**示教器使用说明书**

**目录**

[1 示教器简介 1](#_Toc19841)

[1.1 界面布局 1](#_Toc32577)

[1.2 常用按键介绍 2](#_Toc16436)

[2 功能模块 3](#_Toc2060)

[2.1用户自定义模块 3](#_Toc27127)

[2.2 配置管理模块 4](#_Toc9680)

[2.3 变量管理模块 5](#_Toc2985)

[2.3.1 变量的新建 6](#_Toc4965)

[2.3.2 变量的删除 8](#_Toc9)

[2.3.3 变量的复制 8](#_Toc24454)

[2.3.4 变量的剪切 9](#_Toc32259)

[2.3.5 变量的重命名 9](#_Toc9248)

[2.3.6 变量过滤器的使用 10](#_Toc17336)

[2.4 项目管理模块 14](#_Toc11030)

[2.4.1项目和程序的新建 15](#_Toc28348)

[2.4.2项目和程序的删除 16](#_Toc4600)

[2.4.3项目和程序的重命名 16](#_Toc2917)

[2.4.4程序的加载 17](#_Toc11435)

[2.4.5程序的打开 17](#_Toc15202)

[2.4.5程序的终止 18](#_Toc9747)

[2.5 程序管理模块 19](#_Toc5681)

[2.5.1程序的编辑 20](#_Toc5943)

[2.5.2程序的指令插入 22](#_Toc14900)

[2.6 位置管理模块 27](#_Toc20612)

[2.7 信息报告管理模块 28](#_Toc31597)

[3 运动指令组 29](#_Toc11794)

[3.1 PTP 29](#_Toc17715)

[3.2 Lin 29](#_Toc22292)

[3.3 Circ 31](#_Toc11859)

[3.4 PTPRel 32](#_Toc2532)

[3.5 LinRel 33](#_Toc31612)

[3.6 StopRobot 34](#_Toc24898)

[3.7 回零指令组 35](#_Toc27123)

[3.7.1 RefRobotAxis 35](#_Toc9412)

[3.7.2 RefRobotFinished 36](#_Toc11476)

[3.7.2 WaitRefFinished 36](#_Toc1416)

[3.8 WaitIsFinished 36](#_Toc6511)

[3.9 WaitJustInTime 36](#_Toc29089)

[3.10 MoveRobotAxis 36](#_Toc9093)

[3.11 PTPSearch 37](#_Toc15328)

[3.12 Linsearch 37](#_Toc4203)

[4 设置指令组 38](#_Toc737)

[4.1 Dyn 38](#_Toc16489)

[4.2 Dynovr 38](#_Toc22919)

[4.3 Refsys 39](#_Toc10176)

[4.4 Tool 39](#_Toc1097)

[4.5 Ovl 39](#_Toc7492)

[4.6 Ramp 40](#_Toc31233)

[4.7 OriMode 40](#_Toc4720)

[5 系统功能指令组 41](#_Toc3101)

[5.1 WaitTime 41](#_Toc10582)

[5.2 Stop 41](#_Toc31069)

[5.3 Info 41](#_Toc32479)

[5.4 ...:...(赋值) 42](#_Toc1928)

[5.5 //...(注释) 42](#_Toc29745)

[5.6 Warning 42](#_Toc25530)

[5.7 Error 42](#_Toc22592)

[5.8 Random 42](#_Toc17981)

[5.9 时间计量命令 42](#_Toc6857)

[5.10 数学运算指令 42](#_Toc32266)

[5.11 位运算指令及转换指令 42](#_Toc15437)

[6 流程控制指令 43](#_Toc26975)

[6.1 CALL 43](#_Toc2762)

[6.2 Wait 43](#_Toc19478)

[6.3 IF...THEN..END\_ID,ELSEIF..THEN,ELSE... 43](#_Toc31035)

[6.4 WHILE...DO...END\_WHILE 45](#_Toc14751)

[6.5 LOOP...DO..END\_LOOP 45](#_Toc4142)

[6.6 RUN,KILL 46](#_Toc14275)

[6.7 RETURN 46](#_Toc10221)

[6.8 GOTO...IF...GOTO..LABEL 46](#_Toc10404)

[7 输入输出指令组 49](#_Toc24846)

[8 触发指令组 50](#_Toc15763)

**1 示教器简介**

* 1. **界面布局**

该示教器是自主研发的人机界面交互软件，该人机界面具有易操作、人性化、符合人机工程学的特点。



图1.1 示教器主界面

如图1.1所示为示教器的主界面，图左侧分别为状态和配置配置管理部分，而右侧按钮为机器人动作操作按钮，底部的按钮时调节部分。左侧4个状态显示栏表示了系统运行状态，系统正常启动为RUN状态栏变绿，并显示为RUN\_ON，发生错误时EPR\_OFF会变绿，并显示为EPR\_ON。机器人上电时PWR\_OFF会变绿，并显示为PWR\_ON。

左侧7个按钮，分别为7个模块界面的切换按钮，自上而下依次为用户自定义模块、配置管理模块、变量管理模块、项目管理模块、程序管理模块、位置管理模块、信息报告模块。右侧机器人动作操作部分，通过按“+”与“-”按键可以在编程或者点动时调节机器人的坐标位置，点击换页按钮可以翻到下一页(附加轴页)。开始和停止按钮与程序运行和停止有关。

底部伺服上电按钮用于机器人上电或下电，切换坐标按钮用于切换机器人坐标系(轴坐标系、世界坐标系、工具手坐标系)，单步/连续按钮用于切换程序进入单步模式还是连续模式。V-和V+用于调节机器人运动速度。

在主界面顶部，有一个状态栏，它包含了机器人的操作模式、机器人状态及名称、坐标系、运动调节速度、项目程序名称、程序状态及执行模式、急停开关的状态、使用者等级等有关机器人系统状态方面的信息。具体如下图所示：



图1.2 主界面状态栏部分

* 1. **常用按键介绍**

常用按键如下：

1、手动/自动开关：有三种工作模式的切换

1. 示教模式：示教机器人，此时菜单栏中会显示T
2. 自动模式：执行程序，机器人自动运行，该模式下无法接受外部的开始信号，此时菜单栏会显示A
3. 远程模式：通过外部信号进行的操作，相当于遥控，该模式下不再接受示教编程器的开始指令。

2、开始按钮：为自动模式下程序的开始

3、停止按钮：为自动模式下程序的暂停

4、伺服上电按钮：为自动模式下的上下伺服

5、切换坐标按钮：用于切换坐标系，一共有三种坐标系

1）关节坐标系，此时界面将显示A1、A2、A3、A4、A5、A6

2）世界坐标系，此时界面将显示X、Y、Z、A、B、C

3)工具手坐标系：不常用，一般用于坐标系的标定

6、单步/连续切换按钮：用于切换程序进入单步模式还是联系模式

7、V-、V+按钮：用于调节机器人的运动速度。

8、急停按钮：出现意外状况时，点击该按钮终止程序的运行。

9、上图左上角的按钮：用于切换各个模块的界面，目前考虑如下模块

1）变量管理模块：该模块可观察当前所有的项目变量和程序变量，操作者可在该模块界面下新建变量，并且进行示教，为新建的变量记录当前机器人的位置，从而在程序管理模块下直接调用指令。

2）项目管理模块：负责项目和程序的新建，以及程序的加载、关闭、打开、删除等操作

3）程序管理模块：在示教模式下，对加载的程序进行指令的新建、指令变量的示教，以及在自动模式下，程序的单步和连续运行。

4）位置管理模块：对机器人进行关节坐标系和直角坐标系下的点动操作，相应的位置变化将在该模块界面上实时观测到。

**2 功能模块**

接下来将对示教器的7大模块中各个控件的作用和功能进行详细的阐述。

**2.1用户自定义模块**

**2.2 配置管理模块**

**2.3 变量管理模块**

点击变量管理按钮，进入变量管理模块界面，界面中将分布着三大类型的变量，分别为全局变量、项目变量以及程序变量，下面对三种变量的概念进行阐述：

1. 全局变量：适用于所有项目的所有程序的数据变量，即在加载任一项目的任一程序中，插入指令时，可调用相应数据类型的全局变量。
2. 项目变量：适用于当前项目的所有程序的数据变量，即在当前项目下加载任一的程序，插入指令时，可调用相应数据类型的项目变量。
3. 程序变量：适用于当前程序的数据变量，即只在加载当前项目的当前程序时，插入指令时，可调用相应数据类型的程序变量。

明确三种数据类型的定义后，将对变量管理模块下的各个控件的作用进行阐述：



图2.1 变量管理模块界面

如图2.1所示为变量管理模块的界面，全局变量的节点时唯一的，并可看到目前还有两个项目变量project1和project2，以及四个程序变量test1[project1]、test2[project1]、test3[project2]、test4[project2]，通过中括号的形式可以清楚地看到各个程序变量都属于哪一项目。

变量管理界面下还有如下控件：

1. 变量按钮：点击该按钮会展开新建、删除、复制、粘贴、剪切、重命名选项，用以对某项目或子目录下的变量进行操作。
2. 示教按钮：点击该按钮，用于示教保存在程序运行过程中需要的位置。(功能未实现)
3. 清楚未用变量按钮：点击该按钮，可以删除所有没有使用的变量。(功能未实现)
4. 变量类型过滤器：选择相应的数据类型，将过滤选中的数据类型，方便使用者知道选中类型的数据变量在全局、项目和程序变量下的分布情况。

目前变量管理模块支持十种数据类型变量的操作，分别是INTEGER、FLOAT、BOOL、STRING、DTNAMIC、AXIPOS、WORLDREFSYS、CARTREFSYS、CARTPOS、OVERLAP。

