ABB 机器人 RAPID 常用指令详解-中文

1.88.MoveAbsJ—把机器人移动到绝对轴位置

用途：

MoveAbsJ（绝对关节移动）用来把机器人或者外部轴移动到一个绝对位置，该位置在轴定位中定义。

使用实例：

- 终点是一个单一点
- 对于 IR6400C 中的不明确的位置，例如携带超过机器人范围的工具运动。

MoveAbsJ 指令中机器人的最终位置，既不受工具或者工作对象的影响，也不受激活程序更换的影响。但是机器人要用到这些数据来计算负载、TCP 速度和转角点。相同的工具可以被用在相邻的运动指令中。

机器人和外部轴沿着一个非直线的路径移动到目标位置。所有轴在同一时间运动到目标位置。

该指令只能被用在主任务 T_ROB1 中，或者在多运动系统中的运动任务中。

基本范例：

该指令的基本范例说明如下。

也可参看第 207 页更多范例。

例1　　MoveAbsJ p50, v1000, z50, tool2;

机器人将携带工具 tool2 沿着一个非线性路径到绝对轴位置 p50，以速度数据 v1000 和 zone 数据 z50。

例2　　MoveAbsJ *, v1000\T:=5, fine, grip3;

机器人将携带工具 grip3 沿着一个非线性路径到一个停止点，该停止点在指令中作为一个绝对轴位置存储（用*标示）。整个运动需要 5 秒钟。

项目：

MoveAbsJ [\Conc] ToJointPos [\ID] [\NoEOffs] Speed [\V] | [\T] Zone [\Z] [\Inpos] Tool [\Wobj]

[\Conc]:

并发事件

数据类型：switch

当机器人正在移动的时候执行的后续指令。该项目通常不使用，但是当和外部设备通讯、不需要同步的时候可以用来缩短循环周期。

当使用项目\Conc 的时候，连续运动指令的数量限制为 5。在包含 StorePath-RestoPath 的程序段中不允许包含项目\Conc 的运动指令。

如果该项目忽略并且 ToJointPos 不是一个停止点，在机器人到达程序 zone 之前一段时间后续指令就开始执行了。

该项目不能用在多运动系统的坐标同步运动中。

ToJointPos：

到达的关节位置。

数据类型：jointtarget

机器人和外部轴的绝对目标轴位置。它被定义为一个命名的位置或者直接存储在指令中（在指令中用*标示）。

[\ID]：

同步 ID

数据类型：identno

该项目必须使用在多运动系统中，如果并列了同步运动，则不允许在其他任何情况下使用。

指定的 ID 号在所有协同的程序任务中必须相同。该 ID 号保证在 routine 中运动不会混乱。

[\NoEOffs]：

没有外部偏移量

数据类型：switch

如果项目\NoEOffs 设为 1，MoveAbsJ 运动将不受外部轴的激活偏移量的影响。

Speed：

数据类型：speeddata

运动所用的速度数据。速度数据定义了 TCP、工具再定位和外部轴的速度。

[\V]：

速度

数据类型：num

该项目用来在指令中直接指定 TCP 的速度，单位 mm/s，它替代在速度数据中指定的相应的速度。

[\T]：

时间

数据类型：num

该项目用来指定机器人运动的总时间，单位秒。它替代相应的速度数据。

Zone：

数据类型：zonedata

运动的 zone 数据。Zone 数据描述了产生的转角路径的大小。

[\z]：

Zone

数据类型：num

该项目用来在指令中直接指定机器人 TCP 的位置精度。转角路径的长度用毫米给出，替代 zone 数据中指定的相应数据。

[\Inpos]：

到位

数据类型：stoppointdata（停止点数据）

改项目用来指定机器人 TCP 在停止点位置的收敛性判别标准。该停止点数据代替在 zone 参数中指定的 zone。

Tool：

数据类型：tooldata

运动过程中所携带的工具。

TCP 的位置和工具的负载在工具数据中定义。TCP 位置用来计算运动的速度和转角路径。

[\Wobj]：

工作对象

数据类型：wobjdata

在运动过程中使用的工作对象。

如果机器人抓着工具的时候，该项目可以忽略。但是，如果机器人抓着工作对象， 也就是说工具是静止的，或者带有外部轴，那么该项目必须指定。

在有并列工具或者有并列外部轴的情况下，系统使用该数据计算运动的速度和转角路径，该数据在工作对象中定义。

程序执行：

MoveAbsJ 运动不会受激活的程序转移的影响，并且如果使用了可选项目\NoEOffs，将没有外部轴的偏移。如果不使用\NoEOffs，外部轴的目标位置将会受到激活的外部轴偏移的影响。工具按照轴角度插补移动到绝对轴目标位置。这就是说每一个轴都按照固定的速度运动，并且所有轴都在同一时间到达目标位置，这样就形成一个非线性的路径。

总的来说，TCP 大约按照编程的速度运动。在 TCP 运动的同时，工具重新定向，并且外部轴也在运动。如果重新定向的或者外部轴的程序要求的速度不能达到，TCP 的速度将被减小。

当转换到路径的下一段的时候通常会产生转角路径。如果停止点在 Zone 数据中指定，只有在机器人和外部轴到达合适的轴位置的时候程序才能继续执行。

更多范例：

关于如何使用该指令，更多范例说明如下：

例1　　MoveAbsJ *, v2000\V:=2200, z40 \Z:=45, grip3;

Grip3 沿着一个非线性路径运动到一个存储在指令中的一个绝对轴位置。执行的运动数据为 v2000 和 z40。TCP 的速度大小是 2200mm/s，zone 的大小是 45mm。

例2　　MoveAbsJ p5, v2000, fine \Inpos :=inpos50, grip3;

Grip3 沿着一个非线性路径运动到绝对轴位置 p5。当停止点 fine 的 50%的位置条件和 50%的速度条件满足的时候，机器人认为它已经到达位置。它等待条件满足最多等 2 秒。参看 stoppointdata 类型的预定义数据 inpos50。

例3　　MoveAbsJ \Conc, *, v2000, z40, grip3;

Grip3 沿着一个非线性路径运动到一个存储在指令中的一个绝对轴位置。当机器人运动的时候，也执行了并发的逻辑指令。

例4　　MoveAbsJ \Conc, * \NoEOffs, v2000, z40, grip3;

和以上的指令相同的运动，但是它不受外部轴的激活的偏移量的影响。

例5　　GripLoad obj_mass;

MoveAbsJ start, v2000, z40, grip3 \Wobj:=obj;

机器人把和固定工具 grip3 相关的工作对象 obj 沿着一个非线性路径移动到绝对轴位置 start。

限制：

为了能够后台运行中包括指令 MoveAbsJ，并且避免单一点和模糊区的问题，并发指令满足以下的要求是很必要的（参看下图）

下图显示了后台运行 MoveAbsJ 指令的一些限制。

语法：

MoveAbsJ [‘\’ Conc ‘,’] [ToJointPos’ :=’] <关节目标表达式（IN）>

[‘\’ ID ‘:=’ <identno 类型的表达式(IN)>] [‘\’ NoEOffs] ‘,’

