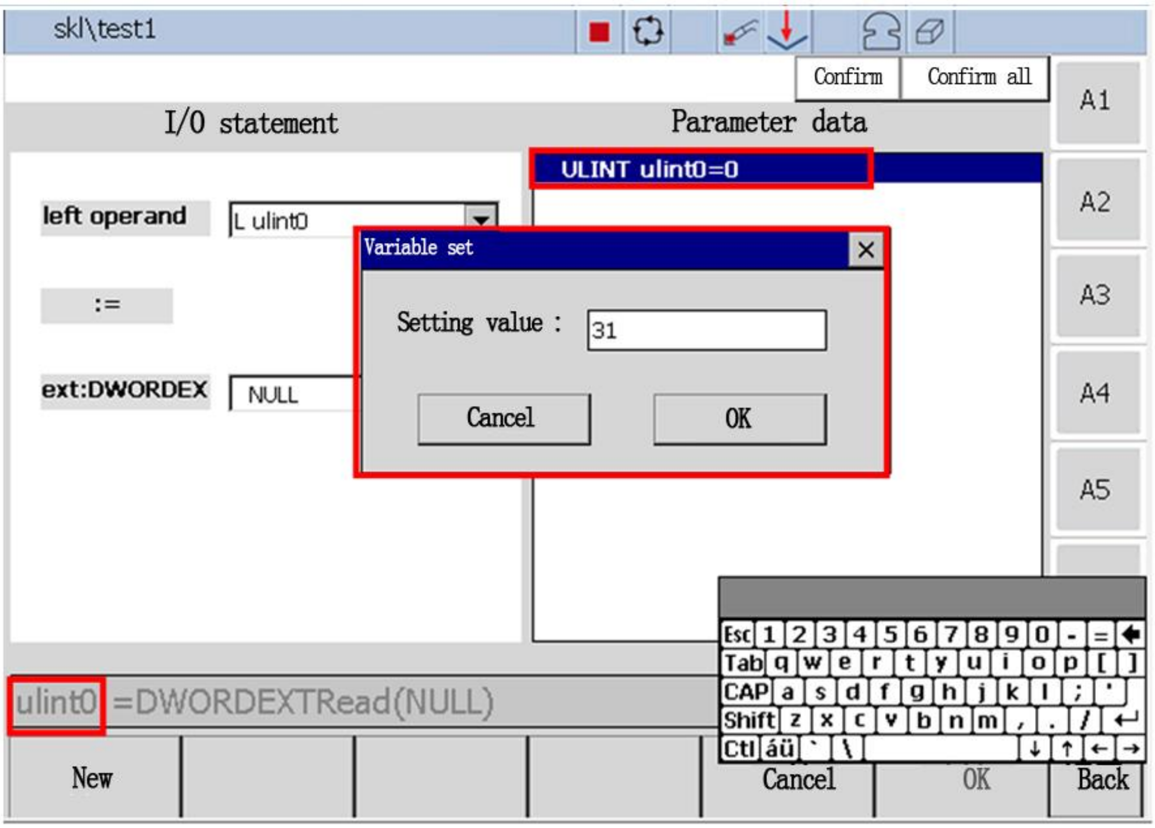


skl\test1

Type: ULINT

Name : ulint0

Range : Program



skl\test1

ULINT ulint0=0

left operand: L ulint0

:=

ext:DWORDEX: NULL

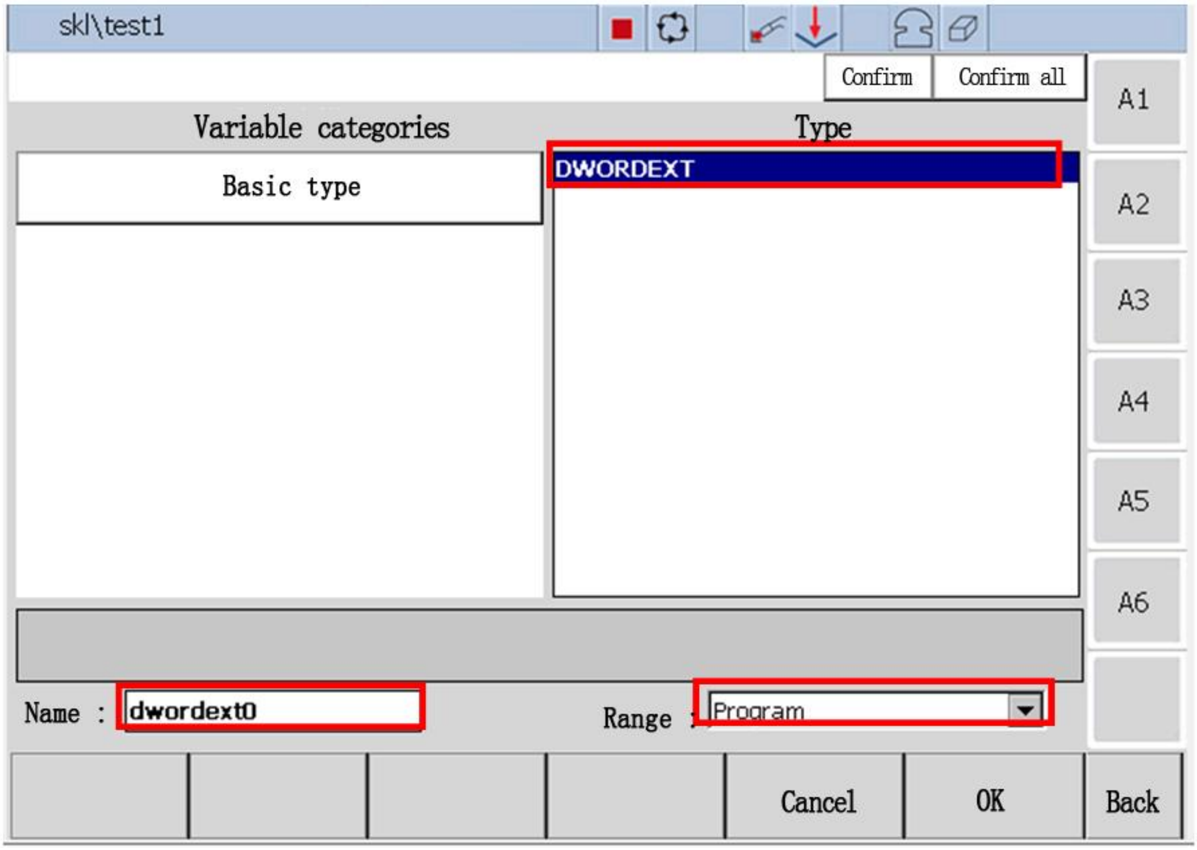
Variable set

Setting value : 31

ulint0 =DWORDEXTRead(NULL)

Confirm
Confirm all
I/O statement
Parameter data
Cancel
OK
A1
A2
A3
A4
A5
Esc 1 2 3 4 5 6 7 8 9 0 - =
Tab q w e r t y u i o p [ ]
CAP a s d f g h j k l ; '
Shift z x c v b n m , . / ↵
Ctl áü ` \ ↓ ↑ ← →
New
Cancel
OK
Back

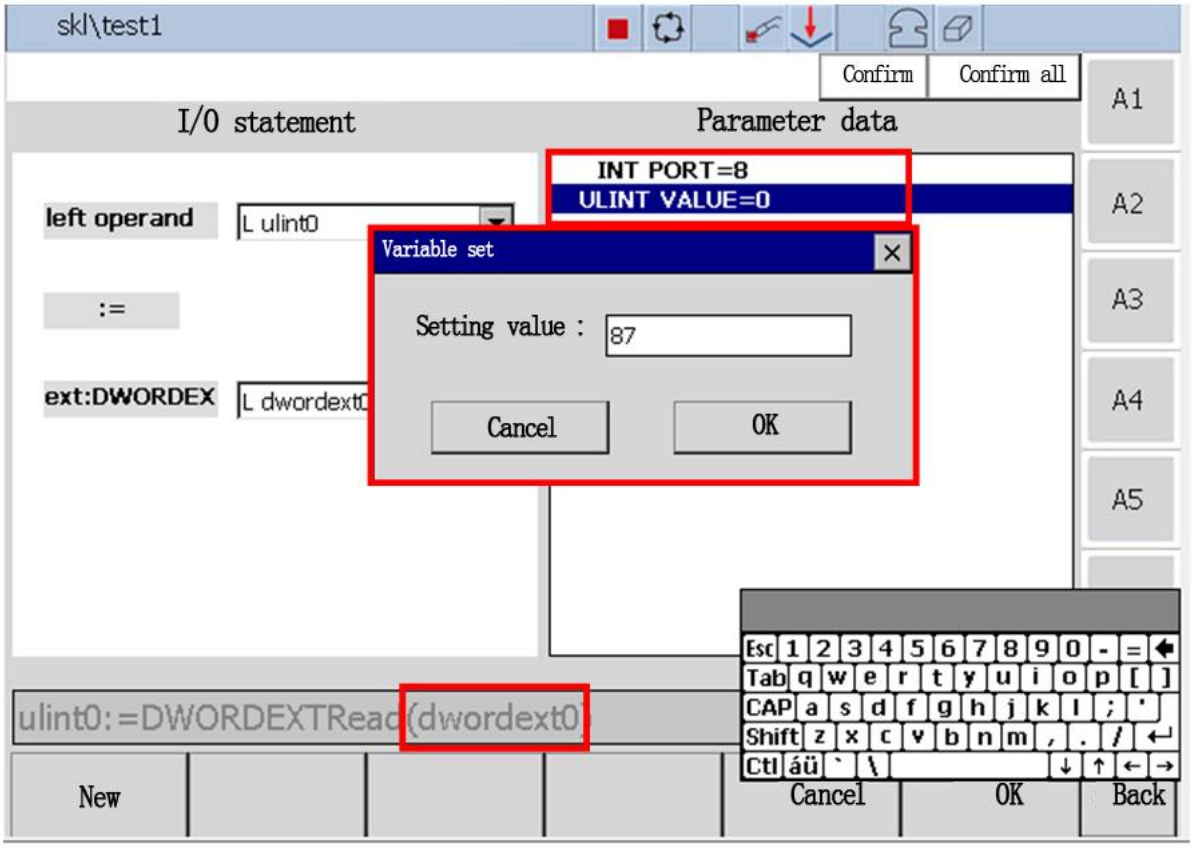
- Create the second variable and parameter setting



DWORDEXT

Name : dwordext0

Range : Program



left operand L ulint0

:=

ext:DWORDEX L dwordext0

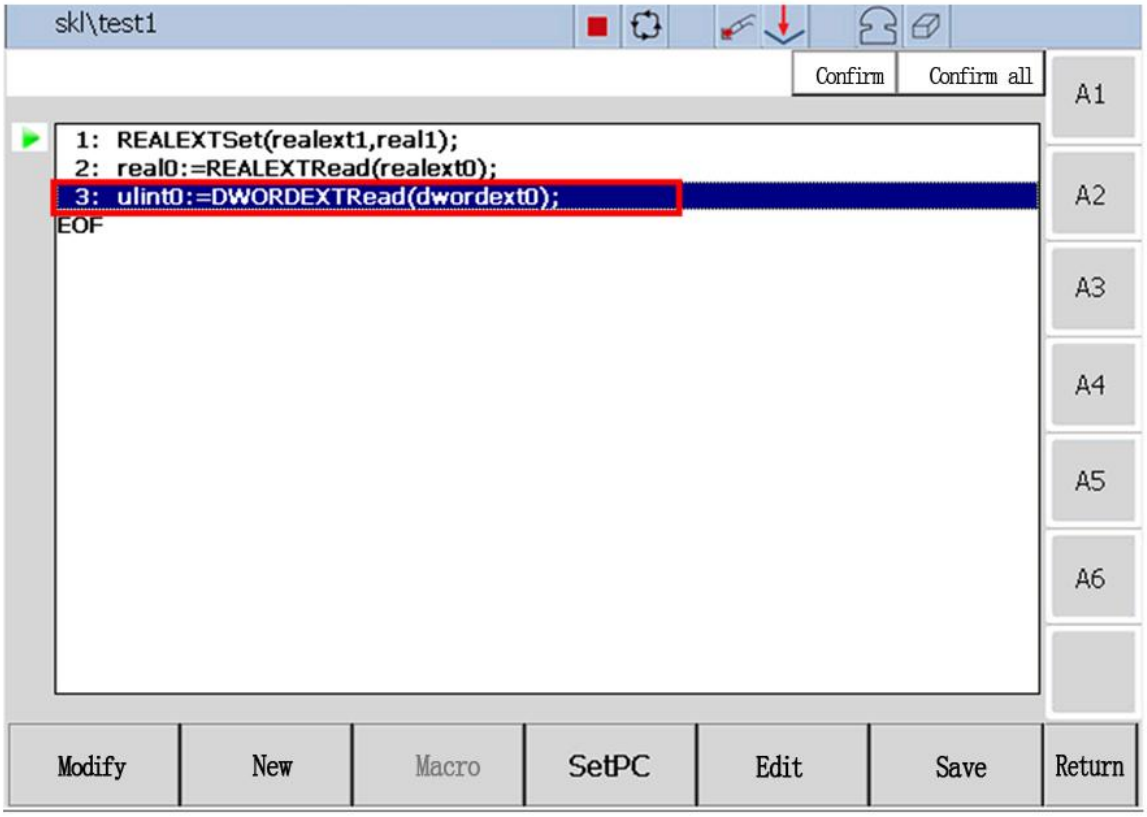
INT PORT=8

ULINT VALUE=0

Setting value : 87

ulint0:=DWORDEXTRead(dwordext0)

- Click OK and complete the creation of the statement.



skl\test1

```
1:  REALEXTSet(realext1,real1);
2:  real0:=REALEXTRead(realext0);
3:  ulint0:=DWORDEXTRead(dwordext0);
EOF
```

- **Statement format**

```
DWORDVAR= DWORDEXTRead(ext);
```

- **Parameter**

ext : BOOLEXT / /the structure, including the number and the value of the digital input signal port

- **Function description**

This instruction is used to read the specified variable in PLC. The return value is DWORD.

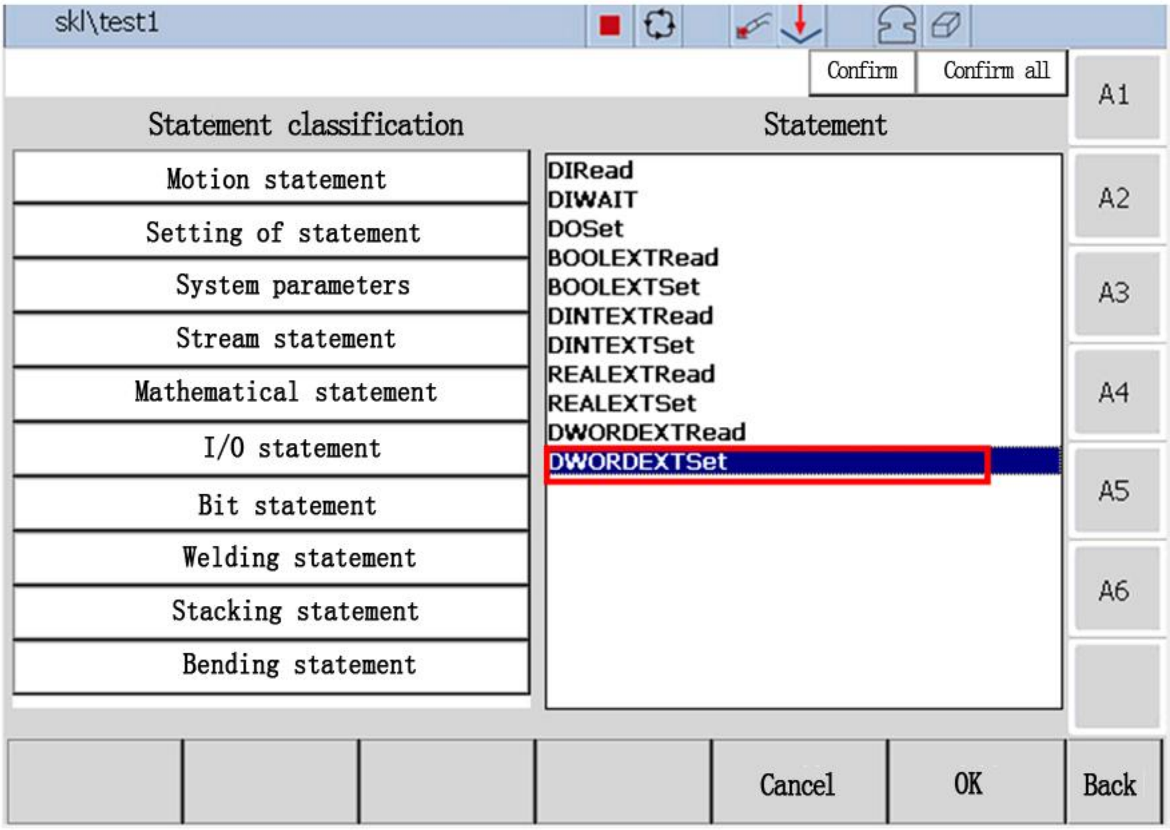
- **Sample**

ulint0= DWORDEXTRead(dwordext0);

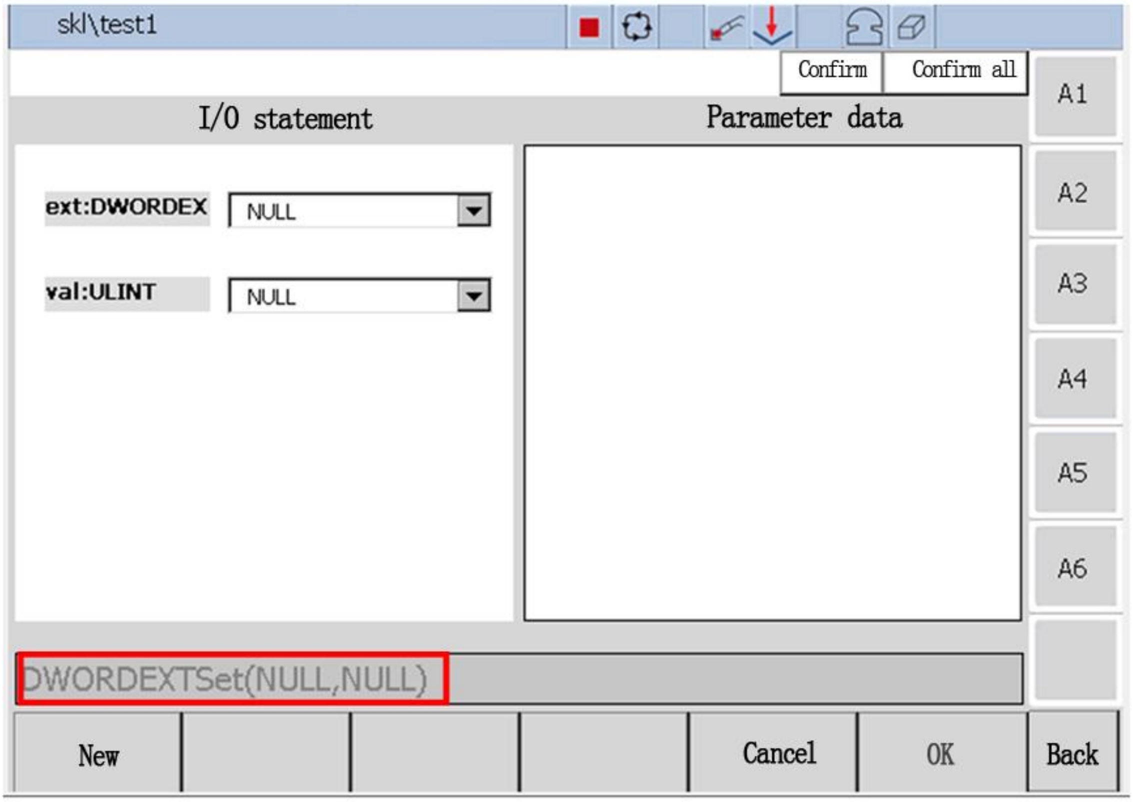
The program function is to assign the data in dwordext0 to ulint0.

3.5.11 DWORDEXTSet instruction

- Create DWORDEXTRead as shown in the following diagram.



- Click OK and enter the following interface.

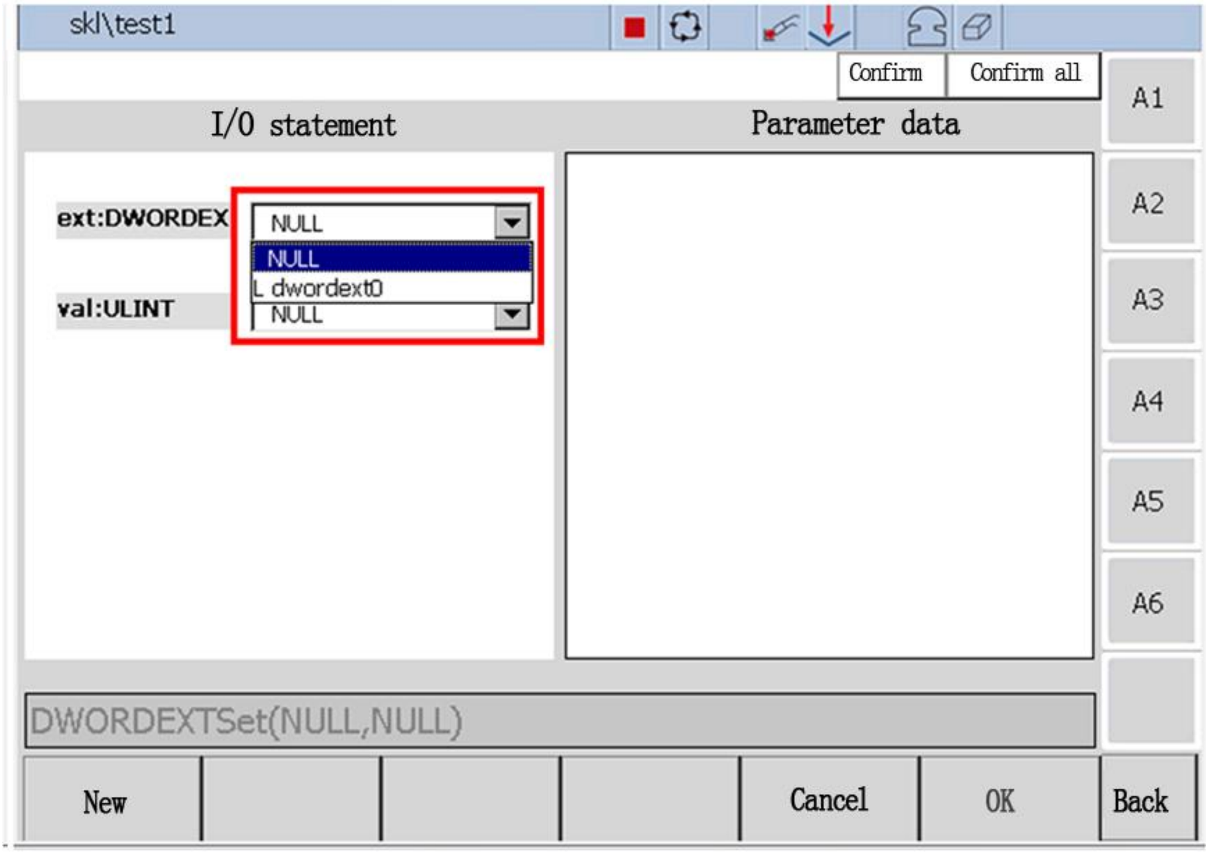


ext:DWORDEX NULL

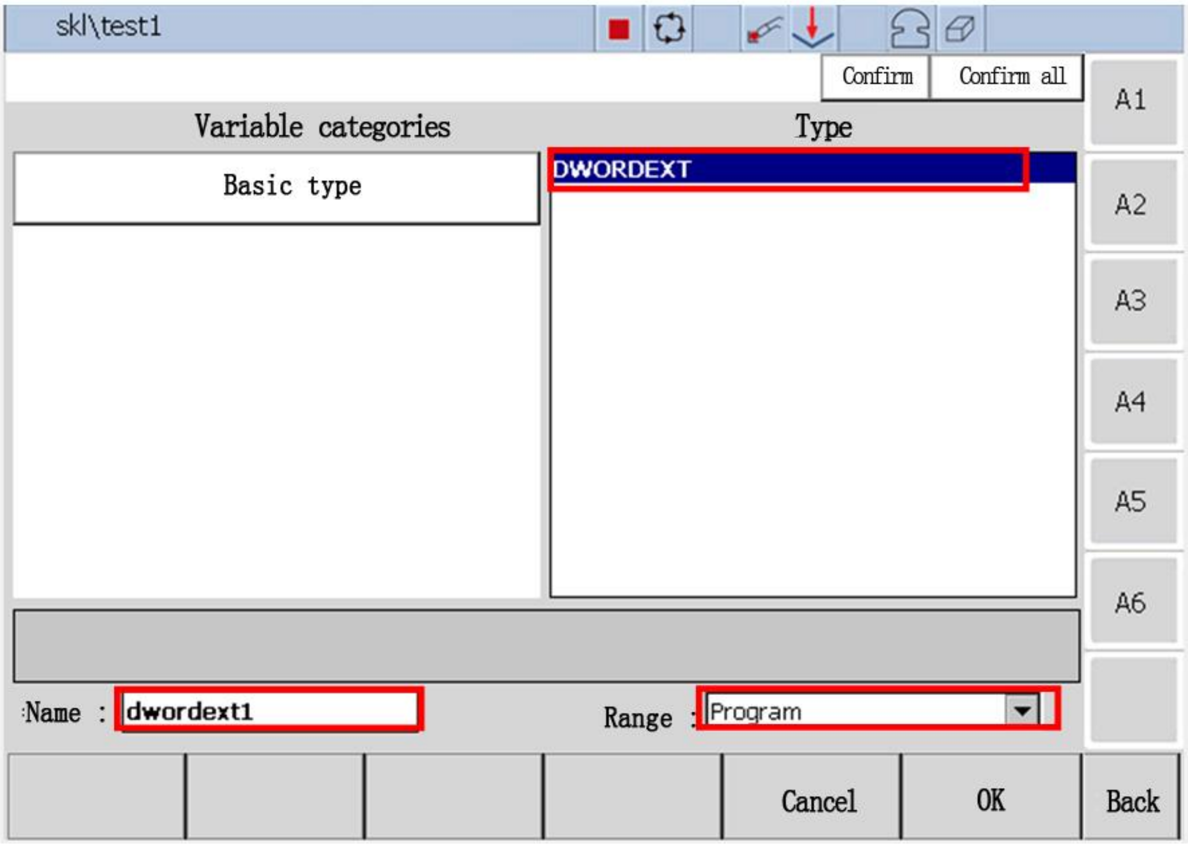
val:ULINT NULL

DWORDEXTSet(NULL,NULL)

