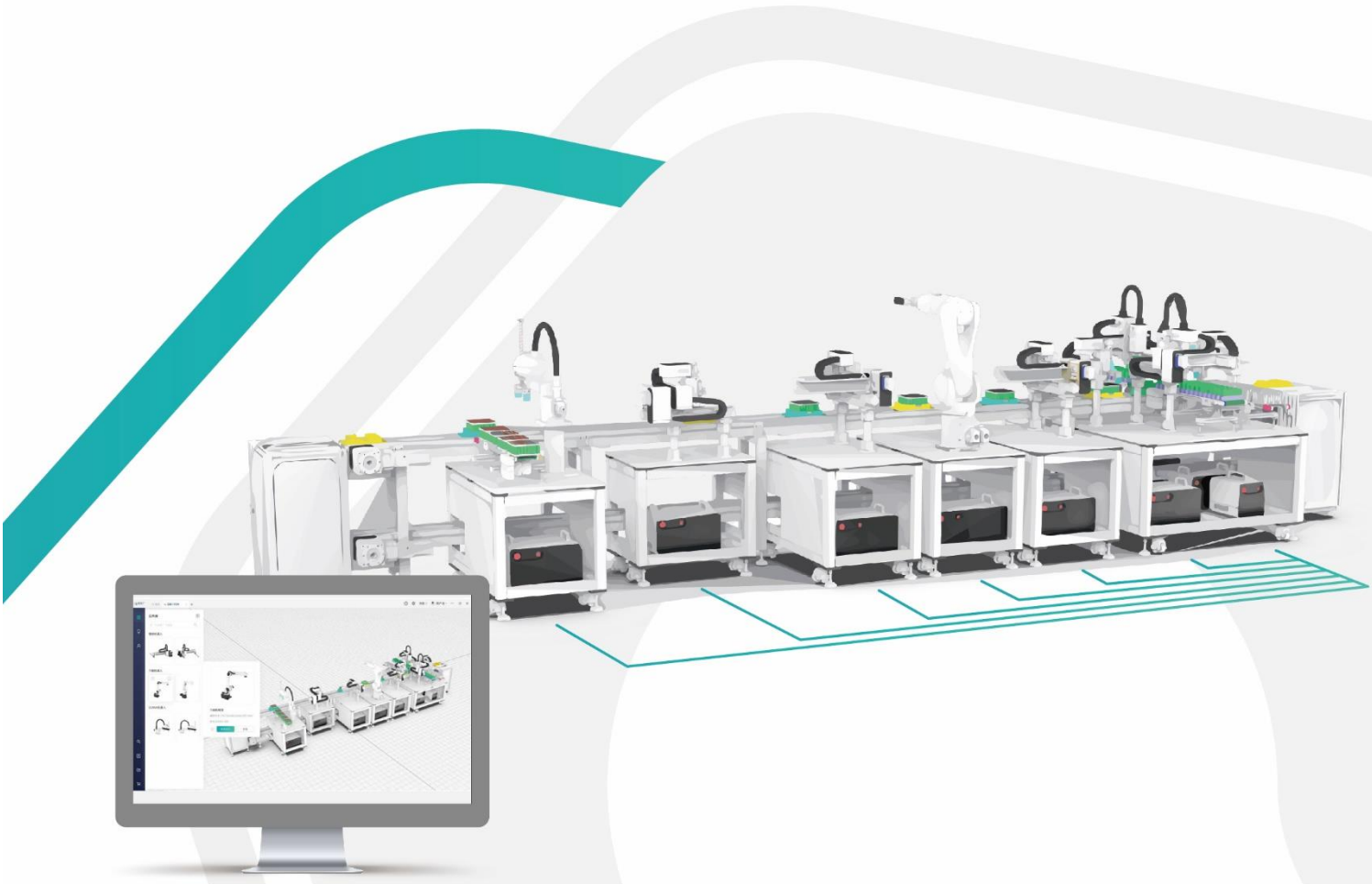
COCTRL
杭州芯控智能科技有限公司

FSM-ControlSys 示教器软件

用户手册

V1.7



杭州芯控智能科技有限公司
HANGZHOU CORE CONTROL ROBOTICS CO.,LTD.

感谢购买本公司产品，本手册为安全使用本公司产品而需要遵守的内容，在使用之前，请务必仔细阅读相关手册，并且在理解该内容的前提下正确使用。

有关本产品的详细功能，请用户通过相关说明书充分理解其规格。

我们试图在本手册中描述尽可能多的情况。然而，对于那些不必做的和不可能做的情况，由于种种原因，我们没有描述。因此对于那些在手册中没有描述的情况，可以视为“不可能”的情况。

- **版权声明**

- **版本更新记录**

变更日期	版本号	适配软件版本	变更说明
2023.03.03	V1.7	V2.2 及以下所有版本	更新示教器界面、增加相关应用示例
2022.11.10	V1.6	V2.2 及以下所有版本	更新六轴机器人 ACS 坐标系建系方向
2022.10.08	V1.5	V2.2 及以下所有版本	更新 4.4.1 控制器网络扫描与连接操作说明
2022.09.13	V1.4	V2.2 及以下所有版本	统一术语、更新添加 4.6.1 章节用户坐标系说明
2022.08.10	V1.3	V2.1 所有版本	参数配置二维码扫描添加功能添加
2022.07.01	V1.2	V2.1 所有版本	全局变量内容添加
2022.06.15	V1.1	V2.1 所有版本	Delayset、GOTO 指令添加、TCP 通讯示例添加
2022.06.01	V1.0	V2.0 所有版本	初版文件发布

目录

1 控制系统介绍

1.1 基本情况介绍

示教器软件 FSM-ControlSys（以下简称 FC 软件）基于 Corecontroller 开放式编程软件引擎开发，为杭州芯控智能科技有限公司（以下简称杭州芯控）自研产线规划软件（软件名：FactorySteam，以下简称 FSM 软件）附属控制器模块，可支持 FSM 软件虚拟仿真场景内机器人等执行机构控制，同时搭配杭州芯控自研 C²Cube 运动控制柜，可实现多种型制机器人示教、编程、运行。

FC 软件运行于 Windows 系统，可独立于示教器硬件，运行于笔记本电脑、主机、工控一体机等其它主机设备。

1.2 多台控制拓扑关系

FC 软件搭配示教器作为上位机，可通过 C²Cube 运动控制柜，实现单一示教器控制多台机器人或多伺服轴独立/协同动作，所有机器人程序可使用单一示教器编辑并独立运行，控制系统拓扑关系如下：

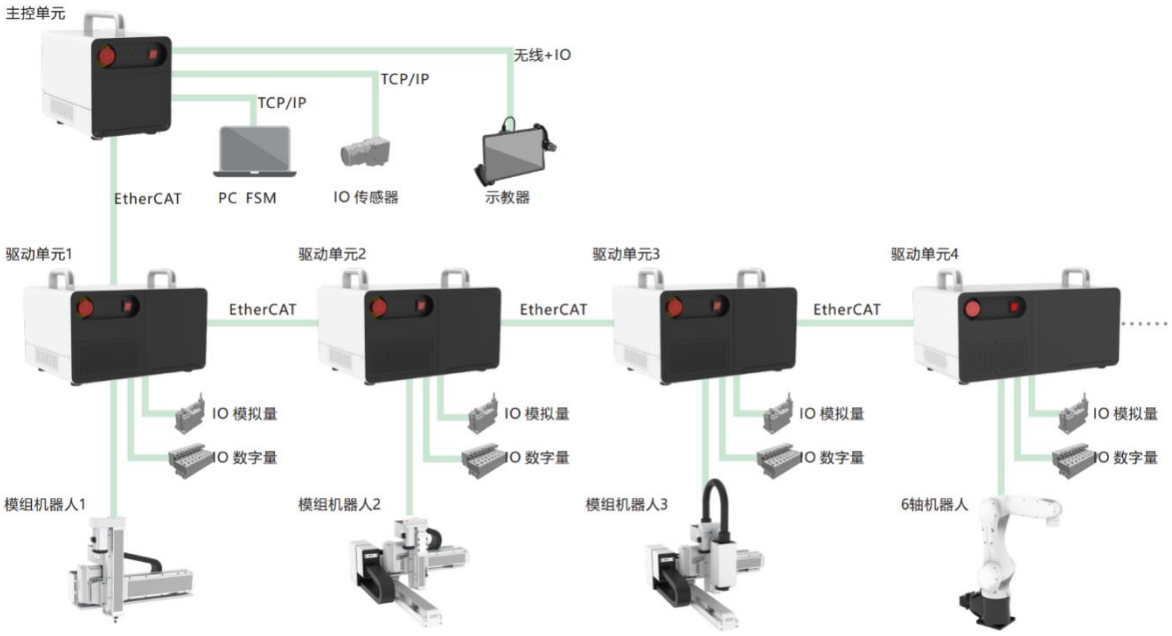
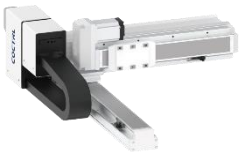
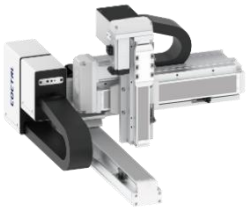


图 1

1.3 支持功能总览

FC 软件支持 Corecontroller 软件所有基础功能，同时针对开发机器人编程语言。相关功能总览如下：

- **先进功能：**
  1. 支持以太网、WLAN（2.4GHz wifi）方式连接控制器；
  2. 支持多台机器人使用同一示教设备编程调试；
  3. 支持单台示教设备调试多台控制器；
  4. 机器人语言编程；
- **支持轨迹控制功能：**
  1. 轨迹规划: PTP 运动、CP 运动（混合 Blend 运动）；
  2. 速度曲线：梯型加/减速、二次曲线型加/减速、S 曲线型加/减速；
- **支持机器人运控算法：**
  1. 2 轴/3 轴/4 轴悬臂及龙门坐标机器人运动算法库；
  2. 耦合及非耦合式 3 轴/4 轴 SCARA、6 轴机器人运动算法库；
  3. AGV、复合机器人运动算法库；

<table>
<tr><td colspan="2">2 轴/3 轴/4 轴悬臂及龙门坐标机器人：</td></tr>
<tr><td>  </td><td>  </td></tr>
<tr><td colspan="2">2 轴十字型悬臂坐标机器人</td></tr>
<tr><td>  </td><td>  </td></tr>
<tr><td>3 轴悬臂坐标机器人</td><td>4 轴悬臂坐标机器人</td></tr>
<tr><td colspan="2">注:龙门型机器人为长跨距悬臂机器人加辅助滑轨组成</td></tr>
</table>

SCARA 机器人、6 轴机器人：	
	
3/4 轴 SCARA（L1-一二轴线性式）机器人	3/4 轴 SCARA（L2-二轴线性式）机器人
	
3/4 轴 SCARA（L3-三轴线性式）机器人	6 轴机器人
AGV、复合机器人：	
	
二维码导航式 AGV	AGV+六轴复合机器人

2 完整控制系统组成介绍

2.1 示教器软件基本介绍

FC 软件（完整调试上位机）由集成开发环境（即 IDE 模块）与 Corecontroller 编译引擎组成，为减少对硬件性能占用，FC 软件 IDE 模块与编译引擎已做拆分，除杭州芯控所供标准设备外，亦可通过安装软件选配安装于不同 Windows10 系统调试设备。

2.1.1 教器软件适配硬件要求

具备以下条件的 PC 机或便携式 PC 机

FC 软件 IDE 模块与编译引擎全装，性能要求：	
系统	Windows 10，推荐 64 位操作系统
CPU	主频 2GHz 及以上
内存	3GB 及以上
硬盘空间	2.5GB 及以上
FC 软件 IDE 模块独立安装，性能要求：	
系统	Windows 10，推荐 64 位操作系统
CPU	1GHz 及以上
内存	1GB 及以上
硬盘空间	500MB 及以上
编译引擎独立安装，性能要求：	
系统	Windows 10，推荐 64 位操作系统
CPU	1.5GHz 及以上
内存	2GB 及以上
硬盘空间	2GB 及以上
注： 上述性能参数为控制器协同控制 16 轴时测定数据，所控轴数增加，控制引擎所需内存要求越高。	

2.1.2 标准设备安装及适配情况

安装设备	设备型号	FC 软件		控制器 Runtime
		IDE	编译引擎	
控制柜（单主板）	C10-19 系列	×	×	√
控制柜（单主板）	C10-i5 系列	×	√	√
控制柜（双主板）	C10-19-19 系列	×	√	√
单体控制器	MC200	×	×	√
驱控一体柜（单主板）	C30-19 系列	×	×	√
驱控一体柜（单主板）	C40-19 系列	×	×	√
示教器	FP2000	√	√	×

2.1.3 完整控制系统构成及连接方式

完整控制系统由 FC 软件（即上位机，包含 **IDE 模块**与**编译引擎**）、控制器/柜（Runtime 下位机）及受控设备构成。

编程调试设备通过以太网（可经过集线器、交换机等）或 WLAN（2.4GHz wifi）与控制器/柜连接，使用 FC 软件编写用户程序，将程序下载到控制器/柜中运行、监控。

2.1.4 软件的获取

软件安装包获取途径：

如采购为杭州芯控智能科技有限公司提供标准示教设备，其内部已安装相应软件，无需独立下载安装。

如需独立下载软件安装包，需在杭州芯控智能科技有限公司官网（www.corecontrol.cn）的“产品选型”对应“下载中心”页面下载。

由于杭州芯控智能科技有限公司在不断完善产品和资料，建议用户在需要时及时更新软件版本，查阅最新发布的参考资料，有利于用户的应用设计。

3 软件界面及按键功能介绍

3.1 软件界面整体展示

FC 软件可实现多台机器人同时控制，所有界面分为三类：进入首页、工程配置界面（包含功能：机器人及其协同设备添加、全局变量编程页添加）、机器人及其协同设备运动控制编程界面（包含功能：参数配置、示教编程、工具标定、用户标定、IO 配置），相关界面如下：

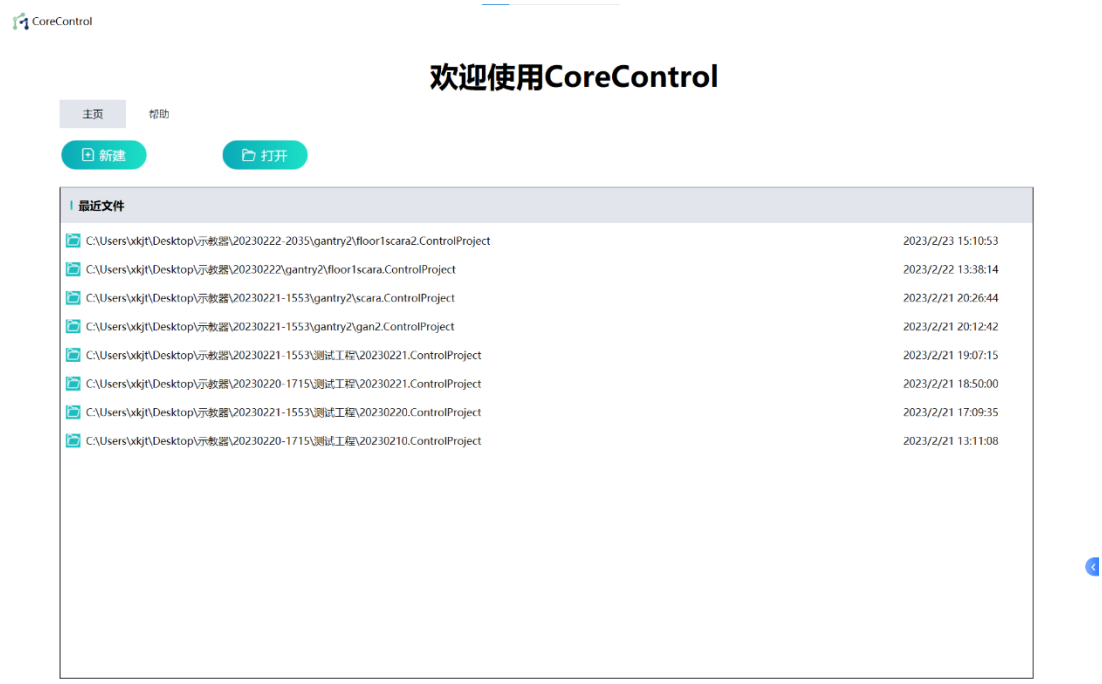


图 2 进入首页

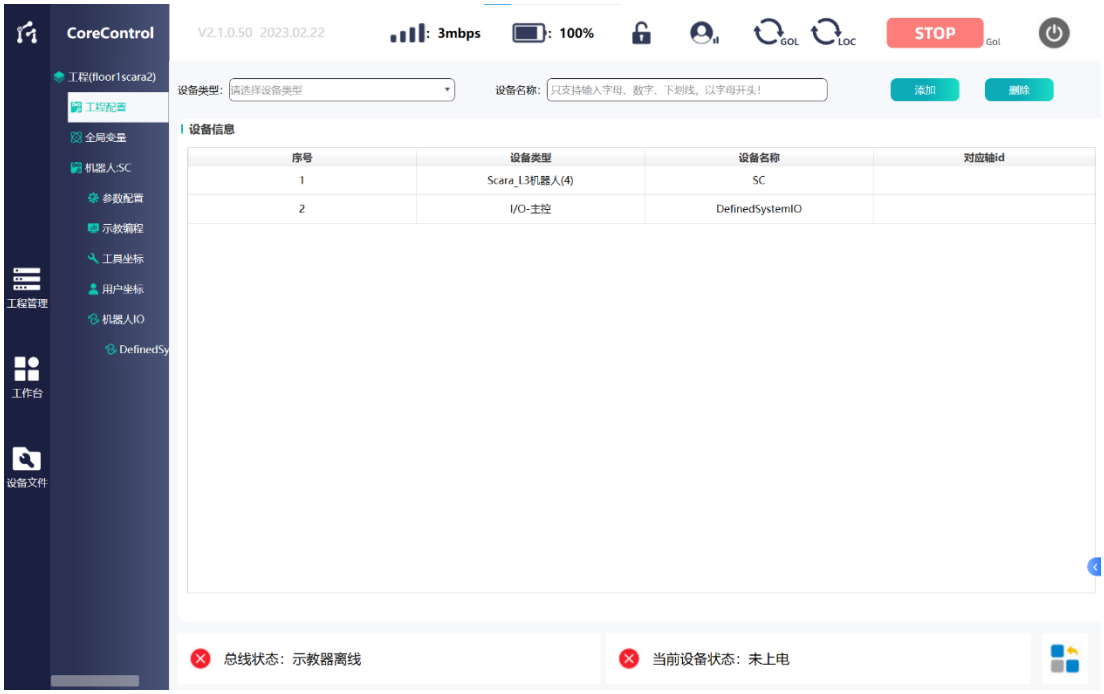


图 3 工程配置页面

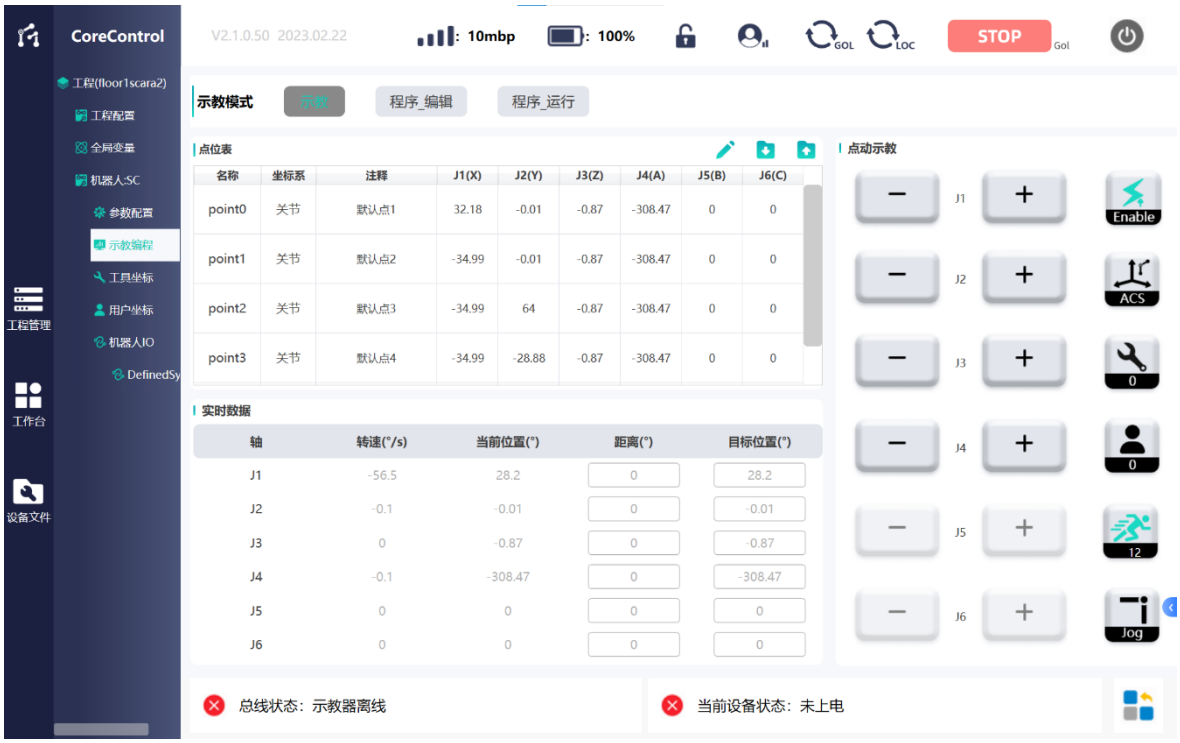


图 4 机器人及其协同设备运动控制编程界面

3.2 按键功能介绍

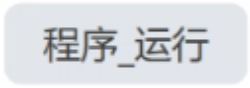
示教器上所有按键功能介绍如下：

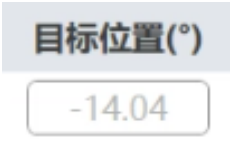
按键图示	名称	功能
	主页按钮	回到示教器主页面
	帮助按钮	获取示教器帮助页面
	新建工程按钮	新建一个示教器工程
	打开工程按钮	打开一个已经存在的工程
	确定按钮	确定一系列操作
	取消按钮	否定一系列操作
	选择网络 下拉菜单	显示所有连接到的控制器网络
	重新扫描按钮	重新扫描控制器网络
	关闭按钮	关闭当前页面
	锁定按钮	锁定示教器页面，放置误触
	用户管理按钮	切换管理员权限
	局部复位按钮	复位当前机器人
	全局复位按钮	复位整个控制器和全部机器人
	急停按钮	急停
	退出按钮	退出示教器
	电量显示	显示当前示教器的剩余电量

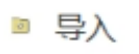
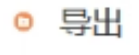
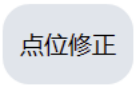
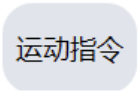
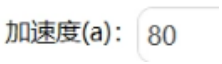
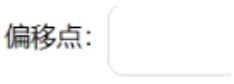
	延迟显示	显示当前示教器的通讯延迟
	工程管理按钮	按下后弹出工程管理相关的按钮
	新建工程按钮	新建一个示教器工程
	打开工程按钮	打开一个示教器工程
	保存工程按钮	保存当前示教器工程
	工程另存为按钮	将当前工程另存为一个工程
	登录设备按钮	登录网络选择中的控制器
	退出登录按钮	退出当前登录的控制器
	键盘按钮	弹出屏幕键盘
	日志按钮	显示控制器日志
	网络选择按钮	选择控制器网络
	算法安装按钮	安装各类机器人算法库和运行库
	安装设备描述文件按钮	安装 EtherCat 设备的设备描述文件
	工程配置按钮	修改当前工程中含有的机器人和 IO 设备
	设备选择菜单栏	选择需要添加的机器人和 IO 类型（支持的类型：6 轴机器人、UR 机器人、SCaraL1 机器人、SCaraL2 机器人、SCaraL3 机器人、4 轴直角坐标机器人、三轴直角坐标机器人、IO）

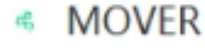
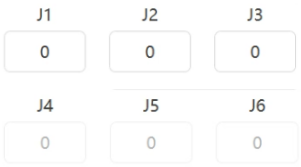
	设备名称输入栏	对选择要添加的机器人和 IO 设备命名 （只支持输入字母、数字、下划线，并以字母开头）
	添加设备按钮	将选中的设备类型添加至控制设备中
	删除设备按钮	删除设备列表中选中的的机器人或 IO 设备
	全局变量配置按钮	配置控制器中的全局变量
	变量名输入栏	对需要添加的全局变量命名
	变量类型选择菜单	对需要添加的全局变量数据类型进行选择（支持的类型：BIT、BOOL、BYTE、INT、LINT、LREAL、REAL、STRING、STRINGLIST）
	变量初始值赋值输入栏	对需要添加的全局变量数据类型进行初始值赋值
	变量确定按钮	添加配置完全的全局变量
	变量删除按钮	删除已经添加的全局变量
	机器人参数配置按钮	对添加的机器人进行详细参数配置
	关节选择菜单栏	对机器人各个轴单独选择
	设备生产商选择菜单	配置选中轴的驱动器生产厂家（支持的类型：松下、汇川等支持 EtherCat 的总线设备）
	设备类型选择菜单	配置驱动器的具体类型（如汇川 630、660、松下 MBDLN05BE 等）
	总线地址选择菜单	选择该设备在 EtherCat 总线下的从站地址，由 1000 开始
	重新扫描设备按钮	重新扫描控制器下的所有 EtherCat 设备

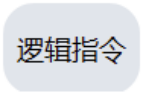
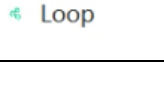
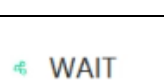
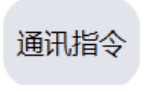
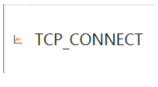
反向	驱动器反向确认	勾选反向后，电机的正方向与不勾选时相反
脉冲数 < = >电机圈数	脉冲数对应电机圈数	脉冲数和电机圈数的对应关系，输入多少个脉冲电机就旋转几圈
电机圈数 < = > 减速机输出圈数	减速比	电机输入多少圈对应减速机输出多少圈
减速机输出圈数 < = >程序应用单位	末端输出单位	减速机输出圈数对应直线模组前进多少 mm 或旋转末端旋转多少度
软件错位反应 加速度[u/s²]:	软件错位反应加速度	轴出现报警时，程序停下轴的加速度
减速度[u/s²]:	软件错位反应减速度	轴出现报警时，程序停下轴的减速度
激活	软件位激活使能确认开关	勾选时软限位启用，不勾选时不启用软限位
正[°]:	正限位	轴正向能到达的最大位置
负[°]:	负限位	轴反向能到达的最大位置
零位编码值:	零位编码值	轴位于零位时对应驱动器中的编码值
S曲线	速度斜坡类型	轴的速度斜坡类型
虚拟模式 模数 有限	轴的类型	轴的类型，可以选择有限轴、模数轴或虚拟轴
旋转 平移	电机类型	可以选择直线电机或旋转电机
速度[mm/s]:	轴速度动态限制	程序中轴能达到的最大速度
加速度[mm/s²]:	轴加速度动态限制	程序中轴能达到的最大加速度
减速度[mm/s²]:	轴减速度动态限制	程序中轴能达到的最大减速度
加加速度[mm/s³]:	轴加加速度动态限制	程序中轴能达到的最大加加速度

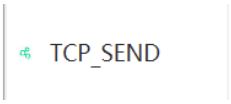
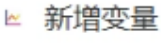
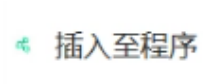
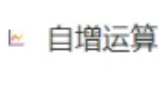
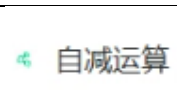
	扫描二维码按钮	二维码扫描按钮，可以扫描二维码配置机器人参数
	关节参数保存按钮	关节参数保存，可以保存当前已配置的关节参数
	机器人参数保存	当所有关节参数保存完成后，保存并添加机器人运动学模型
	示教编程按钮	切换至机器人示教和程序编辑页面
	示教按钮	切换至示教界面
	程序编辑页面按钮	切换至机器人程序编辑页面
	程序运行页面按钮	切换至机器人程序运行页面
	机器人使能按钮	按下后机器人使能，松开机器人下使能
	正向点动按钮	按下后，机器人向选中坐标系对应关节或直角坐标系对应正方向运动
	反向点动按钮	按下后，机器人向选中坐标系对应关节或直角坐标系对应反方向运动
	坐标系选择按钮	按下后，可以选择各种坐标系（关节坐标系 ACS、笛卡尔坐标系 MCS、工具坐标系 TCS、用户坐标系 PCS）
	工具坐标选择	按下后，可以选择 0~9 中任意一个工具系中的一个
	用户坐标选择	按下后，可以选择 0~9 中任意一个用户系中的一个
	点动速度选择	按下后，可以拖动滑动条改变点动速度大小

	机器人运动模式选择	按下后，可以选择机器人示教运动模式（可选 5 种模式：单轴点动、单轴相对、单轴绝对、多轴相对、多轴绝对）
	点位记录按钮	按下后，点位表会记录当前坐标系下的机器人点位
	点位导入按钮	将文件中的点位保存进点位表
	点位导出按钮	将点位表中的点位导出进文件
	记录到点按钮	将当前坐标系下的机器人点位覆盖点位表中选中的点
	点前插入按钮	将当前坐标系下的机器人点位向前插入点位表中选中的点
	点后插入按钮	将当前坐标系下的机器人点位向后插入点位表中选中的点
	移动到点按钮	长按时，机器人移动至点位表中选中的点
	删除该点按钮	按下后，删除点表中选择的对应点位
	点位操作退出按钮	按下后，关闭点位操作窗口，退出对点位的操作
	移动距离输入栏	在相对模式下，输入要移动的距离
	目标位置输入栏	在绝对模式下，输入要移动至的位置
	程序文件按钮	按下后可以对程序文件进行操作

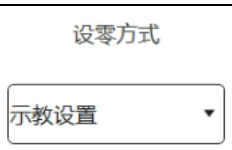
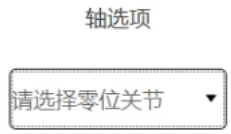
	程序保存按钮	按下后可以保存程序文件
	程序文件导入按钮	按下后可以导入程序文件
	程序文件到处按钮	按下后可以导出程序文件
	点位修正按钮	按下后可以对点位进行修正
	运动指令按钮	按下后可以插入一系列运动指令
	MOVEL 指令	按下后可以打开 MOVEL 指令插入窗口
	起点选择栏	选择需要插入的第一个直角坐标系的点
	终点选择栏	选择需要插入的最后一个直角坐标系的点（起点至终点中的全部点会自动插入，如起点时 point1，终点时 point5，若 point2、3、4 中有直角坐标系的点会自动插入）
	速度输入栏	输入运动轨迹的速度
	加速度输入栏	输入运动轨迹的加速度
	Blend 运动过度参数	在多段轨迹 Blend 运动时的过度参数
	偏移点选择栏	将运动目标点增加一段偏移
	程序语句入方式选择	选择程序末位时，程序语句会插入 ENDMAIN 前一行；选择光标位置时，程序会插入光标位置该行

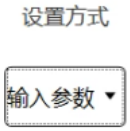
	语句插入确定按钮	点击后，语句行插入
	语句取消插入	点击后，关闭语句插入窗口
	MOVEJ 指令	按下后可以打开 MOVEJ 指令插入窗口
	MOVER 指令	按下后可以打开 MOVER 指令插入窗口
	MOVER 运动坐标系选择	选择在具体坐标系下相对于当前点运动（分为关节、笛卡尔、工具、用户坐标系）
	相对运动在坐标系下的运动距离	当为关节坐标系时，对应数字为各个关节的相对运动距离；当为直角坐标系时，对应数字为 XYZABC 方向的运动距离
	具体的工具或用户选择	坐标系选择为工具坐标系时可以选择工具 0~9；坐标系选择为用户坐标系时可以选择用户 0~9
	MOVEC 指令	按下后可以打开 MOVEC 指令插入窗口
	圆弧类型	可以选择需要运动的圆弧类型
	MOVEC 运动的第一个点	MOVEC 运动的第一个点
	MOVEC 运动的第二个点	MOVEC 运动的第一个点
	BLEND 运动插入按钮	按下后可以打开 BLEND 指令插入窗口，其中的 MOVEL、MOVEJ、MOVEC 运动将会连续且不停止

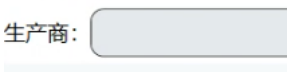
	POSITIONSET 指令插入按钮	按下后可以打开 POSITIONSET 指令插入窗口
	HOMEJ 指令插入按钮	按下后可以打开 HOMEJ 指令插入窗口
	逻辑指令按钮	按下后可以插入一系列逻辑指令
	GOTO 指令插入按钮	按下后可插入 GOTO 指令，程序跳转至对应标签下的行
	判断指令插入按钮	程序段中插入一个 IF 判断
	循环指令插入按钮	程序中插入一个循环
	While 循环插入按钮	程序中插入一个带判断的 While 循环
	设置输出指令插入按钮	程序中插入一个设置输出语句
	等待输入指令插入按钮	程序中插入一个等待输入的语句
	WAIT 等待指令插入按钮	程序中插入一个等待全局变量的语句
	DELAY 延时指令插入按钮	程序在此中断等待一定时间后继续运行
	延时输出指令插入按钮	程序设置一个后台延时，在一定时间后自动设置一个输出，程序不会在此等待
	NOTICE 指令插入按钮	程序运行至 NOTICE 指令时暂停，并弹出一个可自定义文字的窗口
	CLOCK 指令插入按钮	程序插入一个可以暂停的定时器
	通讯指令按钮	按下后可以插入一系列通讯指令
	TCP_CONNECT 指令插入按钮	程序插入一条连接至服务端的 TCP 连接指令

	IP 地址输入框	目标服务端的 IP 地址输入栏
	端口号输入框	目标服务端的端口输入栏
	TCP_SEND 指令插入按钮	程序插入一条发送数据的 TCP 发送指令
	TCP_READ 指令插入按钮	程序插入一条读取数据的 TCP 读取指令
	变量指令按钮	按下后可以插入一系列变量指令
	新增变量按钮	按下后在变量声名区新增一个局部变量
	程序内新增变量按钮	按下后在程序内声明一个局部变量
	POINTL变量	点位变量 按下后新增一个点位变量
	USERFRAME变量	
	自增运算	变量自增加按钮 插入一段语句，使选中的 INT 变量加 1
	自减运算	变量自减少按钮 插入一段语句，使选中的 INT 变量减一
	编译按钮	按下后，程序开始编译
	注释按钮	按下后，选择程序行，输入注释内容后插入至程序；或注释掉原本存在的程序

复制	程序复制按钮	按下后，复制所选的程序行
粘贴	粘贴按钮	按下后，粘贴复制的程序行
删除	删除按钮	按下后，删除所选的程序行
撤销	撤销按钮	按下后，撤销上一步操作
重做	重做按钮	按下后，重新开始上一步操作
	新增子程序按钮	按下后，新增一个子程序
	子程序命名输入框	新增子程序时的子程序名称输入栏
	删除子程序按钮	按下后，选择一个子程序进行删除
	重命名子程序按钮	按下后，选择一个子程序进行重命名
	程序运行模式选择菜单	程序执行时，可以选择执行的模式(自动单次：自动执行一遍；自动无限：循环执行主程序；手动条件：按下启动按钮后执行选中行，并执行程序中的判断；手动运动：按下按钮执行选中行，不执行程序中的判断)
	运行倍率控制条	滑动可以控制程序中运动速度的倍率
程序	程序监视按钮	按下后，监视界面切换至程序监视页面
IO	IO 监视按钮	按下后，监视界面切换至 IO 监视页面
全局变量	全局变量监视按钮	按下后，监视界面切换至全局变量监视页面

	隐藏监视按钮	按下后，隐藏监视页面
	显示监视按钮	按下后，显示监视页面
	程序执行行指针	显示当前程序正在执行的语句
	程序执行按钮	按下后，程序开始执行
	程序暂停按钮	按下后，程序暂停运行
	快捷操作首页返回按钮	按下后，快捷操作页面返回主页面
	关节零位页面跳转按钮	按下后，快捷操作页面跳转至关节零位设置页面
	机器人零位设置方式菜单栏	可以选择两种设置机器人零位的方式，（1 通过示教设置，2 通过零位编码器值设置）
	机器人设置零位轴选择菜单	可以选择机器人将要设置零位的轴，每个轴需要单独设置
	机器人零位编码值	机器人在零位上的驱动器编码器值
	设置零位按钮	按下后，机器人会根据零位设置方法确定零位；以示教方式进行设置时，会以当前位置为零位；以零位编码器值为零位时，会以给出的零位编码值为零位
	关节限位页面跳转按钮	按下后，快捷操作页面跳转至关节限位设置页面
	限位激活使能	勾选后，限位激活；取消勾选，取消限位

	限位设置方式选择菜单	有两种方式设置限位（1.输入参数：直接输入正负限位位置；2.示教设置：将机器人移动至正负限位上，点击输入栏确定正负限位值）
	设置限位轴选择菜单	选择需要设置限位的轴
	正限位输入框	正限位数值
	负限位输入框	负限位数值
	限位设置按钮	根据当前参数选择设置限位参数
	IO 输入输出页面跳转按钮	按下后，快捷操作页面跳转至 IO 输入输出页面
	输入输出值滑块	显示当前 IO 的状态， 此状态为 False，  此状态为 True；输出值可以通过此滑块快捷设置状态
	全局变量输入输出	按下后，快捷操作页面跳转至全局变量输入输出页面
	快速启动配置按钮	按下后，进入快速启动配置页面，设置主程序的快速启动操作
	工具坐标页面按钮	按下后，切换至工具坐标页面
	向上切换按钮	按下后，切换下一个工具

	向下切换	按下后，切换上一个工具
	复制按钮	按下后，复制一个工具坐标系的参数
	添加按钮	按下后，将工具坐标系的参数添加至选中的工具中
	删除按钮	按下后，删除选中工具的工具坐标系参数
	工具标定点位记录按钮	按下后，记录工具标定需要的点位中的一个
	工具标定按钮	按下后，开始工具坐标系六点标定
	用户坐标页面按钮	按下后，切换至用户坐标页面
	向上切换按钮	按下后，切换下一个用户坐标
	向下切换	按下后，切换上一个用户坐标
	复制按钮	按下后，复制一个用户坐标系的参数
	添加按钮	按下后，将用户坐标系的参数添加至选中的用户坐标中
	删除按钮	按下后，删除选中用户坐标的用户坐标系参数
	工具标定点位记录按钮	按下后，记录用户坐标标定需要的点位中的一个
	用户坐标系标定按钮	按下后，开始用户坐标系三点标定
	机器人 IO 页面按钮	按下后，切换至机器人 IO 设置页面
	IO 设备生产商选择菜单	配置选中 IO 设备的生产商（支持市面上绝大多数 EtherCat 设备）

	IO 设备型号选择菜单	配置具体 IO 设备的型号
	IO 设备总线地址选择菜单	配置 IO 设备在 EtherCat 总线中的地址（从 1000 开始）
	IO 变量名输入栏	可以输入此 IO 的名称（不支持中文）
	IO 变量关联内部变量选择菜单	可以通过外部 IO 关联内部的变量信号（共 6 个，ROBOT.POWER 机器人上电、ROBOT.STOP 机器人停止、ROBOT.SUSPEND 机器人暂停运行、ROBOT.RUN 机器人程序运行、ROBOT.RESET 机器人复位、ROBOT.QUICKSTART 机器人程序快速启动）
	IO 关联快捷动作开关	置为 True 后，可以在机器人快捷动作中快速使用输出输出功能
	IO 更改值	在此输入栏可以更改 IO 的实时值
	IO 实时值显示栏	再次栏可以显示 IO 当前的实时值
	IO 描述栏	可以在此栏添加中文描述 IO 的具体内容
	IO 设备添加按钮	做出的更改需要通过修改按钮保存，并在下一次登录时完成修改

4 示教软件功能说明

4.1 软件功能及完整使用流程章节导航

FC 软件工程为多任务的执行模式，即可以“同时”执行几个任务，每个任务可以有若干个用户程序组织单元（简称 POU），每台机器人程序即为独立 POU 程序。通过 FC 软件可以实现单一工程内多台机器人程序独立示教动作、运行、监控。

使用 FC 软件编写用户程序并下载到控制器/柜中运行、监控（即完整使用流程）需完成以下操作：

1. 调试环境准备--章节 4.2；
2. 工程文件创建/打开--章节 4.3；
3. 设备扫描添加与机械参数配置--章节 4.4；
4. 机器人启动及关闭--章节 4.5；
5. 机器人坐标系图解及手动示教--章节 4.6；
6. 工具坐标系/用户坐标系标定及相关选用--章节 4.7；
7. 辅助快捷功能--章节 4.8
8. 程序编辑及运行--章节 4.9；

4.2 调试环境准备

4.2.1 硬件安装及设备连接

C²Cube 运动控制器/柜与调试设备的连接途径有两种：以太网、WLAN（2.4GHz wifi），对应控制柜配置情况详见《C²Cube 运动控制柜使用手册》。

机器人与控制柜设备连接方式及要点详见《机器人使用手册》。

4.2.2 调试设备 IP 地址修改

须保证调试设备与 C²Cube 运动控制器/柜处于同一网段，如错误，调试软件无法扫描到控制器设备。对应现象详见章节 4.4

C²Cube 运动控制器/柜以太网连接方式对应默认 IP 地址配置为 192.168.1.143，子网掩码：255.255.0.0。

C²Cube 运动控制器/柜 WLAN 连接方式对应默认 IP 地址配置为 192.168.0.100，子网掩码：255.255.0.0。

调试设备（windows 10 系统）IP 地址修改方式如下：

图 5 IP 调试设备 IP 地址修改

4.3 新建及打开以往工程文件

FC 软件文件为工程文件（后缀名.CCProject），内部包含所有本工程已配置所有机器人参数、程序文本、点位表单、IO 配置等信息。

4.3.1 新建工程

打开软件后，于首页点击“新建”按键。

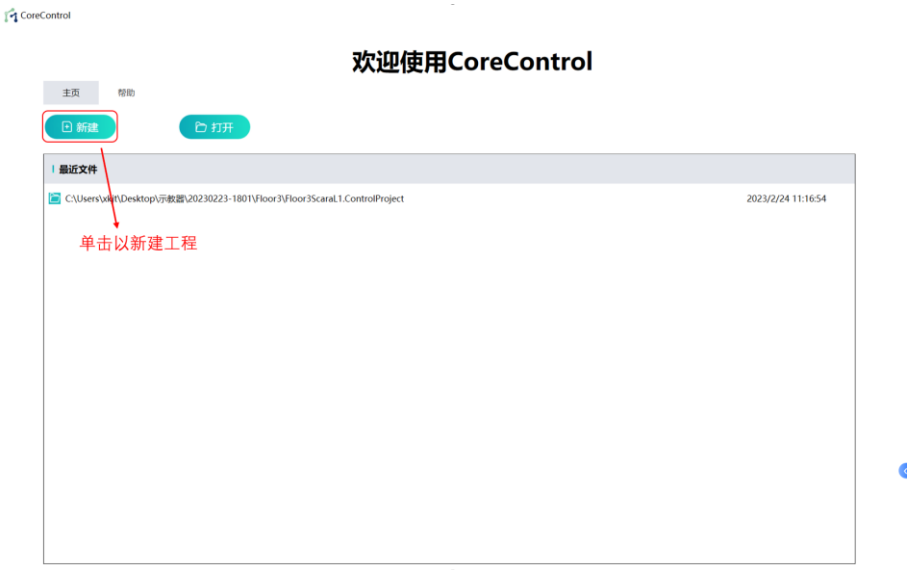
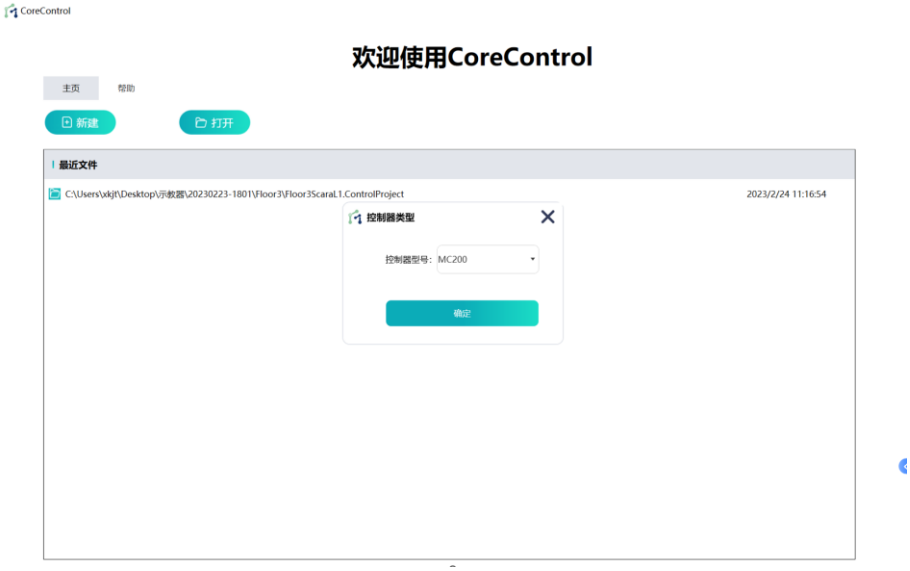


图 6 软件首页

注意

新建工程必须要选择控制柜型号(MC200、C10、C30、C40等)，控制柜的具体型号请参考收发货单，或控制柜背后的铭牌标签；



4.3.2 打开以往工程

打开以往工程方式有两种，如下所示：

1. 于进入首页点击“打开”按键，于打开文件窗口选择合适路径下工程文件。

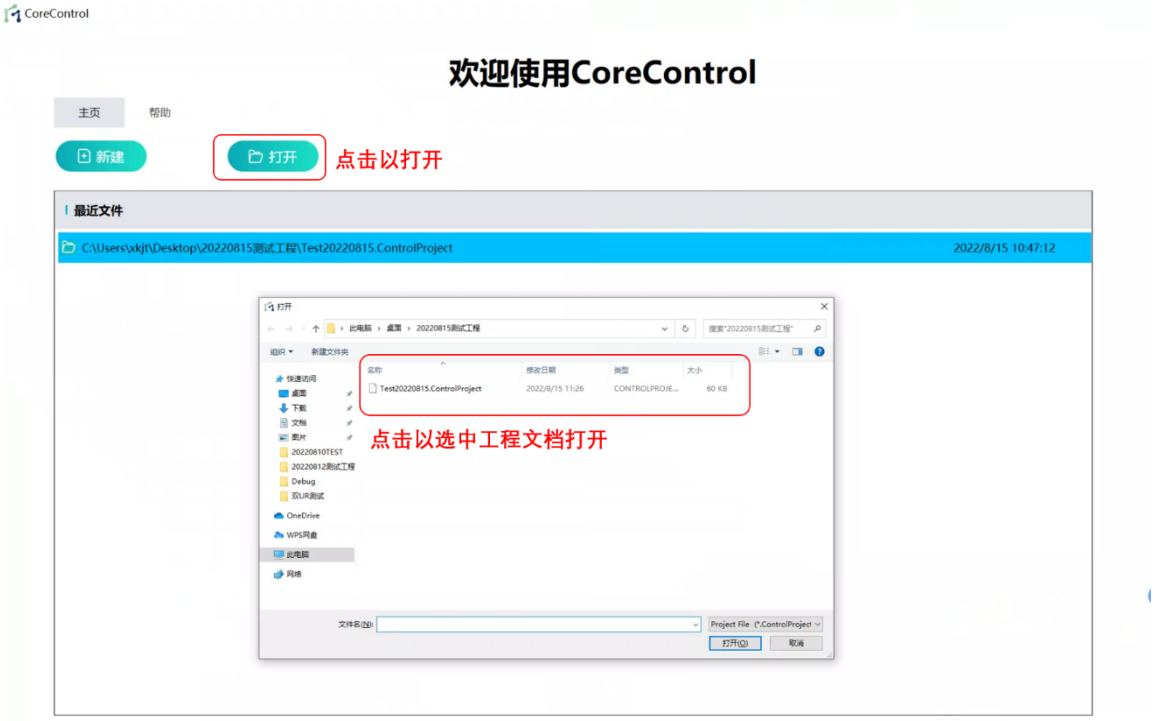


图 7 历史文件打开

2. 于进入首页点击“最近使用”窗口内所需打开工程文件，点击弹窗“确定”按键，打开指定工程文件。

图 8 历史文件打开

4.4 设备扫描添加与机械参数配置

4.4.1 控制器网络扫描与连接

调试设备连接控制柜的途径有两种：以太网、WLAN（2.4GHz wifi）。在完成 4.2 章节所述对应 IP 地址修改操作后，如使用 WLAN 方式连接，需先执行 Windows 系统连接 WLAN 操作，芯控公司控制柜激发 2.4GHz wifi 默认名称“CC10-AP-数字”，再进行以下扫描网络操作；如使用以太网方式连接，则直接进行以下扫描网络操作

工程新建或打开（工程初始化）同时默认进行控制器网络扫描工作。于“选择网络”下拉窗点击选择已扫描出控制器网络，点击“确认”进行网络连接。

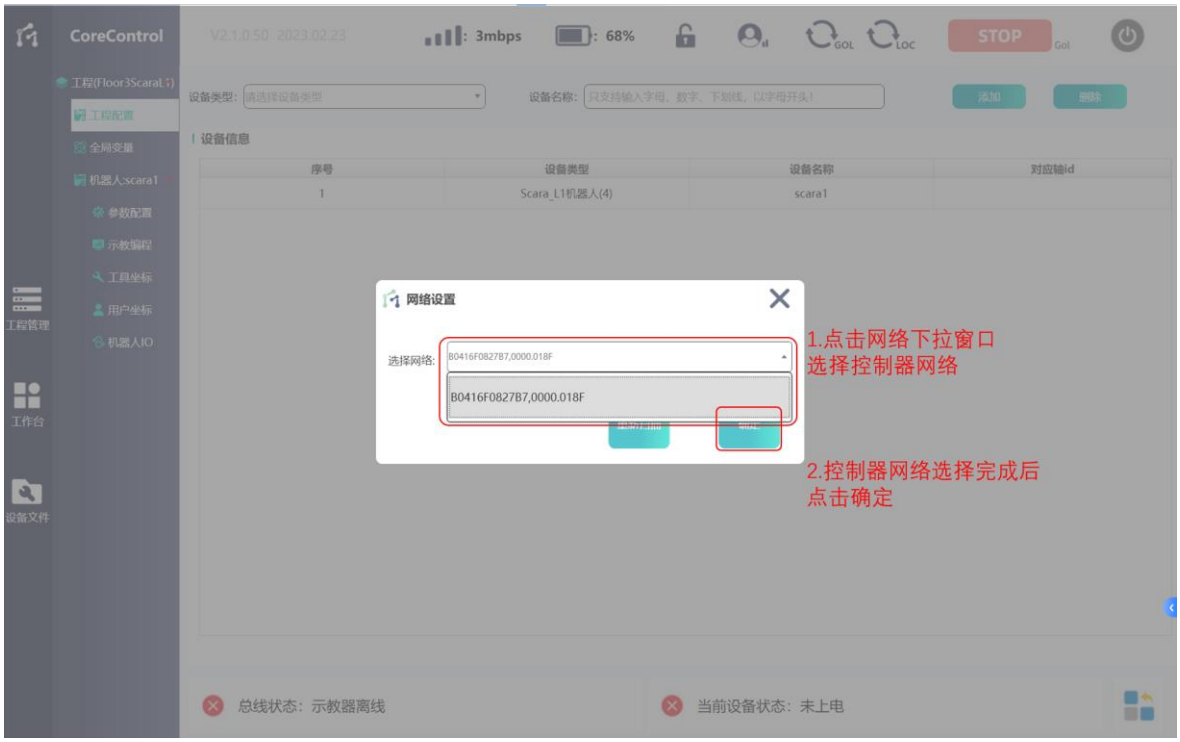


图 9 网络连接

如下拉窗内容为空，为设备网络未扫描到，点击“重新扫描”，静待按键显示内容由“扫描中”切换为“重新扫描”后，再次进行上述网络连接操作；当重新扫描时间超过 30s 仍未扫描到控制器时，示教器软件会进行提示。

