卡诺普
CROBOTP

控制器占据半壁江山的机器人品牌

让客户用好机器人

卡诺普编程指令说明书

PROGRAMMING INSTRUCTION MANUAL

CRP-JS-YF2-NT-00017-A1

请确保相关说明书到达本产品的最终使用者手中。

十分感谢您选用本公司产品！

本产品相关手册请妥善保管，以备需要时查阅！

如设备需要转手，请将相关资料一并转交对方！

机器人相关手册未做说明的按键、功能、选项视为不具备，请勿使用！

2021-8-10	初稿
2021-11-25	修订内容

目 录

一、 编程方法

1. 打开电源开关，等待开机完成。然后将模式开关拨到：示教（TEACH）模式。

2. 切换到程序列表界面，新建或打开已有程序。

图 1.1

3. 点击主菜单中的【编程指令】键，或者功能键区域的对应键【运动】【逻辑】调用常用指令。使用功能区按键时，重复点击按键，将调用该目录下不同指令。点击【上一条指令】调用上次使用的指令，界面如下图所示。

图 1.2

4. 输入正确指令参数后，按住安全开关，再点击【指令正确】输入指令；点击【指令退出】则取消指令输入。

二、指令行结构及内容说明

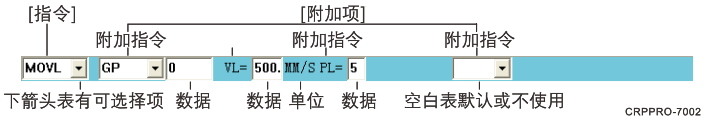


图2.1

• 白色框为需要选择或输入内容。不使用则不需要输入或选择任何内容。

• 带有黑色下箭头标志，该位置内容，只能通过上下键选择，不可直接输入。

• 程序指令行可选择或输入内容分为：指令、数据（变量）、其他指令相关的要素（例如运动指令必须要指定速度，判断指令必须指定判断符），还有状态、编号等

• 指令：需要调用的指令，如MOVJ、JUMP、CALL、INC 等。

• 数据（变量）：例如变量GP及GP的变量号，延时的时间，运动速度，常数D的值等等，该数据根据指令的不同可以为小数、整数、负数以及字符等。

• 判断符：判断前后条件关系，如==、＞、＜、>=、<=、=。

• 状态：（ON=1，OFF=0），该状态可以转化为0、1数据使用。

• 编号：该编号只能为0及以上整数，根据指令的不同。编号范围也不相同。

• 各个指令所带的附加项不尽相同，输入指令时请加以注意。

三、指令结构举例说明

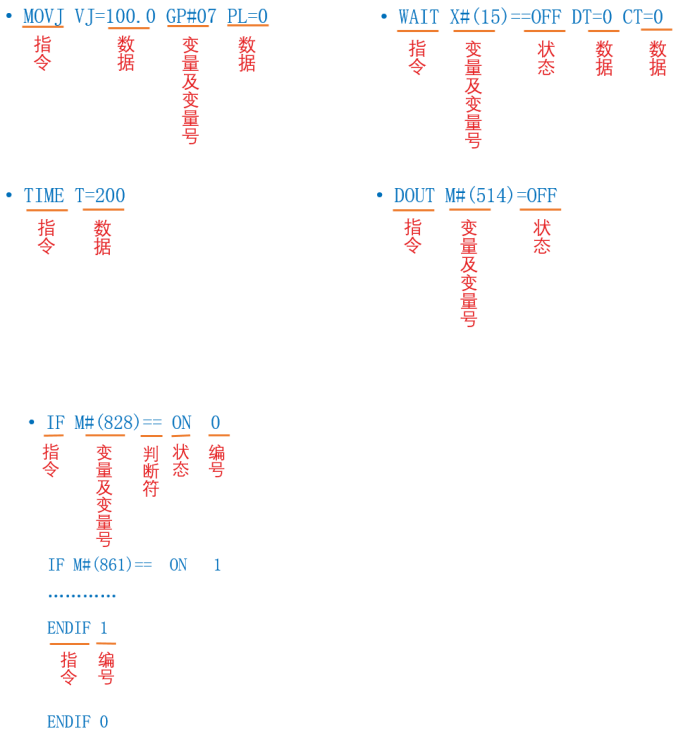
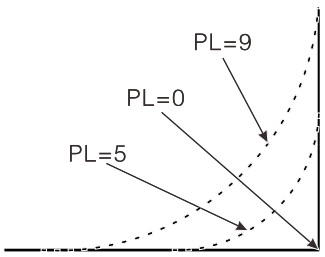


图 3.1

四、运动指令要素

运动指令的要素有运动方式（指令），速度，点位（位置变量），平滑度，以及其他特殊功能的附加项。

<table>
<tr><th>运动方式（指令）</th><th>附加项格式、定义</th><th>说明</th></tr>
<tr><td>关节(MOVJ)</td><td>VJ=<百分比>：关节运行速度</td><td>单位1%，最大值为100%。实际轴速度=参数中轴运动最高速度× VJ × 自动运行倍率。</td></tr>
<tr><td>直线(MOVL)</td><td rowspan="3">VL=<直线运行速度>：直线运行速度</td><td rowspan="3">单位MM/S，最大值为参数设定直线运动最高速度。实际运行速度=直线运动最高速度×VL× 自动运行倍率。</td></tr>
<tr><td>圆弧(MOVC)</td></tr>
<tr><td>整圆(MOVCA)</td></tr>
<tr><td>圆弧(MOVC)</td><td>POINT=<圆弧上的点></td><td>范围1-3；圆弧上的点，一段圆弧必须由3个点组成；</td></tr>
<tr><td>关节(MOVJ)</td><td rowspan="4">PL=<平滑度>：平滑度</td><td rowspan="4">范围0-9。简单的说就是过渡的弧度，确定您是以直角方式过渡还是以圆弧方式过渡。假如两条直线要连接起来，怎么连接，就需要您对此变量进行设置。PL数值选择参考下图。
 
</td></tr>
<tr><td>直线(MOVL)</td></tr>
<tr><td>圆弧(MOVC)</td></tr>
<tr><td>整圆(MOVCA)</td></tr>
</table>

五、指令中常用变量说明

变量按照数据形式不同，可以分为：位置变量（GP，LP，OP，VP/NP）、状态变量（X，Y，M）、整型变量（GI，LI）、浮点型变量（GD，LD）、输出模拟量变量（AO）、字符串变量（GS）

变量按照作用范围不同，可以分为：局部变量（LP，LI，LD）和全局变量（GP，GI，GD，GS）下面以数据形式分类分别介绍。

5.1 位置变量介绍

GP<变量号><数据号>：全局P变量	1．全局P变量，该变量值记录机器人各关节姿态，坐标等相关位置数据，是多个数据组合。变量号范围：0-999，数据号范围为：0-11。
	2．GP数组中，系统开放了部分数据信息，用数据号标识。每一个数据号代表的含义不同；数据号范围为：0-11。
	3．数据号表示可以单独调用GP变量的组合数据中某一数据，数据号的定义为： 0：对整个数据组合进行操作；1：对X轴数据进行操作； 2：对Y轴数据进行操作；3：对Z轴数据进行操作； 4：对J1轴数据进行操作；5：对J2轴数据进行操作； 6：对J3轴数据进行操作；7：对J4轴数据进行操作； 8：对J5轴数据进行操作；9：对J6轴数据进行操作； 10：对J7轴数据进行操作；11：对J8轴数据进行操作；
	4．不同程序调用同一变量号时，该变量的数据为同一个数据，当被第二次赋值时，将覆盖第一次的数据。所以在使用已经使用过的变量号时需要慎重，以免发生误动作或危险。
	5．该变量可在“运行准备”-“变量”-“全局P变量”表中查看。
	6．GP变量需要在变量表中记录，只要程序中使用了GP变量都需要先记录GP点位，程序中可以做GP点运算修改，不能直接记录和修改。
	7．GP0-GP39为预留的通用全局位置变量，其余GP均被系统应用为专用全局位置变量；
	8．专用GP的使用说明请详见《变量使用定义表A0》。

LP<变量号><数据号>：局部P变量	1.局部P变量，该变量值记录机器人各关节姿态，坐标等相关位置数据，是多个数据组合。变量号范围：0-999。
	2.数据号表示可以单独调用GP变量的组合数据中某一数据，数据号的定义为： 0：对整个数据组合进行操作； 1：对X轴数据进行操作； 2：对Y轴数据进行操作； 3：对Z轴数据进行操作： 4：对J1轴数据进行操作； 5：对J2轴数据进行操作； 6：对J3轴数据进行操作； 7：对J4轴数据进行操作； 8：对J5轴数据进行操作； 9：对J6轴数据进行操作； 10：对J7轴数据进行操作； 11：对J8轴数据进行操作；
	3. 不同的程序调用同一变量号时，各自对应的数据不一样，互不干涉，各自独立。如一个程序调用变量LP0。第二个程序也调用了LP0，两个LP0互不干扰，各自独立。
	4. 该变量只有在调用程序打开时，才会在变量表中出现。可在“运行准备”-“变量”-“局部P变量”表中查看，记录，清除，手动修改X,Y,Z的值；手动试运行此点位。
OP<变量号>：偏移P变量	1. 寻位偏移变量OP：该变量用于程序点位的偏移，其数值可以通过手动输入或寻位计算的方式进行设置，是多个数据组合。变量号范围：0-1000。
	2. 数据号表示可以单独调用OP变量的组合数据中某一数据，数据号的定义为： 0：对整个数据组合进行操作； 1：对X轴数据进行操作； 2：对Y轴数据进行操作； 3：对Z轴数据进行操作； 4：对A姿态数据进行操作； 5：对B姿态数据进行操作； 6：对B姿态数据进行操作； 其中，位置偏移量X,Y,Z可以通过手动输入的方式设置，旋转偏移量A,B,C无法进行手动设置。
	3. OP变量可以应用于直线或圆弧运动路径的偏移；
	例：单个点的偏移： MOVL VL=100mm/s PL=0 OFFSET OP#(1) 例：几条路径整体偏移 OFFSETSTART OP#(2) MOVL VL=100 mm/s PL=0 … OFFSETEND
	4.OP变量只能作用于直线/圆弧运动，不能作用于关节运动；禁止在偏移路径中存在关节运动行指令。

VP/NP<变量号>:寻位变量	1.VP和NP为寻位基准位置变量和寻位位置变量，其作用是存储寻位的记录点。VP与NP成对使用，通过“寻位工艺”中的旗标进行管理。当旗标打开时，寻到的位置点存在VP变量中，作为基准位置；当旗标关闭时，寻到的位置点存在NP变量中，即为寻位位置；VP与NP变量号的范围均为：0~999。
	2.VP/NP变量有如下的属性参数： X,Y,Z：寻位点的空间位置。 T：寻位时使用的工具坐标系 U：寻位时使用的用户坐标系
	3.在VP或NP列表中，选中一个VP/NP点位，可以进行该点位的试运行。但当前工具坐标系和NP的工具坐标系需要一致，否则可能会出现异常报警。
	4.寻位应用中，通常在寻位的方向路径程序行中设置寻位变量，程序行中只能选择NP，通过旗标控制写入NP还是VP。
	示例： MOVL VL=100 PL=0 SEACH NP[0] 当工艺中设置旗标=ON时，寻位点存储在VP[0]；旗标=OFF时，寻位点存储在NP[0]；

★**注意：**激光寻位的寻位点存储在VP中无效。

5.2 状态变量说明

X<变量号>：通用输入口	1.通用输入X接口，为状态变量（ON=1，OFF=0），范围：0-559。其中X0-X23在IO板上有物理接口；X96-X111需要连接外部IO扩展模块，才有物理接口；其他X触点被系统占用。
	2. 接口状态可在“监视”-“IO口”-“通用输入口监视”中查看。
	3．系统占用的X接口说明请详见《CRP-CD□0-CRX8 PLC说明书》。
Y<变量号>：通用输出口	1. 通用输出Y接口，该接口对应硬件物理接口，为状态变量（ON=1，OFF=0），范围0-559。其中Y0-Y23在IO板上有物理接口；Y64-Y79需要连接外部IO扩展模块，才有物理接口；其他Y线圈被系统占用。
	2. 接口状态可在“监视”-“IO口”-“通用输出口监视”中查看。
	3. 系统占用的Y接口说明请详见《CRP-CD□0-CRX8 PLC说明书》。
M<变量号>：内部辅助继电器	1. 内部辅助M继电器。该继电器为状态变量（ON=1，OFF=0），范围：0-4095。M500-M4095是留给客户使用，M0-M499已被系统占用；系统占用的M含义请参阅《变量使用定义 A0》。
	2. 该继电器状态可在“监视”-“PLC”-“M”下查看。

5.3 整型变量说明

GI<变量号>：全局整型寄存器	1.全局整型寄存器GI，存储整型数据。GI变量号范围：0-1023。其中GI0-GI49为预留的通用整型寄存器；其他GI，系统已占用为特殊寄存器。
	2. GI值可在“运行变量”-“变量”-“全局I变量”中查看、修改。
	3. 特殊寄存器GI的使用说明请详见《变量使用定义A0》。
	4. 不同程序调用同一变量号时，该变量的数据为同一个数据，当被第二次赋值时，将覆盖第一次的数据。所以在使用已经使用过的变量号时需要慎重，以免发生误动作或危险。
LI<变量号>：局部整型寄存器	1. 局部整型寄存器LI，存储整型数据。LI变量号范围：0-1000。
	2. 不同的程序调用同一变量号时，各自对应的数据不一样，互不干涉，各自独立。如一个程序调用变量LI0。第二个程序也调用了LI0，两个LI0互不干扰，各自独立。
	3. 该变量只有在调用程序打开时，才会在变量表中出现。可在“运行准备”-“变量”-“局部I变量”表中查看、修改。