- Create the first variable and set the parameter

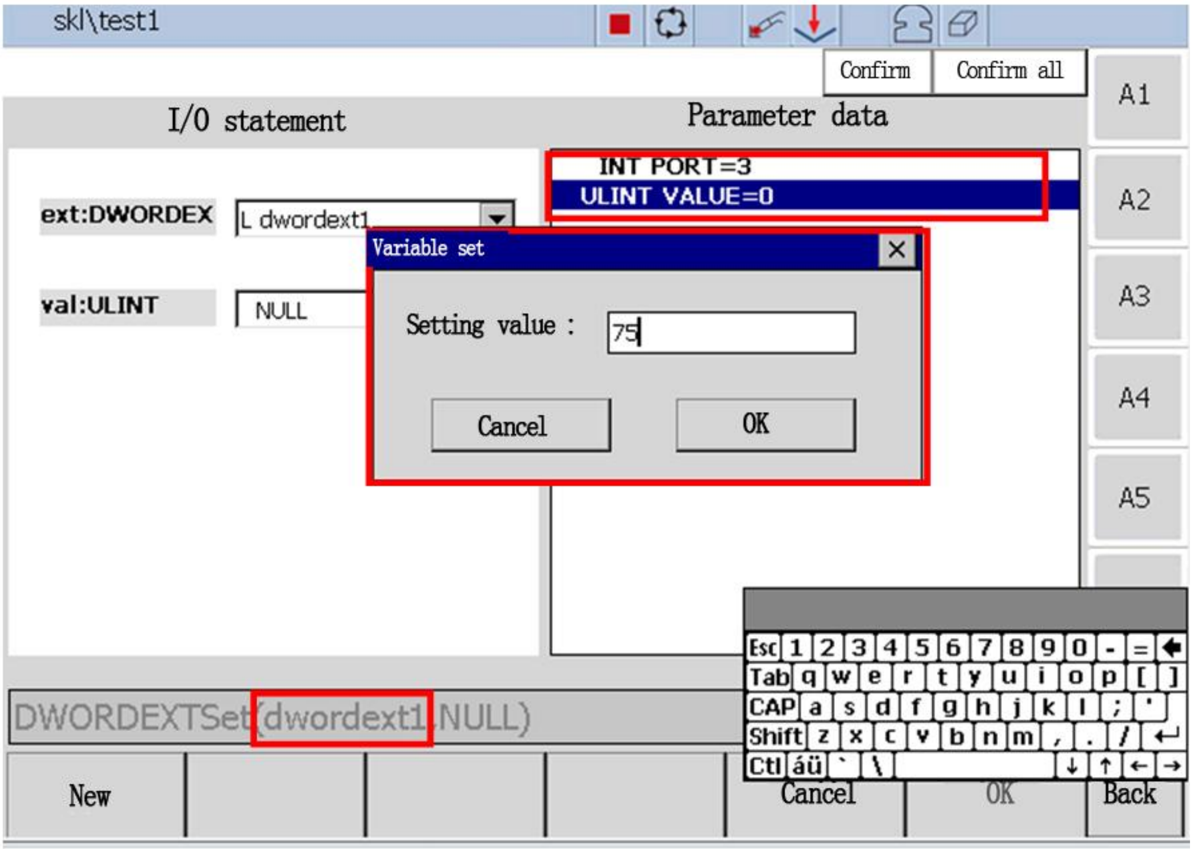


DWORDEXTSet(NULL,NULL)



Name : dwordext1

Range : Program



skl\test1

I/0 statement

Parameter data

ext:DWORDEX L dwordext1

val:ULINT NULL

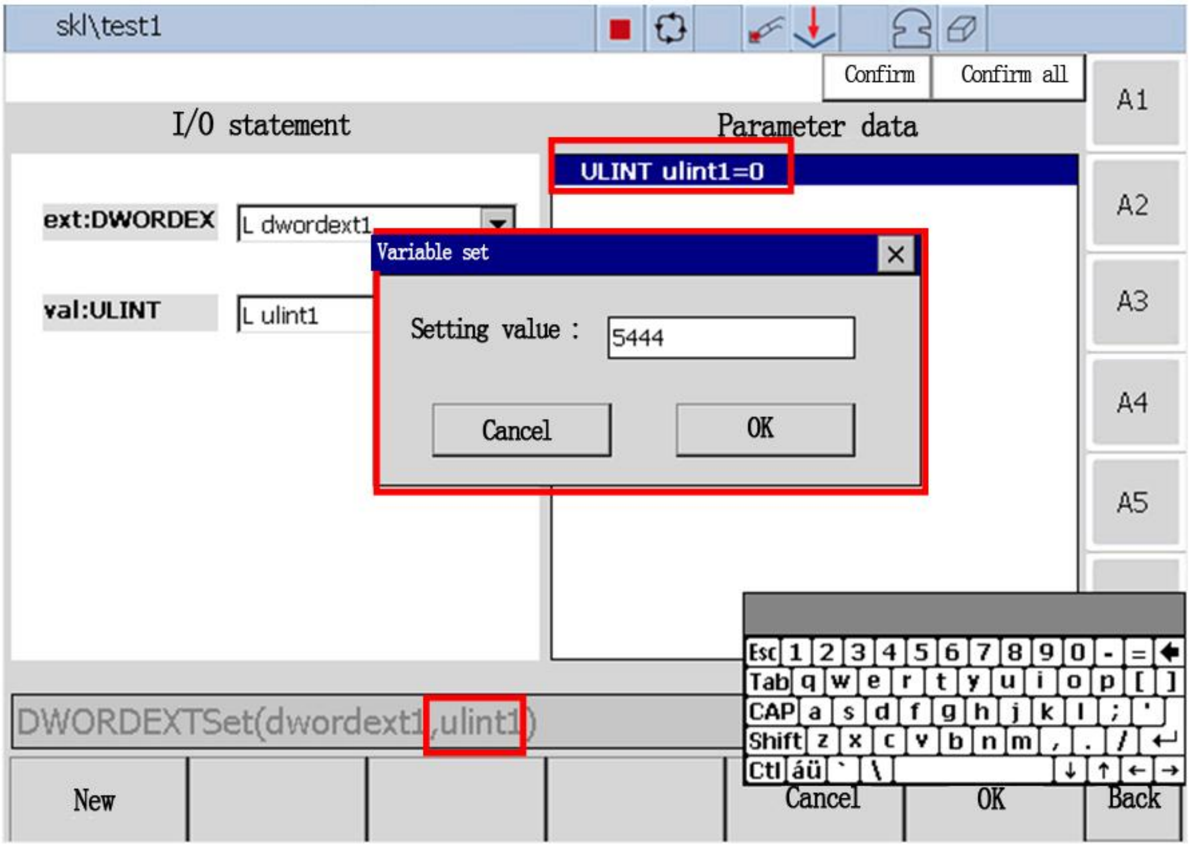
INT PORT=3
ULINT VALUE=0

Variable set

Setting value : 75

DWORDEXTSet(dwordext1,NULL)

- Create the second variable and set the parameter



skl\test1

I/0 statement

Parameter data

ext:DWORDEX L dwordext1

val:ULINT L ulint1

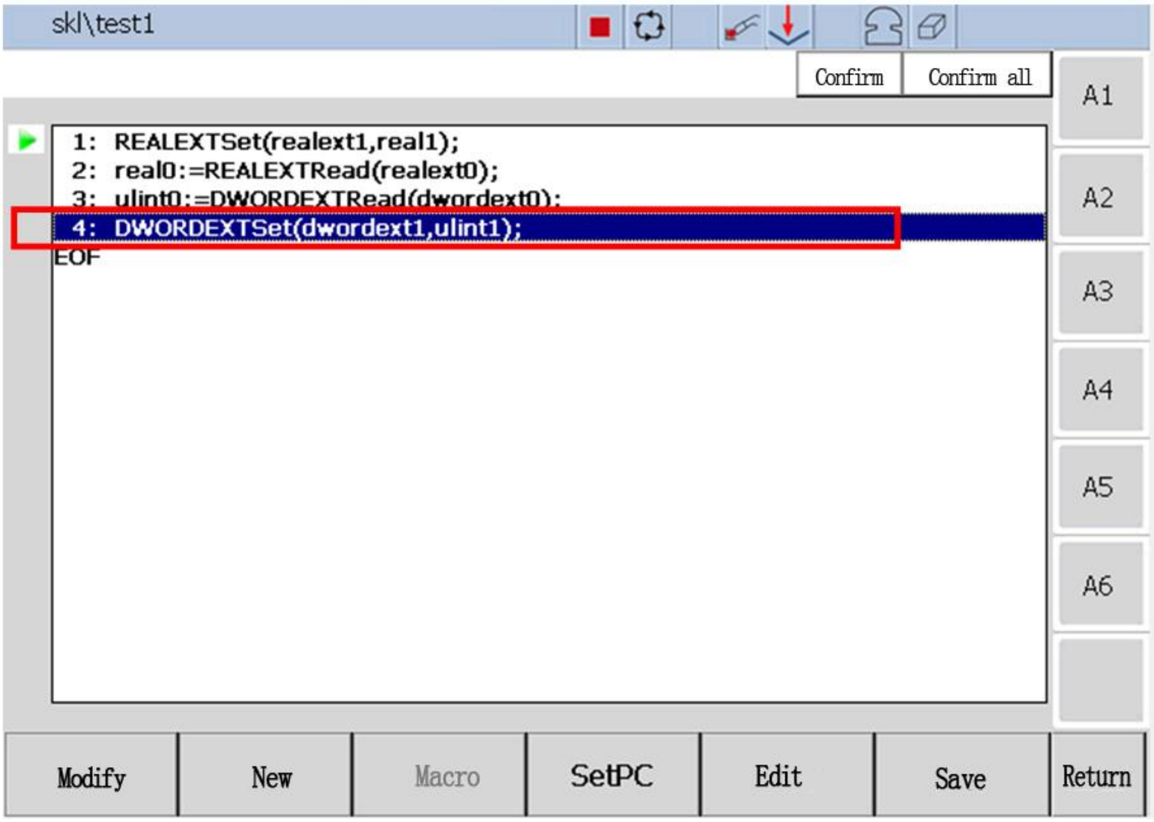
ULINT ulint1=0

Variable set

Setting value : 5444

DWORDEXTSet(dwordext1,ulint1)

- Click OK and complete the addition of the statement.



skl\test1

```
1: REALEXTSet(realext1,real1);
2: real0:=REALEXTRead(realext0);
3: ulint0:=DWORDEXTRead(dwordext0);
4: DWORDEXTSet(dwordext1,ulint1);
EOF
```

➢ **Statement format**

DWORDEXTSet(ext, val);

➢ **Parameter**

ext : BOOLEXT / /the structure, including the number and the value of the digital input signal port

val：DWORD（ULINT）

➢ **Function description**

This instruction is used to set the specified variable in PLC.

➢ **Sample**

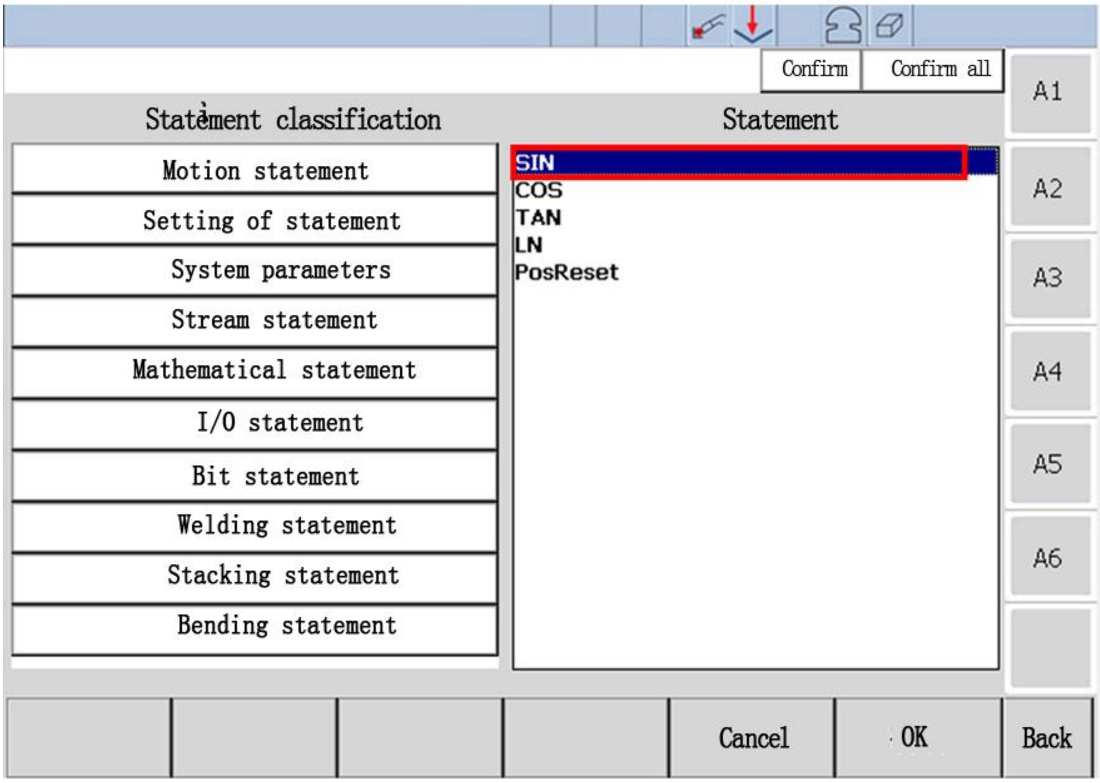
DWORDEXTSet(dwordext1, ulint1);

Assign the value of variable ulint to dwordext1.

3.6 Mathematical function instruction

3.6.1 SIN

- Create a new instruction SIN as shown in the following diagram.



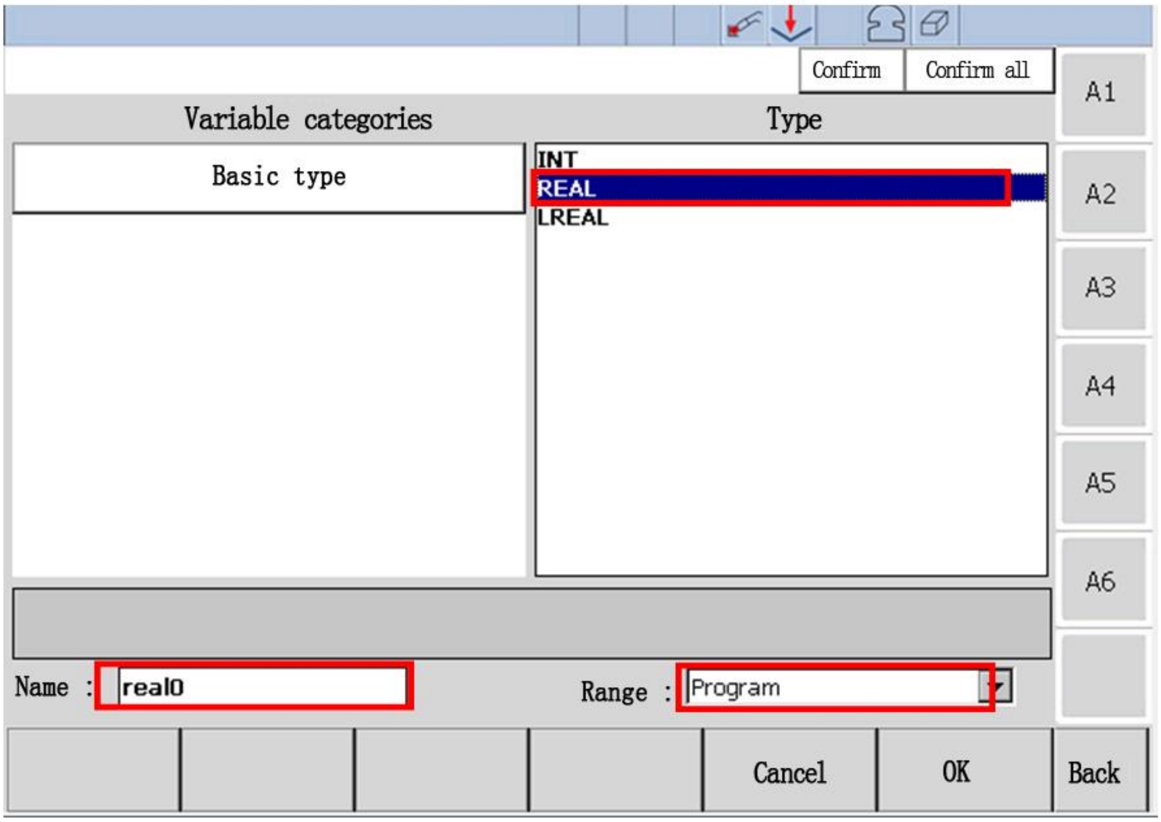
Statement classification

- Motion statement
- Setting of statement
- System parameters
- Stream statement
- Mathematical statement
- I/O statement
- Bit statement
- Welding statement
- Stacking statement
- Bending statement

Statement

- SIN
- COS
- TAN
- LN
- PosReset

- Create OK and create the first variable and set the parameter



Variable categories

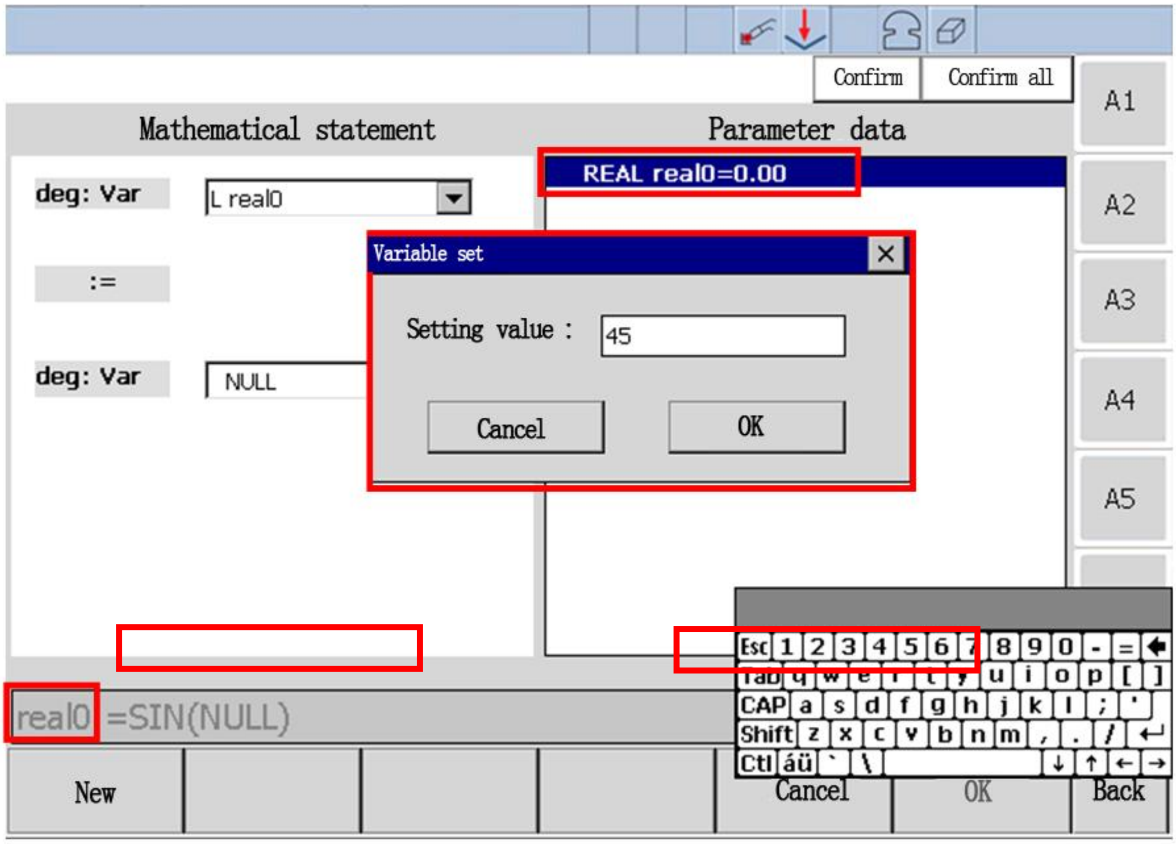
- Basic type

Type

- INT
- REAL
- LREAL

Name : real0

Range : Program



deg: Var L real0

:=

deg: Var NULL

REAL real0=0.00

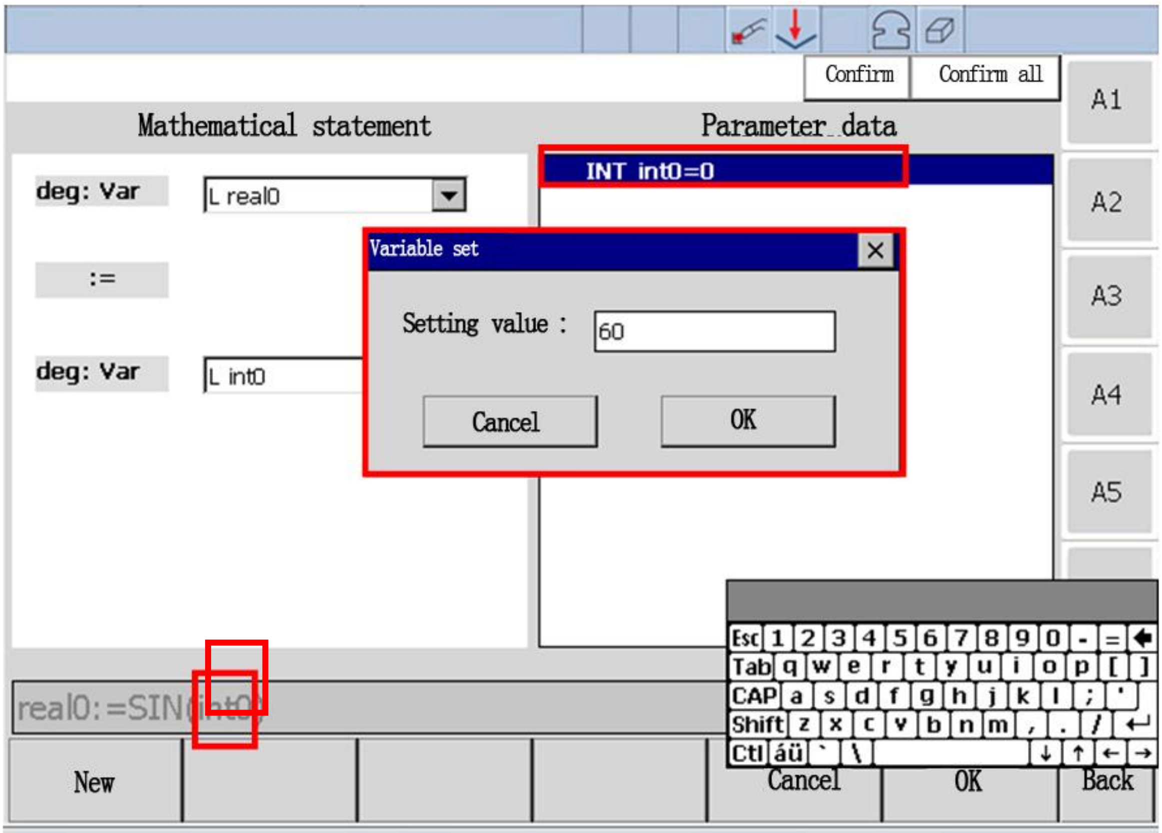
Variable set

Setting value : 45

Cancel OK

real0 =SIN(NULL)