**2.3.1 变量的新建**

以FLOAT类型全局变量的新建为例，对示教器变量管理模块的操作进行阐述：



图2.2 新建FLOAT类型变量

如图2.2所示，在变量类型范围为<全部>的条件下，首先选中全局变量节点，此时全局变量条目变成灰色，然后点击变量按钮，选择新建选项，此时将弹出一个变量新建的对话框，输入变量名，选择数据类型为FLOAT类型，点击确定，此时在全局变量节点下将看到所新建的变量，如下图所示：

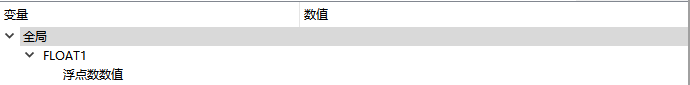


图2.3 新建FLOAT类型变量显示情况

如图2.3所示，全局变量节点下有一个名为FLOAT1的变量节点，在FLOAT1变量节点下显示的为该变量名的数据类型，可以通过点击变量左边的下三角按钮，对节点进行展开和隐藏的操作，此时若想给该变量赋值，可以点击该模块的设置按钮，具体操作步骤为：选中“浮点数数值”，此时该条目将变成灰色，然后点击设置按钮，将弹出一个设置参数数值的对话框，如下图所示：

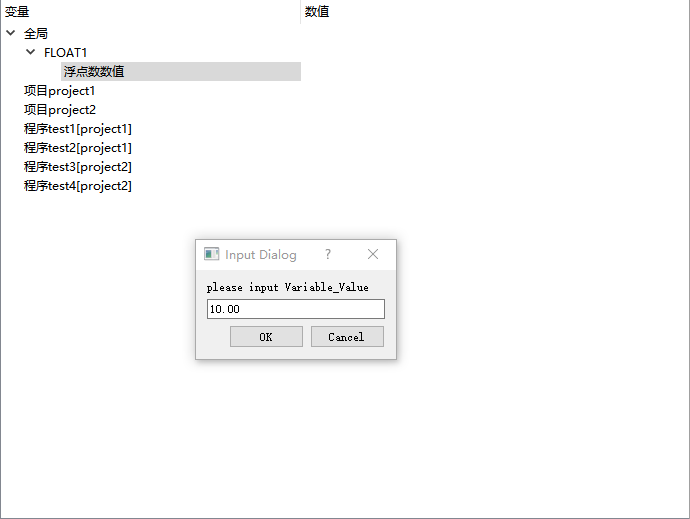


图2.4 设置FLOAT变量数据类型数值

如图2.4所示，点击设置按钮，将弹出一个设置选中参数数值的对话框，输入数值为10.00，点击确定，所设置的数值将在选中的“浮点数数值”旁边的数值条目中显示，其他的参数数值设置操作以此类推。

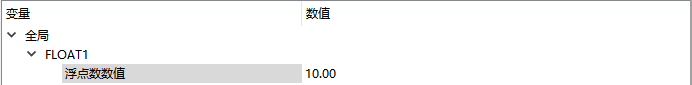


图2.5 FLOAT变量数值显示情况

如图2.5所示，新建FLOAT变量成功，并且成功设置其数值类型为10.00，若想改变FLOAT1变量的数值，点击设置按钮，重新输入想要改变的数值即可。

到此为止，FLOAT类型变量的新建操作结束，若想新建其他数据类型的变量，操作和新建FLOAT变量类似，这里将不再进行详细的阐述。

**2.3.2 变量的删除**

以FLOAT类型全局变量的删除为例，对示教器变量管理模块的操作进行阐述：

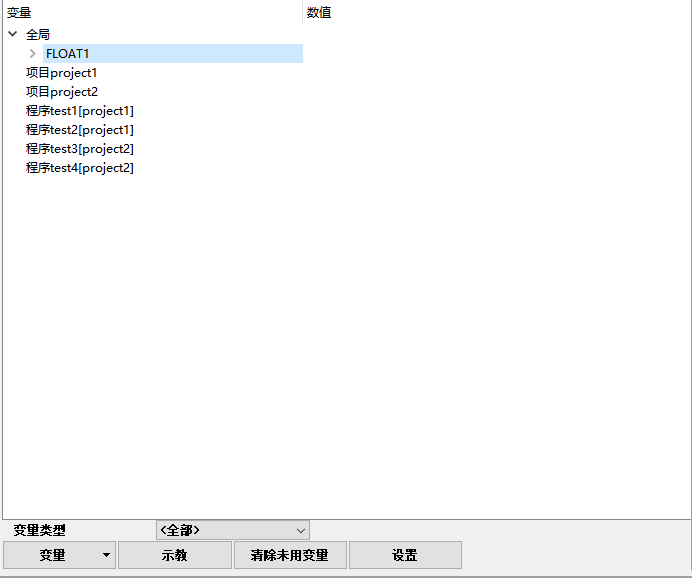


图2.6 删除FLOAT数据类型变量

如图2.6所示，全局变量节点下分布一个名称为“FLOAT1”的数据类型，若想删除该变量，只需要选中该变量名，如上图所示，“FLOAT1”呈现选中状态，此时点击变量按钮，选择删除选项，即可成功删除所选中的变量。

**2.3.3 变量的复制**

以FLOAT类型项目变量的复制为例，对示教器变量管理模块的操作进行阐述：

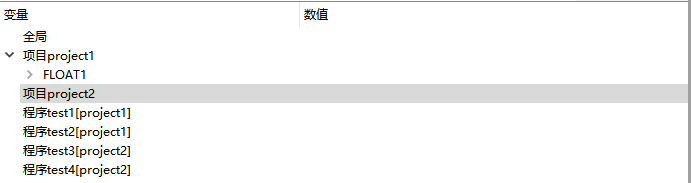


图2.7 变量管理模块显示情况

如图2.7所示，此时在“项目project1”节点下有一名称为“FLOAT1”的浮点型数据类型变量，若想在“项目project2”下所有的程序中都可以调用该变量，可以进行变量的复制来实现。具体的操作为：首先选中想要复制的变量名称，这里为“FLOAT1”，然后点击变量按钮，选择复制选项，然后选中该变量的作用域，这里为“项目project2”，然后点击变量按钮，选择粘贴选项，此时将成功在“项目project2”节点下新建一名为“FLOAT1”的变量，它的名称和数值将和“项目project1”节点下的“FLOAT1”变量一样。如下图所示：

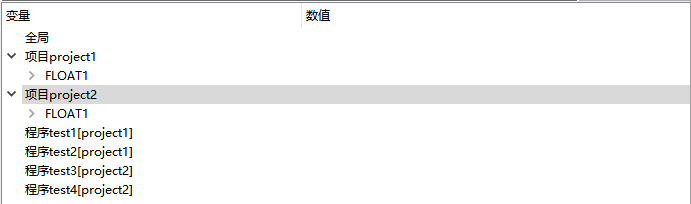


图2.8 复制FLOAT1变量

如图2.8所示，成功将“项目project1”节点下的FLOAT1复制到了“项目project2”节点下，因此，在加载project2下的任一程序时，都可以调用该变量，提高操作和开发的效率。

**2.3.4 变量的剪切**

变量的剪切操作和变量的复制操作是一样的，以上文所述复制FLOAT1变量为例，变量的剪切和复制区别在于，剪切变量之后，原来“项目project1”下的FLOAT1变量将被删除，而复制变量，将不删除原来“项目project1”下的FLOAT1变量，具体对变量进行复制还是剪切操作，将视具体开发情况而定。

**2.3.5 变量的重命名**

以FLOAT类型项目变量(FLOAT1)的重命名为例，对示教器变量管理模块的操作进行阐述：

首先，选中想要进行重命名的变量名称，此时该变量将呈现被选中状态，然后点击变量按钮，选择重命名选项，此时将弹出一个变量重命名的对话框，如下图所示：

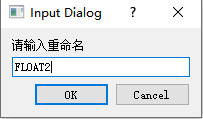


图2.9 变量重命名输入对话框

如图2.9所示，在输入框中输入重新命名的变量名称，点击OK按钮，原来的变量名称将进行改变，如下图所示：



图2.10 变量重命名

如图2.10所示，变量的重命名操作成功，之后在加载project1下的程序中，插入指令时，可以选择调用该名称的变量。

**2.3.6 变量过滤器的使用**

变量过滤器的作用在于，当全局、项目和程序变量新建了许多不同数据类型的变量，可以通过变量过滤器来筛选不同数据类型的变量，方便观察不同数据类型变量的数量和作用域。

如下图所示，为当前新建变量的情况：



图2.11 不同数据类型变量监测情况

若想筛选出数据类型为AXISPOS的变量，可以点击变量类型的下拉框，选择AXISPOS，此时，变量管理模块将只显示数据类型为AXISPOS的变量，如下图所示：

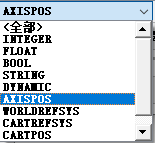


图2.12 筛选AXIPOS数据类型变量



图2.13 AXISPOS变量显示情况

如图2.13所示，当前界面显示只在“程序test2[project1]”变量下新建了一个名为AXISPOS1的AXISPOS数据变量。

需要注意的时，此时变量过滤器处于“AXISPOS”筛选状态，在该状态下若想新建变量，将只能新建AXISPOS类型的变量，其他情况类似，若此时变量过滤器处于“INTEGER”筛选状态，将只能新建INTEGER数据类型的变量。

因此，只有在变量过滤器处于“<全部>”筛选状态时，才可以新建任一数据类型的变量，当变量过滤器处于其他的筛选状态时，只能新建当前变量过滤器对应数据类型的变量，通过这种方式来新建变量的意义在于：提高变量新建的针对性和准确性，同时可以省去用户在新建变量时选择相应数据类型的操作，提高操作的效率和可靠性。