5.4 其他变量说明

GD<变量号>：全局整型寄存器	1.全局浮点型寄存器GD，存储浮点型数据。GI变量号范围：0-99。 其中GD0-GD79为预留的通用整型寄存器；GD80-GD99,系统已占用为特殊寄存器。
	2. GD值可在“运行变量”-“变量”-“全局D变量”中查看、修改。
	3.不同程序调用同一变量号时，该变量的数据为同一个数据，当被第二次赋值时，将覆盖第一次的数据。所以在使用已经使用过的变量号时需要慎重，以免发生误动作或危险。
LD<变量号>：局部浮点型寄存器	1. 局部浮点型寄存器LD，存储浮点型数据。LD变量号范围：0-1000。
	2. 不同的程序调用同一变量号时，各自对应的数据不一样，互不干涉，各自独立。如一个程序调用变量LD0。第二个程序也调用了LD0，两个LD0互不干扰，各自独立。
	3. 该变量只有在调用程序打开时，才会在变量表中出现。可在“运行准备”-“变量”-“局部D变量”表中查看、修改。
AO<变量>：输出模拟量变量	1. 此变量为需要输出的模拟量电压值，值的范围为0-10v；
	2. 变量号范围为：AO1-AO23，其中AO1-AO4在AVO接口有物理输出口，其他没有物理接口，可通过通讯写入或读取变量值。
	3. 变量值为浮点型数据变量，该数值可定义到小数后三位（.000），带正负号。
	4. 全局变量，不同程序调用同一变量号时，该变量的数据为同一个数据。当被第二次赋值时，将覆盖第一次的数据。
	5. 该变量可在“监视”-“IO口”-“模拟量监视”中查看。

GS<变量号>：全局字符串变量	GS字符串变量，范围0-99。该变量主要用于读取条形码，外部IO口编码数据使用。配合READ DATA TO或READ IO BCD指令使用。如： READDATATO GS#（0） ：读取条码到GS0变量 PAUSE IF GS#(0)=123456 ：当GS0=123456时，暂停。 ……
D<变量>：自定义数据变量	自定义数据变量，其中变量为用户自行输入的具体数据。该数据可以带正负号，可输入小数。

六、编程指令列表

类别	英文指令	中文说明
1. 运动	1. MOVJ	关节移动
	2. MOVL	直线运动
	3. MOVC	圆弧运动
	4. MOVCA	整圆运动
2. 逻辑	1. DOUT	数字量输出
	2. AOUT	模拟量输出
	3. WAIT	条件等待
	4. TIME	延时指令
	5. PAUSE	暂停
	6. JUMP	条件跳转
	7. CALL	子程序调用
	8. ;	注释
	9. *	跳转标号
	10. RET	子程序返回
	11. TDOUT	输出带时间
3. 运算	1. ADD	加法运算
	2. SUB	减法运算
	3. MUL	乘法运算
	4. DIV	除法运算
	5. INC	加1运算
	6. DEC	减1运算
	7. SET	赋值
	8. MOD	求余(留用)
4. 码垛	1 PALLET	搬运
5.焊接	1. ARC START	起弧
	2. ARC END	起弧结束
	3. WEAVE	摆弧
	4. WEAVE END	摆弧结束
	5. SPOT	点焊
	6. STITCHSTART	鱼鳞纹焊接开始

5.焊接	7. STITCHEND		鱼鳞纹焊接结束
	8. OPENLASER		打开激光
	9. CLOSELASER		关闭激光
	10. SEARCHLASER		激光搜寻
	11. OPENLASERTRACK		开始激光跟踪
	12. CLOSELASERTRACK		结束激光跟踪
	13. ARCTRACKSTART		开始电弧跟踪
	14. ARCTRACKEND		结束电弧跟踪
	15. MP		多层多道焊接开始
	16. MPEND		多层多道焊接结束
	17. SETLASERPARA		改变激光参数组
	18. SETARCJOB		调用焊机工作参数组
	19. SETARCMODE		设置焊机工作模式
	20. ADLASERTRACKSTART		自适应激光跟踪开始
	21. ADLASERTRACKEND		自适应激光跟踪结束
	22. SETARCSPEED		设置焊接速度
6. 辅助指令	1. SPEED		速度改变
	2. IF	1. IF	条件判断
		2. ELSE IF	
		3. ELSE	
		4. END IF	
	3. FOR		备用
	4. WHILE	1. WHILE	循环
		2. END WHILE	
	5. SWITCH	1. SWITCH	条件选择
		2. CASE	
		4. DEFAULT	
		5. BREAK	
		6. END SWITCH	
	6. PULSE TARANSFORM		轴脉冲跳转
	7. PULSE TARANSFORM END		轴脉冲跳转结束
	10. MOTION FINISH		运动完成
	12. FORBID AXIS		轴禁止开始
	13. FORBID AXIS END		轴禁止解除
	14. COUNTCOORD		坐标计算
	15. JUDGE START		冲压状态开始
	16. JUDGE END		冲压状态结束
	17. OTALAR		超时报警
7. 视觉	1. RUN VISION		视觉运行
	2. GET VISON DATA		获得视觉数据

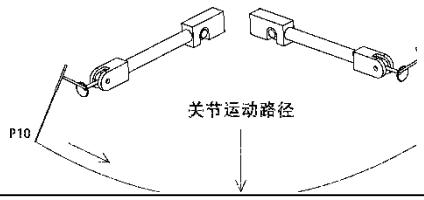
7. 视觉	3. CLEAR VISON DATA	清除视觉数据
	4. TRIGGER VISON	视觉指令触发
	5. CHANGE VISON	改变视觉模型
	6. WAITTRIGGER	等待触发结果
8. 跟踪	1. TARCK START	跟踪开始
	2. TARCK END	跟踪结束
	3. GET TRACK DATA	获得跟踪数据
	4. CLEAR TARCK DATA	清除跟踪数据
	5 RUN IO CUTIN	后台IO检测
9. 通讯	1. READ BCD TO	读取IO口BCD数据
	2. READ DATA TO	读取数据到
	3. SET BCD IO	把十进制输出
	4. SEND DATA TO	预留
10. 特殊指令	1. RESET	程序复位
	2. CHAN GEP	程序指针跳转
	3. SEARCH START	寻位开始
	4. SEARCH END	寻位结束
	5. OFFSET START	偏移开始
	6. OFFSET END	偏移结束
	7. COUNT OFFSET	计算偏移量
	8. START TIME	计时开始
	9. END TIME	计时结束
	10. MULTI TURN START	预留
	11. MULTI TURN END	预留
	12. COUNT LOADFAC	计算负载率指令
	13. COUNT STORQUE	计算静态转矩百分比指令
	14. COUNT DYNASTH	计算动态数据指令
	15. COUNT FRICTION	计算摩擦
	16. END COUNT FRICTION	结束计算摩擦
	17. ZOOM START	缩放开始
	18. ZOOM END	缩放结束
	19. ANGLEDIS PLAY	焊枪角度显示开始
	20. ANGLEDIS PLAY RST	焊枪角度显示复位
	21. TOWER COUNT START	开始塔角计算
	22. TOWER COUNT END	结束塔角计算
	23. TOWER CHANGE WELD	塔角焊缝计算
	24. TOWER COUNT POSE START	塔脚姿态计算开始
	25. TOWER COUNT POSE END	塔脚姿态计算完成
	26. COUNT STEE LANGLE	角钢轨迹计算
11. 折弯指令	1. BENDTRACK	折弯跟随

11. 折弯指令	2. BENDSYS	折弯同步
	3. BENDFLATBACK	折弯回平
12. 碰撞指令	1. SHCKSET	碰撞等级设置
	2. SHCKRST	碰撞等级解除
	3. SHCKACT	碰撞检测激活
	4. SHCKDEACT	碰撞检测禁用

七、指令详情

7.1 运动指令

7.1.1 关节运动（MOVJ）

关节运动（MOVJ）	指令功能	• 机器人以最快捷的方式运动至目的点，其运动状态不完全可控，但运动路径保持唯一。 •指令格式： MOVJ VJ=100% LP#10 PL=9 。 • MOVJ指令常用于机器人在空间大范围移动，如下图所示。 
	使用举例	MOVJ VJ=30% PL=3 MOVJ VJ=30 GP#0 PL=3 UNTIL M#(0)==ON

• 各关节运动的速度为：各轴设定速度×VJ×自动倍率。
• 各轴设定的速度在<参数>-<速度参数>中查看，该设定值是由各轴电机的最大转速限制。

10	1轴关节最大速度（°/s）	65
11	2轴关节最大速度（°/s）	65
12	3轴关节最大速度（°/s）	60
13	4轴关节最大速度（°/s）	100
14	5轴关节最大速度（°/s）	95
15	6轴关节最大速度（°/s）	165

图 7.1

7.1.2 直线运动（MOVL）

直线运动 （MOVL）	指令功能	•MOVL指令功能：机器人从起始点沿直线路径运动至目的点； •指令格式： MOVL VL=200mm/s LP#1 PL=0 •MOVL指令常用于机器人在空间小范围移动或必须采用直线轨迹时，例如焊接直线焊缝
	程序举例	MOVL VL=500MM/S PL=0 MOVL VL=500MM/S GP#3 PL=5 UNTIL X#(0)==ON MOVL VL=500MM/S PL=0 DOUT Y#(0)==ON START 10.0 END 10.0 MOVL VL=500MM/S PL=0 AOUT A1=2.0 A2=2.0

• 直线运动速度为：VL×自动倍率
• 直线最大速度在<参数>-<速度参数>中查看，该设定值是由各轴电机的最大转速限制。

6	直线最大移动速度（mm/s）	1500

图 7.2

7.1.3 圆弧运动（MOVC）

圆弧运动 （MOVC）	指令功能	用圆弧插补方式移动到示教位置。各关节按照VL×自动倍率，以圆弧插补运动。整圆需要由两段圆弧构成，使用至少四段圆弧指令，具体请参考综合实例。	
	附加项	[空白] GP<变量号> LP<变量号>	位置数据，屏幕该栏显示空白 GP变量，变量号：0-999 LP变量，变量号：0-999
		VL=<直线运行速度>	直线运行速度，单位MM/S，最大值为参数直线运动最高速度。
		PL=<平滑度>	PL平滑度：0-9
		<变量>POINT	变量范围：1-3。一段圆弧轨迹通必须是由三段圆弧指令实现的，三段圆弧指令分别定义了圆弧的起始点、中间点、结束点。1POINT表圆弧的起点，2POINT表圆弧的中间点，3POINT表圆弧的终点。

编程示例1：

最简单的平面圆弧
MOVC VL=100mm/s PL=0 POINT=1 //圆弧第一个点B
MOVC VL=100mm/s PL=0 POINT=2 //圆弧第二个点C
MOVC VL=100mm/s PL=0 POINT=3 //圆弧第三个点D

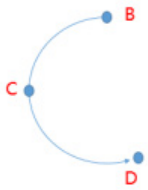


图 7.3

编程示例2：

需要变换姿态，用圆弧的方式画一个姿态圆

• 针对姿态圆运动，我们采用圆弧指令实现，在规划圆弧的至少要选取3段圆弧。（避免就近原则引起的反向运行）

• 编写姿态圆时，应当注意姿态变化的角度，尽量避免小距离，大姿态的运动轨迹。

• 在保证机器人末端轨迹的时候，距离小，姿态变化大，机器人在变化姿态的时候，速度会很快，造成机器人抖动或者超速报警，影响末端轨迹。

• 以下图为例：整圆运动，只是考虑C姿态的情况下，在寻找点位的时候，尽量保证坐标点与姿态都是均等分。从而来达到保证运动轨迹效果。

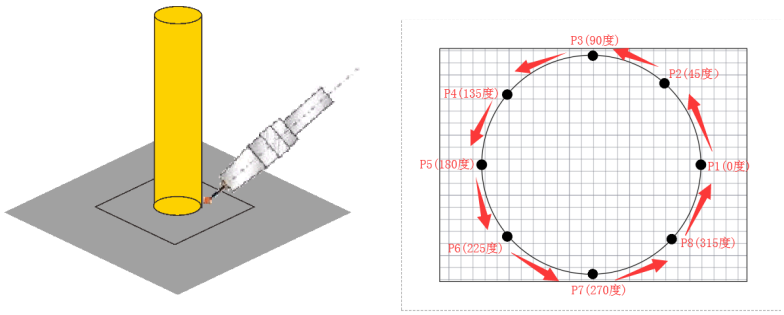


图 7.4

编程示例三：

直线过度圆弧

• 系统可以默认直线末端点为圆弧的第一个点，减少重复记录点以及避免重复点位引起的停顿问题。即（L）MOVL-2-3的方式。

MOVL VL=100mm/s PL=0 //直线末端点（圆弧起始点）
MOVC VL=100mm/s PL=0 POINT=2 //圆弧第二个点
MOVC VL=100mm/s PL=0 POINT=3 //圆弧第三个点

• 当需要在圆弧第一点启动焊接，ARCstart指令前的直线点和ARCstart指令后的点位必须同一个点，用复制程序行的方式保证同一个点(将第1行复制到第3行)；

1 MOVL VL=100mm/s PL=0 //直线末端点（圆弧起始点）
2 ARCSTART 0
3 MOVL VL=100mm/s PL=0 //直线末端点（圆弧起始点）
4 MOVC VL=100mm/s PL=0 POINT=2 //圆弧第二个点
5 MOVC VL=100mm/s PL=0 POINT=3 //圆弧第三个点
6 ARCEND 0

★**注意：**即使第1行与第3行在同一个点手动示教记录，也不可以。因为在同一个点手动示教记录，坐标值会有细微的差别。当同时使用摆弧功能时，会出现异响。

• 带姿态运动时，同样要避免小距离、大姿态情况。

• 在直线到圆弧第一个点时，要保证圆弧第一个点的姿态，在直线运动过程中带姿态变化，到圆弧第一个点时同时也变化好姿态，提高效率。如在直线过程中姿态变化大，可以在中间多加几个点位来保证轨迹。如下图，直