- Create the second variable and set the parameter



deg: Var L real0

:=

deg: Var L int0

INT int0=0

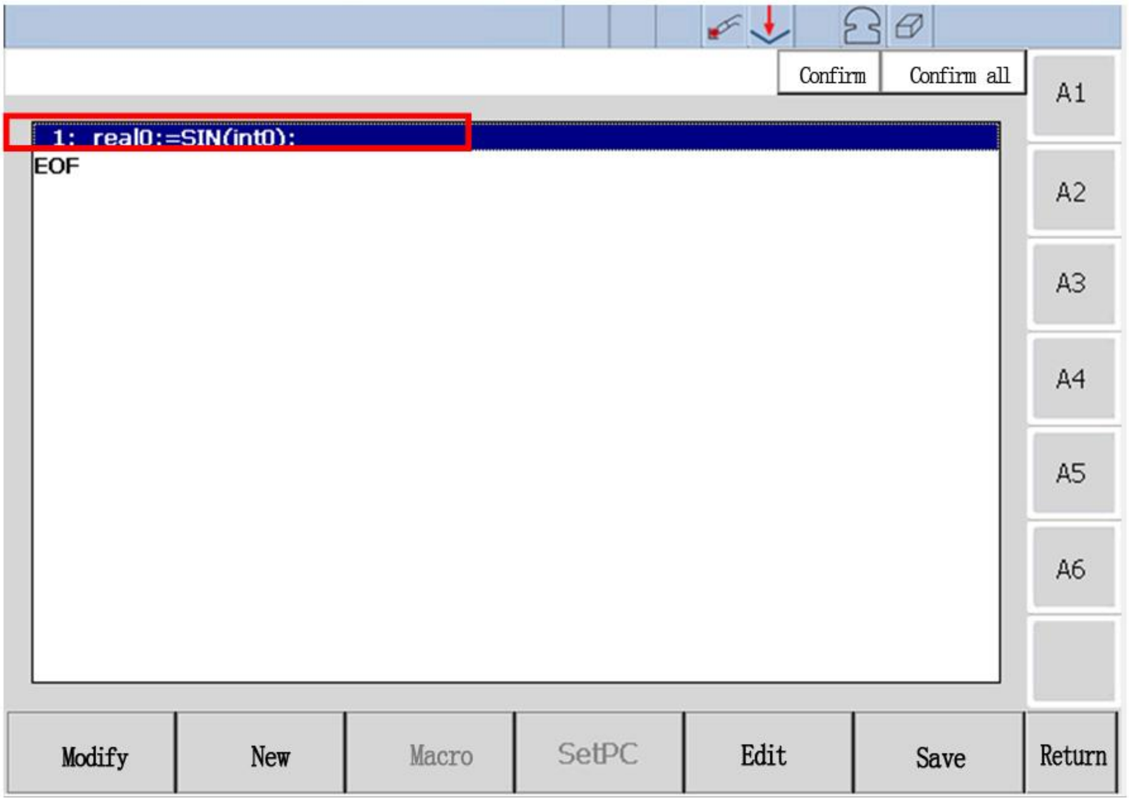
Variable set

Setting value : 60

Cancel OK

real0:=SIN(int0)

- Click OK and complete the creation of the statement



```
1: real0:=SIN(int0);
EOF
```

- **Statement format**

```
real:= SIN(deg);
```

- **Parameter**

  Parameter deg can be INT, FLOAT or DOUBLE (in angles)

- **Function description**

  Assign the sin value of a real number to the other real number.

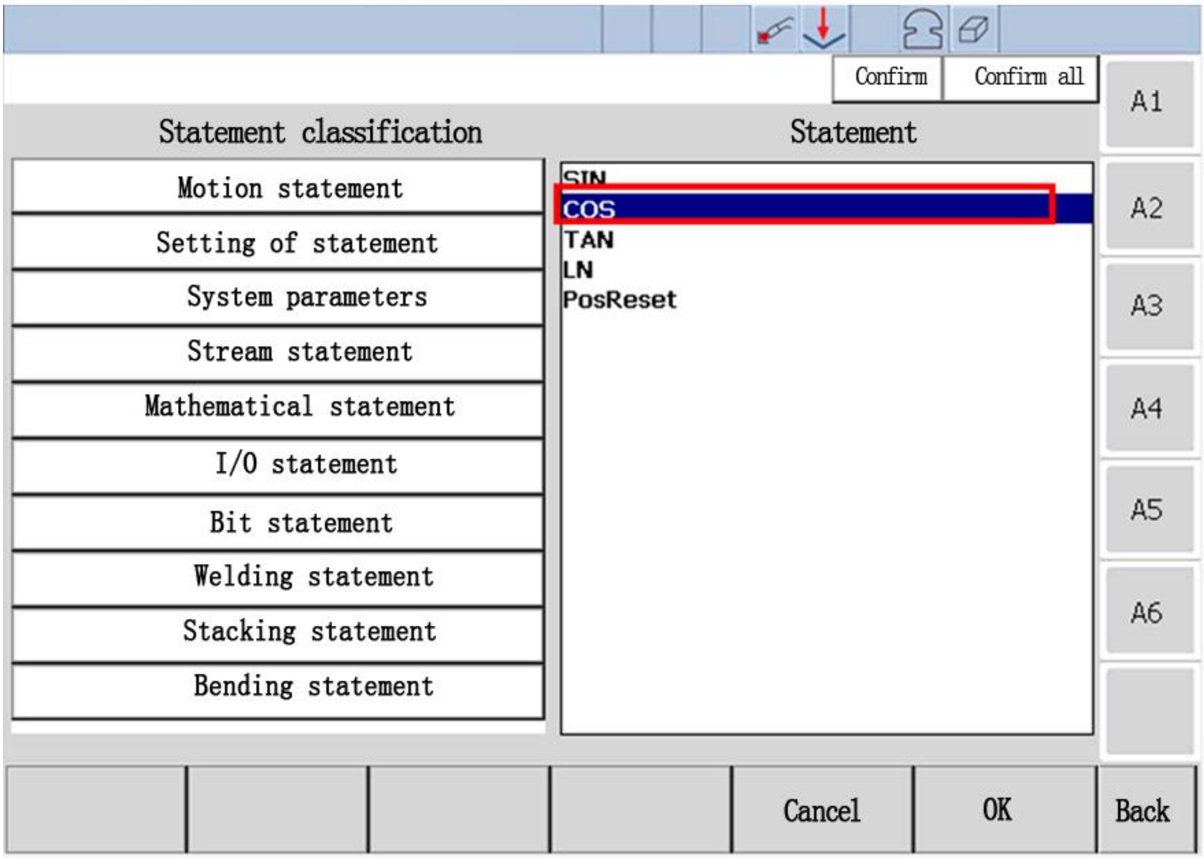
- **Sample**

  Real0:=SIN(int0);

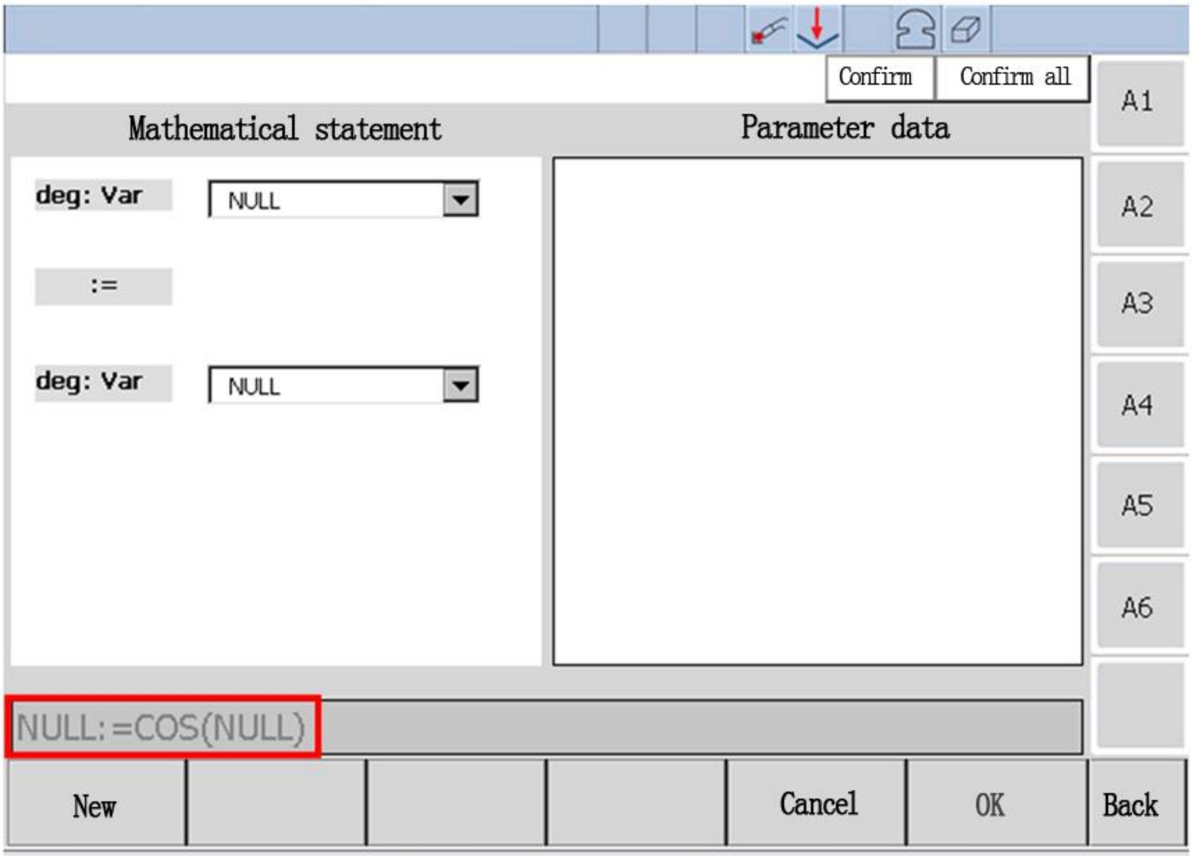
  Assign the sin value of the real number int0 to the other real number real0.

3.6.2 COS

- Create a new instruction COS as shown in the following diagram.

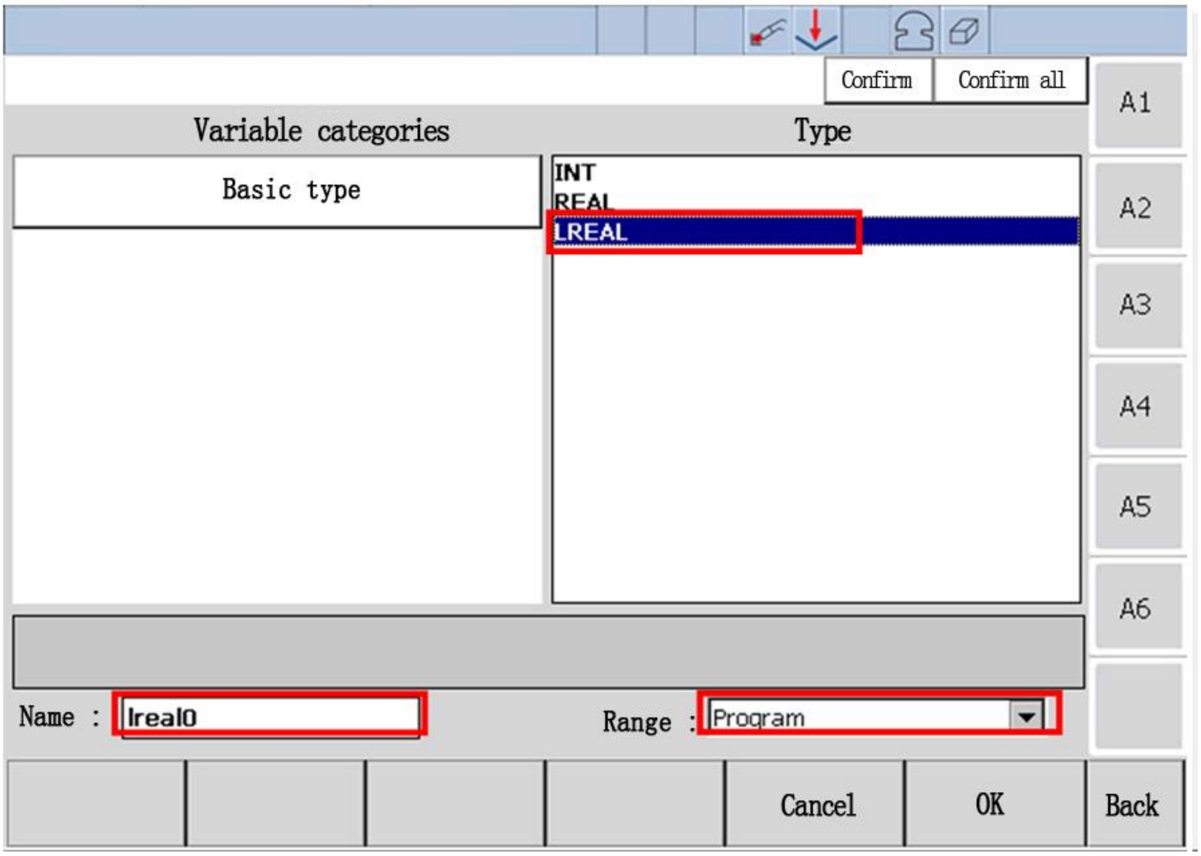


- Click OK and enter the next interface



NULL:=COS(NULL)

- Create the variable on the left side of variables and set the parameter

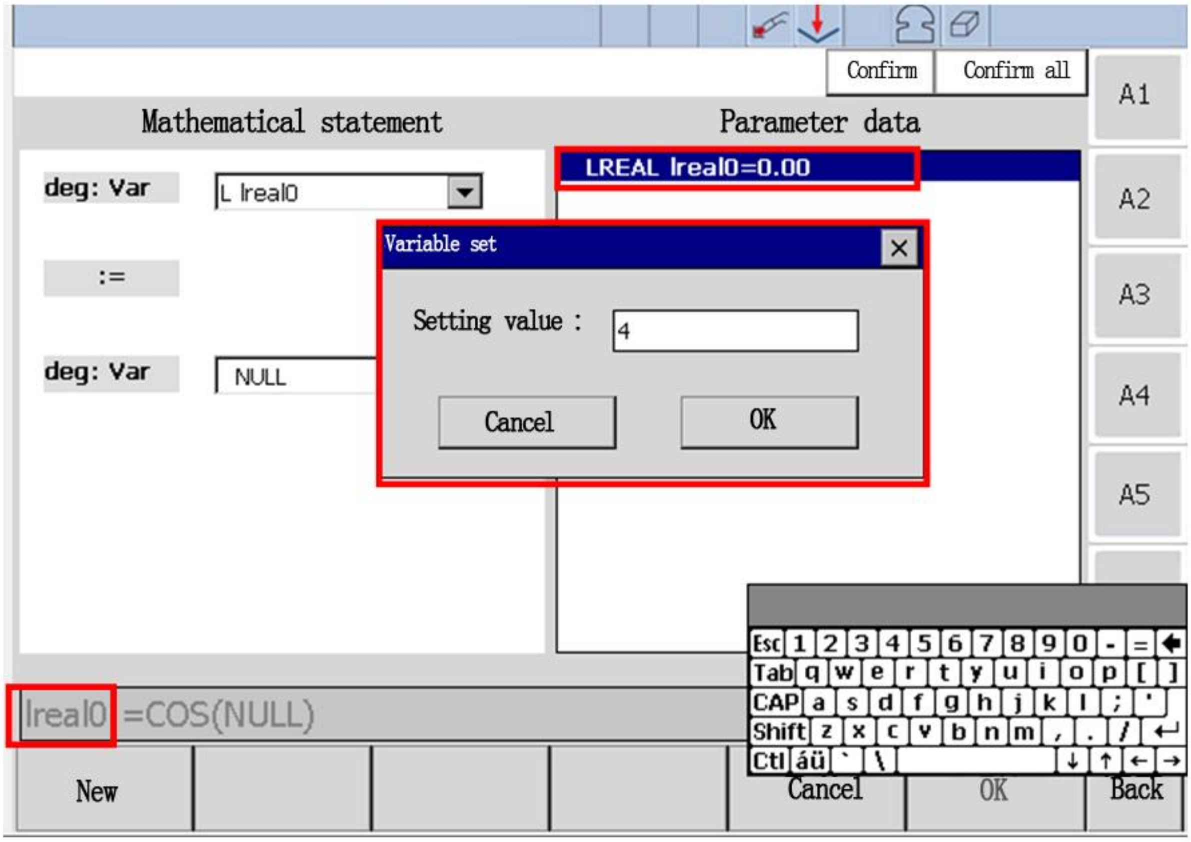


Variable categories: Basic type

Type: INT, REAL, LREAL

Name : lreal0

Range : Program



Mathematical statement

deg: Var L lreal0

:=

deg: Var NULL

Parameter data

LREAL lreal0=0.00

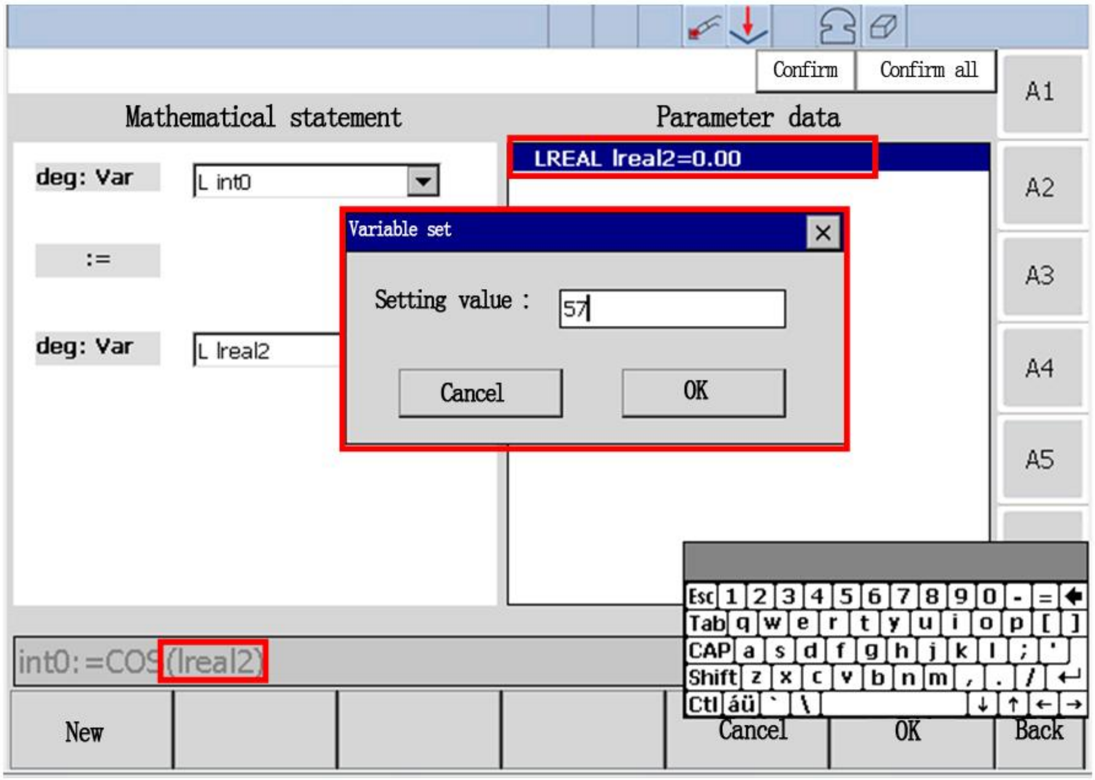
Variable set

Setting value : 4

Cancel OK

lreal0 =COS(NULL)

- Create the variable in COS on the right side and set the parameter.



deg: Var L int0

:=

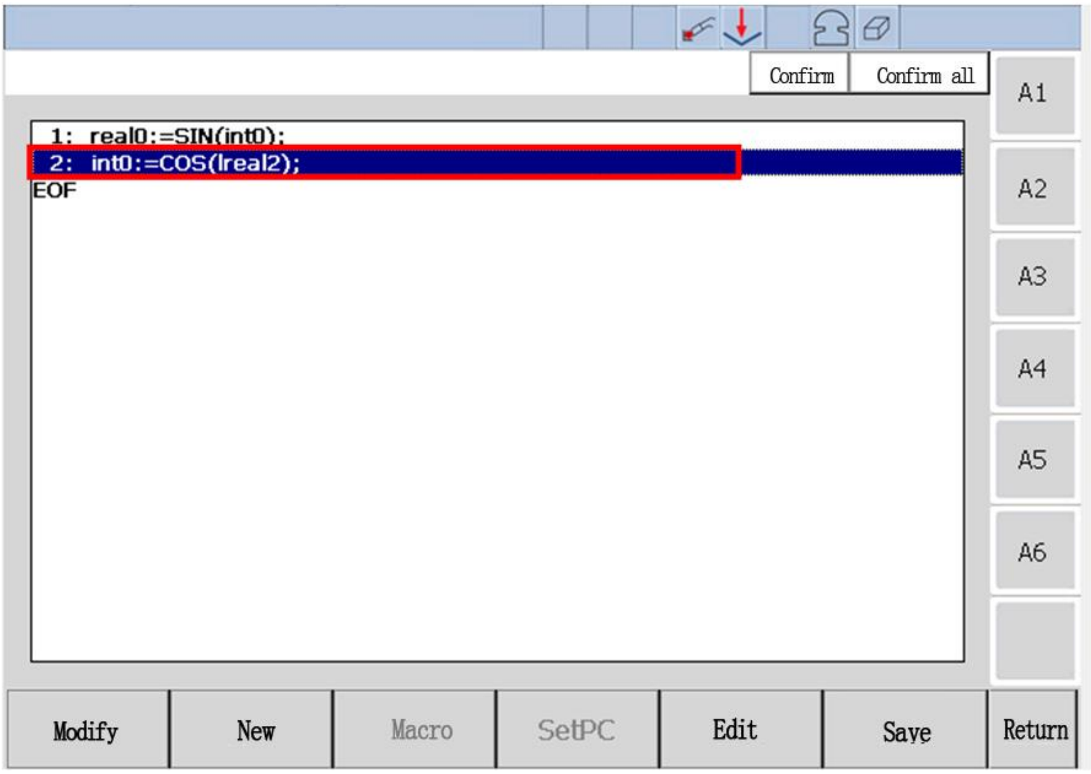
deg: Var L lreal2

LREAL lreal2=0.00

Setting value : 57

int0:=COS(lreal2)

- Click OK and complete the creation of the instruction



```
1:  real0:=SIN(int0);
2:  int0:=COS(lreal2);
EOF
```

- **Statement format**

```
real:= COS(deg);
```

- **Parameter**

Parameter deg can be INT, FLOAT or DOUBLE (in angles)

- **Function description**

Assign the cosine value of a real number to the other real number.