下面以变量过滤器处于“AXISPOS”筛选状态时，新建一个作用域为“项目project1”的变量为例，阐述变量新建的操作：首先选中对应的作用域，即“项目project1”,此时选中的条目将呈现选中状态，然后点击变量按钮，选择新建选项，将弹出如下图所示的对话框：

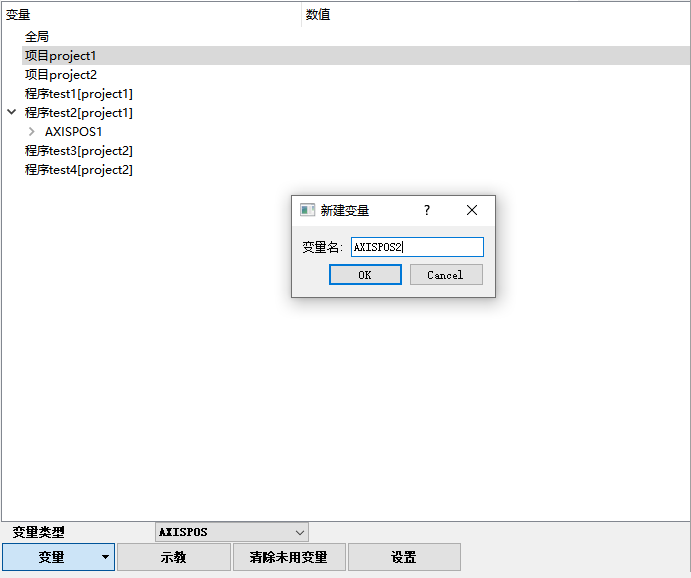


图2.14 AXISPOS筛选状态下新建变量

如图2.14所示可以清楚看到，变量新建对话框和变量过滤器处于“<全部>”筛选状态时的变量新建对话框的区别，此时由于变量过滤器处于“<AXISPOS>”筛选状态，所以此时新建变量将默认的数据类型为AXISPOS，因此，用户只需要输入变量的名称即可，点击OK按钮之后，所新建的变量如下图所示：

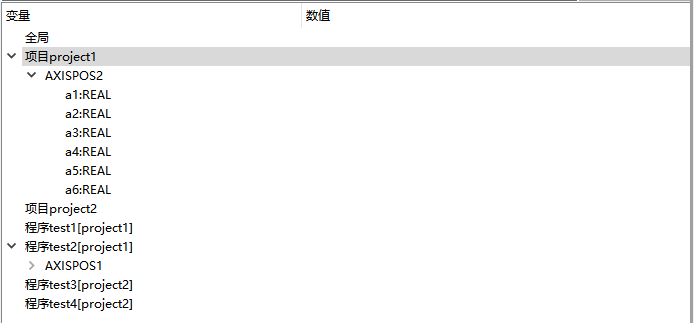


图2.15 AXISPOS下新建名为AXISPOS2的变量

如图2.15所示，点击OK按钮之后，将在“项目project1”节点下新建一名称为“AXISPOS2”的变量，该变量节点下为关节坐标系下6个轴的名称，若想设置各个轴的参数数值，只需选中对应的参数名称，点击设置按钮，输入数值，点击确定按钮即可。

**2.4 项目管理模块**

项目管理模块的作用主要是对项目和程序进行管理，下面对该模块下的控件操作进行阐述。



图2.16 项目管理模块界面

如图2.16所示，该模块界面下主要包含以下几个控件：

1. 加载按钮：选中对应的程序进行加载，此时将自动跳转到程序管理模块，可对程序进行任何操作，如程序的示教、编程和运行。
2. 打开按钮：选中对应的程序进行打开，此时将自动跳转到程序管理模块，并只能对程序进行可读操作。
3. 终止按钮：选中对应的程序进行终止，关闭正在打开或者加载的程序。
4. 信息按钮：暂未考虑
5. 刷新按钮：暂未考虑
6. 文件按钮：该按钮包含对项目、程序的新建、删除以及重命名操作。

**2.4.1项目和程序的新建**

如图2.16所示，当前项目管理模块下已存在两个项目，分别为project1和project2,project1下有程序test1和test2，project2下有程序test3和test4。

此时点击文件按钮，选择新建项目选项，将弹出项目新建对话框，如下图所示：

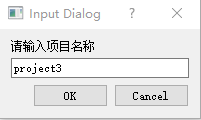


图2.17 项目新建对话框

如图2.17所示，在对话框中输入新建的项目名称，点击OK按钮，此时观察项目管理模块，可以看到新建了一个名为peoject3的项目，如下图2.18所示：



图2.18 新建project3项目

此时若想在project3项目下新建一个程序，具体操作为：选中project3，然后点击文件按钮，选择新建程序选项，将弹出程序新建对话框，如下图2.19所示：

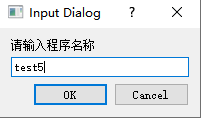


图2.19 程序新建对话框

如图2.19所示，在对话框中输入新建的程序名称，点击OK按钮，此时观察项目管理模块，可以看到在project3节点下新建了一个名为test5的程序，如下图2.20所示：

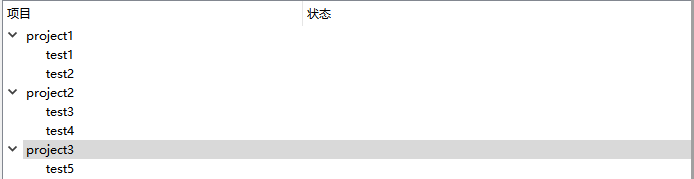


图2.20 新建test5程序

**2.4.2项目和程序的删除**

若想删除项目下程序，以“project3”下的程序“test5”为例，选中test5，点击文件按钮，选择删除选项，此时可成功删除该程序，该程序将从project3节点中移除，如下图2.21所示：

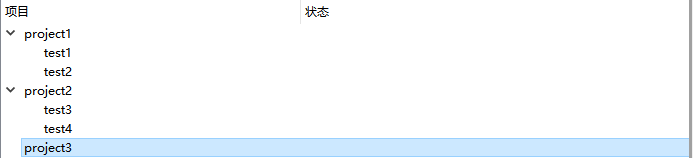


图2.21 删除test5程序

若想删除指定的项目，以“project3”为例，选中project3，点击文件按钮，选择删除选项，此时可成功删除该项目，该项目将从项目管理界面中移除，如下图2.22所示：

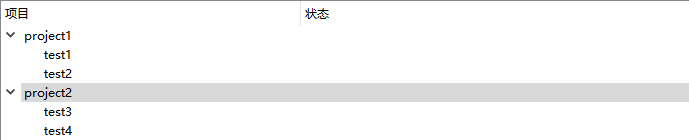


图2.22 删除project3项目

**2.4.3项目和程序的重命名**

若想对项目或者程序进行重命名，以project1项目下的test1为例，选中test1，点击文件按钮，选择重命名选项，此时将弹出一个重命名对话框，如下图2.23所示：

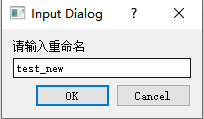


图2.23 重命名test1

如图2.23所示，输入程序的新名称，点击OK按钮，此时test1的名称将变为新输入的名称，如下图2.24所示：

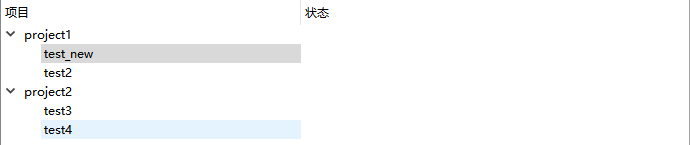


图 2.24 重命名test1后的显示情况

**2.4.4程序的加载**

在对程序进行加载操作后，将自动跳转到程序管理模块，在该模块中可对程序进行任何的操作，例如指令的编程，程序的编译以及程序的运行等等。以加载project1项目下的test\_new程序为例，进行阐述：

首先选中test\_new，点击加载按钮，此时程序将自动跳转到程序管理界面，如下图2.25所示：



图2.25 跳转到程序管理界面

此时返回项目管理界面，可以看到对应项目和程序的状态，由于当前加载的程序为test\_new程序，因此test\_new程序和其所在的项目project1将显示为加载状态，其余的项目和程序将显示为“----”状态，如下图2.26所示：

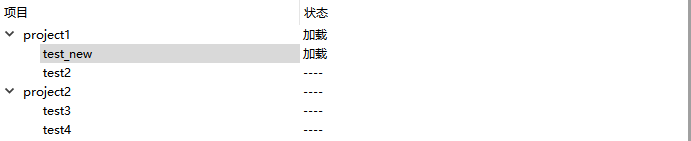


图2.26 项目和程序的加载状态显示

**2.4.5程序的打开**

在对程序进行打开操作后，将自动跳转到程序管理模块，此时用户对程序只有读权限，不可对程序进行任何的修改操作，以project1项目下的test\_new程序为例，选中test\_new，点击打开按钮，此时程序跳转到了程序管理模块，如下图所示：



图2.27 打开程序test\_new

如图2.27所示，此时打开test\_new程序，程序指令为空，并且程序管理模块的背景为灰色，该模块的各个控件也呈灰色，不可点击，证明在打开操作状态，用户对打开的程序只拥有读的权限。此时返回项目管理模块，也可看到项目和程序的状态，如下图2.28所示：

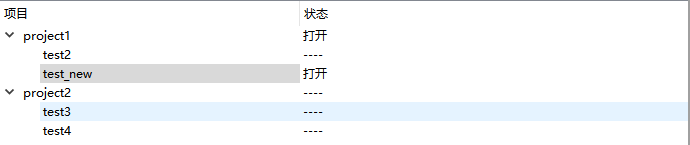


图2.28 项目和程序的打开状态显示

**2.4.5程序的终止**

若当前有程序处于加载或者打开的状态，可通过选中对应程序，点击终止按钮，清空程序管理模块下程序的显示，并且终止程序的编译、运行或者编程操作，下图2.29为项目管理模块的状态显示：