线段P1点至1P点，需要带姿态走，为保证末端轨迹，应当带姿态记录两个点，P1与P2点距离远即可变化稍大一点姿态来保证末端。

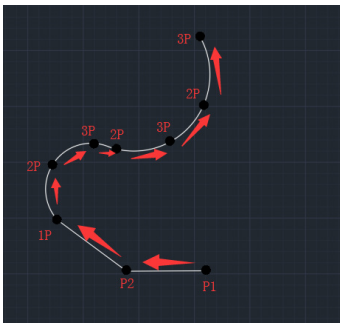


图 7.5

编程示例四：

连续圆弧段

• 遵循1（L）-2-3-2-3-2-3的方式找点，即上一段圆弧的终点作为下一段圆弧的起点，保证圆弧的连贯性。

• 小圆弧段：距离短，先变换好预想姿态，再移动位置记录。

• 大圆弧段：严格按照取该段弧的弧顶位置。

针对连续圆弧轨迹精度要求高的情况，可尽量把连续圆弧细分化，从而来保证运动轨迹及姿态变化。

MOVL VL=100mm/s PL=0 //直线末端点（圆弧1起始点）
MOVC VL=100mm/s PL=0 POINT=2 //圆弧1第二个点
MOVC VL=100mm/s PL=0 POINT=3 //圆弧1第三个点(圆弧2的起始点)
MOVC VL=100mm/s PL=0 POINT=2 //圆弧2第二个点
MOVC VL=100mm/s PL=0 POINT=3 //圆弧2第三个点

★注意：禁止编连续圆弧时采用1-2-3-1-2-3这种方式，这种方式在圆弧试运行时会出现无法预知的轨迹，可能会造成撞枪等故障。

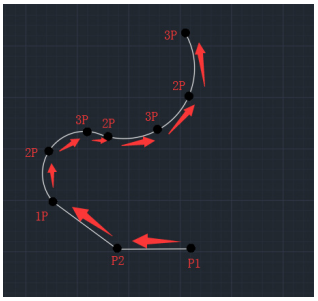


图7.6

7.1.4 整圆运动（MOVCA)

4. 整圆运动（MOVCA)	指令功能	用整圆弧插补方式实现一个整圆运动。各关节按照VL×自动倍率，以整圆插补运动。P1确定圆心，P2确定运动方向，P3同P1和P2确定一个圆弧面，即确定一个圆心一个半径，一个起始点和一个圆所在的平面。	
	附加项	[空白] GP<变量号> LP<变量号>	位置数据，屏幕该栏显示空白 GP变量，变量号：0-999 LP变量，变量号：0-999
		VL=<直线运行速度>	直线运行速度，单位MM/S，最大值为参数直线运动最高速度。
		PL=<平滑度>	PL平滑度：0-9
		<变量>POINT	变量范围：1-3。一段完整的整圆指令必须要有三个点（P1点不能省略或用上一个点），每个点的意义如下：P1确定圆中心点，确定运动到圆心的速度，确定是顺时针还是逆时针画圆；P2点，确定圆的起始点方向，机器人画圆的速度；P3点，确定圆所在的平面）
	注意	若在程序中使用不同的工具或用户坐标系，请使用CHANGE坐标系的指令 若在画圆的中途停止，再次运行时，机器人会运行到圆心位置再重新画圆。 若该指令中缺少一个点（P1-P3),机器人会报错。	

程序举例：

1 MOVL VL=100 PL=0；过渡点

2 MOVL VL=100 PL=0；圆心点

3 MOVCA VL=100MM/S PL=0 POINT=1 R=10.0 Dir=CW； P1确定圆中心点，确定运动到圆心的速度，确定是顺时针还是逆时针画圆（CW表示顺时针 CCW表示逆时针）。

4 MOVCA VL=100MM/S PL=0 POINT=2；P2点，确定圆的起始点方向，机器人画圆的速度。

7.1.5 运动指令附加项说明

运动指令附加项	[空白]	不使用附加项。
	UNTIL	[UNTIL]：使用条件，当条件满足，该程序行停止执行，转入下一行执行。否则执行当前行直到结束后，转入下一行。详见辅助项说明1。
	COORD	附加轴1和附加轴2同时协同，详见辅助项说明4
	COORD1	附加轴1单独协同，详见辅助项说明4
	COORD2	附加轴2单独协同，详见辅助项说明4
	ST+	强制执行远边动作

运动指令附加项	ST-	强制执行近边动作
	ACC	关节加速度，值越大加速越快
	OP	偏移变量，用于偏移示教轨迹
	DOUT	运动中输出状态信号，详见下文“ 2.DOUT用法举例 ”
	AOUT	运动中输出模拟量信号，详见下文“ 3.AOUT用法举例 ”
	OP	偏移变量，用于偏移示教轨迹，详见下文“ 5.OP用法举例 ”

1. UNTIL用法举例

使用条件，当条件满足，该程序行停A止执行，转入下一行执行。否则执行当前行直到结束后，转入下一行。条件可以使用X、M、T、C。本功能可用于沾浆、舀铝水、放板机等。

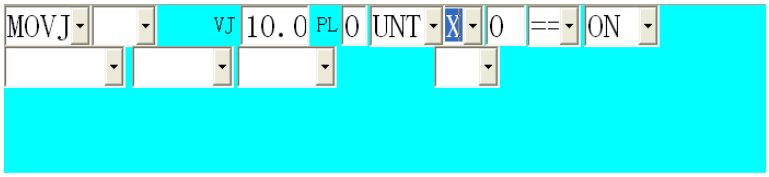
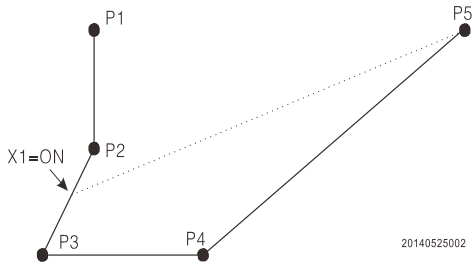


图 7.7



点位	行数	程序
P1	1	MOVJ VJ=30% PL=0
P2	2	MOVL VL=500MM/S PL=0
P3	3	MOVL VL=200MM/S PL=0 UNTIL X#(1)==ON
	4	JUMP *22 IF X#(1)==ON
P4	5	MOVL VL=200MM/S PL=0
	6	*22
P5	7	MOVJ VJ=30% PL=0

★说明：

1.X1有效后,系统退出当前执行的第三行程序,直接跳转到第7行执行。

2.X1有效时间要长点，否则会执行5行程序。

2. DOUT用法举例

本附加项用于，设定程序开始多少距离切换某个信号（M、Y）有效或无效，到离结束点多少距离再还原该信号。

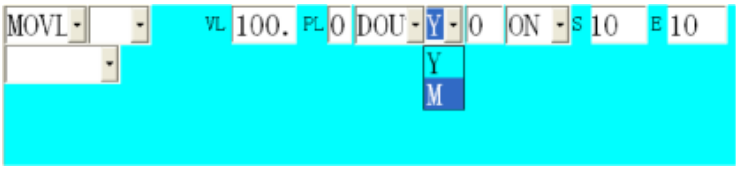


图 7.8

程序实例：
MOVL VL=500MM/S PL=0 DOUT Y#(0)==ON START 10.0 END 10.0
程序说明：
执行本程序后，运动10MM后，Y0口开启（ON），机器人运动到距离结束点10MM时，关闭Y0口（OFF），然后程序运行到结束点。
本功能可用于喷涂运动中开关枪等。

3. AOUT用法举例

AOUT用法举例
本附加项用于机器人在运动过程中或结束位置输出指定模拟量。

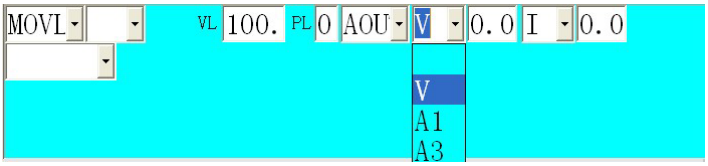


图 7.9

1）V I模拟量输出：

V I模拟量输出只能用于焊接指令中间。指令不在焊接指令之间时，系统执行到该指令行，系统报警，并提示：没有焊接状态，不能在运动中使用AOUT改变焊接电流电压。
V对应焊机电压，I对应焊接电流。执行本指令从起点到终点时，模拟电压由焊接电压到指定电压线性变化。程序举例：

焊接电流200A对应模拟量4.95V，焊接电压20V对应模拟量4.95V；焊接电流400A对应模拟量9.95V，焊接电流40V对应模拟量9.95V。

ARCSTART#（0）
MOVL VL=500MM/S PL=0 AOUT V=40.0 I=400.0A
ARCEND#（0）
程序说明：程序执行到第1行时，A1 A2先输出起弧电压；第1行指令结束时，A1 A2模拟量输出4.95V（焊接电流电压）。执行第2行到结束时，A1 A2模拟量从4.95V线性变化到9.95V。执行第3行时电压又变为灭弧电压。
可用于焊接电流电压需要变化的场合。

2）A1 A2模拟量输出：

在焊接指令之间使用本指令时，将不生效。系统按照焊接工艺设定输出。不在焊接指令之间时，系统执行到本指令行结束再输出A1

A2对应模拟量。简称到点输出。

3）A3 A4模拟量输出：

指令不管在焊接指令之间，还是不在焊接指令之间，系统执行本指令行时，机器人从起点到终点，A3 A4模拟量从开始点的值线性变化到结束点指令指定数值。简称线性输出。

程序举例：

MOVJ VJ=30% PL=0 ； A点 A3=0 A4=0

MOVL VL=500MM/S PL=0 AOUT A3=10.0 A4=10.0 ； B点

程序说明：机器人移动从A点运行到B点，A3 A4模拟量由0V线性变化到10V。

4. COORD用法举例

COORD用法举例：

本附加项用于机器人在直线、圆弧运动时，和外部轴实现协同动作的作用。

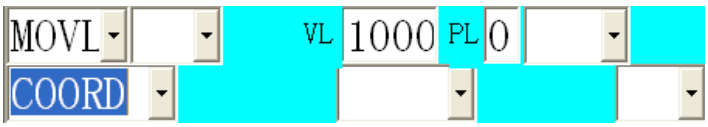


图 7.10

程序举例：

1 MOVL VL=100MM/S PL=0 TOOL=1 COORD；直线运动，协同轴1、2同时开启。

2 MOVL VL=200MM/S PL=0 TOOL=1 COORD1；直线运动，协同轴1开启。

3 MOVC VL=100MM/S PL=0 TOOL=1 POINT=2 COORD2；圆弧运动，协同轴2开启

4 MOVC VL=100MM/S PL=0 TOOL=1 POINT=3 COORD；圆弧运动，协同1、2同时开启。

5. OP用法举例

OP用法指令：

本附加项用于机器人在直线、圆弧运动中带入局部OP偏移量。

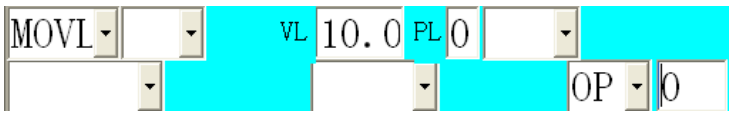


图 7.11

程序举例：

1 MOVL VL=100MM/S PL=0 TOOL=1 OFFSET OP#(0)；带入OP0局部偏移，移动到1点。

2 MOVL VL=100MM/S PL=0 TOOL=1 OFFSET OP#(0)；带入OP0局部偏移，移动到2点。

3 MOVL VL=100MM/S PL=0 TOOL=1 OFFSET OP#(2)；带入OP2局部偏移，移动到3点。

★ 注意
OP变量的值可以在：运行准备→变量→寻位变量→OP变量中查看和修改。

7.2 逻辑指令

7.2.1 数字量输出（DOUT)

数字量输出（DOUT）	指令功能	控制变量的状态。数字量只有两种形式，因此在使用该指令时只有两种状态，即“ON（有效）”和“OFF（无效）”两种状态。	
	附加项	Y<变量号>=ON/OFF	控制输出口Y<变量号>，ON或OFF状态，变量号范围参考《硬件说明书》或《PLC说明书》。
		M<变量号>=ON/OFF	
	程序举例	DOUT Y#(0)=ON　控制输出口Y0为ON状态 DOUT M#(0)=OFF　控制辅助继电器M0为OFF状态	

7.2.2 模拟量输出（AOUT)

模拟量输出（AOUT）	指令功能	输出模拟量口的模拟电压。	
	附加项1	AO#<变量号>=	指定需要输出的模拟量端口号，范围S40两路A1-A2，S80四路A1-A4。
	附加项2	<变量>	变量为需要输出的模拟量电压值，范围0.000-10.000V。
	程序举例	AOUT AO#(1)=0.000　输出A1口模拟电压为0V AOUT AO#(2)=10.000　输出A2口模拟电压为10V	

7.2.3 条件等待（WAIT）

条件等待（WAIT）	指令功能	当您所设定的条件满足时，则程序往下执行；当您所设定的条件不满足时，则程序一直停在这里，直到满足您所设定的条件为止。若设定了后面的时间，当条件不满足时，设定时间满足后则会继续执行下面的程序。	
	附加项1	X<变量号>==ON/OFF	判断输入口X<变量号>的状态是ON还是OFF。变量号范围参考《硬件说明书》。
		M<变量号>==ON/OFF	判断辅助继电器M<变量号>的状态是ON还是OFF。变量号范围参考《PLC说明书》。

条件等待（WAIT）	附加项2	T=<变量>	等待时间，单位：1MS。T=0一直等待所设条件；T≠0时，条件和时间一个先到就往下执行。
	程序举例	WAIT X#(0)==ON T=0 持续等待X0口有效。 WAIT M#(1)==OFF T=500 在500MS内等待M1继电器有效，否则顺序执行。	