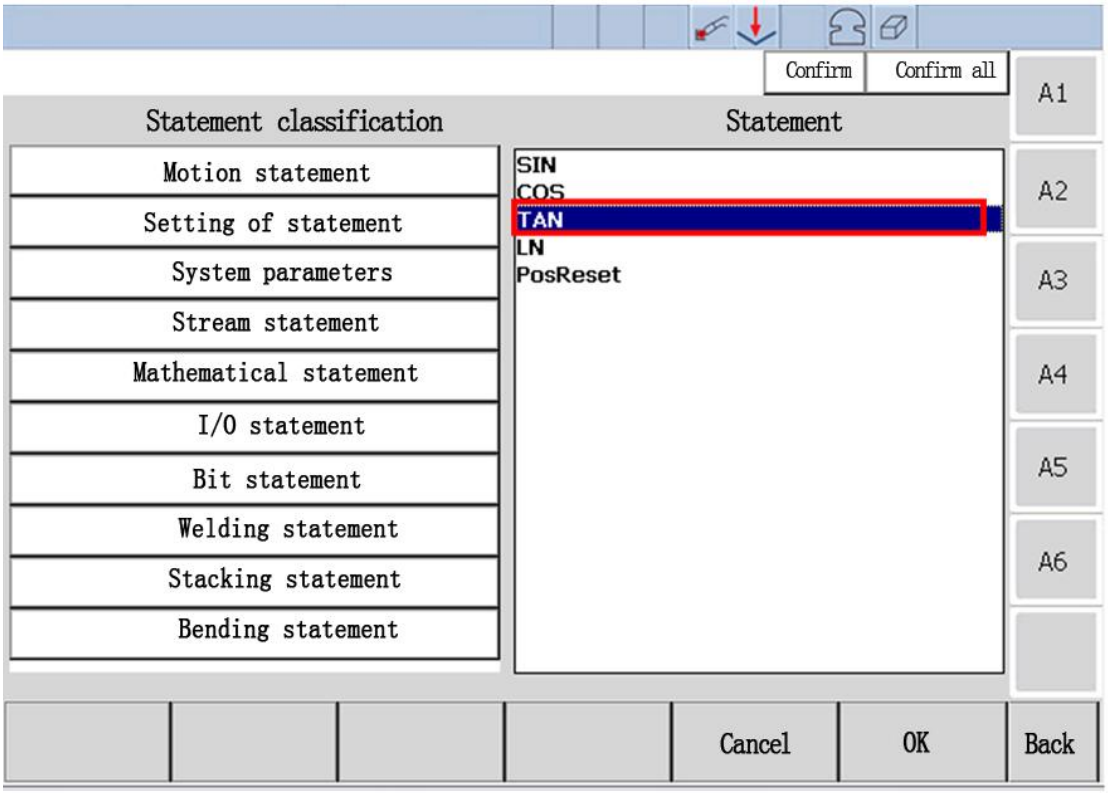
- **Sample**

int0:=COS(real2);

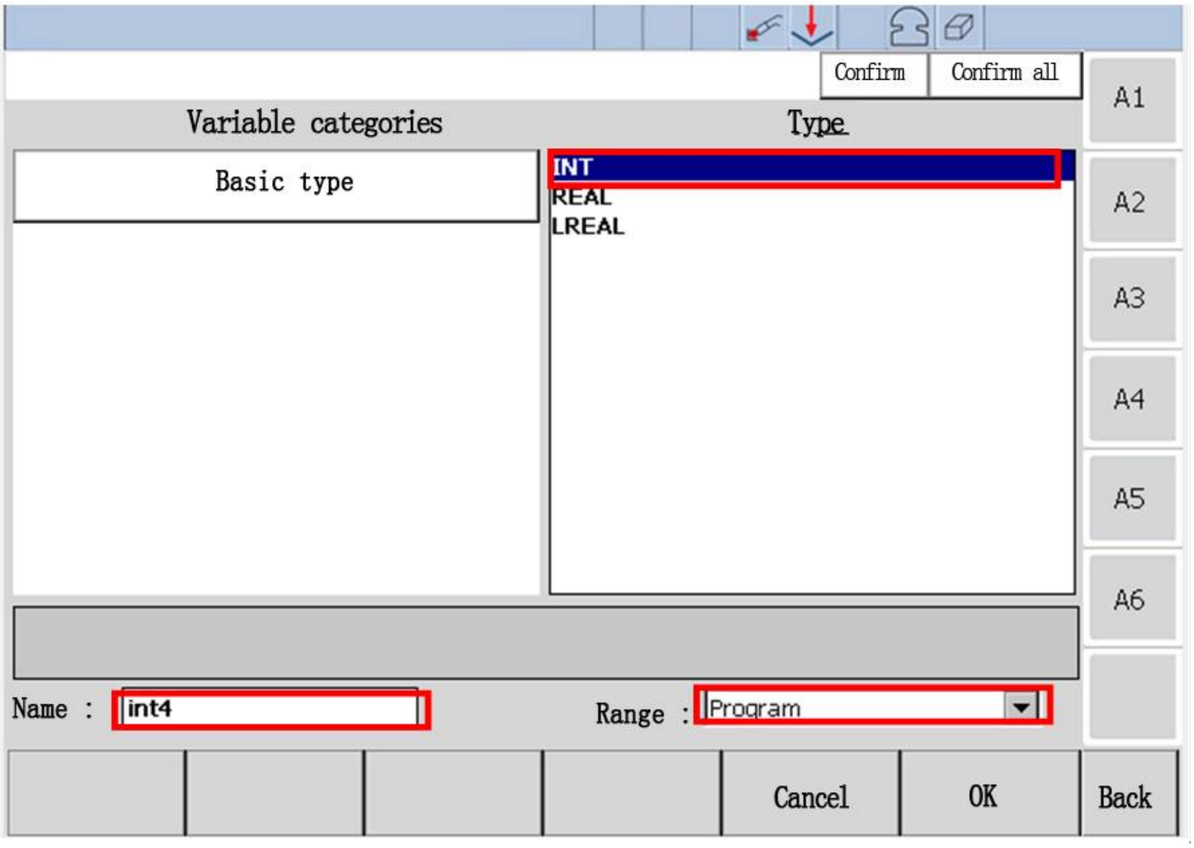
Program function: assign the cosine value of the real number real2 to the real number int0.

3.6.3 TAN

- Create the statement TAN as shown in the following diagram.



- Click OK and then create and rename the right side of variable

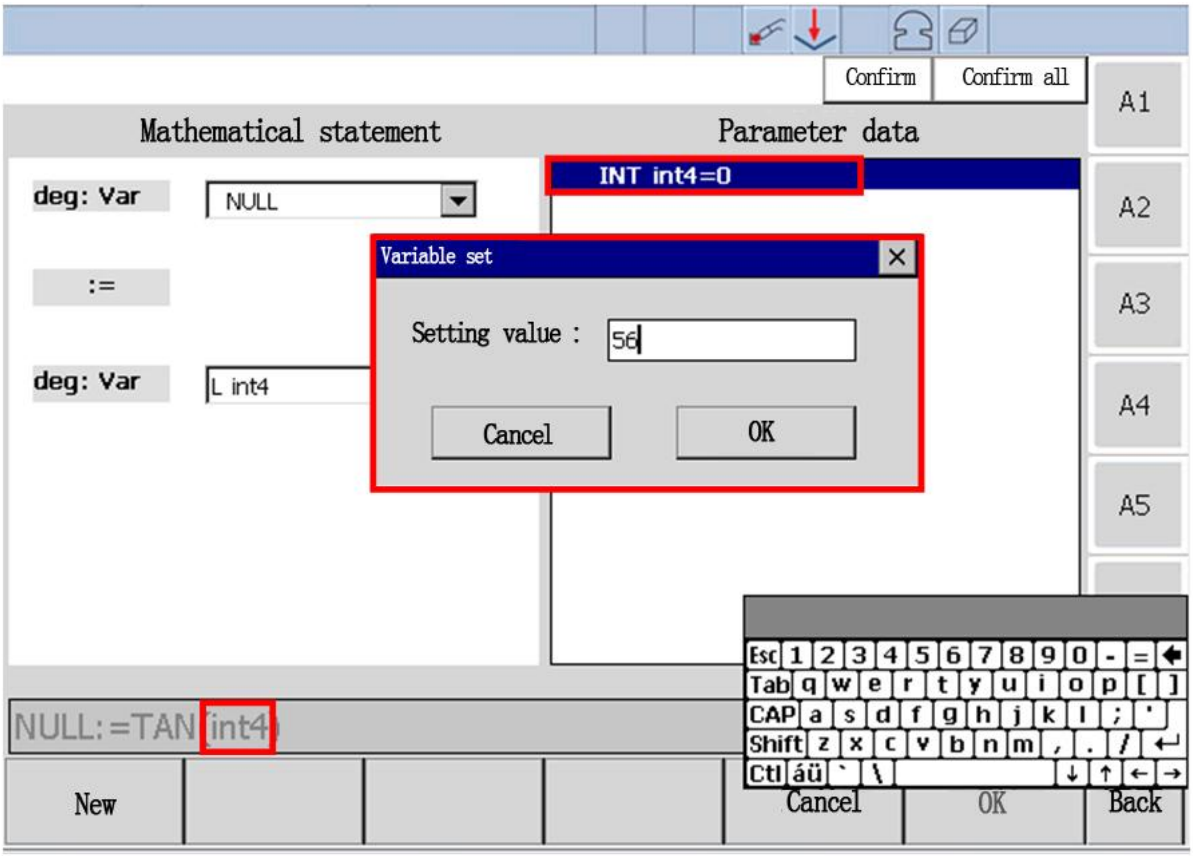


Variable categories: Basic type

Type: INT
REAL
LREAL

Name : int4

Range : Program


Confirm
Confirm all
A1
A2
A3
A4
A5
Esc 1 2 3 4 5 6 7 8 9 0 - =
Tab q w e r t y u i o p [ ]
CAP a s d f g h j k l ; '
Shift z x c v b n m , . /
Ctl áü ` \ ↓ ↑ ← →
New
Cancel
OK
Back

Mathematical statement

deg: Var NULL

:=

deg: Var L int4

Parameter data

INT int4=0

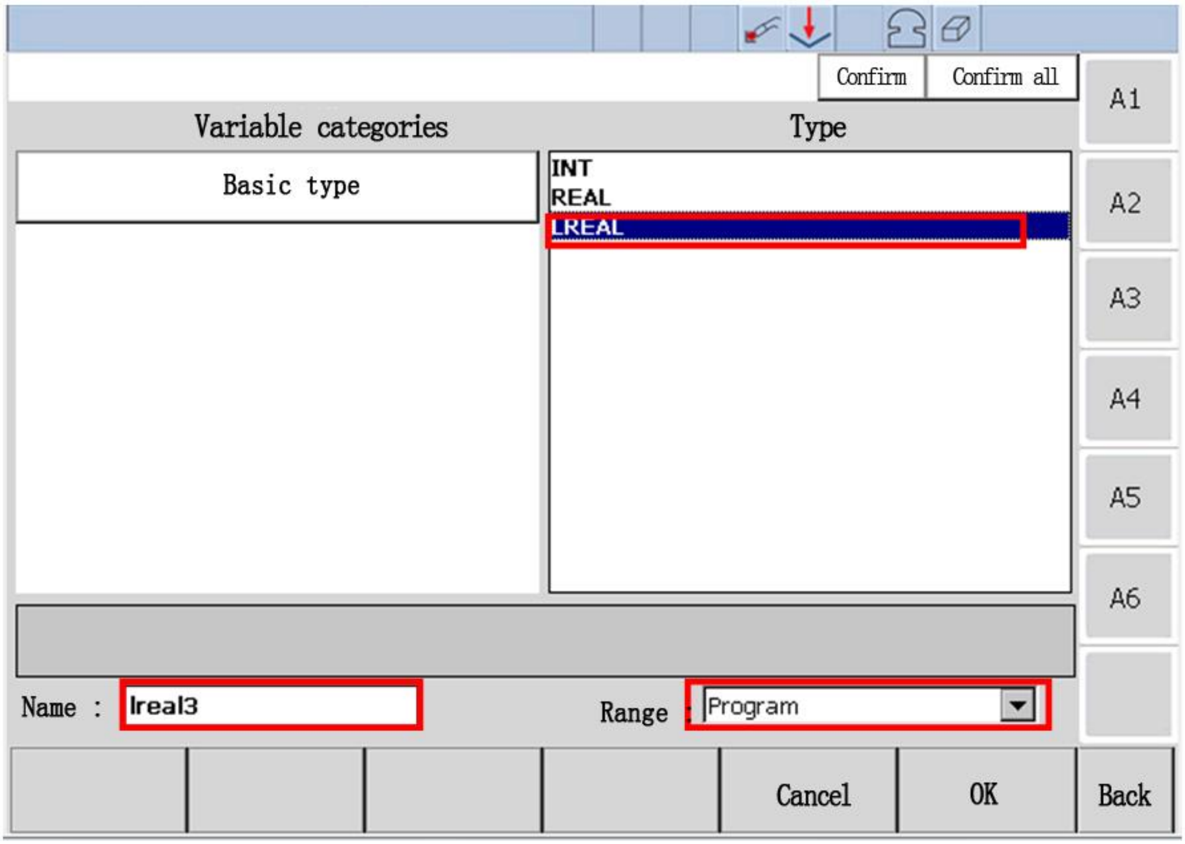
Variable set

Setting value : 56

Cancel OK

NULL:=TAN(int4)

- Create the left side of variable and set the parameter

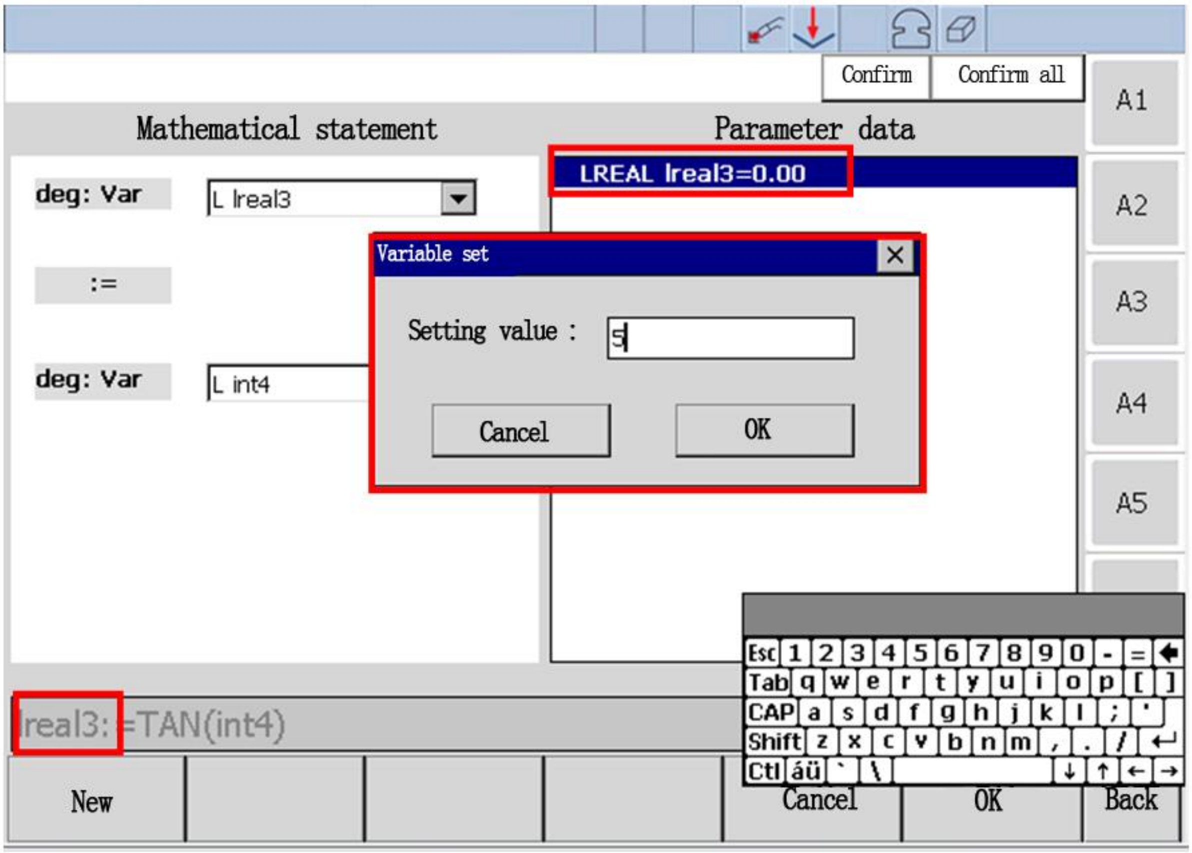


Variable categories: Basic type

Type: INT, REAL, LREAL

Name : lreal3

Range : Program


Confirm
Confirm all
A1
A2
A3
A4
A5
Variable set
Cancel
OK
New
Cancel
OK
Back
Esc 1 2 3 4 5 6 7 8 9 0 - =
Tab q w e r t y u i o p [ ]
CAP a s d f g h j k l ; '
Shift z x c v b n m , . /
Ctl áü ` \ ↓ ↑ ← →

Mathematical statement

deg: Var L lreal3

:=

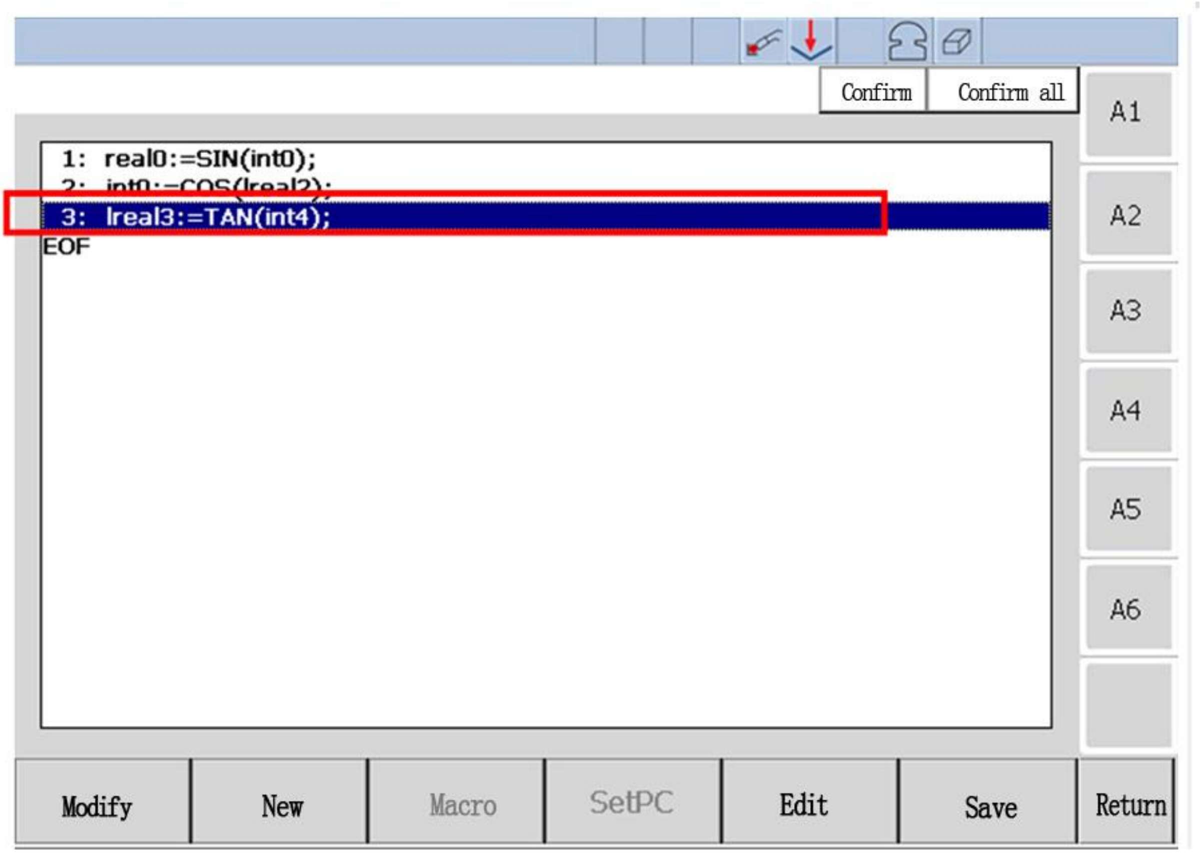
deg: Var L int4

Parameter data: LREAL lreal3=0.00

Setting value : 5

lreal3:=TAN(int4)

- Click OK and complete the creation of the statement as shown in the following diagram.



```
1:  real0:=SIN(int0);
2:  int0:=COS(lreal2);
3:  lreal3:=TAN(int4);
EOF
```

- **Statement format**

```
real:= TAN(deg);
```

- **Parameter:**

Parameter deg can be INT, FLOAT or DOUBLE (in angles)

- **Function description**

Assign the tan value of a real number to the other real number.

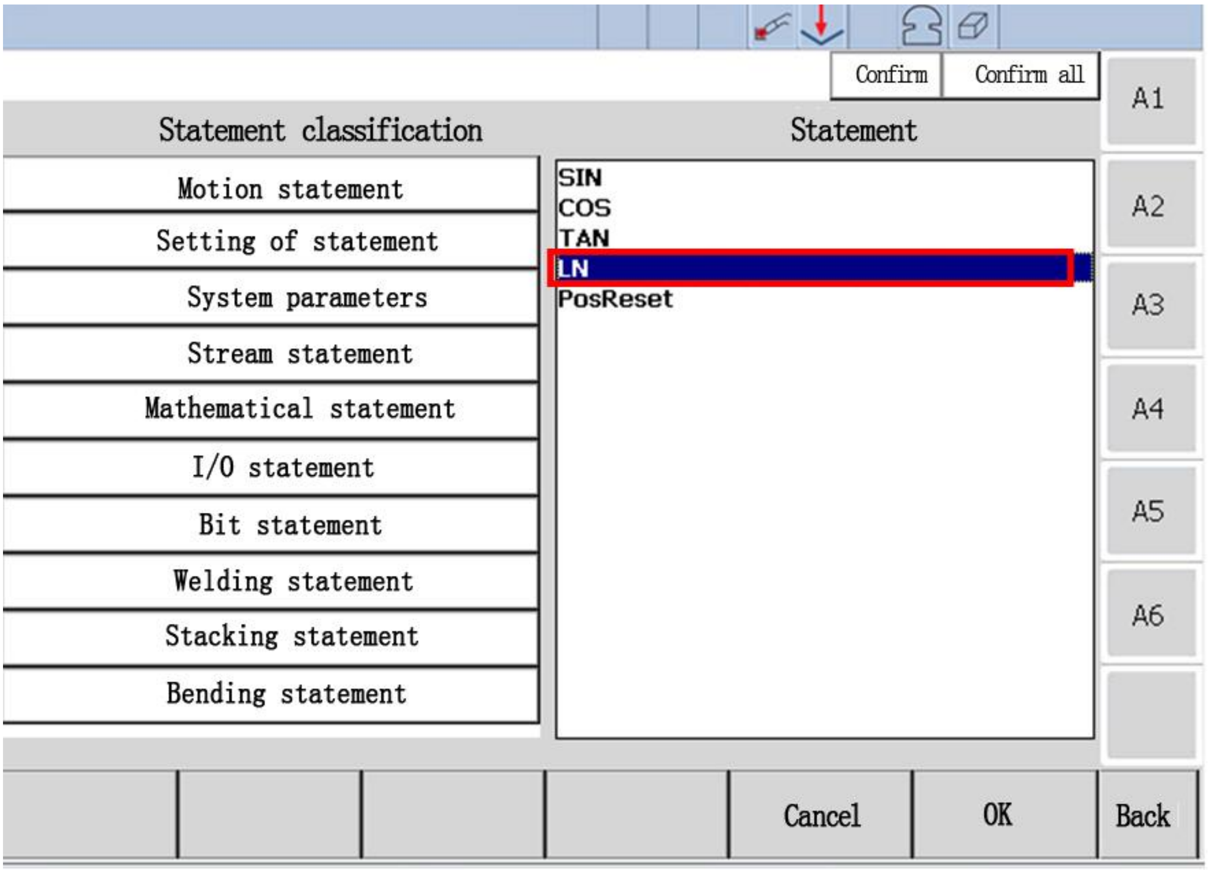
- **Sample**

lreal3:=TAN(int4);

Program function: assign the tan value of the real number int4 to the real number lreal3.