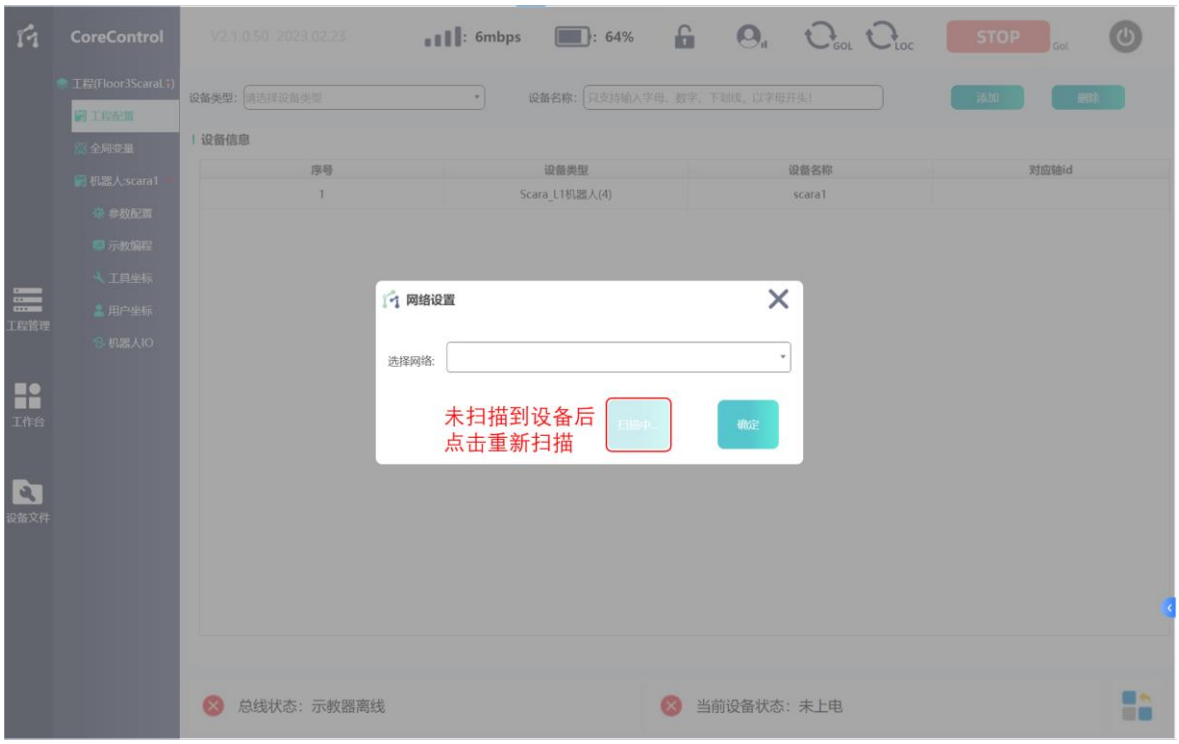
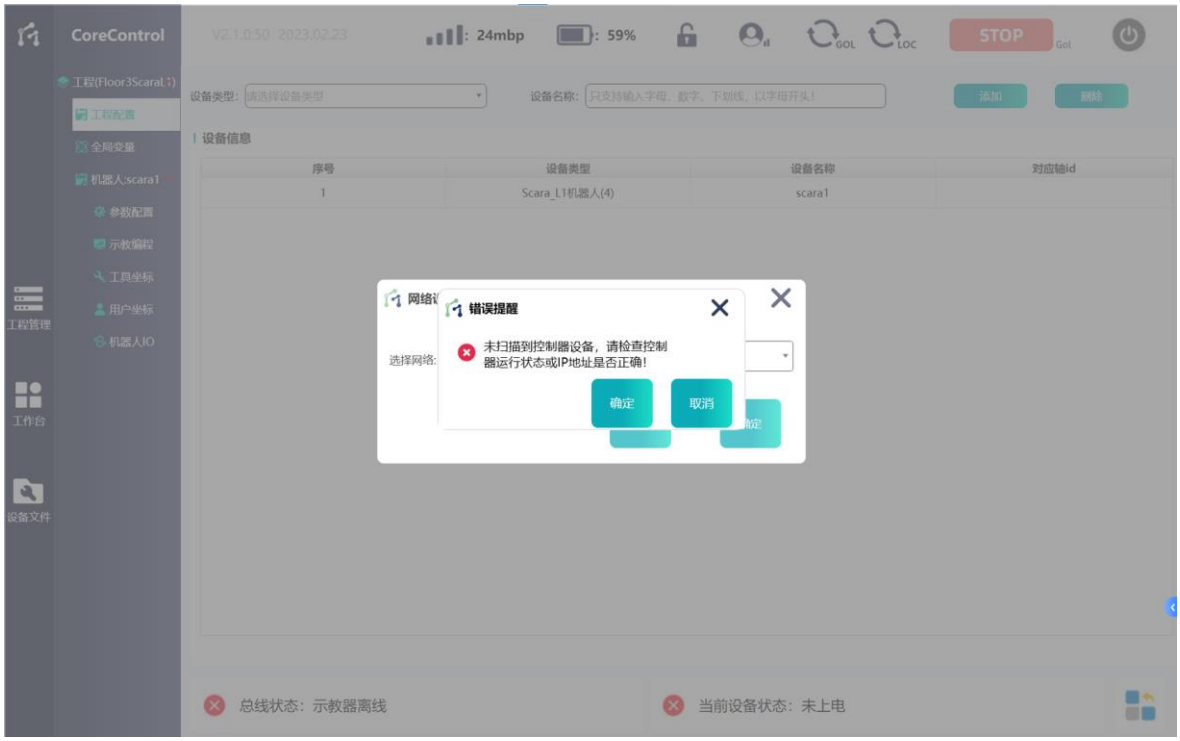


图 10 重新扫描



注意

不同控制器对应网络不同，与示教器处于同一网段的所有控制器网络均可扫描并显示。选择网络前必须确认选中控制器网络为所需调试控制器网络，以免发生误操控引发机器人误动作。

4.4.2 机器人设备添加

在工程配置页面，于“设备类型”下拉窗选择需要添加的机器人类型，于“设备名称”输入机器人名称，点击“添加”按钮即可添加机器人。机器人添加之后，需要配置机器人参数。

图 11 设备添加

4.4.2.3 机器人参数手动添加及修改

用户可点击机器人下方的“参数配置”按钮查看、配置、修改机器人的参数。

警告

机器人参数配置涉及较深机器人相关专业知识，对机器人运动状态起决定性影响，非技术人员不得随意配置、修改，且需确认相关参数与实际设备完全一致。
使用过程中，如有疑问，建议联系售后处理。

参照以下步骤，结合所提供的《机器人参数配置表》对机器人每个关节的配置参数：

1. 选择机器人参数配置页面，填写机器人参数-DH 参数、杆长、耦合比；

DH 参数、杆长：用以描述机器人连杆坐标系的参数，详见相关机器人使用手册。DH 参数为六轴机器人使用、杆长由 SCARA 机器人使用。

耦合比：因机械结构导致一轴转动时引发的其他轴发生跟随转动，偏差轴为补偿转动的单位角度即为耦合比，详细定义见相关机器人使用手册。

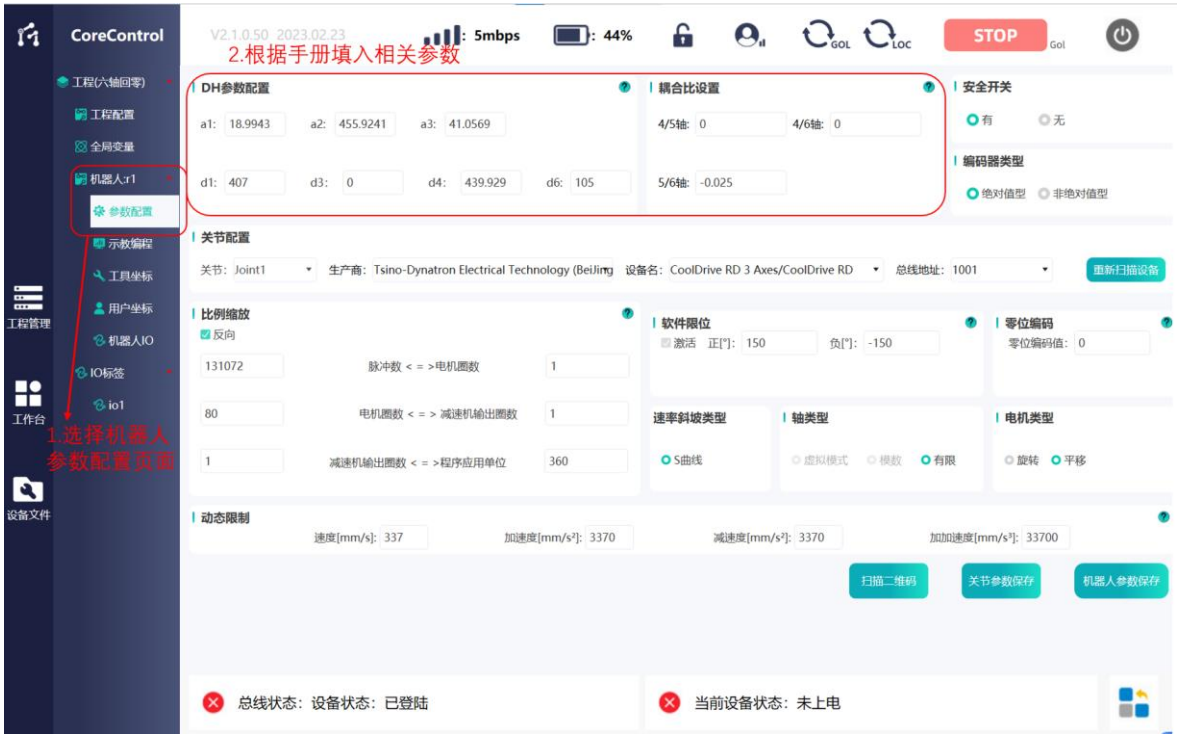


图 12 机器人本体参数配置

2. 选择、填写关节点击设备（电机）生产商、设备名、总线地址信息；

生产商：由设备本体安装驱动器对应设备描述文件决定，为驱动器厂家名称。

设备名：由设备本体安装驱动器对应设备描述文件决定，为对应控制的伺服电机型号或版本号。

总线地址：默认由 1000 位开始，EtherCAT 总线中单一通讯设备（I/O、伺服驱动器）设备即占 1 位，控制器后第 1 件设备即为 1000 位，其余依次后排。

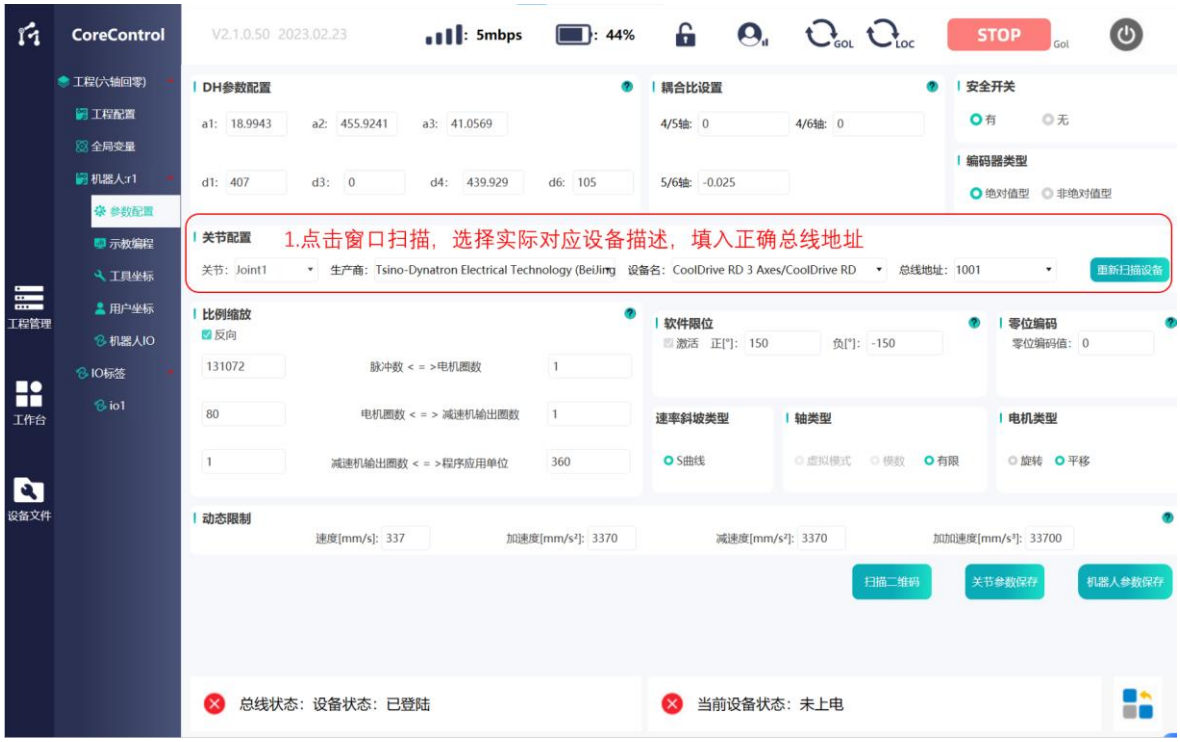


图 13 设备生产商信息配置

3. 根据《机器人参数配置表》填写对应的配置页面中的电机参数：

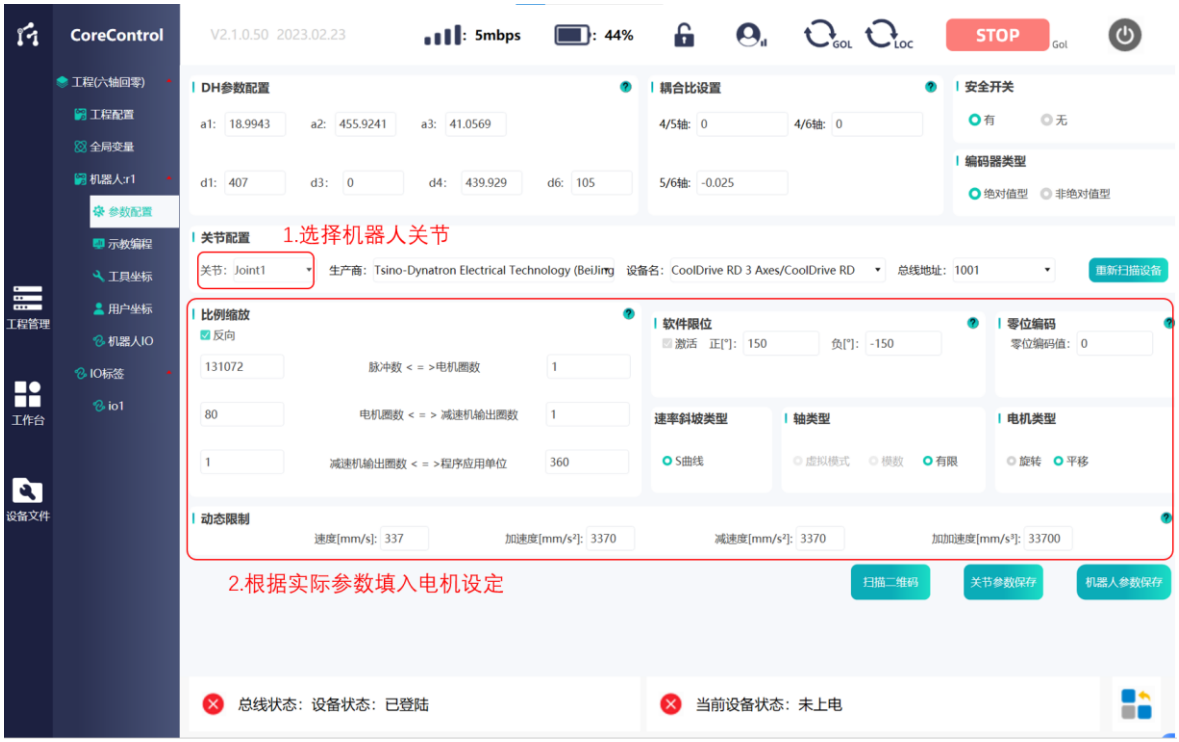


图 14 电机参数配置

4. 机器人各轴参数填写完成需点击“关节参数保存”、所有参数填写完成需点击“机器人参数保存”，完成机器人设备添加：

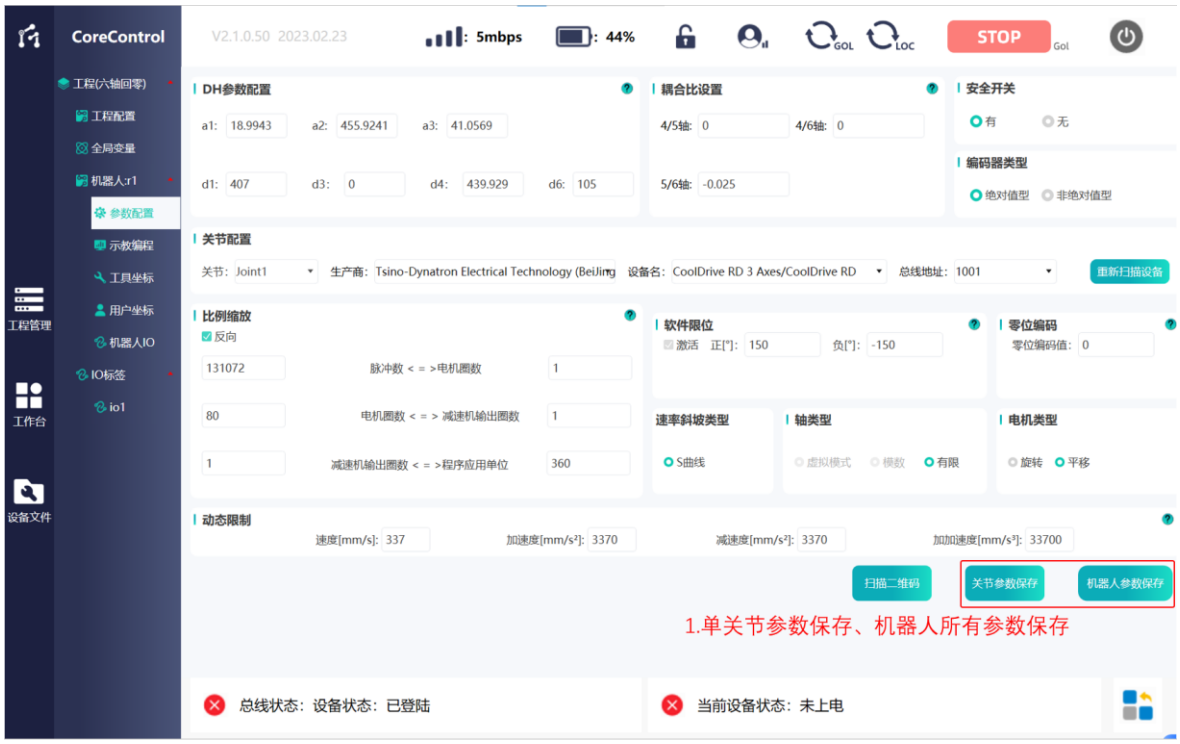


图 15 机器人参数保存

注意

所有参数新增、修改均需执行关节轴参数保存、机器人参数保存，否则无法切换至其他界面或仍按照上次保存参数运行。

若选择机器人关节电机生产商、选择驱动器具体型号和配置总线地址时，下拉菜单中没有正确的对应设备或无设备，为相关设备描述文件（XML 文件）未安装需点击状态栏“设备管理”按键，安装实机设备对应文件。

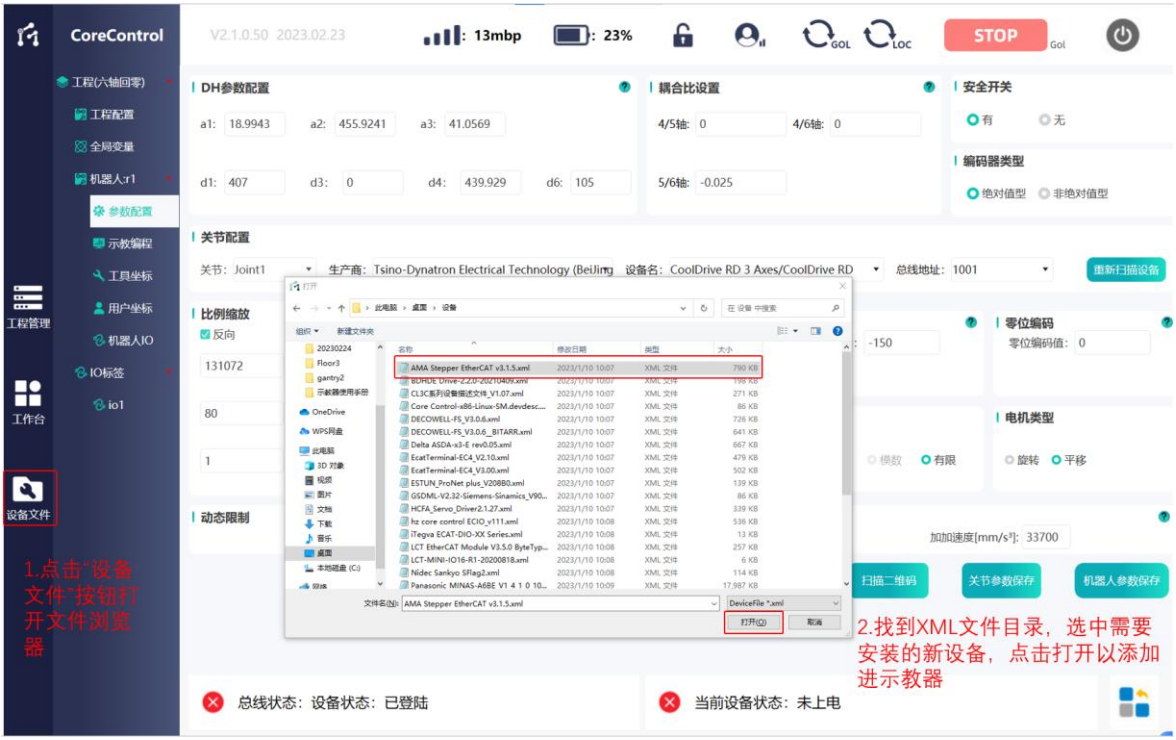


图 16 设备描述文件导入

安装完成后点击“重新扫描设备按钮”，等待控制器执行一遍设备扫描，扫描完成后再观察有无对应设备以进行参数配置工作。

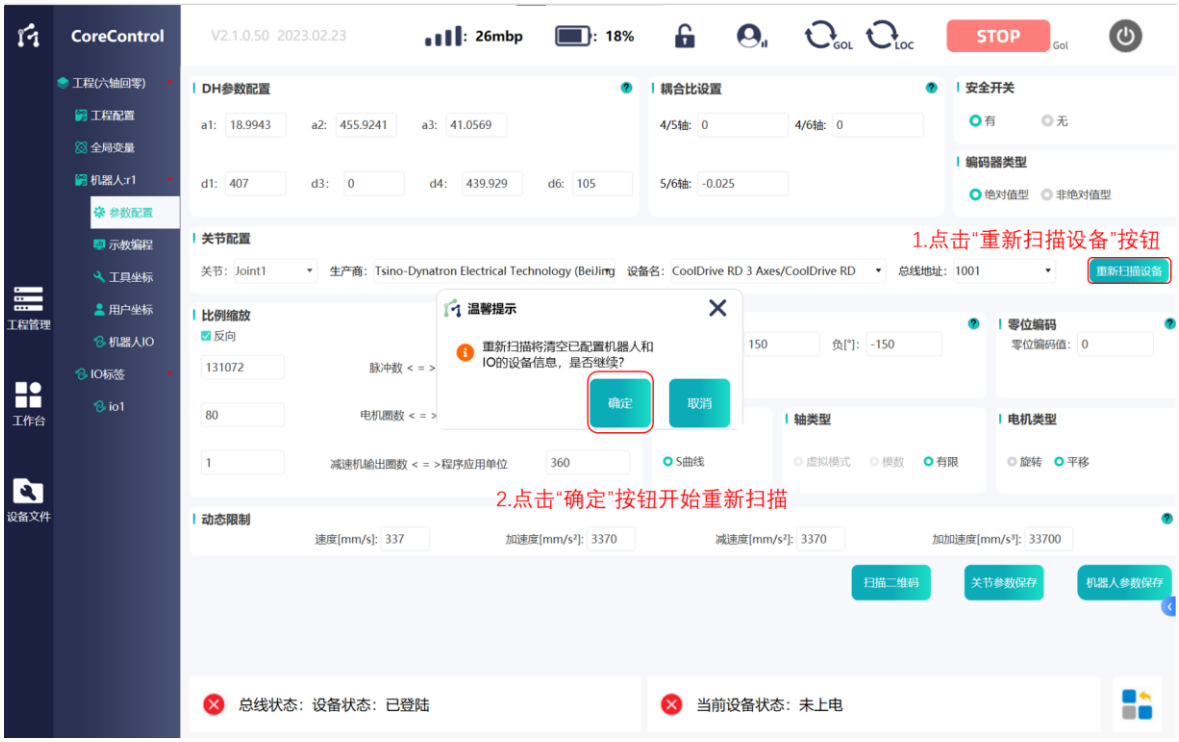


图 17 EtherCAT 总线设备地址扫描

注意

参数配置项总线地址必须与实际设备于EtherCAT总线中连接顺序一致。已配置工程对应设备发生接入顺序变动、设备新增等需执行总线地址重新扫描配合操作。

4.4.2.4 机器人参数扫码添加及修改

芯控公司出厂机器人设备本体自带参数配置二维码，用户可以于参数配置页面点击“扫描二维码”按钮，通过示教器自带相机拍摄自动解析填写机器人参数。

但是，机器人关节电机生产商，驱动器型号和总线地址仍需要根据实际总线情况进行填写。参考 4.4.2.3 机器人参数手动添加及修改章节内容第 2 步填写并保存。

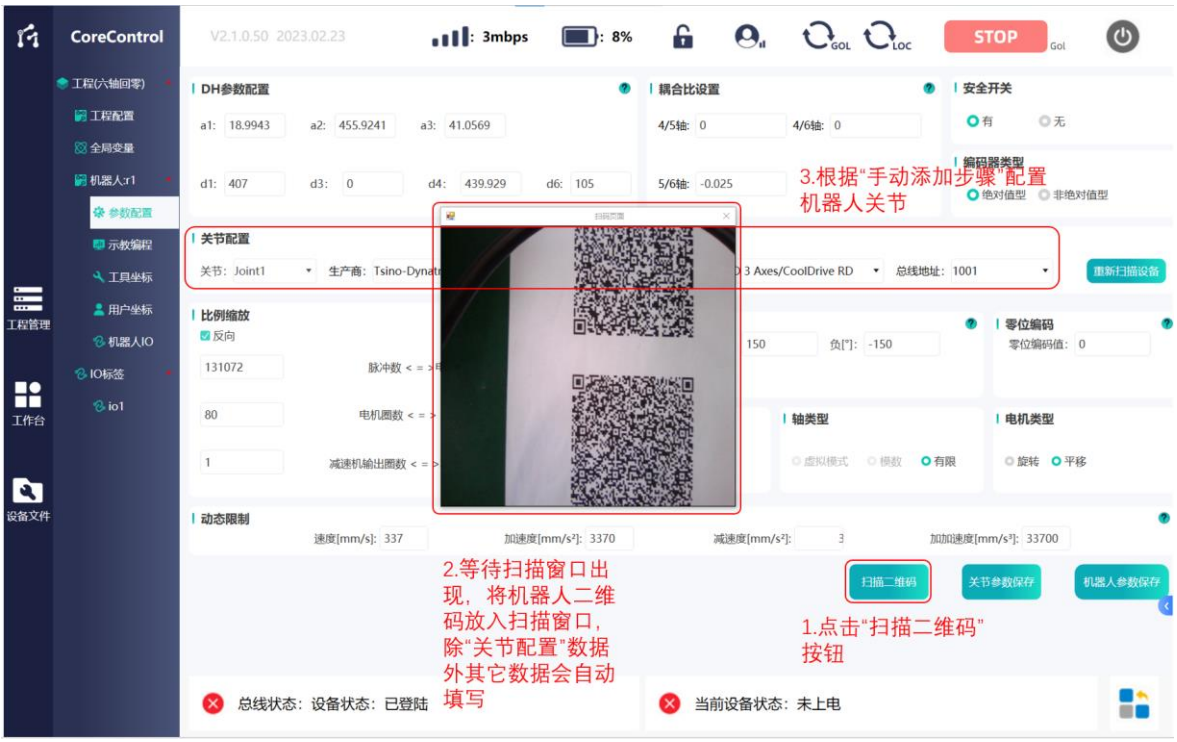


图 18 二维码扫描填入参数

4.4.3 I/O 设备添加及 I/O 变量关联

4.4.3.1 I/O-主控与 I/O-通用

FC 软件包含了两种 I/O 设备，一种是 I/O-主控，另一种是 I/O-通用。

I/O-主控是 C10、C30、C40 型号控制柜上固定的 IO 模块，新建机器人工程时必须添加；而 I/O-通用则可以根据实际情况按需添加。

I/O 主控可以设置为系统 I/O 或机器人 I/O，名称为“DefinedSystemIO”不可更改，设备厂家、设备类型和总线地址默认且不能修改，其中输入变量关联了 systemstop 和 dedaman 输入，此输入与示教器的夹持器上的急停和 deadman 按钮相关联；

4.4.3.2 I/O 设备添加及变量定义

添加 I/O-主控或 I/O-通用模块进行控制时，可以选择添加在全局还是机器人设备下。无论是全局 I/O 还是机器人 I/O 都可以在不同机器人程序中被任意调用。

机器人 I/O 可以添加至机器人示教界面的快捷动作中，详见 4.8.3I/O 快捷输入输出章节内容，方便用户快速执行 I/O 动作，全局 I/O 则没有此操作。

I/O 设备添加通过“工程配置”进行，其配置流程与机器人配置基本相同：

图 19 I/O-主控设备添加

图 20 I/O-通用设备添加

于需配置详细 I/O 参数标签下打开 I/O 界面，参照以下步骤进行 I/O 详细配置及变量定义：

1. 点击进入相关 I/O 配置界面，进行设备信息登录，生产商、设备名、总线地址相关定义及要求与机器人轴电机设备信息一致。

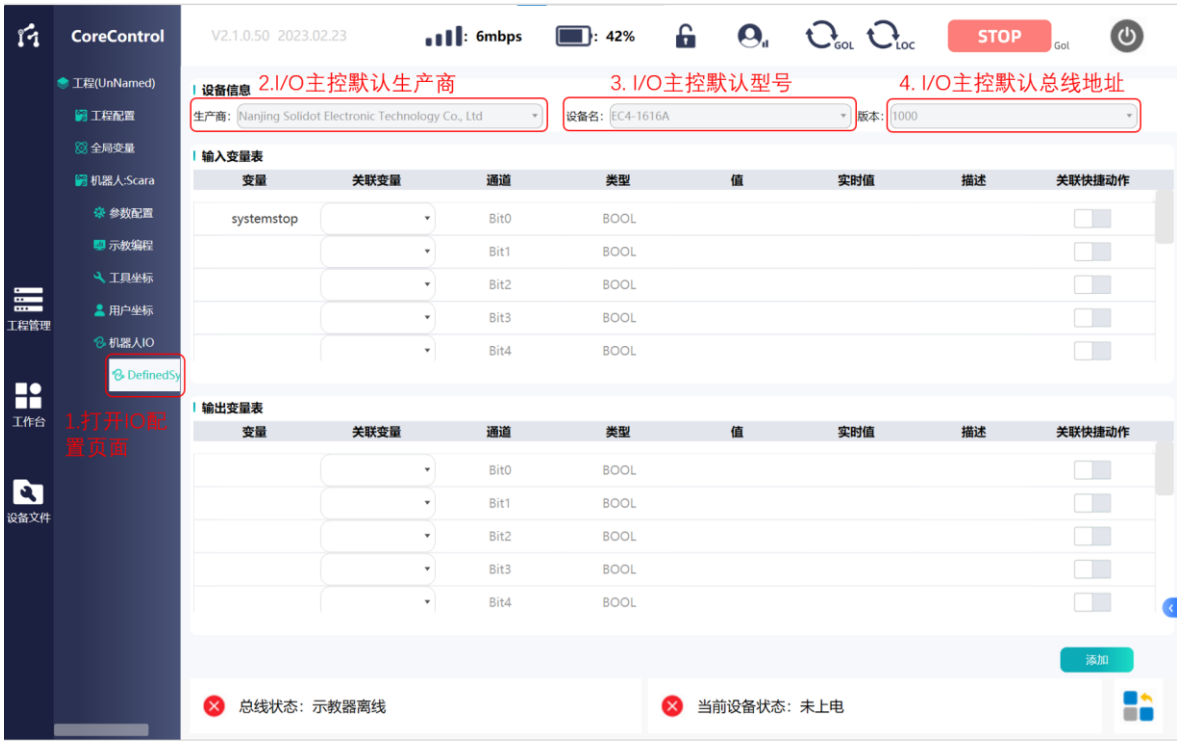


图 20 I/O-主控默认设置

图 21 I/O 设备生产商信息选择

2. 于需使用的 I/O 点位填入程序引用变量名、初始值（可为空）、描述（注释，可中文），选择关联快捷动作，并确认添加。

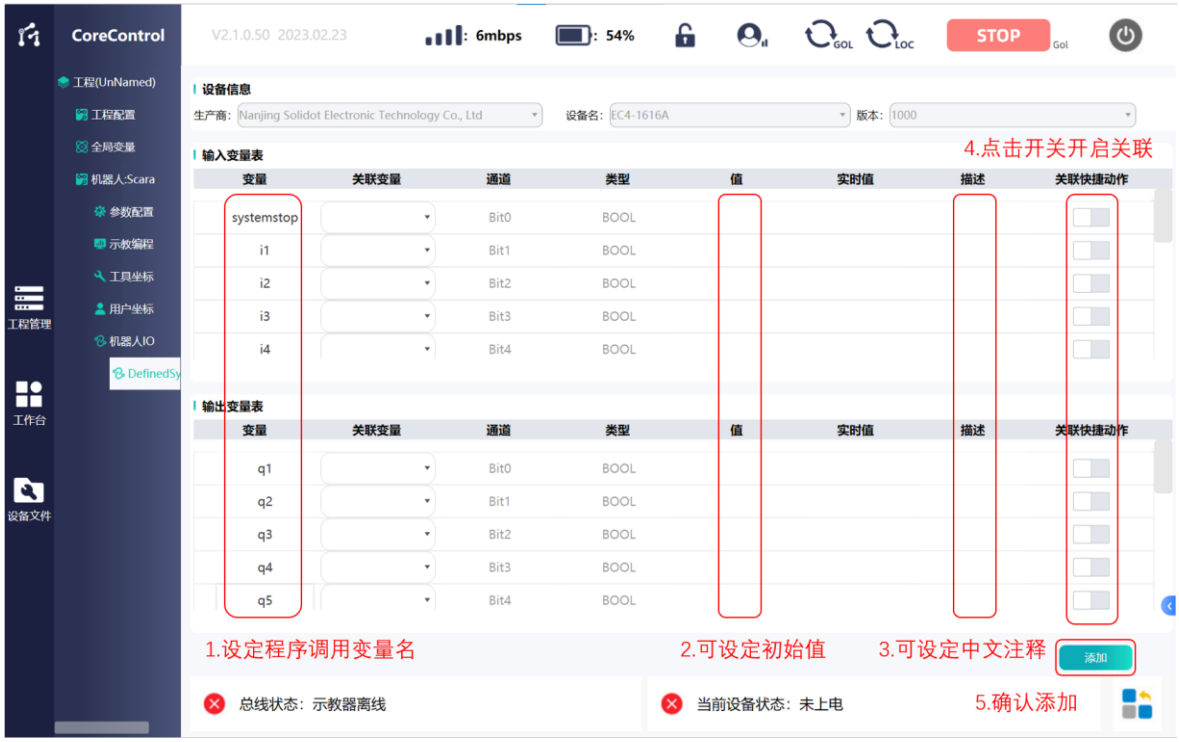


图 22 I/O 变量声明

3. I/O 设备实时值监测

图 23 I/O 变量实时值监测显示

注意

I/O配置内定义变量需注意与程序内变量一致，否则会导致程序编译失败。

4.4.3.3 I/O 关联系统变量

FC 软件支持 I/O 关联系统内部变量，以实现外部按键控制机器人使能、停止、暂停、启动、复位五项功能。

机器人 I/O 添加时，于任意点位对应“关联变量”项选择需绑定变量内容：空（即无关联）、ROBOT.POWER-机器人使能、ROBOT.STOP-机器人停止、ROBOT.SUSPEND-机器人暂停、ROBOT.RUN-机器人运行、ROBOT.RESET-机器人复位。

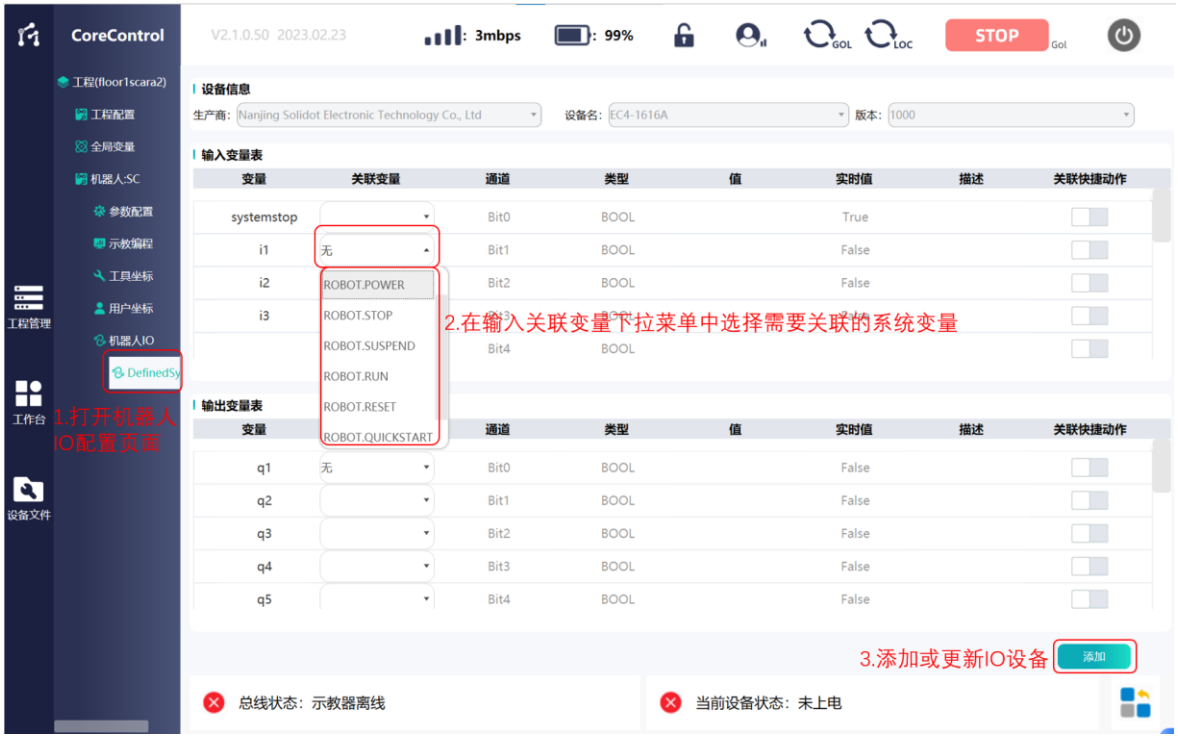


图 24 I/O 关联系统内部变量

注意

目前仅上述5个系统变量可供关联，变量不可重复关联。
每台机器人设备对应的系统变量不同，每台设备下系统变量仅支持控制当前设备，可通过I/O串联的方式实现同时触发。

4.4.4 全局变量定义及添加

FC 软件支持多台机器人程序通过全局变量实现触发时机、触发条件、位置锁定等非算法层面协同，故示教器所提供全局变量可供所有机器人使用和修改。与局部变量相区分。

FC 软件所支持的变量数据类型有：BIT、BOOL、BYTE、INT、LINT、LREAL、REAL、STRING、STRINGLIST；数据的取值范围和具体作用参考 5.1 局部变量定义章节内容。

添加全局变量时，需要打开全局变量页面，输入变量名并选择需要的数据类型，设置对应的初始值后点击“确定”按钮，变量即被添加入全局变量中。

若添加的变量为“BOOL”类型，还可以关联至快捷动作中，具体参考“4.8.4 全局变量输入输出”。

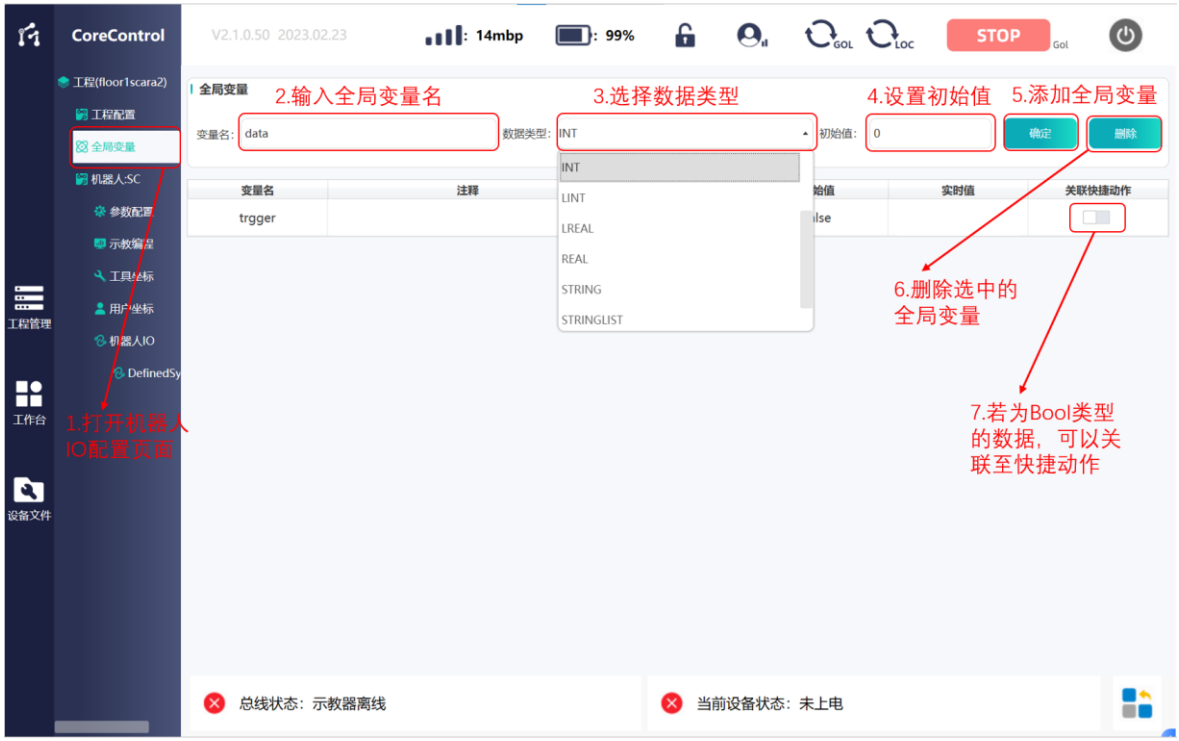


图 25 全局变量声明添加

4.5 机器人使能、启动及关闭

确认所有人都离开机器人工作区域或臂展可达范围，所有安全装置在适当位置并正常工作中，遵循以下方法对机器人马达进行上电（使能）及解抱闸操作。

4.5.1 设备登录

设备登录不复位机器人轴状态，登录后开始自动创建 EtherCAT 通讯，并提示登录成功。

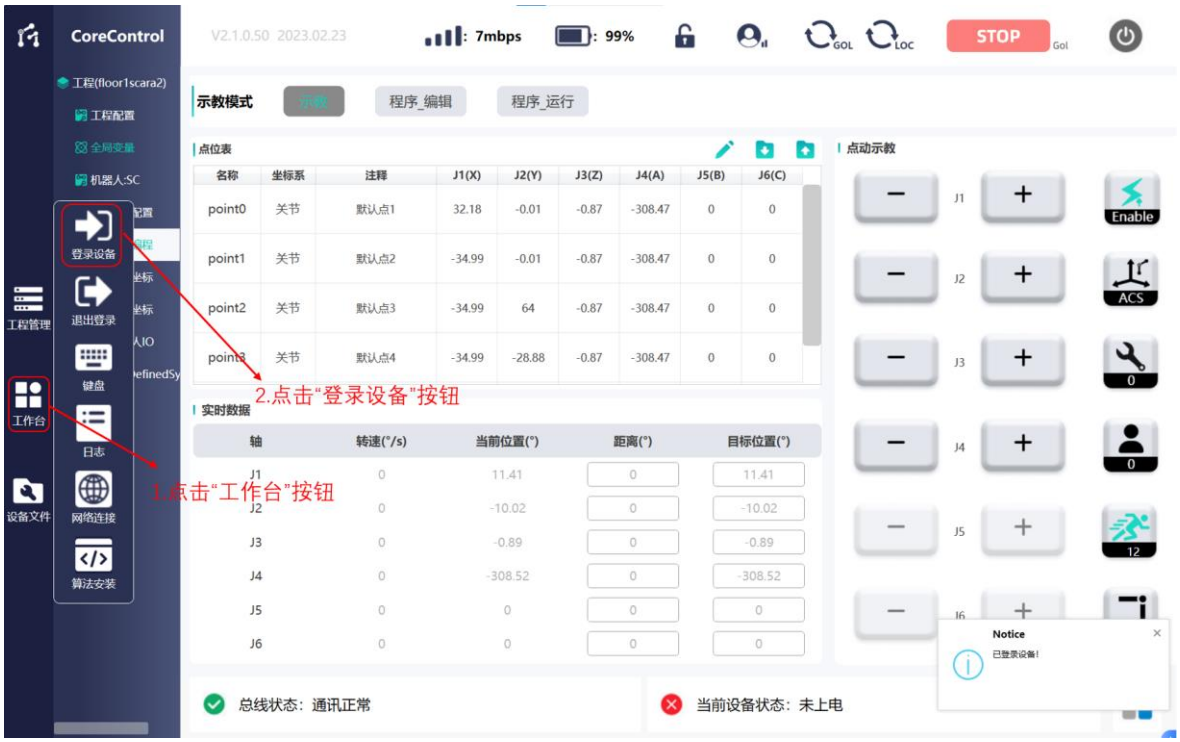


图 26 工程登录

部分工程因下挂管理的设备较多，存在登录设备后，EtherCAT 通讯未能同时结束创建，可等待 1-5S 完成通讯后进行操作。

如设备示教器此前已经登录后发生网络连接中断或运行过程中关闭示教器软件，可直接重新登录连接设备，以同步程序运行状态、轴状态、I/O 状态等。

4.5.2 电机使能

设备登录、EtherCAT 通讯建立后，手动示教、程序运行前需通过“使能”按键给电机使能、上电动作。

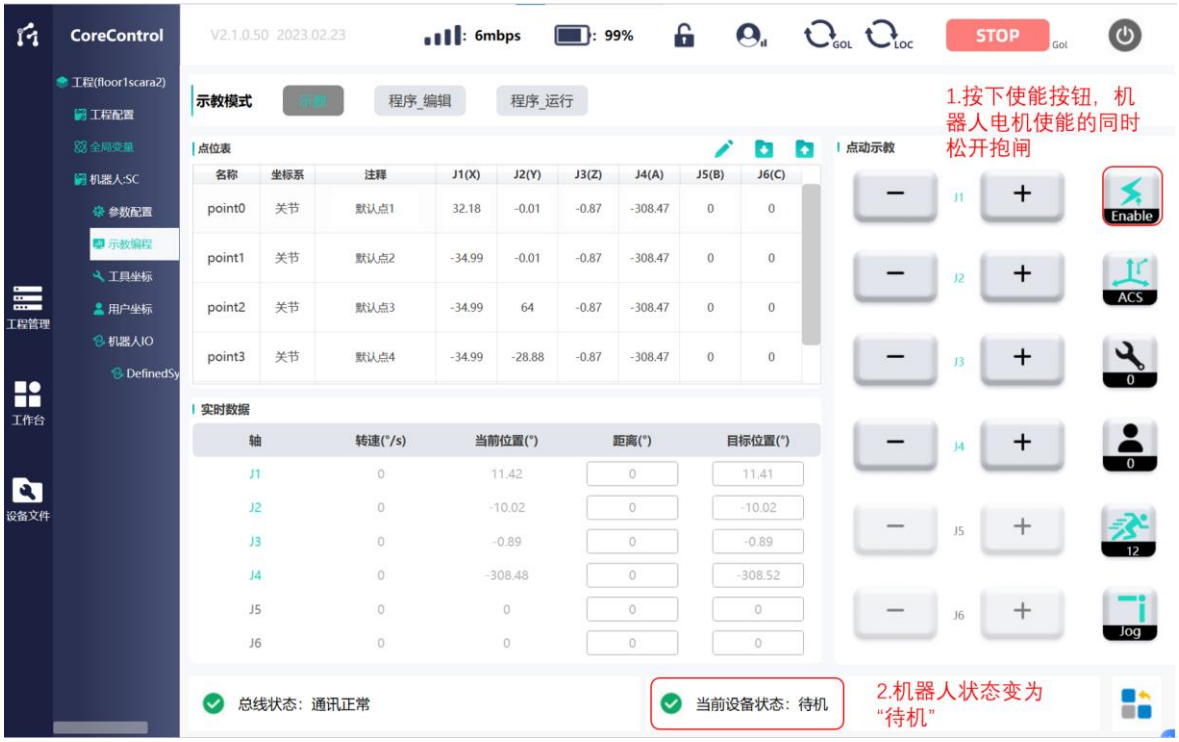


图 27 电机使能

4.5.3 电机刹车释放

电机刹车释放（解抱闸）分为两种情况：

1. 芯控自研机器人，电机使能同时自动解除抱闸，释放刹车；

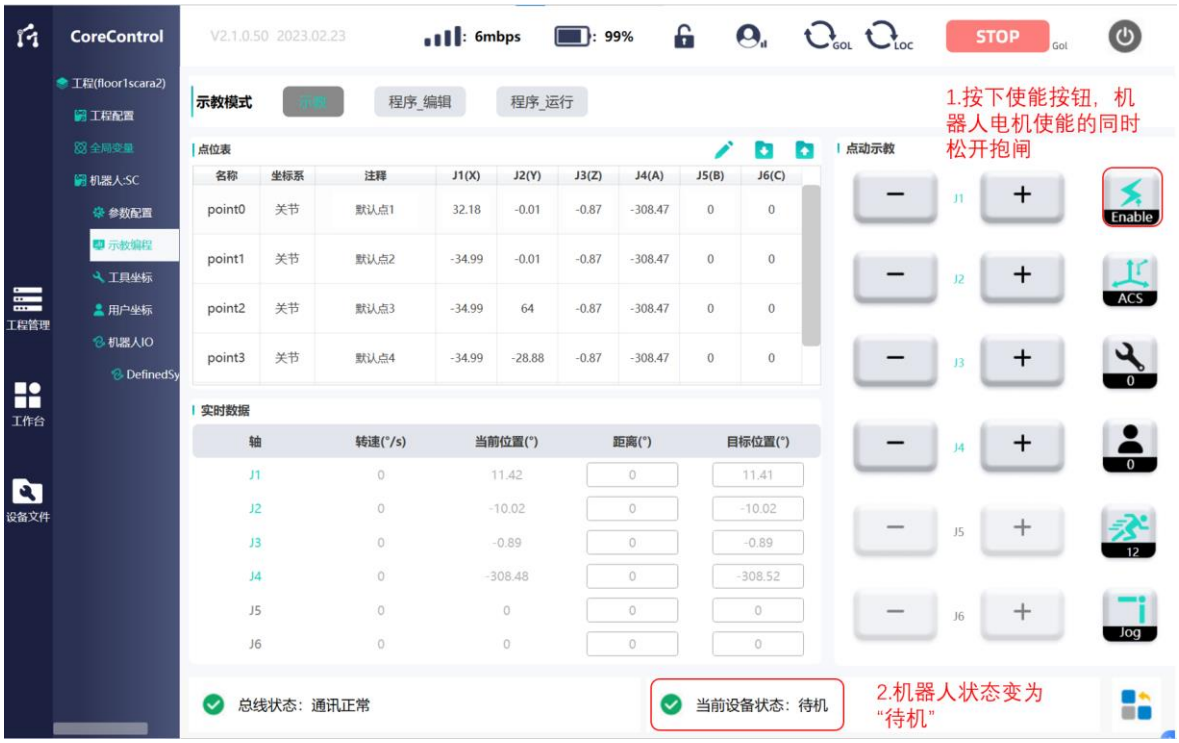


图 28 电机刹车释放

2. 部分其他品牌机器人电机刹车释放需要增加 PDO 通讯，使用独立按键释放，

如使用 FC 软件及控制器作为机器人控制器控制该类机器人本体，需调整 PDO 通讯内容，并添加快捷输入输出按键作为解抱闸按键。

此功能 FC 软件暂未开放，如有需要请联系芯控售后人员技术支持。

4.5.4 设备退出登录

设备退出登录复位机器人轴所有状态，本工程控制下的所有机器人停止动作，电机马达下电，同时抱闸自动吸合。机器人及电机动作与“急停”按键触发相同。

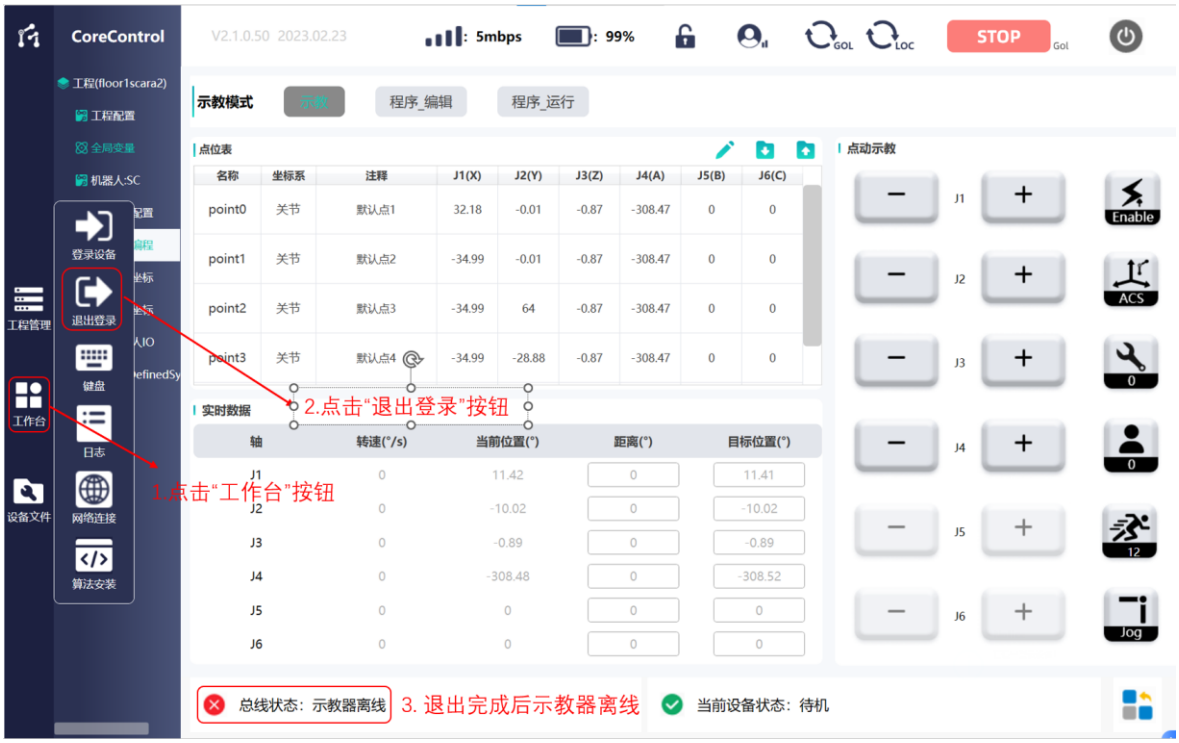


图 29 工程退出登录

4.6 机器人坐标系标定及手动示教

机器人坐标系标定结果受各电机参数（正反向设定）、轴零位标定共同影响，最终建系结果满足**笛卡尔右手定则**。电机参数设定参照 4.4.2 机器人设备添加章节内容，轴零位标定参照 4.6.2 关节零位、关节限位标定章节内容。