7.2.4 延时指令（TIME）

延时指令（TIME）	指令功能	在指定时间能等待（延时），时间结束，程序往下执行。	
	附加项	T=<变量>	指定（延时）等待时间，变量单位：1MS，范围：0-99999
	程序举例	TIME T=50　　延时50MS	

7.2.5 后台延时输出（TDOUT））

后台延时输出（TDOUT）	指令功能	用于在后台延时输出状态，该指令在后台执行，不占用程序时间。		
	附加项1	Y<变量号>	=	ON/OFF
		M<变量号>		
	附加项2	T=<变量>	延时时间，变量单位ms，范围为0-99999	
	程序举例	DOUT Y#(0)=ON　　；输出Y#(0）为ON TDOUT Y#(0)=OFF T=1000　　；后台延时1000ms输出Y#(0)为OFF		

7.2.6 暂停（PAUSE）

<table>
<tr><td rowspan="13">暂停
(PAUSE)</td><td>指令功能</td><td colspan="4">1. 无条件暂停，程序暂停，直到按“运行键”程序再继续执行。
2. 有条件暂停，只有当后面条件满足时，程序暂停，否则连续运行。</td></tr>
<tr><td rowspan="2">附加项1</td><td colspan="2">[空白]</td><td colspan="2">无条件暂停（无附加项2）</td></tr>
<tr><td colspan="2">IF</td><td colspan="2">条件暂停：条件满足暂停，否则顺序执行。</td></tr>
<tr><td rowspan="11">附加项2</td><td>X<变量号></td><td rowspan="11">==
＞
＜
>=
<=
=</td><td rowspan="5">ON
OFF</td><td rowspan="11">各变量号范围：
X/Y：0-559
M：0-4059
GI：0-1023
LI：0-999
GD：0-299
LD/GP/LP：0-999
TC/T：0-59
C/CC：0-19
GS：0-99</td></tr>
<tr><td>M<变量号></td></tr>
<tr><td>Y<变量号></td></tr>
<tr><td>T<变量号></td></tr>
<tr><td>C<变量号></td></tr>
<tr><td>GI<变量号>
LI<变量号>
GD<变量号>
LD<变量号>
GP<变量号>
<数据号>
LP<变量号>
<数据号>
TC<变量号>
CC<变量号></td><td>GI<变量号>
LI<变量号>
GD<变量号>
LD<变量号>
GP<变量号>
LP<变量号>
TC<变量号>
CC<变量号>
D<变量号></td></tr>
<tr><td>GS<变量号></td><td><字符串></td></tr>
<tr><td>程序举例</td><td colspan="4">PAUSE ；无条件暂停
PAUSE IF X#(0)==ON ； 只有当X0=ON时才暂停，否则继续执行。
PAUSE IF GD#(1)==LD#(1) ；只有当GD1=LD1时，程序才暂停，否则继续执行。</td></tr>
</table>

7.2.7 跳转（JUMP）

<table>
<tr><td rowspan="8">条件跳转
（JUMP）</td><td>指令
功能</td><td colspan="4">程序跳转指令，分有条件和无条件跳转。
说明：
1、使用此条指令时，要配合使用标号指令（*）。标号就是您所要将程序跳转到的位置。
2、后面不加条件，只要程序执行到此行，则直接跳到标号所处的位置。
3、后面有条件，当程序执行到该行指令时，程序不一定跳转，只有当后面的条件满足时，程序才跳转到标号所处的位置。条件不满足，程序顺序执行。</td></tr>
<tr><td>附加
项1</td><td>*<变量></td><td colspan="3">跳转标号、标记，表示跳转到具有*<变量>的位置；变量可以为任意字符、数字。</td></tr>
<tr><td rowspan="2">附加
项2</td><td>[空白]</td><td colspan="3">无条件跳转</td></tr>
<tr><td>IF</td><td colspan="3">条件跳转：
若条件满足，跳转到*<变量>所在位置。
若条件不满足，程序顺序执行。</td></tr>
<tr><td rowspan="11">附加
项3</td><td>X<变量号></td><td rowspan="11">==
>
<
>=
<=
=</td><td rowspan="5">ON
OFF</td><td rowspan="11">各变量号范围：
X/Y：0-559
M：0-4059
GI：0-1023
LI：0-999
GD：0-299
LD/GP/LP：0-999
TC/T：0-59
C/CC：0-19
GS：0-99</td></tr>
<tr><td>M<变量号></td></tr>
<tr><td>Y<变量号></td></tr>
<tr><td>T<变量号></td></tr>
<tr><td>C<变量号></td></tr>
<tr><td>GI<变量号>
LI<变量号>
GD<变量号>
LD<变量号>
GP<变量号>
<数据号>
LP<变量号>
<数据号>
TC<变量号>
CC<变量号>
D<变量号></td><td>GI<变量号>
LI<变量号>
GD<变量号>
LD<变量号>
GP<变量号>
LP<变量号>
TC<变量号>
CC<变量号></td></tr>
<tr><td>GS<变量号></td><td><字符串></td></tr>
<tr><td>程序
举例</td><td colspan="4">1 * 345T ；标号*345T位置
……
6 JUMP *SES IF M#(1)==ON ；如果M1=ON，跳转到*SES位置。
……
12 JUMP *345T ；无条件跳转*345T位置。
13 *SES ；标号*SES位置
…… ；后续程序</td></tr>
</table>

7.2.8 子程序调用（CALL）

<table>
<tr><td rowspan="8">子程序调用
（CALL）</td><td>指令
功能</td><td colspan="4">子程序调用指令，包含有条件调用和无条件调用。
说明：
1、子程序的建立和主程序的建立唯一的区别就是在编写完所有的程序之后，子程序的末尾加上RET指令。
2、直接选择或输入程序文件名即可（后面可不加条件），只要程序执行到此行，则直接调用该子程序；若后面有条件，只有当后面的条件满足时，才调用该子程序。
3、在使用call无条件指令时，我们在机器人内部设有固定的子程序调用，用来控制滑台及喷枪（例：自转90度、一枪开启等）。</td></tr>
<tr><td>附加
项1</td><td><程序名></td><td colspan="3">程序名为调用的子程序名称。可通过下拉菜单选择调用的子程序名。</td></tr>
<tr><td rowspan="3">附加
项2</td><td>[空白]</td><td colspan="3">无条件调用，运行到本行直接调用<程序名>(子程序的程序名)。</td></tr>
<tr><td>IF</td><td colspan="3">条件调用：
条件满足，调用<程序名>指定的子程序。
条件不满足，程序顺序执行。</td></tr>
<tr><td>GS</td><td colspan="3"></td></tr>
<tr><td rowspan="13">附加
项3</td><td>X<变量号></td><td rowspan="13">==
>
<
>=
<=
=</td><td rowspan="5">ON
OFF</td><td rowspan="13">各变量号范围：
X/Y：0-559
M：0-4059
GI：0-1023
LI：0-999
GD：0-299
LD/GP/LP：0-999
TC/T：0-59
C/CC：0-19
GS：0-99</td></tr>
<tr><td>M<变量号></td></tr>
<tr><td>Y<变量号></td></tr>
<tr><td>T<变量号></td></tr>
<tr><td>C<变量号></td></tr>
<tr><td>GI<变量号>
LI<变量号>
GD<变量号>
LD<变量号>
GP<变量号>
<数据号>
LP<变量号>
<数据号>
TC<变量号>
CC<变量号>
D<变量号></td><td rowspan="8">GI<变量号>
LI<变量号>
GD<变量号>
LD<变量号>
GP<变量号>
LP<变量号>
TC<变量号>
CC<变量号></td></tr>
<tr><td>GS<变量号></td><td><字符串></td></tr>
<tr><td>程序
举例</td><td colspan="4">CALL 125 ；调用程序名为125的子程序
CALL %SBS IF LD#(1) > LD#(2) ；如果LD1>LD2，则调用SBS子程序。</td></tr>
</table>

7.2.9 注释（；）

注释（；）	指令功能	注释指令，解释说明。在执行程序时，此部分的内容不执行，相当于提示使用者这里是什么意思，主要方便读者更加轻松的理解该程序。	
	附加项	；<注释内容>	注释内容为用户自己定义内容。便于用户自己查看，理解程序。
	程序举例	；tqzl ：注释内容tqzl，本行不执行。	

注释也可在编辑新程序或修改指令时，直接在指令弹框下“注释”后的空白框进行编辑，点击指令确定后，注释显示于程序后。

7.2.10 跳转标号（*）

跳转标号（*）	指令功能	标记JUMP跳转位置，需和JUMP指令配合使用。	
	附加项	*<变量>	变量可以为任意字符或数字。
	程序举例	* ssb4 ：标记跳转位置：*ssb4 ……	

7.2.11 子程序返回（RET)

RET	指令功能	用于子程序的返回。返回调用程序的界面，接CALL程序行后继续运行主程序。	
	附加项	无	
	程序举例	RET ;子程序返回	

7.3 运算指令

7.3.1 加法运算（ADD)

<table>
<tr><td rowspan="11">加法运算（ADD）</td><td>指令功能</td><td colspan="4">将前一变量和后一变量相加，结果赋值给前一个变量。如：A=A+B。本指令只能使用数据变量。</td></tr>
<tr><td rowspan="2">附加项1</td><td rowspan="2">GI<变量号>
LI<变量号>
GD<变量号>
LD<变量号>
GP<变量号><数据号>
LP<变量号><数据号>
TC<变量号>
CC<变量号></td><td rowspan="2">附加项2</td><td rowspan="2">GI<变量号>
LI<变量号>
GD<变量号>
LD<变量号>
GP<变量号><数据号>
LP<变量号><数据号>
TC<变量号>
CC<变量号>
D<变量></td><td rowspan="2">将前一变量和后一变量相加，结果赋值给前一变量。</td></tr>
<tr></tr>
<tr><td rowspan="2">附加项3</td><td>[空白]</td><td colspan="2">不参考点</td><td rowspan="5">只在做GP运算时使用。不用时均选择空白。</td></tr>
<tr><td>参考GP<变量号></td><td colspan="2">参考本GP点作点运算</td></tr>
<tr><td rowspan="4">附加项4</td><td>[空白]</td><td colspan="2">无参考坐标系</td></tr>
<tr><td>用户</td><td colspan="2">参考用户坐标系</td></tr>
<tr><td>工具</td><td colspan="2">工具坐标系</td></tr>
<tr><td>协同</td><td colspan="2">协同坐标系</td></tr>
<tr><td>程序举例</td><td colspan="4">ADD TC#(4) GP#1(1) ： TC4=TC4+GP1 ；X轴坐标
ADD CC#(1) 20.5 ： CC1=CC1+20.5</td></tr>
</table>

7.3.2 减法运算（SUB）

<table>
<tr><td rowspan="8">加法运算（ADD）</td><td>指令功能</td><td colspan="4">将前一变量和后一变量相加，结果赋值给前一个变量。如：A=A-B。本指令只能使用数据变量。</td></tr>
<tr><td>附加项1</td><td>GI<变量号>
LI<变量号>
GD<变量号>
LD<变量号>
GP<变量号> <数据号>
LP<变量号> <数据号>
TC<变量号>
CC<变量号></td><td>附加项2</td><td>GI<变量号>
LI<变量号>
GD<变量号>
LD<变量号>
GP<变量号> <数据号>
LP<变量号> <数据号>
TC<变量号>
CC<变量号>
D<变量></td><td>将前一变量和后一变量相减，结果赋值给前一变量。</td></tr>
<tr><td rowspan="2">附加项3</td><td>[空白]</td><td colspan="2">不参考点</td><td rowspan="5">只在做GP运算时使用。不用时均选择空白。</td></tr>
<tr><td>参考GP<变量号></td><td colspan="2">参考本GP点作点运算</td></tr>
<tr><td rowspan="4">附加项4</td><td>[空白]</td><td colspan="2">无参考坐标系</td></tr>
<tr><td>用户</td><td colspan="2">参考用户坐标系</td></tr>
<tr><td>工具</td><td colspan="2">工具坐标系</td></tr>
<tr><td>协同</td><td colspan="2">协同坐标系</td></tr>
<tr><td>程序举例</td><td colspan="4">SUB TC#(4) GP#1(1) ： TC4=TC4-GP1 ； X轴坐标
SUB CC#(1) 20.5 ： CC1=CC1-20.5</td></tr>
</table>

7.3.3 乘法运算（MUL）

乘法运算（MUL）	指令功能	将前一变量乘以后一变量，结果赋值给前一个变量。如：A=A×B。本指令只能使用数据变量。		
	附加项	GI<变量号> LI<变量号> GD<变量号> LD<变量号> GP<变量号><数据号> LP<变量号><数据号> TC<变量号> CC<变量号>	GI<变量号> LI<变量号> GD<变量号> LD<变量号> GP<变量号><数据号> LP<变量号><数据号> TC<变量号> CC<变量号> D<变量>	将前一变量乘以后一变量，结果赋值给前一个变量
	程序举例	MUL TC#(4) GP#1(1) ： TC4=TC4×GP1 X轴坐标 MUL CC#(1) 20.5 ： CC1=CC1×20.5		

7.3.4 除法运算（DIV）

乘法运算（DIV）	指令功能	将前一变量除以后一变量，结果赋值给前一个变量。如：A=A÷B。本指令只能使用数据变量。		
	附加项	GI<变量号> LI<变量号> GD<变量号> LD<变量号> GP<变量号><数据号> LP<变量号><数据号> TC<变量号> CC<变量号>	GI<变量号> LI<变量号> GD<变量号> LD<变量号> GP<变量号><数据号> LP<变量号><数据号> TC<变量号> CC<变量号> D<变量>	将前一变量除后一变量，结果赋值给前一个变量。
	程序举例	DIV TC#(4) GP#1(1) ： TC4=TC4÷GP1 X轴坐标 DIV CC#(1) 20.5 ： CC1=CC1÷20.5		