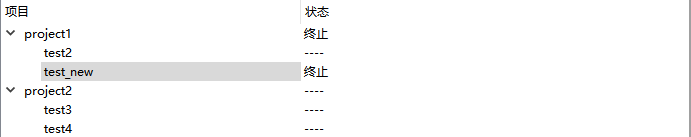


图2.29 项目和程序的终止状态显示

**2.5 程序管理模块**

程序管理模块的作用在于对加载的程序进行编程、编译、运行等操作，是7大模块中最为重要的一个模块。

以加载项目project1下的test\_new程序为例，加载的程序在程序管理模块下如图2.30所示：

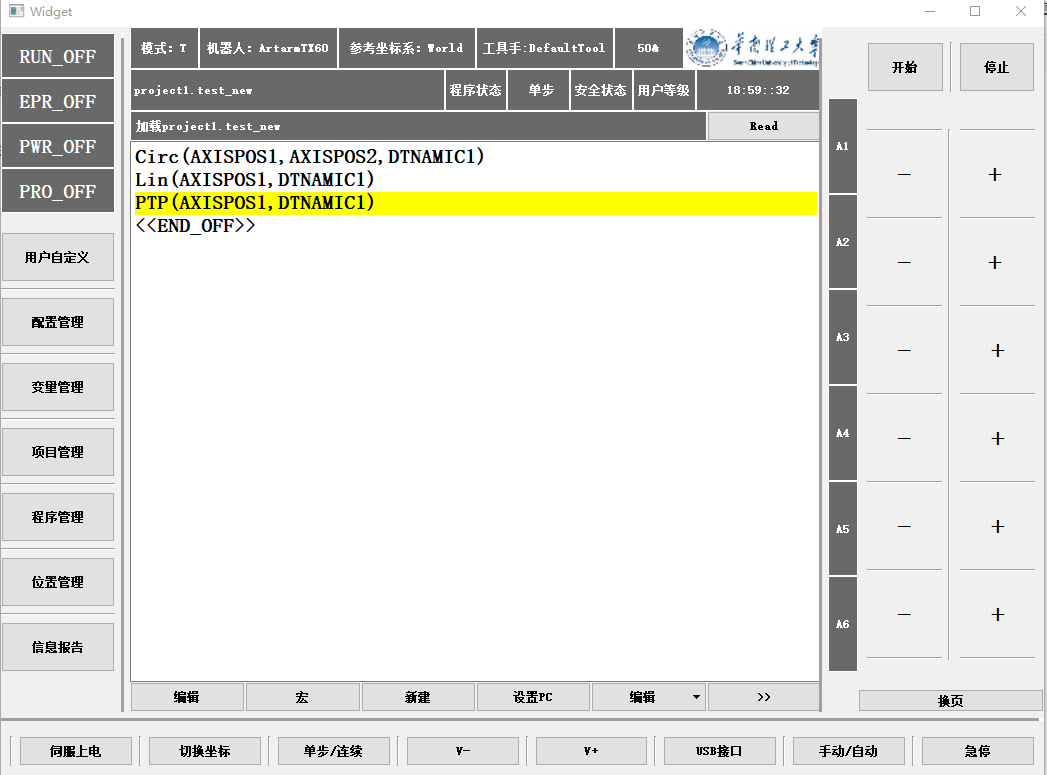


图2.31 程序管理模块界面

如图2.31所示为程序管理模块的界面，可以看到test\_new程序中已经插入了Circ、Lin、PTP三条指令，其中<<EOF\_OFF>>为结尾符，不具有编程意义，作用在于通过选中该行，在上行可插入指令。

程序管理模块底部的按钮作用如下：

1. 编辑按钮：点击该按钮，按钮的名称将显示为“确认”，此时用户可对程序进行任意的修改，修改完成之后，点击“确认”，将对所修改的操作进行保存。
2. 宏按钮：暂不考虑。
3. 新建按钮：点击新建按钮，可进入指令库中新建指令，并插入到程序当中。
4. 编辑按钮：该按钮和第一个编辑按钮具有不同的作用，该编辑按钮是以选中行为基础，对程序进行复制、粘贴、剪切、删除、注释、取消注释等操作。
5. >>按钮：该按钮为翻页按钮，暂不考虑。

**2.5.1程序的编辑**

若想在某行中添加注释，可通过第一个程序编辑按钮实现，这里以在PTP指令后添加一行注释为例，具体操作为：

首先选中PTP指令所在行，该行呈现黄色状态，然后点击程序管理模块下第一个编辑按钮，此时PTP所在行出现光标闪动，此时通过外设键盘或者虚拟键盘为该行添加注释，如下图2.32所示：

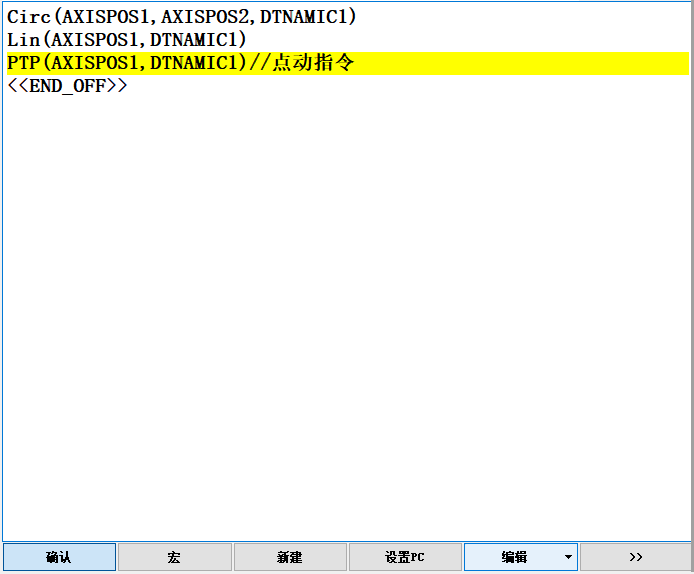


图2.32 为程序管理模块添加注释

如图2.32所示为程序添加注释时的情况，添加完注释之后，点击确认按钮，此时程序管理模块的一个按钮名称将变回“编辑”状态，同时将保存对程序的编辑操作，至此，第一个编辑按钮的演示完毕，若想对程序任意位置进行相应的编辑操作，都可以采取类似的操作完成。

同时，还可通过程序管理模块下的第5个按钮对程序进行编辑操作，这个按钮基于所选行对程序进行编辑操作，包括复制、粘贴、剪切、删除、注释、取消注释等。

这里以复制程序指令PTP到指令LIn上方为例，对程序的复制操作进行阐述：

首先选中PTP所在行，此时PTP所在行呈现黄色状态，然后点击程序管理模块下，带有下三角符号的编辑按钮，选择复制选项，然后选中lin指令所在行，此时lin指令所在行呈现黄色状态，最后点击编辑按钮，选择粘贴选项，此时便成功在Lin指令上方粘贴PTP指令，如下图2.33所示：

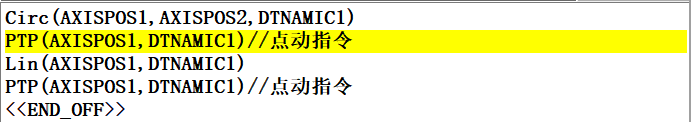


图2.33 程序管理模块下的复制操作

若想对程序进行剪切操作，可以通过类似的方式，选中剪切指令所在行，然后点击编辑按钮下的剪切选项实现，这里以剪切图2.33所示程序第二行的PTP指令到Circ指令上方为例，对程序的剪切操作进行阐述：

首先选中PTP所在行，此时PTP所在行呈现黄色状态，然后点击程序管理模块下，带有下三角符号的编辑按钮，选择剪切选项，此时选中的PTP指令所在行将消失，然后选中Circ指令所在行，此时Circ指令所在行呈现黄色状态，最后点击编辑按钮，选择粘贴选项，此时便成功在Circ指令上方粘贴PTP指令，如下图2.34所示：

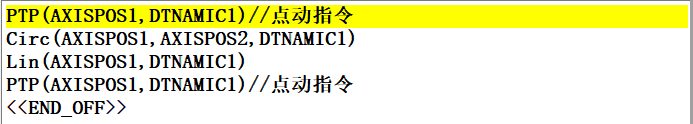


图2.34 程序管理模块下的剪切操作

若想对程序进行删除操作，以删除第一行的PTP指令为例，对程序的删除操作进行阐述：

首先选中PTP指令所在行，此时PTP所在行呈现黄色状态，然后点击程序管理模块下，带有下三角符号的编辑按钮，选择删除选项，此时便可成功删除PTP指令，如下图2.35所示：

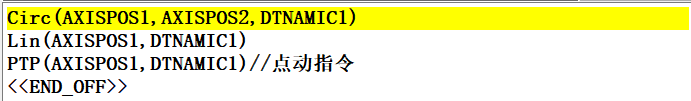


图2.35 程序管理模块下的删除操作

若想将程序中某行的指令注释掉，以注释程序中第一行Circ指令为例，对程序的注释操作进行阐述：

首先选中Circ指令所在行，此时Circ所在行呈现黄色状态，然后点击程序管理模块下，带有下三角符号的编辑按钮，选择注释选项，此时便可成功注释Circ指令，如下图2.36所示：