FC 软件支持常见机器人的四种坐标系：关节坐标系-ACS、笛卡尔坐标系-MCS，工具坐标系-TCS、用户坐标系-PCS。于手动示教界面点击“坐标系”切换按键，弹窗选择切换。

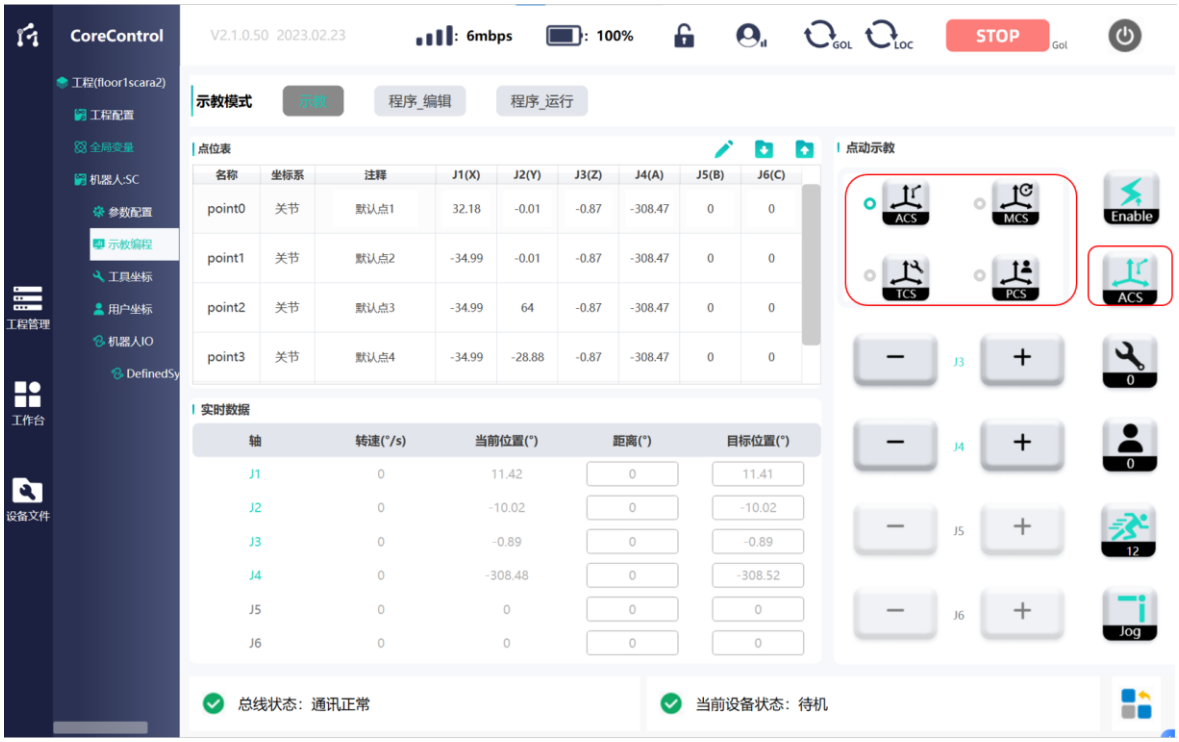


图 30 坐标系切换

注意

手动示教、程序运行均须于控制器处于设备登录状态、EtherCAT通讯状态状态正常情况下才能执行相应操作

注意

除关节坐标系（ACS）外，所有坐标系（MCS、TCS、PCS）如上述诸图所示正常无偏差均须保证各轴电机参数设定无误、关节零位设定无误。
轴电机参数设定详见4.4.2章节内容。
关节零位设定详见4.8.1章节内容

4.6.1 常见机器人坐标系图解及相应点动

下述内容为四大坐标系图解、不同类型机器人所支持坐标差异，以及选中坐标系下相应“单轴点动”示教效果。

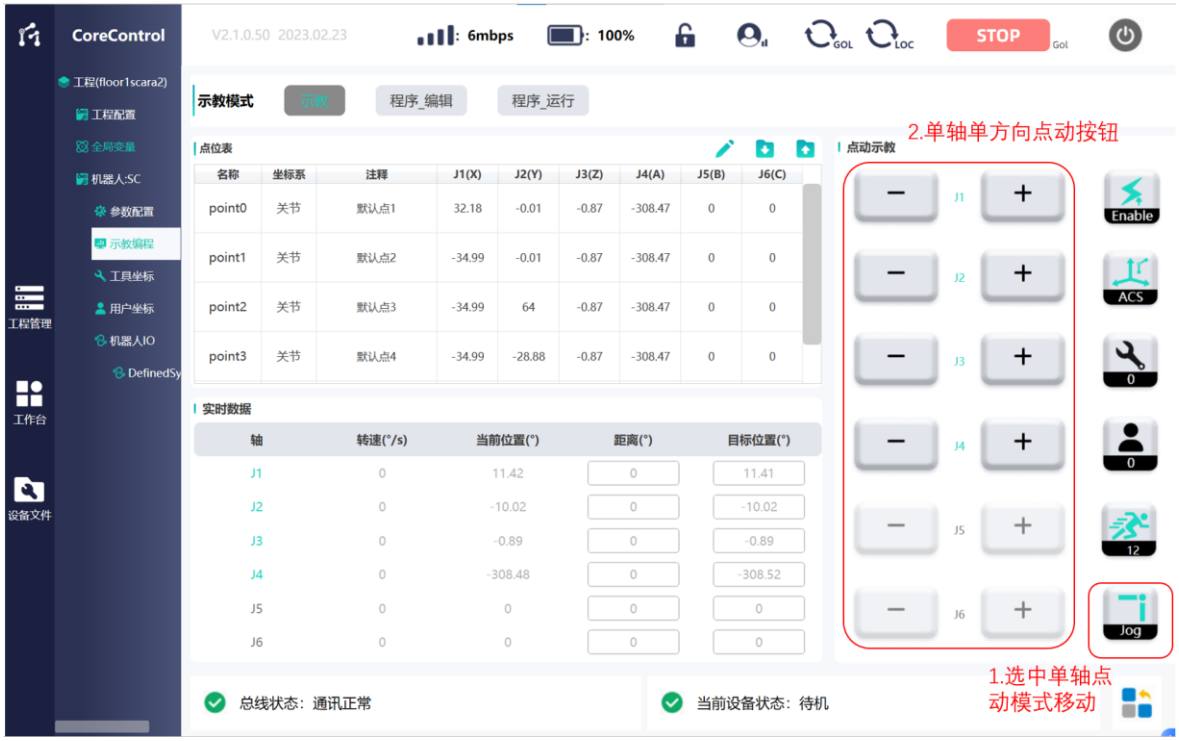


图 31 单轴点动

4.6.1.1 六轴机器人坐标系及相应点动

关节坐标系-ACS：选中该坐标系情况下，可单独进行机器人轴关节绕轴旋转。

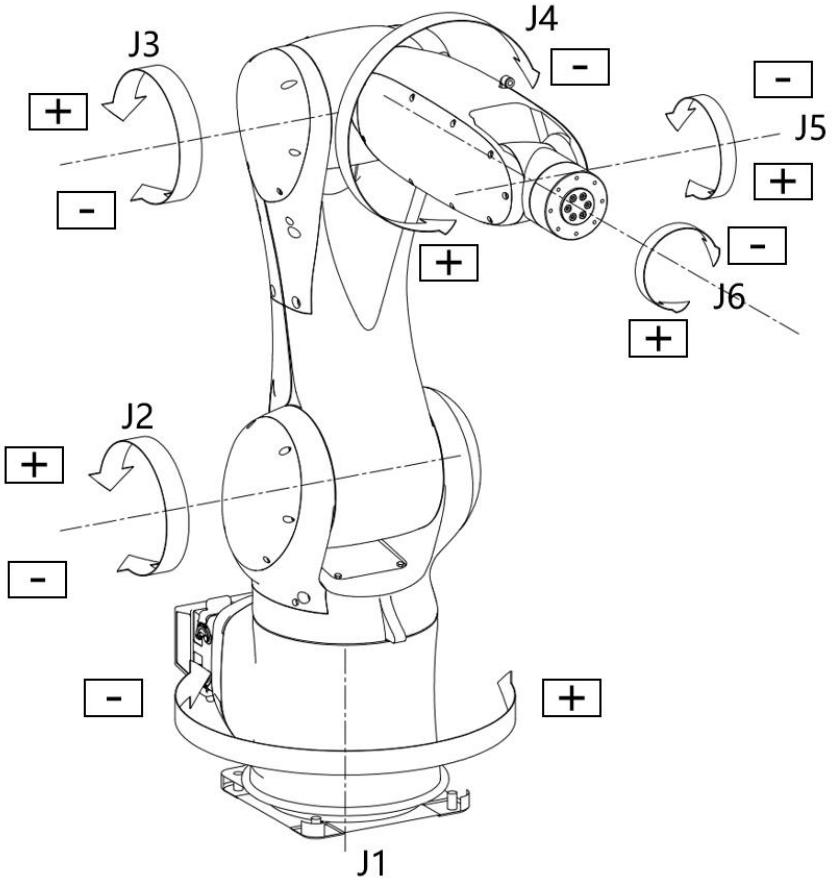


图 32 六轴机器人关节坐标系

笛卡尔坐标系-MCS：选中该坐标系情况下，可单独进行机器人沿固定直角坐标系平移或绕坐标系轴旋转。坐标系的变换值 X、Y、Z、A、B、C 均为 0。

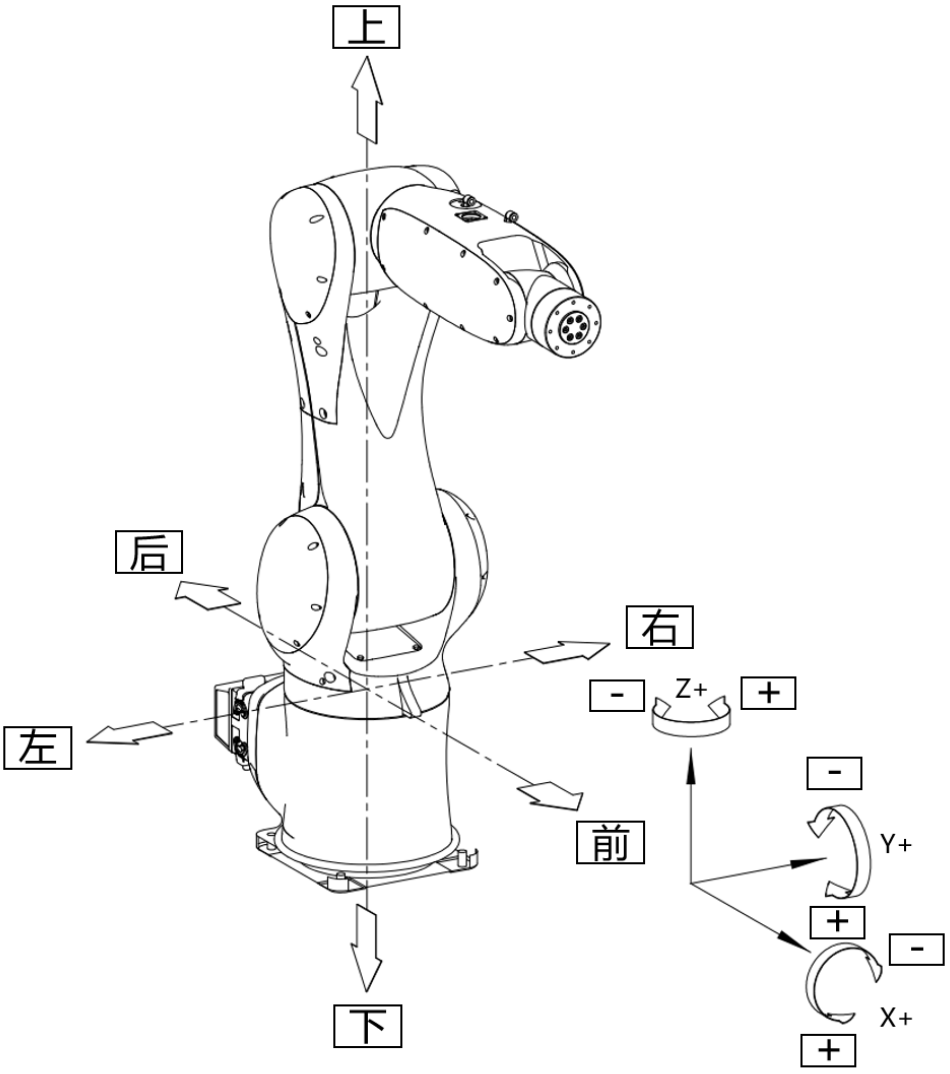


图 33 六轴机器人笛卡尔坐标系

工具坐标系-TCS：选中该坐标系情况下，可单独进行机器人沿工具坐标系轴平移或绕坐标系轴旋转。工具坐标系随机器人位姿变化而变化，且随登录选中的工具变换而变换。

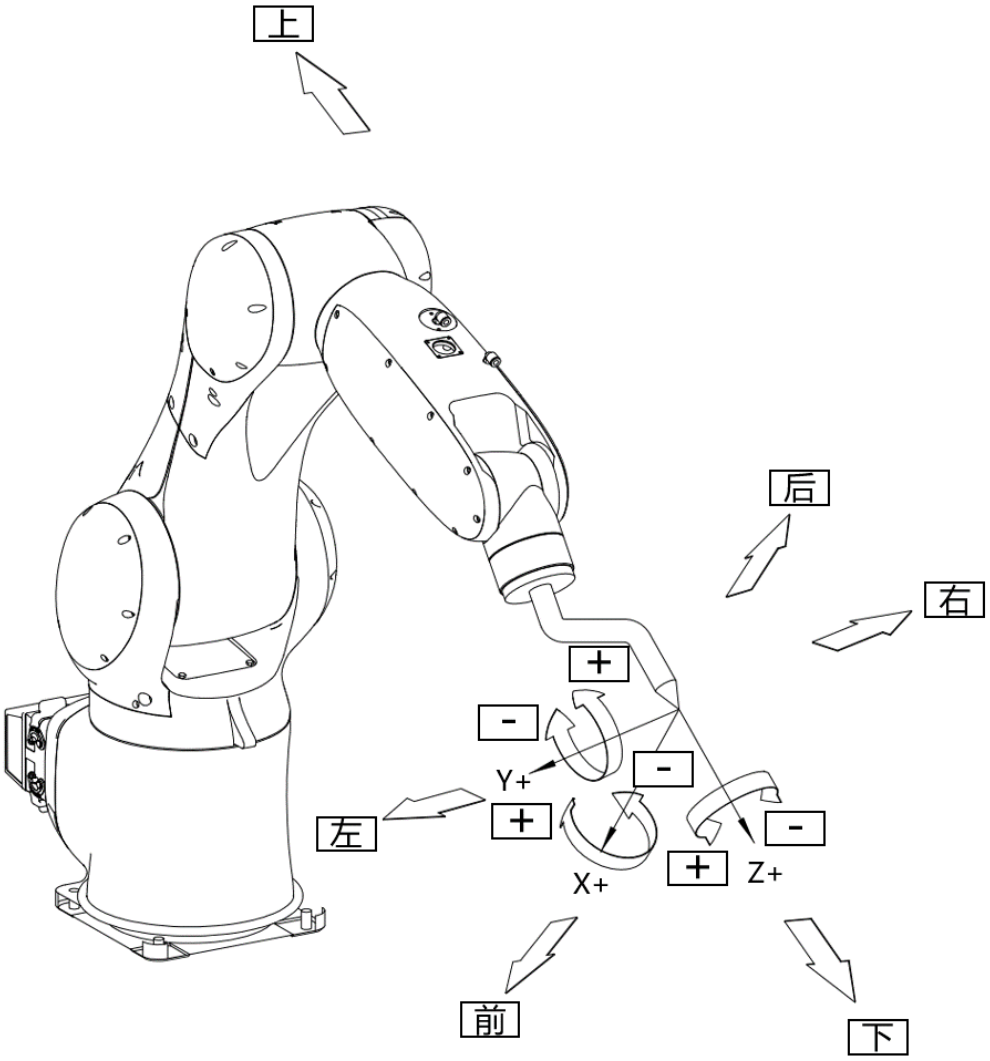


图 34 六轴机器人工具坐标系

用户坐标系-PCS：选中该坐标系情况下，可单独进行机器人沿用户坐标系轴平移或绕坐标系轴旋转。坐标原点及系轴方向随登录选中的用户坐标 变换而变换。详见章节 4.7 内容。

4.6.1.2 SCARA 机器人坐标系及相应点动

FC 软件支持多种 SCARA 机器人，相关图示如下：

SCARA-L3 型：

关节坐标系-ACS：选中该坐标系情况下，可单独进行机器人轴关节绕轴旋转。

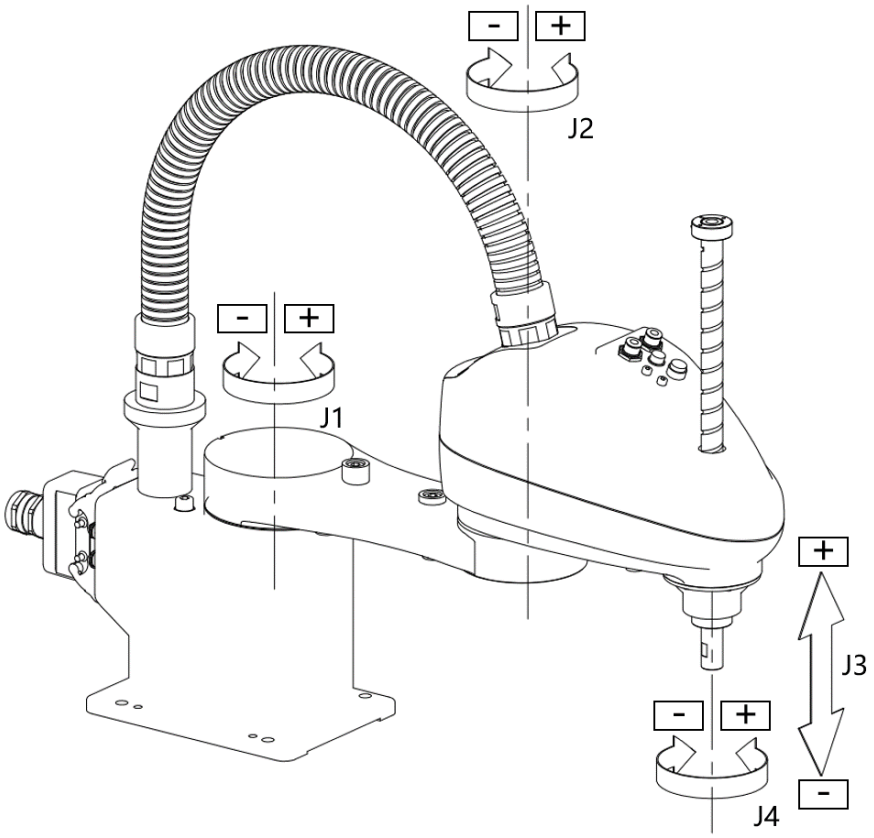


图 35 SCARA-L3 型机器人关节坐标系

笛卡尔坐标系-MCS：选中该坐标系情况下，可单独进行机器人沿固定直角坐标系平移或绕坐标系轴旋转。坐标系的变换值 X、Y、Z、A、B、C 均为 0。

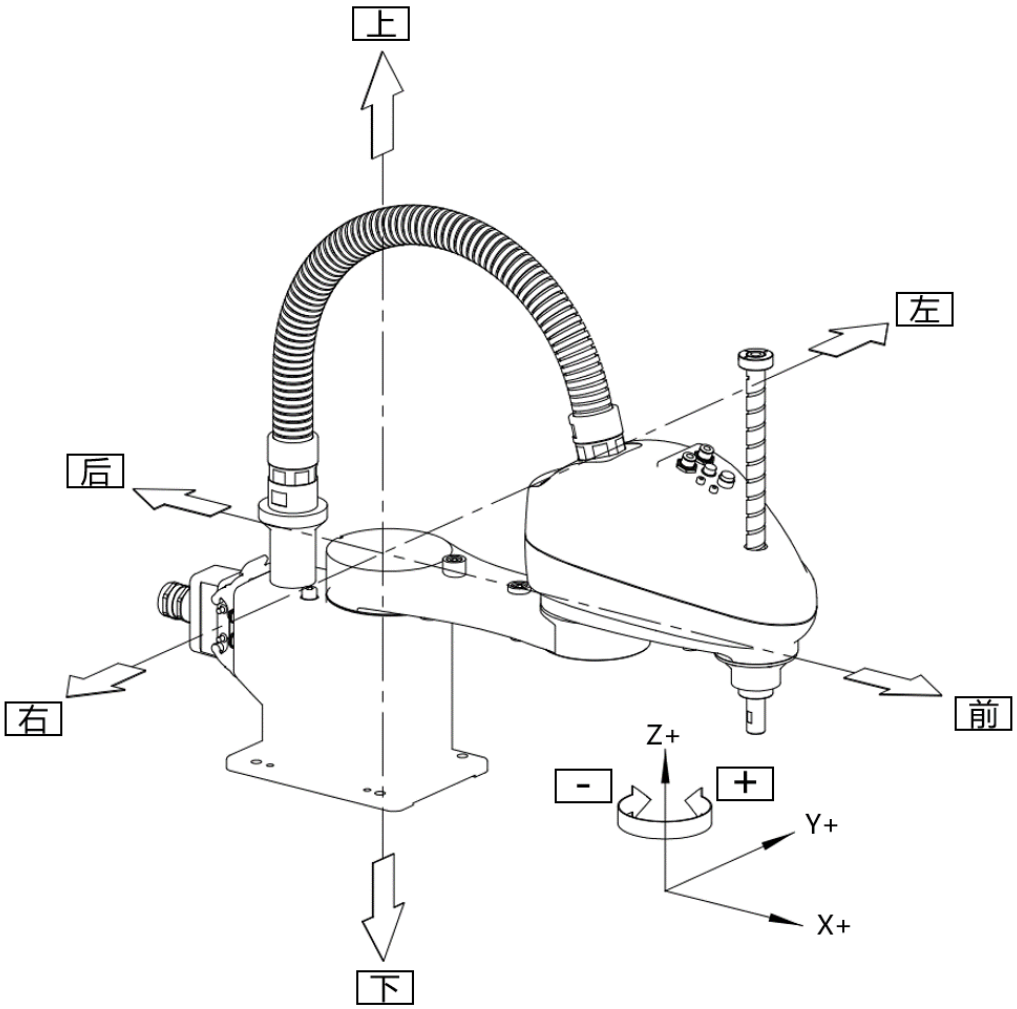


图 36 SCARA-L3 型机器人笛卡尔坐标系

工具坐标-TCS：选中该坐标系情况下，可单独进行机器人沿工具坐标系轴平移或绕坐标系轴旋转。工具坐标系随机器人位姿变化而变化，且随登录选中的工具变换而变换。

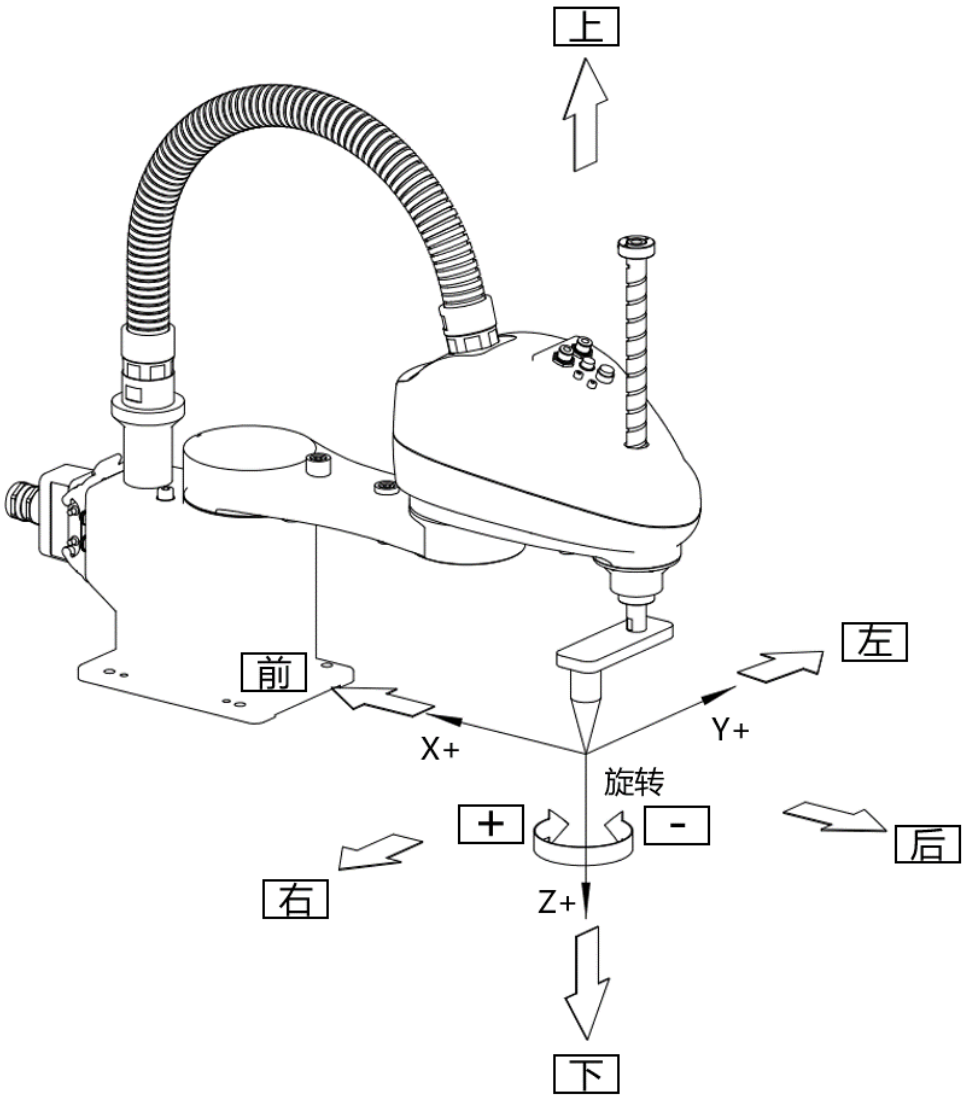


图 37 SCARA-L3 型机器人工具坐标系

用户坐标系-PCS：选中该坐标系情况下，可单独进行机器人沿用户坐标系轴平移或绕坐标系轴旋转。坐标原点及系轴方向随登录选中的用户坐标 变换而变换。详见章节 4.7 内容。

SCARA-L2 型

关节坐标系-ACS：选中该坐标系情况下，可单独进行机器人轴关节绕轴旋转。

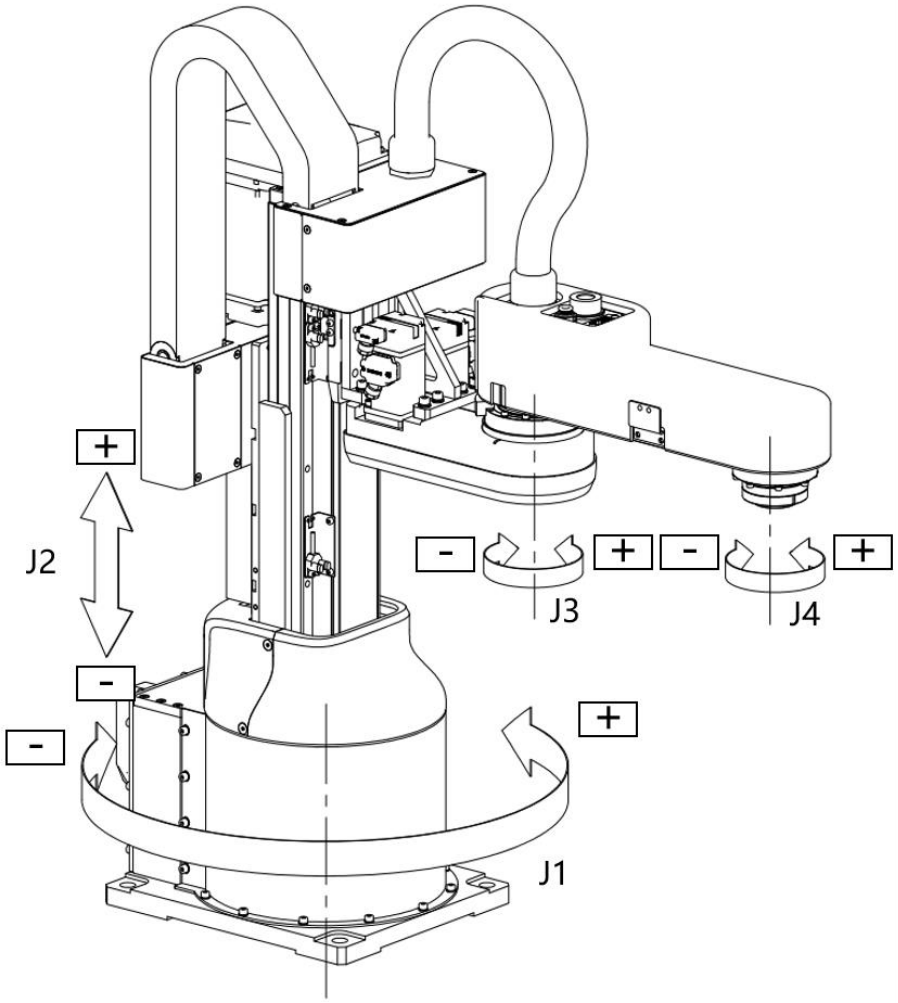


图 38 SCARA-L2 型机器人关节坐标系

笛卡尔坐标系-MCS：选中该坐标系情况下，可单独进行机器人沿固定直角坐标系平移或绕坐标系轴旋转。坐标系的变换值 X、Y、Z、A、B、C 均为 0。

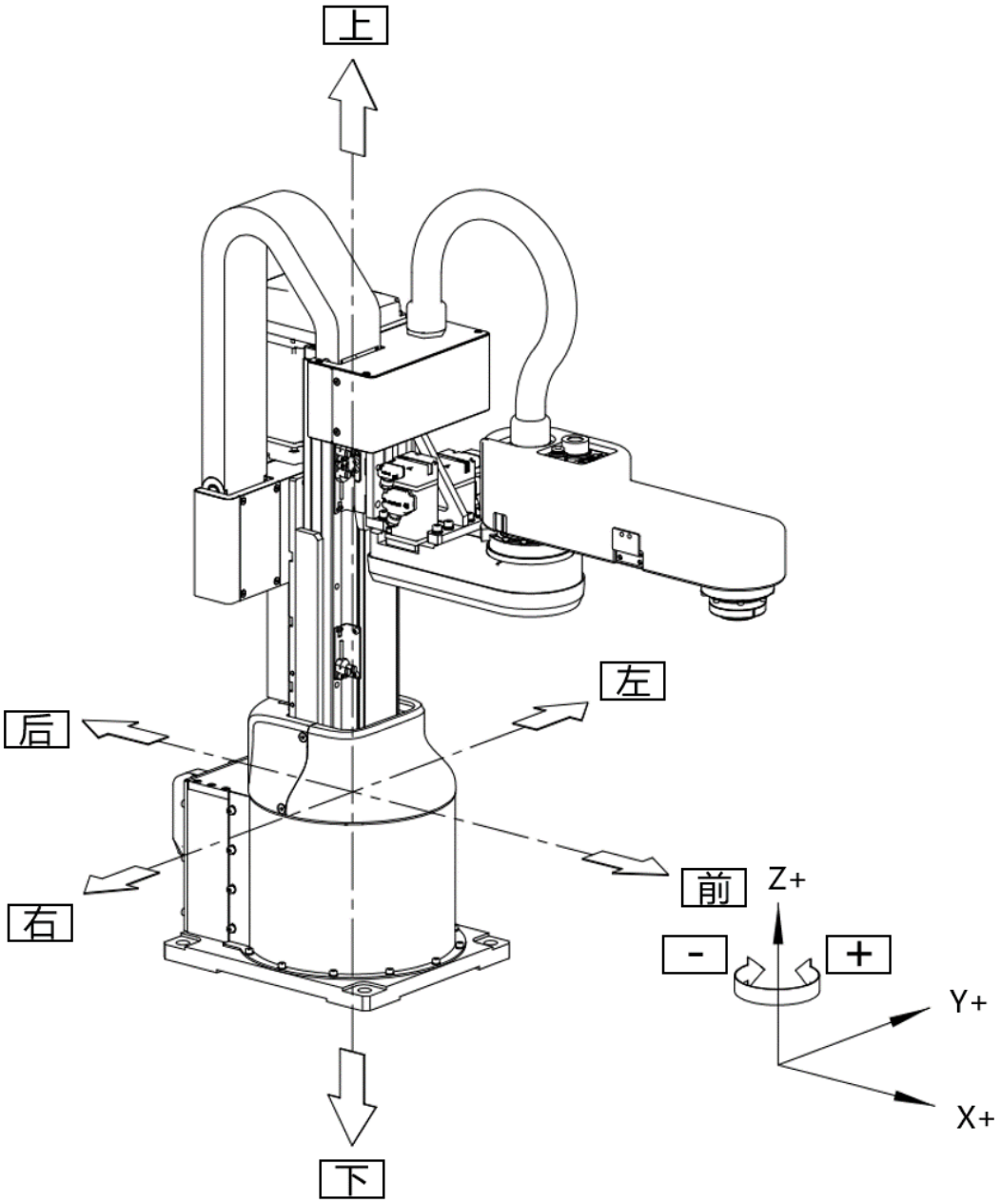


图 39 SCARA-L2 型机器人笛卡尔坐标系

工具坐标-TCS：选中该坐标系情况下，可单独进行机器人沿工具坐标系轴平移或绕坐标系轴旋转。工具坐标系随机器人位姿变化而变化，且随登录选中的工具变换而变换。

坐标系型制同 SCARA-L3 型机器人工具坐标系型制。

用户坐标系-PCS：选中该坐标系情况下，可单独进行机器人沿用户坐标系轴平移或绕坐标系轴旋转。坐标原点及系轴方向随登录选中的用户坐标 变换而变换。详见章节 4.7 内容。

SCARA-L1 型

关节坐标系-ACS：选中该坐标系情况下，可单独进行机器人轴关节绕轴旋转。

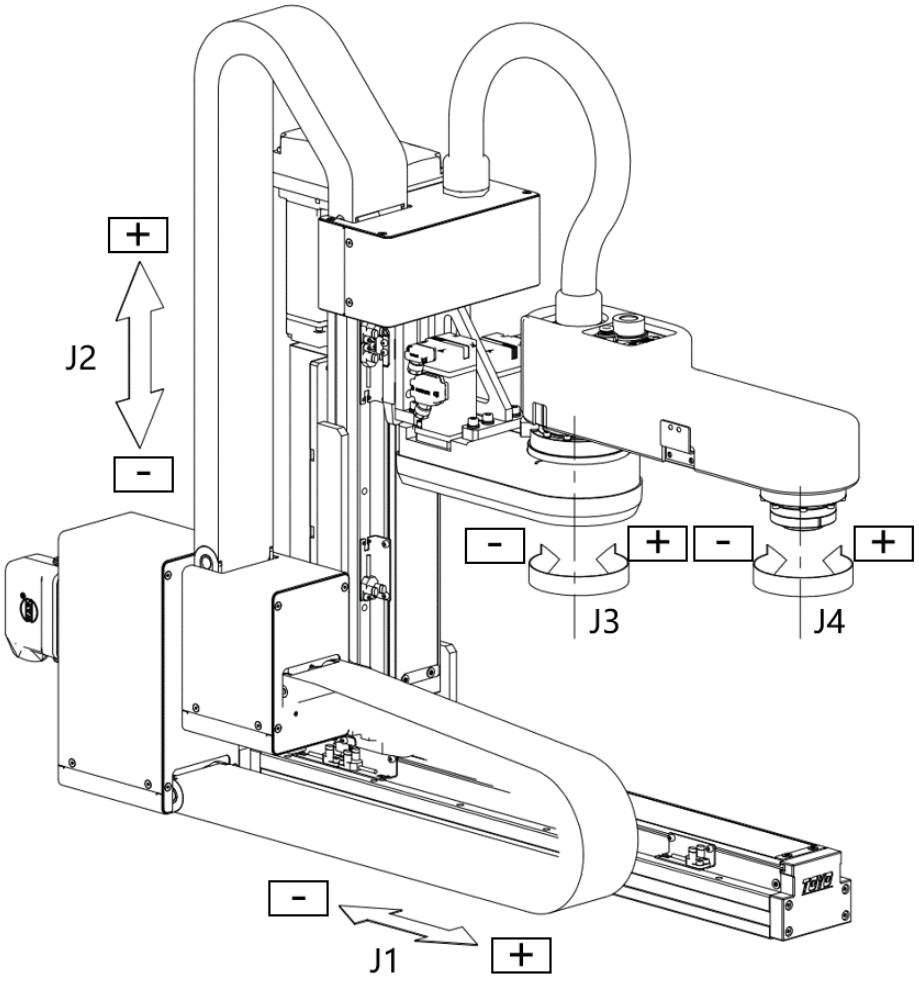


图 40 SCARA-L1 型机器人笛卡尔坐标系

笛卡尔坐标系-MCS：选中该坐标系情况下，可单独进行机器人沿固定直角坐标系平移或绕坐标系轴旋转。坐标系的变换值 X、Y、Z、A、B、C 均为 0。

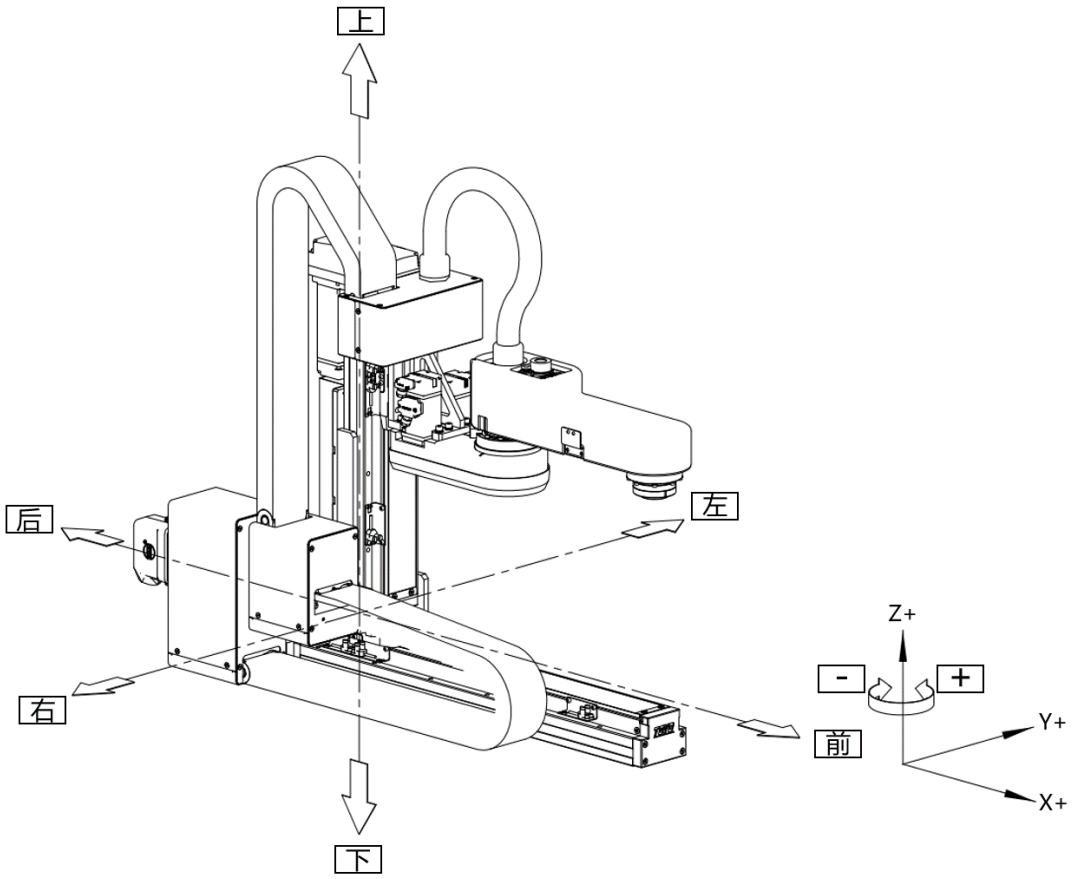


图 41 SCARA-L1 型机器人关节坐标系

工具坐标-TCS：选中该坐标系情况下，可单独进行机器人沿工具坐标系轴平移或绕坐标系轴旋转。工具坐标系随机器人位姿变化而变化，且随登录选中的工具变换而变换。

坐标系型制同 SCARA-L3 型机器人工具坐标系型制。

用户坐标系-PCS：选中该坐标系情况下，可单独进行机器人沿用户坐标系轴平移或绕坐标系轴旋转。坐标原点及系轴方向随登录选中的用户坐标 变换而变换。详见章节 4.7 内容。

4.6.1.3 二轴/三轴/四轴直角坐标机器人坐标系及相应点动

FC 软件支持二轴/三轴/四轴直角坐标机器人控制，直角坐标机器人坐标基本一致，以下以四轴直角坐标机器人为例说明：

关节坐标系-ACS：选中该坐标系情况下，可单独进行机器人轴关节绕轴旋转。

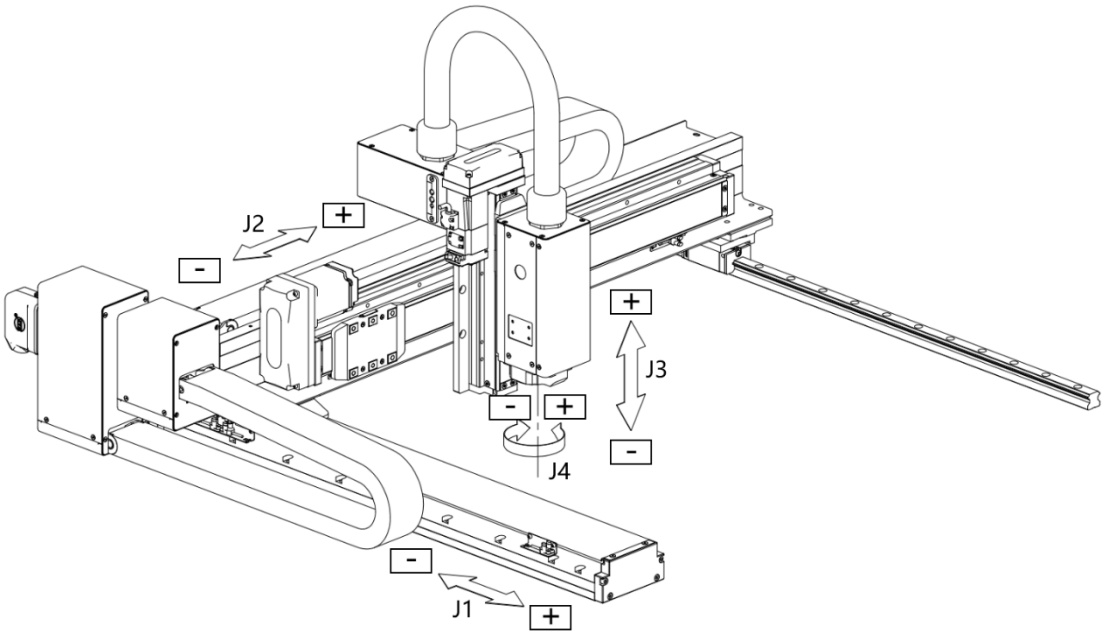


图 42 四轴直角坐标机器人关节坐标系

笛卡尔坐标系-MCS：选中该坐标系情况下，可单独进行机器人沿固定直角坐标系平移或绕坐标系轴旋转。坐标系的变换值 X、Y、Z、A、B、C 均为 0。

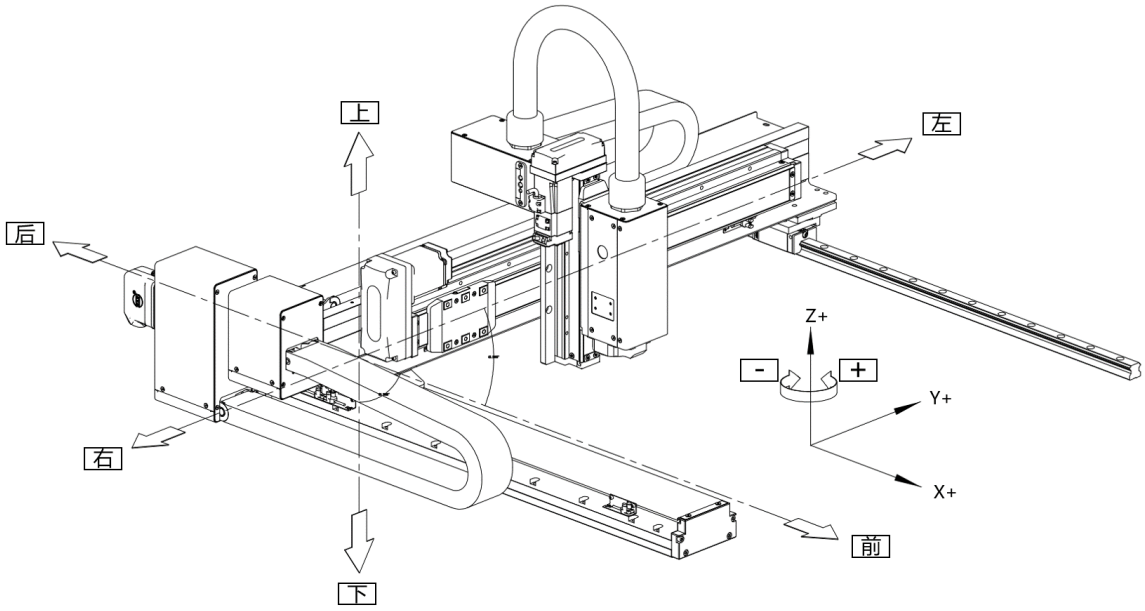


图 43 四轴直角坐标机器人笛卡尔坐标系

用户坐标系-PCS：选中该坐标系情况下，可单独进行机器人沿用户坐标系轴平

移或绕坐标系轴旋转。坐标原点及系轴方向随登录选中的用户坐标 变换而变换。详见章节 4.7 内容。

4.6.2 机器人其他点动示教方式

FC 软件支持总计六种机器人示教点位方式，包括单轴点动、单轴相对距离移动、多轴相对距离移动、单轴绝对点位移动、多轴绝对点位移动、点到点移动，以实现机器人位姿点位示教以及相应。

注意

使用工具坐标系（TCS）及用户坐标系（PCS）下执行相关动作时，需先对TCS及PCS进行标定或手动输入设定，相关设定、标定、调用方式详见4.7章节内容。

4.6.2.1 单轴点动

即单轴 JOG 模式，见 4.6.1 常见机器人坐标系图解及相应点动章节内容

4.6.2.2 单轴相对距离移动

选中“单轴相对距离移动”移动模式，在各轴 “移动距离”窗口输入移动距离，点动相应轴项后 “+”/“-”按键，机器人相应单轴运动至“当前位置”加上/减去对应“移动距离”的位置处，如移动过程中松开按键机器人停止移动，继续点动 “+”/“-”按键，目标位置将更新为新“当前位置”加上/减去对应“移动距离”的位置处。如下图所示：

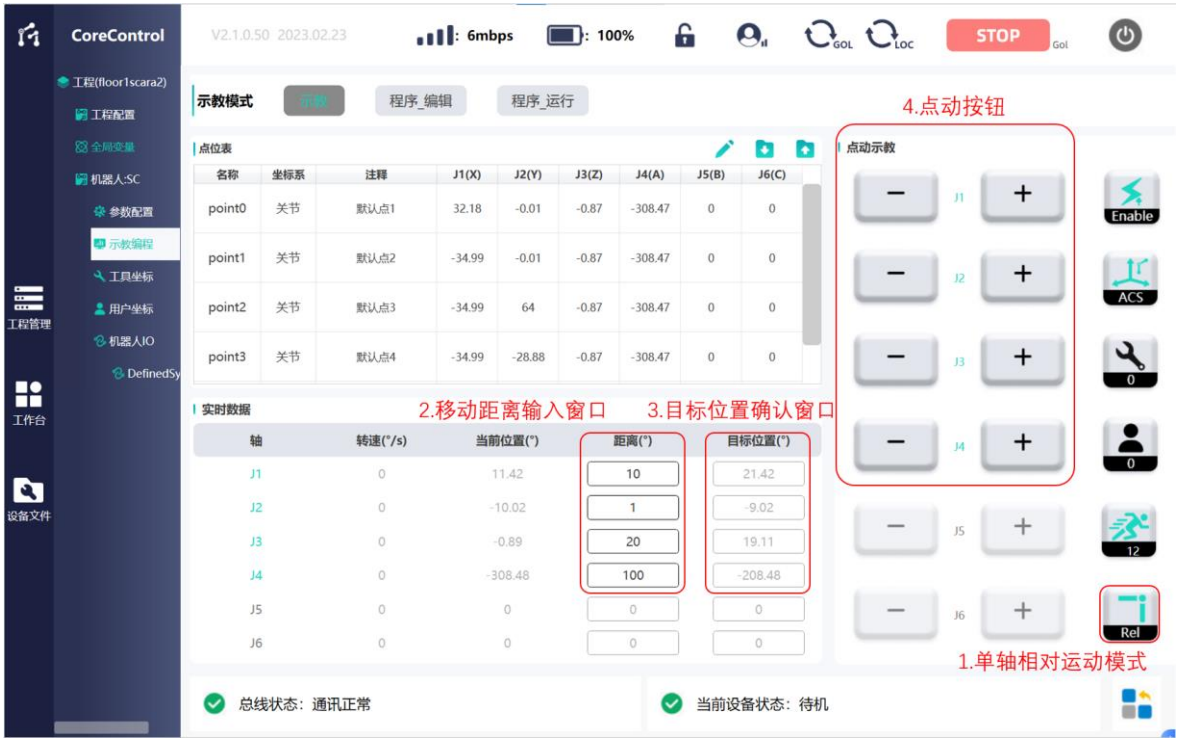


图 44 单轴相对移动模式操作

4.6.2.3 多轴相对距离移动

选中“多轴相对距离移动”移动模式，在各轴“移动距离”窗口输入移动距离，点动“移动”按键，机器人整体运动至“当前位置”加上对应“移动距离”的位置处。如移动过程中松开按键机器人停止移动，继续点动“多轴移动”按键，目标位置将更新为新“当前位置”加上对应“移动距离”的位置处。如下图所示：