[Speed ‘:=’] <speeddata 类型的表达式(IN)>

[‘\’ V ‘:=’ <num 类型的表达式（IN）>]

| [‘\’ T’ :=’ <num 类型的表达式(IN)>] ‘,’

[‘\’ Z ‘:=”] <num 类型的表达式(IN)>

[‘\’ Inpos’ :=’ <stoppointdata 类型的表达式(IN)>] ‘,’

[Tool ‘:=’]<tooldata 类型的恒量(PERS)>

[‘\’Wobj’ :=’ wobjdata 类型的恒量(PRS)>] ‘;’

相关信息：

相关信息	参看
其它定位指令	RAPID 参考手册—RAPID 概述，RAPID 摘要—运动部分
关节目标的定义	第 959 页 Jointtarget—关节位置数据
速度的定义	第 1010 页 speeddata—速度数据
Zone 数据的定义	第 1047 页 zonedata—zone 数据
停止点数据的定义	第 1014 页 stoppointdata—停止点数据
工具的定义	第 1031 页 tooldata—工具数据
工作对象的定义	第 1039 页 wobjdata—工作对象数据
运动综述	RAPID 参考手册—RAPID 概述，运动和 I/O 原理部分
并发的程序执行	RAPID 参考手册—RAPID 概述，运动和 I/O 原理—用逻辑指令同步部分

1．89．MoveC—让机器人做圆周运动

用途：

该指令用来让机器人 TCP 沿圆周运动到一个给定的目标点。在运动过程中，相对圆的方向通常保持不变。

该指令只能在主任务 T_ROB1 中使用，在多运动系统中的运动任务中使用。

基本范例：

该指令的基本范例说明如下：

也可参看第 212 页更多范例。

例 1　　Move p1, p2, v500, z30, tool2;

Tool2 的 TCP 圆周运动到 p2，速度数据位 v500, zone 数据为 z30.圆由开始点、中间点 p1 和目标点 p2 确定。

例 2　　MoveC *, *, v500 \T:=5, fine, grip3;

Grip3 的 TCP 沿圆周运动到存储在指令中的 fine 点（第二个*标记）。中间点也存储在指令中（第一个*标记）。

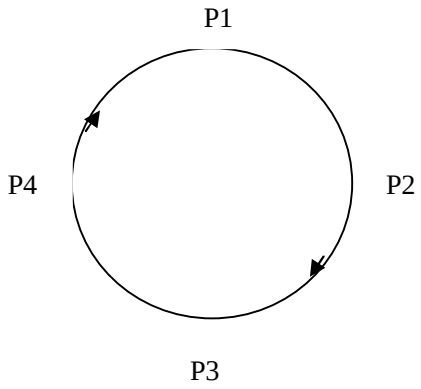
整个运动需要 5 秒钟。

例 3　　MoveL p1, v500, fine, tool1;

MoveC p2, p3, v500, z20, tool1;

MoveC p4, p1, v500, fine, tool1;

下图说明了怎么用两个 MoveC 指令画一个完整的圆。



项目：

MoveC [\Conc] CirPoint ToPoint [\ID] Speed [\V] | [\T] Zone [\z] [\Inpos] Tool [\Wobj] [\Corr]

[\Conc]：

并发事件

数据类型：switch

当机器人运动的同时，后续的指令开始执行。该项目通常不使用，但是当使用飞点（flyby points）时，可以用来避免由 CPU 过载引起的不想要的停止。当使用高速度并且编程点相距较近时这是很有用的。例如，当和外部设备通讯并且外部设备和机器人通讯不要求同步的时候，这个项目也很有用。

使用项目\Conc 的时候，连续的运动指令的数量限制为 5 个。在包括 StorePath—RestorePath 的程序段中不允许使用带有\Conc 项目的运动指令。

如果不使用该项目，并且 ToPoint 不是停止点，在机器人到达程序 zone 之前一段时间后续指令就开始执行了。在多运动系统中的坐标同步运动中不能使用该项目。

CirPoint：

数据类型：robtarget

机器人的圆轴上的中间点。这是圆轴上处于起点和终点之间的点。为了获得最好的精度，最好选择起点和终点的中间位置附近的点。如果太接近起点或者终点，机器人将会报警。中间点定义为一个命名的位置或者直接存储在指令中（在指令中用*标记）。不使用外部轴的位置。

ToPoint：

数据类型：robtarget

机器人和外部轴的目标点。定义为一个命名的位置或者直接存储在指令中（在指令中用*标记）。

[\ID]：

同步 ID

数据类型：identno

该项目必须使用在多运动系统中，如果并列了同步运动，则不允许在其他任何情况下使用。

指定的 ID 号在所有协同的程序任务中必须相同。该 ID 号保证在 routine 中运动不会混乱。

如果并列了同步运动，不允许在其他任何情况下使用。

Speed：

数据类型：speeddata

应用到运动中的速度数据。速度数据定义 TCP、工具再定位和外部轴的速度。

[\V]：

速度

数据类型：num

该项目用来在指令中直接指定 TCP 的速度，单位 mm/s。它代替速度数据中指定的相应的速度。

[\T]：

时间

数据类型：num

该项目用来指定机器人和外部轴运动的总时间，单位秒。它代替相应的速度数据。

Zone：

数据类型：zonedata

运动的 zone 数据。它描述产生的转角路径的大小。

[\Z]：

Zone

数据类型：num

该项目用来在指令中直接指定机器人 TCP 的位置精度。转角路径的长度以毫米为单位给出，它代替 zone 数据中指定的 zone。

[\Inpos]：

到位

数据类型：stoppointdata（停止点数据）

改项目用来指定机器人 TCP 在停止点位置的收敛性判别标准。该停止点数据代替在 zone 参数中指定的 zone。

Tool：

数据类型：tooldata

运动过程中所使用的工具。TCP 是运动到指定目标的点。

[\Wobj]：

工作对象

数据类型：wobjdata

机器人在指令中定位的相关到的工作对象。

该项目可以忽略，如果忽略了，定位相关到世界坐标系。在另一方面，如果使用了静态 TCP 或者并列外部轴，为了执行相关到工作对象的圆周，该项目必须被指定。

[\Corr]：

改正

数据类型：switch

如果使用该项目的话，通过 CorrWrite 指令写到改正入口的改正数据将被添加到路径和目标位置。

程序执行：

机器人和外部单元以下说明移动到目标位置：

- 工具的 TCP 按照程序中的定常速度作圆周运动。
- 工具按照定常速度重新定向，从开始位置的方向到目标点的方向。
- 重新定向相对于圆周路径执行。因此如果开始点和目标点的方向相对于路径是相同的，在移动过程中相对方向保持不变（参看下图）。