7.3.5 加1运算（INC）

加1运算（INC）	指令功能	将变量加数字1，结果赋值给变量。如：A=A+1。本指令只能使用数据变量，多用于计数。	
	附加项	GI<变量号>	将变量加数字1，结果赋值给变量。
		LI<变量号>	
		GD<变量号>	
		LD<变量号>	
		GP<变量号> <数据号>	
		LP<变量号> <数据号>	
		TC<变量号>	
		CC<变量号>	
	程序举例	INC TC#(4) ： TC4=TC4+1 INC CC#(1) ： CC1=CC1+1	

7.3.6 减1运算（DEC）

减1运算（DEC）	指令功能	将变量减数字1，结果赋值给变量。如：A=A-1。本指令只能使用数据变量，多用于计数。	
	附加项	GI<变量号>	将变量减数字1，结果赋值给变量。
		LI<变量号>	
		GD<变量号>	
		LD<变量号>	
		GP<变量号> <数据号>	
		LP<变量号> <数据号>	
		TC<变量号>	
		CC<变量号>	
	程序举例	DEC TC#(4) ： TC4=TC4-1 DEC CC#(1) ： CC1=CC1-1	

7.3.7 赋值指令

赋值 （SET）	指令功能	将后一变量的值赋给前一变量，如A=B。本指令只能使用数据变量。		
	附加项	GI<变量号> LI<变量号> GD<变量号> LD<变量号> GP<变量号> <数据号> LP<变量号> <数据号> TC<变量号> CC<变量号>	GI<变量号> LI<变量号> GD<变量号> LD<变量号> GP<变量号> <数据号> LP<变量号> <数据号> TC<变量号> CC<变量号> D<数据>	将后一变量的值赋给前一变量。
	程序举例	SET TC#(4) GP#1(1) ： TC4=GP1 X轴坐标 SET CC#(1) TC#(1) ： CC1=TC1 SET TC#(1) 5.000 ： TC1=5		

7.4 码垛指令

码垛 （PALLET）	指令功能	运行码垛工艺。运行本指令，程序将调用预先设定好的码垛工艺，变量号为工艺号码，范围0-199。 码垛相关信息详见《码垛工艺说明》。	
	附加项	#<变量号>	变量号为需要调用的码垛工艺号码。
	程序举例	PALLET#(1) ： 调用码垛工艺1	

7.5 焊接指令

7.5.1 起弧（ARCSTART）、收弧指令（ARCEND）

起弧 (ARC START) 收弧(ARC END)	指令功能	运行本指令，程序将调用预先设定好的焊接参数，起弧。变量号为焊接参数文件号，范围0-7。 该两条指令成对使用。焊接相关信息详见《焊接工艺说明》。	
	附加项	#<变量号>	变量号为需要调用的焊接参数文件号。
	程序举例	ARCSTART#(1) 调用焊接参数文件1，起弧开始焊接 …… ARCEND#(1) 起弧结束，焊接完成	

7.5.2 摆弧（WEAVE）、摆弧结束（WEAVEEND）

摆弧 （WEAVE） 摆弧结束 (WEAVEEND)	指令功能	运行本指令，程序将调用预先设定好的摆弧参数，摆弧。变量号为摆动焊接参数文件号，范围0-7。 该两条指令成对使用 摆弧相关信息详见《焊接工艺说明》。	
	附加项	#<变量号>	变量号为配对使用摆弧指令的焊接参数文件号。
	程序举例	WEAVE#(1) …… WEAVEEND#(1)	调用摆弧参数文件1，摆弧。 摆弧路径 摆弧结束

焊接摆弧综合实例：

```
ARCSTART #(1)          ；调用1号焊接参数进行焊接
WEAVESINE #(1)         ；调用1号摆动参数
MOVL VL=100MM/S PL=0       ；走焊接轨迹
WEAVEEND     ；摆动结束
ARCEND #(1)    ；1号焊接工艺结束
```

7.5.3 伺服通讯点焊（SPOT）

伺服通讯点焊 (SPOT)	指令功能	针对焊接压力曲线工艺，配合点焊焊机使用 注：要使用SPOT指令，需先在机构参数27号设置为5（点焊工艺），才能选择。	
	附加项	#<文件号>	焊接压力曲线文件号
		S<文件号>	通讯焊机内焊接工艺
		P<变量>	点焊保持压力
	程序举例	MOVL…　　；直线运动一段距离 SPOT(0) S(0)P(20)　；调用系统点焊压力曲线文件（0）和焊机内部焊接工艺（0），保持压力（20） MOVL…　　；直线运动一段距离 SPOT(0) S(0)P(20) MOVL…	

7.5.4 鱼鳞焊指令

<table>
<tr><td rowspan="5">鱼鳞焊开始
(STITCHSTART)

鱼鳞焊结束
（STITCHEND)</td><td>指令功能</td><td colspan="4">STITCHSTART鱼鳞纹焊接开始，与起弧指令ARCSTART配合使用。
STITCHEND鱼鳞纹焊接结束，与收弧指令ARCEND配合使用。</td></tr>
<tr><td rowspan="3">附加项</td><td>T<变量>ms</td><td>点焊时间，单位ms</td><td rowspan="3">L2
<变量>mm</td><td rowspan="3">空走距离，单位mm</td></tr>
<tr><td>L1<变量>mm</td><td>焊接距离，单位mm</td></tr>
<tr><td>KL1
<变量>mm</td><td>焊接距离，单位mm
使用鱼鳞焊KL1功能后，焊接时不检测起弧及灭弧反馈信号，机器人在鱼鳞焊时，不会停止，一直运动到灭弧点。</td></tr>
<tr><td>程序举例</td><td colspan="4">ARCSTART#(0) ；打开焊接工艺（0）
STITCHSTART T=300 L2=3.0 ；开始鱼鳞焊接，点焊300ms,空走3mm，再点焊300ms
MOVL VL=30MM/S PL=0 ；焊接轨迹，鱼鳞焊功能接入，空走的速度为30MM/S,焊接轨迹是直线
STITCHEND ；鱼鳞焊接结束
ARCEND#(0) ；焊接工艺（0）结束</td></tr>
</table>

7.5.5 打开、关闭激光指令

<table>
<tr><td rowspan="3">打开激光
(OPENLASER)

关闭激光
（CLOSELASER）</td><td>指令功能</td><td colspan="2">在激光寻位中，用于打开和关闭激光。</td></tr>
<tr><td>附加项</td><td>#<文件号></td><td>系统设置的对应激光跟踪工艺号。</td></tr>
<tr><td>程序举例</td><td colspan="2">OPENLASER#(0) ；打开激光，调用激光工艺（0）
CLOSELASER ；关闭激光</td></tr>
</table>

7.5.6 激光搜寻（SEARCHLASER)

<table>
<tr><td rowspan="3">激光搜寻
（SEARCHLASER)</td><td>指令功能</td><td colspan="2">在激光跟踪中，用于激光搜寻焊缝起点。</td></tr>
<tr><td>附加项</td><td>#<文件号></td><td>激光跟踪器上设置的对应激光跟踪工艺号。</td></tr>
<tr><td>程序举例</td><td colspan="2">ASEARCHLASER#(0) ；激光搜寻焊缝起点，调用激光跟踪器工艺（0）</td></tr>
</table>

7.5.7 打开、关闭激光跟踪指令

打开激光跟踪(OPENLASERTRACK) 关闭激光跟踪（CLOSELASERTRACK）	指令功能	在激光跟踪焊接中，用于打开和关闭激光跟踪。	
	附加项	#<文件号>	跟踪器对应激光跟踪工艺号。
	程序举例	OPENLASERTRACK#(0) ；打开激光功能，调用激光工艺（0） CLOSELASERTRACK ；关闭激光跟踪	

激光跟踪综合示例：

MOVJ VJ=50% PL=0 ；关节运动

OPENLASER#(0) ；打开激光你，调用系统（0）号激光工艺

MOVL VL=100MM/S PL=0 ；直线运动到焊接起始点

SEARCHLASER#(1) ；激光搜寻，调用跟踪器（1）号工艺

ARCSTART #(0) ；焊接开始，调用（0）号焊接工艺

OPENLASERTRACK#(1) ；打开激光跟踪，调用跟踪器（1）号工艺

MOVL VL=10MM/S PL=0；焊接轨迹

CLOSELASERTRACK ；关闭激光跟踪

ARCEND#(0) ；焊接结束

CLOSELASER ；关闭激光

7.6 辅助指令

7.6.1 速度改变（SPEED）

速度改变（SPEED）	指令功能	本指令通过附加项的比例值，乘以之后程序行的速度，来调整其后关节运行行速度。 VJ<比例>对其后VJ速度有效。VL<比例>对气候VL速度有效。 SPEED VJ=100或SPEED VL=100取消速度改变功能。 本指令主要用于需要对某部分程序行调速时使用。而再现模式中的运行倍率对整个程序都有效。	
	附加项1	VJ=<比例>	范围：0-100，对其后VJ速度有效。
		VL=<比例>	范围：0-100，对其后VL速度有效。
	附加项2	[空白]	
		GI<变量>	范围：1-4096

程序举例:

MOVL VL=500MM/S PL=0 ；VL速度变为：500X30%=150MM/S。 ；之后VL速度均乘以100%。

SPEDE VJ=100 ；

MOVL VL=500MM/S PL=0 ； VL速度变为：500X100%=500MM/S，等同于没有变速、变速取消。

7.6.2 条件判断 （IF）

<table>
<tr><td rowspan="5">1.（IF）
2.（ELSEIF）</td><td>指令功能</td><td colspan="4">IF条件判断指令。本指令由IF、ELSEIF（可省略或重复使用）、ELSE、ENDIF构成一个完整结构。
首先判断IF条件是否成立，成立则执行IF之后的语句。如果IF条件不成立，则判断ELSEIF之后条件（本指令行可根据实际选择使用或不使用），ELSEIF条件成立，则执行ELSEIF之后程序。如果ELSEIF条件不成立，则执行ELSE之后程序。程序结尾使用ENDIF。</td></tr>
<tr><td rowspan="3">附加项1</td><td>X<变量号>
M<变量号>
Y<变量号>
T<变量号>
C<变量号></td><td>==</td><td>ON/
OFF</td><td rowspan="3">IF条件判断，条件成立，执行该语句之后程序。
如果IF条件不成立，执行ELSEIF条件判断，如果ELSEIF之后条件成立，则执行ELSEIF之后程序。
如果IF和ELSEIF之后条件都不成立，则执行ELSE之后程序。
ENDIF结束IF判断。</td></tr>
<tr><td>GI<变量号>
LI<变量号>
GD<变量号>
LD<变量号>
GP<变量号>
<数据号>
LP<变量号>
<数据号>
TC<变量号>
CC<变量号></td><td>>
<
>=
<=
=</td><td>GI<变量号>
LI<变量号>
GD<变量号>
LD<变量号>
GP<变量号>
LP<变量号>
TC<变量号>
CC<变量号>
D<变量号></td></tr>
<tr><td>GS<变量号></td><td></td><td><字符串></td></tr>
<tr><td>附加项2</td><td><编号></td><td colspan="3">指令结构标号，一个完整的指令结构内该标号必须一致，否则程序将报错或跳出。
一个完整指令结构所使用的编号和另一个完整指令结构编号可以相同。
标号范围：0-8。</td></tr>
<tr><td>3.（ELSE）
4.（END IF）</td><td>附加项</td><td><编号></td><td colspan="3">指令结构标号，标号范围：0-8。</td></tr>
</table>

程序举例1（IF）：

RIF X#(0)==ON 0 ;如果X0=ON执行下条指令。

```
INC TC#(0)          ；TC0=TC0+1
ELSEIF X#(1) = =OFF   ；如果X0=OFF，执行下条指令。
DEC TC#(0)      ；TC0=TC0-1
ELSE 0   ；如果都不满足，则执行下条指令。
ADD TC#(0) TC#(0)    ；TC0=TC0+TC0
END IF 0      ；IF指令结束
```

程序举例2（ELSEIF）：

```
IF X#(0)==ON 0     ;如果X0=ON。执行TC0=TC0+1
INC TC#(0)
ELSEIF X#(1)==ON 0      ;如果X1=ON，执行TC0=TC0+2
ADD TC#(0) 2.000
ELSEIF X#(2)==ON 0       ;如果X2=ON，执行TC0=TC0+3
ADD TC#(0) 3.000
ELSE 0           ;否则，执行
TC0=TC0+TC0
ADD TC#(0) TC#(0)
ENDIF 0    ;0号IF指令结构结束
```

程序举例3（ELSE）：

```
IF X#(1)==ON 0   ；如果X1=ON。执行TC0=TC0+1
INC TC#(0)
ELSE 0    ；否则执行TC0=TC0+3
ADD TC#(0) 3.000
END IF 0    ；0号IF指令结构结束
IF X#(1)==OFF 0   ；如果X1=OFF，执行TC0=TC0+4
ADD TC#(0) 4.00
ELSE 0         ；否则执行TC0=TC0+5
ADD TC#(0) 5.00
END IF 0       ；0号IF指令结构结束
```