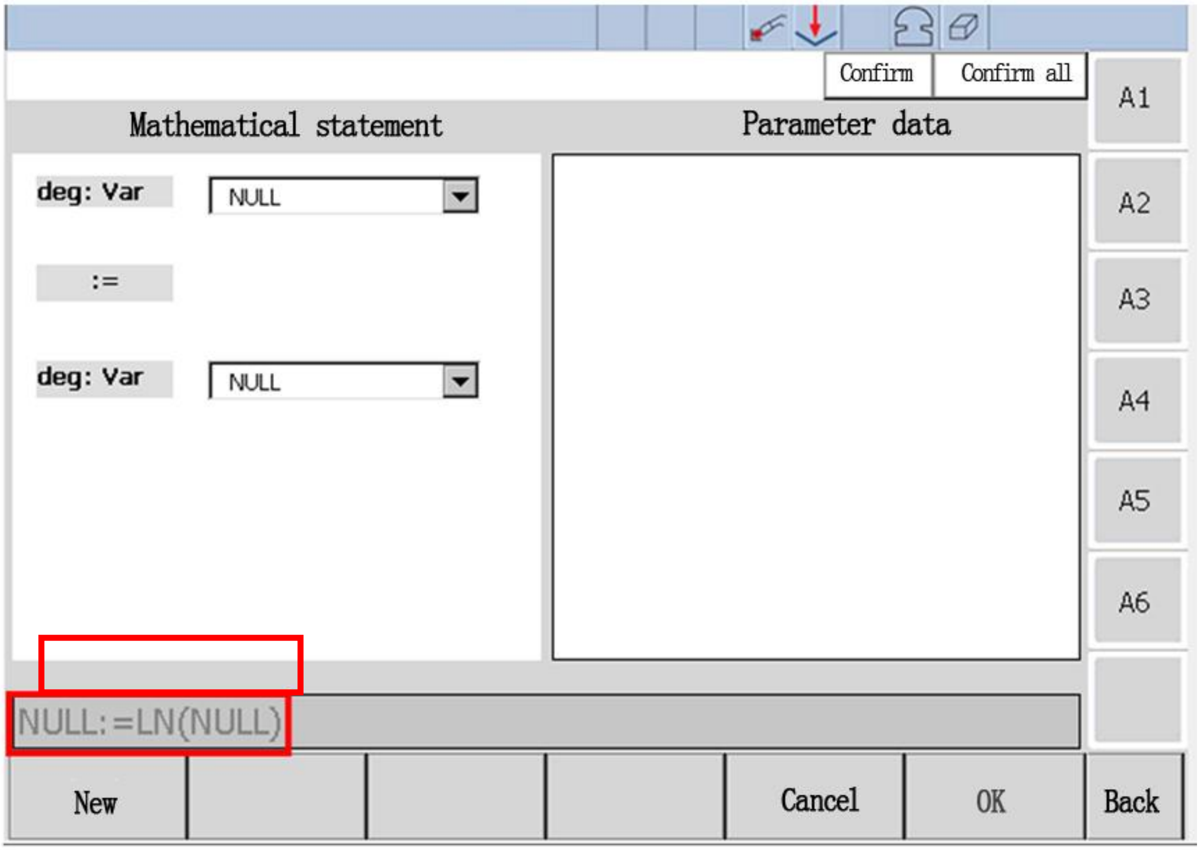
3.6.4 LN

- Create the statement LN as shown in the following diagram.



Statement classification	Statement
Motion statement	SIN
Setting of statement	COS
System parameters	TAN
Stream statement	LN
Mathematical statement	PosReset
I/O statement	
Bit statement	
Welding statement	
Stacking statement	
Bending statement	

- Click OK and enter the next interface



Mathematical statement

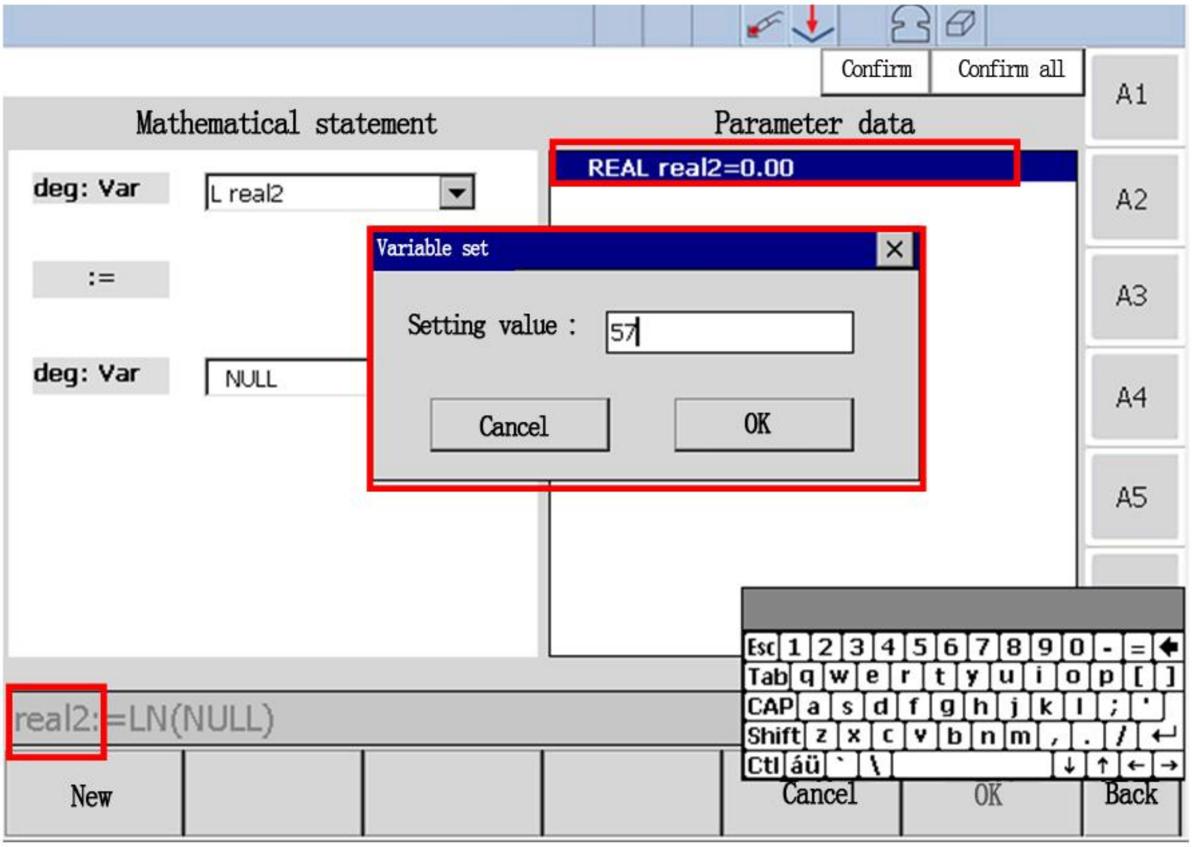
deg: Var NULL

:=

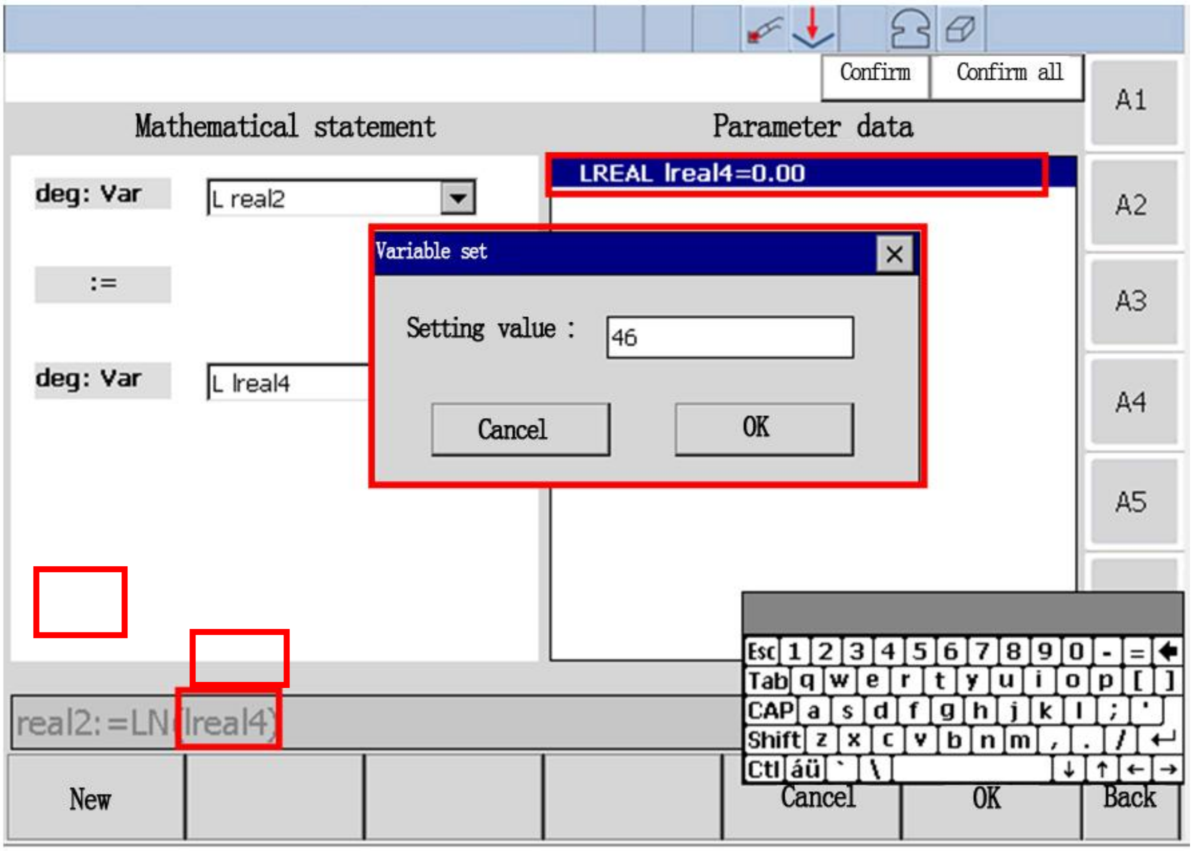
deg: Var NULL

NULL:=LN(NULL)

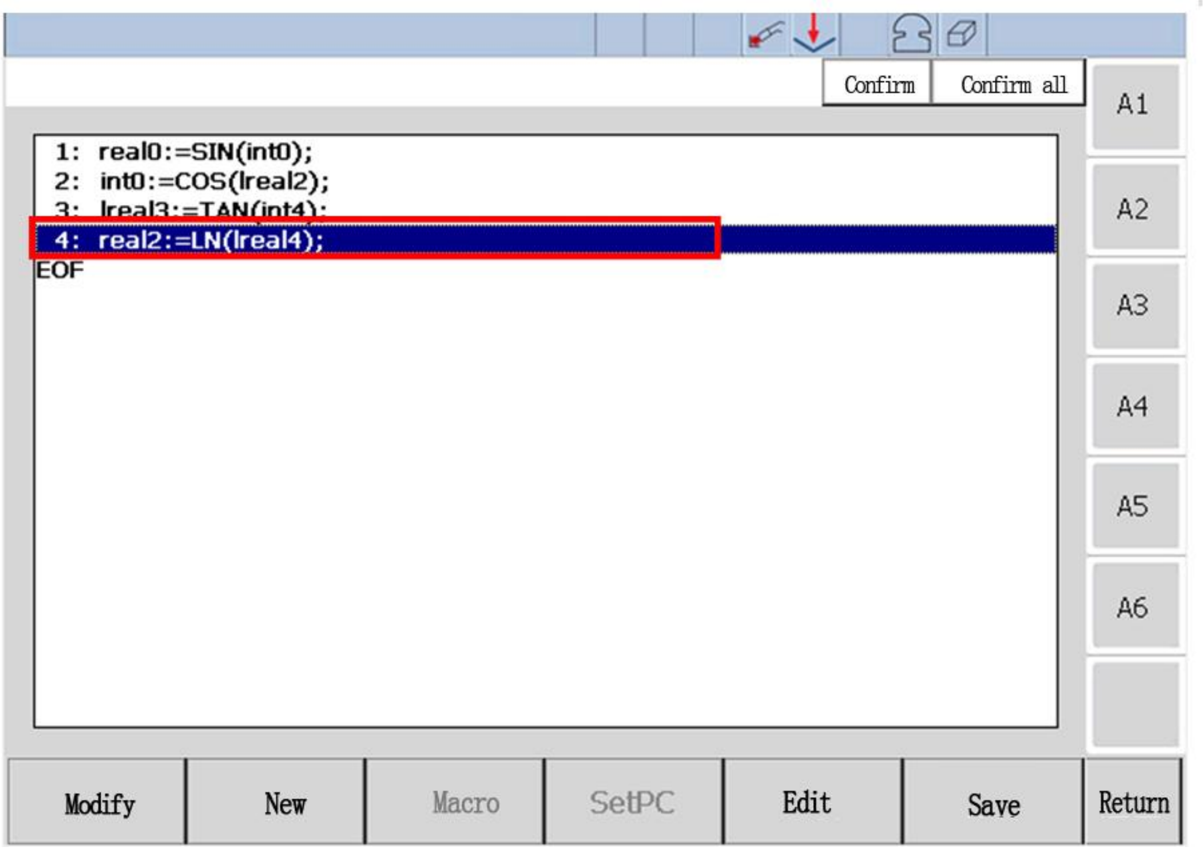
- Create the left side of variable and set the parameter.



- Create the right side of variable and set the parameter.



- Click OK and complete the creation of the statement



```
real0:=SIN(int0);
int0:=COS(lreal2);
lreal3:=TAN(int4);
real2:=LN(lreal4);
EOF
```

- **Statement format**

```
real:= LN(deg);
```

- **Parameter**

Parameter deg can be INT, FLOAT or DOUBLE (in angles).

- **Function description**

Assign the log value of a real number to the other real number.

- **Sample**

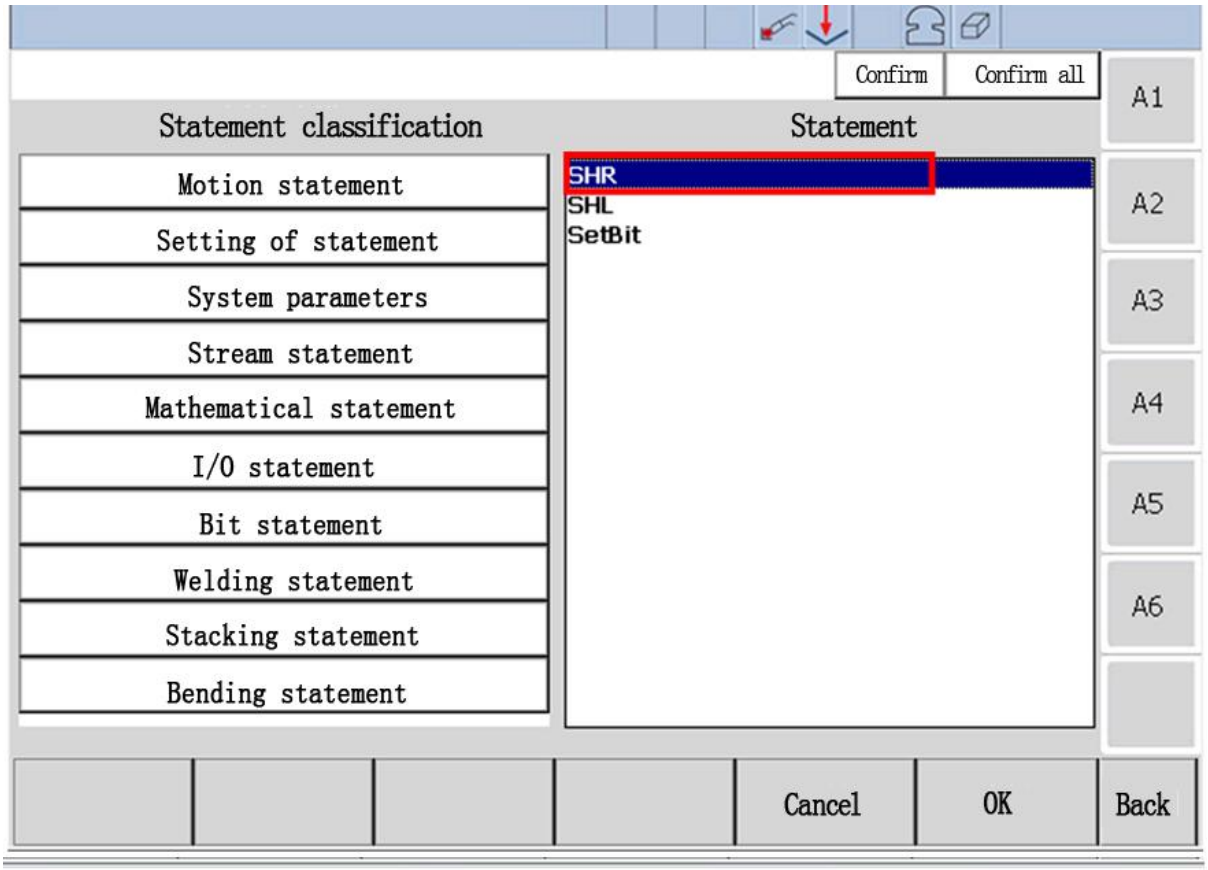
real2:=LN(lreal4);

Program demonstrates: assign the log value of the real number lreal4 to the real number real2.

3.7 Bit operation instruments

3.7.1 SHL

- Create the statement SHL as shown in the following diagram.



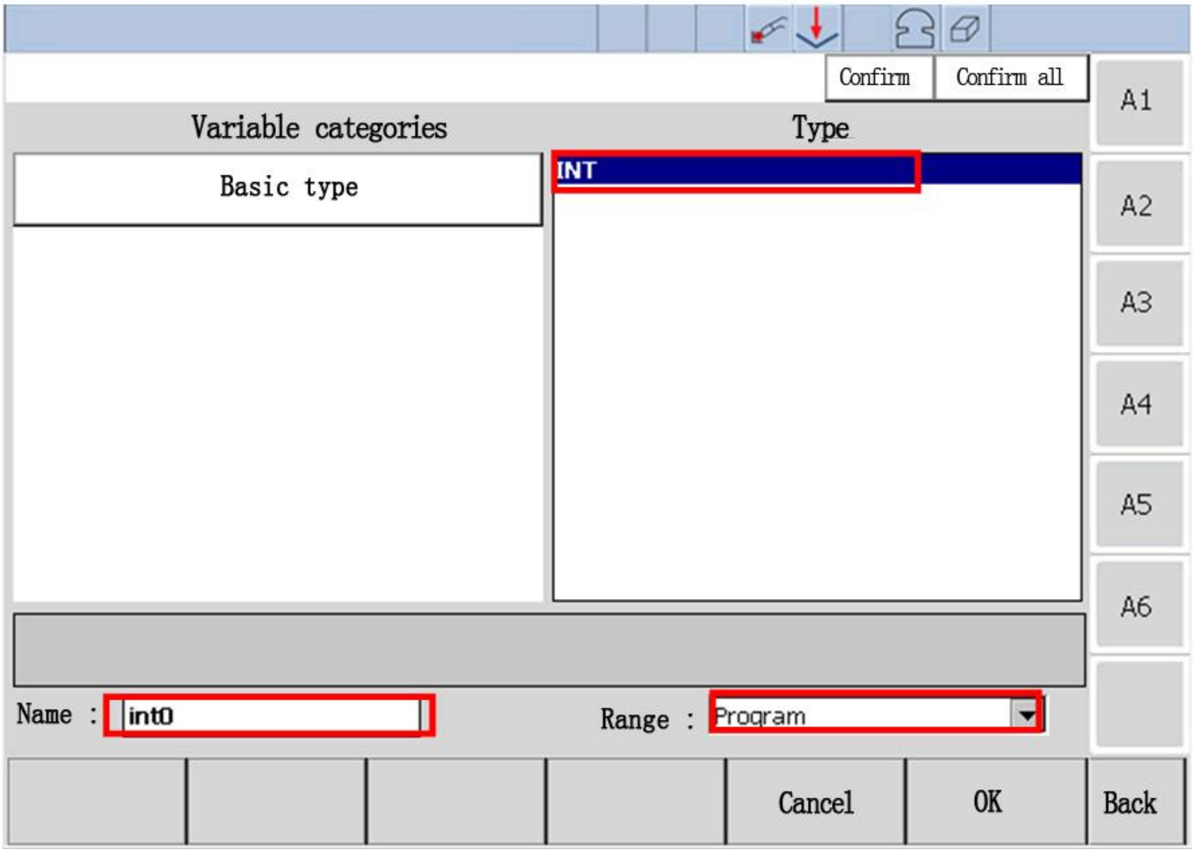
Statement classification

- Motion statement
- Setting of statement
- System parameters
- Stream statement
- Mathematical statement
- I/O statement
- Bit statement
- Welding statement
- Stacking statement
- Bending statement

Statement

- SHR
- SHL
- SetBit

- Click OK and enter the variable creation and setting interface. Next create the left side of variable.

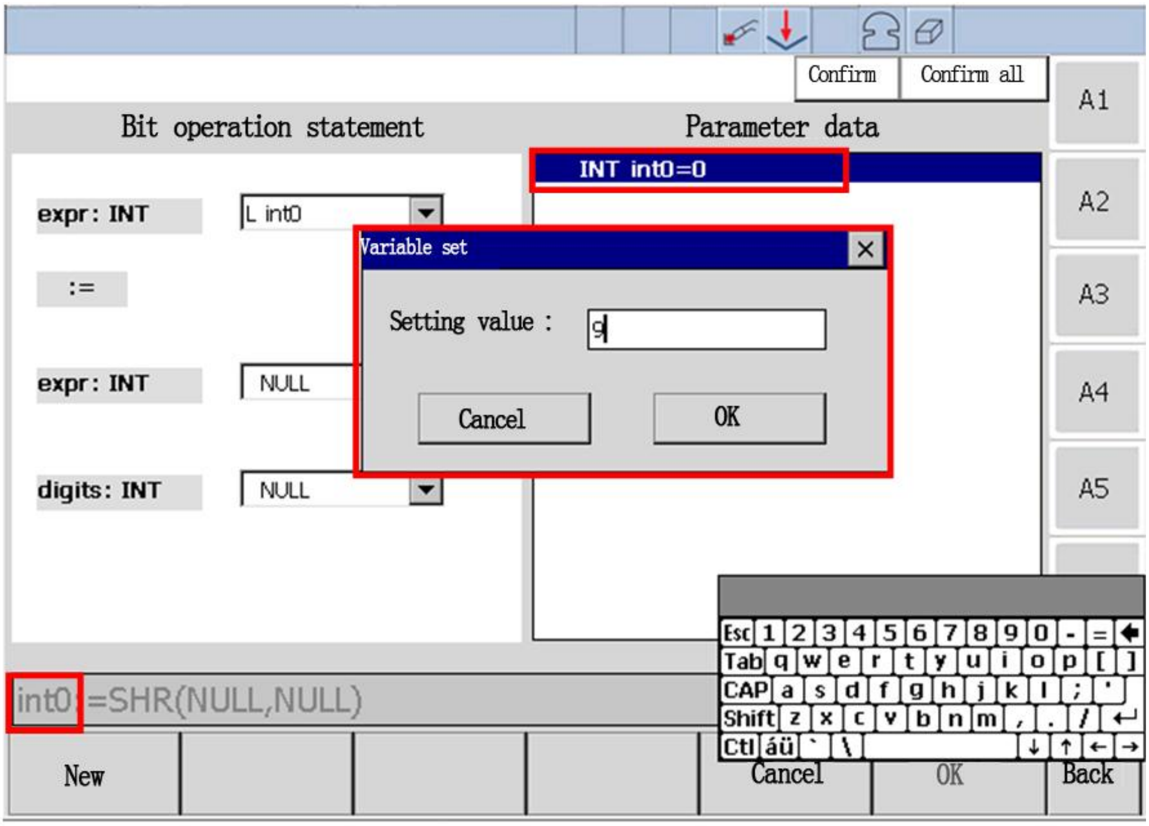


Variable categories: Basic type

Type: INT

Name : int0

Range : Program


Confirm
Confirm all
A1
A2
A3
A4
A5
Esc 1 2 3 4 5 6 7 8 9 0 - =
Tab q w e r t y u i o p [ ]
CAP a s d f g h j k l ; '
Shift z x c v b n m , . /
Ctl áü ` \
New
Cancel
OK
Back

Bit operation statement

Parameter data

INT int0=0

expr: INT L int0

:=

expr: INT NULL

digits: INT NULL

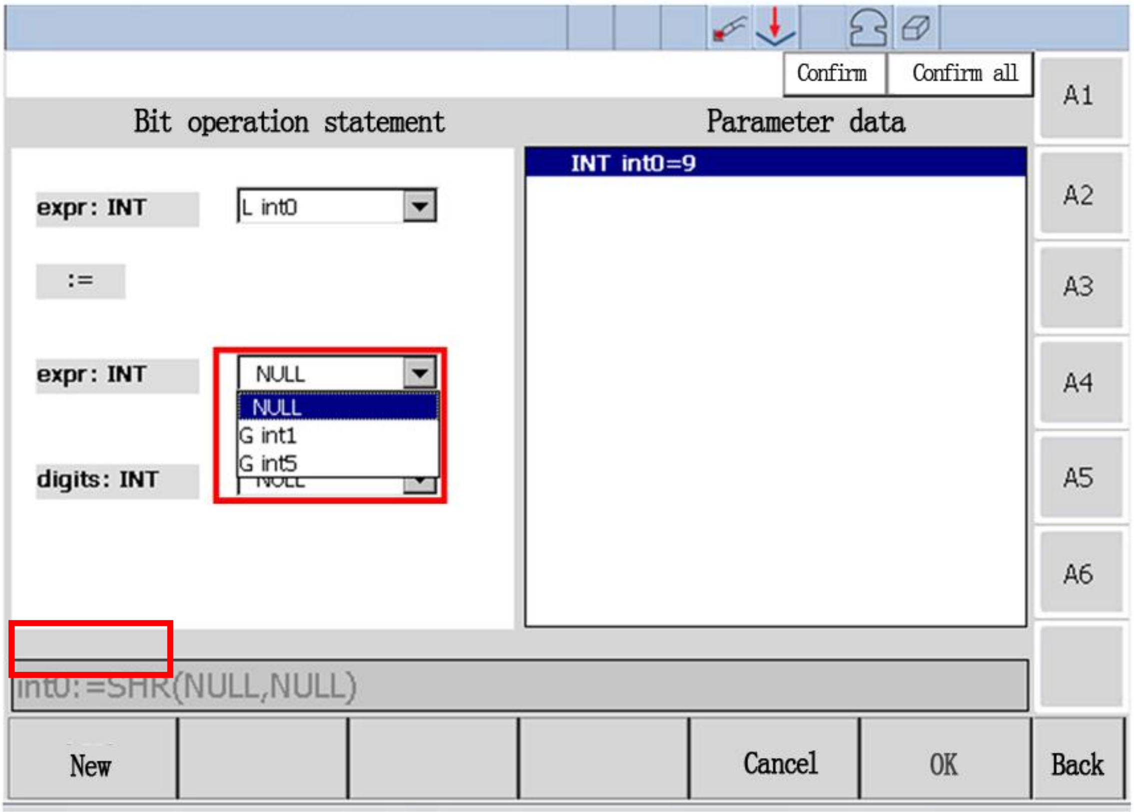
Variable set

Setting value : 9

Cancel OK

int0:=SHR(NULL,NULL)

- Next create the right side of variable and set the parameter.