下图说明了圆周运动过程中的工具方向。

圆周点的方向没有到达，它只是用来区别重新定向中两个可能的方向。沿着路径重新定向的精度只取决于开始点和目标点的方向。

圆周运动过程中的工具方向的不同模式在指令 CirPathMode 中有描述。

非并列的外部轴以定常速度执行，目的是和机器人轴同时到达目标点。圆周点中的位置没有用到。

如果重新定向或者外部轴不能达到程序中的速度，TCP 得速度将被减小。

当运动转换到路径中的下一段的时候通常会产生转角路径。如果停止点在 zone 数据中指定，在机器人和外部轴到达合适位置的时候，程序才能继续执行。

更多范例：

如何使用该指令得更多范例说明如下：

例1　　MoveC *, *, v500 \V:=550, z40 \Z:=45, grip3;

Grip3 的 TCP 圆周运动到存储在指令中的位置。运动中把数据设定到 v500 和 z40 执行；TCP 的速度是 550mm/s，zone 的大小是 45mm。

例2　　MoveC p5, p6, v2000, fine \Inpos := inpos50, grip3;

Grip3 的 TCP 圆周运动到停止点 p6。当停止点 fine 的 50%的位置条件和 50%的速度条件满足的时候，机器

人认为它到达该点。它等待条件满足最多等两秒。参看 stoppointdata 数据类型的预定义数据 inpos50。

例3　　MoveC \Conc, *, *, v500, z40, grip3;

Grip3 的 TCP 圆周运动到指令中存储的位置。圆周点也存储在指令中。当机器人移动的时候，执行后续逻辑指令。

例4　　MoveC cir1, p15, v500, z40, grip3 \Wobj :=fixture;

Grip3 的 TCP 经过圆周点 cir1 圆周运动到位置 p15。这些位置在 fixture 的对象并列系统中指定。

限制：

对于 cirPoint 和 Topoint 如何放置有一些限制，如下图描述：

- 起点和 ToPoint 之间的最小距离是 0.1 毫米。
- 起点和 CirPoint 之间的最小距离是 0.1 毫米。
- 从起点到 CirPoint 和 ToPoint 之间的最小角度是 1 度。

在接近这些限制的时候，精度将会很差，即如果圆的起点和 ToPoint 相距较近，圆倾斜引起的缺陷可能远大于编程点所使用的精度。

确保机器人在程序执行过程中可以到达 Circle Point（圆周点），必要的话把圆再分段。

当机器人停止在圆周路径上，执行模式从向前到向后得改变，或者相反，是不允许的，并且将导致错误信息。

警告！

当 TCP 在圆周点和终点之间的时候，MoveC 指令（或者任何其它包括圆周运动的指令）不允许从开头执行。否则机器人将不能执行编程的路径（从和编程路径方向不同的方向绕圆周路径定位）。

语法：

MoveC ['\' Conc ','] [CirPoint' :='] <robtarget 类型的表达式(IN)> ','

[ToPoint' :='] < robtarget 类型的表达式（IN）> ','

['\'ID ':=' <identno 类型的表达式（IN）>]','

[Speed ':='] <speeddata 类型的表达式（IN）>

['\' V ':=' <num 类型的表达式（IN）>]

['\' T ':=' <num 类型的表达式（IN）>] ','

[zone ' :=']<zonedata 类型的表达式（IN）>

['\' Z ':=' <num 类型的表达式(IN)>]

['\'Inpos' :=' <stoppointdata 类型的表达式（IN）>] ','

[Tool ':='] <tooldata 类型的恒量（PERS）>

['\'Wobj ':=' <wobjdata 类型的恒量（PERS）>]

['\' Corr] ';'

相关信息：

相关信息	参看
其他位置指令	RAPID 参考手册－RAPID 概述，RAPID 摘要－运动部分
速度的定义	第 1010 页 speeddata—速度数据
Zone 数据的定义	第 1047 页 zonedata—zone 数据
停止点数据的定义	第 1014 页 stoppointdata—停止点数据
工具的定义	第 1031 页 tooldata—工具数据
工作对象的定义	第 1039 页 wobjdata—工作对象数据
写到改正入口	第 67 页 CorrWrite－写到改正入口
在圆周运动中工具重新定向	第 32 页 CirPathMode－在圆周路径中工具重新定向
运动综述	RAPID 参考手册—RAPID 概述，运动和 I/O 原理部分
并列系统	RAPID 参考手册－RAPID 概述，运动和 I/O 原理－并列系统部分
并发的程序执行	RAPID 参考手册—RAPID 概述，运动和 I/O 原理—用逻辑指令同步部分

1.90．MoveCDO－圆周移动机器人并且在转角处设置数字输出

用途：

MoveCDO（圆周移动数字输出）用来把 TCP 圆周移动到一个给定的目标点。指定的数字输出在目标点的转角路径的中间被置位/复位。在运动过程中，相对于圆周的方向通常保持不变。

该指令只能用在主任务 T_ROB1，或者多运动系统的运动任务中。

基本范例：

该指令的基本范例说明如下。

例1　　MoveCDO p1, p2, v500, z30, tool2, do1, 1;

Tool2 的 TCP 圆周移动到位置 p2,速度数据 v500 和 zone 数据 z30。圆周由开始点、圆周点 p1 和目标点 p2 确定。在转角路径 p2 的中间位置设置输出 do1。

项目：

MoveCDO CirPoint ToPoint [\ID] Speed [\T] Zone Tool [\Wobj] Signal Value

CirPoint：

数据类型：robtarget

机器人的圆周点。圆周点是圆周上开始点和目标点之间的一个位置。为了获得最好的精度，它最好处于开

始点和目标点一半的位置。如果它太靠近开始点或者目标点，机器人将给出一个警告。圆周点定义为一个命名的位置或者直接存储在指令中（在指令中用*标记）。不使用外部轴的位置。

ToPoint：

数据类型：robtarget

机器人和外部轴的目标点。定义为一个命名的位置或者直接存储在指令中（在指令中用*标记）。

[\ID]：

同步 ID

数据类型：identno

该项目必须用在并列了同步运动的多运动系统中，不允许在其它任何条件下使用。

在所有协作的程序任务中，指定的 ID 号码必须相同。ID 号保证了在 routine 中运动不会混淆。

Speed：

数据类型：speeddata

应用到运动中的速度数据。速度数据定义 TCP、工具重新定向和外部轴的速度。

[\T]：

时间

数据类型：num

该项目用来指定机器人和外部轴运动的总时间，单位秒。它用来替换相应的速度数据。

Zone：

数据类型：zonedata

运动的 zone 数据。Zone 数据描述产生的转角路径的大小。

Tool：

数据类型：tooldata

当机器人运动时时用的工具。工具中心点就是运动到目标点的那个点。

[\Wobj]：

工作对象

数据类型：wobjdata

工作对象（对象坐标系统），就是在指令中机器人相关到的对象。

该项目可以忽略，如果忽略的话，位置相关到世界坐标系。另一方面，如果使用了静止 TCP 或者并列的外部轴，为了执行相关到工作对象的圆周，该项目必须指定。

Signal：

数据类型：signaldo

要改变的数字输出信号的名称。

Value：

数据类型：dionum

期望的信号的数值（0 或者 1）。

程序执行：

关于圆周运动得更多信息参看指令 MoveC。

在飞点的转角路径的中间位置，数字输出信号置位/复位，如下图所示。

下图说明在转角路径 MoveC 指令的数字输出信号的置位/复位。

对于停止点，我们推荐使用“正常”的编程顺序，即 MoveC＋SetDO。但是当在指令 MoveCDO 中使用停止点、当机器人到达停止点的时候，数字输出信号置位/复位。