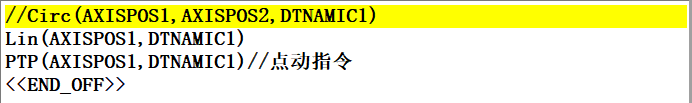


图2.36 程序管理模块下的注释操作

若想将程序中某行注释的指令取消注释，以取消程序中第一行Circ指令的注释为例，对程序的取消注释操作进行阐述：

首先选中Circ指令所在行，此时Circ所在行呈现黄色状态，然后点击程序管理模块下，带有下三角符号的编辑按钮，选择取消注释选项，此时便可成功将Circ指令所在行取消注释，如下图2.37所示：

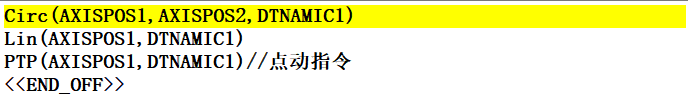


图2.37 程序管理模块下的取消注释操作

**2.5.2程序的指令插入**

若想在程序某行指令上方，插入新的指令，可通过选中指令所在行，点击新建按钮完成相应的指令插入操作。

点击新建按钮之后，会进入到如下图2.38所示的指令选择界面：

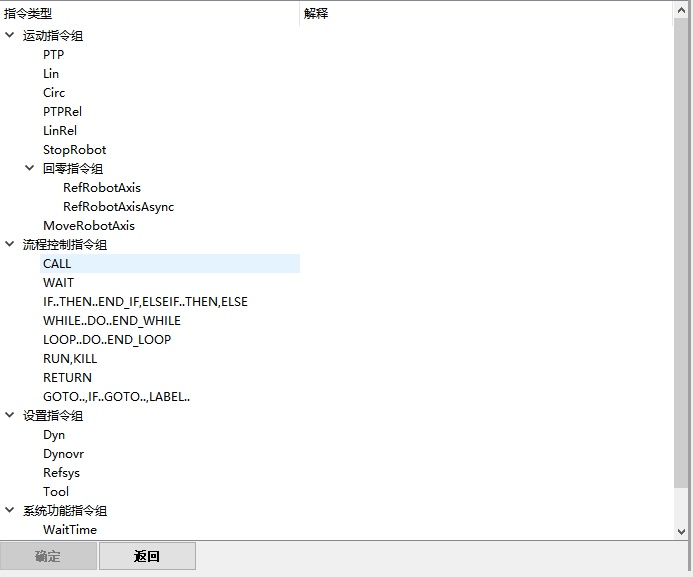


图2.38 程序管理模块下的指令选择界面

如图2.38所示为指令选择界面，可以看到界面上显示了许多指令，大概分为四类，分别是：运动指令组、流程控制指令组、设置指令组、系统功能指令组，每类指令组下都包含若干具体的指令，各个指令指令将在相应的指令章节进行详细的阐述，在指令选择界面底部有两个按钮，分别是确认和返回按钮，作用如下：

1. 确定按钮：选中对应的指令，点击确定，将进入到指令对应的设置界面中，从而进行下一步的操作，注意，该按钮只有在选中指令时，才具有点击的权限，一般情况下显示为灰色状态，不可点击。
2. 返回按钮：点击该按钮，返回到程序管理模块的程序编辑界面。

由于指令较多，具体各个指令的插入将在对应指令章节中进行详细地介绍。这里只以PTP指令的插入为例，对指令的插入操作进行阐述：

首先，选中运动指令组下的PTP指令，此时PTP指令呈现选中状态，用户获得对确定按钮的点击权限，点击确定按钮，进入PTP指令的设置界面，如下图2.39所示：

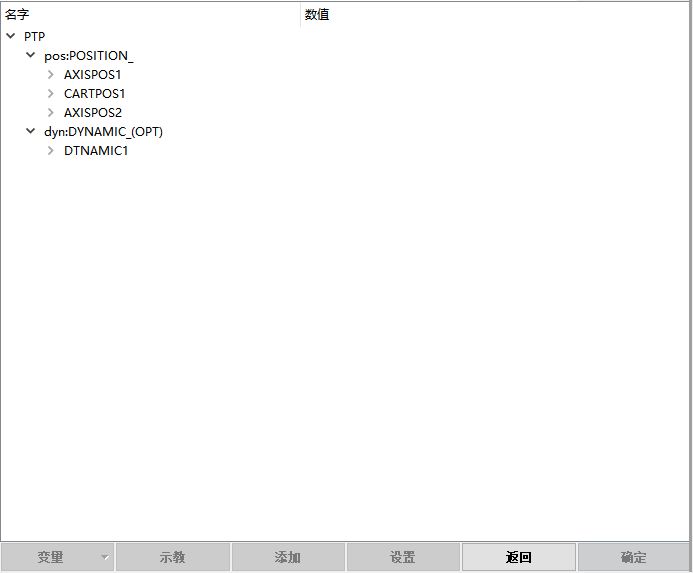


图 2.39 PTP指令设置界面

如图2.39所示，PTP指令设置界面具有若干按钮，各按钮的作用如下：

1. 变量按钮：可新建pos和dyn对应的数据类型变量。（暂不考虑）
2. 示教按钮：记录当前机器人点位点位运动的位置信息。（暂不考虑）
3. 添加按钮：添加指定的pos或dyn参数变量到PTP指令中。
4. 设置按钮：设置pos或dyn的内部参数数值。
5. 返回按钮：点击该按钮，返回指令选择界面。
6. 确定按钮：选中PTP指令，点击该按钮，将指令插入程序编辑界面。

PTP指令表示机器人TCP末端进行点到点运动（point to point），执行这条指令时所有的轴会同时插补运动到目标点。其中共有两个参数可配置，分别是pos和dyn，在整个PTP指令中，dyn参数是可选的，根据实际工艺进行选择。下面将对pos参数和dyn参数进行简要介绍：

pos参数辨识TCP点的位置，即执行PTP这条指令之后（以图2.39所示的AXISPOS1为例），TCP点会走到对应的AXISPOS1点，其内部参数如图2.40所示：

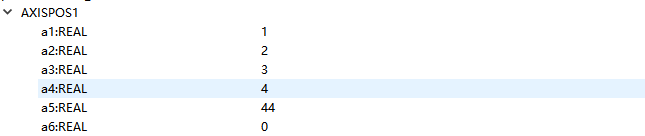


图2.40 pos内部参数

如图2.40所示，pos内部参数a1、a2、a3、a4、a5、a6分别表示各个轴的位置，表示机器人6个轴的位置，如果只有三个轴的话，只显示到a3，其他的以此类推。后面的值辨识轴相对于零点的位置，如果是旋转轴的话，单位是读，如果是直线轴的话，单位是mm。

注意，pos参数想包含两种数据类型的变量，分别是AXISPOS和CARTPOS，即关节坐标系和笛卡尔坐标系下表示的点的位置信息变量。

dyn参数表示执行这条指令过程中机器人的动态参数，其包含的数据类型为DYNAMIC，其中包括12个参数，具体如下图2.41所示：

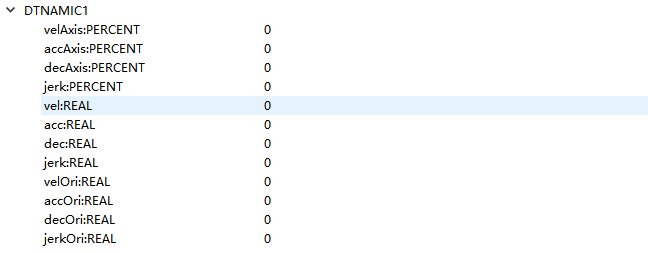


图2.41 dyn内部参数

如图2.41所示，其中velAxis,accAxis,decAxis,jerkAxis分别表示在自动运行模式时的轴速度，轴加速度和减速度，轴的加加速度，其值是一个相对于最大值的百分比，值的范围是0到100。系统的默认值为0，在PLC配置中可以设置，但有时候默认值和PLC配置值会有略微偏差。

另外4个参数vel、acc、dec、jerk分别表示在自动运行模式下运动时TCP点的速度，加速度，减速度和加加速度，在PLC中可以设置。

还有最后4个参数velOri、accOrc、decOri、jerkOri分别表示在自动运行模式下运行时TCP姿态变化的速度，加速度，减速度和加加速度，在PLC中可以设置。

了解完PTP指令下的参数配置后，然后对PTP指令插入操作进行阐述。

以PTP指令只配置pos参数为例，操作步骤为：

首先选中pos参数下想配置的具体pos参数，如图2.39所示，可选择的pos变量分别为AXISPOS1、AXISPOS2、CARTPOS1。这里选中AXISPOS,此时，用户获得添加按钮的点击权限，点击添加按钮，可将该变量配置到PTP指令中，如下图2.42所示：

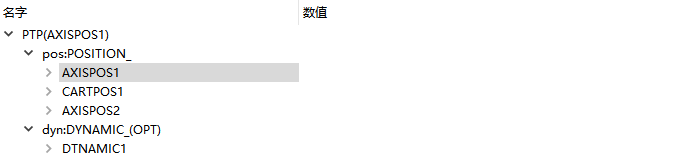


图2.42 配置AXISPOS1参数配置到PTP指令

若想配置dyn参数到PTP指令中，也可通过类似的操作实现，这里不再过多阐述。

若想将配置完成的PTP指令插入到程序编程界面中，这里以图2.42所示的PTP指令为例进行阐述，具体操作为：

首先选中PTP（AXISPOS1）所在行，此时PTP指令所在行呈现选中状态，用户获得确定按钮的点击权限，点击确定按钮，将跳转到程序编程界面，并且所选的PTP（AXISPOS1）指令将成功插入到程序编程界面中，如下图2.43所示：