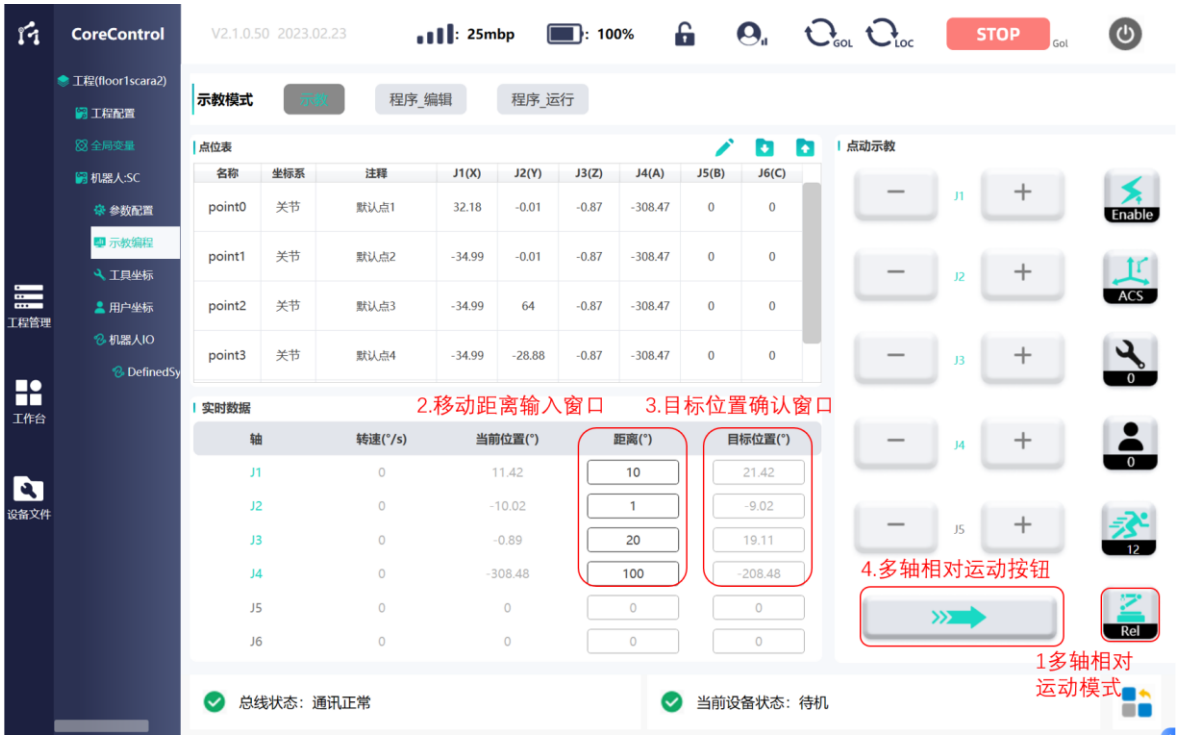


图 45 多轴相对移动模式操作

4.6.2.4 单轴绝对点位移动

选中“单轴绝对点位移动”移动模式，在各轴“目标位置”窗口输入目标点坐标，点动相应轴项“+”按键，单轴开始向目标点移动，到达指定点位或松开按键机器人停止移动。绝对点位移动终点位置由设定位置决定，与点击“移动”按键次数无关。如下图所示：

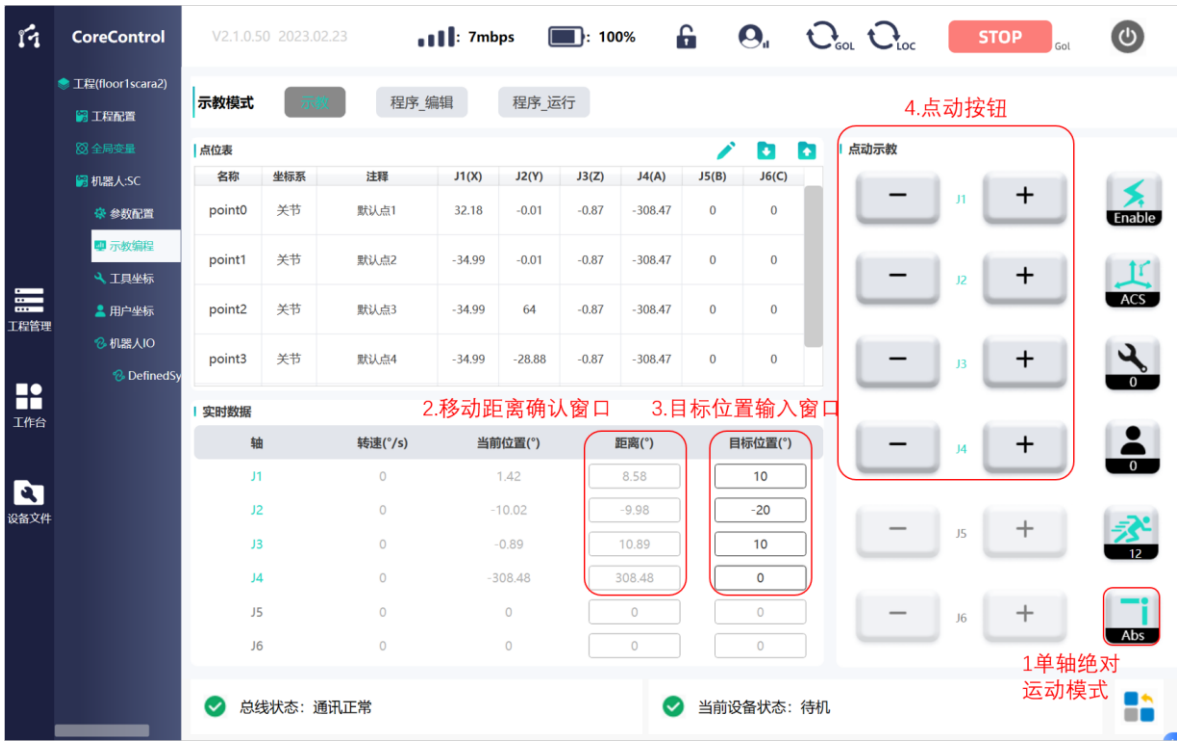


图 46 单轴绝对移动模式操作

4.6.2.5 多轴绝对点位移动

选中“多轴绝对点位移动”移动模式，在各轴“目标位置”窗口输入目标点坐标，点动“移动”按键，多轴同时开始向各自目标点移动，到达指定点位或松开按键机器人停止移动。绝对点位移动终点位置由设定位置决定，与点击“移动”按键次数无关。

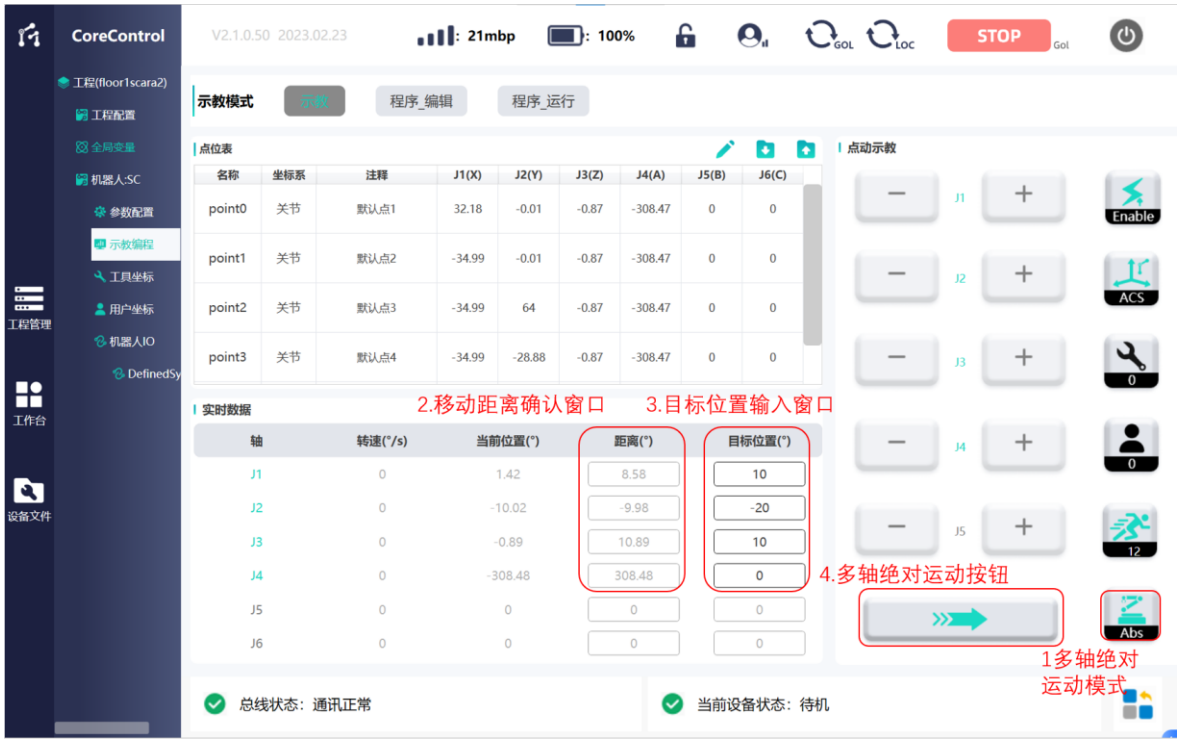


图 47 多轴绝对移动模式操作

4.6.2.6 移动到指定点

选中“点到点”移动模式，于点位表单击需到达点，长按“点到点运动”按键，根据所选坐标系的不同，机器人以相应 PTP 方式移动至指定点，到达指定点位或松开按键机器人停止移动。终点位置由选定点位决定，与点击按键次数无关。如下图所示：

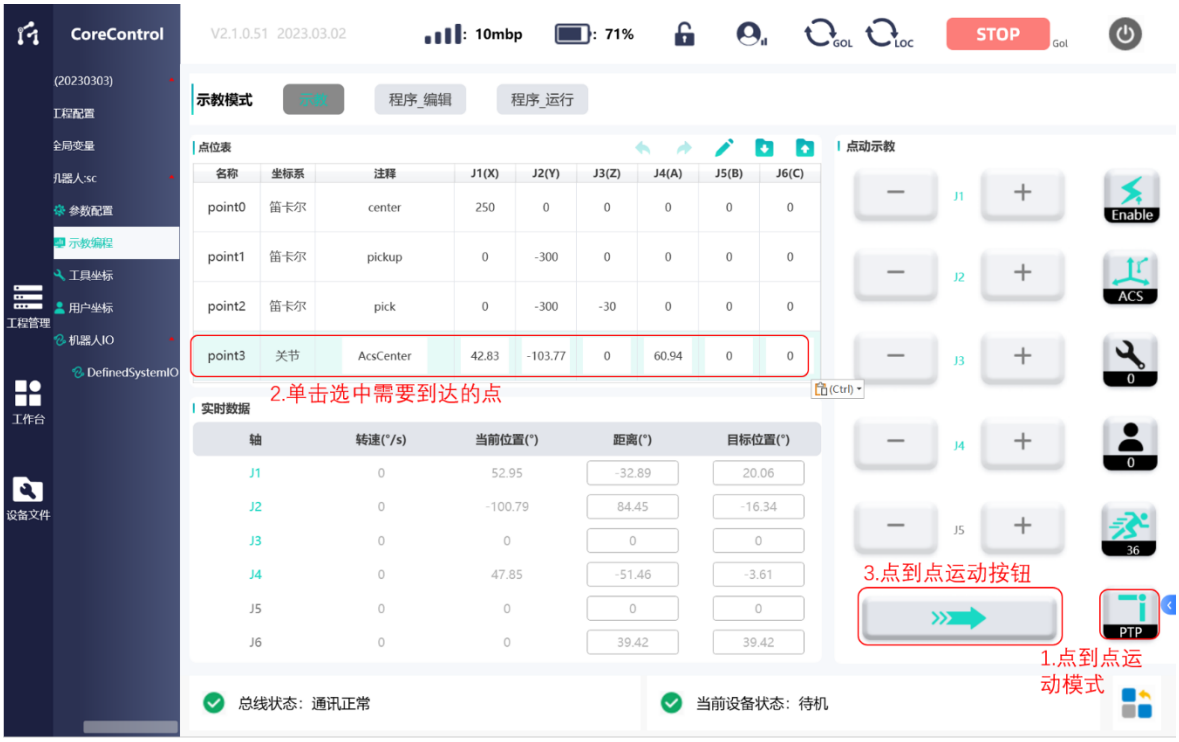


图 48 定点移动操作

4.6.3 机器人点位记录

FC 软件为编译式控制逻辑，程序中点位需独立示教记录，编程时进行插入调用。FC 软件点位信息记录通过“点位总表”窗口实现。

4.6.3.1 点位总表窗口介绍

点位总表窗口实现功能包括：点位坐标记录、坐标值显示及修改、点位注释名添加修改、点位在点状态显示、点表文件形式导入导出，点表默认已记录 6 点，可执行所有相关操作，窗口布局如下：

点位表

名称	坐标系	注释	J1(X)	J2(Y)	J3(Z)	J4(A)	J5(B)	J6(C)
point0	关节	默认点1	32.18	-0.01	-0.87	-308.47	0	0
point1	关节	默认点2	-34.99	-0.01	-0.87	-308.47	0	0
point2	关节	默认点3	-34.99	64	-0.87	-308.47	0	0
point3	关节	默认点4	-34.99	-28.88	-0.87	-308.47	0	0

图 49 点表布局

4.6.3.2 点位坐标记录

点位坐标记录方式共四种，包括记录到点、点前插入、点后插入、点表末位快捷插入。

1. 记录到点（当前点坐标覆盖）

在点表中双击需要被覆盖的点，点表中会弹出功能选择框，选中“记录到点”，示教器将会把机器人当前位置和坐标系覆盖记录进选择的点位中，点位覆盖动作不可撤销，确保被覆盖的点是不需要的。

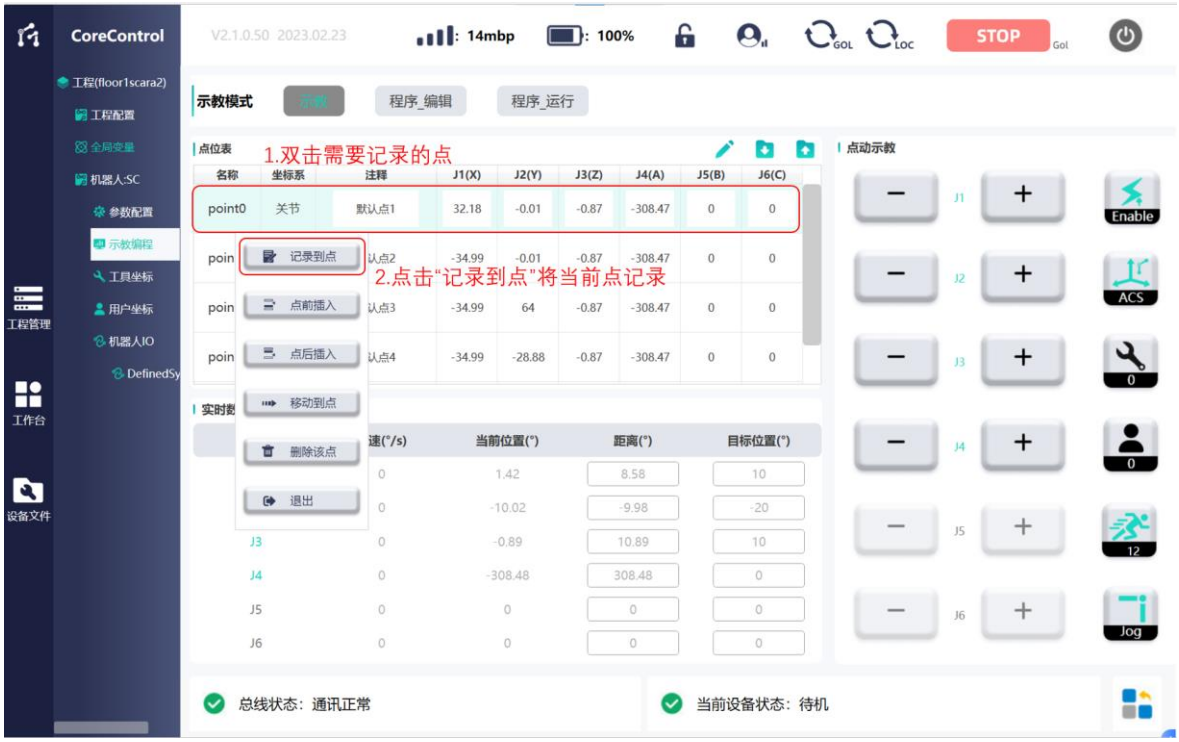


图 50 记录到点操作

2. 点前插入

在点表中双击需要插入的点，点表中会弹出功能选择框，选中“点前插入”，机器人当前点和坐标系将会记录到一个最新的“point”中，并在点表中插入至此点前一

行。

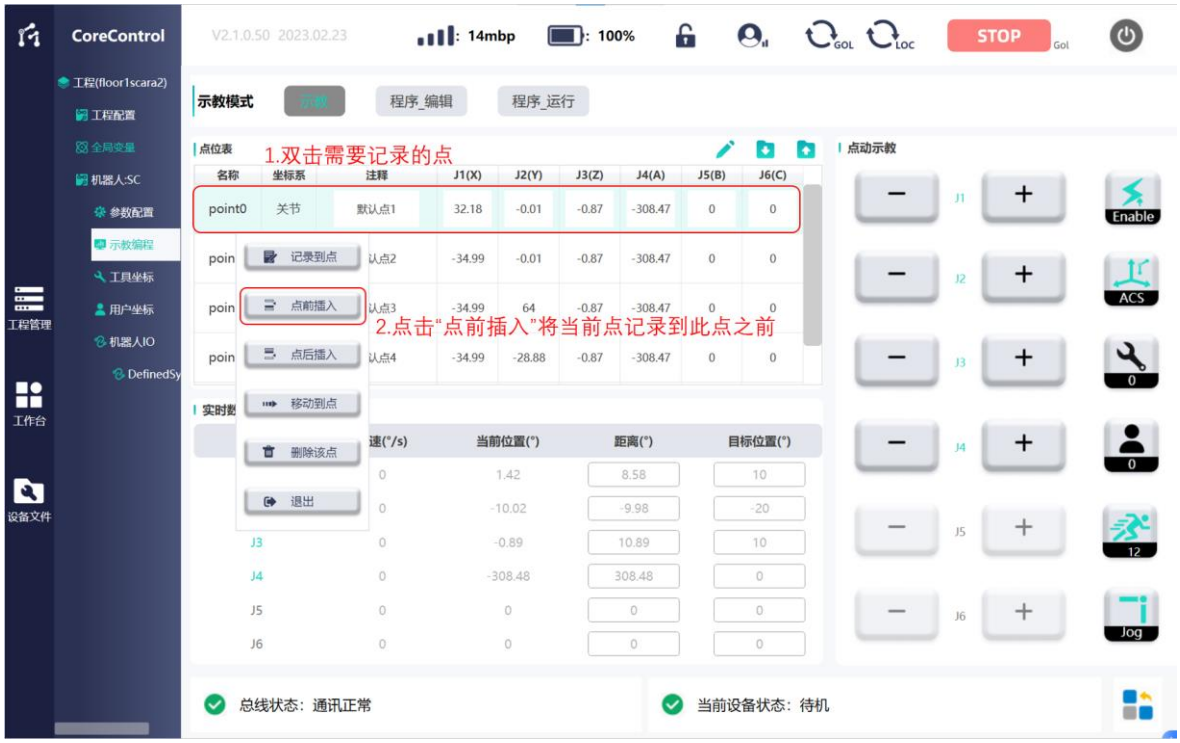


图 51 点前插入操作

3. 点后插入

在点表中双击需要插入的点，点表中会弹出功能选择框，选中“点后插入”，机器人当前点和坐标系将会记录到一个最新的“point"中，并在点表中插入至此点后。

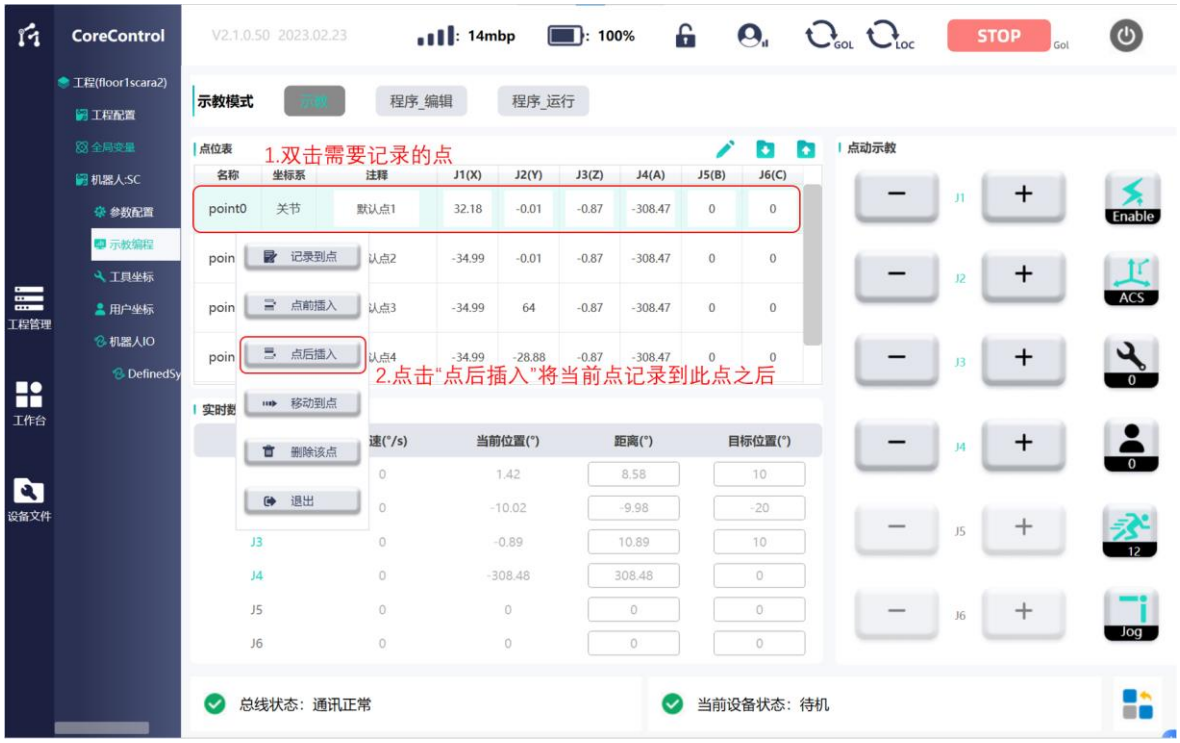


图 52 点后插入操作

4. 点表末位快捷插入

在点表上方点击“记录点”按钮，可以将机器人当前点和坐标系存为一个“point”快速插入至点表末尾。

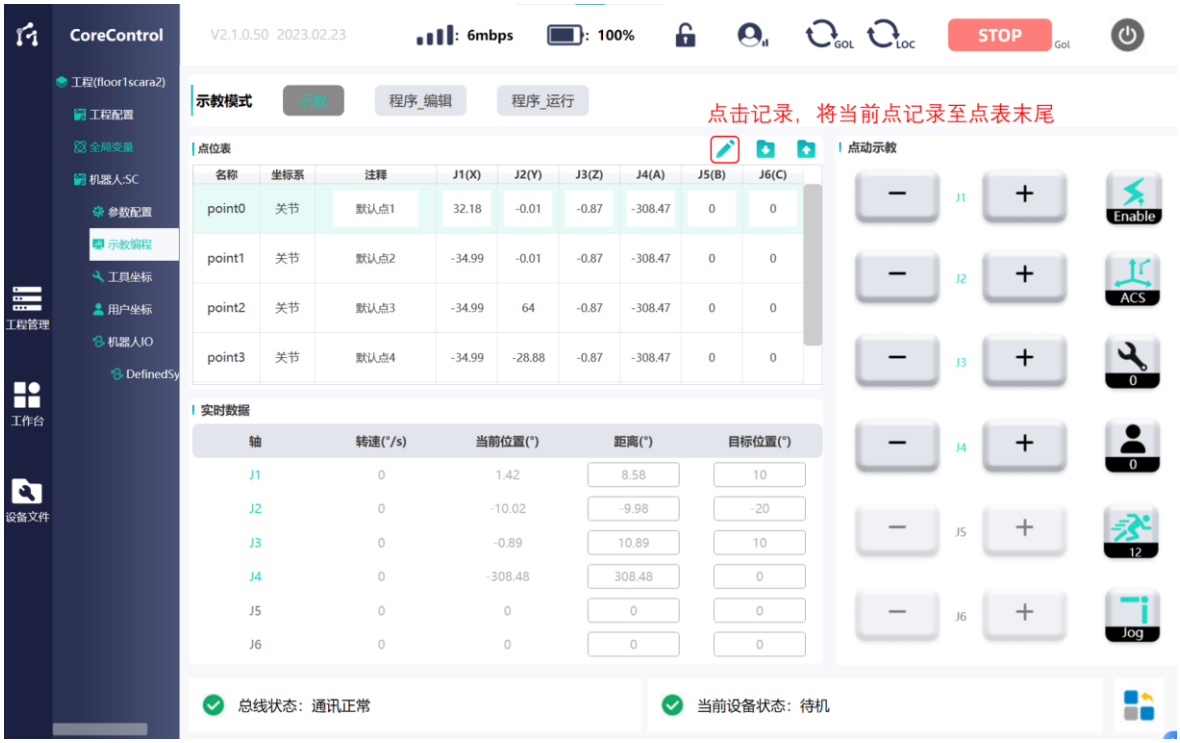


图 53 快捷记录操作

4.6.3.3 点位坐标值显示及修改

点位总表窗口显示所有点位各轴或欧拉角坐标值，未记录点参数默认为“0”，点表记录点数上限 200 点位，点表右侧点击拖动显示。

点击点位后坐标值表格可手动填写坐标值，填写完成需点击“点位保存”，将数据上载保存。如下图所示：

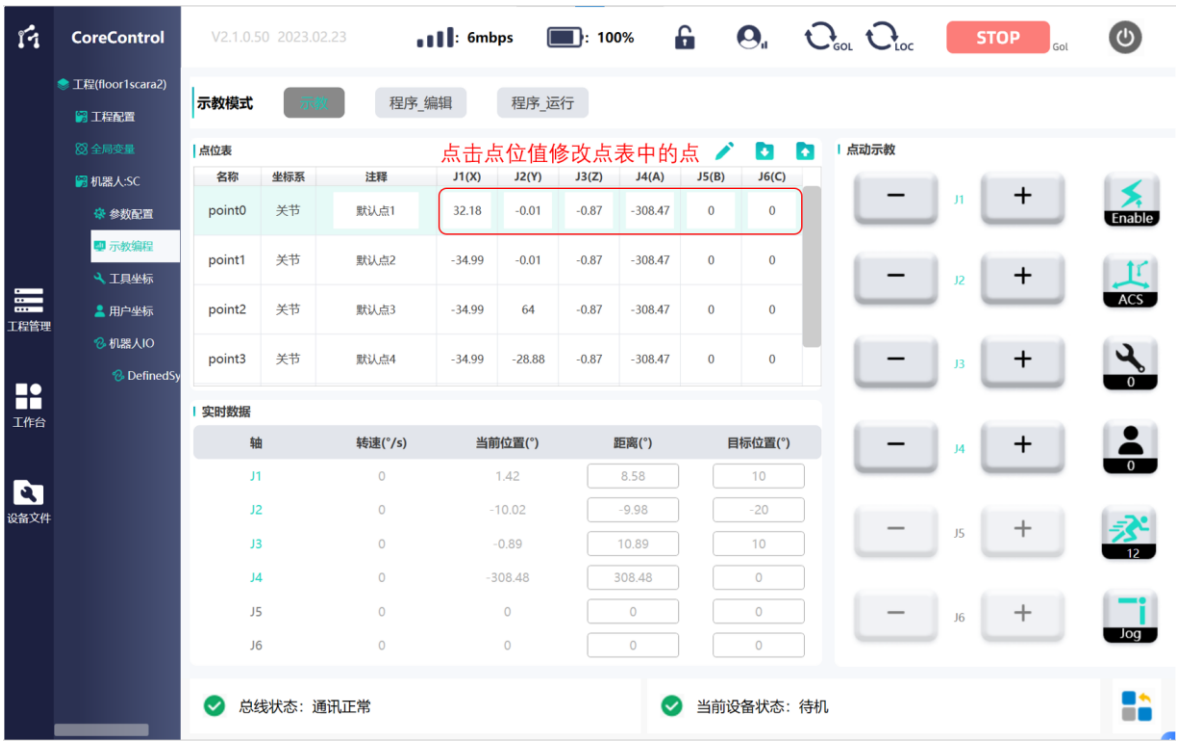


图 54 点位数据修改操作

4.6.3.3 点位注释名添加修改

点击各点在点位总表窗口中相应“注释”项可添加修改各点注释名，“注释”名支持中英文输入，填写完成需点击“点位保存”，将数据上载保存。如下图所示：

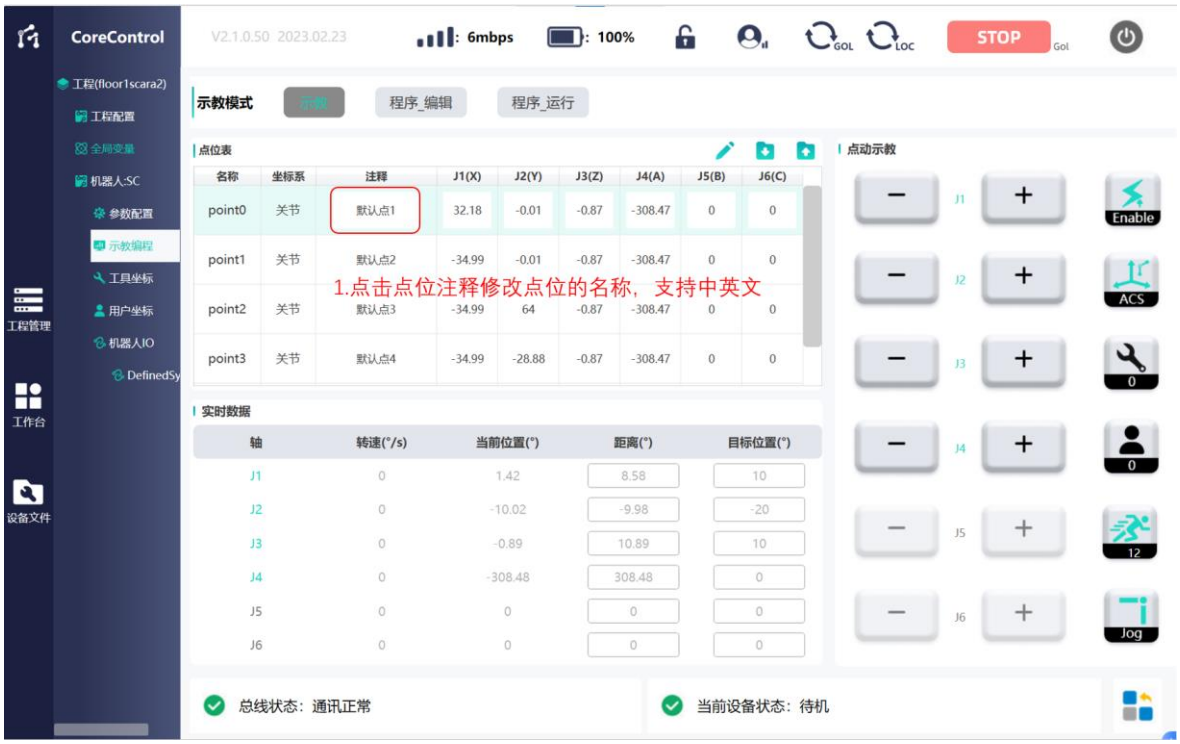


图 55 点位注释及修改操作

4.6.3.3 点表文件形式导入导出

点击“文件导出”可将点表以 fsmdata 文件形式导出至指定存储位置，点击“文件导入”可将指定位置点表 fsmdata 文件导入覆盖当前系统存储执行的点表，导入点表格式需与导出文件格式一致，否则无法读取。

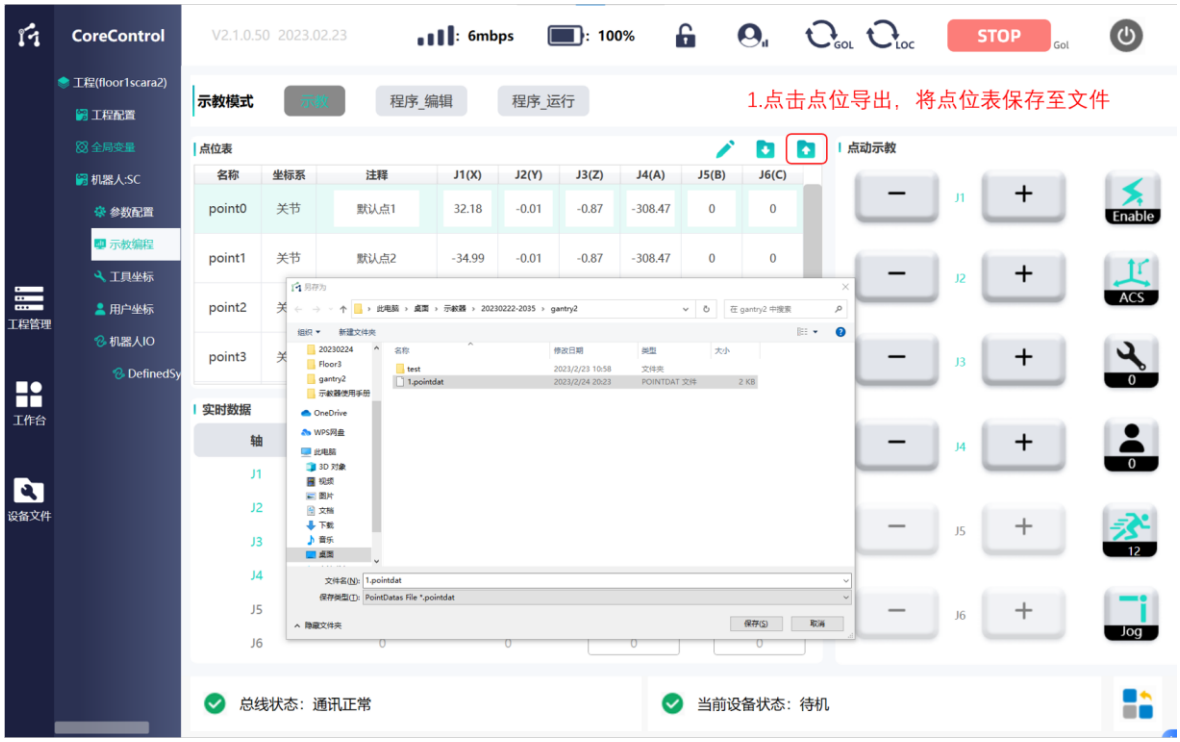


图 56 点表导出操作

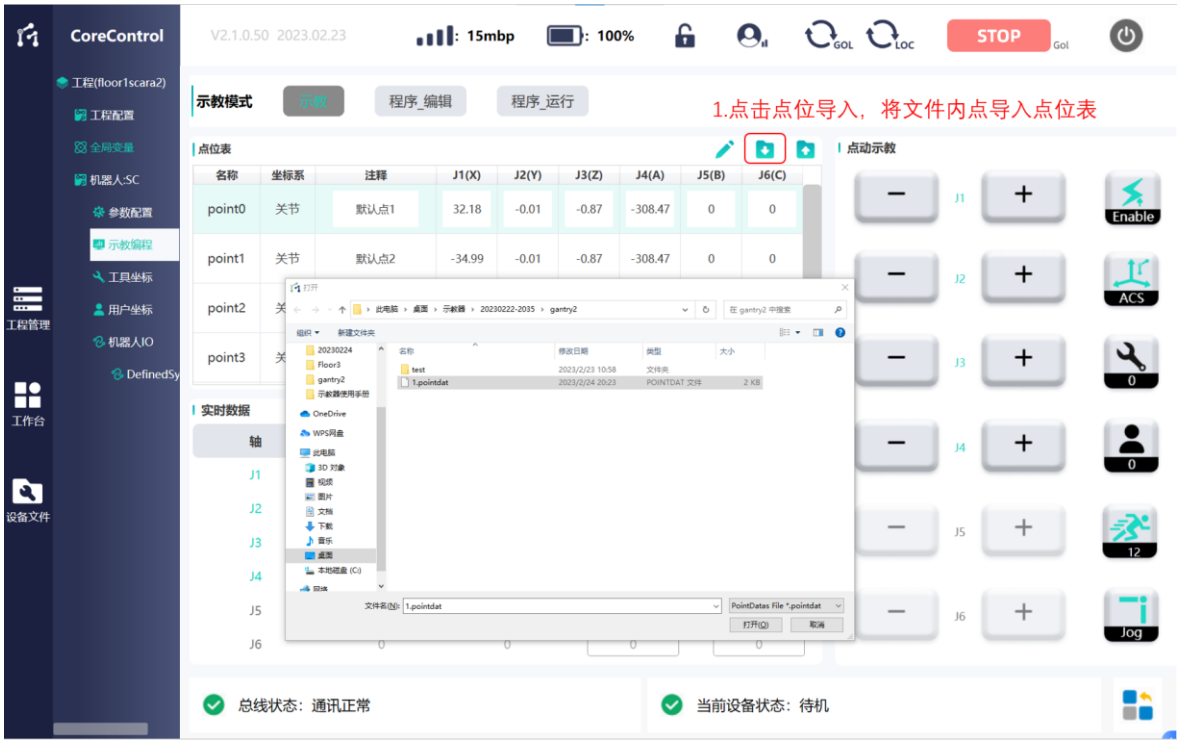


图 57 点表导入操作

4.7 工具坐标系/用户坐标系标定及相关选用

FC 软件支持工程内所控每台机器人总计 10 个工具的标定及选用、10 个用户坐标系的标定及选用。所有工具未经标定默认 TCP 为（0,0,0,0,0,0），所有用户坐标系未经标定默认原点为（0,0,0,0,0,0），未经标定亦可进行选用。

4.7.1 工具坐标系标定、设定

4.7.1.1 工具坐标系标定方式说明

FC 软件支持工具坐标示教六点标定，示教点及示教顺序如下图所示:

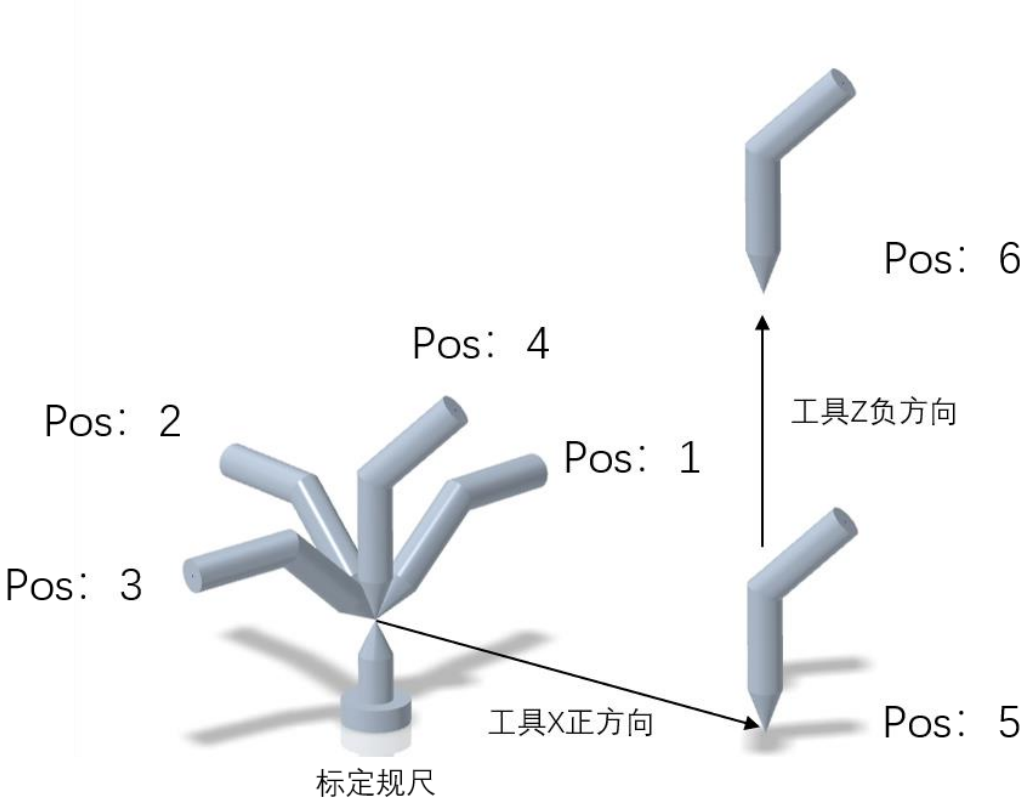


图 58 工具标定六点示意

六点标定算法支持 4 轴 SCARA 机器人、4 轴直角坐标机器人、6 轴机器人使用，2 轴直角坐标机器人及 3 轴直角坐标机器人因不具备旋转轴，不使用工具坐标系。六点标定各点说明如下：

1. 1-4 点为基本位姿，要求工具固定位置瞄准并接触同一测量规尺同一原点，此时各姿态应均包括同一位置数据，而姿态数据不同 。确保每个姿态间角度差异 45°~90°，六轴法兰面在每个姿态中应处于不同平面。
   SCARA 机器人因缺少 RX、RY 自由度，1-4 点示教器时法兰面在每个姿态中可处于同一平面，其他要求一致。
2. 第 5 点为方向位姿，用于标定当前工具对应工具坐标系的 X 正方向，需示教 4 点后使工具沿所需定为该工具坐标系的 X 正方向移动大于 100mm。

3. 第 6 点为方向位姿，用于标定当前工具对应工具坐标系的 Z 轴方向，需示教 5 点后使工具沿所需定为该工具坐标系的 Z 负方向上抬大于 100mm。

注意

工具坐标系标定用六点无严格示教顺序，可乱序示教，需确保记录时对应姿态点位记录至对应定义点内。
需确保标定的六点符合上述说明内的要求，否则导致标定失败。

4.7.1.2 工具标定点记录方法

于管理树栏选中“工具坐标”选项，进入工具坐标标定、设定界面，界面内仅支持基于 ACS、MCS 坐标系动作，工具标定仅记录笛卡尔坐标系（MCS）下坐标值。

点击“↑”/“↓”按键，切换所需标定的工具序号，依照 4.7.1 章节所述要求控制机器人至指定点后，点击“快捷记录按键”依次将点位信息记录入点表窗口：

图 59 工具标定操作

如需手动选择待记录点、修改已记录点信息，点击所需修改点，点击“快捷记录点”，以覆盖数据。

所有点全部按照要求记录后，该工具对应坐标于底部自动生成，该工具即可于 4.6 章节所述动作中调用，如六点偏差过大导致无法工具自动计算，建议删除点位信息重新示教。

4.7.1.2 工具坐标系手动输入

点击“↑”/“↓”按键，切换所需标定的工具序号，于工具坐标生成显示窗口

点击输入笛卡尔坐标系（MCS）偏移值。

图 60 工具偏移值手动输入操作

4.7.1.3 工具坐标系复制、添加

FC 软件支持单台机器人下挂 10 件工具内部之间的工具坐标复制添加，同时支持工程内下挂所有机器人之间的工具坐标复制添加。

于期望被复制的机器人工具坐标界面，通过“↑”/“↓”按键，切换所需复制的工具序号，点击“复制”按键，该工具信息即被复制添加于内存：

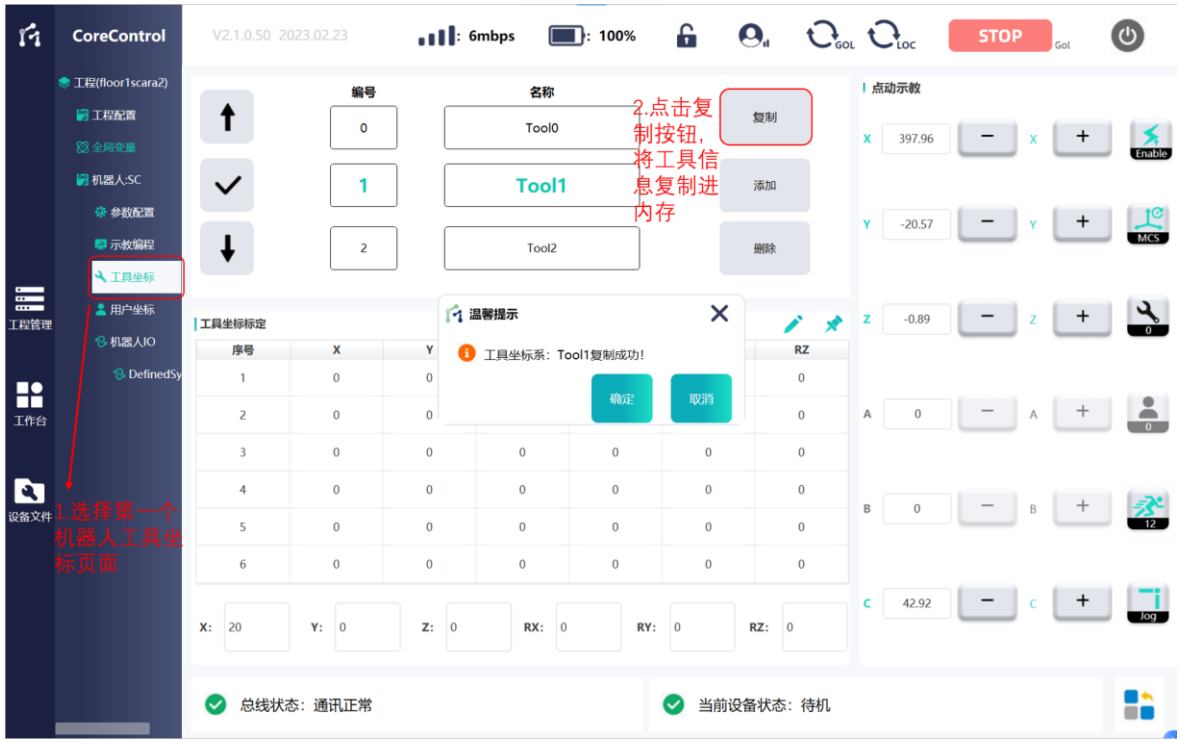


图 61 工具复制操作

于期望添加的机器人工具坐标界面，通过“↑”/“↓”按键，切换所需添加的工具序号，点击“添加”按键，该工具信息即被复制添加于指定工具序号内。

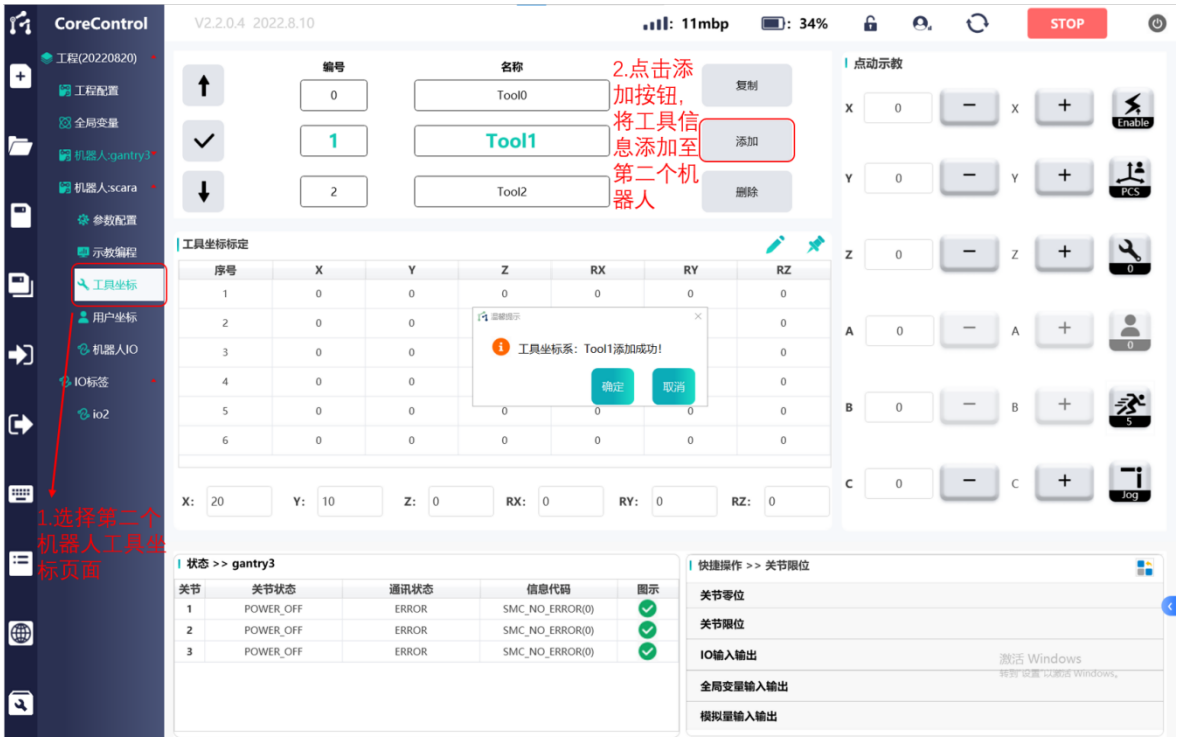


图 62 工具复制添加操作

4.7.1.4 工具坐标系查询

点击“↑”/“↓”按键，切换所需查询标定的工具序号，于工具坐标生成显示窗口偏移值及工具坐标值，操作同 4.7.1.2 章节内容。

4.7.2 用户坐标系标定、设定

4.7.2.1 用户坐标系标定方式说明

FC 软件支持用户坐标系通过示教三点标定，示教点及示教顺序如下图所示：

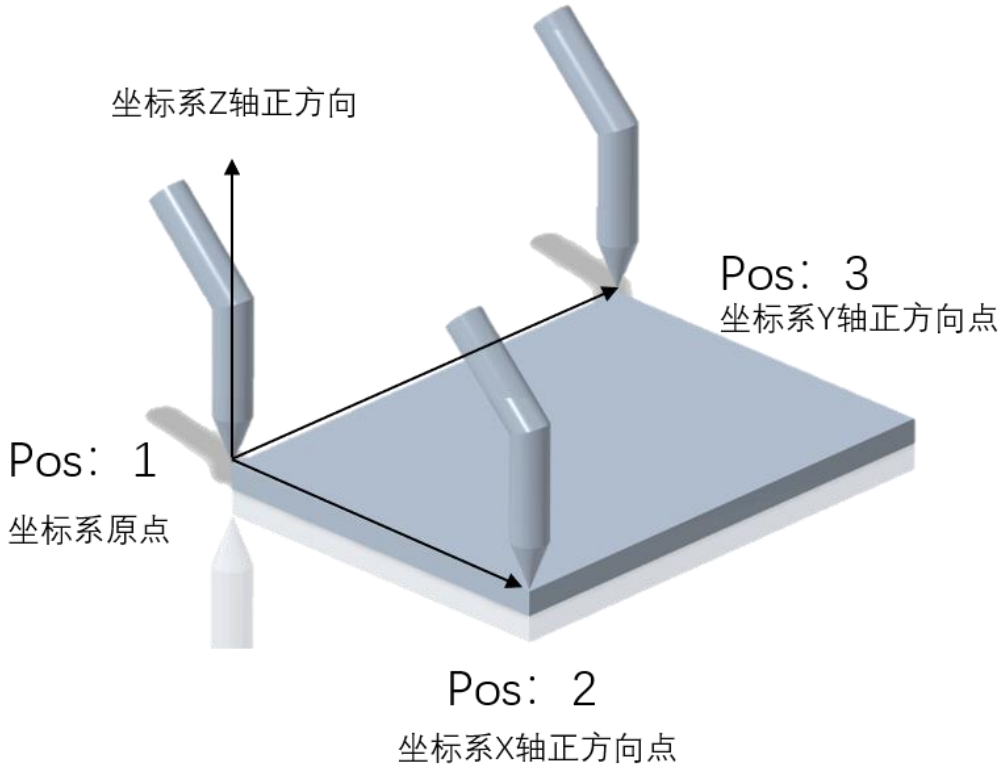


图 63 用户坐标系三点标定示意

三点标定算法支持所有轴机器人使用。三点标定各点说明如下：

1. 第 1 点为该用户坐标系的原点位置，用户坐标系标定算法只对标定点位置信息进行读取运算，对示教时姿态无明确要求。
2. 第 2 点为方向位姿，用于标定当前用户坐标系的 X 正方向，原点与第 2 点的连线方向即为该用户坐标系 X 轴正方向，第 2 点与第 1 点间距建议大于 100mm，用户坐标系标定算法只对标定点位置信息进行读取运算，对示教时姿态无明确要求。
3. 第 3 点为方向位姿，用于标定当前用户坐标系的 Y 正方向，原点与第 3 点的连线方向即为该用户坐标系 Y 轴正方向，第 3 点与第 1 点间距建议大于 100mm，用户坐标系标定算法只对标定点位置信息进行读取运算，对示教时姿态无明确要求。

注意

用户坐标系标定用三点无严格示教顺序，可乱序示教，需确保记录时对应姿态点位记录至对应定义点内。
需确保标定的三点符合上述说明内的要求，否则导致标定失败。

4.7.2.2 用户标定点记录方法

于管理树栏选中“用户坐标”选项，进入用户坐标标定、设定界面，界面内仅支持基于 ACS、MCS 坐标系动作，用户标定仅记录笛卡尔坐标系（MCS）下坐标值。

点击“↑”/“↓”按键，切换所需标定的用户序号，依照 4.6 章节所述要求控制机器人至指定点后，点击“快捷记录按键”依次将点位信息记录入点表窗口：

图 64 用户坐标系标定操作

如需手动选择待记录点、修改已记录点信息，点击所需修改点，点击“快捷记录点”，以覆盖数据。

所有点全部按照要求记录后，该用户坐标系坐标于底部自动生成，该用户即可于 4.6 章节所述动作中调用

4.7.2.2 用户坐标系手动输入

点击“↑”/“↓”按键，切换所需标定的用户序号，于用户坐标生成显示窗口点击输入笛卡尔坐标系（MCS）偏移值。

图 65 用户坐标系标定操作

4.7.2.3 用户坐标系复制、添加

FC 软件支持单台机器人下挂 10 件用户内部之间的用户坐标复制添加，同时支持工程内下挂所有机器人之间的用户坐标复制添加。

于期望被复制的机器人用户坐标界面，通过“↑”/“↓”按键，切换所需复制的用户序号，点击“复制”按键，该用户信息即被复制添加于内存：

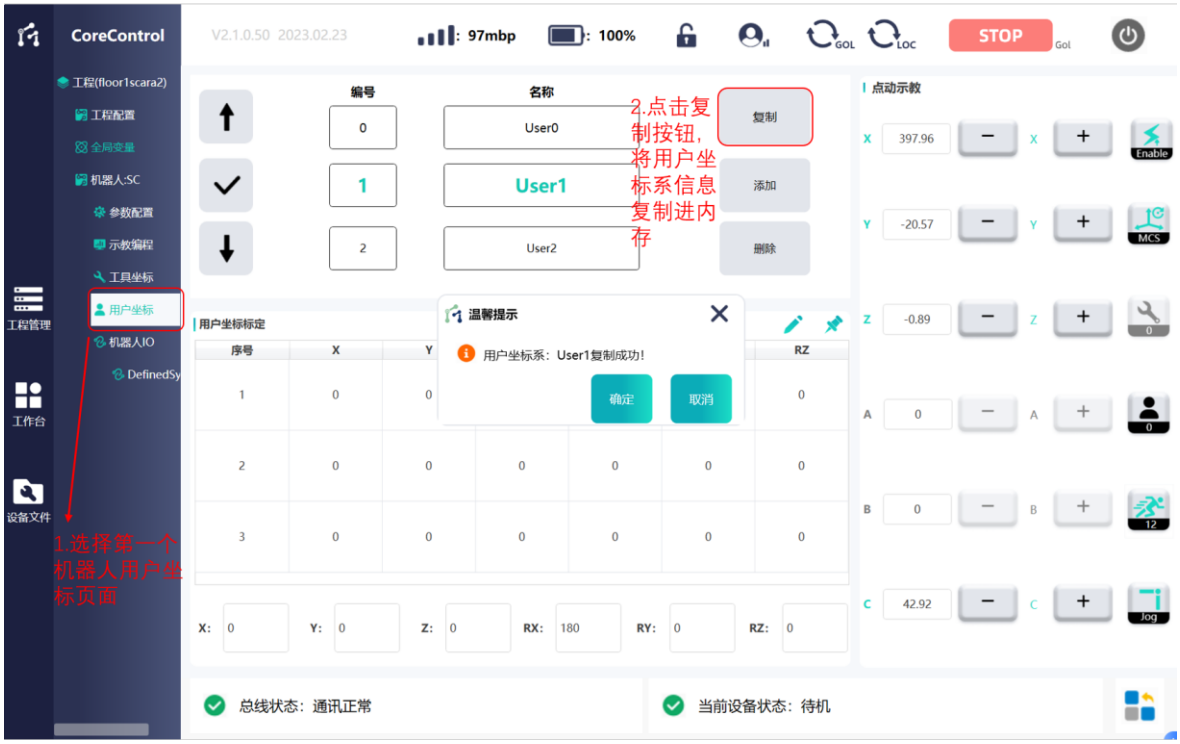


图 66 用户坐标系复制操作

于期望添加的机器人用户坐标界面，通过“↑”/“↓”按键，切换所需添加的用户序号，点击“添加”按键，该用户信息即被复制添加于指定用户序号内。

4.7.2.4 用户坐标系查询

点击“↑”/“↓”按键，切换所需查询标定的用户序号，于用户坐标生成显示窗口偏移值及用户坐标值。操作同 4.7.2.2 章节内容.