Bit operation statement

Parameter data

INT int0=9

expr: INT L int0

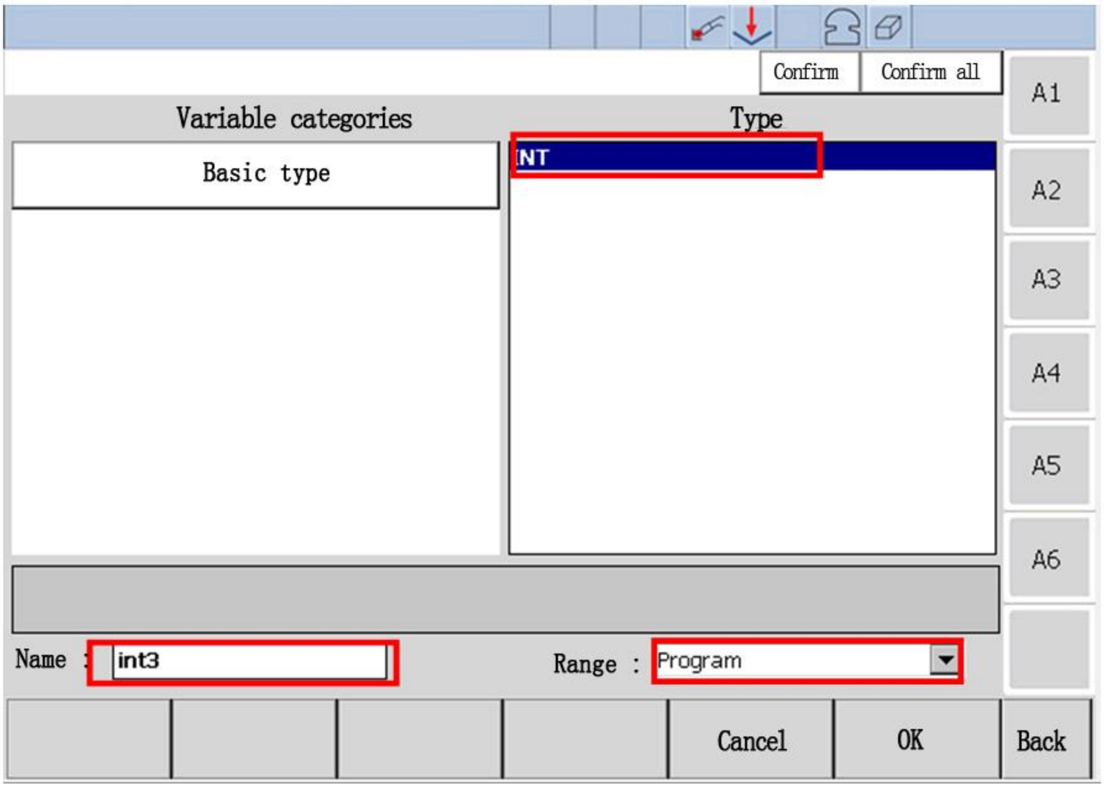
:=

expr: INT NULL

NULL
G int1
G int5

digits: INT NULL

int0:=SHR(NULL,NULL)

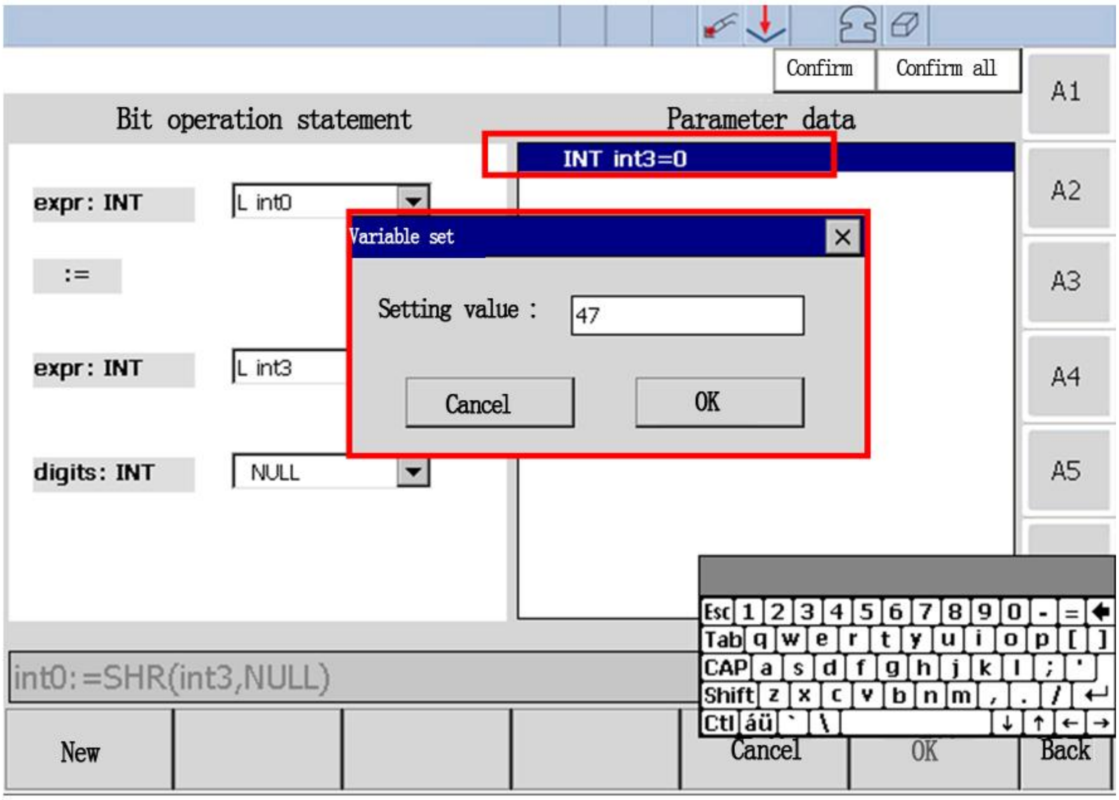


Variable categories: Basic type

Type: INT

Name : int3

Range : Program



Bit operation statement

expr: INT L int0

:=

expr: INT L int3

digits: INT NULL

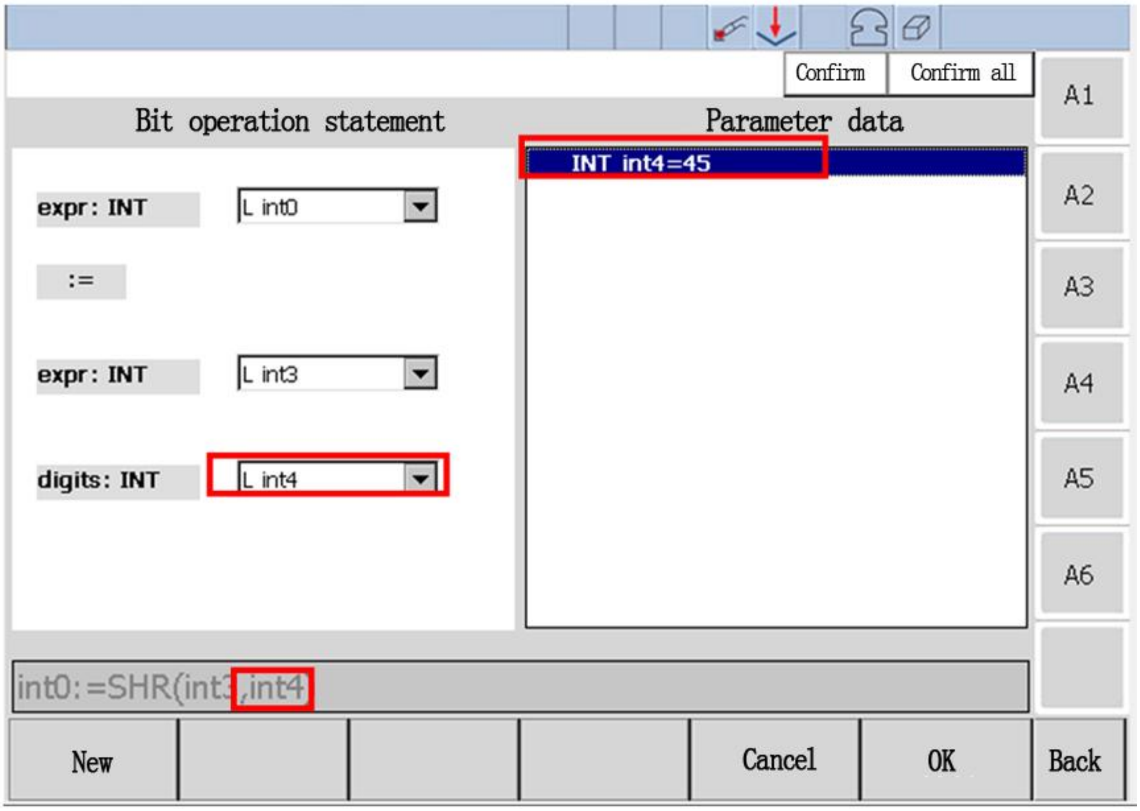
Parameter data: INT int3=0

Variable set

Setting value : 47

int0:=SHR(int3,NULL)

Confirm
Confirm all
A1
A2
A3
A4
A5
Cancel
OK
New
Cancel
OK
Back
Esc 1 2 3 4 5 6 7 8 9 0 - =
Tab q w e r t y u i o p [ ]
CAP a s d f g h j k l ; '
Shift z x c v b n m , . /
Ctl áü ` \ ↓ ↑ ← →



Bit operation statement

expr: INT L int0

:=

expr: INT L int3

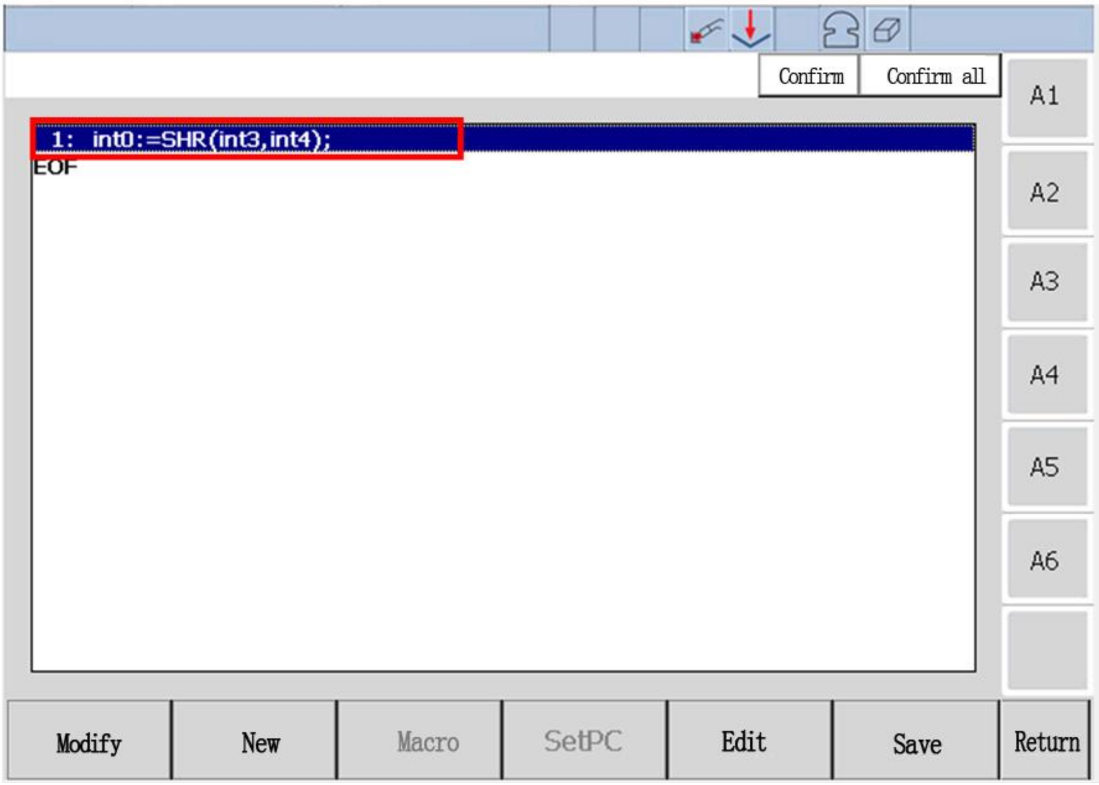
digits: INT L int4

Parameter data

INT int4=45

int0:=SHR(int3,int4)

- Click OK and complete the creation of the statement as shown in the following diagram.



```
1:  int0:=SHR(int3,int4);
EOF
```

- **Statement format**

```
int:=SHL（Expr，digit）;
```

- **Parameter**

  Expr the value that will be moved

  digits the bits of the value that will be moved

- **Function description**

  Move Expr digit bits to the left and then assign it to int.

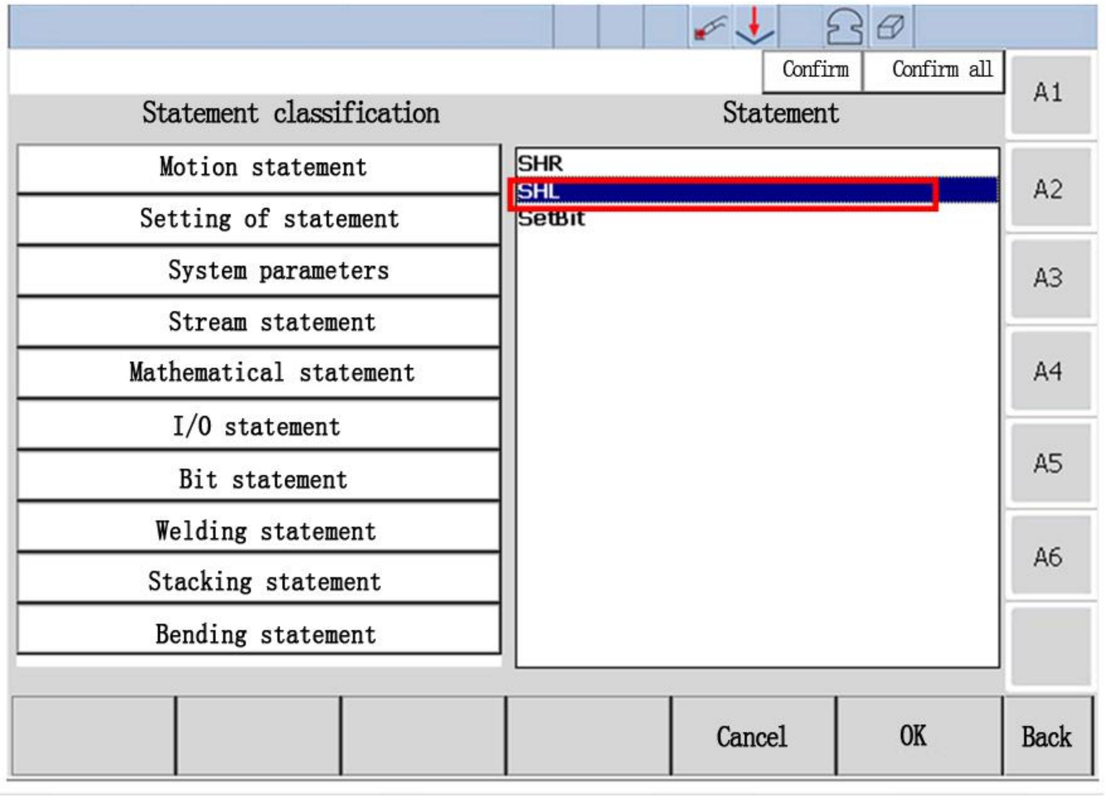
- **Sample**

  int0:=SHL(int3,int4);

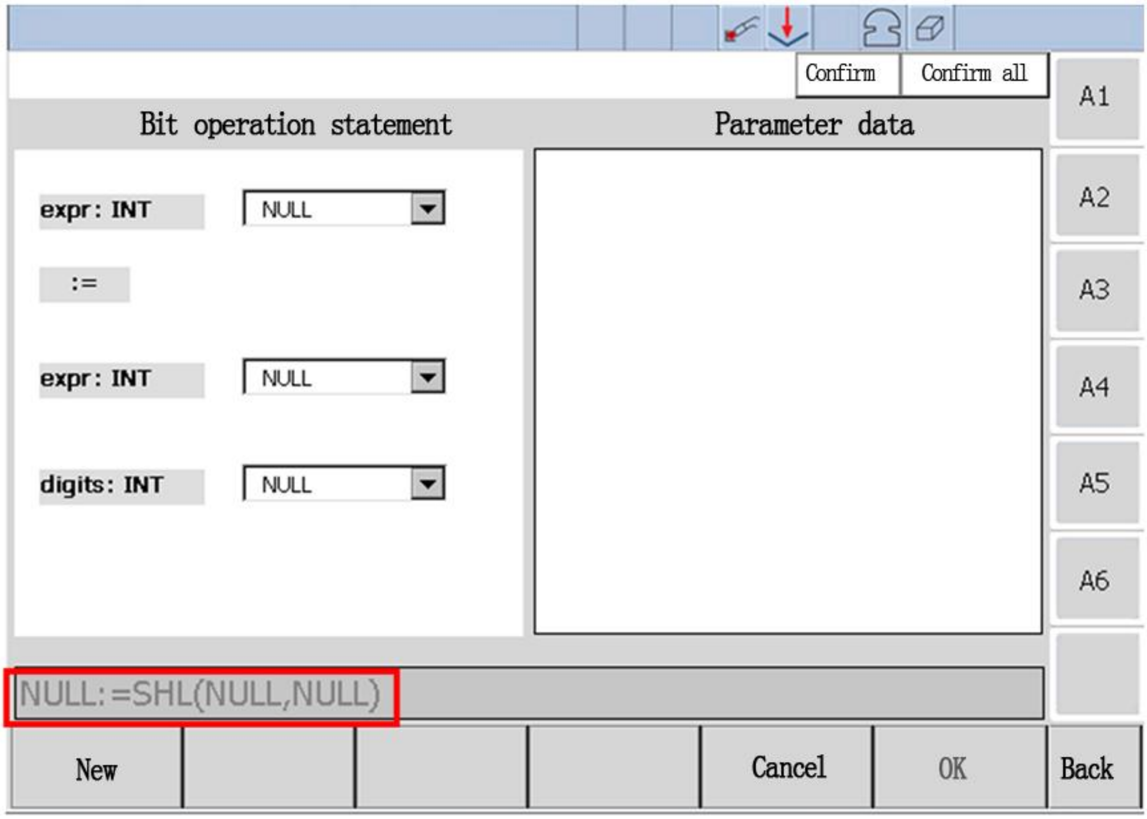
  Program function: move int3 int4 bits to the left and then assign it to int0.

3.7.2 SHR

- Create the statement SHR as shown in the following diagram.



- Click OK and enter the following interface



expr: INT NULL

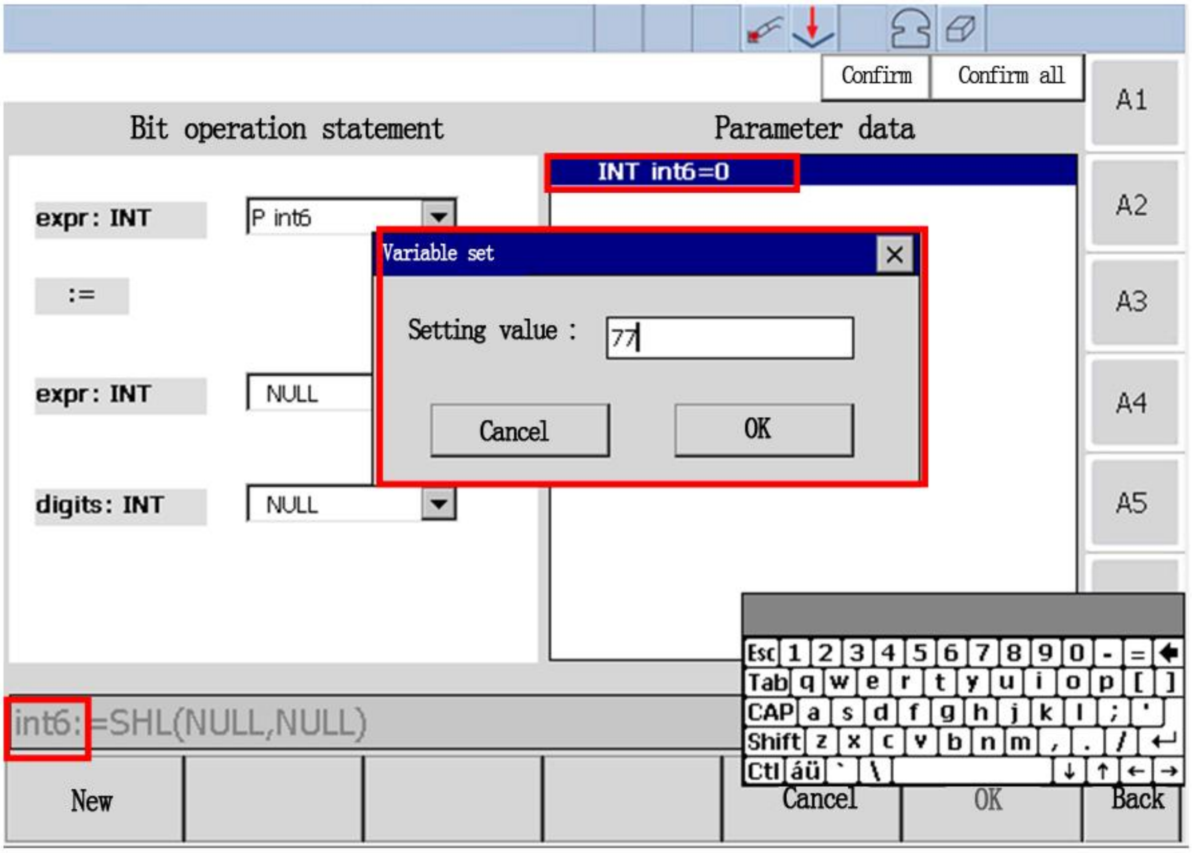
:=

expr: INT NULL

digits: INT NULL

NULL:=SHL(NULL,NULL)

- Next create the variable and set the parameter.