7.6.3 FOR（备用）

FOR（备用）	指令功能		
	附加项		
	程序举例		

7.6.4 循环（WHILE）（ENDWHILE）

<table>
<tr><td>指令功能</td><td colspan="5">WHILE循环指令，本指令由WHILE和ENDWHILE构成一个完整结构。
当WHILE后的条件满足要求时，即条件为ON时，执行WHILE和ENDWHILE里面的程序，直到WHILE条件后的指令不满足要求，则退出该循环。
判断条件一定要在循环部分中进行设置，否则会死循环。</td></tr>
<tr><td rowspan="3">循环
（WHILE）</td><td rowspan="2">附加项1</td><td>X<变量号>
M<变量号>
Y<变量号>
T<变量号>
C<变量号></td><td rowspan="2">==
>
<
>=
<=
=</td><td>ON/
OFF</td><td rowspan="2">WHILE循环指令的条件判断。
当条件满足要求时，即条件为ON时，执行WHILE和ENDWHILE之间的程序行。
当条件不满足时，程序跳出该循环区间，执行ENDWHILE之后程序行。</td></tr>
<tr><td>GI<变量号>
LI<变量号>
GD<变量号>
LD<变量号>
GP<变量号>
<数据号>
LP<变量号>
<数据号>
TC<变量号>
CC<变量号>
GS<变量号></td><td>GI<变量号>
LI<变量号>
GD<变量号>
LD<变量号>
GP<变量号>
LP<变量号>
TC<变量号>
CC<变量号>
D<变量号>
<字符串></td></tr>
<tr><td>附加项2</td><td><编号></td><td colspan="3">指令结构标号，一个完整的指令结构内该标号必须一致，否则程序将报错或跳出。
一个完整指令结构所使用的编号和另一个完整指令结构编号可以相同。
标号范围：0-8。</td></tr>
<tr><td>结束循环
（ENDWHILE）</td><td>附加项1</td><td><编号></td><td colspan="3">结束指令结构标号，标号范围：0-8。</td></tr>
</table>

程序举例1：

WHILE <条件><编号>　　；判断条件，成立：执行……部分程序行； 不成立：执行ENDWHILE后程序行。

……

ENDWHILE <编号>

……

程序举例2：

```
SET TC#(1) 0.000     ；赋值TC1=0
WHILE TC#(1) < 20 0    ；如果TC1<20，执行IF部分
IF X#(1)==ON 0     ；IF判断：如果X1=ON则TC1=TC1+1
INC TC#(1)
ELSE 0       ；否则TC1=TC1+2
ADD TC#(1) 2.000
ENDIF 0      ；IF判断结束
ENDWIHILE 0     ；如果TC1＜20，则执行TC1=15
SET TC#(1) 15.000
```

7.6.5 条件选择（SWITCH →SWITCH ）

<table>
<tr><td rowspan="5">1.SWITCH</td><td>指令功能</td><td colspan="2">SWITCH条件选择指令，本指令由SWITCH、CASE（可重复使用）、BREAK、DEFAULT、ENDSWITCH构成一个完整的指令结构。
计算SWITCH后变量的值，判断和那个CASE后的数值相等。
找到相等CASE后，程序从该位置开始执行，执行到第一个BREAK结束，并跳转到ENDSWITCH行。</td></tr>
<tr><td rowspan="3">附加项1</td><td>X<变量号>
M<变量号>
Y<变量号>
T<变量号>
C<变量号></td><td>状态变量，CASE内容为0或者1</td></tr>
<tr><td>GI<变量号>
LI<变量号>
GD<变量号>
LD<变量号>
GP<变量号><数据号>
LP<变量号> <数据号>
TC<变量号>
CC<变量号></td><td>数据变量，CASE内容为具体数据</td></tr>
<tr><td>GS<变量号></td><td>字符变量，CASE内容为具体字符</td></tr>
<tr><td>附加项2</td><td><编号></td><td>指令结构标号，一个完整的指令结构内该标号必须一致，否则程序将报错或跳出。
同程序中指令结构标号可以相同。
标号范围：0-8。</td></tr>
</table>

程序举例：

```
SWITCH TC#(2) 0      ；SWITCH计算变量TC2，CASE10即TC2=10执行输出A1口1V电压
CASE 10 0
AOUT AO#(1)=1.000
BREAK 0          ；CASE10结束转到ENDSWITCH
CASE 20 0          ；CASE20即TC2=20执行输出A1口2V电压
AOUT AO#(1)=2.000
BREAK 0          ；CASE20结束转到ENDSWITCH
CASE 30 0   CASE30即TC2=30执行输出A1口3V电压
AOUT AO#(1)=3.000
BREAK 0          ；CASE30结束转到ENDSWITCH
CASE 40 0        ；CASE40即TC2=40执行输出A1口4V电压
AOUT AO#(1)=4.000
BREAK 0        ；CASE40结束转到ENDSWITCH
CASE 50 0        ；CASE50即TC2=50执行输出A1口5V电压
AOUT AO#(1)=5.000
BREAK 0          ；CASE50结束转到ENDSWITCH
CASE 60 0        ；CASE60即TC2=60执行输出A1口6V电压
AOUT AO#(1)=6.000
BREAK 0        ；CASE60结束转到ENDSWITCH
CASE 70 0        ；CASE70即TC2=70执行输出A1口7V电压
AOUT AO#(1)=7.000
BREAK 0        ；CASE70结束转到ENDSWITCH
CASE 80 0        ；CASE80即TC2=80执行输出A1口8V电压
AOUT AO#(1)=8.000
BREAK 0       ；CASE80结束转到ENDSWITCH
CASE 90 0      ；CASE90即TC2=90执行输出A1口9V电压
AOUT AO#(1)=9.000
BREAK 0        ；CASE90结束转到ENDSWITCH
DEFAULT 0    ；TC2的结果不等于以上数值，则执行输出A1口10V电压
AOUT AO#(1)=10.000
ENDSWITCH 0     ；SWITCH指令结束
```

7.6.6 条件选择(SWITCH →CASE)

2. CASE	指令功能	SWITCH条件选择指令，本指令由SWITCH、CASE（可重复使用）、BREAK、DEFAULT、ENDSWITCH构成一个完整的指令结构。 计算SWITCH后变量的值，判断和那个CASE后的数值相等。 找到相等CASE后，程序从该位置开始执行，执行到第一个BREAK结束，并跳转到ENDSWITCH行。	
	附加项	<数据>	SWITCH后变量的可能值。
		<编号>	指令结构标号，一个完整的指令结构内该标号必须一致，否则程序将报错或跳出。 同程序中指令结构标号可以相同。 标号范围：0-8。

程序举例：

```
SWITCH TC#(1) 0      ；SWITCH计算变量TC1
CASE 1 0
CASE 2 0
CASE 3 0
CASE 4 0
CASE 5 0
DOUT Y#(18)==ON    ；TC1=1/2/3/4/5都执行输出Y18口有效（Y18=ON）
BREAK 0      ；CASE1-5结束转到ENDSWITCH
DEFAULT 0    ；；TC1不等于以上数据则，输出Y18关闭（Y18=OFF）
DOUT Y#(18)==OFF
ENDSWITCH 0   ；SWITCH指令结束
```

7.6.7 条件选择(SWITCH →DEFAULT)

条件选择(SWITCH →DEFAULT)	指令功能	WITCH条件选择指令，本指令由SWITCH、CASE（可重复使用）、BREAK、DEFAULT、ENDSWITCH构成一个完整的指令结构。 如果计算变量的值都不等于CASE后数值，程序则执行DEFAULT行程序。	
	附加项	<编号>	指令结构标号，一个完整的指令结构内该标号必须一致，否则程序将报错或跳出。 同程序中指令结构标号可以相同。 标号范围：0-8。

程序举例：

SWITCH <变量><编号>

CASE <数值><编号>

……

BREAK <编号>

DEFAULT <编号>

……

ENDSWITCH <编号>

7.6.9 条件选择(SWITCH →ENDSWITCH)

条件选择(SWITCH →ENDSWITCH)	指令功能	SWITCH条件选择指令，本指令由SWITCH、CASE（可重复使用）、BREAK、DEFAULT、ENDSWITCH构成一个完整的指令结构。 SWITCH指令结束标志，表示SWITCH指令结束。	
	附加项	<编号>	指令结构标号，一个完整的指令结构内该标号必须一致，否则程序将报错或跳出。 同程序中指令结构标号可以相同。 标号范围：0-8。

程序举例：

SWITCH <变量><编号>

CASE <数值><编号>

……

BREAK <编号>

DEFAULT <编号>

……

ENDSWITCH <编号>

7.6.8 轴脉冲跳转（PULSETRANSFORM）

轴脉冲跳转（PULSETRANSFORM）	指令功能	本指令由PULSETRANSFORM和PULSETRANSRORMEND构成一个完整指令结构。 本指令将输出轴的脉冲转移到输入轴上，即原来发脉冲给输出轴运动；执行本指令后，脉冲输入到输入轴，输入轴按照所输入脉冲运动，输出轴将停止运动，输入轴原有脉冲也无效。 本指令对关节坐标有效。	
	附加项	<输出轴号>	输出轴为需要被跳转的轴号，输入轴为跳转到的轴号，范围：1-8。
		<转入轴号>	

程序举例：

PULSETRANSFORM#(1)(2)

……

PULSETRANSFORMEND

7.6.9 轴脉冲跳转结束(PULSETARANSFORMEND)

轴脉冲跳转结束 (PULSETARANSFORMEND)	指令功能	本指令由PULSETRANSFORM和PULSETRANSRORMEND构成一个完整指令结构。 本指令将输出轴的脉冲转移到输入轴上，即原来发脉冲给输出轴运动；执行本指令后，脉冲输入到输入轴，输入轴按照所输入脉冲运动，输出轴将停止运动，输入轴原有脉冲也无效。 本指令对关节坐标有效。	
	附加项	无	
	程序举例	PULSETRANSFORM#(1)(2) …… PULSETRANSFORMEND	

7.6.10 定位完成(MOTIONFINISH)

定位完成 (MOTIONFINISH)	指令功能	定位完成对MOTIONFINISH指令前一行移动（MOVJ、MOVL、MOVC）程序有效。 工作过程：前一行移动程序将准确移动到程序设定位置，这个过程中，系统将走完该行程序缓冲区所有数据，然后比较电机编码器反馈数据，直到电机反馈位置与程序行设定位置一致（通俗讲：就是系统脉冲要发完，电机也要停止到位，误差要在允许范围），MOTIONFINISH指令才结束。否则将等待在该指令行，当超过该指令内部设定时间后，系统将提示：定位完成超时。	
	附加项	无	
	程序举例	MOVL VJ=10% PL=5 ；准确移动到该行指定位置 MITIONFINISH ；运行定位完成指令，对上一行移动程序有效。	

7.6.11 轴禁止、轴禁止解除

轴禁止 （FORBIDAXIS） 轴禁止解除 （FORBIDAXIS END）	指令功能	FROBIDAXIS和FORBIDAXISEND两条指令需成对使用。 禁止设定轴移动和解除设定轴禁止移动状态。 用于需要某些需要禁止某轴移动的场合，禁止移动结束后需要解除该状态。 本指令对关节坐标有效。	
	附加项	<轴号>	设定需要禁止移动或解除禁止移动的轴号。
	程序举例	FORBIDAXIS#(1) ；禁止J1轴移动 …… ；中间程序 FORBIDAXISEND# (1) ；解除J1轴禁止状态	

7.6.12 坐标计算（COUNTCOORD）

坐标计算 （COUNTCOORD）	指令功能	执行该指令，系统读取当前各轴编码器数据，然后计算到当前坐标中。	
	附加项	[空白]	
	程序举例	COUNTCOORD ；读取所有轴编码器数据，计算坐标。	

7.6.13 冲压状态开始（JUDGESTART)、冲压状态结束（JUDGEEND）

冲压状态开始 （JUDGESTART） 冲压状态结束 （JUDGEEND）	指令功能	冲压状态开始。 冲压状态结束。			
	附加项1	X<变量号> M<变量号> Y<变量号>	==	OFF/ON	JUDGESTART条件判断，条件成立，执行该语句之后程序。
		<状态值>	stop=值（0和1两个选项，0代表不停机，1代表停机）		
		<报警号>	报警内容为报警号JD001的内容		
		<编号>	指标号范围：0-8。		
	程序举例	JUDGESTART X0==OFF stop1 JD001 0 …… JUDGEEND ；如果X0=OFF，则系统报警，并停机。报警内容为报警号JD001的内容； 掉料报警可添加多段；标号用以区分开始与结束的组别。默认标号为0。			

7.6.14 超时报警（OTALAR）

超时报警（OTALAR）	指令功能	执行该指令，系统后台检测相应的输入信号，条件不满足则报警			
	附加项	X<变量号> M<变量号> Y<变量号>	==	OFF/ON	执行超时报警指令，后台检测X，M，Y的信号 。
		<时间>	等待时间		
		<报警号>	弹框显示报警号		
	程序举例	OTALAR X#(0)==OFF T=3000 OT008 ；后台检测X0的信号，等待1000ms，X0==OFF还不满足，则报警。			

7.7 视觉指令

7.7.1 视觉运行(RUNVISION)

视觉运行（RUN VISION）	指令功能	视觉运行指令，系统将调用指令参数中设定的工艺号相关参数，然后与系统建立连接关系。 当工艺文件选择定时或定距触发时，运行本指令后，视觉系统将开始按照设定连续触发。获得数据存入系统。 连接过程在程序连续运行时，将保持，直到程序完成退出时才断开。即：一个程序有多个本指令时，只生效第一个，一个程序连续运行时，第一次执行本指令有效，后续执行无效。	
	附加项	#<工艺号>	视觉文件号码，范围：0-9。一个工艺文件号对应一套视觉系统。
	程序举例	RUNVISION#(0) ；以0号工艺文件参数建立系统与视觉之间的连接。	

7.7.2 获得视觉数据（GETVISIONDATA）

获得视觉数据（GET ISIONDATA）	指令功能	将视觉缓冲区内数据导入到GP52和GP53中（GP52和GP53数值相同）。	
	附加项	<工艺号>	视觉文件号码，范围：0-9。一个工艺文件号对应一套视觉系统。
	程序举例	GETVISONDATA #(0) 将数据导入GP52和GP53中。	

7.7.3 清除视觉数据（CLEARVISONDATA）

清除视觉数据 （CLEAR VISONDATA）	指令功能	清除当前缓存区内所有视觉数据。	
	附加项	<工艺号>	视觉文件号码，范围：0-9。一个工艺文件号对应一套视觉系统。
	程序举例	CLEARVISONDATA#(0) ；清除缓存区内，0号工艺号对应相机所获得的工件数据。	