4.8 辅助快捷功能

FC 软件支持关节限位设定、关节零位设定、已关联 IO 快捷输入输出、已关联全局变量输入输出快速启动五项功能。

图 67 快捷动作总览

4.8.1 关节零位设定

机器人轴轴关节零位设定共分三种方式，其中两种为快捷输入输出设定，其余一种为机器人参数配置界面直接填入（效果与第 2 项轴关节零位编码值设定一致）。已标定零位数据寄存于工程文件内部，相关参数掉电保持，不随控制器/柜关机、示教器关闭影响。

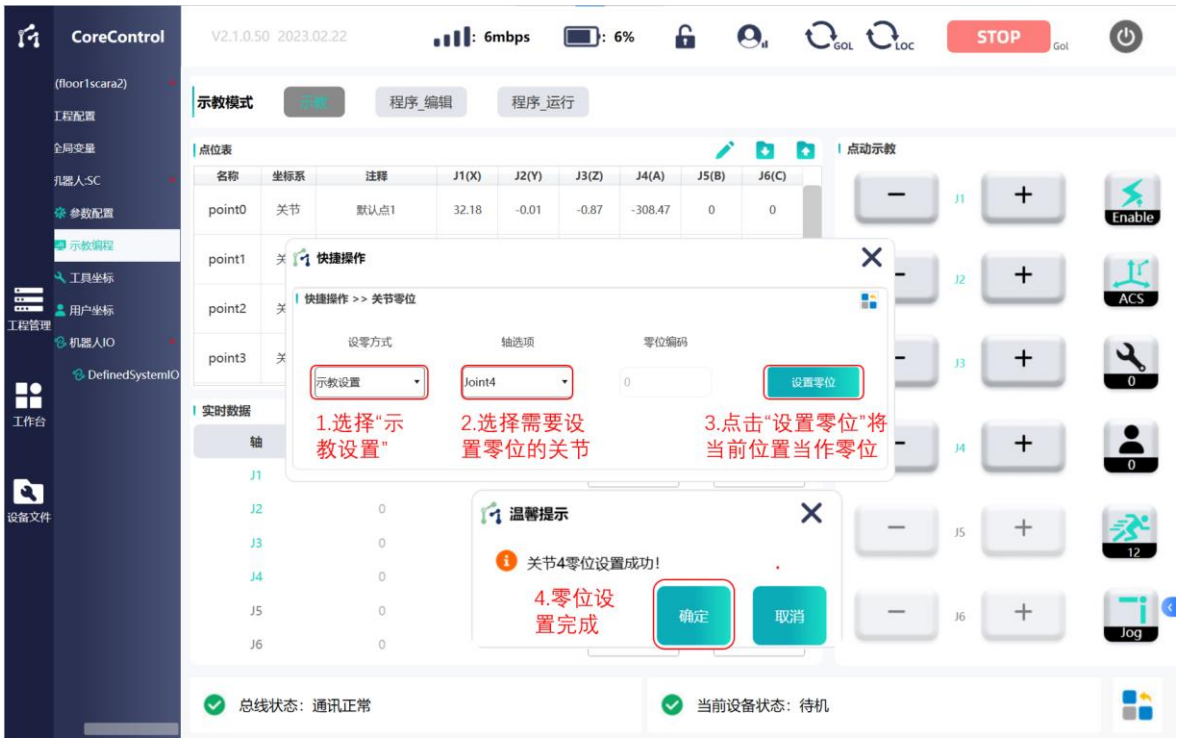
选择需要设置关节零位的机器人，切换至相应的“示教编程”页面，在快捷动作中选择“关节零位设置”。

图 68 关节零位功能选择

1. 轴关节零位示教标定

选择设零方式为“示教设置”，选择需要设置零位的机器人关节，若关节不在零位位置，按照“4.6 机器人坐标系标定及手动示教”提供的方法，手动示教至零位位置，按下“设置零位”按钮，弹出提示框并点击确定后，机器人关节零位设置完成。

图 69 关节零位标定



2. 轴关节零位编码值设定

选择设零方式为“零位编码值”，选择需要设置零位的轴并输入零位编码，点击“设置零位”按钮，待提示窗口弹出后，点击“确定”按钮后该关节零位设置成功。

之后此关节的编码器值等于输入的零位编码时，关节位于零位上。

零位编码需要用户使用相应电机驱动器调试软件读取零位位置时电机编码器值，或由芯控公司提供《机器人参数配置表》或对应二维码扫码获得。

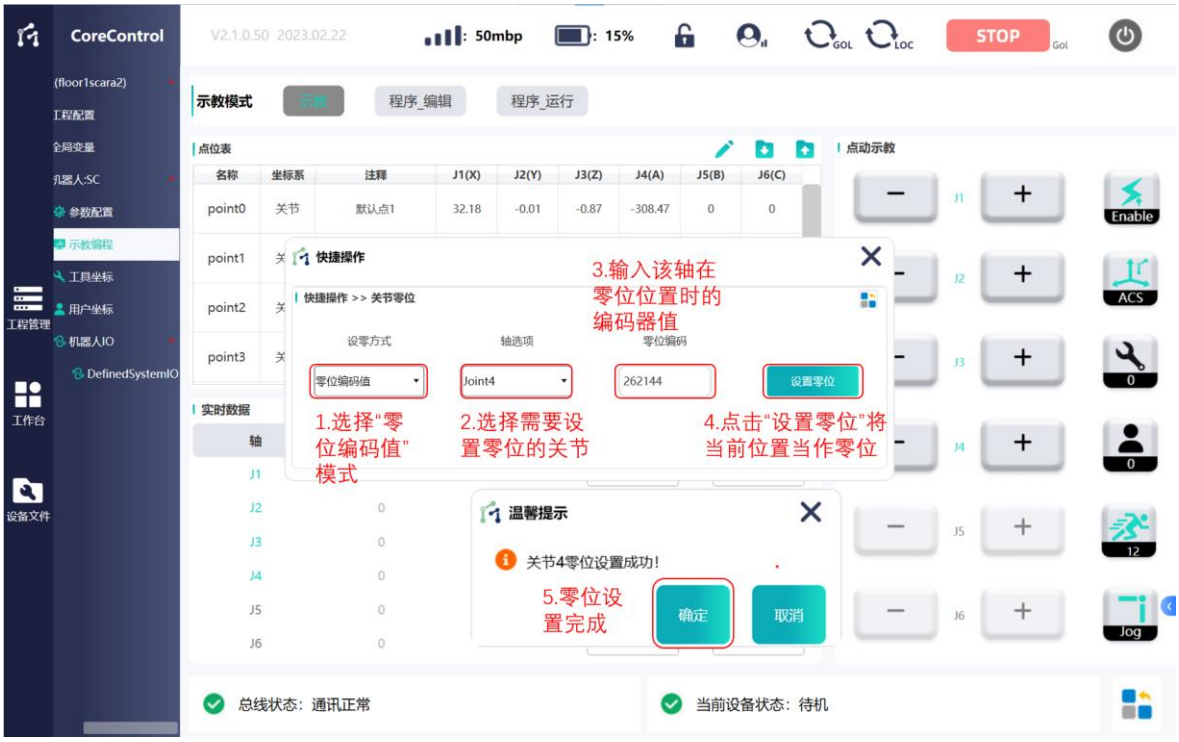


图 70 关节零位设定

注意

机器人轴零位设定关系到机器人坐标系准确与否，进行机器人除关节坐标系（ACS）外坐标系下移动前须优先进行机器人轴关节零位标定。
机器人轴限位标定、激活前须优先进行机器人轴关节零位标定，否则易出现轴使能即报错超出限位。

4.8.2 关节限位设定

机器人轴轴关节限位设定共分两种方式，已标定限位数据寄存于工程文件内部，相关参数掉电保持，不随控制器/柜关机、示教器关闭影响。

选择需要设置关节限位的机器人，切换至相应的“示教编程”页面，在快捷动作中选择“关节限位”。

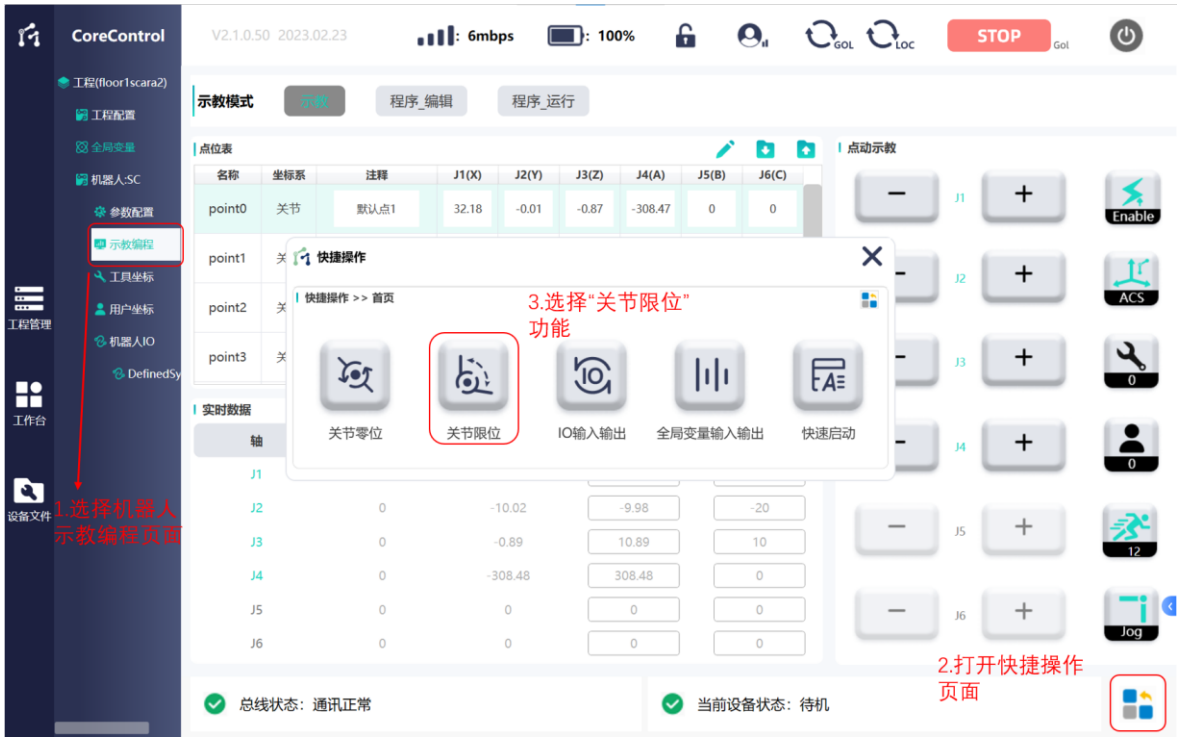


图 71 关节限位功能选中

注意

机器人轴限位标定、激活前须优先进行机器人轴关节零位标定，否则易出现轴使能即报错超出限位。
关节零位设定详见4.8.1章节内容。

1. 轴关节限位示教标定

在关节限位快捷动作中，左侧的“激活”确认栏可以选择是否激活限位，根据自身情况选择后，选择设置方式为“示教设置”后，用户可通过“4.6 机器人坐标系标定及手动示教”提供的方法，手动示教单轴至正限位或负限位上，点击“正限位”或“负限位”下方的输入框，数据自动写入，最后点击“设置限位”，完成标定。

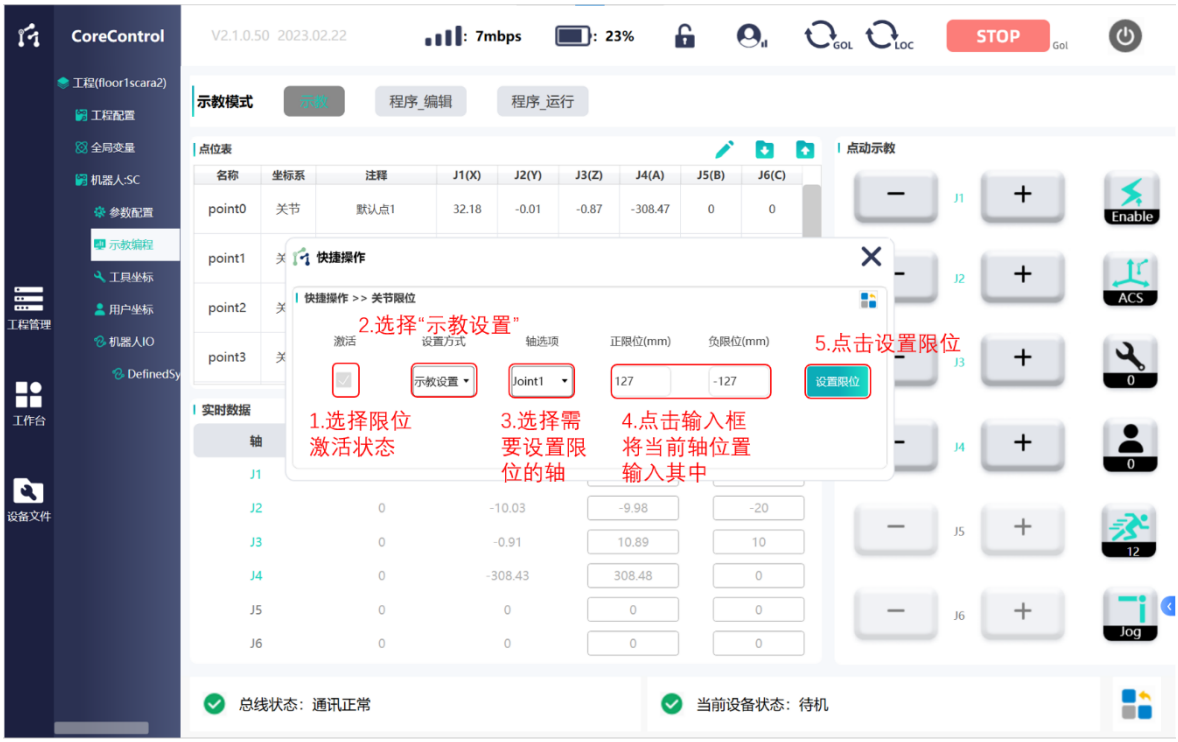


图 72 关节限位标定

2. 轴关节限位数值设定

在关节限位快捷动作中，左侧的“激活”确认栏可以选择是否激活限位，根据自身情况选择后，选择设置方式为“输入参数”后，轴选项选择需要设置限位的关节，在“正限位”、“负限位”的输入框中直接输入需要的正限位、负限位的值，点击“设置限位”按钮，待右下角提示窗口弹出后，此时关节限位设置完成。

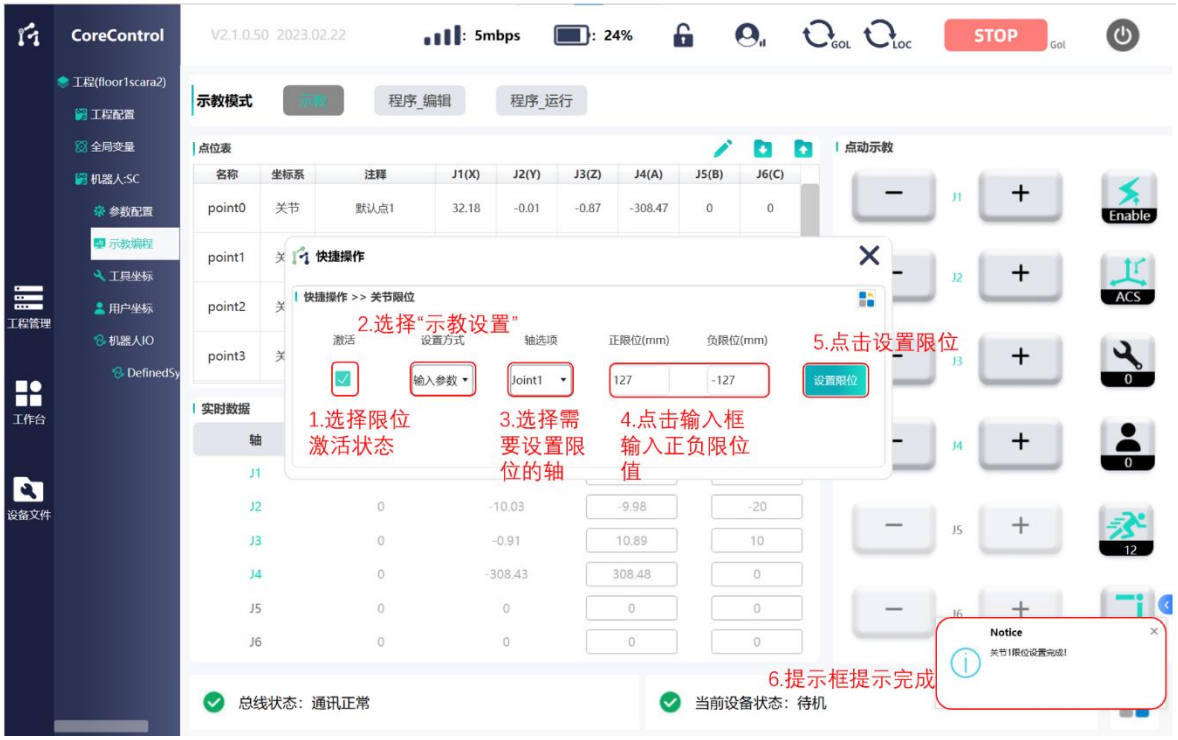


图 73 关节限位设定

3. 超出限位恢复

关节限位为软件限位，与轴光电限位性质不同，触发关节限位可通过 FC 软件进行复位后反向移动解除报错。

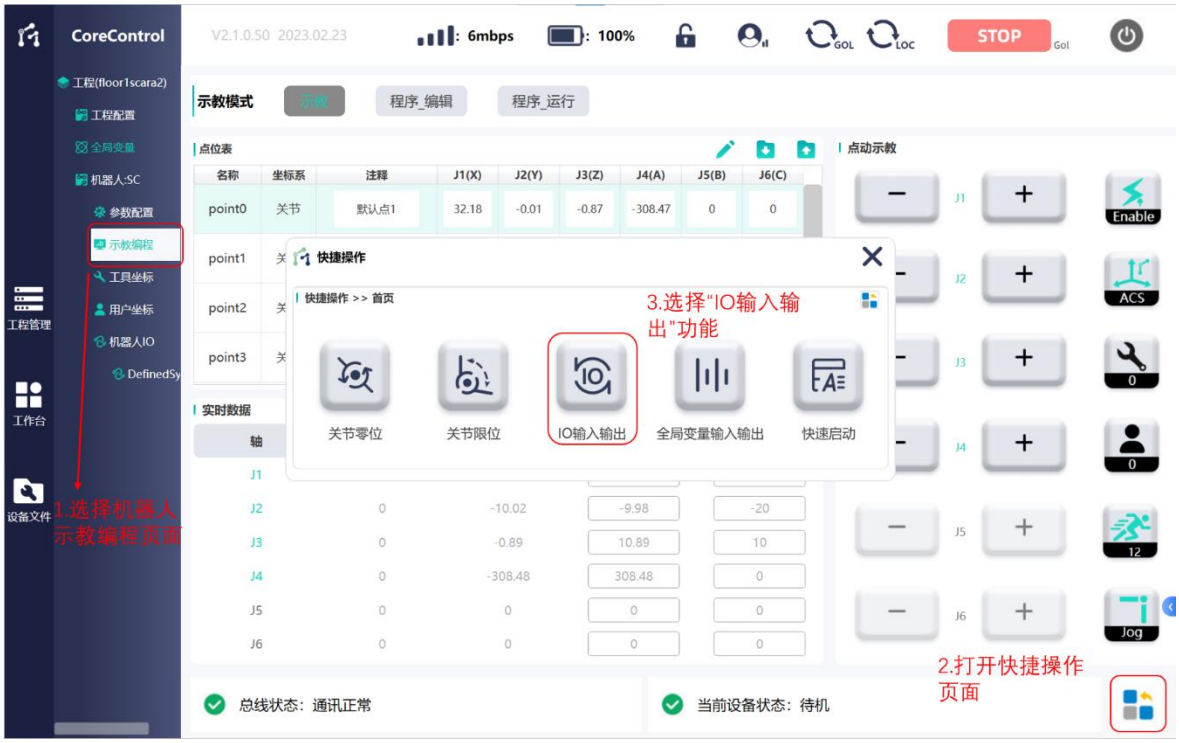
4.8.3 I/O 快捷输入输出

I/O 快捷输入输出功能用于示教、编程及程序运行环境下，外部 I/O 关联设备缺失无法测试程序合理性或需执行动作以检测设备通讯等情况时使用。

通过 4.4.3 I/O 设备添加及变量关联章节内容所述方式配置 I/O 设备后，于配置界面中相关需要关联至快捷动作栏的 I/O 点后“关联至快捷动作”栏下关联开关开启，如需取消关联即关闭开关：

图 74 I/O 关联快捷输入输出

在机器人示教界面，打开快捷操作页面，选择“IO 输入输出”功能；



已关联 I/O 点于 I/O 快捷输入输出显示，点击“开关”按键，即对该 I/O 点进行强制输入输出高低电平操作，开关开启即为“True”，反之为“False”：

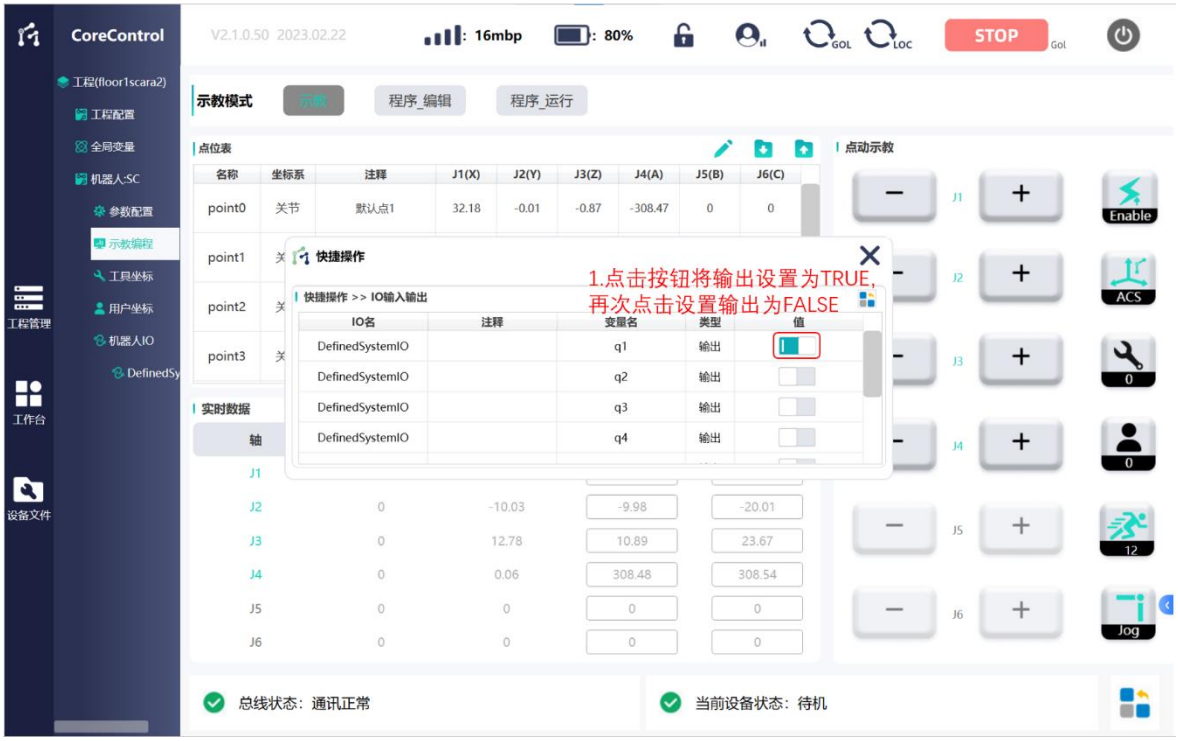


图 75 I/O 快捷输入输出

不同机器人示教编程界面内只显示相应下挂的 I/O 内选择关联的点位，当需切换不同机器人下挂 I/O 以触发相应功能时，需切换机器人以切换 I/O。当前全局 I/O 不支持快捷输入输出。

注意

程序自动运行状态下执行I/O快捷输入输出需确保输入输出信号及点位符合当前程序判断，否则可能导致干扰程序步判断执行问题 。

4.8.4 全局变量输入输出

全局变量快捷输入输出功能用于示教、编程及程序运行环境下，外部关联设备缺失无法测试程序合理性或需执行动作以检测设备通讯等情况时使用。

通过 4.4.4 全局变量添加章节内容所述方式添加全局变量后，于配置界面中相关需要关联至快捷动作栏的全局变量点后“关联至快捷动作”栏下关联开关开启，如需取消关联即关闭开关：

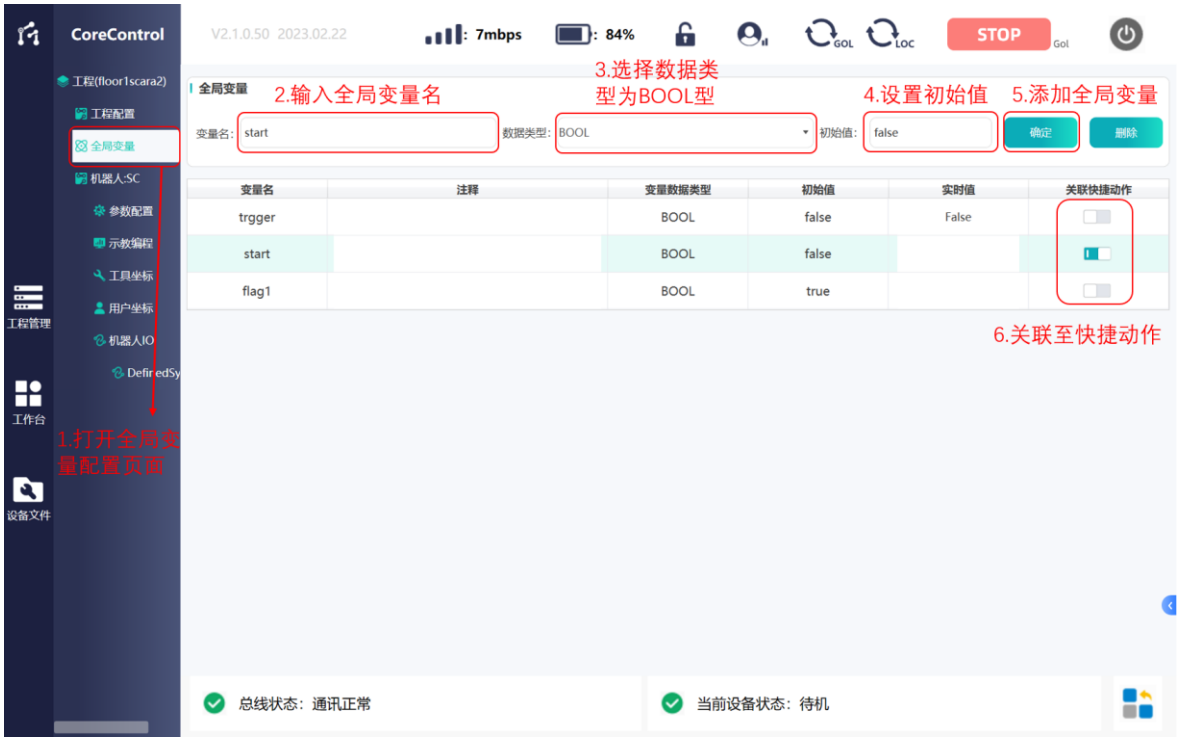
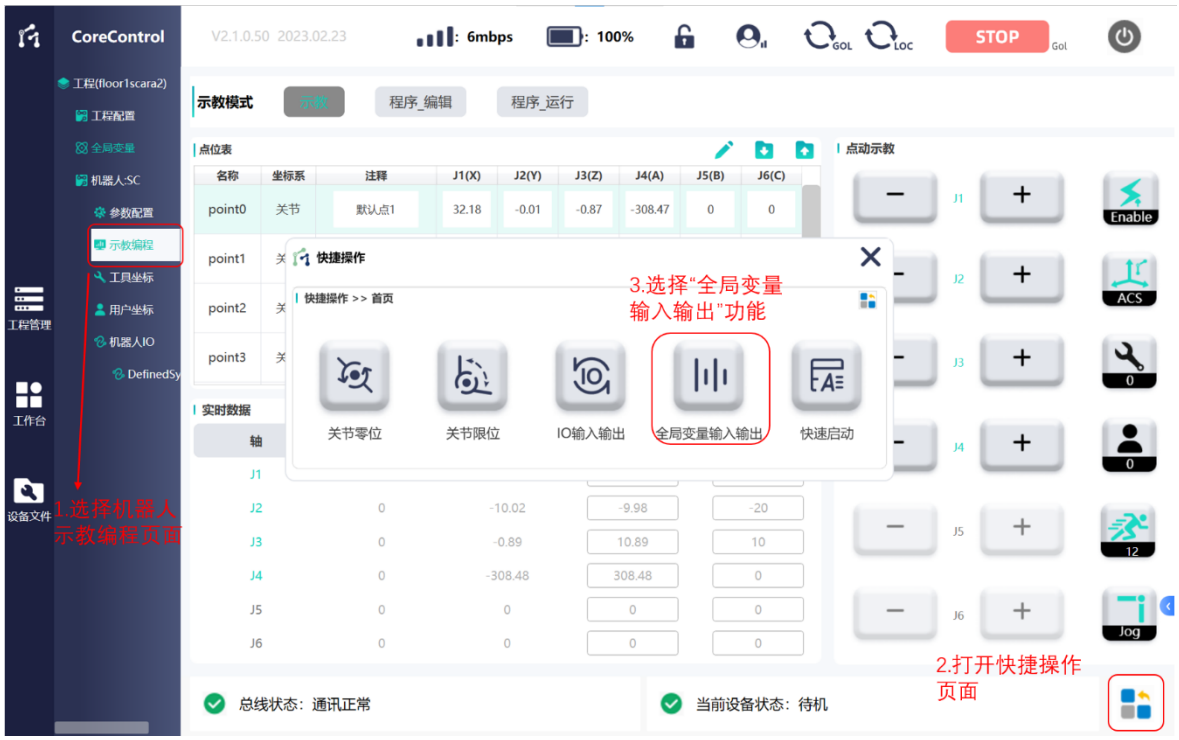


图 76 全局变量关联快捷输入输出

已关联全局变量点于全局变量快捷输入输出显示，当前仅 BOOl 类型全局变量支持关联至快捷输入输出。

在机器人示教界面，打开快捷操作页面，选择“全局变量输入输出”功能；



点击“开关”按键，即对该全局变量点进行强制输入输出高低电平操作，开关开启即为“True”，反之为“False”：

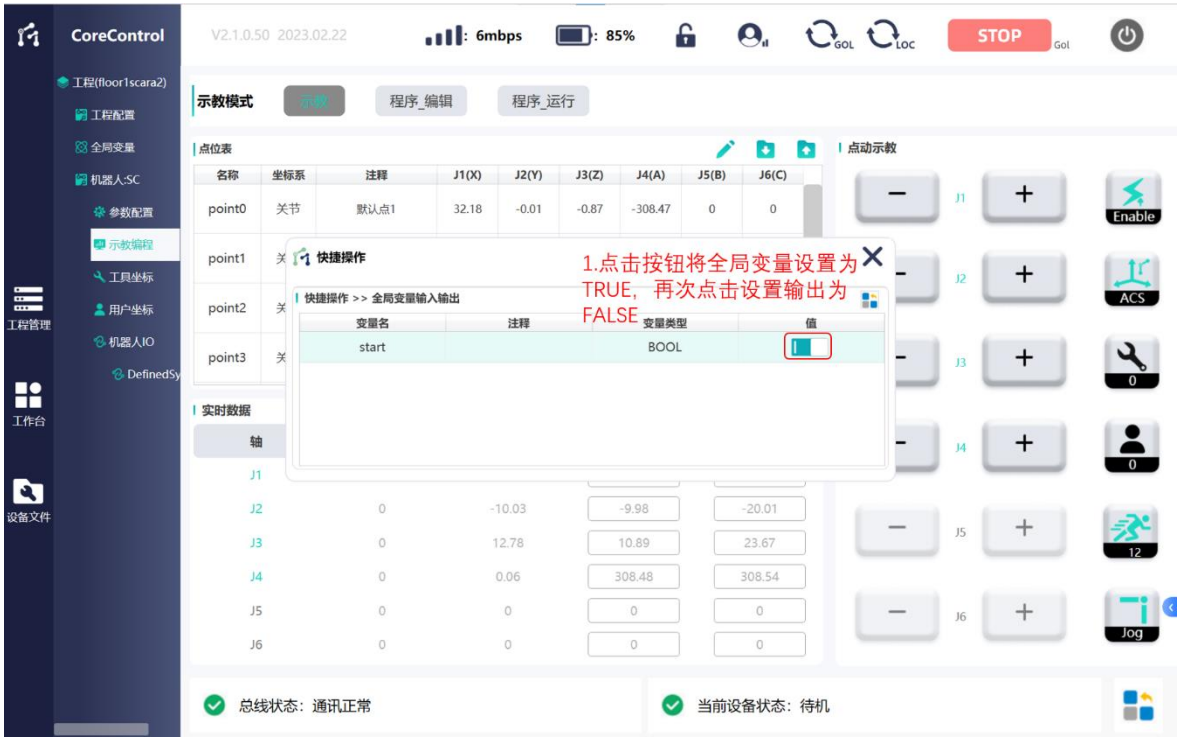


图 77 全局变量快捷输入输出

全局变量为工程内所有机器人共用，快捷输入输出栏内显示的变量不随机器人的切换而切换。

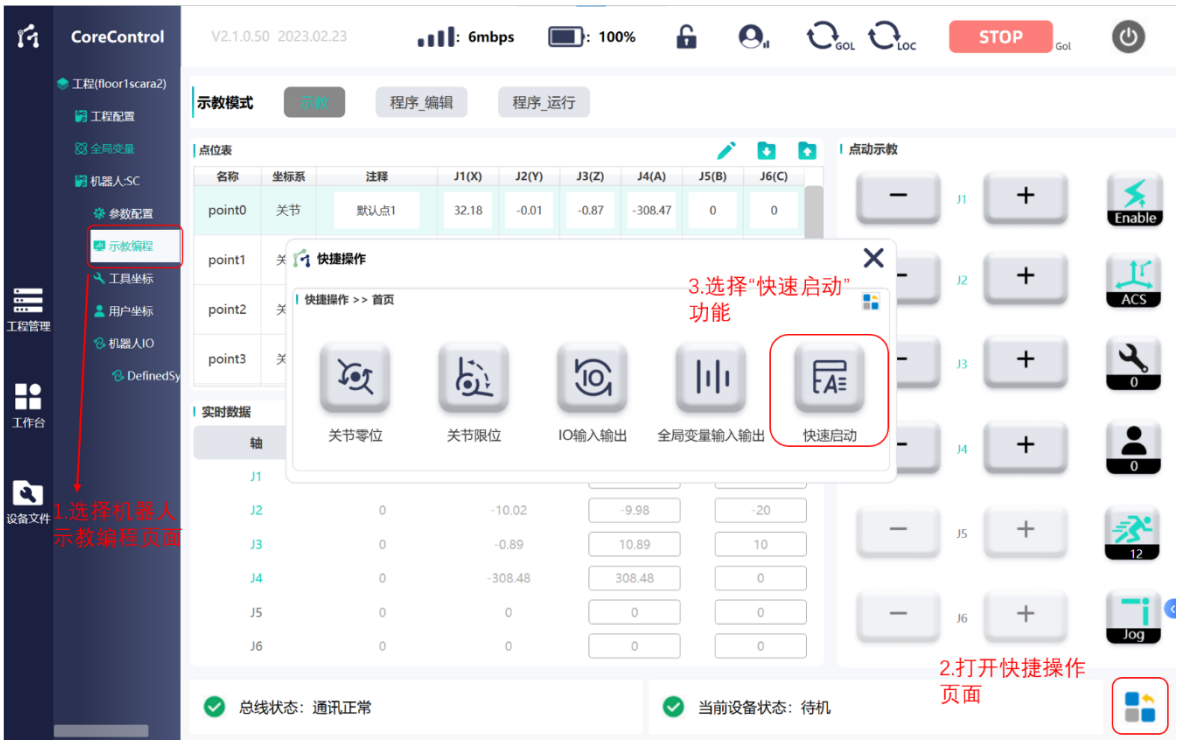
注意

程序自动运行状态下执行全局变量快捷输入输出需确保输入输出信号符合当前程序判断，否则可能导致干扰程序步判断执行问题 。

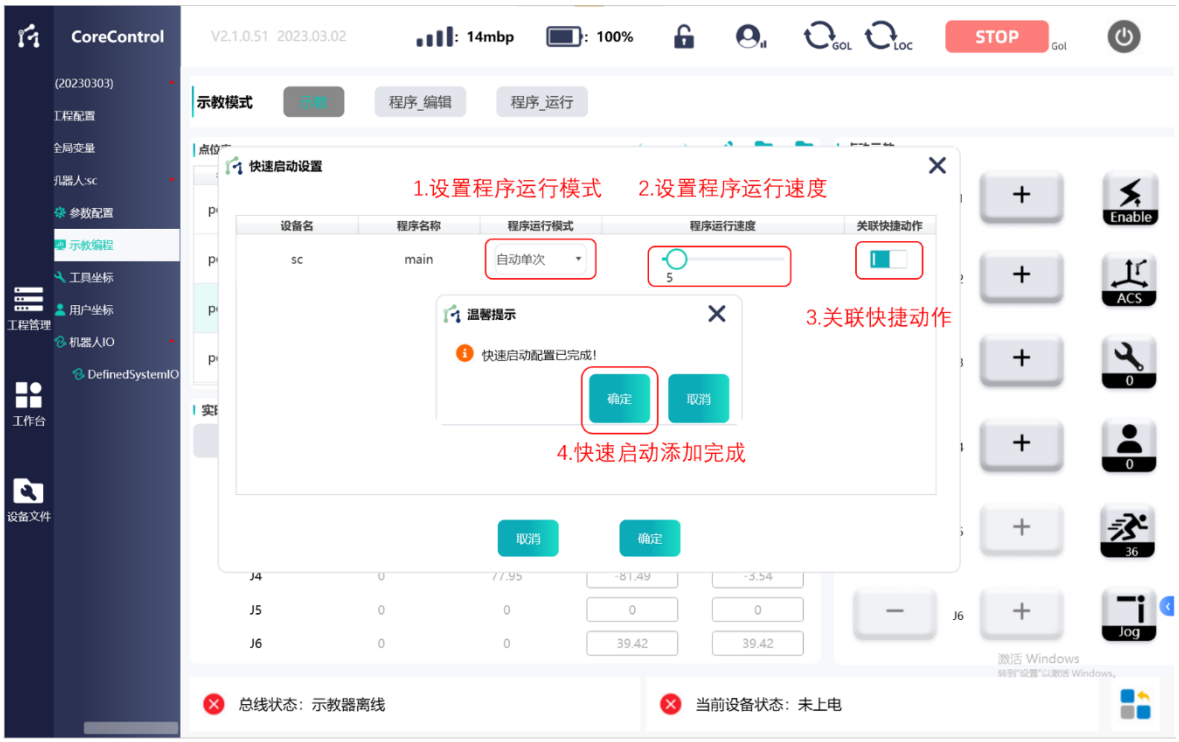
4.8.5 快速启动

快速启动功能适用于：程序编辑并调试完成后，无需连接示教器，通过外部输入按钮启动机器人程序。

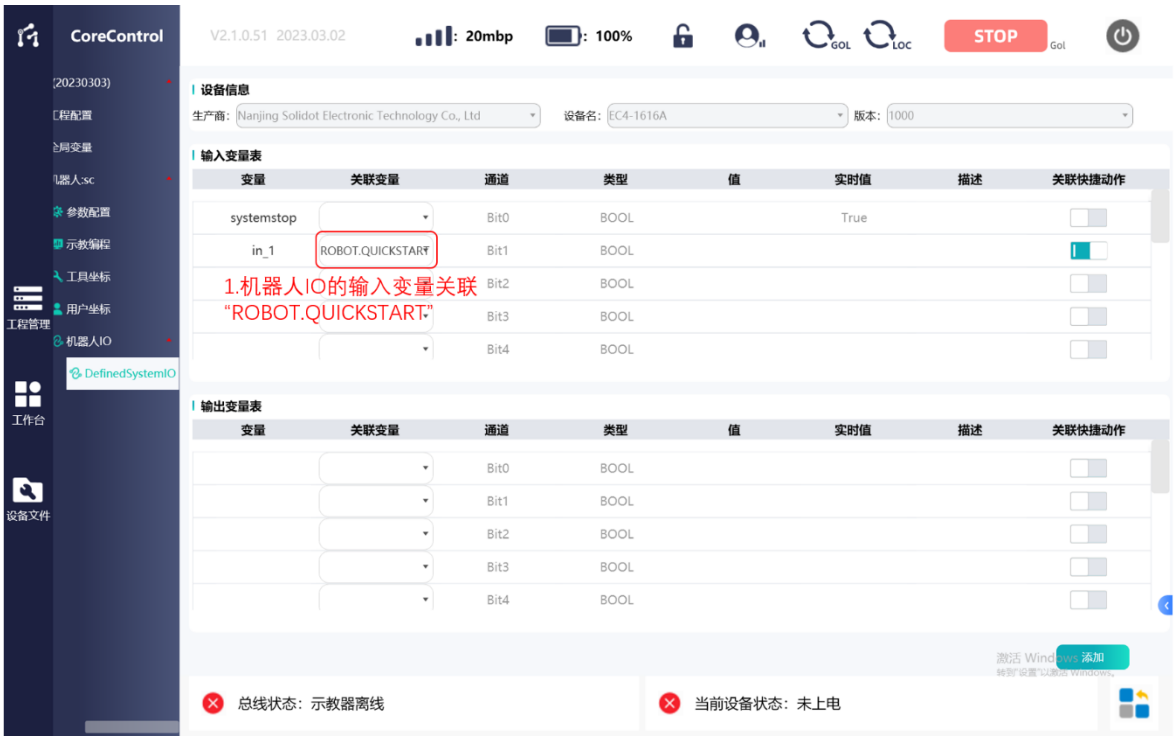
在机器人示教界面，打开快捷操作页面，选择“快速启动”功能；



在弹出的“快速启动”设置窗口，可以选择程序的运行模式，程序的运行速度，以及是否关联快捷动作；



关联快捷动作前，需要在机器人 IO 中，添加的输入中关联“ROBOT.QUICKSTART”动作，未关联此动作，则快速启动无法使用。


CoreControl
V2.1.0.51 2023.03.02
20mbp
100%
STOP
设备信息
生产商: Nanjing Solidot Electronic Technology Co., Ltd
设备名: EC4-1616A
版本: 1000
输入变量表
变量 关联变量 通道 类型 值 实时值 描述 关联快捷动作
systemstop Bit0 BOOL True
in_1 ROBOT.QUICKSTART Bit1 BOOL
Bit2 BOOL
Bit3 BOOL
Bit4 BOOL
1.机器人IO的输入变量关联
“ROBOT.QUICKSTART”
输出变量表
变量 关联变量 通道 类型 值 实时值 描述 关联快捷动作
Bit0 BOOL
Bit1 BOOL
Bit2 BOOL
Bit3 BOOL
Bit4 BOOL
添加
总线状态：示教器离线
当前设备状态：未上电
工程管理
工作台
设备文件
机器人IO
DefinedSystemIO
参数配置
示教编程
工具坐标
用户坐标

4.9 程序编辑及运行

FC 软件采用离线编译式编程方式，所有程序涉及点位数据、变量运用需先行执行 4.4 章节 I/O 设备添加及变量关联工作、4.6 章节机器手动示教工作。程序编辑时先关数据为调用方式，程序编辑完成及修改后需进行编译操作，以将程序编译下发至控制器 runtime（下位机）执行、监控。

4.9.1 程序模版说明

点击“程序编辑”按键，切换机器人示教编程界面至程序编辑界面。

图 78 程序编辑界面

FC 软件编程模板由两部分组成，分别为局部变量定义部分、程序主体部分。

局部变量定义部分用于定义当前编程界面对应机器人程序内部使用的变量，仅可由本台机器人程序调用。

程序主体部分由系统标识符“MAIN”/“ENDMAIN”及程序文本主体组成，其中系统标识符不可删除。程序编译后运行即运行程序主体部分。

图 79 程序编辑区说明

程序模板支持导出至 TXT 文档由其他设备编辑并导入。

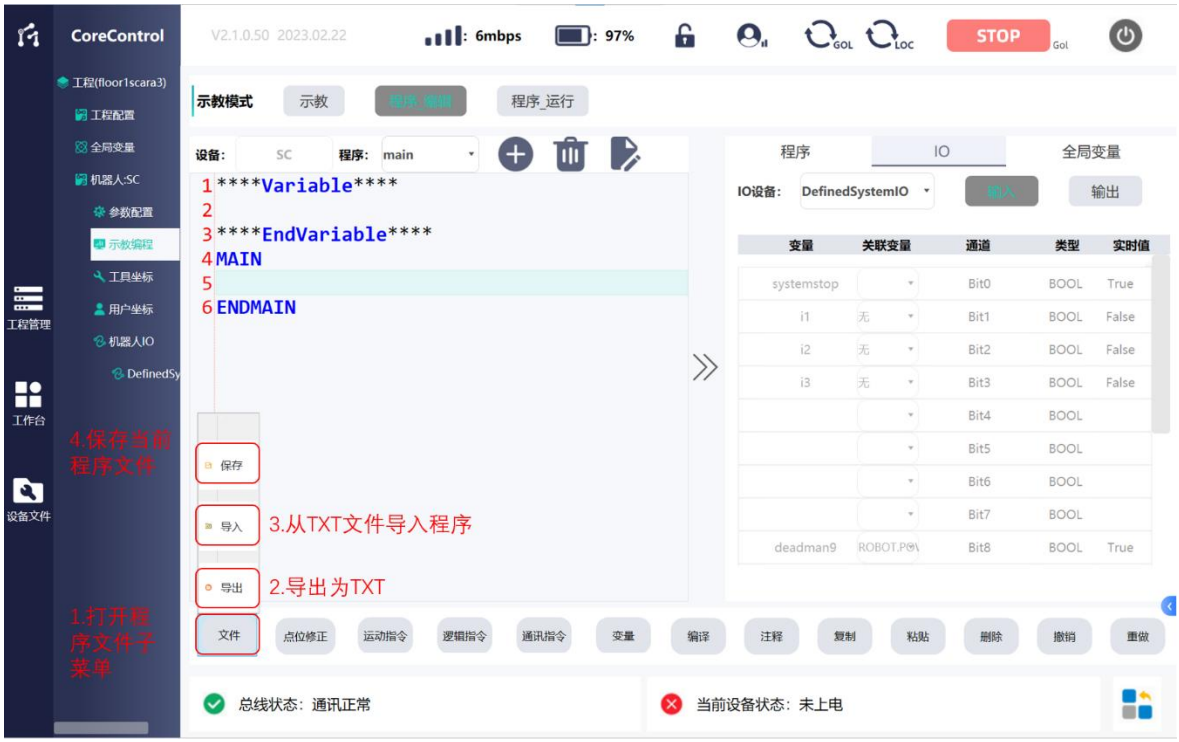


图 80 程序导入及导出

4.9.2 变量定义

程序编程所需全局变量由 4.4.4 章节所述全局变量添加内容定义，非编程界面定义，此处做调用。

局部变量定义通过于变量定义部分定义，手动点击选中变量定义区空白行，点击“变量”按键，于弹窗中输入需定义变量名及初始值（可初始值不赋值），点击“确定”即完成变量定义添加。