INT int6=0

expr: INT P int6

:=

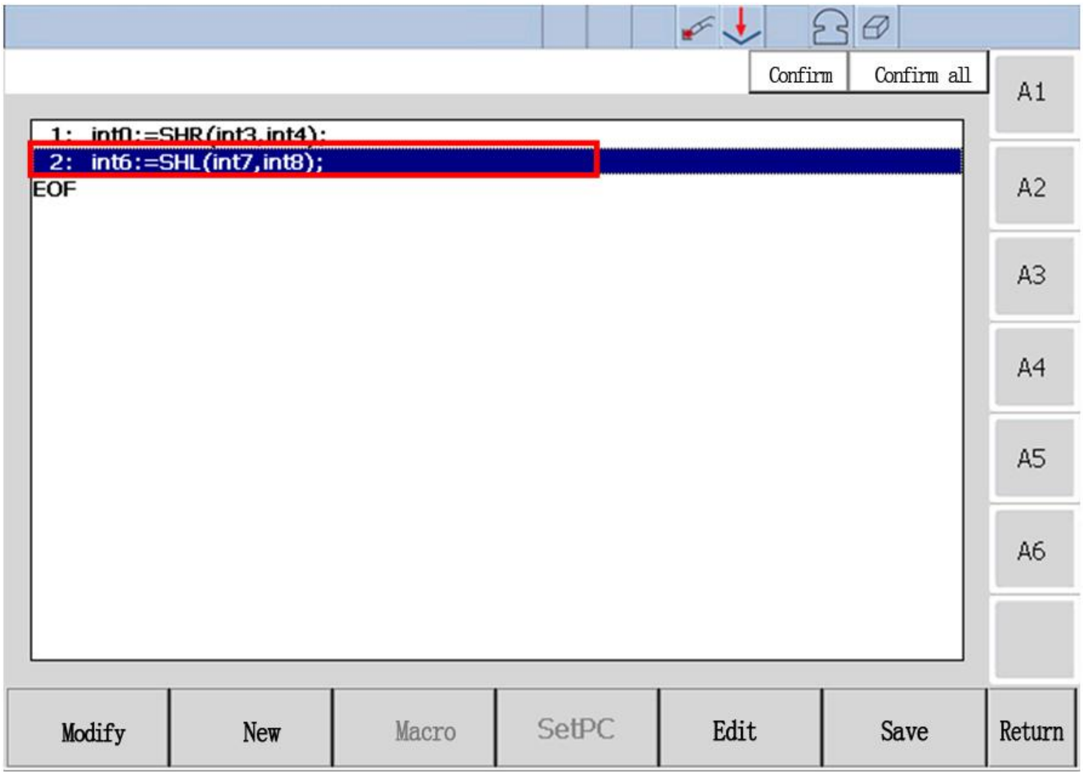
expr: INT NULL

digits: INT NULL

Setting value : 77

int6:=SHL(NULL,NULL)

- After setting all the three variables, click OK and complete the addition of the statement as shown in the following diagram.



```
1:  int0:=SHR(int3,int4);
2:  int6:=SHL(int7,int8);
EOF
```

- **Statement format**

int:=SHR（Expr，digit）；

- **Parameter**

  Expr the value that will be moved

  digits the bits of the value that will be moved

- **Function description**

  Move Expr digit bits to the right and then assign it to int.

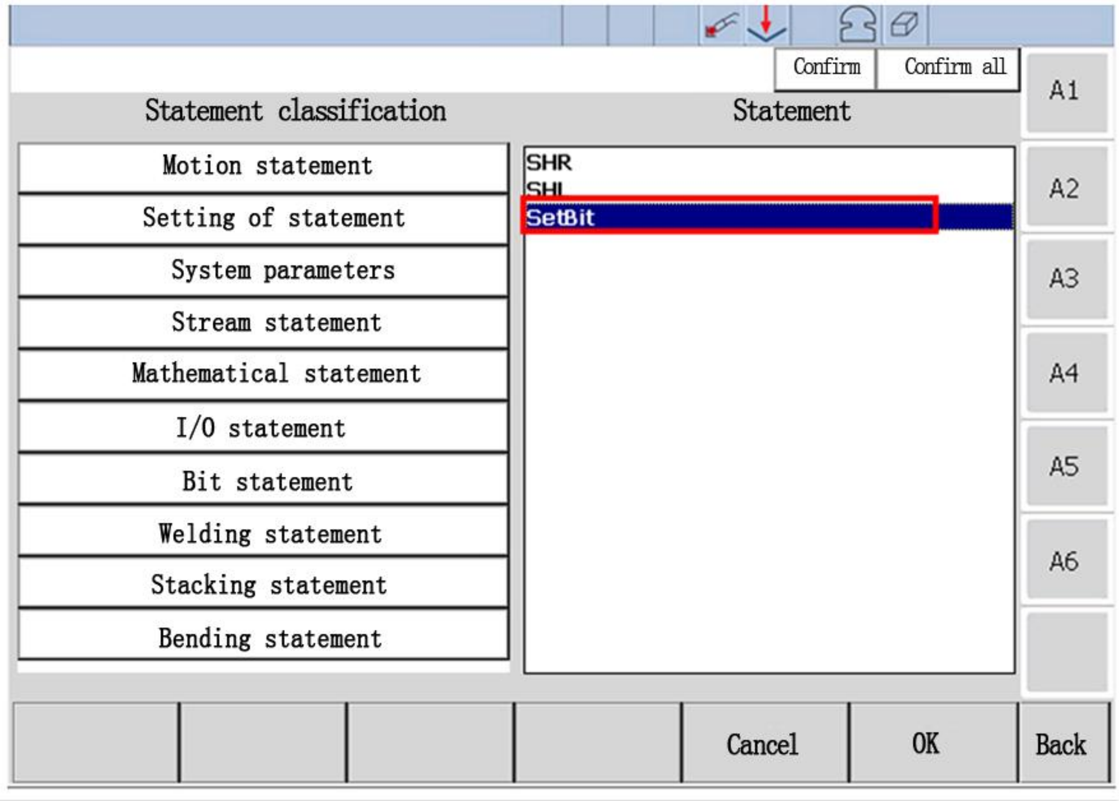
- **Sample**

  int6:=SHR(int7,int8);

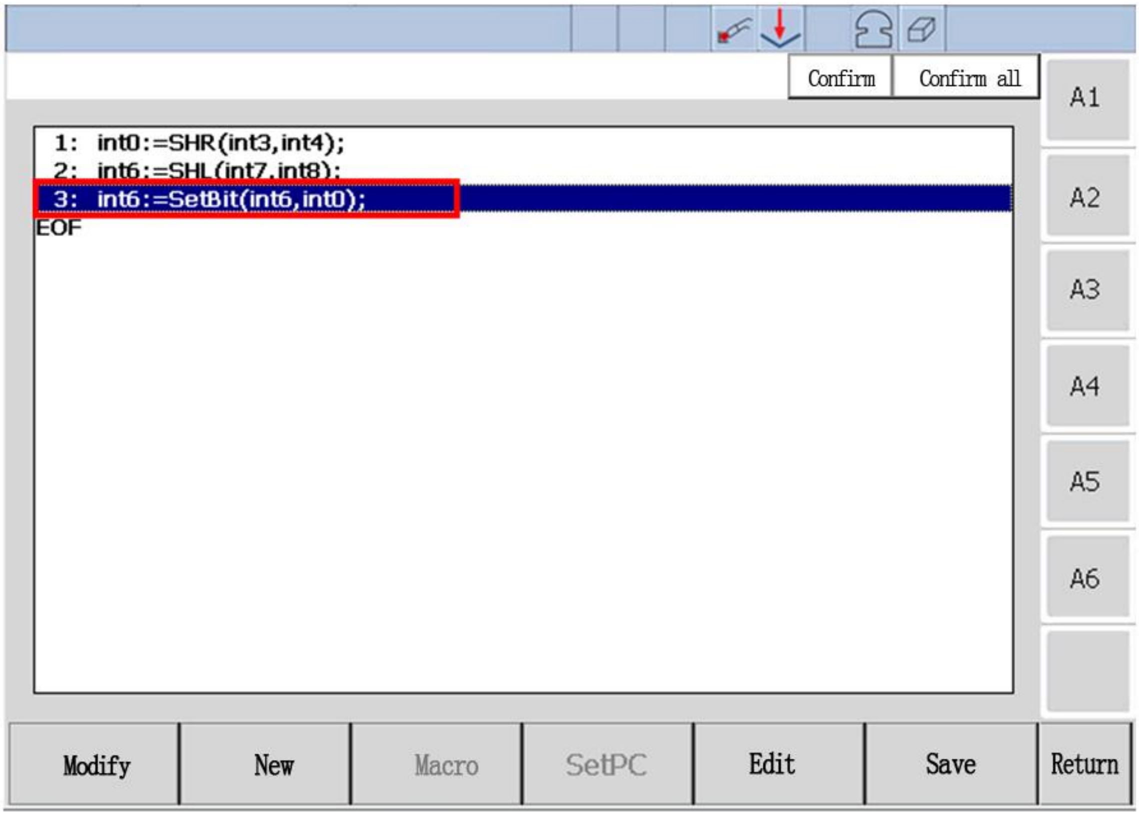
  Program demonstrates: move int7 int8 bits to the right and then assign it to int6.

3.7.3 SetBit

- Create SetBit as shown in the following diagram.



- For the same principle, create the statement, create and set the variable. After completion click OK to complete the addition of the statement.



```
1:  int0:=SHR(int3,int4);
2:  int6:=SHL(int7,int8);
3:  int6:=SetBit(int6,int0);
EOF
```

- **Statement format**

```
int:=SetBit（val,bitNr）；
```

- **Parameter**

  val   the value that will be set

  bitNr   the bit that needs to be set

- **Function description**

  Set the value of bitNr bit in val to 1 and then assign it to int.

- **Sample**

  int6:=SetBit(int6,int7);

  Program function: set the int7th position of int6 to 1 and then assign it to int6.

3.8 Operators

3.8.1 Mathematical operators

+ add operation

- minus operation

* multiply operation

/ division operation

% complementation

Usage: both sides can be digits or the basic data type variable

3.8.2 Relational operators

<   less than

<= less than or equal to

>   more than

>= more than or equal to

=   equal to

!= unequal to

Usage: both sides can be digits or the basic data type variable

3.8.3 Separators and other symbols

:   colon

Usage: used as the assignment symbols

,   comma

Usage: used between parameters.

( ) braces (be applicable for the parameter, initialization list and an arithmetic expression)

Usage: used after the instruction that needs parameters.

Appendix: Error codes and processing methods

ID	Description in English	Description in Chinese	Reason	Processing method
0		Servo error	Servo error	It can see the error code and processing of the servo.
				Suggested processing method:
				1) Clear the error to continue to operate; if the error is still reported, this sort of error needs to be corrected manually.
				2) If it is unable to clear the error, it needs to check whether the wire is virtually connected, whether the control mode of the servo is right, and whether the motor or servo is damaged.

100	Robot is not initiated	Robot is not initiated	The initialization block isn’t called in PLC program.	The initialization block needs to be added in PLC program.
				The new software and engineering shall be provided by the System Department.
101	Parameter is illegal	Parameter is illegal	-	Submit a BUG
102	Pointer is Zero		-	Submit a BUG
103	Robot is not ready to switch on	Robot is not ready to switch on	A certain servo axis is not ready.	When checking the situation of a servo, if a servo is reporting the error, manual operation shall be carried out to solve the servo error.
150	Error change operation mode when robot moving	Cannot change the operation mode when the robot is running	The operation mode is changed when the robot is moving.	If it is the human that turns the key when robot is running, it is the normal error report. It shall avoid to directly switch the locomotion pattern when the robot is moving. It can suspend motion firstly before switching.

151	Robot is disabled when it is moving	Robot is disabled when it is running	The enable is cut off when the robot is running.	It cannot operate the enable when the robot is running, therefore, the error happens. It needs check whether the servo reports the error or not. If it isn't the error caused by the manual operation, it is likely that the wire which is set up to operate the robot to a certain place breaks off or is interrupted in a certain place. It needs to check the line and consider to add the line shield.
200	No base joint	No joint axis	1) The joint axis isn't equipped; 2) the configuration file isn't contacted with.	1) Check whether the configuration files is damaged or not; 2) the receiver function shall be called in the correct order in PLC program.
201	Joint number is wrong	The number of joint is wrong	When receiving the type of the joint, if the input joint name is an Aux joint and it doesn't exist, this	Same as 200

			error will be reported.	
202	Aux joint number is zero	The number of aux joints is zero	When the number of aux joints is zero and it isn't a guide rail, the error is reported.	Same as 200
203	Aux joint number is out of three	The number of aux joints is more than three in a aux joint system.	Same as 200	Same as 200
250	Error Create Queue	Queue creation is failed	-	Submit a BUG
251	Program is NULL	Program is NULL	-	Submit a BUG
252	No enough Memory	No enough memory	-	Submit a BUG
280	Segment Queue is NULL	Path queue pointer is NULL	-	Submit a BUG
281	Segment Queue is full	Path queue is full	-	Submit a BUG
282	Segment Queue is empty	Path queue is empty	-	Submit a BUG
283	Segment Queue is out of range	Path queue is out of range	-	Submit a BUG

300	MovePoint Queue is wrong	MovePoint queue is wrong	-	Submit a BUG
301	MovePoint Queue is NULL	MovePoint queue is NULL	-	Submit a BUG
302	MovePoint Queue is full	MovePoint Queue is full	-	Submit a BUG
303	MovePoint Queue is empty	MovePoint queue is empty	-	Submit a BUG
304	MovePoint Queue is clear	MovePoint Queue is clear	-	Submit a BUG
305	MovePoint Queue is out of range	MovePoint Queue is out of range	-	Submit a BUG
320	Queue is full	Queue is full	-	Submit a BUG
321	Queue is empty	Queue is empty	-	Submit a BUG
322	Queue is out of range	Queue is out of range	-	Submit a BUG
323	Queue item is already exist	Queue item already exists	-	Submit a BUG
324	Queue item is not exist	Queue item does not exist	-	Submit a BUG
340	Stack is full	Stack is full	-	Submit a BUG
341	Stack is empty	Stack is empty	-	Submit a BUG
342	Path end flag	Path end flag	-	Submit a BUG
343	Stack is out of range	Stack is out of	-	Submit a BUG

		range		
360	Too many reference system in a program	Too many reference coordinate systems in a program	The used reference coordinate systems are more than 50 in a program.	Decrease the number of the reference coordinate system.
500	ErrNotAutomaticExternal	Not in the external automatic mode	When calling function, the robot isn’t in the external automatic mode.	The robot shall be switched to the external automatic mode.
501	ErrFBBusy	"Busy" program block	PLC Interface function is offending.	PLC program is suggested to be simplified.
502	ErrNoControlAuthority	No control authority	When PLC Interface function is called, there is no control authority.	The authorized function shall be called in PLC program.
550	Illegal correct dimension	Illegal dimension of searching correction	Wrong searching bits	Submit a BUG
551	Illegal CD when correct	Illegal searching parameter	When correcting, it uses the illegal searching data.	In accordance with the right locating operation, using the searching

				function by firstly searching and then correcting.
552	the CD Refsyses are not same	The CD Refsyses are different in searching correction program	Using the searching points which are in the different reference coordinate systems.	It can be corrected by using the searching points in the same coordinate system.
553	Laser searching while touch device	Touch sensor is selected as the laser searching sensor.	The laser searching function is called when the touch searching is equipped.	Determine whether to use the laser searching or the touch searching. If the touch searching is used, the touch searching is selected in the searching statement. If the laser searching, the laser sensor needs to be equipped.
554	Wire contacting before searching	The wire contacts with the workpiece before searching	The wire contacts with the workpiece before searching.	1. Adjust the torch position; 2. Adjust the workpiece position.
555	No workpiece is found during searching	No workpiece is found during searching	Workpiece is out of the searching range.	1. Increase the searching scope; 2. Adjust the workpiece position

556	Position of workpiece out of search distance	Position of the workpiece is out of searching distance	Worrpiece is out of the acceptable deviation range.	1. Increase the deviation tolerance; 2. Adjust the workpiece position
557	Illegal search process		-	Submit a BUG
558	Arc on out of time	Arc on out of time	Arcing time is more than 10s.	1. Check whether the welding machine is normal or not.
				2. Check whether the communication between the welding machine and the robot is normal or not.
				3. Check whether the welding torch is normal or not.
559	Illegal arc on process		-	Submit a BUG
560	Arc off out of time	Arc off out of time	Arc failure is more than 10s	Check whether the communication between the welding machine and the robot is normal or not.
561	Illegal arc off process		-	Submit a BUG
562	Illegal arc settings except set,mode,job		-	Submit a BUG
563	Error RobotType was Used,not General Robot	Non-six-joint serial robot. The joint	Same as the Description	Check the type of the robot equipped by the configuration files. If it is

		swing is only used in the general machine.		the non-six-joint serial robot, it needs to change the arcing means.
564	CurrentTool not appropriate for WristWeave	Current tool is not appropriate for Wrist Weave	Same as the Description	1. Change the tool
				2. Change the welding pose.
565	Do not support double search yet	Do not support double searching yet	-	Submit a BUG
1000	Joint velocity limit is NULL	Joint velocity limit parameter is NULL	1) The joint axis isn't equipped; 2) the configuration file isn't contacted with.	1) Check whether the configuration files is damaged or not; 2) the receiver function shall be called in the correct order in PLC program.
1001	Joint limit parameters is NULL	Joint limit parameters is NULL	1) The joint axis isn't equipped; 2) the configuration file isn't contacted with.	1) Check whether the configuration files is damaged or not; 2) the receiver function shall be called in the correct order in PLC program.
1002	Joint1 is singular	Joint1 is singular	Same as the Description	Change the mode of motion to PTP or fine tune the position point.
1003	Joint1 is out of space	Joint1 is out of space	1. The joint1 is out of space;	Check whether it is out of space or not. If it is,

			2. Given cartesian position and pattern cannot get the joint solution.	the joint shall be jogged back to the space. If the error information is caused by the initialization of the path, it shall determine whether the robot can arrive at the given cartesian position and pattern or not.
1004	Joint2 is out of limit	Joint2 is out of limit	Same as 1003	Same as 1003
1005	Joint3 is out of limit	Joint3 is out of limit	Same as 1003	Same as 1003
1006	Joint4 is out of limit	Joint4 is out of limit	Same as 1003	Check whether it is out of space or not. If it is, the joint shall be jogged back to the space. If it isn’t, it shall determine whether it is located near the singular location or not. If it is near the singular location, fine tune the target point of the robot. If it is neither out of space nor singular, then reduce the operating speed. If it continues to

				report that the axis is out of space, it is the software defects.
1007	Joint5 is out of limit	Joint5 is out of limit	Same as 1003	Check whether it is out of space or not. If it is, the joint shall be jogged back to the space. If it isn’t, it shall determine whether it is located near the singular location or not. If it is near the singular location, fine tune the target point of the robot. If it is neither out of space nor singular, then reduce the operating speed. If it continues to report that the axis is out of space, it is the software defects.
1008	Joint6 is out of limit	Joint6 is out of limit	Same as 1003	Check whether it is out of space or not. If it is, the joint shall be jogged back to the space. If it isn’t, it shall determine

				whether it is located near the singular location or not. If it is near the singular location, fine tune the target point of the robot. If it is neither out of space nor singular, then reduce the operating speed. If it continues to report that the axis is out of space, it is the software defects.
1009	Joint3 does not have Real solve	Joint3 does not have solution	Same as Description	Same as 1002
1010	Joint2 does not have Real solve	Joint2 does not have solution	Same as Description	Same as 1002
1011	Joint5 is singular	Joint5 is singular	Same as Description	Same as 1002
1012	Joint5 is singular and joint6 is out of limit	Joint5 is singular and joint6 is out of limit	Same as Description	Same as 1002
1013	AuxJoint1 is out of limit	Aux joint1 is out of limit	Aux joint1 is out of limit or speed.	When the aux joint1 is out of limit or speed, it shall determine whether the aux joint 1 is

				reported to be out of limit or not. If it is, jog it back. If it isn’t, reduce the speed of the path.
1014	AuxJoint2 is out of limit	AuxJoint2 is out of limit.	Same as 1013	Same as 1013
1015	AuxJoint3 is out of limit	AuxJoint3 is out of limit.	Same as 1013	Same as 1013
1016	AuxJoint4 is out of limit	AuxJoint4 is out of limit.	Same as 1013	Same as 1013
1017	AuxJoint5 is out of limit	AuxJoint5 is out of limit.	Same as 1013	Same as 1013
1018	AuxJoint6 is out of limit	AuxJoint6 is out of limit.	Same as 1013	Same as 1013
1019	PalletizerRobotOri is err	Posture error of Cartesian points in four-joint palletizer robot	The wrong posture of Cartpos	When the user sets the Cartpos pose, please input the correct posture or demonstrate the cartpos point by jogging.
1020	Joint1 is out of speed	Joint1 is out of speed	Same as Description	Reduce the speed or the percentage of the speed. If the error still exists, submit a bug.
1021	Joint2 is out of speed	Joint2 is out of speed	Same as Description	Same as 1020
1022	Joint3 is out of speed	Joint3 is out of speed	Same as Description	Same as 1020
102	Joint4 is out of speed	Joint4 is out of	Same as	Same as 1020

3		speed	Description	
1024	Joint5 is out of speed	Joint5 is out of speed	Same as Description	Same as 1020
1025	Joint6 is out of speed	Joint6 is out of speed	Same as Description	Same as 1020
1026	Joint5 is near singular and joint 4 out of speed	Joint5 is almost singular and joint 4 out of speed	Same as Description	Same as 1020
1027	Joint5 is near singular and joint 6 out of speed	Joint5 is almost singular and joint 6 out of speed	Same as Description	Same as 1020
1500	Error PTP initial	PTP path initialization error	-	Submit a BUG
1501	Error PTPREL initial	PTPREL path initialization error	-	Submit a BUG
1502	Error Line initial	Linear path initialization error	-	Submit a BUG
1503	Error LineREL initial	Relative linear path initialization error	-	\Submit a BUG
150	Error Circle initial	Circle path	-	Submit a BUG

4		initialization error		
1505	Error CircleAngle initial	CircleAngle path initialization error	-	Submit a BUG
1506	Error Spline initial	Spline path initialization error	-	Submit a BUG
2000	DH parameter is NULL	DH parameter pointer is NULL	1) The joint axis isn’t equipped; 2) the configuration file isn’t contacted with.	1) Check whether the configuration files is damaged or not; 2) the receiver function shall be called in the correct order in PLC program.
2001	DH parameter is NULL in Inv_kinematics	DH parameter pointer is NULL in inv_kinematics	1) The joint axis isn’t equipped; 2) the configuration file isn’t contacted with.	1) Check whether the configuration files is damaged or not; 2) the receiver function shall be called in the correct order in PLC program.
2002	Error happened in getting DH Parameters	Error happens in getting DH Parameters	1) The joint axis isn’t equipped; 2) the configuration file isn’t contacted with.	1) Check whether the configuration files is damaged or not; 2) the receiver function shall be called in the correct order in PLC program.
2100	Reference type is wrong	The type of reference	-	Submit a BUG

		coordinate system is wrong		
2101	Position type is wrong	Position type is wrong	-	Submit a BUG
2102	Axis type is wrong	Axis type is wrong. Not the liner axis or the rotation axis	-	Submit a BUG
2103	Ramp type is wrong	Velocity planning type is wrong	-	Submit a BUG
2104	Segment type is wrong	Path type is wrong	-	Submit a BUG
2200	override parameter is wrong	Speed control parameter is wrong	-	Submit a BUG
2201	target length is less than zero	The length of the path target is less than zero	-	Submit a BUG
2202	Cycle time less than zero	Interpolation cycle is less than zero	1) The joint axis isn't equipped; 2) the configuration file isn't contacted with.	1) Check whether the configuration files is damaged or not; 2) the receiver function shall be called in the correct order in PLC program.
220	Blend condition is not one	Overlap sign	-	Submit a BUG