在执行模式继续逐步向前而不是逐步向后时，指定的 I/O 信号被置位/复位。

限制：

按照指令 MoveC 的常规限制。

语法：

MoveCDO [CirPoint ‘:=’] <robtarget 类型的表达式（IN）> ‘,’

[ToPoint’ :=’] <robtarget 类型的表达式（IN）> ‘,’

[‘\’ ID ‘:=’<identno 类型的表达式（IN）>]’,’

[Speed ‘:=’]<speeddata 类型的表达式（IN）>

[‘\’ T ‘:=’ < num 类型的表达式（IN）>] ‘,’

[Zone ‘:=’]<zonedata 类型的表达式（IN）> ‘,’

[Tool ‘:=’]<tooldata 类型的恒量（PERS）>

[‘\’ Wobj’ :=’ <wobjdata 类型的恒量（PERS）>]’,’

[Signal ‘:=’] <signaldo 类型的变量（VAR）>] ‘,’

[Value ‘:=’] <dionum 类型的表达式（IN）>] ‘;’

相关信息：

相关信息	**参看**
其他位置指令	RAPID 参考手册－RAPID 概述，RAPID 摘要－运动部分
圆周运动机器人	第 209 页 MoveC－圆周移动机器人
速度的定义	第 1010 页 speeddata—速度数据
Zone 数据的定义	第 1047 页 zonedata—zone 数据

工具的定义	第 1031 页 tooldata—工具数据
工作对象的定义	第 1039 页 wobjdata—工作对象数据
运动综述	RAPID 参考手册—RAPID 概述，运动和 I/O 原理部分
坐标系	RAPID 参考手册－RAPID 概述，运动和 I/O 原理－坐标系部分
带 I/O 设定的运动	RAPID 参考手册—RAPID 概述，运动和 I/O 原理—用逻辑指令同步部分

1.91．MoveCSync－圆周移动机器人，并且执行一个 RAPID 程序

用途：

MoveCSync（同步圆周移动）用来圆周移动 TCP 到一个给定的目标位置。在目标点的转角路径的中间位置，指定的 RAPID 程序开始运行。在运动过程中，相对于圆周的方向通常保持不变。

该指令只能用在主任务 T_ROB1，或者多运动系统的运动任务中。

基本范例：

该指令的基本范例说明如下。

例2　　MoveCSync p1, p2, v500, z30, tool2, “proc1”;

Tool2 的 TCP 圆周移动到位置 p2,速度数据 v500 和 zone 数据 z30。圆周由开始点、圆周点 p1 和目标点 p2 确定。在转角路径 p2 的中间位置程序 proc1 开始执行。

项目：

MoveCSync CirPoint ToPoint [\ID] Speed [\T] Zone Tool [\Wobj] ProcName

CirPoint：

数据类型：robtarget

机器人的圆周点。圆周点是圆周上开始点和目标点之间的一个位置。为了获得最好的精度，它最好处于开始点和目标点一半的位置。如果它太靠近开始点或者目标点，机器人将给出一个警告。圆周点定义为一个命名的位置或者直接存储在指令中（在指令中用*标记）。不使用外部轴的位置。

ToPoint：

数据类型：robtarget

机器人和外部轴的目标点。定义为一个命名的位置或者直接存储在指令中（在指令中用*标记）。

[\ID]：

同步 ID

数据类型：identno

该项目必须用在并列了同步运动的多运动系统中，不允许在其它任何条件下使用。

在所有协作的程序任务中，指定的 ID 号码必须相同。ID 号保证了在 routine 中运动不会混淆。

Speed：

数据类型：speeddata

应用到运动中的速度数据。速度数据定义 TCP、工具重新定向和外部轴的速度。

[\T]：

时间

数据类型：num

该项目用来指定机器人和外部轴运动的总时间，单位秒。它用来替换相应的速度数据。

Zone：

数据类型：zonedata

运动的 zone 数据。Zone 数据描述产生的转角路径的大小。

Tool：

数据类型：tooldata

当机器人运动时时用的工具。工具中心点就是运动到目标点的那个点。

[\Wobj]：

工作对象

数据类型：wobjdata

工作对象（对象坐标系统），就是在指令中机器人相关到的对象。

该项目可以忽略，如果忽略的话，位置相关到世界坐标系。另一方面，如果使用了静止 TCP 或者并列的外部轴，为了执行相关到工作对象的圆周，该项目必须指定。

ProcName：

程序名称

数据类型：string

在目标点的转角路径的中间位置要执行的 RAPID 程序的名称。

程序执行：

关于圆周运动得更多信息参看指令 MoveC。

当 TCP 到达 MoveCSync 指令的目标点的转角路径的中间位置时，指定的 RAPID 程序开始执行，如下图所示。

下图说明在转角路径的中间位置用户定义的 RAPID 程序的执行。

对于停止点，我们推荐使用“正常”的编程顺序，即 MoveC＋其他 RAPID 程序。

下表描述了在不同执行模式下指定的 RAPID 程序的执行：

执行模式	RAPID 程序的执行
继续或者循环	按照该描述
逐步向前	在停止点
逐步向后	一点也不执行

限制：

按照指令 MoveC 的常规限制。

当程序停止后，从连续执行或循环执行切换到逐步向前或者向后将导致错误。该错误告诉用户模式切换将导致路径上的执行队列的 RAPID 程序的执行错误。

指令 MoveCSync 不能用在 TRAP 层次上。指定的 RAPID 程序不能用逐步执行测试。

语法：

MoveCSync [CirPoint ‘:=’] <robtarget 类型的表达式（IN）> ‘,’

[ToPoint’ :=’] <robtarget 类型的表达式（IN）> ‘,’

[‘\’ ID ‘:=’<identno 类型的表达式（IN）>]’,’

[Speed ‘:=’]<speeddata 类型的表达式（IN）>

[‘\’ T ‘:=’ < num 类型的表达式（IN）>] ‘,’

[Zone ‘:=’]<zonedata 类型的表达式（IN）> ‘,’

[Tool ‘:=’]<tooldata 类型的恒量（PERS）>

[‘\’ Wobj’ :=’ <wobjdata 类型的恒量（PERS）>]’,’

[ProcName ‘:=’] <sting 类型的表达式（IN）>] ‘;’