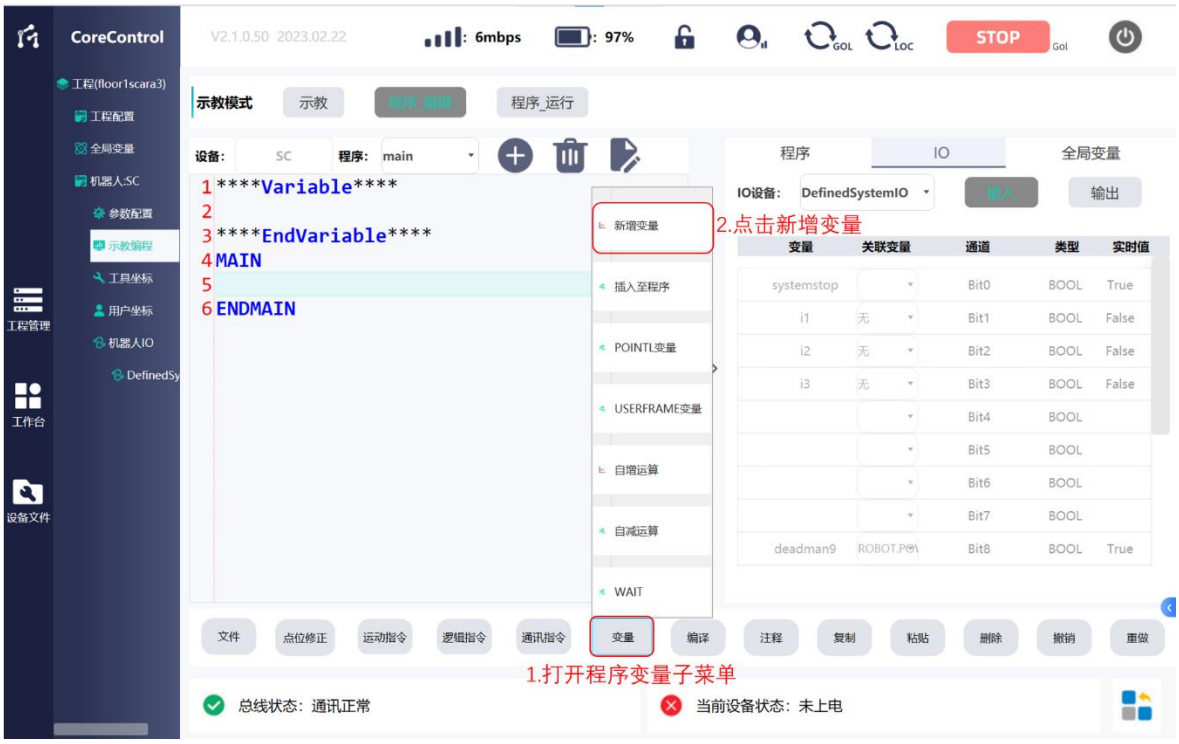


图 81 变量定义选项

图 82 变量定义操作窗

4.9.3 程序文本编辑

程序编辑窗做程序编辑使用，编程基本流程为选中指定行（步）-点击相应指令窗-输入相关参数确定插入程序文本-编译。

4.9.3.1 程序步选中及程序步内容选中

用户可以在程序任意空白处双击或点击左侧菜单栏“键盘”按钮，唤出系统键盘。

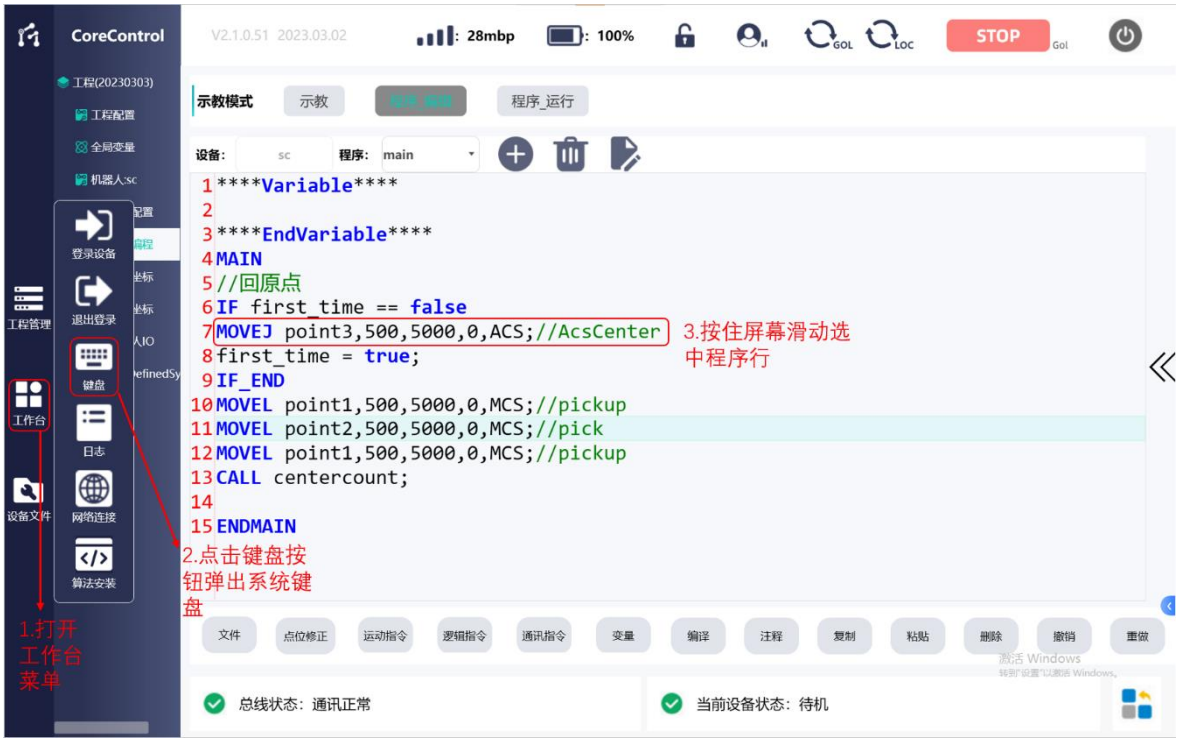


图 83 程序步单击选中操作

双击程序行右侧空白位置时，打开系统键盘；

双击程序行中特定语句时，可以选中程序行中具体元素，如 point 点或速度等；

单击程序行，并滑动，可以选中程序行需要修改的部分；

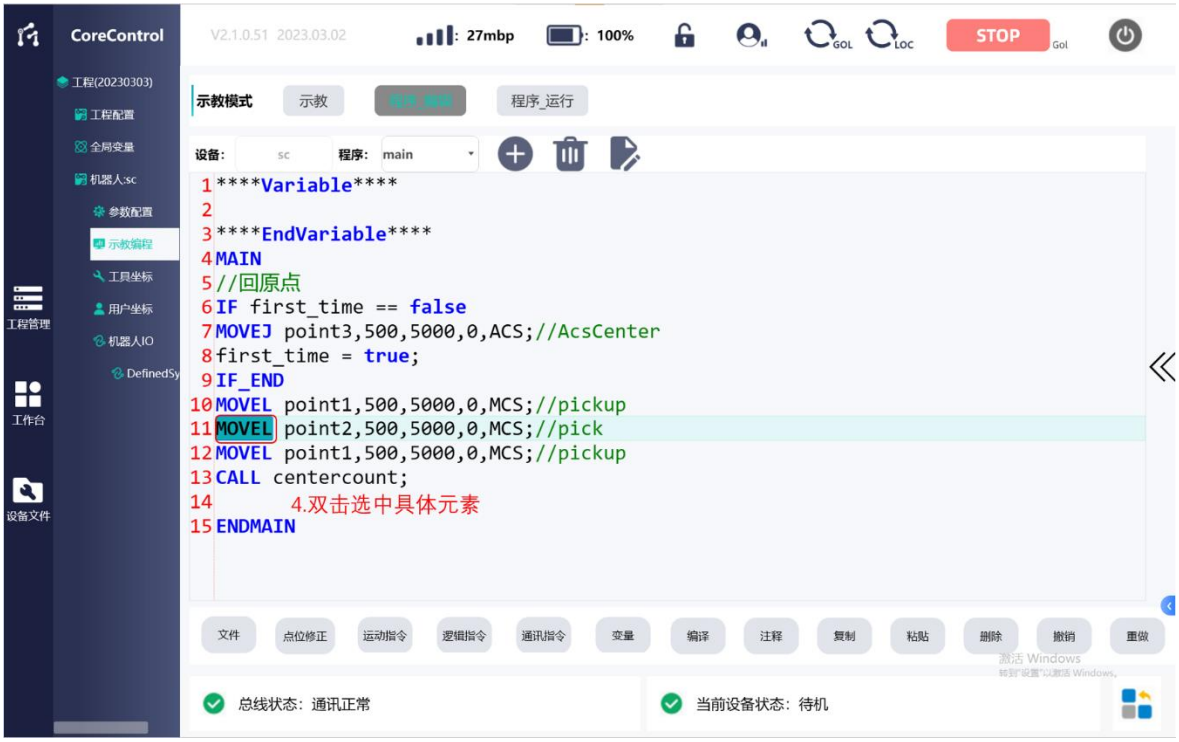


图 84 程序内容双击选中操作

4.9.3.2 程序步复制、粘贴及删除

用户可对程序步进行复制、粘贴和删除操作，需要单击并拖动选中程序某一步，待其被绿色背景覆盖后，再使用下方“复制、粘贴、删除”等操作按钮进行操作。

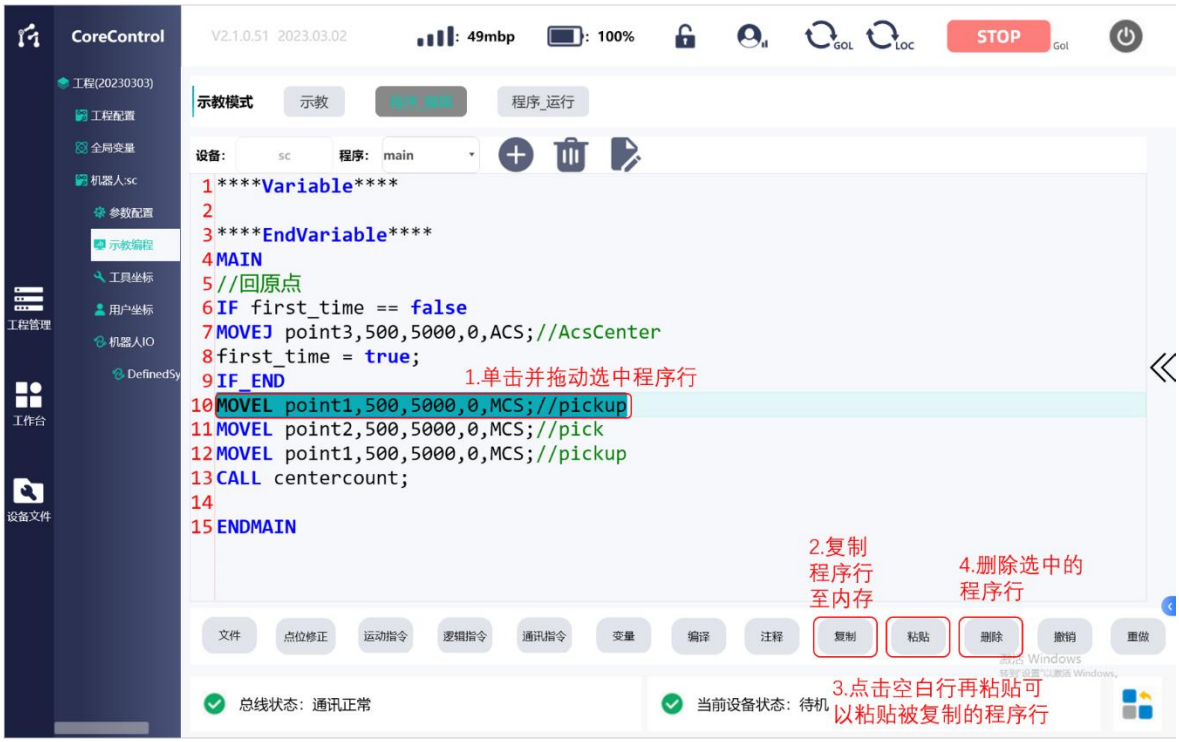


图 85 程序步复制粘贴

4.9.3.3 程序步注释

为增加程序的可读性，用户可以将程序主体中的任意行进行注释。使用注释时，可以选择“注释整行”或“注释说明”。

若程序中某一步不想被执行，可以选择“注释整行”。

图 86 程序步注释整行

若想添加说明语句，可以选择“注释说明”。

图 87 程序步内容注释说明

4.9.3.4 指令窗介绍

示教器提供了 “运动指令”和“逻辑指令”供用户选择；

“运动指令”中用户可以选择“MOVEL”、“MOVEJ”、“MOVER”、“MOVEC”、“BLEND”等指令，具体指令的功能和使用方法参考 5.2 章节运动指令内容。

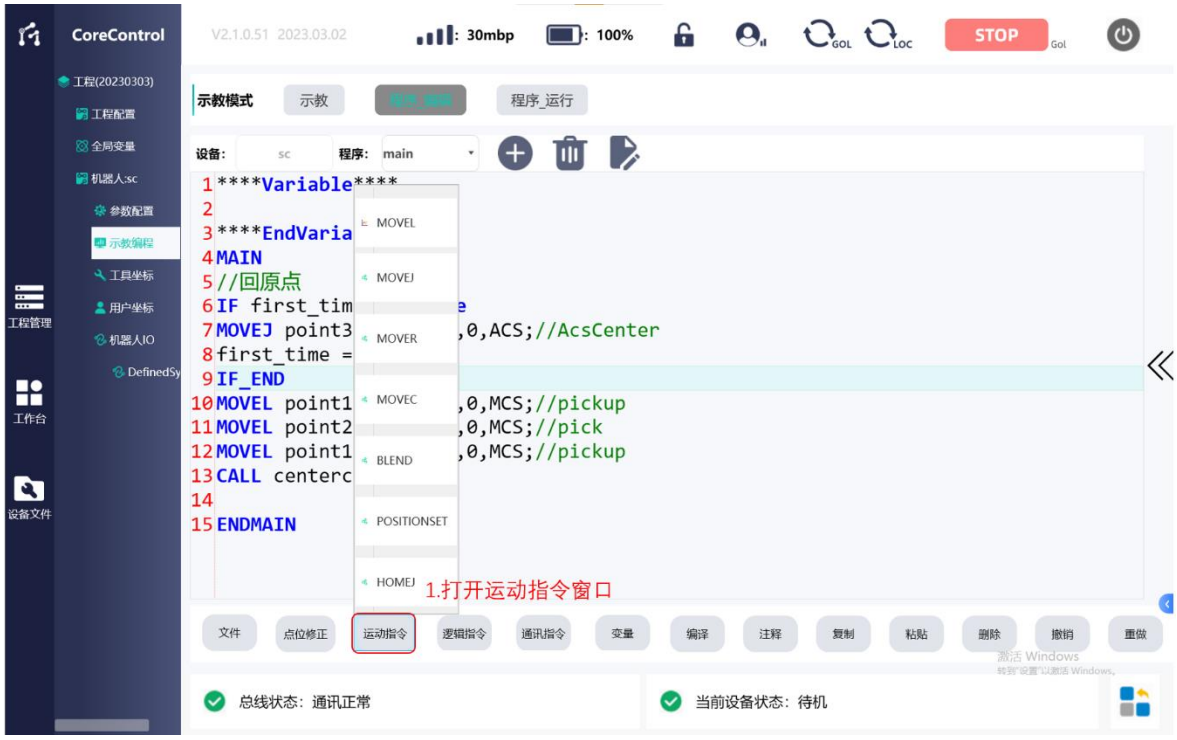


图 88 运动指令插入按键

“逻辑指令”中用户可以选择“IF”、“LOOP”、“While”、“SETDO”、“WAITDI”、“WAIT”、“DELAY”、“DELAYSET”等指令，具体指令的功能和使用方法参考 5.3 章节逻辑指令内容。

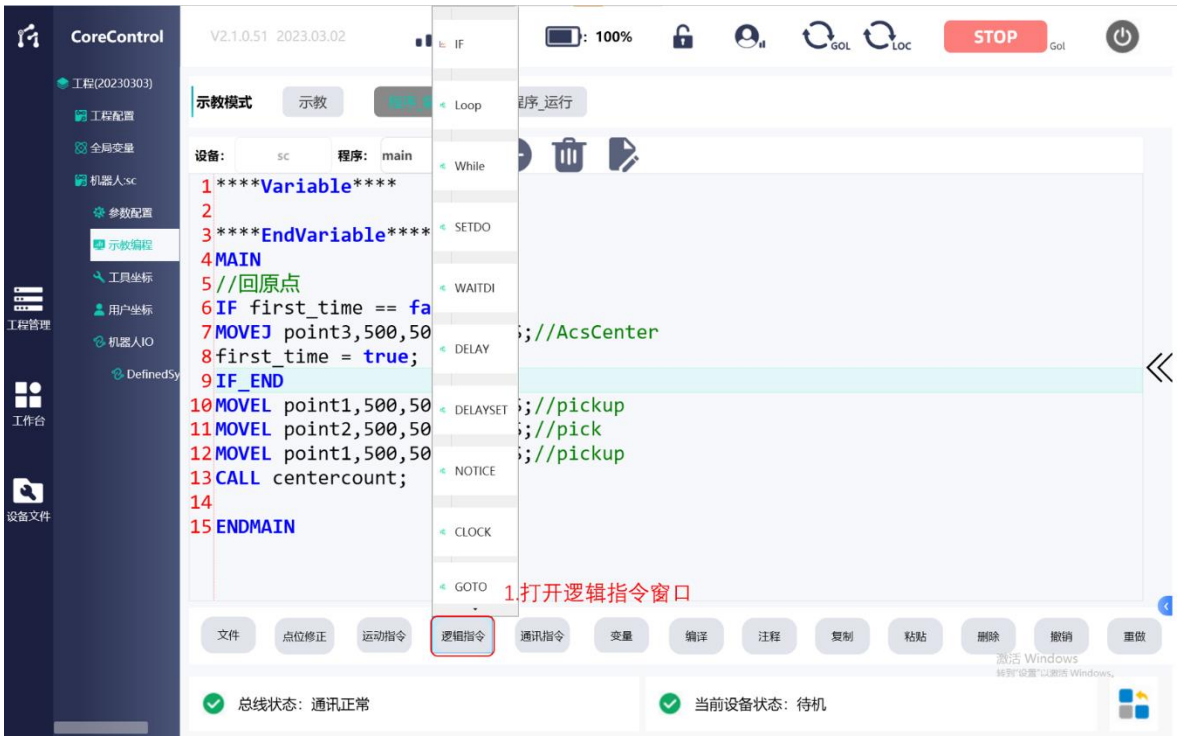


图 89 逻辑指令插入按键

4.9.3.5 程序编译

用户程序编写完成后，需使用“编译”功能按钮可以将程序主体进行编译，并检查其中的语法和逻辑问题，有问题的部分将会在弹出菜单中弹出。

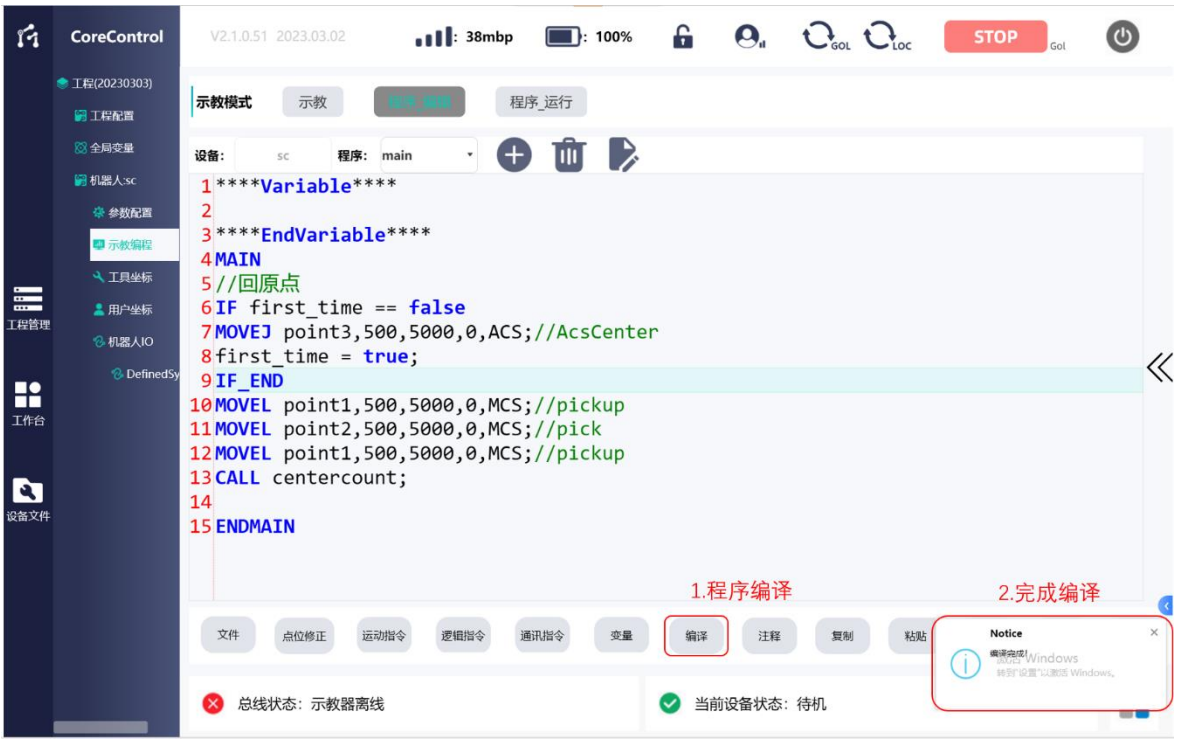


图 90 程序编译

注意

程序中调用的点位对应数据修改、变量名调整、程序文本修改等涉及程序内容与上次编译后不一致的变动均需对程序进行再次编译。如不执行，运行程序内容将仍为上次编译后内容。

4.9.4 程序运行

程序运行需首先进行程序编译，点击“程序运行”按键，切换机器人示教编程界面至程序运行界面。程序文本运行 跟踪显示窗口以显示程序文本，未进行过程序编译操作的程序运行跟踪窗口未无程序文本显示。

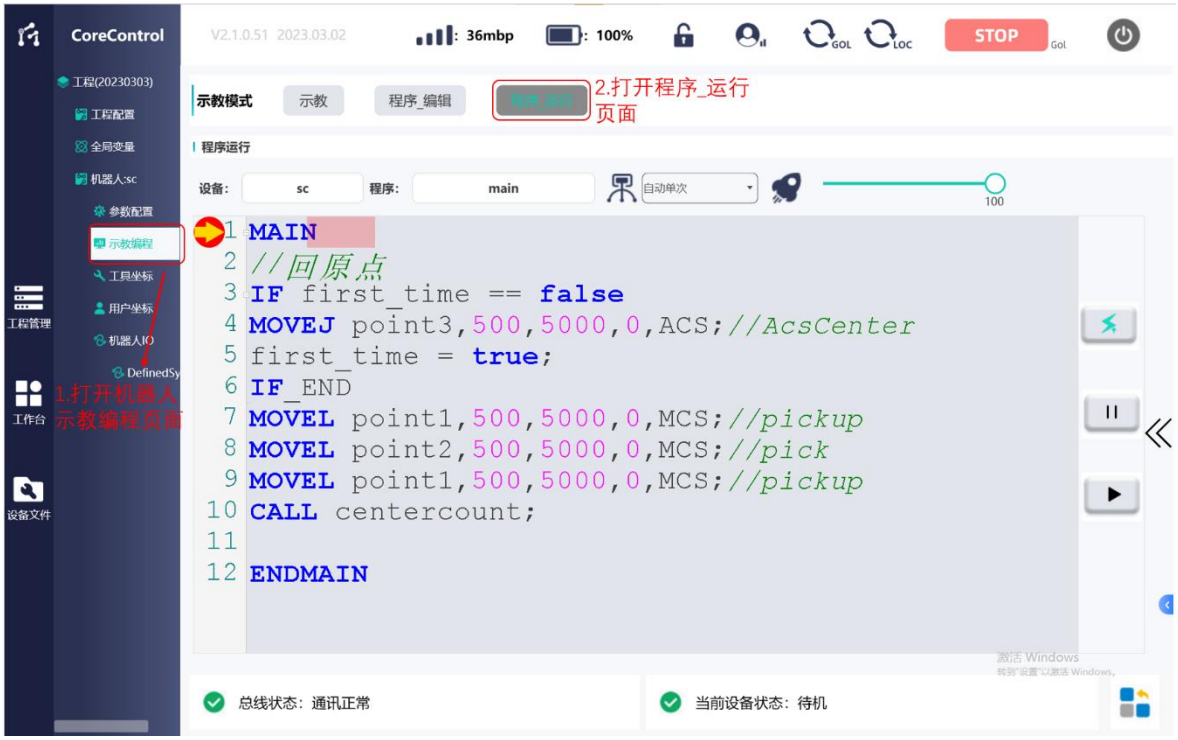


图 91 程序运行窗

4.9.4.1 程序窗口操作

程序运行窗口中，可供用户操作的部分如下，包括“程序运行模式菜单”、“程序运行速度比例条”、“使能、停止、启动”按钮。“程序运行速度比例条”可以控制机器人程序模式下的运行速度，机器人实际的运行速度计算公式为：

*实际运行速度 = 程序语句中速度*运行速度比例/100*

（如：语句中速度为 200，速度比例为 50，则实际运行速度为：200*50/100=100）

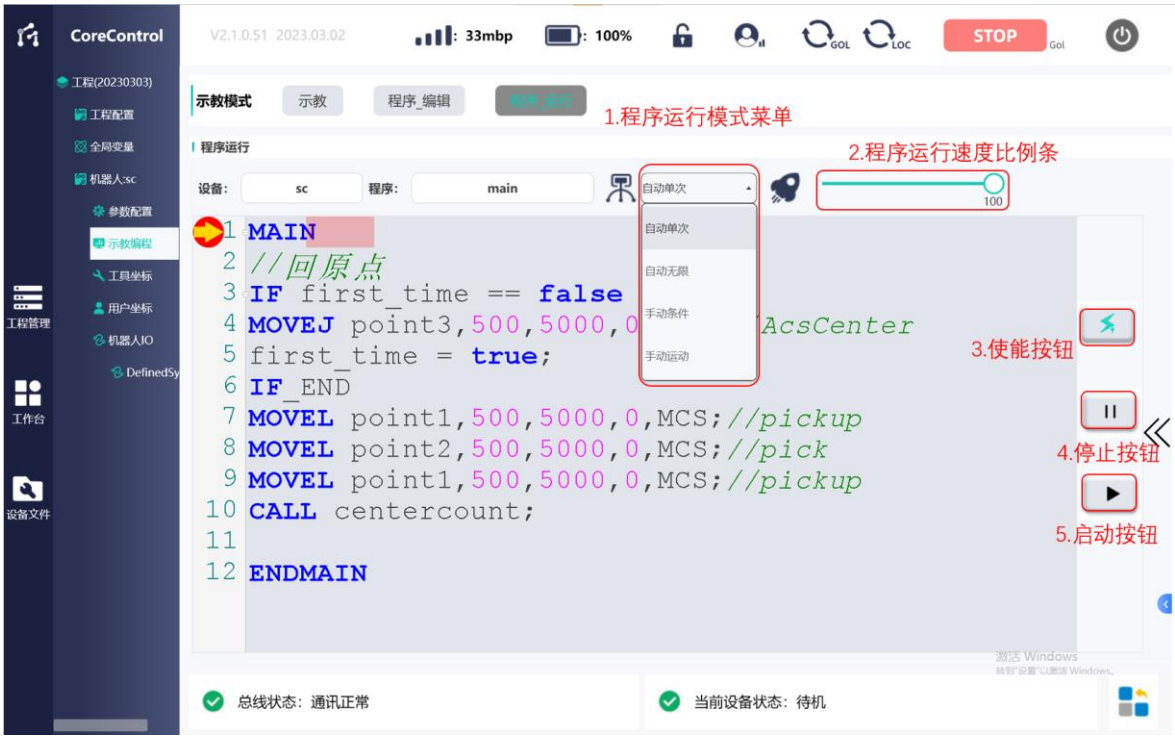


图 92 程序运行模式切换操作

4.9.4.2 程序运行按键操作

“使能按钮”：按下使能按钮后，按键变蓝，此时机器人上电并松开抱闸；再次按下，按钮变灰，此时机器人下电，抱闸夹紧；

“停止按钮”：按下停止按钮后，程序运行状态变为停止；

“启动按钮”：按下按钮，若程序未执行或处于停止状态，程序变为运行状态；

4.9.4.3 程序运行方式

“自动单次”：程序运行模式为此模式时，示教器会由 MAIN 开始向下执行，执行完最后一行指令到 ENDMAIN 时，程序终止；

“自动无限”：程序运行模式为此模式时，示教器会由 MAIN 开始向下执行，执行完最后一行指令到 ENDMAIN 时，程序回到 MAIN 重新开始执行；

“手动条件”：程序运行模式为此模式时，示教器会由指针指向的行开始运行，按住“启动”按钮才会执行当前语句，松开按钮则程序停止，直到再次按下“启动”按钮；且程序中的判断等逻辑语句会正常执行；

“手动动作”：程序运行模式为此模式时，示教器会由指针指向的行开始运行，按住“启动”按钮才会执行当前语句，松开按钮则程序停止，直到再次按下“启动”按钮；且程序中的判断等逻辑语句不会被执行；

5 编程语言

5.1 变量数据类型

5.1.1 基础数据类型

5.1.1.1 BIT

变量说明：

存储位数据，初始状态有两种：0 和 1。支持声明为局部变量。这种数据类型常作为逻辑变量使用，用来表示真、假或是、否等值选择。存储空间为 1 位。如果不关注存储空间的问题，绝大多数情况下可以用 BOOL 型数据来代替。

声明格式：

BIT x = 0(1);

//BIT 变量名称 = 初始状态（0 或者 1）；**//**

5.1.1.2 BOOL

变量说明：

存储布尔型变量，也就是逻辑型变量。存储空间为 8 位。初始状态有两种：TRUE 和 FALSE。支持声明为全局变量和局部变量。语句指令中的条件表达式常用 BOOL 变量的 TRUE 或 FALSE 状态来判断。

声明格式：

BOOL x =FALSE(TRUE);

//BOOL 变量名称 = 初始状态（FALSE 或 TRUE）;**//**

使用示例：

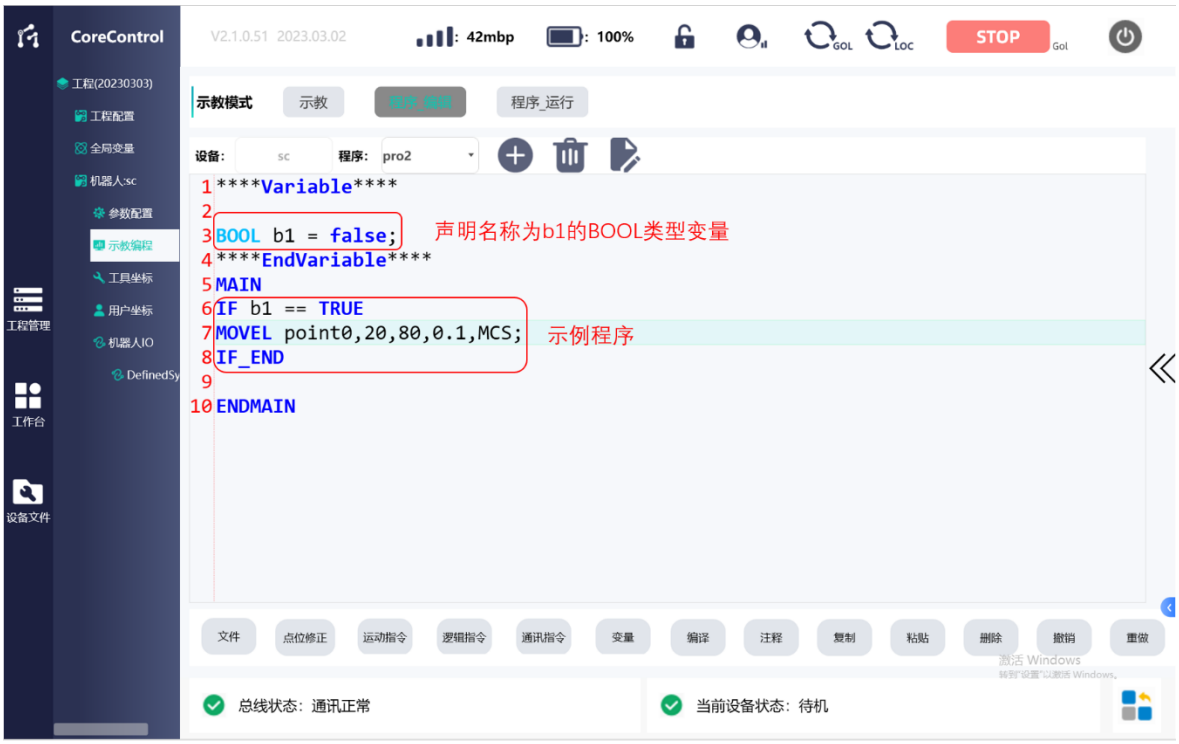


图 93 BOOL 型变量使用示例

程序文本示例：

```
b1 = TRUE;
IF b1 == TRUE
MOVEL point0,20,80,0.1,MCS;
IF_END
```

程序文本说明：

将变量 b1 置位为 TRUE。判断变量 b1 的当前状态是否为 TRUE，如果是则执行 MOVEL 运动指令，机器人 TCP 点以速度为 20，加速度为 80 的直线轨迹运动到点 point0，到达点 point0 后减速到 0。程序结束。

5.1.1.3 BYTE

存储字节型变量，占据一个字节存储 8 位无符号数，存储范围为 0 到 255。支持声明为全局变量和局部变量。

5.1.1.4 INT

变量说明：

存储 32 位有符号整数变量，存储范围为-2147483648 到+2147483647。支持声明为全局变量和局部变量。整数在计算机中用补码来表示和存储，有符号整型的最高位为符号位，符号位为"0"表示正整数，符号位为"1"表示负整数。多用于计数，变

量计算等场合。

声明格式：

INT x = xx;

//INT 变量名称 = 初始值（整数）;**//**

使用示例:

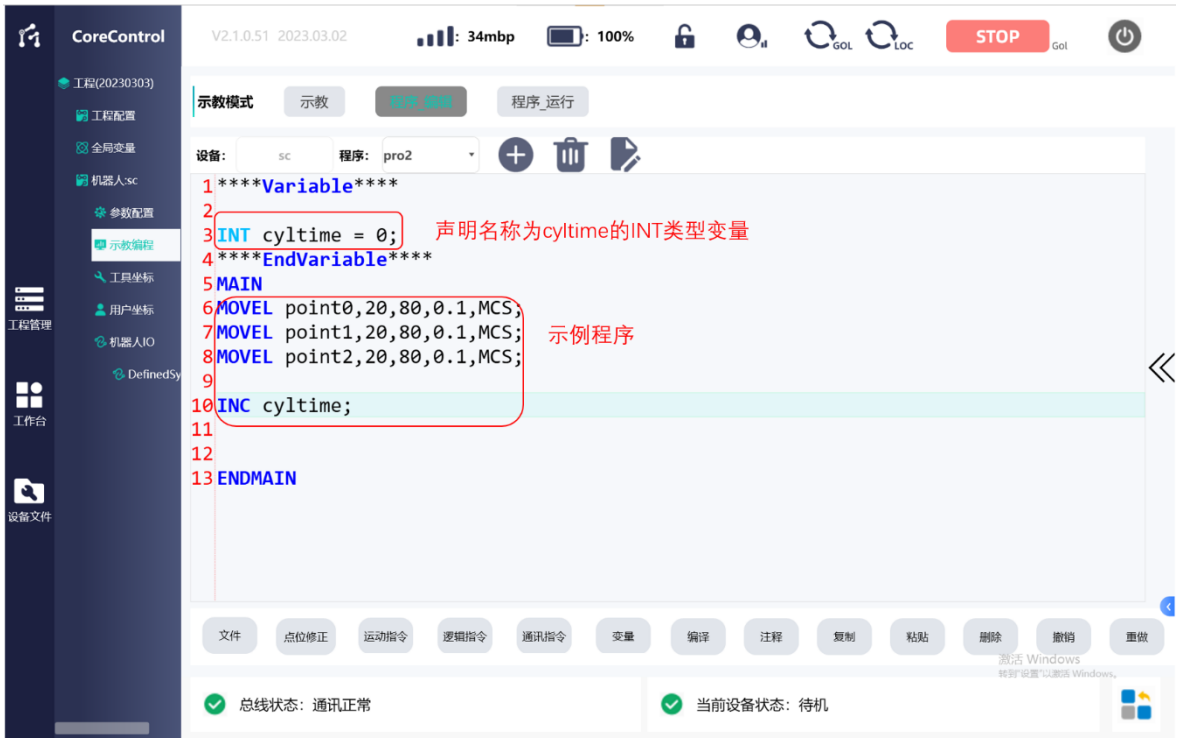


图 94 INT 型变量使用示例

程序文本示例：

MOVEL point0,20,80,0.1,MCS;

MOVEL point1,20,80,0.1,MCS;

MOVEL point2,20,80,0.1,MCS;

INC cyltime;

程序文本说明：

声明的变量 cyltime 用于计数程序执行次数（或是产量计数等）。机器人以直线轨迹依次运动到点 point0 ，point1 ，point2。然后使用 INC 指令将变量 cyltime 的值加 1。结束程序此次循环。程序后续每执行一次，变量 cyltime 的值就会增加 1。自增指令 INC 指令说明参见 5.3.4 INC。

5.1.1.5 LINT

变量说明：

存储 64 位有符号整数变量，存储范围为 -9223372036854775808 到

+9223372036854775807。支持声明为局部变量。

声明格式：

LINT x = xx;

//LINT 变量名称 = 初始值（整数）**;//**

使用示例：

与 INT 型变量基本相同，唯一不同点是 LINT 型的数据范围比 INT 型更大。

5.1.1.6 REAL

变量说明：

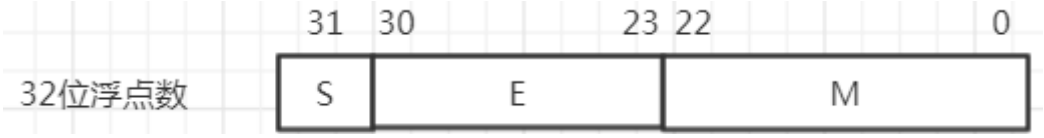
存储浮点（实）数变量，存储空间为 32 位。存储范围为-3.402823e+38 到 -1.175495e-38 ，±0.0，+1.175495e-38 到 +3.402823e+38。支持声明为全局变量和局部变量。REAL 类型变量多用于计算，存储数据等。

数据类型 REAL 的操作数由以下三部分组成：

1. 符号：该符号由第 31 位的信号状态确定。第 31 位的值可以是“0”（正数）或“1”（负数）。

2. 以 2 为底的 8 位指数：该指数按常数增加（基值 +127），因此其范围为 0 ～ 255。

3. 23 位尾数：仅显示尾数的小数部分。尾数为标准化的浮点数，其整数部分始终为 1，且不会保存。



如图符号位 S，指数 E 和尾数 M。

处理 REAL 数据类型时会精确到 6 位数。

输入示例：1.0e-5 ， 1.34 。

声明格式：

REAL x = x;

//REAL 变量名称 = 初始值（实数）;//

使用示例：

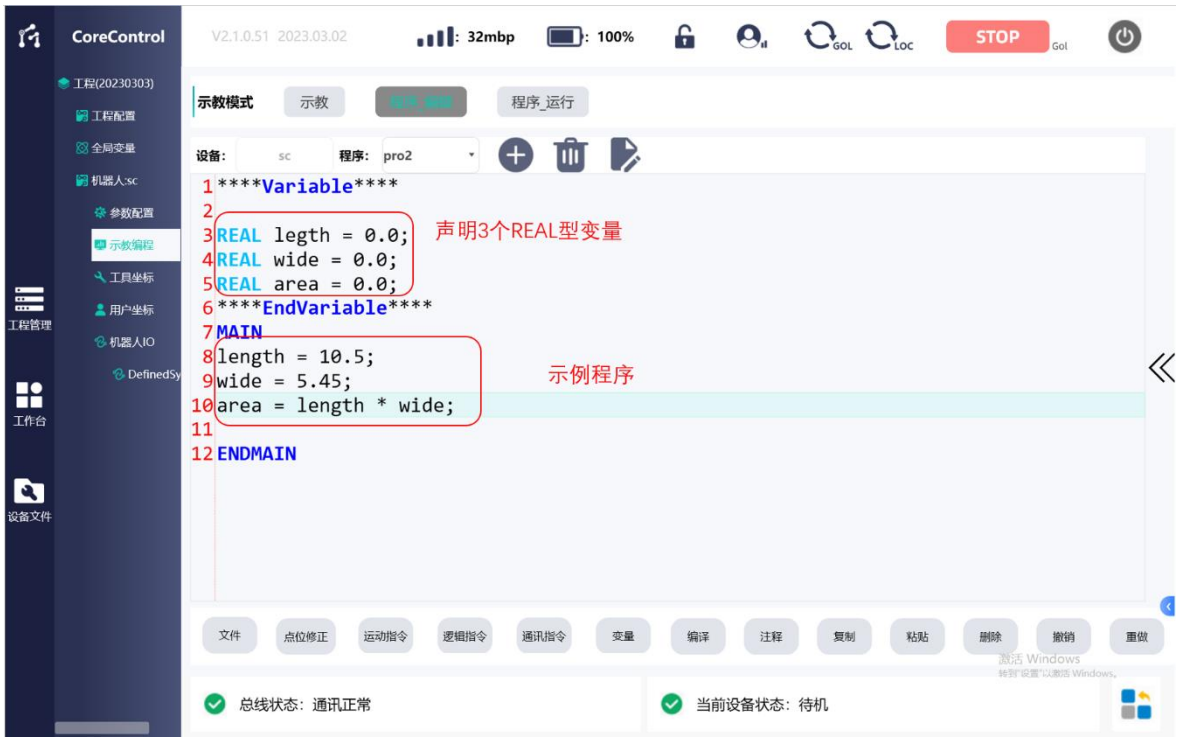


图 95 REAL 型变量使用示例

程序文本示例：

```
length = 10.5;
wide = 5.45;
area = length * wide;
```

程序文本说明：

声明三个 REAL 型变量分别表示矩形的长、宽和面积。并将 length（长）赋值为 10.5（单位未指定），wide（宽）赋值为 5.45。变量赋值可以在声明时直接赋值为实际值，也可以在程序中赋值为实际值。此处为在程序中赋值。然后计算长乘以宽得到矩形的面积，即变量 area 的值（单位未指定）。

5.1.1.7 LREAL

变量说明：

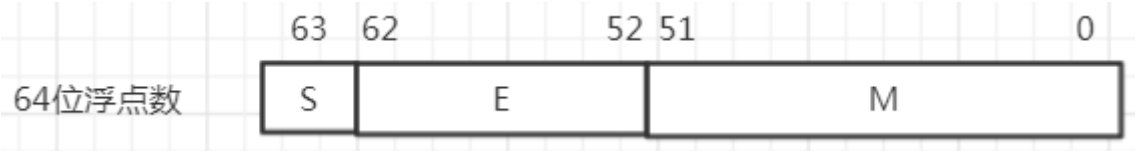
存储长实数变量，存储空间为 64 位, 存储范围为 -1.7976931348623157e+308 到 -2.2250738585072014e-308 ， ± 0.0 ， +2.2250738585072014e-308 到 +1.7976931348623157e+308。支持声明为局部变量。与 REAL 不同的是存储空间不同，记录的数据值精度也不一样。

数据类型 LREAL 的操作数由以下三部分组成：

1. 符号：该符号由第 63 位的信号状态确定。第 63 位的值可以是“0”（正数）或“1”（负数）。

2. 以 2 为底的 11 位指数：该指数按常数增加（基值 +1023），因此其范围为 0 ～ 2047。

3. 52 位尾数：仅显示尾数的小数部分。尾数为标准化的浮点数，其整数部分始终为 1，且不会保存。



如图符号位 S，指数 E 和尾数 M。

处理 LREAL 数据类型时会精确到 15 位数。

声明格式：

LREAL x = xx;

//LREAL 变量名称 = 初始值（长实数）;**//**

使用示例：

与 REAL 型变量基本相同，唯一不同点是取值范围（或精度）不同。

5.1.1.8 STRING

变量说明：

存储字符串变量，一个 STRING 数据类型的变量可以包含任何字符。如果未定义大小，则默认情况下分配 80 个字符。通常，字符串变量不限制字符串长度。但是字符串处理函数（如 STR_SPLIT）限制处理 1~255 之间的长度。声明字符串变量时尽量不要超过 255 个字符。STRING 变量类型可声明为全局变量和局部变量。常用于存储通讯中交互的数据。如 TCP 通讯中发送和接收的数据。

声明格式：

STRING x = xx;

//STRING 变量名称 = 初始字符串;**//**

（初始值也可不指定）

使用示例：

参见 5.5.4 TCP 通讯示例。

5.1.1.9 STRINGLIST

变量说明：

存储字符串数组变量。数组格式为 ARRAY[0..99] OF STRING。常用于存储一段长字符串经过切割处理后得到的字符串数组。常用于存储通讯中接收字符串经过解析处理后的字符串组。

声明格式：

STRINGLIST x;

STRINGLIST 变量名称;

使用示例：

参见 5.5.4 TCP 通讯示例。

5.1.1.10 CLOCK

变量说明：

计时器变量，用于和计时指令 CLOCK 搭配来实现计时功能。常见使用场景有：计算节拍；超时报警等。

声明格式：

CLOCK x ;

//CLOCK 计时器名称;**//**

使用示例：

参见 5.3.14 CLOCK 指令程序示例。

5.1.2 点位数据类型

5.1.2.1 POINTL

变量说明：

存储机器人 TCP 位置和姿态的点位数据。位置包含 X，Y，Z 三个坐标，姿态包含 A，B，C 三个旋转角度。一般用于声明笛卡尔坐标系下除示教点外的自定义点位数据。例如用于笛卡尔坐标系下示教点的偏移功能，实现视觉定位偏移等功能。支持声明为局部变量。

声明格式：

POINTL x = [x, x, x, x, x, x];

//POINTL 变量名称 = [X 坐标，Y 坐标，Z 坐标，A 姿态，B 姿态，C 姿态];**//**

使用示例：

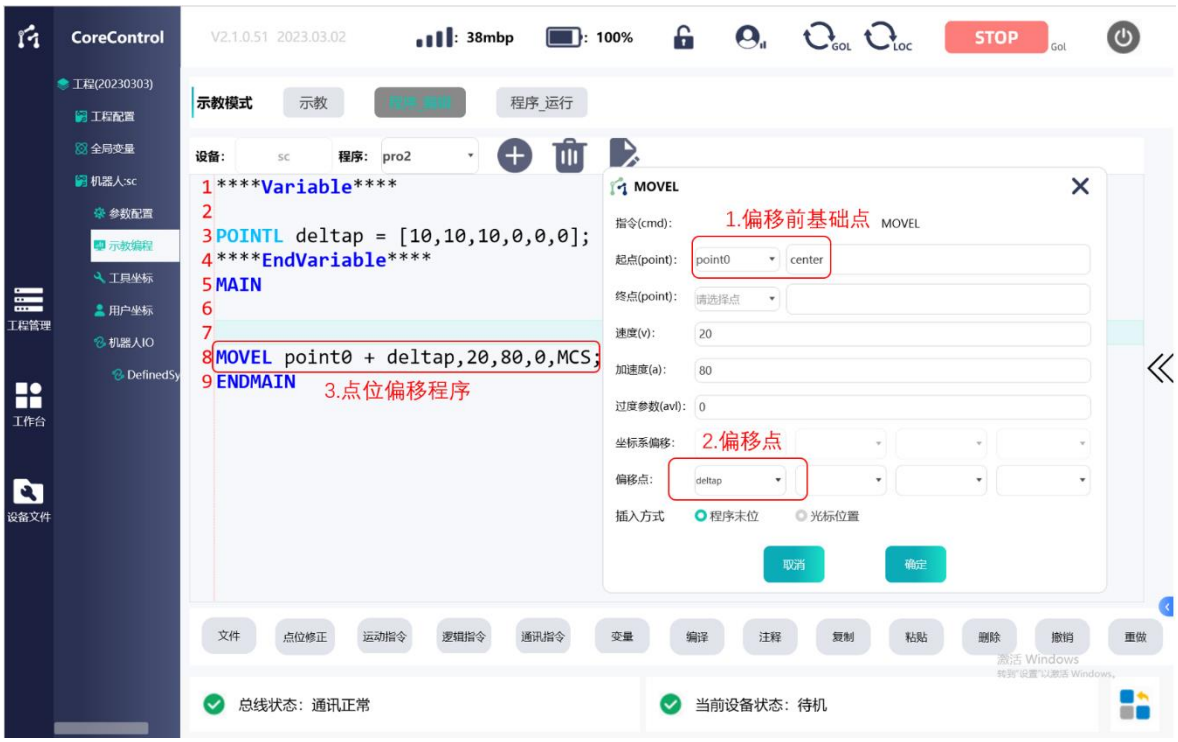


图 96 POINTL 型变量使用示例

程序文本示例：

MOVEL point0 + deltap1,20,80,0.1,MCS;

声明了一个 POINTL 变量 deltap，X，Y，Z 的值为 10，A ,B ,C 的值为 0。使用 MOVEL 指令的点偏移功能使其作为点 point6 的偏移值。即机器人 TCP 以直线轨迹移动到以点 point6 的位置为基础向 X 方向、Y 方向、Z 方向分别偏移 10mm 后得到的偏移之后的点上。

MOVEL 的点偏移功能参见 5.2.1 MOVEL。

5.1.2.2 POINTJ

变量说明：

存储机器人每个关节轴位置的点位数据。点位包含 J1 ~ J6 的角度信息。一般用于声明关节坐标系下除示教点外的自定义点位数据，例如用于关节坐标系下示教点的偏移功能。支持声明为局部变量。

声明格式：

POINTJ x = [x, x, x, x, x, x];

//POINTJ 变量名称 = [X 坐标，Y 坐标，Z 坐标，A 姿态，B 姿态，C 姿态];**//**

使用示例：

使用方式与 5.1.2.1POINTL 变量类型一致。

5.1.3 数据类型转换指令

说明：

数据类型转换指令将初始变量转换为其他数据类型。转换前的数据称为操作数，转换后的数据称为目标数据。仅指定目标数据类型，而不指定初始数据类型（操作数的数据类型）。如果类型转换指令的操作数超出目标数数据类型值的范围，则结果输出取决于处理器类型。

从较大数据类型转换为较小数据类型时，可能会丢失信息（如从 LREAL 转换为 REAL）。

指令	功能
TO_BOOL	转换为布尔型数据
TO_INT	转换为 INT 型数据
TO_REAL	转换为 REAL 型数据
TO_STRING	转换为字符串

使用示例：

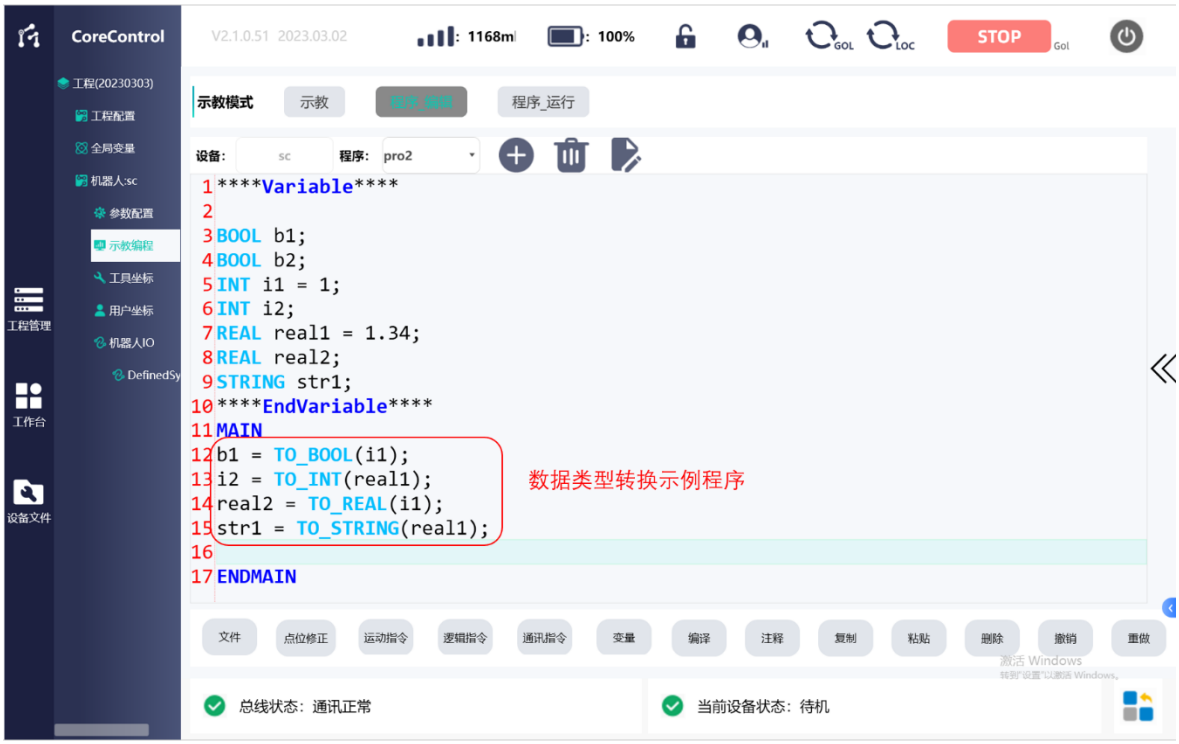


图 97 变量类型转换使用示例

程序文本示例：

```
b1 = TO_BOOL(i1);
i2 = TO_INT(real1);
real2 = TO_REAL(i1);
str1 = TO_STRING(real1);
```