7.7.4 视觉指令触发（TRIGGERVISON）

视觉指令触发（TRIGGER VISON）	指令功能	当指令对应工艺号文件中选择指令触发时，使用本指令将触发视觉系统工作，计算工件数据，并将得到的数据存入视觉缓冲区中，将数据个数存入存入GI50中。 为了减少延迟，本指令直接通过输出口（Y）触发视觉系统。	
	附加项	<工艺号>	视觉文件号码，范围：0-9。一个工艺文件号对应一套视觉系统。
	程序举例	TRIGGERVISON#(0) ；触发0号工艺视觉系统工作。	

7.7.5 相机程序切换（CHANGE VISON）

相机程序切换 （CHANGE VISON）	指令功能	执行本指令，将通过与视觉系统连接的数据线，切换视觉系统调用的工件计算参数文件。	
	附加项	<文件号>	需要切换到的文件名，本系统只支持数字文件名，所以视觉系统中文件也需要使用数字命名。
	程序举例	A工件对应工件计算参数文件为111，B工件对应工件计算参数文件为222。当前视觉系统运行A工件文件。 CHANGVISON#(222) 视觉系统将调用B工件（222）计算参数文件。	

7.7.6 等待触发结果（WAITTRIGGER）

等待触发结果（WAITTRIGGER）	指令功能	执行本指令，必须收到数据后才会往下执行，否则，在此指令一直等待结果。	
	附加项	无	无
	程序举例	WAITTRIGGER ；等待触发结果，没有收到数据，一直在此等待。	

视觉综合实例：

RUNVISON#(0) ；运行视觉0号工艺。

*22 ；跳转标号：*22。

TRIGGERVISON#(0) ；触发一次，获得相机计算工件数据。

TIME T= 200 ；延时200毫秒

JUMP *22 IF GI#(50)<=0.000 ；判断GI50是否有数据，没有跳转到标号*22，重新触发。

GETVISONDATA#(0) ；得到视觉缓冲区的数据，放在GP52和GP53中。

ADD GP#53(3) 10.000 ；GP53的Z方向上提10MM，防止装到物体。

MOVL VL=500MM/S GP#53 PL=0 ；运行到GP53安全点。

MOVL VL=100MM/S GP#52 PL=0 ；运行到GP52抓取点。

DOUT Y#(1)=ON ；抓取

MOVL VL=500MM/S GP#53 PL=0 ；运行到GP53安全点。

…… 后续放物体轨迹、动作。

7.8 跟踪指令

7.8.1 跟踪开始(TARCKSTART)

跟踪开始（TARCK START）	指令功能	系统调用工艺跟踪文件参数，将缓存区当前物体位置数据放到GP50和GP51中（GP50和GP51相同），此时GP50和GP51数据均为适时动态的。直接调用GP50和GP51数据即可开始跟踪。	
	附加项	#<工艺号>	跟踪文件号，范围：0-9。
		<阻塞/不阻塞>	阻塞，一直到物体过A点得到数据程序再往下执行。 不阻塞，不用等到数据，直接往下执行。
	程序举例	TRACKSTART#(0) 阻塞 ；等物体过A点得到准备GP50和GP51数据。	

7.8.2 跟踪结束(TARCK END)

跟踪结束 (TARCK END)	指令功能	追踪结束，机器人停止追踪工件。GP50和GP51数据不再适时。	
	附加项	<工艺号>	跟踪文件号。范围：0-9。
	程序举例	TRACKEND#(0)　停止工件追踪。	

7.8.3 获得跟踪数据(GETTRACKDATA)

获得跟踪数据 (GET TRACKDATA)	指令功能	调用跟踪文件号设定参数，判断跟踪缓存区的数据。如果没有数据，等待。工件没有进入AB范围，等待。工件超出B点放弃这组数据，等待。	
	附加项	<工艺号>	跟踪文件号。范围：0-9。
		X<数据> Y<数据> Z<数据>	各轴补偿数据，如果不需要补偿，则设置为0。
	程序举例	GETRTACKDATA#(0) X#(0.0) Y#(0.0) Z#(0.0) ；判断缓存区数据是否有效。XYZ轴补偿为0。	

7.8.3 清除跟踪数据(CLEAR STACK DATA)

清除跟踪数据 (CLEAR STACK DATA)	指令功能	清除工艺文件对应跟踪缓存区数据。在某些需要清除缓存区的场合使用。	
	附加项	<工艺号>	跟踪文件号。范围：0-9。
		X<数据> Y<数据> Z<数据>	各轴补偿数据，如果不需要补偿，则设置为0。
	程序举例	CLEARTRACKDATA#(0) ；清除0号跟踪文件对应缓存区数据。	

7.8.4 后台IO检测(RUN IO CUTIN)

后台IO检测 (RUN IO CUTIN)	指令功能	运行本指令，IO检测点将持续不停的检测，是否有物体通过该检测点。如有物体通过，则将当前数据存入缓存区，以备调用。	
	附加项	<工艺号>	跟踪文件号。范围：0-9。
		X<数据> Y<数据> Z<数据>	各轴补偿数据，如果不需要补偿，则设置为0。
	程序举例	RUNIOCUTIN#(0) ；后台IO启动，监测点持续检测。	

跟踪综合实例：

MOVL VL=500.00MM\S PL=0 ；运行到中间点上方，等待抓取。

RUNIOCUTIN #(1) ；运行后台IO检测，如果有物体经过IO检测点，会把当前位置压入缓存区。

GETTRACKDATA#(1) X#(0.0) Y#(0.0) Z#(0.0) ；判断跟踪缓存区的数据，如果没有数据，等待；数据没有进入AB范围，等待；数据超出B点，放弃这组数据，等待。

TREACKSTART #(1) ；跟踪开始，将缓存区中当前工件数据放在GP50和GP51里面。

ADD GP#51(3) 10.000 ；GP51的Z方向增加10mm，将GP51作为安全位置。(3)含义参考GP变量单轴数据操作。

MOVL VL=500MM/S GP#51 PL=0 ；移动到GP51安全点。

MOVL VL=100MM/S GP#50 PL=0 ；移动到GP50跟踪点。

DOUT Y#(0) =ON ；输出Y0有效，抓取物体。

MOVL VL=500MM/S GP#51 PL=0 ；提起物体，到GP51安全点。

TRACKEND #(1)；跟踪结束。

…… 后续操作步骤。

详见《跟踪工艺说明书》。

7.9 通讯指令

7.9.1 读取IO口BCD数据(READIOBCD)

读取IO口BCD数据（READIOBCD）	指令功能	读取X输入口状态，输入到设定的变量中。	
	附加项	GI<变量> X<端口> B<个数>	数据存储变量。 读取X输入口的起始位置。 读取X口的个数
	程序举例	READIOBCD #(20) #(0) #(4) ；读取X0-X3口(4个接口)状态数据，储存到GI20变量中。	

7.9.2 读取数据到 (READDATATO)

读取数据到 (READ DATA TO)	指令功能	通过串口通讯数据到后面的变量中，通讯串口号和通讯功能需在操作参数中设置。 本功能主要用于读取条码，代码等。	
	附加项	GI<变量号> GD<变量号> GS<变量号>	通讯数据到GI变量。 通讯数据到GD变量。 通讯数据到GS变量。
	程序举例	READDATATO GI#(20) 读取串口数据到变量GI20中。	

7.9.3 把10进制输出到IO (SETBCDIO)

把10进制输出到IO (SETBCDIO)	指令功能	通过串口通讯数据到后面的变量中，通讯串口号和通讯功能需在操作参数中设置。 本功能主要用于读取条码，代码等。	
	附加项	GI<变量号>	变量号，将该变量中的十进制数据转二 进制通讯数据到GD变量。
		Y<变量号>	输出起始口号，从该接口开始输出。为二进制的低位。
		B<变量号>	接口数，输出多少个接口
	程序举例	SETBCDIO GI2 Y#15 B#4 ；输出到Y15开始的4个接口Y15-Y18。8转二进制为1000，则Y18-Y15 为1 0 0 0 。Y18输出有效。Y17 Y16 Y15 输出无效。	

7.9.4 SENDDATATO

SENDDATATO	指令功能	备用
	附加项	无
	程序举例	无

7.10 特殊指令

7.10.1 程序复位 (RESET)

程序复位 (RESET)	指令功能	执行本指令，系统将逐级退回到程序的首行位置停止。 就算当前运行在子程序中，使用本指令后，系统也将退出子程序，并返回到主程序首行停止。
	附加项	[无]
	程序举例	1 …… ；主程序XX内容 8 CALL 123 ；调用123子程序 9 …… ；子程序结束后执行内容 --------- 1 …… ；子程序123内容 4 RESET ； 程序复位 5 RET ；子程序返回
		上述程序组合中，执行到子程序123中的第4行RESET指令时，系统将直接跳转到主程序XX的第一行停止。 如果子程序123中，没有RESET指令，执行到子程序的RET指令时，系统将跳转到主程序的第9行继续执行。

7.10.2 程序指针跳转 (CHANGEP)

程序指针跳转 (CHANGEP)	指令功能	移动自动运行时行数指针。 如：自动运行时停止在第18行。使用CHANGEP#（-2）指令，程序停止行位置将跳转到18-2=16行位置。	
	附加项	<行数>	需要跳转行数，正数向后；负数向前跳转。
	程序举例	CHANGEP#(-2) ；停止行数向后跳转两行，如停止在18行，执行本指令后将跳转到18-2=16行。	

7.10.3 寻位开始(SEARCHSTART)

寻位开始 (SEARCHSTART)	指令功能	寻位开始，调用1号寻位工艺 寻位结束。	
	附加项	<文件号>	需要调用的寻位文件号，寻位工艺参数中设置； 寻位完成后关闭寻位。
	程序举例	SEARCHSTART#(1) ；调用寻位1号工艺 SEARVCHEND ；寻位结束	

7.10.4 偏移开始、结束指令

偏移开始 (OFFSETSTART) 偏移结束 （OFFSETEND）	指令功能	偏移开始，调用偏移量OP 偏移结束	
	附加项	<文件号>	需要调用的偏移变量OP,由旗标点和寻位位置计算生成； 偏移结束。
	程序举例	OFFSETSTART OP#(1) ；偏移开始，偏移量为OP（1）变量 OFFSETEND ；偏移结束	

7.10.5 计算偏移量（COUNTOFFSET）

计算偏移量 COUNT OFFSET	指令功能	移动自动运行时行数指针。如：自动运行时停止在第18行。使用CHANGEPOINT#（-2）指令，程序停止行位置将跳转到18-2=16行位置。																	
	附加项	角焊缝	1D	参考	[空白] OP#[ ]	保存	OP#[ ]	点	NP#[ ]										
			2D	参考	[空白] OP#[ ]	保存	OP#[ ]	点	NP#[ ]	点	NP#[ ]								
			3D	参考	[空白] OP#[ ]	保存	OP#[ ]	点	NP#[ ]	点	NP#[ ]	点	NP#[ ]						
			2D+	参考	[空白] OP#[ ]	保存	OP#[ ]	线	NP#[ ]	NP#[ ]	点	NP#[ ]	[空白] USE		[变量号]				
			3D+	参考	[空白] OP#[ ]	保存	OP#[ ]	面	NP#[ ]	NP#[ ]	NP#[ ]	线	NP#[ ]	NP#[ ]	点	NP#[ ]			

<table>
<tr><td rowspan="5">计算偏移量

COUNT

OFFSET</td><td rowspan="4">附加项</td><td>内外径</td><td>2D</td><td>参考</td><td>[空白]
OP#[ ]</td><td>保存</td><td>OP#[ ]</td><td>点</td><td>OP#[ ]</td><td>点</td><td>OP#[ ]</td><td>点</td><td>OP#[ ]</td><td>USE</td><td>[变量号]</td></tr>
<tr><td>点</td><td>参考</td><td>OP#[ ]</td><td>保存</td><td>OP#[ ]</td><td>基准</td><td>GP#[ ]</td><td>参考</td><td colspan="7">GP#[ ]</td></tr>
<tr><td>相机</td><td>参考</td><td>OP#[ ]</td><td>保存</td><td>OP#[ ]</td><td>基准</td><td>GP#[ ]</td><td>参考</td><td colspan="7">GP#[ ]</td></tr>
<tr><td>面</td><td>参考</td><td>OP#[ ]</td><td>保存</td><td>OP#[ ]</td><td>基准</td><td>GP#[ ]</td><td>参考</td><td colspan="7">GP#[ ]</td></tr>
<tr><td>程序举例</td><td colspan="14">COUNTOFFSET 角焊缝 2D+ 参考OP#(1) 保存OP#(2)
线：NP#(3) NP#(4) 点：NP#(5) USE#(0);

角焊缝2D+旋转，在用户坐标系#（0）中，两个点NP3和NP4确定一条线，加上一个点NP5确定一个面，参考OP(1),计算出变量保存到OP(2)中。
详细使用方法请参考《CRP寻位功能说明书》</td></tr>
</table>

寻位综合实例：

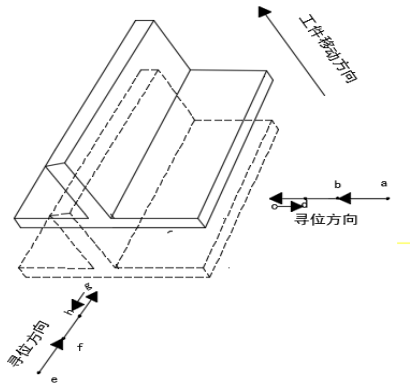


图 7.5

如图上工件的寻位过程：

SEACHSTART 0　　；寻位开始

MOVL VL=100 PL=0 TOOL=1　　；a点

MOVL VL=100 PL=0 TOOL=1 SEACH NP0　　；b点寻c点，自动退d点，数据存NP0

MOVL VL=100 PL=0 TOOL=1　　；e点

MOVL VL=100 PL=0 TOOL=1 SEACH NP1　　；f点寻g点，自动退h点，数据存NP1

SEACHEND　　；寻位结束

COUNTOFFSET 角焊缝 2D NP0 NP1 OP1 ；数据计算。放入OP1中。

OFFSETSTART OP[1] ；开始偏移OP1

MOVL VL=100 PL=0 TOOL=1 ；偏移数据部分程序

……

OFFSETEND ；偏移结束

7.10.6 计时开始、计时结束指令

计时开始 （STARTTIME） 计时结束 （ENDTIME）	指令功能	计时开始。 计时结束。
	附加项	无
	程序举例	STARTTIME计时开始 …… ENDTIME计时结束