相关信息：

相关信息	**参看**	
其他位置指令	RAPID 参考手册－RAPID 概述，RAPID 摘要－运动部分	
圆周运动机器人	第 209 页 MoveC－圆周移动机器人	
速度的定义	第 1010 页 speeddata—速度数据	
Zone 数据的定义	第 1047 页 zonedata—zone 数据	
工具的定义	第 1031 页 tooldata—工具数据	
工作对象的定义	第 1039 页 wobjdata—工作对象数据	
运动综述	RAPID 参考手册—RAPID 概述，运动和 I/O 原理部分	

坐标系	RAPID 参考手册－RAPID 概述，运动和 I/O 原理－坐标系部分	
带 I/O 设定的运动	RAPID 参考手册—RAPID 概述，运动和 I/O 原理—用逻辑指令同步部分	

1.92．MoveExtJ－移动一个或者多个没有 TCP 的机械单元

用途：

MoveExtJ（移动外部关节）只用来移动线性或者旋转外部轴。该外部轴可以属于一个或者多个没有 TCP 的外部单元。

该指令只能用来：

- 和定义为运动任务的实际程序任务一起使用，并且
- 如果任务控制一个或者多个没有 TCP 的机械单元。

基本范例：

该指令的基本范例说明如下：

也参看第 225 页的更多范例。

例1　　MoveExtJ jpos10, vrot10, z50;

移动旋转外部轴到关节位置 jpos10，速度 10°/秒，zone 数据 z50。

例2　　MoveExtJ \Conc, jpos20, vrot10 \T:=5, fine \InPos:=inpos20;

5 秒钟把外部轴移动到关节位置 jpos20。程序立即向前执行，但是外部轴停止在位置 jpos20，直到 inpos20 的收敛性标准满足。

项目：

MoveExtJ [\Conc] To JointPos [\ID] Speed [\T] Zone [\Inpos]

[\Conc]：

并发事件

数据类型：switch

当外部轴运动的同时，后续的指令开始执行。该项目通常不使用，但是当使用飞点（flyby points）时，可以用来避免由 CPU 过载引起的不想要的停止。当使用高速度并且编程点相距较近时这是很有用的。例如，当不要求与外部设备通讯和外部设备和机器人通讯同步的时候，这个项目也很有用。

使用项目\Conc 的时候，连续的运动指令的数量限制为 5 个。在包括 StorePath—RestorePath 的程序段中不允许使用带有\Conc 项目的运动指令。

如果不使用该项目，并且 ToPoint 不是停止点，在机器人到达程序 zone 之前一段时间后续指令就开始执行

了。

在多运动系统中的坐标同步运动中不能使用该项目。

ToJointPos：

到达关节位置

数据类型：jointtarget

外部轴的绝对目标轴位置。定义为一个命名的位置或者直接存储在指令中（在指令中用*标记）。

[\ID]：

同步 ID

数据类型：identno

该项目必须使用在多运动系统中，如果并列了同步运动，则不允许在其他任何情况下使用。

指定的 ID 号在所有协同的程序任务中必须相同。该 ID 号保证在 routine 中运动不会混乱。

如果并列了同步运动，不允许在其他任何情况下使用。

Speed：

数据类型：speeddata

应用到运动中的速度数据。速度数据定义旋转或者线性外部轴的速度。

[\T]：

时间

数据类型：num

该项目用来指定外部轴运动的总时间，单位秒。它代替相应的速度数据。

Zone：

数据类型：zonedata

运动的 zone 数据。它定义停止点或者飞点。如果是飞点它描述线性或者旋转外部轴的减速度或者加速度。

[\Inpos]：

到位

数据类型：stoppointdata（停止点数据）

该项目用来指定机器人 TCP 在停止点位置的收敛性判别标准。该停止点数据代替在 zone 参数中指定的 zone。

程序执行：

线性或者旋转外部轴按照编程的速度移动到编程的点。

更多范例：

```
CONST    jointtarget    j1 :=[[9E9,9E9.9E9.9E9.9E9.9E9.9E9],[0,9E9,9E9,9E9,9E9,9E9,9E9]];
```

```
CONST    jointtarget    j2 :=[[9E9,9E9.9E9.9E9.9E9.9E9.9E9],[30,9E9,9E9,9E9,9E9,9E9]];
CONST    jointtarget    j3 :=[[9E9,9E9.9E9.9E9.9E9.9E9.9E9],[60,9E9,9E9,9E9,9E9,9E9]];
CONST    jointtarget    j4 :=[[9E9,9E9.9E9.9E9.9E9.9E9.9E9],[90,9E9,9E9,9E9,9E9,9E9]];
CONST    speeddata    rot_ax_speed    :=[0,0,0,45];

MoveeExtJ j1, rot_ax_speed, fine;
MoveeExtJ j2, rot_ax_speed, z20;
MoveeExtJ j3, rot_ax_speed, z20;
MoveeExtJ j4, rot_ax_speed, fine
```

在该例子中，旋转独立轴移动到轴位置 0, 30, 60 和 90 度，移动速度为 45 度/秒。

语法：

```
MoveExtJ ['\' Conc ',']
    [ ToJointPos' :=' ]<jointtarget 类型的表达式（IN）>
        ['\' ID ':=' <identno 类型的表达式（IN）>]','
    [Speed ':=' ]<speeddata 类型的表达式（IN）>
        ['\'T ':=' <num 类型的表达式（IN）>] ','
    [Zone ':=' ]<zonedata 类型的表达式（IN）>
        ['\' Inpos ':=' <stoppointdata 类型的表达式（IN）>] ';'
```

相关信息：

相关信息	参看	
其他位置指令	RAPID 参考手册－RAPID 概述，RAPID 摘要－运动部分	
关节目标（jointtarget）的定义	第 959 页 jointtarget－关节位置数据	
速度的定义	第 1010 页 speeddata—速度数据	
Zone 数据的定义	第 1047 页 zonedata—zone 数据	
运动综述	RAPID 参考手册—RAPID 概述，运动和 I/O 原理部分	
当前程序执行	RAPID 参考手册－RAPID 概述，运动和 I/O 原理部分	

1.93．MoveJ－通过关节移动移动机器人

用途：

当运动不必是直线的时候，MoveJ 用来快速将机器人从一个点运动到另一个点。

机器人和外部轴沿着一个非直线的路径移动到目标点，所有轴同时到达目标点。

该指令只能用在主任务 T_ROB1 中，或者在多运动系统中的运动任务中。

基本范例：

该指令的基本范例说明如下：

也可参看第 228 页更多指令。

例1　MoveJ p1, vmax, z30, tool2;

工具 tool2 的 TCP 沿着一个非线性路径到位置 p1，速度数据是 vmax，zone 数据是 z30。

例 2　MoveJ *, vmax \T:=5, fine, grip3;