程序文本说明：

1.将 INT 型变量 i1 转换为 BOOL 型变量 b1，初始赋值 i1=1，转换后的结果 b1=TRUE。

2.将 REAL 型变量 real1 转换为 INT 型变量 i2，初始赋值 real1=1.34，转换后的结果 i2=1。

3.将 INT 型变量 i1 转换为 REAL 型变量 real2，初始赋值 i1=1，转换后的结果 real2=1.0。

4.将 REAL 型变量 real1 转换为字符串变量 str1，初始赋值 real1=1.34,转换后的结果 str1='1.34'。

5.2 运动指令

5.2.1 MOVEL

指令功能：

该指令为直线运动指令，使机器人 TCP 点以直线轨迹运动到目标位置。

程序步指令格式：

MOVEL pointx，x，x，x，x；

//MOVEL 点位变量，速度，加速度，过渡参数，对应坐标系；**//**

编程方式：

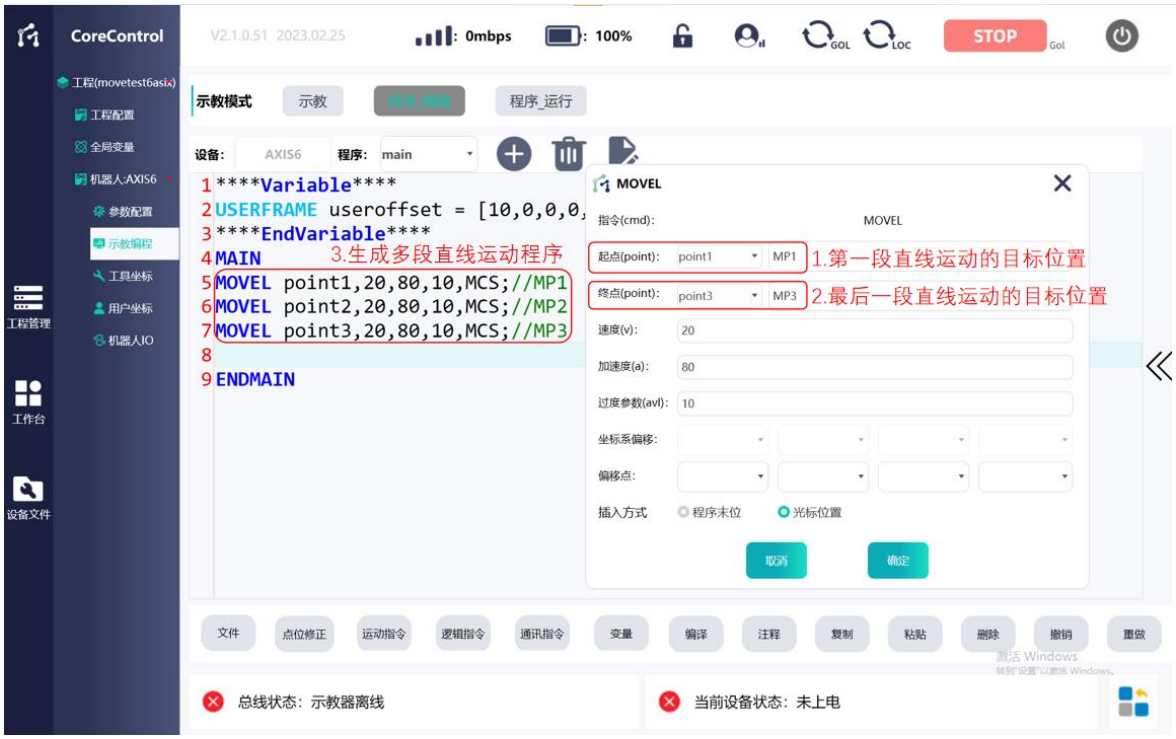


图 98 MOVEL 运动指令使用示例

插入窗参数说明：

1. **程序步起点终点**

MOVEL 指令的编辑插入方式可为单步轨迹程序插入，也支持批量插入多步轨迹程序。

单步轨迹程序插入时：“起点”选择直线运动的目标位置，“终点”留出空白。

多步轨迹程序插入时：“起点”选择第一段直线运动的目标位置，“终点”选择最后一段直线运动的目标位置。

2. 速度

为以当前点位起始点至程序步指定目标点的直线运动的最大速度，单位为 mm/s。

软件会自动生成默认值为 20，请根据实际需求更改。

3. 加速度

为以当前点位起始点至程序步指定目标点的直线运动的最大加速度 ，单位为 mm/s²。 软件会自动生成默认值为 80，请根据实际需求更改。

4. 过渡参数

过渡参数为开启过渡（BLEND）时从该点过渡的半径，详见 5.2.5 blend 指令说明，单位为 mm。

5. 偏移点（偏移变量）

使用点偏移功能，以输入“起点”位置为基础点位置，然后在基础点上进行一个相对偏移，机器人最终目标点位置为基础点偏移后的位置。偏移值由参数“偏移点”指定。指定偏移点需要声明一个点位变量用于存储偏移值信息，变量类型选择为 POINTL。在 X，Y，Z 填入需要偏移的 X，Y，Z 方向长度，在 A，B，C 填入需要偏移的姿态角度。

偏移点使用示例参见 5.1.2.1 POINTL 点变量使用说明。

6. 用户坐标系偏移

若 pointx 为用户坐标系下示教的点位，则可以使用用户坐标系偏移的功能。偏移后的用户坐标系为当前点位示教下的用户坐标系加上偏移值。指定偏移点需要声明一个点位变量用于存储偏移值信息，变量类型选择为 USERFRAME。在 X，Y，Z 填入需要偏移的 X，Y，Z 方向长度，在 A，B，C 填入需要偏移的姿态角度。

程序文本示例：

```
{"AXIS6":{"main":"****Variable****
USERFRAME useroffset = [10,0,0,0,0,0];
POINTL deltax = [8,0,0,0,0,0];
POINTL deltay = [0,10,0,0,0,0];
POINTL deltaz = [0,0,50,0,0,0];
POINTL pickpoint = [0,0,0,0,0,0];
****EndVariable****
MAIN
WAIT bstart == true;
bgriper = false;  //夹爪打开
MOVEJ point7,100,800,0,ACS;  //回原点
nx = 0;
ny = 0;
WHILE bcircleflag
pickpoint = deltax*nx + deltay*ny ;
MOVEJ point8,100,800,0,ACS;// 取料过渡点
MOVEL point6 + pickpoint+deltaz,200,800,0,PCS,[588.64,84.13,518.57,0,180,0]; //
```

取料点上方

MOVEL point6 + pickpoint,50,500,0,PCS,[588.64,84.13,518.57,0,180,0]; //取料点

bgriper = true; //夹爪关闭

DELAY 500; //延时 500ms

MOVEL point6 + pickpoint+deltaz,50,500,0,PCS,[588.64,84.13,518.57,0,180,0]; //取料点上方

MOVEJ point8,100,800,0,ACS;// 取料过渡点

MOVEJ point9,100,800,0,ACS;// 放料过渡点

MOVEL point5 + pickpoint+deltaz,200,800,0,PCS,[588.64,84.13,518.57,0,180,0]; //放料点上方

MOVEL point5 + pickpoint,50,500,0,PCS,[588.64,84.13,518.57,0,180,0]; //放料点

bgriper = false; //夹爪打开

DELAY 500; //延时 500ms

MOVEL point5 + pickpoint+deltaz,50,500,0,PCS,[588.64,84.13,518.57,0,180,0]; //放料点上方

MOVEJ point9,100,800,0,ACS;// 放料过渡点

INC nx;

IF nx == 5

INC ny;

nx = 0;

IF ny == 5

nx = 0;

ny = 0;

bcircleflag = false; //退出循环

IF_END

IF_END

WHILE_END

ENDMAIN"}}

程序文本说明：

这是一段机械手取放料的程序。当全局变量 bstart 为 true 时，夹爪打开，机械手以 100°/s 的速度，加速度为 800°/s²的曲线轨迹在关节坐标系下回到原点 point7。进入 WHILE 循环，机械手执行 MOVEJ 指令以 100°/s 的速度运行到取料过渡点 point8，接着以速度为 200mm/s，加速度为 800mm/s²的直线轨迹在用户坐标系下运行到取料上方点 point6 + pickpoint+deltaz，其中 point6 是基于用户坐标系下的第一个取料点，pickpoint 为相对第一个取料点的偏移值，deltaz 是 POINTL 数据类型的点位。机械手再运行到取料点，关闭夹爪，抓取物料，延时 500ms；再移动到取

料上方点，取料过渡点，放料过渡点，进行放料操作。当 ny 等于 5，则取放料完毕，退出 WHILE 循环。

注意

点参数支持在MCS坐标系，TCS坐标系，PCS坐标系下示教的点。

5.2.2 MOVEJ

指令功能：

该指令表示机器人以各个关节进行点到点的运动，机器人 TCP 点的轨迹可能为不规则的曲线。

程序步指令格式

MOVEJ pointx，x，x，x，x；

//MOVEJ 点位变量，速度，加速度，过渡参数，对应坐标系；**//**

编程方式：

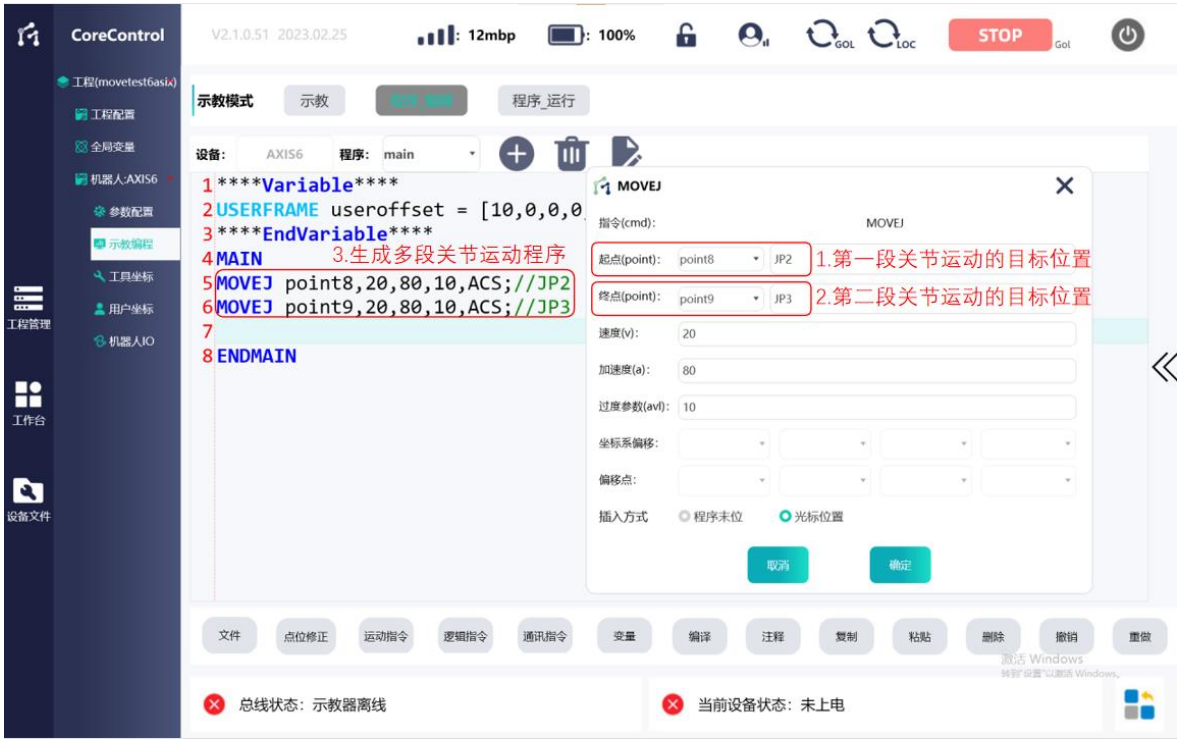


图 99 MOVEJ 运动指令使用示例

插入窗参数说明：

1. 程序步起点终点

同 MOVEL。可插入单段轨迹运动也可插入多段轨迹运动。插入和指定的“起点”

“终点”方法与 MOVEL 相同。

2.速度

为以当前点位起始点至程序步指定目标点的轨迹运动的最大速度，单位为°/s。软件会自动生成默认值为 20，请根据实际需求更改。

3.加速度

为以当前点位起始点至程序步指定目标点的轨迹运动的最大加速度，单位为°/s²。软件会自动生成默认值为 80，请根据实际需求更改。

4.过渡参数

过渡参数为开启过渡（BLEND）时的过渡值。

注意

当前机器人为模组机器人时，此参数不起作用，MOVEJ过渡以90%的速度进行过渡。当前机器人为SCARA机器人和六轴机器人时，此参数为过渡半径，单位为mm。

5.偏移点

与 MOVEL 点偏移功能相似。不同的是指定偏移点的变量类型选择为 POINTJ。在 J1 ~ J6 填入每个关节需要进行偏移的角度。

程序文本示例：

MOVEJ point4,20,80,0.1,ACS;

MOVEJ point5,20,80,0.1,ACS;

程序文本说明：

机器人各个关节以速度为 20°/s，加速度为 80°/s²的曲线轨迹在关节坐标系下运动到目标点 point4，到达点 point4 后减速到 0。然后再启动下一段运动，运动到点 point5，到达点 point5 后减速到 0 结束运动。运动的运动轨迹可能都为不规则曲线，根据示教点的实际位置决定。

注意

点参数仅支持在ACS坐标系下示教的点。

5.2.3 MOVER

指令功能：

机器人相对运动指令，支持关节，笛卡尔，工具以及用户坐标系。此指令可使机器人以当前位置为起点，以运动参数中设定的长度（笛卡尔坐标系等时）或角度

（关节坐标系时）进行相对位移。

程序步指令格式：

笛卡尔坐标系时

MOVER X=x, Y=x, Z=x, A=x, B=x, C=x, x, x, MCS,[0,0,0,0,0,0];

//MOVER X 坐标，Y 坐标，Z 坐标，A 姿态，B 姿态，C 姿态，速度，加速度，MCS,[0,0,0,0,0,0];**//**

关节坐标系时

MOVER J1=x, J2=x, J3=x, J4=x, J5=x, J6=x, x, x, ACS;

//MOVER 关节 1 角度，关节 2 角度，关节 3 角度，关节 4 角度，关节 5 角度，关节 6 角度，速度，加速度，ACS;**//**

工具或用户坐标系时

MOVER X=x, Y=x, Z=x, A=x, B=x, C=x, x, x, TCS(PCS),[x,x,x,x,x,x];

//MOVER X 坐标，Y 坐标，Z 坐标，A 姿态，B 姿态，C 姿态，速度，加速度，TCS(PCS),[工具或用户坐标系的标定结果];**//**

编程方式：

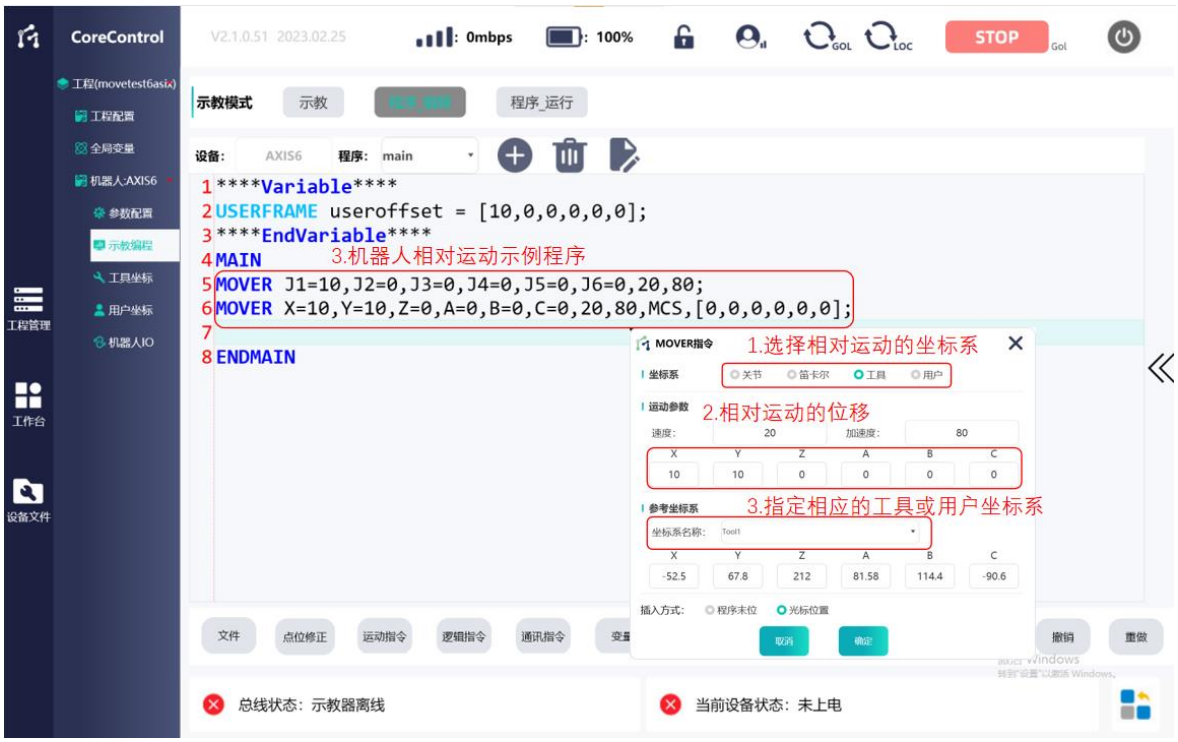


图 100 MOVER 运动指令使用示例

插入窗参数说明：

1.坐标系

可指定关节坐标系，笛卡尔坐标系，工具坐标系，用户坐标系。

2.运动参数

指定速度，加速度。速度加速度单位根据指定的坐标系而决定。以及以机器人当前位置为起点的相对移动的长度（笛卡尔坐标系等）或关节角度（关节坐标系）。

3.参考坐标系

坐标系指定为工具或用户坐标系时，选择设定的对应坐标系。笛卡尔坐标系和关节坐标系时不用选择。

程序文本示例：

MOVER J1=10,J2=0,J3=0,J4=0,J5=0,J6=0,20,80;

MOVER X=10,Y=10,Z=0,A=0,B=0,C=0,20,80,MCS,[0,0,0,0,0,0];

程序文本说明：

机器人以当前位置为起点，关节 1 以 20°/s 的速度 80°/ s^2的加速度正向移动 10°，运动完成后减速到 0。然后机器人再以移动后的位置为起点，向 X 方向正方向，Y 方向正方向移动 10mm，运动完成后结束。

5.2.4 MOVEC

指令功能：

机器人 TCP 点从起始位置经过中间点位置再到终点位置做圆弧运动的指令。机器人经过中间点位置不会减速到 0，到达终点为才减速停止。

程序步指令格式：

MOVEC pointx , pointx, x, x, x, x, x, x, false(true);

//MOVEC 中间点，终点，速度，加速度，过渡参数，对应坐标系，工具或用户坐标系的标定参数，预留参数**//**

编程方式：

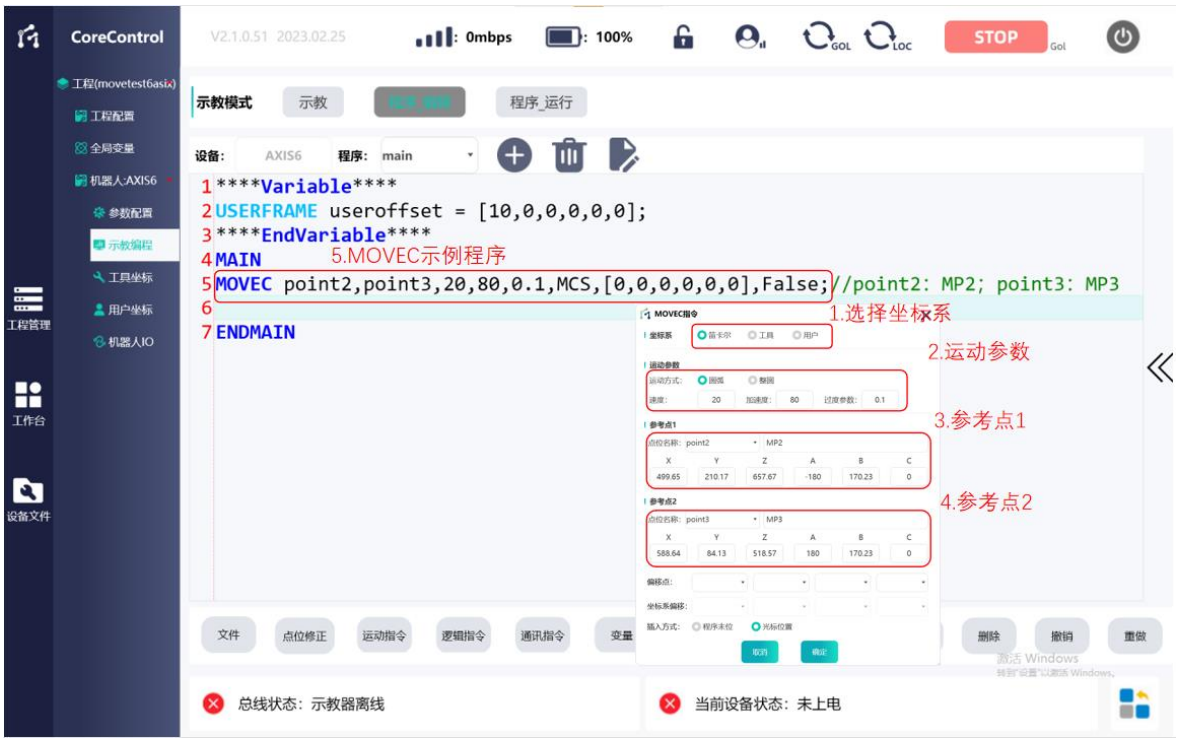


图 101 MOVEC 运动指令使用示例

插入窗参数说明：

1.坐标系

指定圆弧运动参考点所处的坐标系，支持笛卡尔，工具和用户坐标系。

2.运动参数

指定轨迹为圆弧。设定圆弧运动的速度，加速度以及过渡参数。过渡参数为过渡时的半径，单位为 mm。

3.参考点

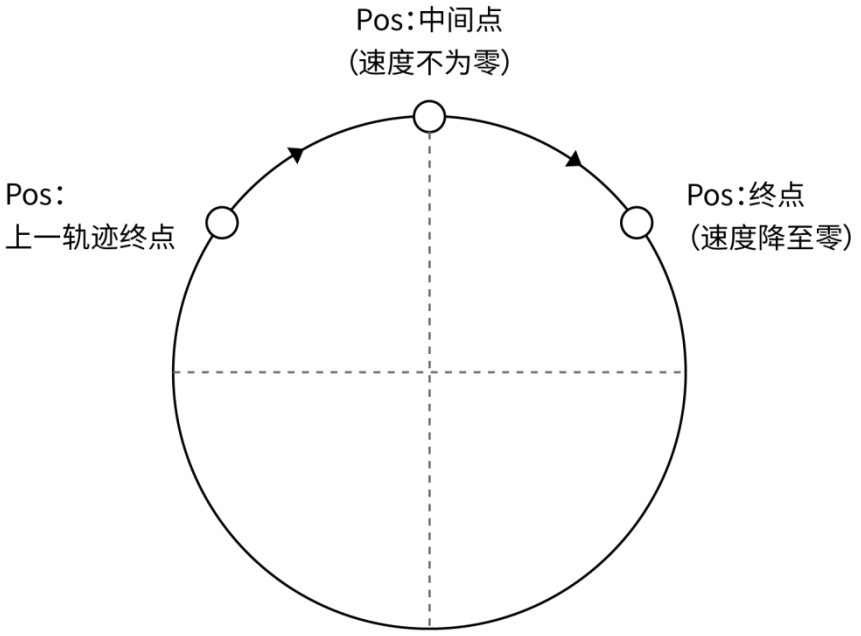


图 102 MOVEC 运动指令轨迹及相关点关系

参考点 1：圆弧中间辅助点的位置。

参考点 2：圆弧终点的位置。

圆弧轨迹以当前点为起始点，以参考点 1 为中间点，参考点 2 为终点。空间任意不共线三点可确定圆形轨迹，本指令默认执行三点间劣弧部分，限定圆心角小于 180°。

程序文本示例：

```
BLEND
MOVEC point7,point8,20,80,0.1,MCS,[0,0,0,0,0,0],False;
MOVEC point9,point6,20,80,0.1,MCS,[0,0,0,0,0,0],False;
BLEND_END
```

程序文本说明：

本示例使用 BLEND 指令将两个半圆圆弧拼接为一个整圆。

第一段圆弧轨迹以机器人初始位置为起点，点 point7 为中间点，经过点 point7 时机器人不会减速，继续以圆弧轨迹运动到点 point8，点 point8 为圆弧终点。因为使用了 BLEND 指令，所以运动到点 point8 也不会减速停止，而是继续向下一段圆弧轨迹运动。第二段圆弧轨迹以点 point9 为中间点，以点 point6 为终点。实际使用中设计整圆轨迹需要注意两段圆弧的四个点的位置。就上述程序而言，第二段圆弧的终点 point6 应该和机器人第一段圆弧轨迹的起始位置重合。

BLEND 指令的功能参见 5.2.5 BLEND 指令的说明。

若程序改成以下方式：

MOVEC point7,point8,20,80,0.1,MCS,[0,0,0,0,0,0],True;

那么机器人会运行一个整圆，从当前点移动到 point7，接着 point8，再回到当前点。

5.2.5 BLEND

指令功能：

开启连续运动轨迹的功能。添加后会生成一个以 BLEND 开始，BLEND_END 结束的程序段。在生成的此程序段中间插入运动指令，如 MOVEL，MOVEJ，MOVEC 等。即可开启这些运动指令的过渡功能。

此功能开启后，机器人每段运动路径的终点不会减速到 0 停止，而是以设定的过渡参数平滑过渡该点。过渡参数在运动指令中设置，参见 5.2.1 MOVEL 等的参数讲解。例如运用于将两段半圆圆弧轨迹拼接为一个整圆。

程序步指令格式：

BLEND

（运动指令）；

（运动指令）；

……

BLEND_END

应用示例及讲解：

参见 5.2.4 MOVEC 的示例讲解。

目前在同一个 BLEND 指令中不支持 MOVEL 和 MOVEJ 混合使用。

5.2.6 HOMEJ

指令功能：

机器人关节回零指令。此指令可使机器人以当前位置为起点，以运动参数中设定的速度和加速度，在关节坐标系下进行回零操作，运行到机器人的零位。

程序步指令格式：

HOMEJ J1, J2, J3, J4, J5, J6, x, x;

// HOMEJ 关节 1，关节 2，关节 3，关节 4，关节 5，关节 6，速度，加速度;**//**

编程方式：

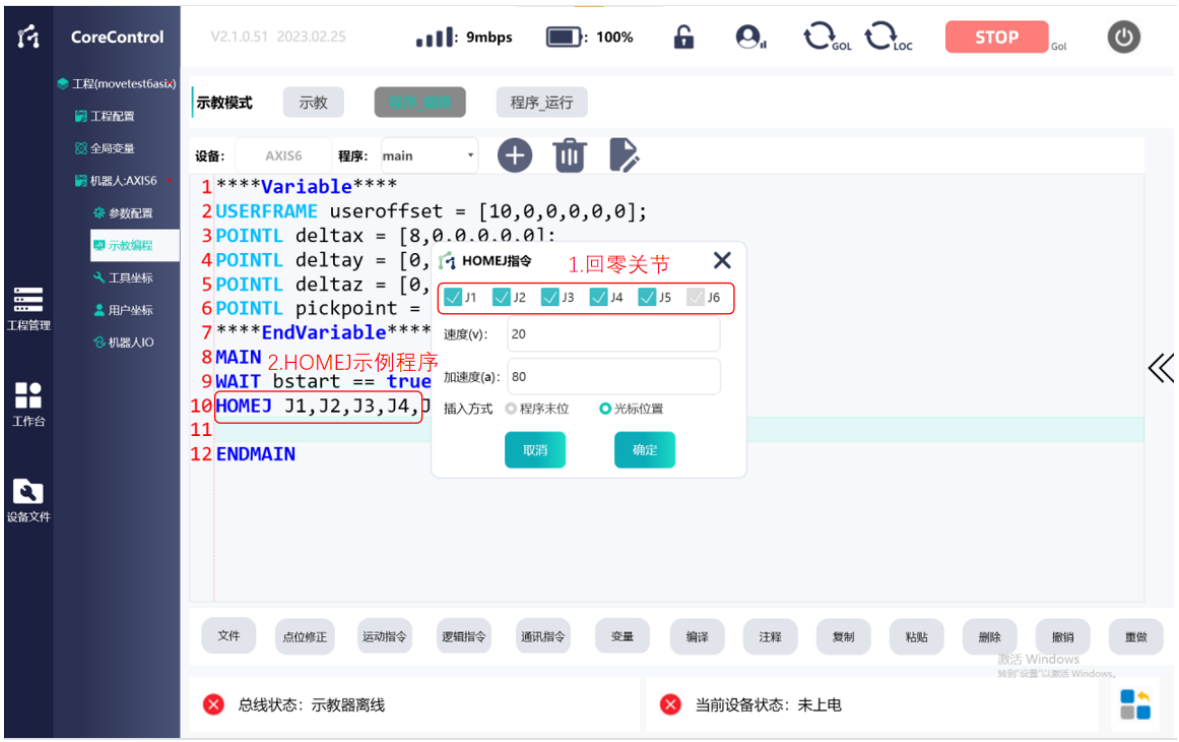


图 103 MOVER 运动指令使用示例

插入窗参数说明：

1.坐标系

可指定需要回零的关节，当勾显示绿色，则指定该关节回零操作，否则不进行回零操作。

2.运动参数

指定速度，加速度。速度单位°/s，加速度单位°/s^2。

程序文本示例：

HOMEJ J1,J2,J3,J4,J5,100,800;

程序文本说明：

机器人以当前位置为起点，关节 1，关节 2，关节 3，关节 4，关节 5 都以 100°/s 的速度，800°/ s²的加速度回零运动；关节 6 不动作。

5.2.7 POSITIONSET

指令功能：

机器人运行过程中置位/复位变量的指令。生产过程中为加快节拍，机器人有时候不需要运行到精确点位，再进行变量置位/复位操作。POSITIONSET 指令可以实现机器人在运行过程中（没有运行到精确点位）对变量进行置位/复位操作。

程序步指令格式：

POSITIONSET 变量,TRUE/FALSE, pointx, X;

// POSITIONSET 变量,TRUE/FALSE, 点位,距离点位的距离 **//**

编程方式：

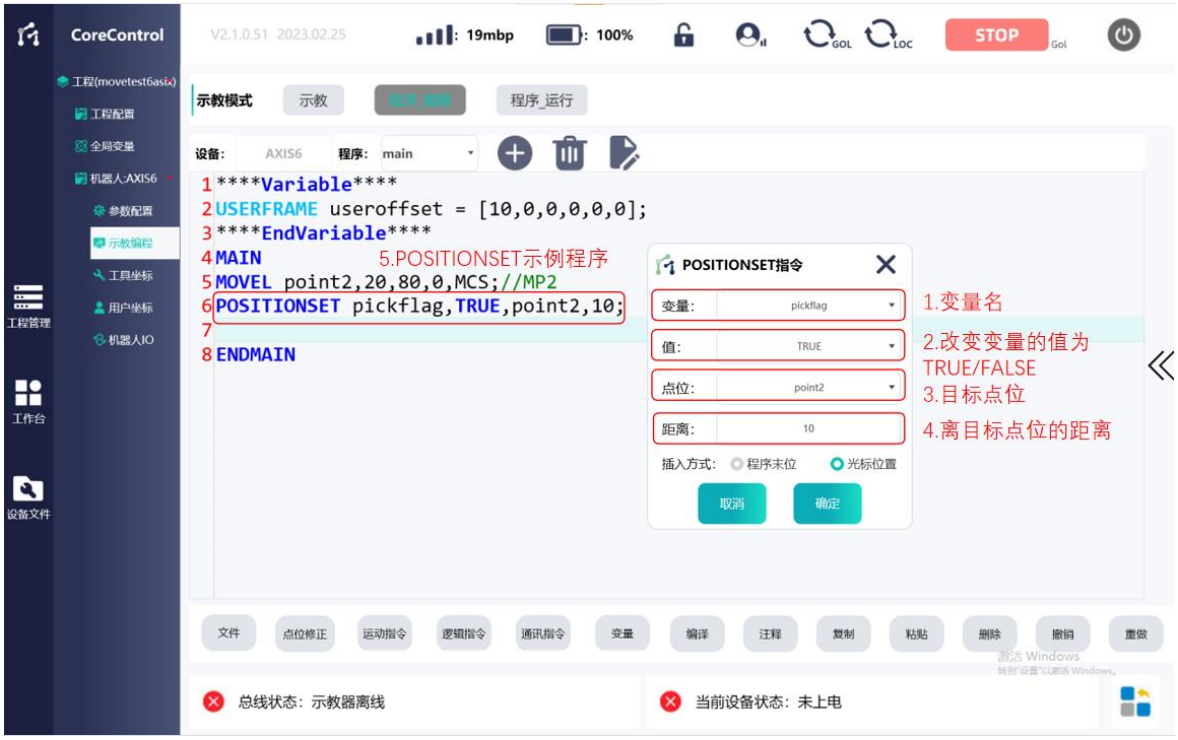


图 104 MOVER 运动指令使用示例

插入窗参数说明：

1.变量名

需要置位/复位的变量。

2.值

TRUE/FALSE。

3.点位

目标点位

4.距离

离目标点位的距离，单位 mm。当机器人与该点位距离为该值时，变量进行置位/复位操作。

程序文本示例：

MOVEL point3,100,800,0,MCS;

POSITIONSET pickflag,TRUE,point3,5;

程序文本说明：

机器人以 100mm/s 的速度，800mm/s^2 的加速度在笛卡尔坐标系下直线运动，在距离 point3 为 5mm 时，变量 pickflag 置 TRUE。

5.3 逻辑指令

5.3.1 IF

指令说明：

IF 指令用于条件判断表达式跳转控制。当表达式结果为真时，程序执行 IF 下程序块的语句。当表达式结果为假时，会跳过条件判断语句，执行后续语句。条件判断表达式结果必须是 BOOL 类型。每一个 IF 指令必须以关键字 IF_END 作为条件判断程序块的结束。

程序步指令格式：

IF （条件判断表达式）

（IF 下其他语句）

IF_END

编程方式：

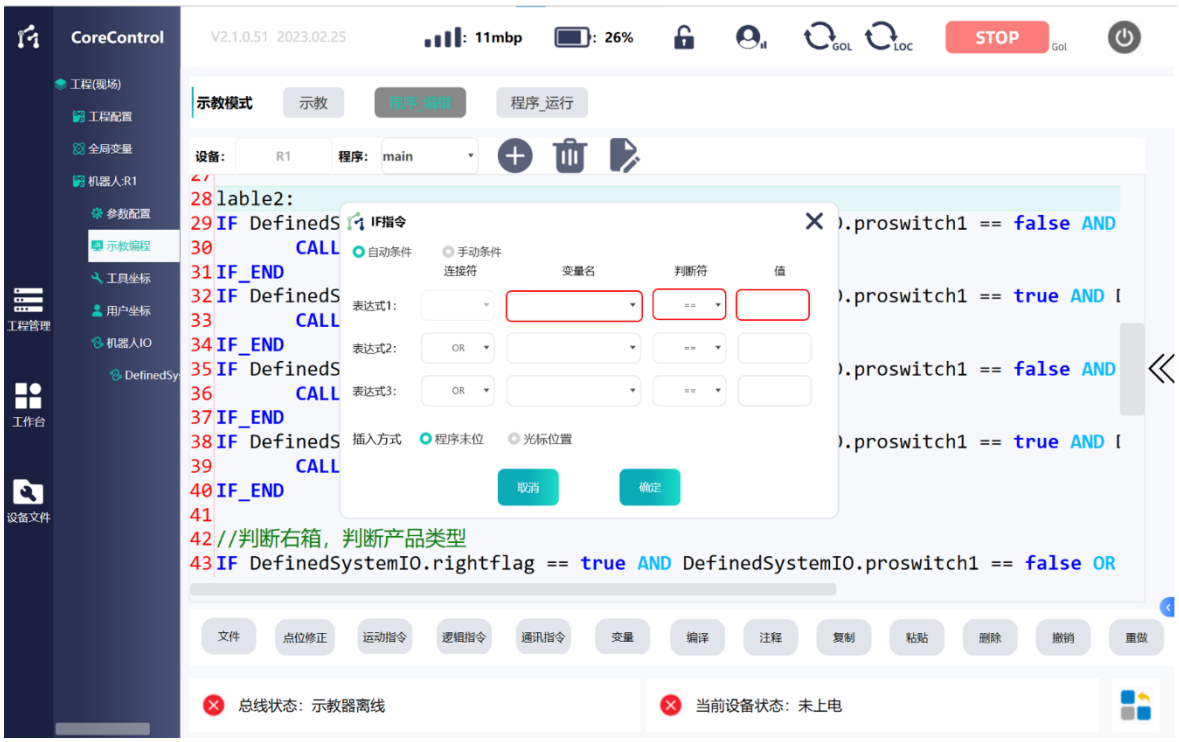


图 105 IF 逻辑指令使用示例

插入窗参数说明：

1.变量名

指定为变量(支持 BOOL 型、INT 型等)，可为全局变量、局部变量和 IO 变量。

2.判断符

支持 ==（相等）;

！=（不等）;

>=（大于等于）;

<=（小于等于）；

>（大于）；

<（小于）；

3.值

指定为常数（BOOL 型则指定为 TRUE 或 FALSE）。

插入 IF 判断语句后，在生成的 IF 和 IF_END 中间程序段插入需要执行的程序块。

程序文本示例：

IF i1 == 1

MOVEL point6,20,80,0.1,MCS;

IF_END

MOVEL point7,20,80,0.1,MCS;

程序文本说明：

当 i1 的值等于 1 时，机器人执行直线运动到点 point6，执行完成后顺序执行直线运动到点 point7 程序步，当 i1 的值不等于 1 时，跳过 IF 语句，机器人直接执行直线运动到点 point7。

5.3.2 LOOP

指令说明：

LOOP 指令是按设定次数进行循环的循环语句，循环次数参数由 LOOP 后的参数指出，参数支持正整数输入。LOOP 语句以关键字 LOOP_END 结束。可以使用指令 BREAK 打断循环，强制结束循环。

程序步指令格式：

LOOP x（循环次数）

(循环的语句)；

LOOP_END

编程方式：

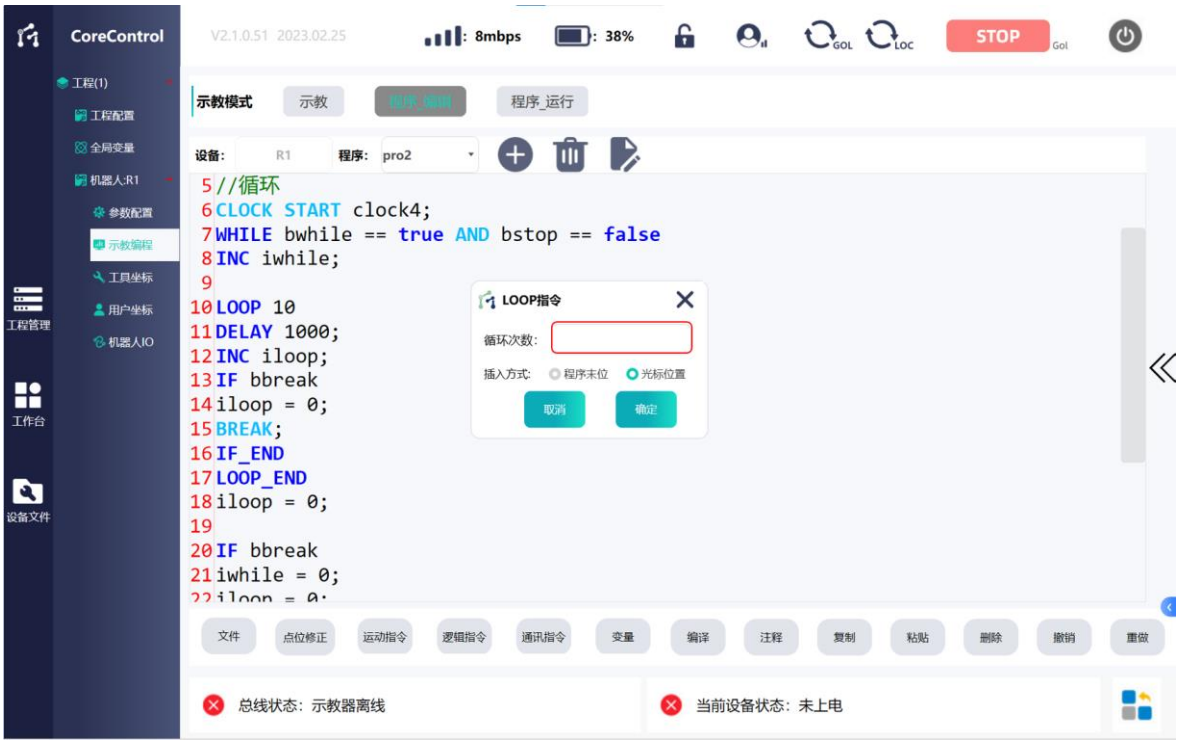


图 106 LOOP 逻辑指令使用示例

插入窗参数说明：

参数为需要进行循环的次数，输入正整数。

程序文本示例：

LOOP 10

DELAY 1000;

INC iloop;

IF bbreak

iloop = 0;

BREAK;

IF_END

LOOP_END

程序文本说明：

循环开始，延时 1000ms，整型变量 iloop 自增 1，判断布尔型变量 bbreak 是否为 TRUE，如果为 TRUE 则将整型变量 iloop 清零，并结束循环。如果变量 bbreak 一直不为 0 则

5.3.3 WHILE

指令说明：

WHILE 指令是执行一个条件判断循环语句。判断结果由 WHILE 后的布尔条件表达式给出。当表达式为真时，执行循环语句块中的语句，并一直循环执行。直到表

达式变为假时，结束循环并执行循环语句后的语句。或者使用指令 BREAK 可以打断循环，强制结束循环。 WHILE 语句以关键字 WHILE_END 结束。条件表达式的设定与 IF 语句相同。

程序步指令格式：

WHILE （条件判断表达式）

（循环的语句）

WHILE_END

编程方式：

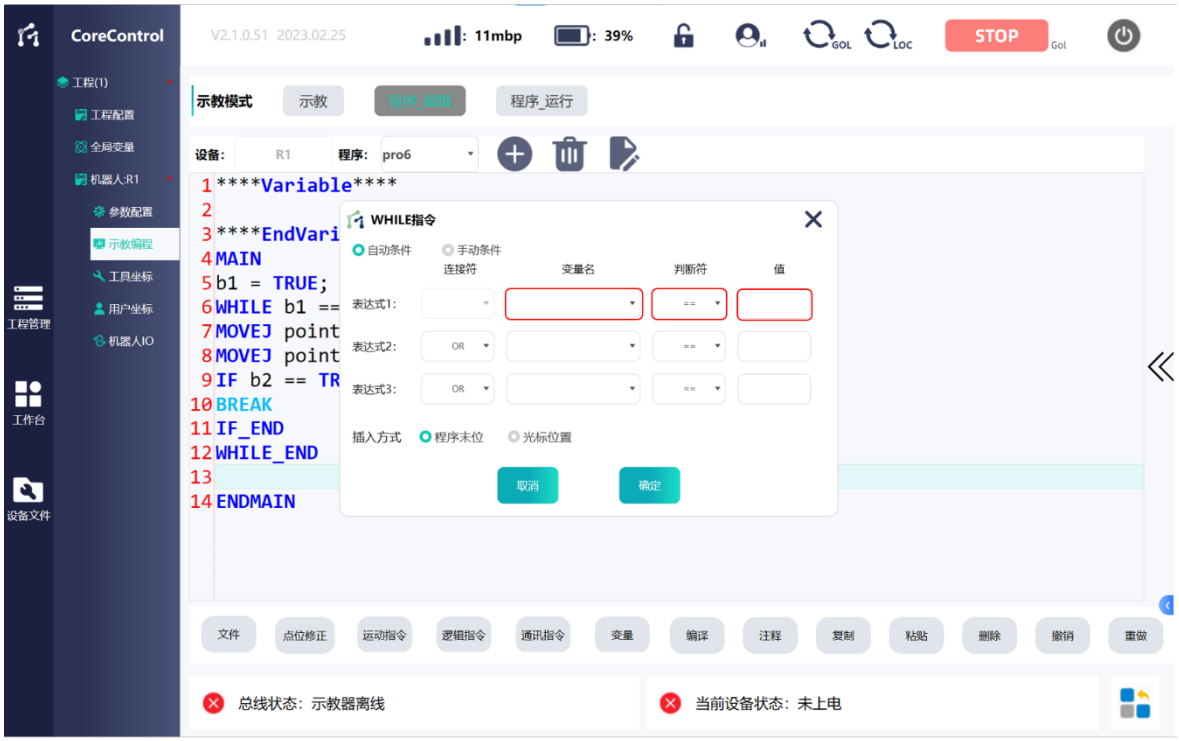


图 107 WHILE 逻辑指令使用示例

插入窗参数说明：

与 IF 语句的插入参数相同，都为条件判断表达式。参见 5.3.1 IF 的参数讲解。

程序文本示例：

b1 = TRUE;

WHILE b1 == TRUE

MOVEJ point4,20,80,0.1,ACS;

MOVEJ point5,20,80,0.1,ACS;

IF b2 == TRUE

BREAK

IF_END

WHILE_END

程序文本说明：

将变量 b1 置为 TRUE。WHILE 循环的条件满足。机器人执行关节移动到点 point4，然后移动到点 point5。并且一直循环下去。如果循环过程中通过外部操作或其他程序设计等使变量 b2 变为 TRUE，IF 语句条件满足，会执行 BREAK 打断循环，跳转到 WHILE_END 后的程序。或者当变量 b1 变为 FALSE 时，WHILE 循环的条件不满足，结束循环，同样会跳转到 WHILE_END 后的程序。

5.3.4 INC

指令说明：

自增指令，支持 INT , LINT 型整型变量。每执行一次程序变量增加 1。.