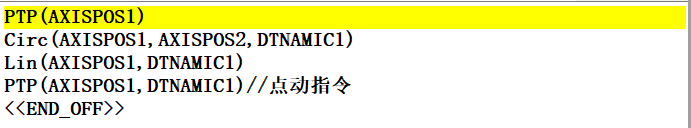


图2.43 插入PTP(AXISPOS1)指令

当完成程序的编程、示教操作时，可以运行程序，运行程序的两种方式如下所示：

1. 手动模式：
2. 点击单步/连续按钮，选择运行模式为单步或者连续。
3. 点击伺服上电按钮，确保机器人处于伺服上电状态。
4. 点击开始按钮，程序单步或连续运行，机器人随着预设的位置信息进行姿态的变化。
5. 自动模式：
6. 点击手动/自动按钮，将示教器拨动至自动模式。
7. 点击单步/连续按钮，选择运行模式为连续模式。
8. 点击伺服上电按钮，确保机器人处于伺服上电状态。
9. 点击开始按钮，程序单步或连续运行，机器人随着预设的位置信息进行姿态的变化。

另外，点击Set\_PC按钮，可以设置程序启动行的光标。

**2.6 位置管理模块**

进入位置管理模块界面，这时候可以看到关节坐标系下机器人的姿态，此时，点击切换坐标按钮，也可以将机器人切换到直角坐标系。

如下图2.44所示为位置管理模块在关节坐标系下的主界面：



图2.44 位置管理模块在关节坐标系的界面

如上图2.44所示，为示教器的位置管理模块的界面，当前界面显示的为关节坐标系下的各个轴的名称、各个轴的转动数值以及它们的单位。同时也可以点击世界按钮，显示在世界坐标系下的位置信息，如下图2.45所示：



图2.45 位置管理模块在世界坐标系的界面

直角坐标系下注意一点，当机器人的5轴在0度时，机器人处于起点位置，此时机器人无法做直线运动，需要将机器人的坐标切换到非零点位置。同理，也可以点击伺服按钮，观察各个电机轴的角度信息。

除了私服、关节和世界按钮之外，还有电动按钮，电动按钮可以在实际操作或编程时改变机器人的坐标系，其中有轴坐标系，世界坐标系，工具坐标系。

另外还有电动速度按钮，可以调节当前机器人运动速度，可以调节的档位有：100%、50%、25%、10%、1.0lnc、0.1lnc。

因此，通过该模块，在示教器点击相应控制按钮控制机器人做点位运动的时候，机器人末端的TCP点的位置信息可以实时的在该模块下观测到，便于进行测试。

**2.7 信息报告管理模块**

**3 运动指令组**

**3.1 PTP**

PTP指令各参数的意义以及该指令的插入已在2.5.2中进行了详细的阐述，这里不再进行解释，详见2.5.2

**3.2 Lin**

Lin指令为一种线性的运动命令，通过该指令可以使机器人TCP末端以恒定的速度直线移动到目标位置。假如直线运动与目标点的TCP姿态不同，那么TCP从起点位置到运动目标位置的同时，TCP姿态会通过姿态连续插补的方式从起点姿态过渡到目标点的姿态。如下图3.1所示为Lin指令的参数配置界面：

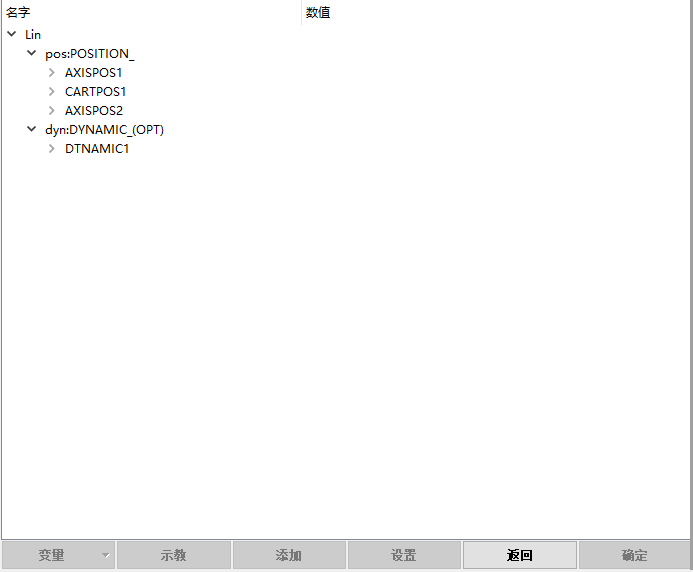


图3.1 Lin指令参数配置界面

如图3.1所示对Lin指令配置的参数主要有pos和dyn参数，下面对两个参数的定义进行阐述：

pos参数是TCP点在空间中的位置，即执行LIN这条指令之后，以图3.1中CARTPOS1为例，TCP点会走到CARTPOS1点，其内部参数如下图3.2所示：

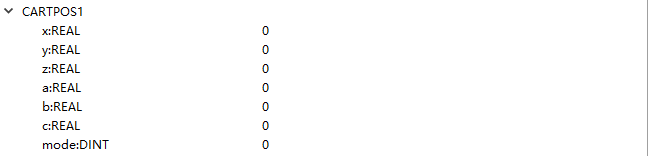


图3.2 pos内部参数

如图3.2所示，x,y,z分别表示TCP点在参考坐标系三个轴上的位置，a,b,c表示TCP点姿态，mode表示机器人运行过程中的插补模式，在指令执行过程中，轨迹姿态插补过程中插补模式不能更改。

dyn参数与PTP指令中的dyn参数一致，具体介绍详见2.5.2

Lin指令与PTP指令类似，也有不同的参数配置形式，如下图3.3所示为Lin指令中只配置pos，没有配置dyn参数：

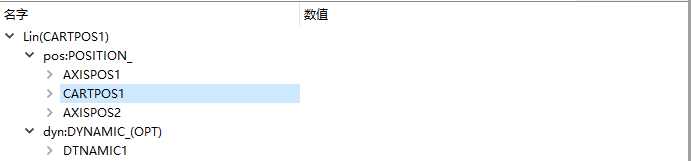


图3.3 Lin指令配置pos参数

如下图3.4所示为Lin指令配置了pos参数和dyn参数：

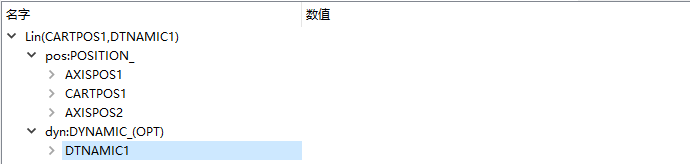


图3.4 Lin指令配置pos和dyn参数

图3.3和图3.4的指令插入情况如下图3.5所示：



图3.5 Lin指令插入情况

**3.3 Circ**

圆弧指令使机器人TCO末端从起点，经过辅助点到目标点做圆弧运动。

该指令必须遵循以下规定：

1. 机器人从TCO末端做整周运动，必须执行两个圆弧运动指令。
2. 圆弧指令中，起始位置、辅助位置以及目标位置必须能够明显的被区分开。

注意：起始位置是上一个运动指令的目标位置或者当前机器人TCP位置。

Circ的指令参数配置界面如下图3.6所示：



图3.6 Circ指令参数配置界面

其中pos参数配置包括圆弧的目标点位置和圆弧的辅助点位置，而dyn参数的设置同PTP和Lin。具体详见2.5.2。

Circ指令也有不同的配置形式，如下图3.7所示为Circ指令只配置pos参数，没有配置dyn参数：

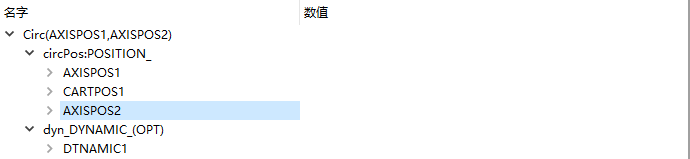


图3.7 Circ指令配置pos参数

如下图3.8所示为Circ指令配置了pos参数和dyn参数：

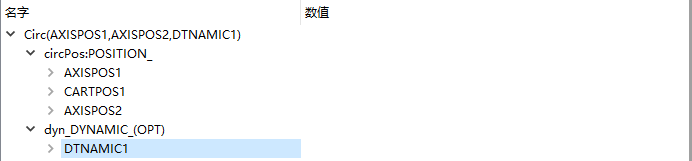


图3.8 Circ指令配置pos和dyn参数

图3.7和图3.8所示的指令插入形式如下图3.9所示：



图3.9 Circ指令插入形式

**3.4 PTPRel**

该指令为PTP插入相对偏移指令，该指令的相对位移可以是位移也可以是角度。该指令总是以当前机器人位置或者上一步运动指令的目标位置为起始位置，然后机器人相对移动位移或者角度偏移。运动还可以设置dyn参数。PTPRel指令的参数配置界面如下图3.10所示：

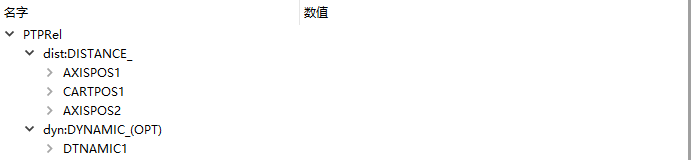


图3.10 PTPRel指令参数配置界面

Dist:DISTANCE\_参数中包含两种数据类型的变量，分别为为AXISPOS和CARTPOS。如果设置为AXISPOS,则是对轴的角度相对位移，设置为CARTPOS则是对位移的相对位移，但都是相对前一运动而言。

dyn参数的配置和PTP、Lin等指令的dyn参数配置一致，这里不再过多阐述，详见2.5.2。

例如生成指令PTP（AXISPOS1）和PTPRel（AXISPOS2）,机器人首先执行PTP（AXISPOS1）指令，然后执行PTPRel（AXISPOS2）指令。当执行PTPRel时则相对于PTP指令的目标点做偏移运动，假如在PTPRel中设置了a1:real为30，那么PTPRel运行时相对于AXISPOS1点向A1的正方向移动了30度，其他轴无转动。