7.10.7 多圈轨迹开始、多圈轨迹结束指令

多圈轨迹开始 （MULTITURNSTART） 多圈轨迹结束 （MULTITURNEND）	指令功能	备用
	附加项	无
	程序举例	

7.10.8 计算负载率指令（COUNTLOADFAC）

计算负载率指令 （COUNTLOADFAC）	指令功能	执行指令，系统会在程序停止时进行负载率计算，并将结果放在对应的寄存器中
	附加项	无
	程序举例	COUNTLOADFAC ；将计算得出的各轴的负载率取整分别写在GI920- GI927中

7.10.9 计算静态转矩百分比指令（COUNTSTORQUE）

计算静态转矩百分比指令 （COUNTSTORQUE）	指令功能	执行指令，系统会在程序停止时进行静态转矩百分比计算，并将结果放在对应的寄存器中
	附加项	无
	程序举例	COUNTSTORQUE ；将计算得出的各轴的静态转矩百分比分别GI930-GI937中

7.10.10 计算动态数据指令（COUNTDYNASTH）

计算动态数据指令（COUNTDYNASTH）	指令功能	执行指令，系统会在程序停止时进行动态数据计算，并将结果放在对应的寄存器中
	附加项	无
	程序举例	COUNTDYNASTH ；将计算得出的各轴的静态转矩百分比分别GI94-GI101中

7.10.11 缩放开始、缩放结束指令

缩放开始（ZOOMSTART） 缩放结束 （ZOOMEND）	指令功能	缩放开始，调用1号缩放工艺。 缩放结束。	
	附加项	<工艺号>	缩放文件号，范围：0-99。
	程序举例	ZOOMSTART（1） ；缩放开始，调用（1）号工艺 …… ZOOMEND ；缩放结束	

7.10.12 焊枪角度显示开始、焊枪角度显示结束

焊枪角度显示开始（ANGLEDISPLAY） 焊枪角度显示复位 （ANGLEDISPLAYRST）	指令功能	执行角度显示指令，才会显示此指令后的直线或者圆弧轨迹时的工具角度。	
	附加项	GP<变量号> <数据号>	LP<变量号> <数据号>
	程序举例	ANGLEDISPLAY LP0 ；焊枪角度显示开始， 显示数据存放在LP0 …… ANGLEDISPLAYRST ；焊枪角度显示复位	

7.10.13 开始塔角计算、结束塔角计算

开始塔角计算（TOWERCOUNTSTART） 结束塔角计算（TOWERCOUNTEND）	指令功能	运行该指令，计算当前焊缝在船型状态时，外部轴的角度值，使后面运动程序外部轴强制到达计算出的角度。	
	附加项	<焊缝工艺号>0-99	<摆弧工艺号>1-4
	程序举例	ANGLEDISPLAY LP0 ；焊枪角度显示开始，显示数据存放在LP0 …… ANGLEDISPLAYRST ；焊枪角度显示复位	

7.10.14 塔角焊缝计算（TOWERCHANGEWELD）

塔角焊缝计算 （TOWERCHANGEWELD）	指令功能	运行该指令，计算当前焊缝在船型状态时，外部轴的角度值，使后面运动程序外部轴强制到达计算出的角度。	
	附加项	<焊缝工艺号>0-99	<摆弧工艺号>1-4
	程序举例	TOWERCHANGEWELD（4）（1） ；塔角焊缝计算， 调用工艺号4和摆弧工艺1 …… TOWERCOUNTEND ；结束塔角计算	

7.10.15 塔角姿态计算开始、塔角姿态计算结束

塔角姿态计算开始 （TOWERCOUNTPOSESTART） 塔角姿态计算结束 （TOWERCOUNTPOSEEND）	指令功能	塔角姿态计算开始，调用1号姿态工艺。 塔角姿态计算结束。	
	附加项	<工艺号>	塔角姿态工艺号，范围：1-99。
	程序举例	TOWERCOUNTPOSESTART（1） ；开始塔角姿态计算，调用工艺号1 …… TOWERCOUNTPOSEEND ；结束塔角姿态计算	

7.10.16 角钢轨迹计算（COUNTSTEELANGLE）

角钢轨迹计算 （COUNTSTEELANGLE）	指令功能	角钢轨迹计算，调用1号轨迹工艺。	
	附加项	<工艺号>	角钢轨迹工艺号，范围：1-99。
	程序举例	TCOUNTSTEELANGLE（1） ；角钢轨迹计算，调用角钢轨迹工艺号1 ……	

7.11 折弯指令

7.11.1 折弯跟随 (BENDTRACK)

折弯跟随 (BENDTRACK)	指令功能	折弯工艺中，调用折弯工艺的指令	
	附加项	<工艺号>	<平滑度>(平滑越大跟随越慢，机器人越抖)
	程序举例	BENDTRACK#(0) (13)　　；开始跟随，调用折弯工艺（0），平滑度为13	

7.11.2 折弯同步(BENDSYS)

折弯同步(BENDSYS)	指令功能	折弯工艺中，用于同步回程的工艺调用。	
	附加项	<工艺号>	UP/DOWN（同步回程（上、下））
	程序举例	TCOUNTSTEELANGLE（1）　；角钢轨迹计算，调用角钢轨迹工艺号1 ……	

7.11.3 折弯回平(BENDFLATBACK)

折弯回平 (BENDFLATBACK)	指令功能	折弯工艺中，在检测到回程到位信号后，机器人回平姿态的指令。	
	附加项	<工艺号>	缩放文件号，范围：0-999。
	程序举例	BENDFLATBACK#(0)　　；检测到回程到位信号，机器人回平姿态。	

折弯综合实例：

```
MOVL VL=100  PL=0  TOOL=1          ；到折弯点
DOUT Y#(7)=ON     ；启动折弯机
BENDTRACK#(0) (13)        ；检测到加压信号开始跟随
BENDSYS#(0) UP      ；检测到回程信号开始向上跟随
BENDFLATBACK#(0)          ；检测到回程信号到位，机器人回平姿态
TIME T=1000       ；延时1S
```

```
DOUT Y#(7)=OFF    ；关闭折弯机
WAIT M#(232)==ON DT=0 CT=500    ；等待折弯机回程到位信号，并持续500ms
MOVL VL=200MM/S PL=0 TOOL=2    ；到折弯点
```

7.12 其他

7.12.1 碰撞等级设置 (SHCKSET)

碰撞等级设置 (SHCKSET)	指令功能	设置碰撞检测等级	
	附加项	<预设值号>	范围1-7
	程序举例	SHCKSET#(1)　；切换碰撞等级预设号1，遇见SHCKRET指令（重启）解除，切换回默认值8号。	

7.12.2 碰撞等级解除 (SHCKRET)

碰撞等级解除 (SHCKRET)	指令功能	碰撞检测等级预设号解除	
	附加项	无	
	程序举例	SHCKACT　；解除前面碰撞等级设置指令设置的预设号（没有通过指令设置预设号，使用此指令也不影响，使用系统默认预设号8）。使用再现远程默认设置8号。	

7.12.3 碰撞检测激活 (SHCKACT)

碰撞检测激活 (SHCKACT)	指令功能	临时激活碰撞检测功能，切换示教模式，关机重启、执行碰撞检测禁用指令失效。	
	附加项	无	
	程序举例	SHCKACT　；临时作用，此 指令不会改变碰撞检测预设文件中设置的是否启用碰撞检测的设置	

7.12.4 碰撞检测禁用指令(SHCKDEACT)

碰撞检测禁用指令(SHCKDEACT)	指令功能	临时激活碰撞检测功能，切换示教模式，关机重启、执行碰撞检测禁用指令失效。	
	附加项	无	
	程序举例	SHCKDEACT ；临时作用， 此指令不会改变碰撞检测预设文件中设置的是否启用碰撞检测的设置。主要用于某些场景需要的力较大，会 报警碰撞，临时关闭，执行操作后开启	

碰撞等级实例：

CHANGETOOL#(1) ；切换1号工具 必须正确使用工具号，1号为抓取时的带负载工具

MOVJ VJ=100% PL=0 TOOL=1 ；关节运动到等待点

MOVL VL=1000MM/S PL=9 TOOL=1 ；中间过程路径点位

MOVL VL=1000MM/S PL=9 TOOL=1

MOVL VL=200MM/S PL=0 TOOL=1 ；准备进入狭窄空间（机床）

SHCKSET#(1) ；切换1号碰撞等级预设号 因为速度慢，可以将灵敏度加高，这样更容易报警，保护机械。

MOVL VL=50MM/S PL=0 TOOL=1 ；慢速移动靠近

DOUT Y#(0)=ON ；放置工件

TIME T=50

CHANGETOOL#(2) ；切换工具2号，2号为放下工件后的负载数据

MOVL VL=50MM/S PL=0 TOOL=2 ；退出

SHCKRET ；解除2号碰撞等级预设值，使用默认值

碰撞检测激活 (SHCKACT)与碰撞检测禁用指令(SHCKDEACT)：

MOVL VL=200MM/S PL=0 TOOL=1 ；必须正确使用工具号，1号为抓取时的带负载工具

SHCKDEACT ；碰撞检测禁用指令，工具重量过大

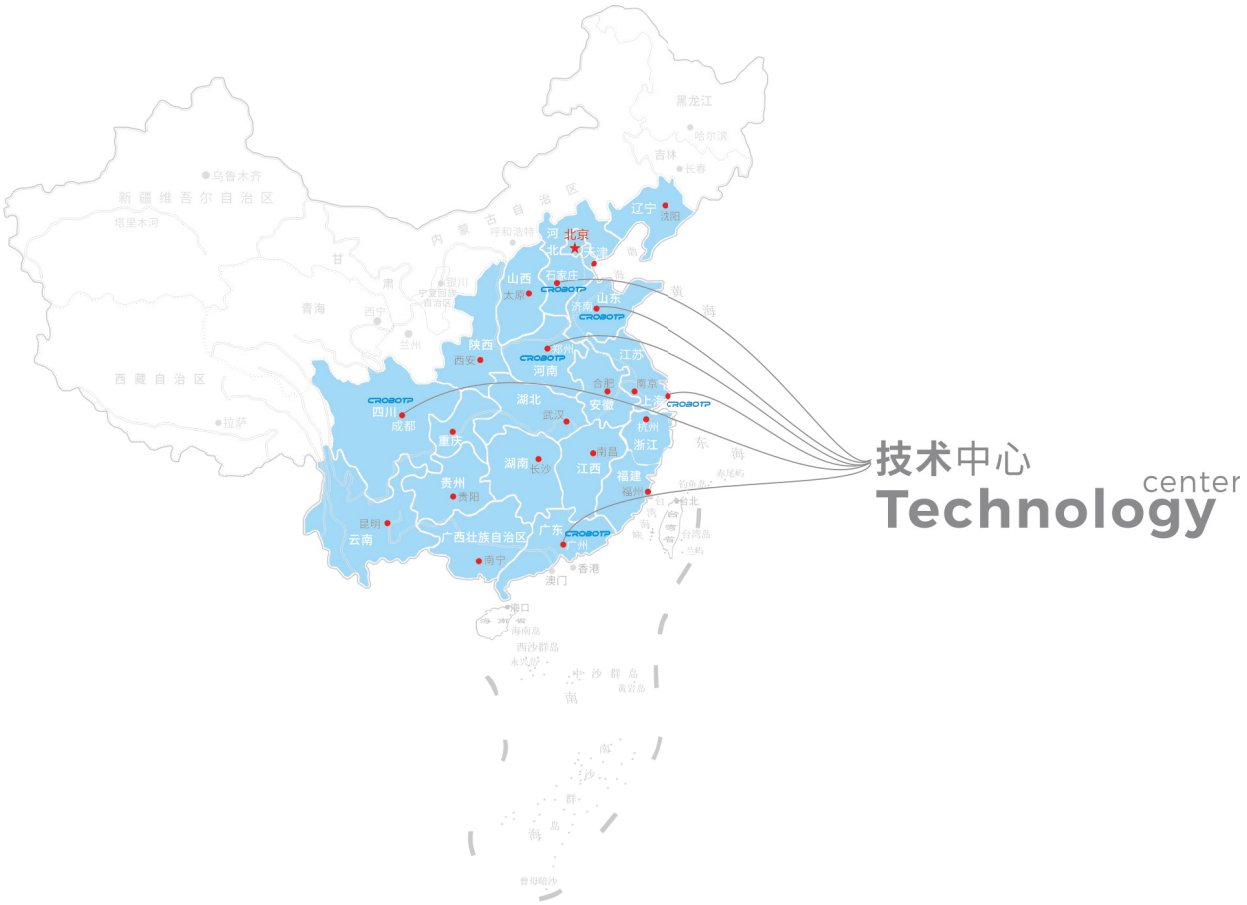
MOVL VL=1000MM/S PL=9 TOOL=1 ；中间过程路径点位

MOVL VL=1000MM/S PL=9 TOOL=1

DOUT Y#(0)=ON ；放工件

SHCKACT ；碰撞检测激活，放完工具后负载小

MOVL VL=50MM/S PL=0 TOOL=1 ；运动指令



微信公众号　抖音号　资料下载

成都卡诺普机器人技术股份有限公司
CHENGDU CRP ROBOT TECHNOLOGY CO.,LTD

86）028-84203568
crobotp@crprobot.com
www.crprobot.com
四川成都市成华区华泰路40号