程序步指令格式：

INC x;

INC 整数（INT 或 LINT 型变量）;

应用示例及讲解：

参见 5.1.1.4 INT 的示例说明。

5.3.5 DEC

指令说明：

自减指令，支持 INT , LINT 型整型变量。每执行一次程序变量减少 1。

程序步指令格式：

DEC x;

DEC 整数（INT 或 LINT 型变量）;

应用示例及讲解：

与 INC 指令相似。

5.3.6 EXIT

指令说明：

用于跳转到程序末端 ENDMAIN 的指令。程序模式为自动单次时，会直接结束程序执行。程序模式为自动循环模式时，会结束当前顺序执行跳转到下一次顺序执行开始。

编程方式：

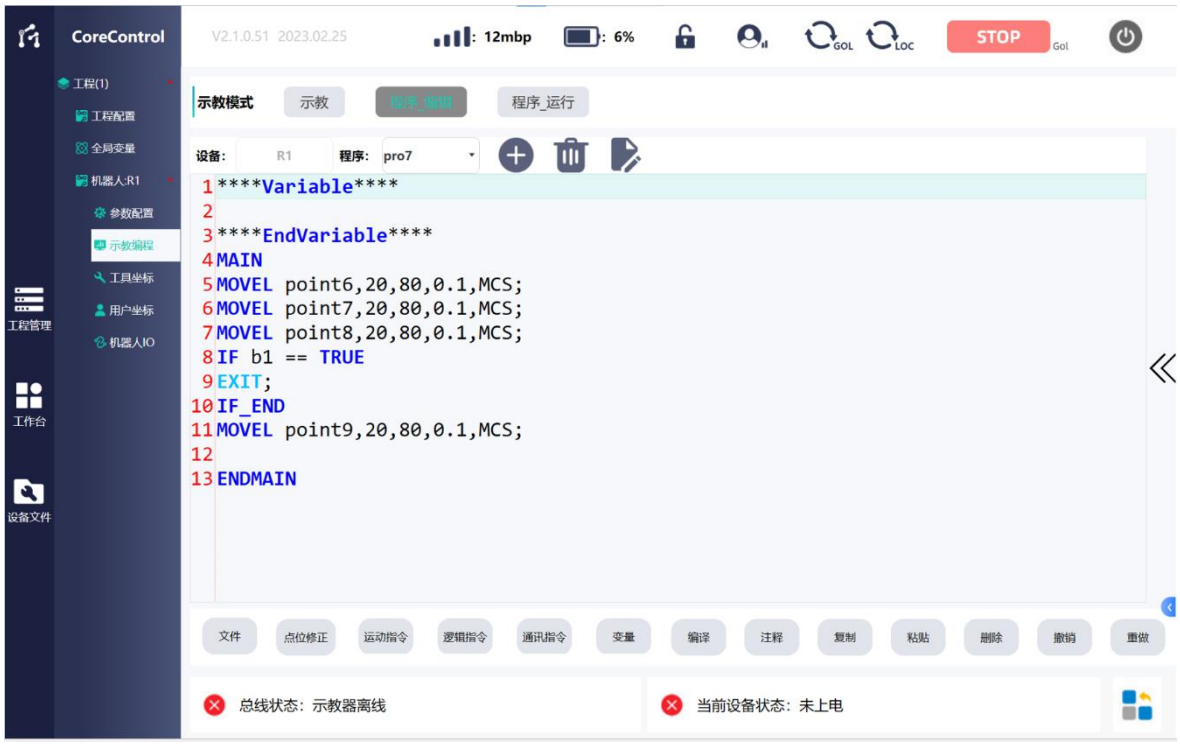


图 108 EXIT 逻辑指令使用示例

程序文本示例：

MOVEL point6,20,80,0.1,MCS;

MOVEL point7,20,80,0.1,MCS;

MOVEL point8,20,80,0.1,MCS;

IF b1 == TRUE

EXIT;

IF_END

MOVEL point9,20,80,0.1,MCS;

程序文本说明：

机器人执行直线运动到点 point6，然后依次运动到点 point7，点 point8。判断变量 b1 是否为 TRUE，若为 TRUE，则执行 EXIT 指令，程序跳转到末端，结束此次程序循环。若不为 TRUE，则继续执行运动到点 point9。

5.3.7 BREAK

指令说明：

使用 BREAK 指令可提前结束循环语句（LOOP 循环和 WHILE 循环），通过在循环语句中调用 BREAK 指令，循环语句块立即结束，并开始执行后续程序。

应用示例及讲解：

参见 5.3.3 WHILE 的示例说明。

5.3.8 SETDO

指令说明：

将 IO 模块的输出端口置位为 TRUE 或 FALSE 的指令。常用于打开 IO 模块输出口连接的物理执行机构如气缸电磁阀等。

程序步指令格式：

SETDO x. x TRUE(或 FALSE);

//SETDO IO 模块名称.IO 变量名称 置位状态;//

编程方式：

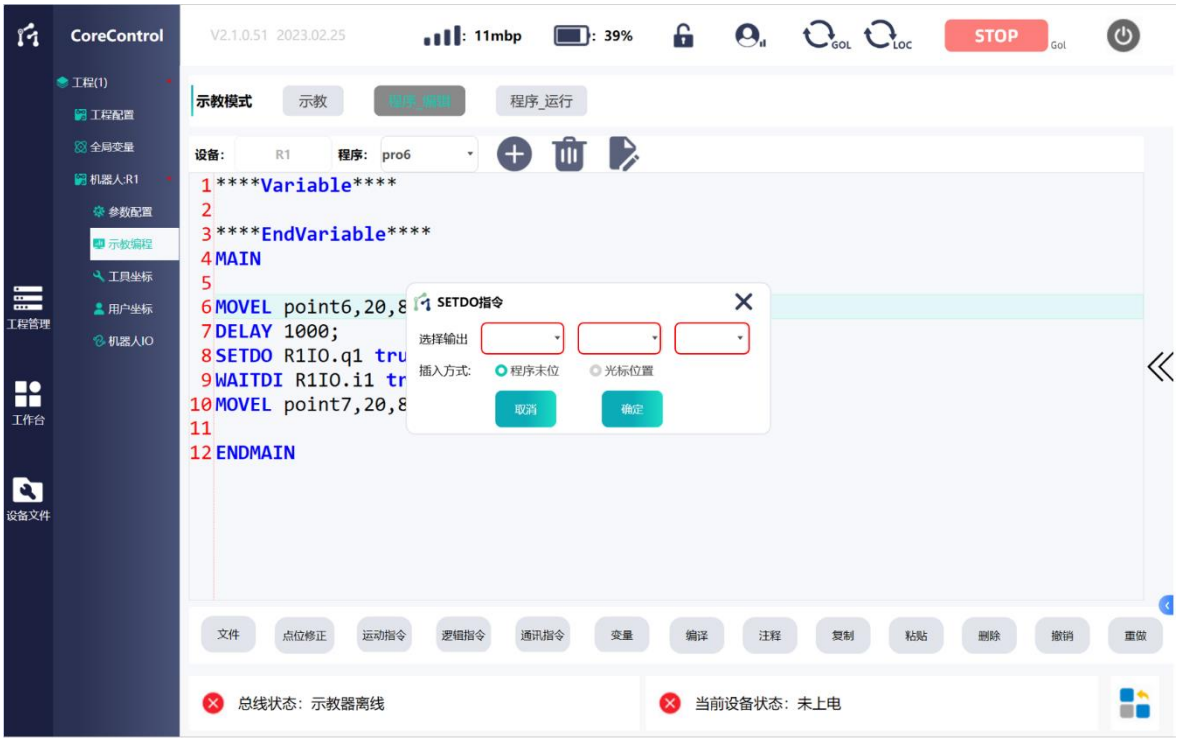


图 109 SETDO 逻辑指令使用示例

插入窗参数说明：

选择输出：

1.IO 模块的名称，为添加 IO 设备时自主命名的。参见 4.4.3 I/O 设备添加及 I/O 变量关联。

2.IO 变量名称，即 IO 输出端口程序调用的名称，参见 4.4.3 I/O 设备添加及 I/O 变量关联。

3.IO 端口的状态，为 TRUE 或是 FALSE。

程序文本示例：

MOVEL point6,20,80,0.1,MCS;

DELAY 1000;

SETDO R1IO.q1 true;

WAITDI R1IO.i1 true;

MOVEL point7,20,80,0.1,MCS;

程序文本说明：

本示例为机器人执行抓取物料动作的程序。

假设已知示教点 point6 为抓取物料点，设定 q1 为控制机器人夹爪电磁阀的 IO 端口，设定 i1 为夹爪闭合感应磁开，示教点 point7 为抓取点上方。

机器人先移动到点 point6，到达点 point6 后使用 DELAY 指令延时 1000ms，打开 IO 模块输出端口 q1，然后等待 IO 模块输入端口 i1 有效。然后机器人再移动到点 point7。

WAITDI 指令说明参见 5.3.9 WAITDI，DELAY 指令说明参见 5.3.11 DELAY。

5.3.9 WAITDI

指令说明：

等待 IO 模块的输入端口被置位（TRUE）或被复位(FALSE)，直到 IO 变量变为指定的状态，才执行后续语句。常用于判断 IO 输入是否有效，比如检测气缸磁开是否有效来判断气缸是否到位。

程序步指令格式：

WAITDI x .x TRUE(或 FALSE);

WAITDI “IO 设备名称”.”IO 变量名” “状态”;

插入窗参数说明：

与 SETDO 指令相似，参数 B 需指定为 IO 模块输入端口程序调用名称。

应用示例及讲解：

参见 5.3.8 SETDO 的示例说明。

5.3.10 WAIT

指令说明：

WAIT 指令使程序执行等待，直到指定的条件满足，才会执行 WAIT 指令后续程序段，否则一直等待下去。现仅支持 BOOL 型变量条件判断。

程序步指令格式：

WAIT x == TRUE(或 FALSE);

//WAIT 变量名称 == 状态；**//**

编程方式：

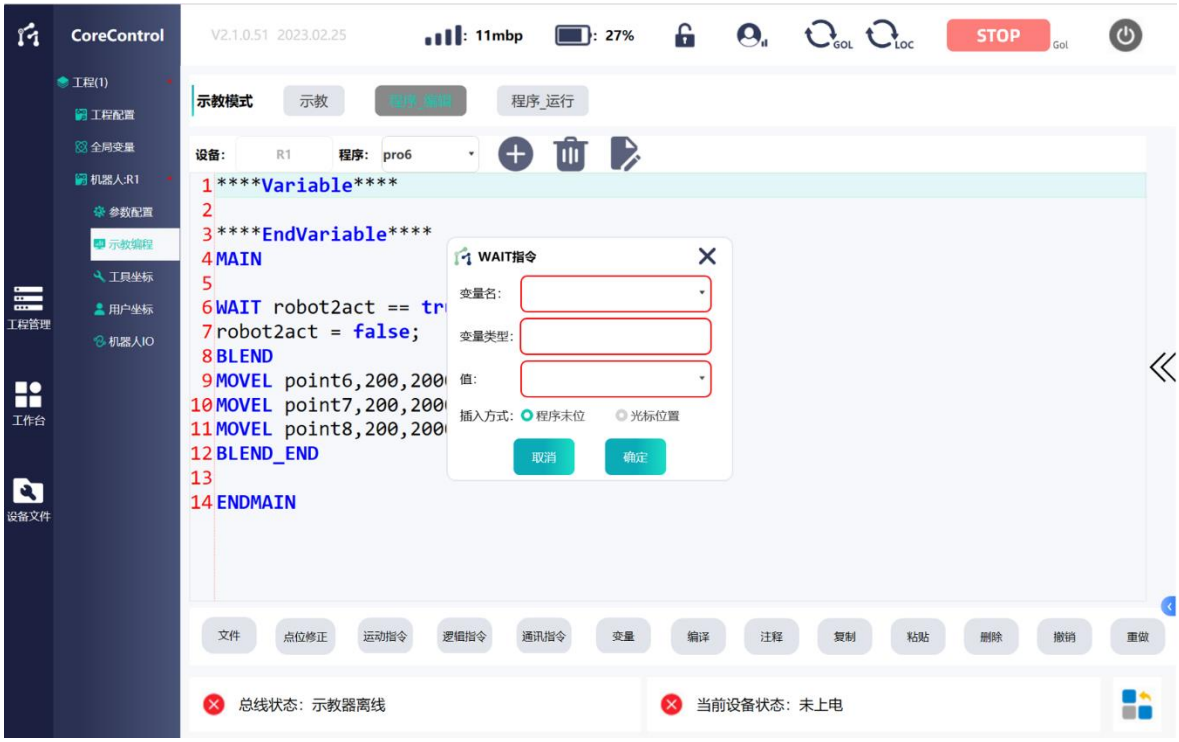


图 110 WAIT 逻辑指令使用示例

插入窗参数说明：

1.变量名

指定 BOOL 型变量名称，可支持全局变量和局部变量。

2.变量类型

变量类型，自动生成，现仅为 BOOL 型。

3.值

变量状态，为 TRUE 或 FALSE。

程序文本示例：

WAIT robot2act == true;

robot2act = false;

BLEND

MOVEL point6,200,2000,0.1,MCS;//

MOVEL point7,200,2000,0.1,MCS;//

MOVEL point8,200,2000,0.1,MCS;//

BLEND_END

程序文本说明：

本示例将 WAIT 指令应用于两个机器人协同工作的场景。声明全局变量 robot2act 为机器人 2 动作的条件标志位。

机器人 1 工作完毕后置位一个全局变量 robot2act 控制机器人 2 开始工作。机

器人 1 的编程未列出，只需在动作完毕后在程序最后加入程序段 robot2act = true 即可。机器人 2 程序（上述程序）先判断变量 robot2act 是否为 TRUE。若不为 TRUE 则一直等待（说明此时机器人 1 程序未执行完毕），若为 TRUE，将 robot2act 复位为 FALSE（为了满足下次程序循环的逻辑），然后机器人 2 执行以过渡方式运动过点 point6, 点 point7, 点 point8。

5.3.11 DELAY

指令说明：

该指令用于使程序延时等待一段时间，指定时间单位为 MS。

程序步指令格式：

DELAY x;

//DELAY 1000;**//**

编程方式：

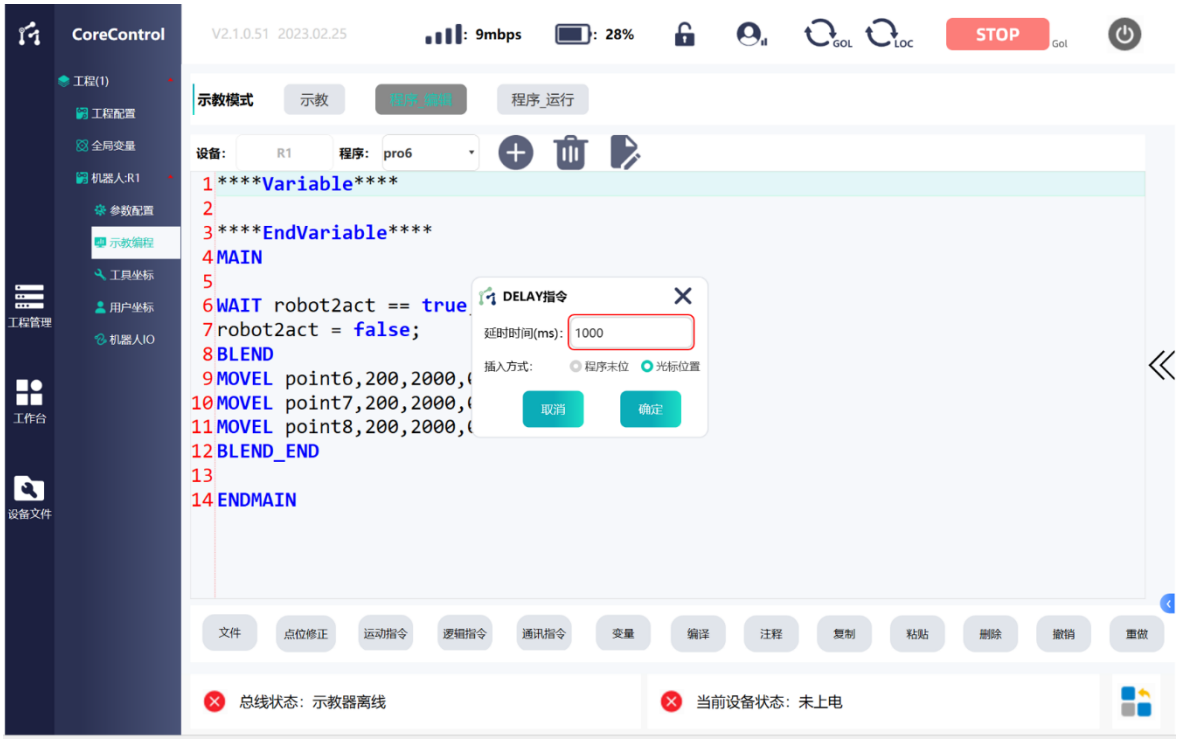


图 111 DELAY 逻辑指令使用示例

插入窗参数说明：

填入程序需要延时等待的时间，单位为 ms。如图延时 1000ms。

应用示例及讲解：

参见 5.3.8 SETDO 的示例说明。

5.3.12 DELAYSET

指令说明：

该指令用于延时置位或复位 BOOL 变量或 IO 变量，指定时间单位为 MS。现支持全局或局部变量和 IO 输出端口变量。此指令的延时不会导致程序延时等待，指令执行完毕后立即执行后续语句，参数变量会在执行 DELAYSET 语句完毕后经过设定的延时时间后置位或复位。

程序步指令格式：

DELAYSET x, TRUE(或 FALSE), x；

//DELAYSET 变量名称，状态，延时时间；**//**

编程方式：

图 112 DELAYSET 逻辑指令使用示例

插入窗参数说明：

1.类型

选择是 IO 变量还是普通 BOOL 型变量（全局或局部）。

2.延时时长

指定延时置位或复位时间，单位为 ms。

3.选择 IO

选择 IO 模块的名称、IO 变量的名称以及变量改变后的状态。

程序文本示例：

DELAYSET R4IO.q1,true,1000;

MOVEL point6,200,2000,0.1,MCS;

MOVEC point7,point8,200,2000,0.1,MCS,[0,0,0,0,0,0],False;

SETDO R4IO.q1 false;

程序文本说明：

程序将 IO 模块 R4IO 的输出端口 q1 延时 1S 打开，此指令发送下去，但是不立即执行打开 IO 端口的动作。然后立即开始（不会延时）执行机器人直线运动到点 point6，到达点 point6 后。开始执行中间点为点 point7, 终点为点 point8 的圆弧轨迹。最后将 IO 端口 q1 关闭。

假设机器人执行直线运动和圆弧运动的时间大于 1S，IO 端口 q1 会在执行完 DELAYSET 指令后经过 1S 后打开，但是不会影响后续运动指令的执行，此示例中是在机器人运动过程中打开。

5.3.13 GOTO

指令说明：

GOTO 指令用于跳转到程序的不同部分。跳转功能需要跳转指令和跳转目标标签搭配使用。在需要开始跳转的程序行插入跳转指令，然后在跳转目标程序段的前一行插入跳转目标标签，跳转目标标签用 “标签名称”加 “ ： ”定义，如“LABEL1:”。标签名称可以自定义。若指定了跳转条件，则在条件满足时跳转，条件不满足时，程序继续向下一行执行。

GOTO 指令格式：

GOTO x;

//GOTO 标签名称;**//**

跳转目标标签格式：

xx ：

//标签名称 ： //

编程方式：

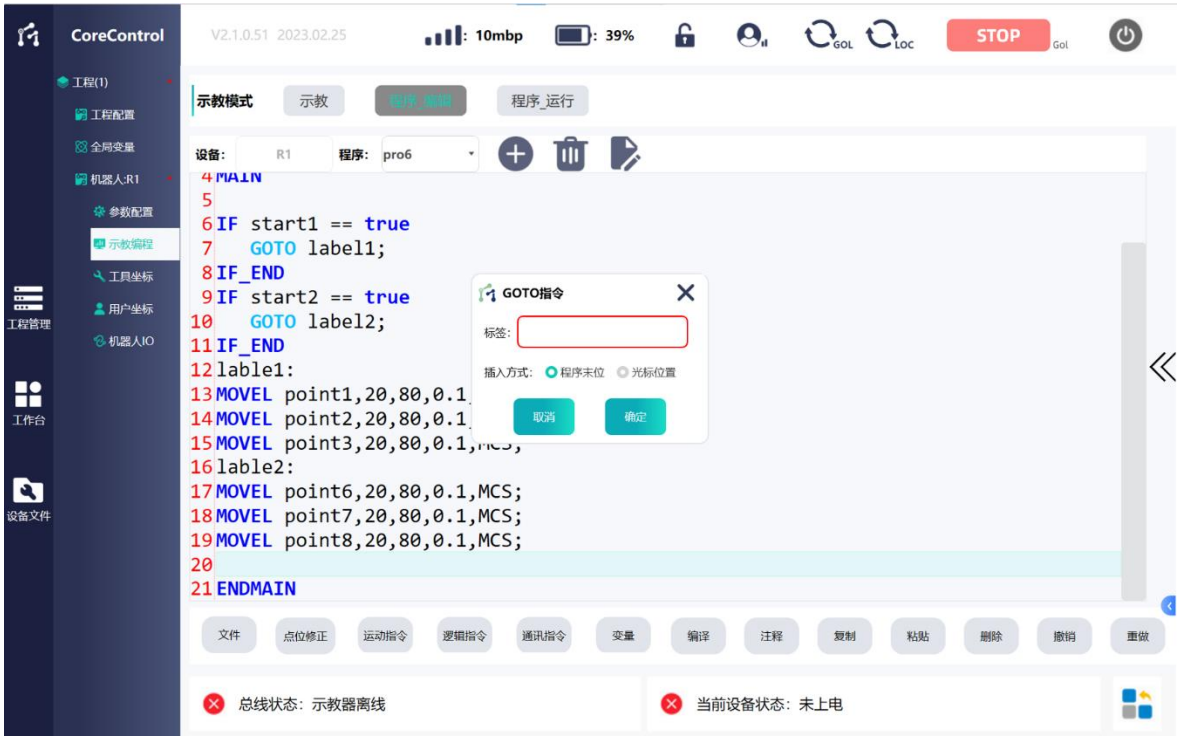


图 113 GOTO 逻辑指令使用示例

插入窗参数说明：

填入标签名称，名称可自定义。但是后续指定跳转目标时标签名称需要对应一致。

程序文本示例：

```
IF start1 == true
  GOTO label1;
IF_END
IF start2 == true
  GOTO label2;
IF_END
lable1:
MOVEL point1,20,80,0.1,MCS;
MOVEL point2,20,80,0.1,MCS;
MOVEL point3,20,80,0.1,MCS;
lable2:
MOVEL point6,20,80,0.1,MCS;
MOVEL point7,20,80,0.1,MCS;
MOVEL point8,20,80,0.1,MCS;
```

程序文本说明：

当变量 start1 为 TRUE，start2 为 FALSE 时，跳转到 label1 后的程序段，机器人依次执行运动到点 point1，point2, point3。然后因为跳转指令 2 的跳转条件未满足，会顺序执行后续程序。机器人再执行依次运动到点 point6 , point7 , point8。

当变量 start2 为 TRUE，start1 为 FALSE 时。跳转到 label2 后的程序段，执行依次运动到点 point6 , point7, point8。然后程序执行结束。（此时不会执行 label1 标签后的程序，因为跳转指令 2 执行后，程序直接跳过了 label1 后的程序段）。

注意

不允许跳转到嵌套程序段，如IF语句，WHILE语句，LOOP语句等。即跳转目标不可定义在这些语句的内部程序段内。

5.3.14 CLOCK

指令说明：

CLOCK 指令用于计时。关联计时器变量和长整型变量，实现开启计时，停止计时，复位计时时间，读取计时时间的功能。插入时分为 4 个子指令：

CLOCK START:开始计时；

CLOCK STOP:停止计时；

CLOCK RESET:复位计时时间；

CLOCK READ:读取计时时间；

CLOCK 指令格式：

CLOCK START x;

CLOCK STOP x;

CLOCK RESET x;

CLOCK READ x x;

//CLOCK START 计时器名称;**//**

//CLOCK STOP 计时器名称;**//**

//CLOCK RESET 计时器名称;**//**

//CLOCK READ 计时器名称 存储时间参数;**//**

编程方式：

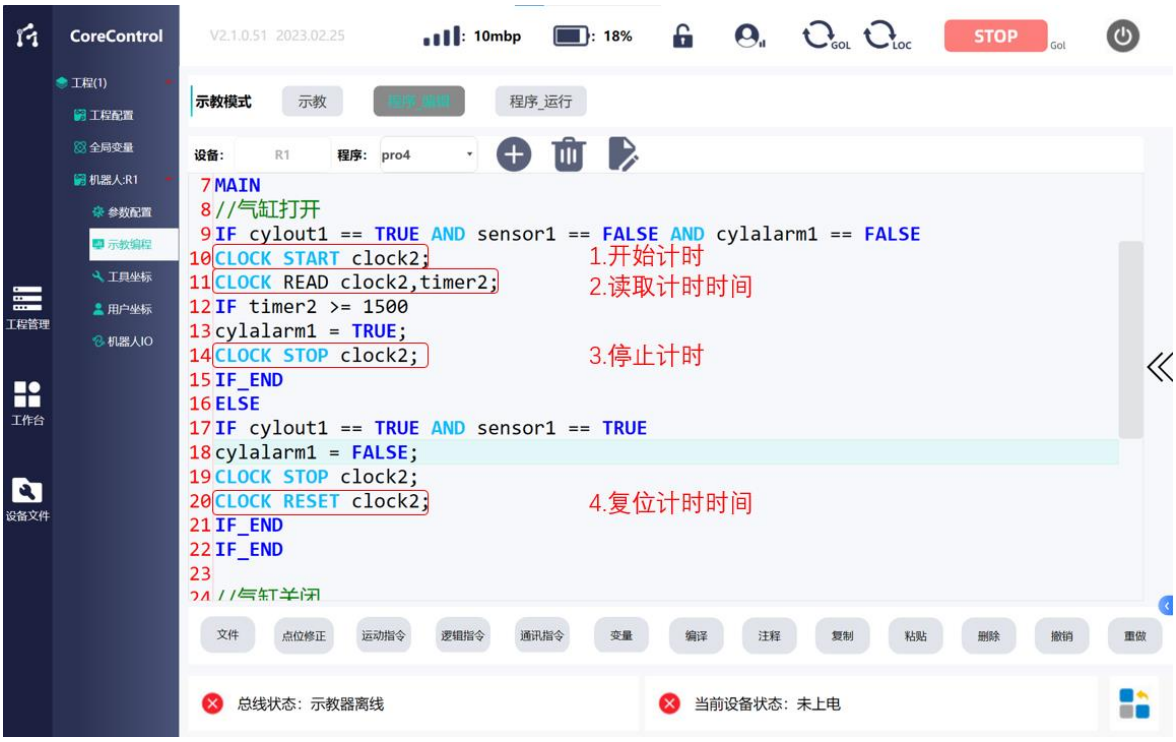


图 112 CLOCK 逻辑指令使用示例

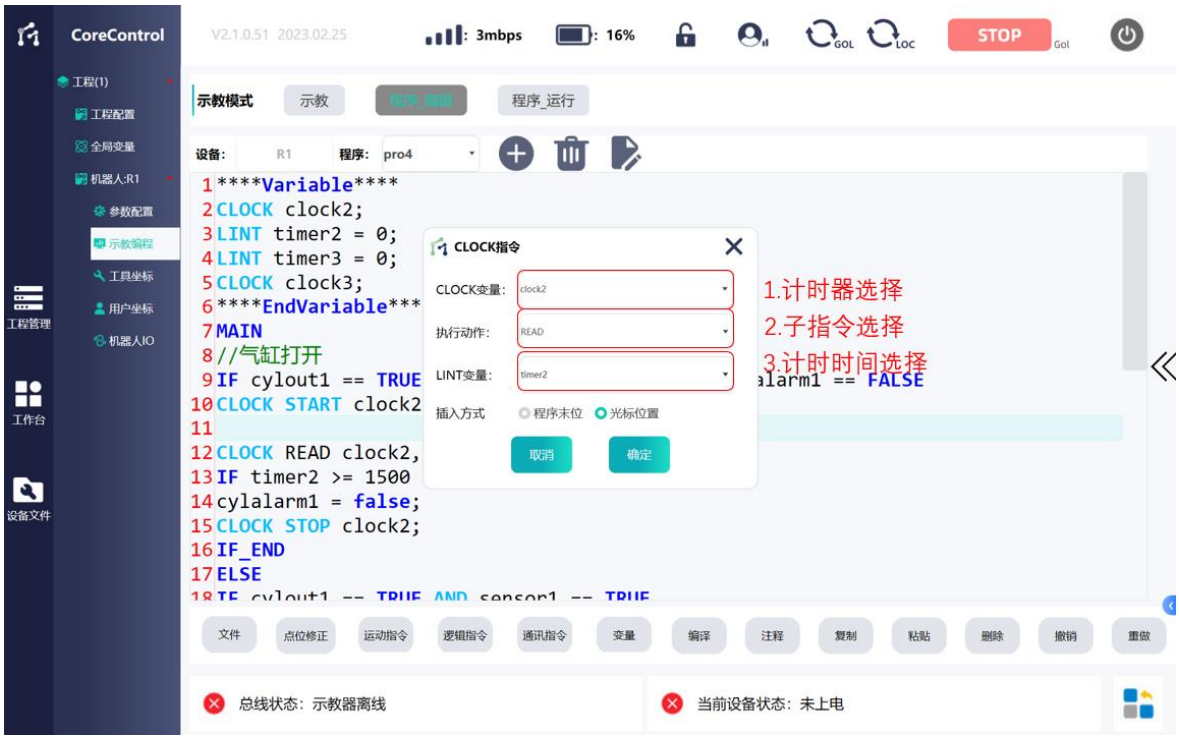


图 113 CLOCK 逻辑指令插入窗

插入窗参数说明：

1.计时器参数

选择事先声明的 CLOCK 局部变量。

2.子指令参数

选择 START、STOP、RESET、READ，分别实现不同的计时功能。

3.计时时间参数

选择事先声明的 LINT 型变量，用以存储计时器的计时时间。计时时间单位为 ms。

注意：只有 CLOCK READ 指令需要选择此参数。

程序文本示例：

```
IF cylout1 == TRUE AND sensor1 == FALSE AND cylalarm1 == FALSE
CLOCK START clock2;
CLOCK READ clock2,timer2;
IF timer2 >= 1500
cylalarm1 = true;
CLOCK STOP clock2;
IF_END
ELSE
IF cylout1 == TRUE AND sensor1 == TRUE
cylalarm1 = false;
CLOCK STOP clock2;
CLOCK RESET clock2;
IF_END
IF_END
```

程序文本说明：

此案例程序为气缸磁开传感器检测计时报警程序。设定当气缸电磁阀（cylout1）打开，气缸伸出时，气缸磁开传感器（sensor1）应该感应到位置位为 TRUE 的状态。若电磁阀打开后一段时间内（程序内设定为 1500ms），气缸磁开传感器（sensor1）仍未置位则发出报警（cylalarm1）。若气缸磁开传感器（sensor1）置位为 TRUE，则说明气缸恢复正常，则复位气缸报警（cylalarm1），并停止计时复位计时器计时时间。

具体逻辑为：

当 cylout1 为 TRUE 且 sensor1 为 FALSE，cylalarm1 为 FALSE 时。满足条件，开始计时（计时器 CLOCK2）并读取计时时间。判断 timer2 是否到达 1500ms，如果满足则将 cylalarm1 置位为 TRUE，并停止计时。若 sensor1 变为 TRUE，则 cylalarm1 置位为 FALSE，停止并复位 CLOCK2，timer2 复位为 0。

5.3.15 CALL

指令说明：

CALL 指令用于在主程序 main 中调用子程序用。主程序执行到此行语句后，会

跳转到相应的子程序内，从子程序的 MAIN 语句行开始执行，当执行完毕到子程序的 ENDMAIN 语句行后。再次跳转回主程序执行 CALL 语句下方的一条语句。

程序步指令格式：

CALL x;

//CALL 子程序名称;**//**

编程方式：

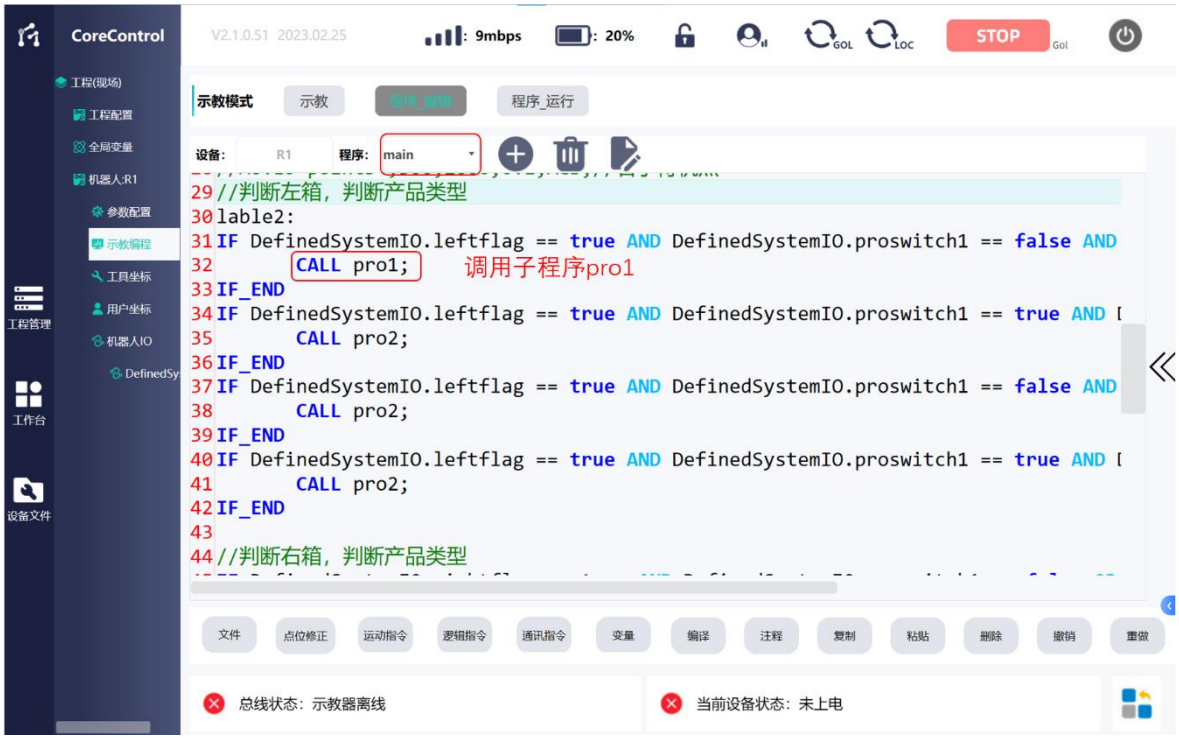


图 114 主程序 CALL 指令

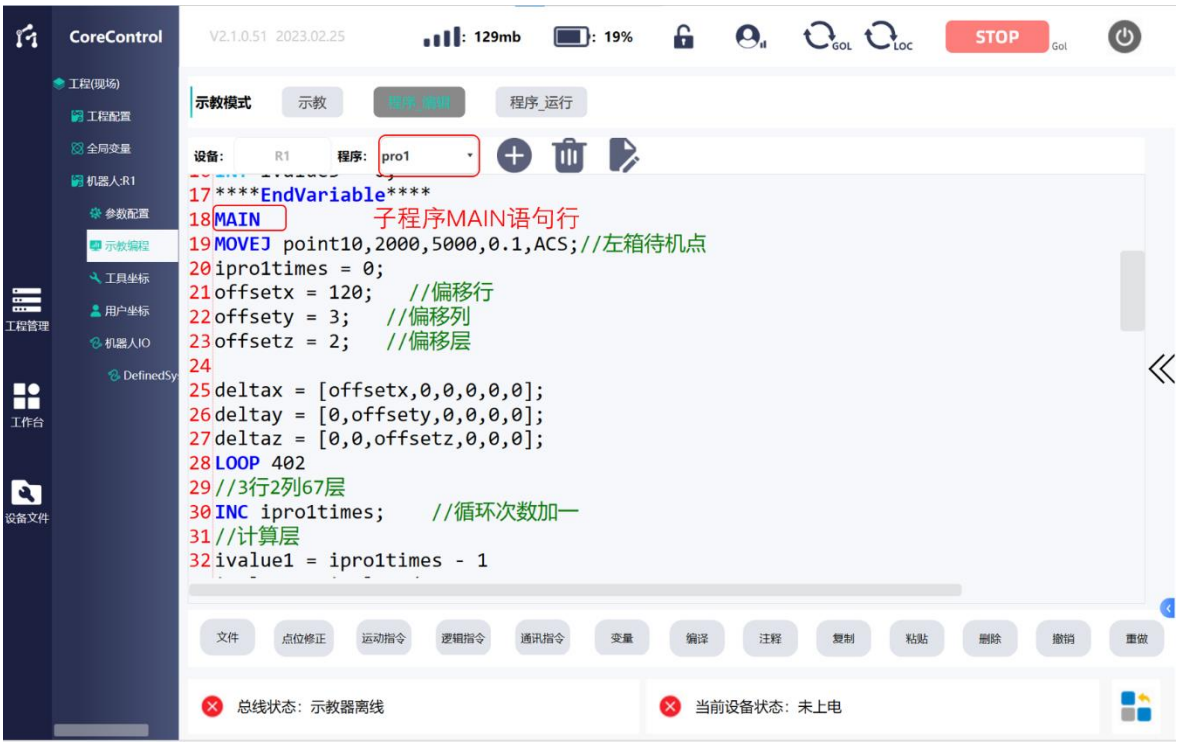


图 115 调用的子程序

程序文本示例：

IF DefinedSystemIO.leftflag == true AND DefinedSystemIO.proswitch1 == false AND DefinedSystemIO.proswitch2 == false

CALL pro1;

IF_END

IF DefinedSystemIO.leftflag == true AND DefinedSystemIO.proswitch1 == true AND DefinedSystemIO.proswitch2 == false

CALL pro2;

IF_END

IF DefinedSystemIO.leftflag == true AND DefinedSystemIO.proswitch1 == false AND DefinedSystemIO.proswitch2 == true

CALL pro2;

IF_END

IF DefinedSystemIO.leftflag == true AND DefinedSystemIO.proswitch1 == true AND DefinedSystemIO.proswitch2 == true

CALL pro2;

IF_END

程序文本说明：

根据 IO 变量的状态判断需要跳转到哪个子程序内。此案例为根据上位机发送的

IO 信号来执行不同的产品规格流程。

若 IO-leftflag 为 TRUE，proswitch1 为 FALSE，proswitch2 为 FALSE 时。执行开始执行子程序 pro1。子程序 pro1 从语句行 MAIN 下方的第一句开始执行，移动到点 point10。

5.3.15 NOTICE

指令说明：

NOTICE 指令用于在程序内进行提示。当执行到此行语句时，程序会暂停，并且弹窗提示 NOTICE 后的内容。

程序步指令格式：

NOTICE x;

//NOTICE 提示内容;**//**

编程方式：

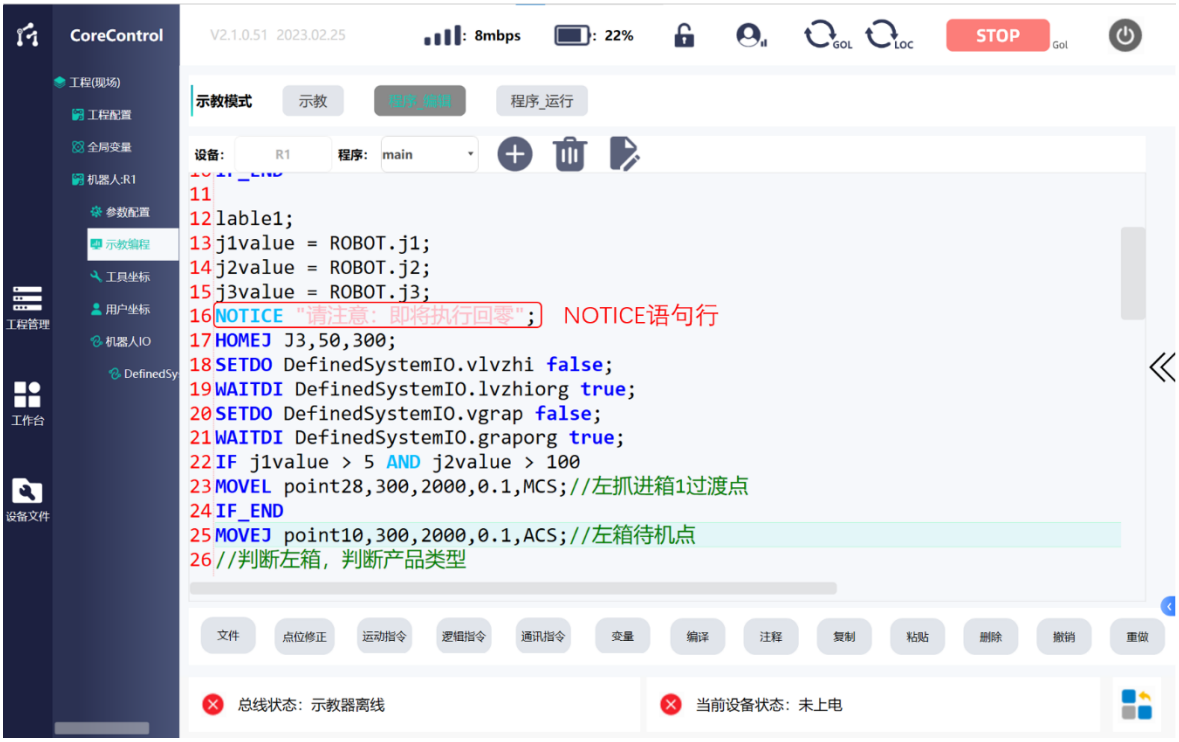


图 116 NOTICE 逻辑指令使用示例

程序文本示例：

lable1;

j1value = ROBOT.j1;

j2value = ROBOT.j2;

j3value = ROBOT.j3;

NOTICE "请注意：即将执行回零";

HOMEJ J3,50,300;

SETDO DefinedSystemIO.vlvzhi false;

WAITDI DefinedSystemIO.lvzhiorg true;

SETDO DefinedSystemIO.vgrap false;

WAITDI DefinedSystemIO.graporg true;

IF j1value > 5 AND j2value > 100

MOVEL point28,300,2000,0.1,MCS;//左抓进箱 1 过渡点

IF_END

程序文本说明：

当执行完读取机器人关节的当前位置后，开始执行 NOTICE 语句。程序暂停并弹窗提示“请注意：即将执行回零”。需要人工点击确认后程序方可继续执行。

5.4 字符串处理指令

5.4.1 STR_SPLIT

指令说明：

用于解析接收的字符串的指令，按设定的分隔符将一个长字符串分割为数个字符串，并将结果存储于字符串数组中。参数指定为需要解析的字符串和字符串分隔符。

解析后的字符串存储在一个字符串数组中，需要声明一个字符串数组用于存储这个字符串数据。字符串数组数据类型为 STRINGLIST。最多可分割出的字符串个数为 100 个。STRINGLIST 型数组变量的元素数量也为 100 个。

程序步指令格式：

STR_SPLIT(x, x) ;

//STR_SPLIT(字符串, 分隔符) ;**//**

应用示例及讲解：

参见 5.5.4 TCP 通讯示例。

注意

字符串变量不限制字符串长度。但是STR_SPLIT指令限制处理1~255之间的长度字符的字符串。

5.4.2 STR_SUB

指令说明：

用于从字符串的指定位置开始向右截取字符串，包含指定位置的字符。

程序步指令格式：

STR_SUB(x, x);

//STR_SUB(初始字符串，从第几位开始向右截取);**//**

应用示例及讲解：

参见 5.5.4 TCP 通讯示例。

5.5 通讯指令

5.5.1 TCP_CONNECT

指令说明：

用于连接 TCP 客户端（或服务器）的指令。本地为客户端时参数指定为服务器的 IP 地址和端口号。本地为服务器时参数指定为本地的 IP 地址和端口号。

编程方式：

程序步指令格式：

TCP_CONNECT x , x;

//TCP_CONNECT IP 地址，端口号;**//**

应用示例及讲解:

参见 5.5.4 TCP 通讯示例。

5.5.2 TCP_SEND

指令说明：

用于向 TCP 客户端（或服务器）发送字符串的指令，进行信息的交互。参数可以直接指定发送的具体字符串，也可以指定为字符串变量。

程序步指令格式：

TCP_SEND x;

//TCP_SEND 发送的字符串; **//**

应用示例及讲解:

参见 5.5.4 TCP 通讯示例。

5.5.3 TCP_READ

指令说明：

用于读取 TCP 客户端（或服务器）发送过来的字符串的指令。参数指定为字符串变量。

程序步指令格式：

TCP_READ x;

//TCP_READ 接收的字符串;**//**

应用示例及讲解:

参见 5.5.4 TCP 通讯示例。

5.5.4 TCP 通讯示例

编程方式:

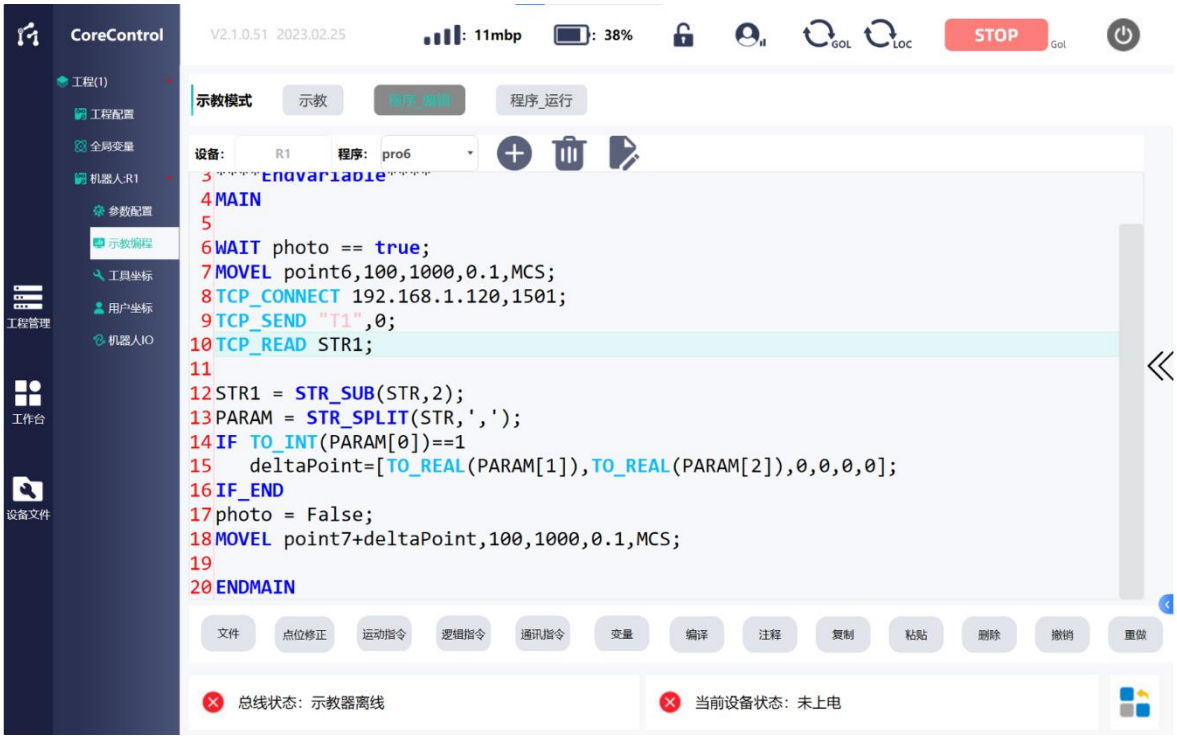


图 117 TCP 通讯指令使用示例

程序文本示例:

WAIT photo == true;

MOVEL point6,100,1000,0.1,MCS;

TCP_CONNECT 192.168.1.120,1501;

TCP_SEND "T1";

TCP_READ STR1;

STR1 = **STR_SUB**(STR,2);

PARAM = STR_SPLIT(STR,',');

IF TO_INT(PARAM[0])==1

deltaPoint=[TO_REAL(PARAM[1]),TO_REAL(PARAM[2]),0,0,0,0];

IF_END

photo = False;

MOVEL point7+deltaPoint,100,1000,0.1,MCS;

程序文本说明:

本示例为与相机 TCP 通讯交互，得到相机拍照后发送出的物料位置的 X 偏移量和 Y 偏移量。相机作为服务器，本地控制器作为客户端。

本示例中相机侧触发拍照的字符串为"T1"，相机反馈信息字符串格式定义为"1(或 0)，x(REAL 常数)， x(REAL 常数)，"，即 ：相机是否拍照成功，X 偏移，Y 偏移。字符串之间用逗号隔开。

等待外部启动信号-变量 photo 变为 TRUE，机器人执行移动到点 point6（拍照点）。连接相机，相机 IP 地址为 192.168.1.120，端口号为 1501。向相机发送字符串"T1"，启动拍照。然后读取相机反馈的字符串，存储在字符串变量 STR1 里面。使用 STR_SUB 指令剔除掉前两个字符，然后再使用 STR_SPLIT 指令分割字符串 STR。判断分割处理后得到的字符串数组 PARAM 第一个元素是否为 1，即相机拍照是否成功。若拍照成功，则将字符串数组 PARAM 的第二个元素，第三个元素设定为点变量 deltaPoint 的 X 坐标和 Y 坐标。然后机器人移动到抓取点，抓取点为在点 point7 基础上偏移 X、Y 的偏移值后得到的新的位置点。

5.6 数学运算符

运算符	功能
+	加
-	减
*	乘
/	除
TRUNC	截尾取整
ROUND	四舍五入
FLOOR	向下取整
MATH.SIN	正弦函数
MATH.COS	余弦函数
MOD	取余数

5.6.1 四则运算

例如加法运算：i=i+1; 把 i 的值加 1 再次赋值给 i。如 i 的当前值为 5，执行此段程序后 i 的值将变为 6。

5.6.2 取整

截尾取整：i=TRUNC(Rel1); 把实数 Rel1 的值去掉小数点后的截取，保留整数部分。如 Rel1 的当前值为 8.9，执行程序后，返回的 i 的值为 8。

四舍五入：i=ROUND(Rel1); 将实数 Rel1 四舍五入后将值赋值给整数 i。如 Rel1 的当前值为 123.456，执行程序后返回 i 的值为 123。

向下取整：i=FLOOR(Rel1); 将实数 Rel1 向下取整后将值赋值给整数 i。与 TRUNC 不同的是 FLOOR 总是取小于等于当前值的最大整数。如 Rel1 的当前值为-8.9, 执行程序后返回的 i 值为-9。

5.6.3 正余弦函数

rel1=MATH.SIN(rel2);计算 rel2 的正弦值并赋值给 rel1。

rel1=MATH.COS(rel2);计算 rel2 的余弦值并赋值给 rel1。

编程方式：

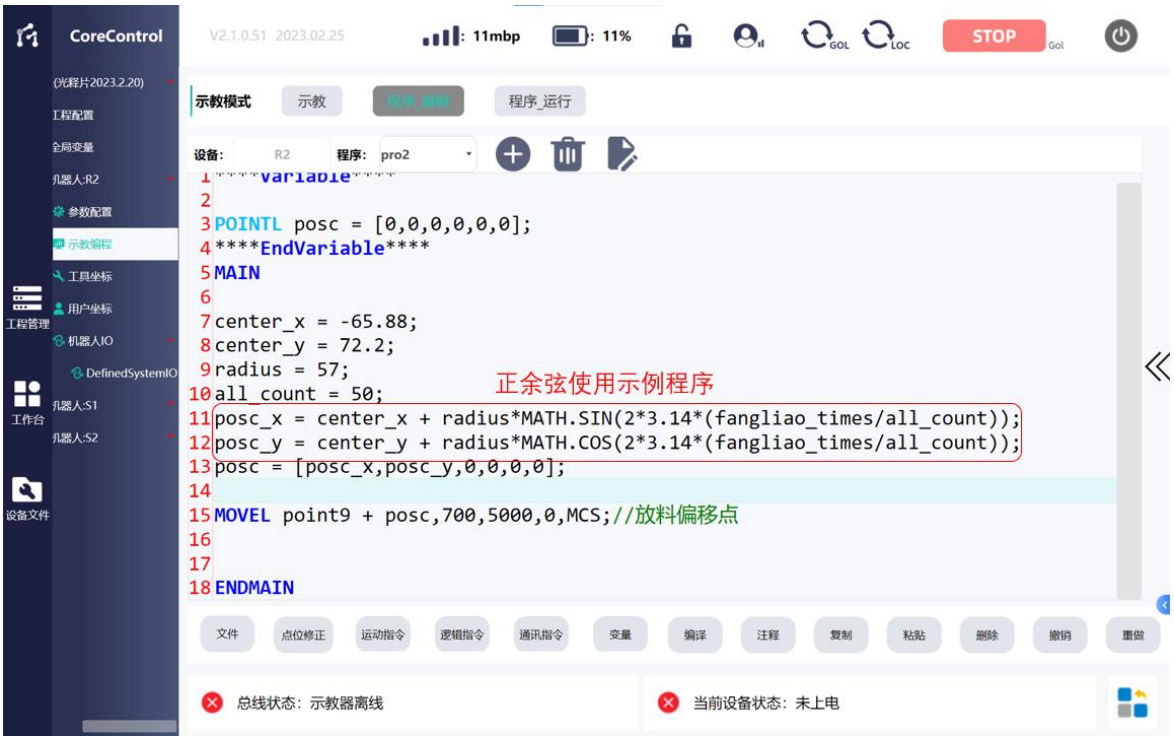


图 118 正余弦函数使用示例

程序文本示例：

center_x = -65.88;

center_y = 72.2;

radius = 57;

all_count = 50;

posc_x = center_x + radius*__MATH.SIN__(2*3.14*(fangliao_times/all_count));

posc_y = center_y + radius*__MATH.COS__(2*3.14*(fangliao_times/all_count));

posc = [posc_x,posc_y,0,0,0,0];

__MOVEL__ point9 + posc,700,5000,0,MCS;//放料偏移点

程序文本说明：

此段程序为圆形码垛编程。根据码垛圆的圆心、半径、总放料个数、当前放料次数求得当前放料点的 X 坐标、Y 坐标。然后使用点偏移功能移动机器人至求出的坐标点上。

先给 4 个实数型变量赋值。分别为圆心的 X 坐标、Y 坐标、半径和总放料个数。然后根据公式使用正弦函数和余弦函数求出当前放料点的 X 坐标和 Y 坐标，赋值给实数型变量 posc_x 和 posc_y。然后 posc_x 和 posc_y 组合成一个 POINTL 变量（即笛卡尔坐标系下的点位）posc。最后使用 MOVEL 指令偏移功能移动到点位 posc 上（point9 示教为坐标全为 0 的点）。

5.6.4 取余数

i1= i2 MOD i3;将整数 i2 取 i3 的余数，余数结果存储到整数 i1 中。

编程方式：

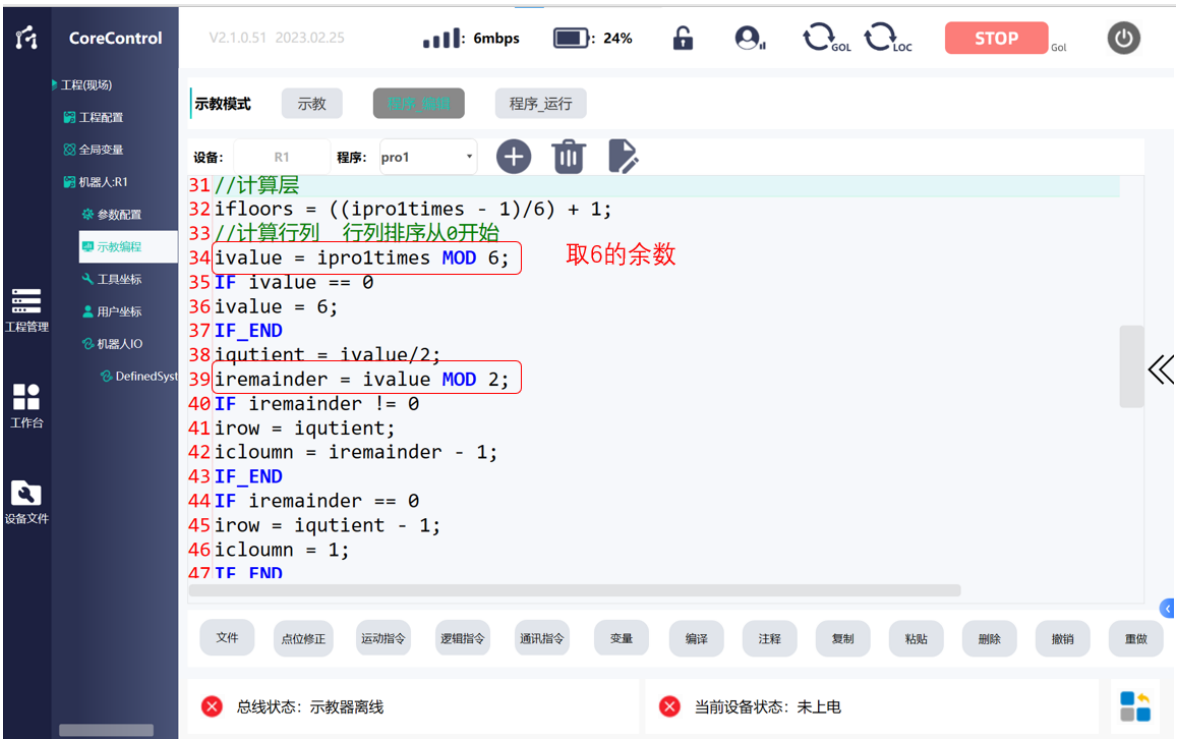


图 119 取余数运算符使用示例

程序文本示例：

```
INC ipro1times;
//计算层
ifloors = ((ipro1times - 1)/6) + 1;
//计算行列 行列排序从 0 开始
ivalue = ipro1times MOD 6;
```

```
IF ivalue == 0
ivalue = 6;
IF_END
iqutient = ivalue/2;
iremainder = ivalue MOD 2;
IF iremainder != 0
irow = iqutient;
icloumn = iremainder - 1;
IF_END
IF iremainder == 0
irow = iqutient - 1;
icloumn = 1;
IF_END
```

程序文本说明：

此段程序为 3 行 2 列 67 层的码垛案例程序，行列序数从 0 开始，层序数从 1 开始（具体起始序数根据使用环境决定）。根据程序循环次数(ipro1times)计算当前循环码垛的行列层。

循环次数由 ipro1times 变量每次循环自增得出。