参数dist中的a1,a2,a3,a4,a5,a6表示的是每个轴相对的偏移量，如果是旋转轴的话，单位是度，如果是直线轴的话，单位是mm。该例使用的是六轴关节机器人，所以有六个参数，单位都是度，如果是三轴的的直线坐标系的话，则只有三个参数，单位是mm。其他的以此类推。

PTPRel指令也具有不同的参数配置形式，如下图3.11所示为PTPRel配置dist参数，不配置dyn参数：

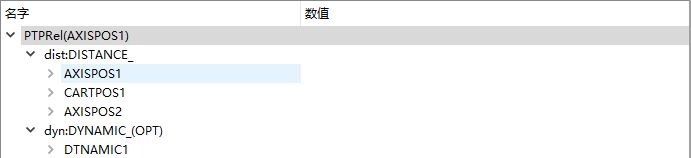


图3.11 PTPRel配置dist参数

如下图3.12所示为PTPRel指令配置dist和dyn参数：

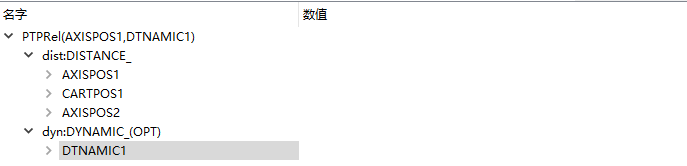


图3.12 PTPRel配置dist和dyn参数

图3.11和3.12指令插入的情况如下图3.13所示：



图3.13 PTPRel指令插入

**3.5 LinRel**

该指令为线性插补相对运动指令，该指令的相对偏移是位移，还有机器人的姿态。该指令总是以当前机器人位置或者上一步运动指令的目标位置为起点位置，然后机器人相对移动位移偏移或者姿态偏移。运动还可以设置dyn参数，与PTPRel类似，LinRel的参数配置界面如下图3.14所示：



图3.14 Lin指令参数配置界面

参数x,y,z表示的是空间坐标系下x,y,z三个方向的相对位移量，单位是mm：a,b,c辨识的是机器人的姿态相对偏移量，单位是度。

dyn参数的配置和PTPRel的dyn参数配置一致，这里不再过多阐述。

同时，LinRel指令的插入形式和PTPRel指令的插入形式也一致，详见3.4介绍。

**3.6 StopRobot**

该指令用来停止机器人运动并且丢弃已经计算好的插补路径。

StopRobot停止的是机器人运动，而不是程序，因此机器人执行该指令后将以机器人停止的位置做运动起点位置，然后计算插补路径以及执行后后续的运动指令。

如下图3.15所示为StopRobot的参数配置对话框：

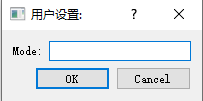


图3.15 StopRobot指令配置对话框

由图3.15所示，StopRobot的参数配置非常简单，只有一个参数Mode需要配置，配置完成，点击OK按钮，即可将StopRobot指令插入到程序当中。如下图3.16所示：



图3.16 StopRobot指令插入

**3.7 回零指令组**

**3.7.1 RefRobotAxis**

该指令用于标定回零位置，可以单步运行，执行后机器人根据配置中的回零方式运动，当机器人到达零后，保存当前机器人轴位置作为该轴的零位。

轴在回零后要做一个设定的目标值，目前改制没有的话，则值回到零点。

如下图3.17所示为RefRobotAxis参数配置界面：

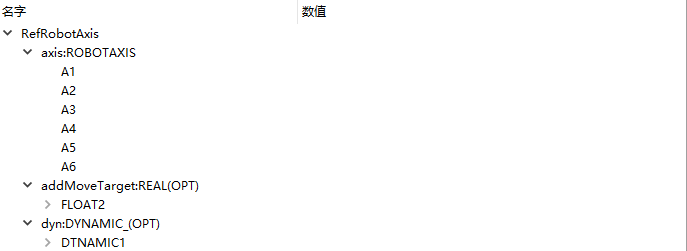


图3.17 RefRobotAxis参数配置界面

由图3.17所示，axis:ROBOTAXIS参数为要回零的轴，addMoveTarget(OPT)为回零后轴要运动的位置，dyn参数的配置和LinRel的dyn参数配置一致。

RefRobotAxis指令有不同的参数配置形式，如下图3.18所示为RefRobotAxis参数配置axis参数，不配置addMoveTarget参数和dyn参数：

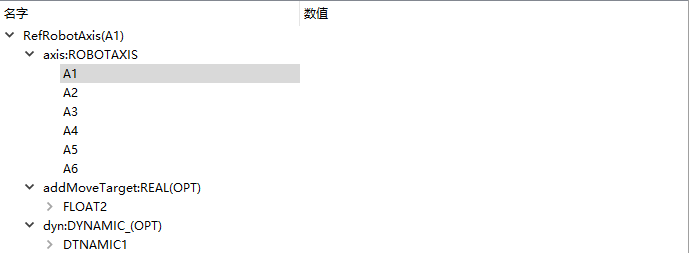


图3.18 RefRobotAxis配置axis参数

如下图3.19所示为RefRobotAxis配置axis和addMoveTarget参数，不配置dyn参数：

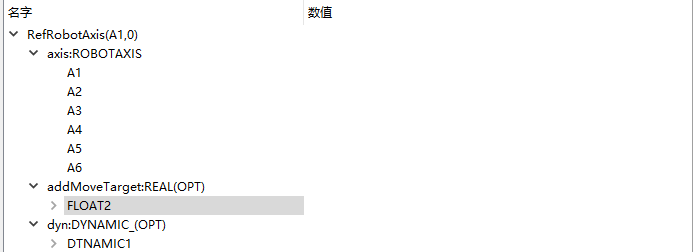


图3.19 RefRobotAxis配置axis和addMoveTarget参数

如下图3.20所示为RefRobotAxis配置axis参数、addMoveTarget参数和dyn参数：

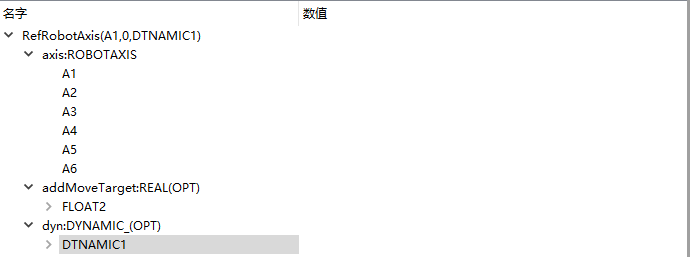


图3.20 RefRobotAxis配置axis、addMoveTarget和dyn参数

**3.7.2 RefRobotFinished**

该指令允许多轴同时回零。这个指令等待机器人回零动作结束，为了能够知道是否完成回零，需要配合WaitRefFinished。

**3.7.2 WaitRefFinished**

**3.8 WaitIsFinished**

**3.9 WaitJustInTime**

**3.10 MoveRobotAxis**

该指令的作用是对机器人的单一轴进行运动。如下图3.21所示为MoveRobotAxis指令的参数配置界面：

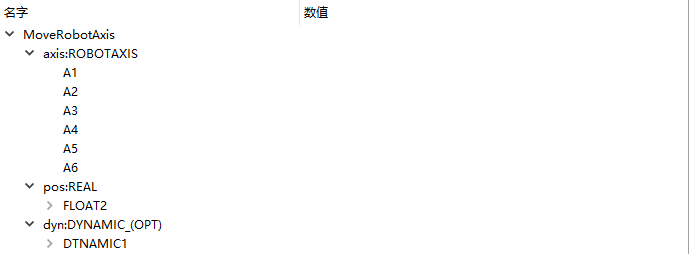


图3.21 MoveRobotAxis指令参数配置界面

由图3.21所示，axis:ROBOTAXIS为目标轴，pos:REAL为目标轴运动的位置值。dyn参数的配置和PTP等指令的dyn参数配置一致。如下图3.22所示为MoveRobotAxis指令配置axis、pos和dyn参数：

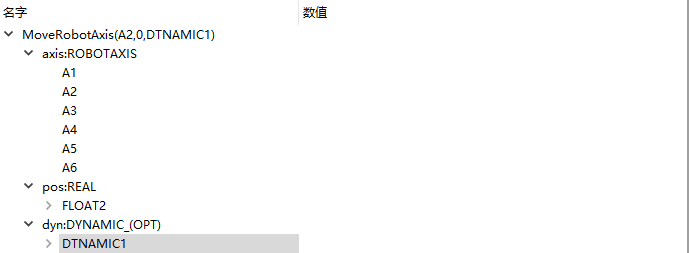


图3.22 MoveRobotAxis配置axis、pos和dyn参数

**3.11 PTPSearch**

**3.12 Linsearch**

**4 设置指令组**

**4.1 Dyn**

该指令配置机器人的动态参数。在PTP运动中配置轴速度的百分比，笛卡尔动态参数使用绝对值参数，执行该指令后，在自动模式下机器人以设定的动态参数运动指导参数被修改。

如下图所示为Dyn的参数配置界面:

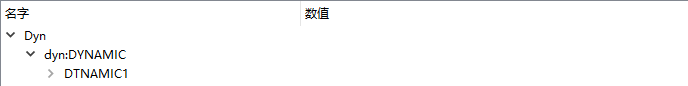


图4.1 Dyn参数配置界面



图4.2 指令对比说明

由图4.2所示，第一、第二条指令的效果和第三条指令的运行效果是相同的。

**4.2 Dynovr**

配置机器人运动的动态倍率参数。执行该指令后可以按照配置的百分比降低机器人动态参数。示教器上的V+、V-按钮时设置倍率参数。

动态倍率变量参数命令会对移动速率参数整体产生影响。

此命令不仅重叠命令一样可以变更移动速度，同时该命令设置的比率还会对加速度、减速度进行限制。

如下图所示为Dynovr的参数配置界面：



图4.3 Dynovr参数配置界面

如下图4.4所示为Dynovr配置ovr参数：

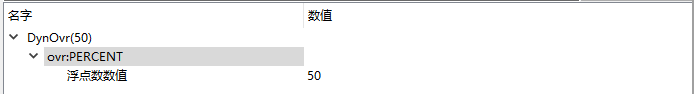


图4.4Dynovr配置ovr参数

如上图所示，机器人在运行的时候，机器人按照倍率参数（此时状态栏显示为50%）乘以动态倍率参数50%的速度来走轨迹（25%）。

**4.3 Refsys**

设置参考系统指令。通过该指令可以为后续运行的位置设定一个新的参考坐标系。如果程序中没有设定参考坐标系，系统默认参考坐标系为世界坐标系。

参考坐标系有三种类型，分别是CARTREFSYS、CARTREFSYSEXT、CARTREFSYSVAR（目前只考虑CARTREFSYS）。如下图4.5所示为Refsys的参数配置界面：



图4.5 Refsys参数配置界面

其中CARTREFSYS类型的主要参数有参考坐标系的基坐标系baseRefSys,即所要建立的参考坐标系时参考哪个坐标系建立的，x,y,z分别是相对于基坐标系的位置偏移量，a,b,c是相对于基坐标系的姿态。

**4.4 Tool**

工具坐标指令是机器人设置一个新工具坐标。通过该指令可以修改机器人末端工作点。如下图4.6所示为Tool指令的参数配置界面：

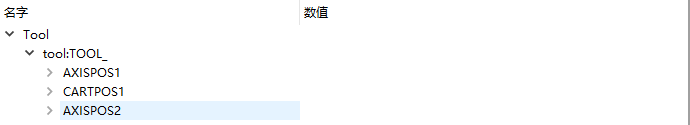


图4.6 Tool参数配置界面

其中Tool指令只有一个参数tool:TOOL需要配置，其包含的数据类型为AXISPOS和CARTPOS。

**4.5 Ovl**

**4.6 Ramp**

**4.7 OriMode**

**5 系统功能指令组**

**5.1 WaitTime**

该指令用于设置机器人等待时间，时间单位为ms，假如设置等待1s，生成命令如下图5.1所示：

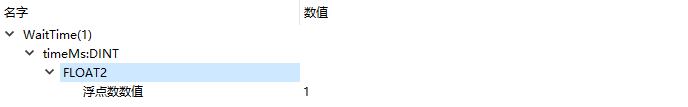


图5.1 WaitTime参数设置

**5.2 Stop**

该命令用于停止所有激活程序的执行。如果指令不带参数，等同于按下了示教器终端的停止按钮。如下图5.2所示为Stop命令的参数设置界面：

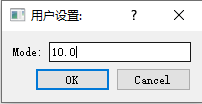


图5.2 Stop参数设置对话框

如图5.2所示，Stop命令只需要设置一个参数Mode，当点击OK按钮之后，指令将插入到程序当中。

**5.3 Info**

发出一个消息通知。信息显示在信息协议和报告协议的信息栏中。

此外，有可能显示两个附加参数的任何类型信息，第一个参数使用“%1”作为占位符，第二个参数使用“%2”作为占位符。test:STRING参数可调用变量管理模块中新建的字符类型变量，如下图所示为Info指令配置test:STRING参数：



图5.3 Info指令配置test:STRING参数

**5.4 ...:...(赋值)**

**5.5 //...(注释)**

**5.6 Warning**

**5.7 Error**

**5.8 Random**

**5.9 时间计量命令**

**5.10 数学运算指令**

**5.11 位运算指令及转换指令**

**6 流程控制指令**

**6.1 CALL**

该指令为调用指令，能够调用其他程序作为子程序，且调用的程序必须在编写程序的项目中。

如下图6.1所示为CALL指令的参数设置对话框：

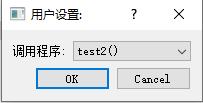


图6.1 CALL指令参数设置对话框

如下图6.2所示为CALL指令插入到程序之后的显示情况：



图6.2 CALL指令插入程序

**6.2 Wait**

该指令为等待指令。当WAIT表达式为TRUE，下一步指令就会执行，否则的话，程序等待直到表达式为TRUE为止。

如下图6.3所示为Wait指令的参数设置对话框：

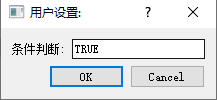


图6.3 Wait指令设置对话框

如下图6.4所示为Wait指令插入导程序之后的显示情况：



图6.4 Wait指令插入程序

**6.3 IF...THEN..END\_ID,ELSEIF..THEN,ELSE...**

IF指令用于条件跳转控制。类似于C++中的IF语句。IF条件判断表达式必须是BOOL类型。每一个IF指令必须以关键字END\_IF作为条件控制结束。

如下图6.5所示为IF指令的参数配置对话框：

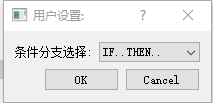


图6.5 IF指令参数配置对话框

如下图6.6所示为条件分支选择的具体选项：



图6.6 条件分支选择

如图6.6所示。条件分支共有三种选择，若选择IF..THEN..，将弹出如下图6.7所示的条件参数设置对话框：

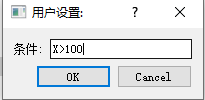


图6.7 IF...THEN..参数设置对话框

如图6.7所示输入条件语句，点击OK按钮，在程序中将插入如下图6.8所示的指令：



图6.8 插入IF..THEN..指令

若在图6.6所示分支中选择ELSE..THEN..语句，将弹出如下图6.9所示的参数设置对话框：

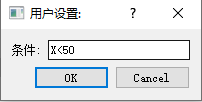


图6.9 ELSE...THEN..条件参数

如图6.9所示输入条件语句，点击OK按钮，在程序中将插入如下图6.10所示的指令：



图6.10 ELSE..THEN..指令插入

若在图6.6所示分支中选择ELSE指令，将直接插入如下图6.11所示的指令：



图6.11 ELSE指令插入

**6.4 WHILE...DO...END\_WHILE**

WHILE指令在满足条件的时候循环执行子语句。循环控制表达式必须是BOOL类型。该指令必须以END\_WHILE作为循环控制结束。

如下图6.12所示为WHILE指令的参数设置对话框：

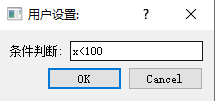


图6.12 WHILE参数设置对话框

如图6.12所示输入条件判断语句，点击OK按钮，将插入指令到程序中，如下图6.13所示：



图6.13 WHILE指令插入程序

**6.5 LOOP...DO..END\_LOOP**

该指令为循环次数控制指令。如所下图所示为LOOP指令的参数设置对话框：

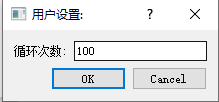


图6.14 LOOP指令参数设置对话框

如图6.14所示输入循环次数为100，点击OK按钮，将插入LOOP指令到程序中，如下图6.15所示：



图6.15 LOOP指令插入程序

**6.6 RUN,KILL**

RUN指令调用一个用户程序，该程序与主程序平行运行，RUN调用的程序必须用KILL指令终止。RUN调用的程序必须是该项目中的程序。

如下图6.16所示为RUN指令的参数设置对话框：

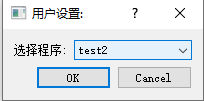


图6.16 RUN指令参数设置对话框

点击图6.16所示的OK按钮，将指令插入程序中，如图6.17所示：



图6.17 RUN指令插入程序

**6.7 RETURN**

该指令用于终止正在运行的程序。

如下图所示为将RETURN指令插入程序中的显示情况：



图6.18 RETURN指令插入程序

**6.8 GOTO...IF...GOTO..LABEL**

GOTO指令用于跳转到程序不同部分，跳转目标通过LABEL指令定义。不允许从外部跳转进入内部程序。内部程序可能是WHILE循环块或者是IF程序块。

IF-GOTO指令相当于一个缩减的IF程序块。IF条件判断表达式必须是BOOL类型。假如条件满足，程序执行GOTO跳转命令，其跳转目标必须由LABEL指令定义。

LABEL指令用于定义GOTO跳转目标。

如下图6.19所示为GOTO指令的参数设置对话框：



图6.19 GOTO指令参数设置对话框

图6.19所示的GOTO分支具有如图6.20所示的分支选择：



图6.20 GOTO分支选择

若选择图6.20中的GOTO...分支，点击OK按钮，将指令插入程序，如下图6.21所示：



图6.21 GOTO...指令插入程序

若选择图6.20中的IF...GOTO...分支，点击OK按钮，将出现如下图6.22所示的参数设置对话框：

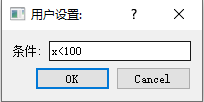


图6.22 IF..GOTO..参数设置对话框

如图6.22所示输入条件判断语句，点击OK按钮，将指令插入程序中，如图6.23所示：



图6.23 IF..GOTO...指令插入程序

若选择图6.20所示的LABEL..分支，将出现如下图6.24所示的参数设置对话框：

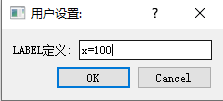


图6.24 LABEL..参数设置对话框

输入图6.24所示的LABEL定义参数，点击OK按钮，将指令插入程序中，如下图6.25所示：



图6.25 LABEL...指令插入程序

**7 输入输出指令组**

**8 触发指令组**