工具 grip3 的 TCP 沿着一个非线性路径运动到存储在指令中的停止点（用*标记）。整个运动需要 5 秒钟。

项目：

MoveJ [\Conc] ToPoint [\ID] Speed [\V] | [\T] Zone [\Z] [\Inpos] Tool [\WObj]

[\Conc]：

并发事件

数据类型：switch

当机器人运动的同时，后续的指令开始执行。该项目通常不使用，但是当使用飞点（flyby points）时，可以用来避免由 CPU 过载引起的不想要的停止。当使用高速度并且编程点相距较近时这是很有用的。例如，当不要求与外部设备通讯或外部设备和机器人通讯同步的时候，这个项目也很有用。

使用项目\Conc 的时候，连续的运动指令的数量限制为 5 个。在包括 StorePath—RestorePath 的程序段中不允许使用带有\Conc 项目的运动指令。

如果不使用该项目，并且 ToPoint 不是停止点，在机器人到达程序 zone 之前一段时间后续指令就开始执行了。

在多运动系统中的坐标同步运动中不能使用该项目。

ToPoint：

数据类型：robtarget

机器人和外部轴的目标位置。定义为一个命名的位置或者直接存储在指令中（在指令中用*标记）。

[\ID]：

同步 ID

数据类型：identno

该项目必须使用在多运动系统中，如果并列了同步运动，则不允许在其他任何情况下使用。

指定的 ID 号在所有协同的程序任务中必须相同。该 ID 号保证在 routine 中运动不会混乱。

如果并列了同步运动，不允许在其他任何情况下使用。

Speed：

数据类型：speeddata

应用到运动中的速度数据。速度数据定义 TCP、工具重新定向或者外部轴的速度。

[\V]：

速度

数据类型：num

该项目用来在指令中直接指定 TCP 的速度，单位 mm/s。它用来代替速度数据中相应的速度。

[\T]：

时间

数据类型：num

该项目用来指定外部轴运动的总时间，单位秒。它代替相应的速度数据。

Zone：

数据类型：zonedata

运动的 zone 数据。它描述产生的转角路径的大小。

[\Z]：

Zone

数据类型：num

该项目用来在指令中直接指定机器人 TCP 的位置精度。转角路径的长度单位是 mm，它代替 zone 数据中相应的 zone。

[\Inpos]：

到位

数据类型：stoppointdata（停止点数据）

该项目用来指定机器人 TCP 在停止点位置的收敛性判别标准。该停止点数据代替在 zone 参数中指定的 zone。

Tool：

数据类型：tooldata

当机器人运动的时候使用的工具。TCP 是移动到指定的目标点的那个点。

[\Wobj]：

工作对象

数据类型：wobjdata

指令中机器人位置相关到的工作对象（坐标系）。该项目可以忽略，如果忽略的话，位置相关到世界坐标系。

另一方面，如果使用了静态TCP或者并列了外部轴，该项目必须指定。

程序执行：

机器人TCP用轴角度插补移动到目标点。也就是说每一个轴都使用一个固定的轴速度并且所有轴同时到达目标点，所走的是一个非线性的路径。

总的来说，TCP按照大约的编程速度运动（无论是否并列了外部轴）。在TCP运动的同时，工具重新定向，外部轴也在运行。如果不能达到工具重新定向或者外部轴的编程的速度，TCP的速度将降低。

当运动路径转到下一段的时候，通常会产生转角路径。如果在zone数据中指定了停止点，只有当机器人和外部轴到达合适的位置的时候，程序执行才会继续。

更多范例：

如何使用改指令的更多范例说明如下：

例1　　MoveJ *, v2000\V:=2200, z40 \Z:=45, grip3;

工具grip3的TCP沿着一个非线性路径移动到存储在指令中的位置。运动执行数据被设定为v2000和z40；TCP的速度和zone大小分别是2200mm/s和45mm。

例2　　MoveJ p5, v2000, fine \Inpos := inpos50, grip3;

工具grip3的TCP沿非线性路径移动到停止点p5。当停止点fine的50%的位置条件和50%的速度条件满足的时候，机器人认为它已经到达该点。它等待条件满足最多等2秒。参考stoppointdata类型的预定义数据inpos50。

例3　　MoveJ \Conc, *, v2000, z40, grip3;

工具grip3的TCP沿着一个非线性移动到指令中存储的位置。当机器人移动的时候，后续逻辑指令开始执行。

例4　　MoveJ start, v2000, z40, grip3 \WObj:=fixture;

工具grip3的TCP沿着一个非线性的路径到位置start。该位置在fixture的对象坐标系统中指定。

语法：

MoveJ

['\' Conc ',']

[ToPoint ':='] <robtarget类型的表达式(IN) > ','_

['\' ID ':=' < identno类型的表达式(IN) >]','_

[Speed ':='] < speeddata类型的表达式(IN)　>_

['\' V ':=' < num 类型的表达式(IN)>]_

| ['\' T ':=' < num 类型的表达式(IN) >] ','_

[Zone ':='] < zonedata类型的表达式(IN)>_

['\' Z ':=' < num类型的表达式(IN)>]_

['\' Inpos ':=' < stoppointdata类型的表达式(IN)>] ','_

[Tool ':='] < tooldata类型的恒量(PERS) >_

['\' WObj ':=' <wobjdata 类型的恒量(PERS)>] ';'

相关信息：

相关信息	参看
其他位置指令	RAPID 参考手册－RAPID 概述，RAPID 摘要－运动部分
速度的定义	第 1010 页 speeddata—速度数据
Zone 数据的定义	第 1047 页 zonedata—zone 数据
停止点数据的定义	第 1014 页 stoppointdata－停止点数据
工具的定义	第 1031 页 tooldata—工具数据
工作对象的定义	第 1039 页 wobjdata—工作对象数据
运动综述	RAPID 参考手册—RAPID 概述，运动和 I/O 原理部分
坐标系	RAPID 参考手册－RAPID 概述，运动和 I/O 原理－坐标系部分
并发的程序执行	RAPID 参考手册—RAPID 概述，运动和 I/O 原理—用逻辑指令同步部分

1.94. MoveJDO－通过关节运动移动机器人在转角设定数字输出

用途：

MoveJDO（关节运动数字输出）在运动不必是直线的时候用来快速把机器人从一个点移动到另一个点。在转角路径的中间位置，指定的数字输出信号被置位/复位。

机器人和外部轴沿着一个非线性的路径移动到目标位置。所有轴在同一时间到达目标位置。

该指令只能用在主任务 T_ROB1 中，或者在多运动系统中的运动任务中。

基本范例：

该指令的基本范例说明如下：

例1 MoveJDO p1, vmax, z30, tool2, do1, 1;