3	or zero	is wrong, which is not one or zero		
2204	Overlap parameter is not appropriate	Overlap parameter is not appropriate	-	Submit a BUG
2205	Orientation interpolation type is wrong	Orientation interpolation type is wrong	-	Submit a BUG
2206	Velocity is not appropriate	Velocity parameter is less than zero	-	Submit a BUG
2207	Acceleration is not appropriate	Acceleration parameter is not appropriate	-	Submit a BUG
2208	Deceleration is not appropriate	Deceleration parameter is not appropriate	-	Submit a BUG
2209	Jerk is not appropriate	Jerk is not appropriate	-	Submit a BUG
2300	Position is NULL	Position pointer is NULL	-	Submit a BUG
2301	Segment is NULL	Path pointer is NULL	-	Submit a BUG
230	RobotBase is NULL	Pointer of the	-	Submit a BUG

2		robot base coordinate system is NULL.		
2500	THe parameter x of sqrt(x) less than zero	The parameter x of sqrt(x) is less than zero	-	Submit a BUG
2501	Rcpos transfer to Posfull error	Structure error by the transfer from Rcpos to Posful	-	Submit a BUG
2502	Error in KineForward function	Forward kinematics error	-	Submit a BUG
2503	Error get robot mode	Robot mode error	-	Submit a BUG
2504	Error Velocity Profile	Velocity profile error		Submit a BUG
3000	JogPath is NULL	Jogging structure pointer is NULL	-	Submit a BUG
3001	JogSegment is NULL	Jogging path pointer is NULL	-	Submit a BUG
3200	Jog Reference system is wrong	Jogging reference coordinate	-	Submit a BUG

		system is wrong		
3201	MoveType Parameter is wrong	Jogging axle type is wrong	-	Submit a BUG
3202	JogMode Parameter is wrong	Jogging mode is wrong		Submit a BUG
3203	Error in T_velocity	Jogging T_velocity planning error	-	Submit a BUG
3204	Error in Absolute Jogging initial	Linear jogging path initialization error at absolute positions	-	Submit a BUG
3205	Get JointPos error in increment Jogging	Joint position error after getting the joint increment jogging	-	Submit a BUG
3206	Get CartPos error in increment Jogging	CartPos error after getting Cartesian increment jogging	-	Submit a BUG
3207	Get JointPos error in absolute (PTP) Jogging	JointPos error after getting PTP jogging at absolute	-	Submit a BUG

		positions		
3208	Get cartPos error in absolute (Line) Jogging	CartPos error after getting linear jogging at absolute positions	-	Submit a BUG
3209	SubDyn is NULL	Velocity structure pointer is NULL	-	Submit a BUG
3210	Direction type is wrong	Jogging direction is wrong	-	Submit a BUG
3211	Jogging Unit is too big or to small	Jogging Unit is too big or to small	-	Submit a BUG
3212	Path velocity greater than 250mm/s	Path velocity is greater than 250mm/s	-	Submit a BUG
3300	Rotation to RPY singular	RPY from the pose matrix to the fixed angle is singular	-	Submit a BUG
3400	Not find Program variable or var illegal	Not find or illegal to use variable		Check whether the variable exists or not from the variable interface; check whether the statement allows to

				insert the variable. If the variable doesn’t exist really, create a new variable with the same name.
3401	Program variable type not exist	Variable type is wrong	Variable type is wrong	Check srd file and manually delete the variable.
3402	Program variable Name empty	Variable name is empty	Variable name is empty	Check srd file and revise based on the format definition.
3403	Program bool variable value err	Boolean variable is wrong	Boolean variable is wrong	Check srd file and revise based on the format definition.
3404	Program numerical variable value err	Format error of integral and floating-point data values	Format error of integral and floating-point data values	Check srd file and revise based on the format definition.
3405	Program string variable value err	Format error of string variable values	Format error of string variable values	Check srd file and revise based on the format definition.
3406	"{"or"}" err	Lack of brace	Lack of brace	Check srd file and revise based on the format definition.
3407	Program ORITYPE variable value err	ORITYPE variable definition error	ORITYPE variable definition error	Check srd file and revise based on the format definition.
340	Program RAMPTYPE	RAMPTYPE	RAMPTYPE	Check srd file and revise

8	variable value err	variable definition error	variable definition error	based on the format definition.
3409	Program Define Var Twice	Variable redefinition	Variable redefinition	Check srd file and delete the unnecessary variables
3410	Program Ref Var Not Find	Variable definition error in the coordinate system	Variable definition error in the coordinate system	Check srd file and revise based on the format definition.
3411	Not Find Project File,Created in Controller	Lack of project files	All project files are deleted.	Create a new project file based on the definition.
3412	Config File Load Failure	Config File Load Failure	Config File Load Failure	Copy a correct configuration file from a normal operating robot.
3413	Config File lose root element	Config File loses the root element	Config File loses the root element	Copy a correct configuration file from a normal operating robot.
3414	Unknown Robot Type	Unknown Robot Type	Unknown Robot Type	Copy a correct configuration file from a normal operating robot.
3415	XML file data irregular	Invalid data in Xml file	Invalid data in Xml file	Copy a correct configuration file from a normal operating robot.
3416	ArcWelding file Load Failure	Welding file definition error	Welding file definition error	Copy a correct welding file from a normal operating robot.

3417	Welding File data irregular	Invalid data in a welding file	Invalid data in a welding file	Copy a correct welding file from a normal operating robot.
3418	Welding File lose root element	Welding File loses the root element.	Welding File loses the root element.	Copy a correct welding file from a normal operating robot.
3419	Touch File load failed	Searching file load failure	Searching file load failure	Copy a correct searching file from a normal operating robot.
3420	Touch File data irregular	Invalid data in a searching file	Invalid data in a searching file	Copy a correct searching file from a normal operating robot.
3421	Touch File lose root element	Searching file loses the root element	Searching file loses the root element	Copy a correct searching file from a normal operating robot.
3422	CD file data num err	Data number error in a searching file	Data number error in a searching file	Copy a correct searching file from a normal operating robot.
3423	ID description file err	Error description file goes wrong	Error description file goes wrong	Copy a correct error description file from a normal operating robot
3424	MultiLayer data file lost	Multilayer and multichannel data file is missing	Forget to copy or import it	Copy a correct error description file from a normal operating robot
3425	MultiLayer file lose root	Multilayer and multichannel data file is damaged	system files process abnormally.	Submit a BUG

3426	MultiLayer file lost variable	Lack of the multilayer and multichannel variables	The variables are deleted by maloperation	Recreate it
3427	MultiLayer pathMemo SetPC forbid	Setpc operation is restricted for use	Setpc operation cannot be used at the moment.	Setpc is started from somewhere else or the first line.
3428	Cannot find the Pallet Var	Cannot find the Pallet Var	The variables are deleted by maloperation	Recreate it
3429	Cannot find the Pallet ref	Cannot find the Pallet reference coordinate system	The variables are deleted by maloperation	Recreate it
3430	Pallet data initial error	Abnormal pallet position	Pallet data has problems, which may be the singular position.	Redemonstrate the pallet data
3431	Pallet act num error,greater or less	The number of running pallet variable will be greater or less than the set value	The number of pallet isn’t consistent with the actual number.	Check the program logic or reload.
3432	Pallet dynamic param error	Singular Dyn param in the pallet	The variables are deleted by maloperation	Recreate it

		statement		
3433	Pallet overlap param error	Singular Ovl parameter in the pallet statement	The variables are deleted by maloperation	Recreate it
3434	Pallet unknown command error	Unknown pallet statement type	System files process abnormally.	Submit a BUG
3435	Cannot find the Pallet finish bool var	Singular Bool param in the pallet statement	The variables are deleted by maloperation	Recreate it
3436	Cannot find the Pallet Reset var	Pallet variable is missing	The variables are deleted by maloperation	Recreate it
3437	Aux num greater than you have purchased	The number of equipped AuxJoints is greater than that of what you have purchased	Contact with the sailor	Contact with the sailor
3438	SP variable used after MultiLayer Initial	SP variable must be used	Firstly initialize, and then use SP	SP variable must be used after the multilayer

		after the multilayer and multichannel initialization	point.	and multichannel initialization. See the instruction.
3439	Pallet variable number out limit	The number of pallet variables is beyond the limits.	At most ten pallet variables are built into a program.	Delete the unnecessary pallet variable
3440	Cannot find the MultiLayer data	Cannot find the multilayer and multichannel data	The variables are deleted by maloperation	Recreate it
3441	Pathmemo cannot towards the backward setpc	Pathmemo can only record the previous statements.	Setpc operation cannot be used at the moment.	Setpc is started from somewhere else or the first line.
3442	Pathmemo not initial	The statement must be used after the multilayer and multichannel initialization	Firstly initialize, and then use SP point.	The statement must be used after the multilayer and multichannel initialization. See the instruction.
3443	Pathmemo param not find	Lack of Pathmemo parameter	The variables are deleted by maloperation	Recreate it

3444	PointCorrect not accordance with MulitiLayer	Parameters in PointCorrect statement aren't consistent with the current multilayer and multichannel variables.	Usage error	Reconfirm the using method. See the instruction.
3445	PointCorrect 2th param error	Lack of the 2nd parameter in PointCorrect statement	The variables are deleted by maloperation	Recreate it
3446	PointCorrect 3th param error	Lack of the 3rd parameter in PointCorrect statement	The variables are deleted by maloperation	Recreate it
3447	PointCorrect 4th param not find	Lack of the 4th parameter in PointCorrect statement	The variables are deleted by maloperation	Recreate it
3448	LayerLoop not accordance with MulitiLayer	Parameters in LayerLoop statement aren't consistent with the current	Usage error	Reconfirm the using method. See the instruction.

		multilayer and multichannel variables.		
3449	LayerLoop cycle time outrange	The number of loops in LayerLoop is beyond the set.	Usage error	Reconfirm the using method. See the instruction.
3450	PTP 1th param not find	Cannot find the 1st parameter in the statement	The variables are deleted by maloperation	Recreate it
3451	PTP 1th param type error	The 1st parameter type error in the statement	Usage error	Reconfirm the using method. See the instruction.
3452	PTP 2th param error	Cannot find the 2nd parameter in the statement	The variables are deleted by maloperation	Recreate it
3453	PTP 3th param error	Cannot find the 3rd parameter in the statement	The variables are deleted by maloperation	Recreate it
3454	Lin 1th param not find	Cannot find the 1st parameter in the statement	The variables are deleted by maloperation	Recreate it
345	Lin 1th param type error	The 1st	Usage error	Reconfirm the using

5		parameter type error in the statement		method. See the instruction.
3456	Lin 2th param error	Cannot find the 2nd parameter in the statement	The variables are deleted by maloperation	Recreate it
3457	Lin 3th param error	Cannot find the 3rd parameter in the statement	The variables are deleted by maloperation	Recreate it
3458	Lin 4th param error	Cannot find the 4th parameter in the statement	The variables are deleted by maloperation	Recreate it
3459	Circ 1th param not find	Cannot find the 1st parameter in the statement	The variables are deleted by maloperation	Recreate it
3460	Circ 1th param type error	The 1st parameter type error in the statement	Usage error	Reconfirm the using method. See the instruction.
3461	Circ 2th param not find	Cannot find the 2nd parameter in the statement	The variables are deleted by maloperation	Recreate it
3462	Circ 2th param type error	The 2nd parameter	Usage error	Reconfirm the using method. See the

		type error in the statement		instruction.
3463	Circ 3th param error	Cannot find the 3rd parameter in the statement	The variables are deleted by maloperation	Recreate it
3464	Circ 4th param error	Cannot find the 4th parameter in the statement	The variables are deleted by maloperation	Recreate it
3465	Circ 5th param error	Cannot find the 5th parameter in the statement	The variables are deleted by maloperation	Recreate it
3466	CircAngle 1th param not find	Cannot find the 1st parameter in the statement	The variables are deleted by maloperation	Recreate it
3467	CircAngle 1th param type error	The 1st parameter type error in the statement	Usage error	Reconfirm the using method. See the instruction.
3468	CircAngle 2th param not find	The 2nd parameter type error in the statement	The variables are deleted by maloperation	Recreate it
3469	CircAngle 2th param type error	The 2nd parameter type error in	Usage error	Reconfirm the using method. See the instruction.

		the statement		
3470	CircAngle 3th param error	Cannot find the 3rd parameter in the statement	The variables are deleted by maloperation	Recreate it
3471	CircAngle 4th param error	Cannot find the 4th parameter in the statement	The variables are deleted by maloperation	Recreate it
3472	CircAngle 5th param error	Cannot find the 5th parameter in the statement	The variables are deleted by maloperation	Recreate it
3473	CircAngle 6th param error	Cannot find the 6th parameter in the statement	The variables are deleted by maloperation	Recreate it
3474	Search_L 1th param not find	Cannot find the 1st parameter in the statement	The variables are deleted by maloperation	Recreate it
3475	Search_L 1th param type error,not support	The 1st parameter type error in the statement	Usage error	Reconfirm the using method. See the instruction.
3476	Search_L 2th param not find	Cannot find the 2nd parameter in the statement	The variables are deleted by maloperation	Recreate it

3477	Search_L 3th param not find	Cannot find the 3rd parameter in the statement	The variables are deleted by maloperation	Recreate it
3478	Search_T 1th param not find	Cannot find the 1st parameter in the statement	The variables are deleted by maloperation	Recreate it
3479	Search_T 1th param type error,not support	The 1st parameter type error in the statement	Usage error	Reconfirm the using method. See the instruction.
3480	Search_T 2th param not find	Cannot find the 2nd parameter in the statement	The variables are deleted by maloperation	Recreate it
3481	Search_T 3th param not find	Cannot find the 3rd parameter in the statement	The variables are deleted by maloperation	Recreate it
3482	CORR_1D param not find	Cannot find the 1st parameter in the statement	The variables are deleted by maloperation	Recreate it
3483	CORR_2D 1th param not find	Cannot find the 1st parameter in the statement	The variables are deleted by maloperation	Recreate it
348	CORR_2D 2th param not	Cannot find	The variables are	Recreate it

4	find	the 2nd parameter in the statement	deleted by maloperation	
3485	CORR_3D 1th param not find	Cannot find the 1st parameter in the statement	The variables are deleted by maloperation	Recreate it
3486	CORR_3D 2th param not find	Cannot find the 2nd parameter in the statement	The variables are deleted by maloperation	Recreate it
3487	CORR_3D 3th param not find	Cannot find the 3rd parameter in the statement	The variables are deleted by maloperation	Recreate it
3488	CORR_6D 1th param not find	Cannot find the 1st parameter in the statement	The variables are deleted by maloperation	Recreate it
3489	CORR_6D 2th param not find	Cannot find the 2nd parameter in the statement	The variables are deleted by maloperation	Recreate it
3490	CORR_6D 3th param not find	Cannot find the 3rd parameter in the statement	The variables are deleted by maloperation	Recreate it
3491	CORR_6D 4th param not find	Cannot find the 4th	The variables are deleted by	Recreate it

		parameter in the statement	maloperation	
3492	CORR_6D 5th param not find	Cannot find the 5th parameter in the statement	The variables are deleted by maloperation	Recreate it
3493	CORR_6D 6th param not find	Cannot find the 6th parameter in the statement	The variables are deleted by maloperation	Recreate it
3494	ARCON param not find	Lack of parameters in the statement	The variables are deleted by maloperation	Recreate it
3495	ARCOFF param not find	Lack of parameters in the statement	The variables are deleted by maloperation	Recreate it
3496	ARCSET param not find	Lack of parameters in the statement	The variables are deleted by maloperation	Recreate it
3500	PLC Command data is empty	PLCInterface command data is empty	Data transmission is chaos.	Check whether the revision information is consistent or not.
3501	Authority is busy	Authority is occupied.	When one interface is getting the authority, other operation interfaces don’t release it.	Determine whether other operation interfaces release the authority or not. If not, the granted authority can be released through the demonstrator or PLC

				programming.
3510	Load program when loading state	Reload the program	Reload the program	If a new program is loaded, firstly unload the loaded program firstly.
3511	program is illegal format	Invalid program format		If the program statement format goes wrong, revise the statement. See the statement format in the instruction.
3512	Adjacency list construction failure	The inner construction of the program is failed	Inner construction of a program is failed.	Reload and submit a bug
3513	Program file wrong	Program file name goes wrong		For lack of a program file, create a new file with the same name. Note: the suffix is srp
3520	Start or load command is failure	Start or load command is failed		Submit a BUG
3521	Software not register	Unregistered software	The software isn’t registered correctly	It needs to get the correct registration code and register.
3522	Invalid register key	Invalid registration code	The registration code is wrong.	It needs to get the correct registration code and register.

3600	Load userinfo file fail	User information file load failure	Lack of the user information file	Copy a correct user file from a normal operating robot
3700	LinISR Pose type not accordance	When the types of the input and output positions aren't consistent in the statement, it reports an error.	The types aren't consistent	Reconfirm the using method. See the instruction.
3701	LinISR Pose type forbid	The statement doesn't allow this type of variable	the type restriction error	Reconfirm the using method. See the instruction.
3702	FeedBackPose type forbid	The feedback position type isn't allowed	the type restriction error	Reconfirm the using method. See the instruction.
3703	LinISR 1th param not find	Cannot find the 1st parameter in the statement	The variables are deleted by maloperation	Recreate it
3704	LinISR 1th param type error,not support	The 1st parameter type error in the statement	Usage error	Reconfirm the using method. See the instruction.
370	LinISR 2th param not find	Cannot find	The variables are	Recreate it

5		the 2nd parameter in the statement	deleted by maloperation	
3706	LinISR 3th param not find	Cannot find the 3rd parameter in the statement	The variables are deleted by maloperation	Recreate it
3707	LinISR 4th param not find	Cannot find the 4th parameter in the statement	The variables are deleted by maloperation	Recreate it
3708	LinISR 5th param not find	Cannot find the 5th parameter in the statement	The variables are deleted by maloperation	Recreate it
3709	LinISR 6th param not find	Cannot find the 6th parameter in the statement	The variables are deleted by maloperation	Recreate it
3711	WLin 1th param not find	Cannot find the 1st parameter in the statement	The variables are deleted by maloperation	Recreate it
3712	WLin 1th param type error,not support	The 1st parameter type error in the statement	Usage error	Reconfirm the using method. See the instruction.
3713	WLin 2th param not find	Cannot find the 2nd	The variables are deleted by	Recreate it

		parameter in the statement	maloperation	
3714	WLin 3th param not find	Cannot find the 3rd parameter in the statement	The variables are deleted by maloperation	Recreate it
3715	WLin 4th param not find	Cannot find the 4th parameter in the statement	The variables are deleted by maloperation	Recreate it
3716	WLin 5th param not find	Cannot find the 5th parameter in the statement	The variables are deleted by maloperation	Recreate it
3717	WLin 6th param not find	Cannot find the 6th parameter in the statement	The variables are deleted by maloperation	Recreate it
3718	WCirc 1th param not find	Cannot find the 1st parameter in the statement	The variables are deleted by maloperation	Recreate it
3719	WCirc 1th param type error,not support	The 1st parameter type error in the statement	Usage error	Reconfirm the using method. See the instruction.
3720	WCirc 2th param not find	Cannot find the 2nd parameter in	The variables are deleted by maloperation	Recreate it

		the statement		
3721	WCirc 2th param type error,not support	The 2nd parameter type error in the statement	Usage error	Reconfirm the using method. See the instruction.
3722	WCirc 3th param not find	Cannot find the 3rd parameter in the statement	The variables are deleted by maloperation	Recreate it
3723	WCirc 4th param not find	Cannot find the 4th parameter in the statement	The variables are deleted by maloperation	Recreate it
3724	WCirc 5th param not find	Cannot find the 5th parameter in the statement	The variables are deleted by maloperation	Recreate it
3725	WCirc 6th param not find	Cannot find the 6th parameter in the statement	The variables are deleted by maloperation	Recreate it
3726	WCirc 7th param not find	Cannot find the 7th parameter in the statement	The variables are deleted by maloperation	Recreate it
3727	ARCJOB param not find	Parameters cannot be found in the statement	The variables are deleted by maloperation	Recreate it

3728	ARCMODE param not find	Parameters cannot be found in the statement	The variables are deleted by maloperation	Recreate it
3729	This type var not support save operation	The type of parameters cannot be saved in the file	Usage error	Reconfirm the using method. See the instruction.
3730	Cannot call global or project var	Cannot call global or project var in the statement	Usage error	Reconfirm the using method. See the instruction.
3731	JiaoDi File load failed	Car platform data file load failure	The file is missing or damaged	Copy a correct error description file from a normal operating robot
3732	JiaoDi File lose root element	Car platform data file is damaged	The data are invalidly missing or damaged	Copy a correct error description file from a normal operating robot
3733	JiaoDi File data irregular	Data in car platform data file are invalid	The data are invalidly missing or damaged	Copy a correct error description file from a normal operating robot
3734	HMI request file not find	The file needed by the demonstrator doesn’t exist.	Lack of the file	Create a new file with the same name
3735	HMI send file not exist in NonReal	The file needed by the demonstrator	Lack of the file	Create a new file with the same name

		doesn’t exist.		
3800	Weave Data Format Err	Invalid format of swing arc variables	The format isn’t consistent with the original one after the software upgrading.	Add two new data in the end of variables, both of which is 0.
3801	Bool Data Format Err	Invalid format of bool variables	Invalid missing or damaging of data	Delete the original variables and create a new variable with the same name.
3802	Only Input UINT Type Var	Only Input UINT Type Var	Usage error	Reconfirm the using method. See the instruction.
3803	variables not within the scope	Variable is not within the scope.	Usage error	Reconfirm the using method. See the instruction.
4000	Program has not loaded	Program has not loaded	When starting the program, there is no program loading.	The robot shall load the program before running it.
4001	Robot has not enabled	Robot has not enabled	When starting the program, the robot has not enabled	The robot shall have enabled before running it.
4002	Do not jog when robot run in auto_mode	Cannot jog when robot runs in auto_mode	Cannot jog when robot runs in auto_mode	Switch to the manual mode to jog

4003	Do not Load a new program when robot is running	Cannot load a new program when robot is running	Cannot load a new program when robot is running	It shall not load a new program when robot is running
4004	Do not jog when robot is moving	Cannot jog when robot is running	Cannot jog when robot is running	It shall not jog when robot is running
4005	Do not kill the program when robot is moving	Cannot close the program when robot is running	Not follow the requests to operate	It shall firstly suspend or stop and then close the program,
4006	Do not Load a new program when robot is moving	Cannot load a new program when robot is running	Not follow the requests to operate	It shall firstly close the program and then load the program after stopping the robot.
4007	Do not operate program when robot is moving	Cannot edit program when robot is running	Not follow the requests to operate	It shall edit program after stopping the robot.
4009	Do not change run mode when robot is moving	Cannot change run mode when robot is running	Not follow the requests to operate	It shall change run mode after stopping the robot.
4010	Do not active ToolAlign when robot is moving	Cannot active ToolAlign when robot is running	Not follow the requests to operate	It shall active ToolAlign after stopping the robot.
401	Do not press Start when	Cannot press	Not follow the	It shall press Start after

1	doing ToolAlign	Start when robot is doing ToolAlign.	requests to operate	stopping the robot.
4012	Do not active ToolAlign in AutoMode	Cannot active ToolAlign in AutoMode	Not follow the requests to operate	Switch to the manual mode
4013	Do not active JogToPosition when robot is moving	Cannot active JogToPosition when robot is running	Not follow the requests to operate	It shall make the robot stop.
4014	Do not press Start when doing JogToPosition	Cannot press Start when robot is doing JogToPosition.	Not follow the requests to operate	Two operations cannot be carried out simultaneously.
4015	Do not active JogToPosition in AutoMode	Cannot active JogToPosition in AutoMode	Not follow the requests to operate	Switch to the manual mode
4016	The drive hardware enable is false	There is no enable in the servo hardware.		Open the servo hardware enable.
4017	Error Not Clear	Error isn't clear.	Not clear the error before opening the enable	Open the enable after clearing the error
4018	Not Disable When ProgramRuning	Close the enable when robot is	Close the enable when robot is running.	Closing the enable temporarily

		running.		
4500	Relative overlap parameter is greater than one	Relative overlap parameter is greater than one	-	Submit a BUG
4501	Relative overlap parameter is less than zero	Relative overlap parameter is less than zero	-	Submit a BUG
4502	OvlAbs parameter LinAx is less than zero	Absolute overlap linear velocity is less than zero	-	Submit a BUG
4503	OvlAbs parameter JointDeg is less than zero	Absolute overlap angular velocity is less than zero	-	Submit a BUG
4600	The orientation is not Norm	The postures aren’t normalized	-	Fine tune the target point of robot