工具tool2的TCP沿着一个非线性路径移动到目标位置p1，速度数据vmax和zone数据z30。在p1的转角路径的中间位置，输出信号do1被置位。

项目：

MoveJDO ToPoint [\ID] Speed [\T] Zone Tool [\WObj] Signal Value

ToPoint：

数据类型：robtarget

机器人和外部轴的目标位置。定义为一个命名的位置或者直接存储在指令中（在指令中用*标记）。

[\ID]：

同步 ID

数据类型：identno

该项目必须使用在多运动系统中，如果并列了同步运动，则不允许在其他任何情况下使用。

指定的 ID 号在所有协同的程序任务中必须相同。该 ID 号保证在 routine 中运动不会混乱。

如果并列了同步运动，不允许在其他任何情况下使用。

Speed：

数据类型：speeddata

应用到运动中的速度数据。速度数据定义 TCP、工具重新定向或者外部轴的速度。

[\T]：

时间

数据类型：num

该项目用来指定外部轴运动的总时间，单位秒。它代替相应的速度数据。

Zone：

数据类型：zonedata

运动的 zone 数据。它描述产生的转角路径的大小。

Tool：

数据类型：tooldata

机器人运动时所使用的工具。TCP 就是移动到目标点的那个点。

[\Wobj]：

工作对象

数据类型：wobjdata

指令中机器人位置相关到的工作对象（坐标系）。该项目可以忽略，如果忽略的话，位置相关到世界坐标系。另一方面，如果使用了静态TCP或者并列了外部轴，该项目必须指定。

Signal：

数据类型：signaldo

要改变的数字输出信号的名称。

Value：

数据类型：dionum

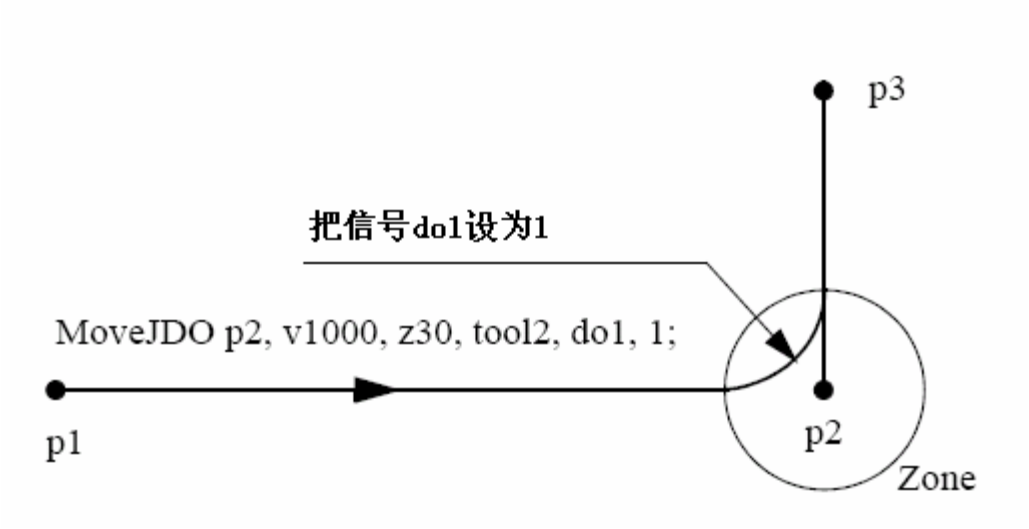
期望的信号数值（0或者1）。

程序执行：

参考指令MoveJ，可以得到关节运动的更多信息。

在飞点的转角路径的中间位置，数字输出信号置位/复位，如下图所示。

下图说明在转角路径 MoveJ 指令的数字输出信号的置位/复位。



对于停止点，我们推荐使用“正常”的编程顺序，即 MoveJ＋SetDO。但是当在指令 MoveJDO 中使用停止点、当机器人到达停止点的时候，数字输出信号置位/复位。

在执行模式继续逐步向前而不是逐步向后时，指定的 I/O 信号被置位/复位。

语法：

MoveJDO_

[ToPoint ’:=’] < robtarget类型的表达式(IN) > ’,’_

[’\’ ID ’:=’ < identno类型的表达式(IN) >]’,’_

[Speed ’:=’] < speeddata类型的表达式(IN) >_

[’\’ T ’:=’ < num类型的表达式(IN) >] ’,’_

[Zone ’:=’] < zonedata类型的表达式(IN)　> ’,’_

[Tool ’:=’] < tooldata类型的恒量(PERS) >_

[’\’ WObj ’:=’ < wobjdata类型的恒量(PERS) >] ’,’_

[Signal ’:=’] < signaldo类型的变量(VAR)>] ‘,’_

[Value ‘:=’] < dionum类型的表达式(IN) >] ’;’

相关信息：

相关信息	**参看**
其他位置指令	RAPID 参考手册－RAPID 概述，RAPID 摘要－运动部分
通过关节运动移动机器人	第 226 页 MoveJ－通过关节运动移动机器人
速度的定义	第 1010 页 speeddata—速度数据
Zone 数据的定义	第 1047 页 zonedata—zone 数据

工具的定义	第 1031 页 tooldata—工具数据
工作对象的定义	第 1039 页 wobjdata—工作对象数据
运动综述	RAPID 参考手册—RAPID 概述，运动和 I/O 原理部分
坐标系	RAPID 参考手册－RAPID 概述，运动和 I/O 原理－坐标系部分
带 I/O 设定的运动	RAPID 参考手册—RAPID 概述，运动和 I/O 原理—用逻辑指令同步部分

1.95.MoveJSync－通过关节运动移动机器人，并且执行一个 RAPID 程序

用途：

MoveJSync（同步关节移动）用来在不要求直线运动的时候把机器人从一个点快速移动到另一个点。在目标点的转角路径的中间位置，指定的 RAPID 程序开始运行。在运动过程中，相对于圆周的方向通常保持不变。

机器人和外部轴沿着一个非线性的路径移动到目标位置。所有轴在同一时间到达目标位置。

该指令只能用在主任务 T_ROB1，或者多运动系统的运动任务中。

基本范例：

该指令的基本范例说明如下。

例 1　　MoveJSync p1, vmax, z30, tool2, “proc1”;

工具 tool2 的 TCP 沿着一个非线性路径移动到位置 p1，速度数据 vmax，zone 数据 z30。在 p1 的转角路径的中间位置程序 proc1 开始执行。

项目：

MoveJSync ToPoint [\ID] Speed [\T] Zone Tool [\WObj] ProcName

ToPoint：

数据类型：robtarget

机器人和外部轴的目标点。定义为一个命名的位置或者直接存储在指令中（在指令中用*标记）。

[\ID]：

同步 ID

数据类型：identno

该项目必须使用在多运动系统中，如果并列了同步运动，则不允许在其他任何情况下使用。

指定的 ID 号在所有协同的程序任务中必须相同。该 ID 号保证在 routine 中运动不会混乱。

如果并列了同步运动，不允许在其他任何情况下使用。

Speed：

数据类型：speeddata

应用到运动中的速度数据。速度数据定义 TCP、工具重新定向或者外部轴的速度。

[\T]：

时间

数据类型：num

该项目用来指定外部轴运动的总时间，单位秒。它代替相应的速度数据。

Zone：

数据类型：zonedata

运动的 zone 数据。它描述产生的转角路径的大小。

Tool：

数据类型：tooldata

机器人运动时所使用的工具。TCP 就是移动到目标点的那个点。

[\Wobj]：

工作对象

数据类型：wobjdata

指令中机器人位置相关到的工作对象（坐标系）。该项目可以忽略，如果忽略的话，位置相关到世界坐标系。另一方面，如果使用了静态TCP或者并列了外部轴，该项目必须指定。

ProcName：

程序名称

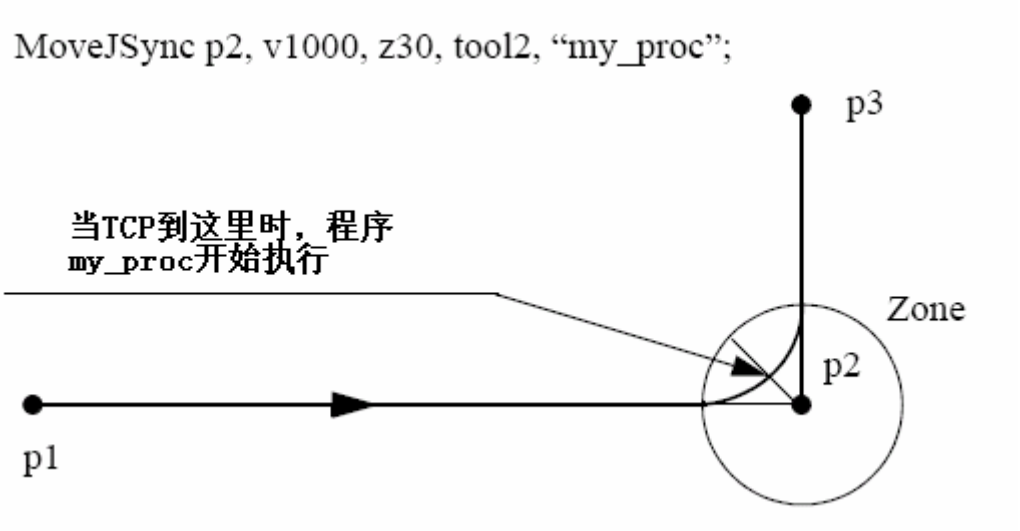
数据类型：string

在目标点的转角路径的中间位置要执行的RAPID程序的名称。

程序执行：

参考指令MoveJ，可以得到关节运动的更多信息。

当TCP到达MoveJSync指令的目标点的转角路径的中间位置时，指定的RAPID程序开始执行，如下图所示。



对于停止点，我们推荐使用“正常”的编程顺序，即 MoveJ＋其他 RAPID 指令。

下表描述了在不同执行模式下指定的 RAPID 程序的执行：

执行模式	RAPID 程序的执行
继续或者循环	按照该描述
逐步向前	在停止点
逐步向后	一点也不执行

限制：

当程序停止后，从连续执行或循环执行切换到逐步向前或者向后将导致错误。该错误告诉用户模式切换将导致路径上的执行队列的 RAPID 程序的执行错误。

指令 MoveJSync 不能用在 TRAP 层次上。指定的 RAPID 程序不能用逐步执行测试。

语法：

MoveJSync

[ToPoint ’:=’] < robtarget类型的表达式(IN) > ’,’

[’\’ ID ’:=’ < identno类型的表达式(IN)>]’,’

[Speed ’:=’] < speeddata类型的表达式(IN) >

[’\’ T ’:=’ < num类型的表达式(IN) >] ’,’

[Zone ’:=’] < zonedata类型的表达式(IN) >

[’\’ Z ‘:=’ < num类型的表达式(IN) >] ’,’

[Tool ’:=’] < tooldata类型的恒量(PERS) >

[’\’ WObj ’:=’ < wobjdata类型的恒量(PERS)>] ’,’

[ProcName‘:=’] <string类型的表达式(IN) >] ’;’

相关信息：

相关信息	参看
其他位置指令	RAPID 参考手册－RAPID 概述，RAPID 摘要－运动部分
通过关节运动移动机器人	第 226 页 MoveJ－通过关节运动移动机器人
速度的定义	第 1010 页 speeddata—速度数据
Zone 数据的定义	第 1047 页 zonedata—zone 数据
工具的定义	第 1031 页 tooldata—工具数据
工作对象的定义	第 1039 页 wobjdata—工作对象数据
运动综述	RAPID 参考手册—RAPID 概述，运动和 I/O 原理部分
坐标系	RAPID 参考手册－RAPID 概述，运动和 I/O 原理－坐标系部分

1.96.MoveL－让机器人作直线运动

用途：

MoveL 用来让机器人 TCP 直线运动到给定的目标位置。当 TCP 仍旧固定的时候，该指令也可以重新给工具定方向。

该指令只能用在主任务 T_ROB1，或者多运动系统的运动任务中。

基本范例：

该指令的基本范例说明如下。

也可以参看第 238 页更多范例。

例1　　MoveL p1, v1000, z30, tool2;

Tool2 的 TCP 沿直线运动到位置 p1，数度数据为 v1000，zone 数据为 z30。

例2　　MoveL *, v1000\T:=5, fine, grip3;

Grip3 的 TCP 沿直线运动到存储在指令中的停止点（用*标记）。整个的运动过程需时 5 秒。

项目：

MoveL [\Conc] ToPoint [\ID] Speed [\V] | [\T] Zone [\Z] [\Inpos] Tool [\WObj] [\Corr]

[\Conc]：

并发事件

数据类型：switch

当机器人运动的同时，后续的指令开始执行。该项目通常不使用，但是当使用飞点（flyby points）时，可以用来避免由 CPU 过载引起的不想要的停止。当使用高速度并且编程点相距较近时这是很有用的。例如，当不要求与外部设备通讯或外部设备和机器人通讯同步的时候，这个项目也很有用。

使用项目\Conc 的时候，连续的运动指令的数量限制为 5 个。在包括 StorePath—RestorePath 的程序段中不

允许使用带有\Conc 项目的运动指令。

如果不使用该项目，并且 ToPoint 不是停止点，在机器人到达程序 zone 之前一段时间后续指令就开始执行了。

在多运动系统中的坐标同步运动中不能使用该项目。

ToPoint：

数据类型：robtarget

机器人和外部轴的目标位置。定义为一个命名的位置或者直接存储在指令中（在指令中用*标记）。

[\ID]：

同步 ID

数据类型：identno

该项目必须使用在多运动系统中，如果并列了同步运动，则不允许在其他任何情况下使用。

指定的 ID 号在所有协同的程序任务中必须相同。该 ID 号保证在 routine 中运动不会混乱。

Speed：

数据类型：speeddata

应用到运动中的速度数据。速度数据定义 TCP、工具重新定向或者外部轴的速度。

[\V]：

速度

数据类型：num

该项目用来在指令中直接指定 TCP 的速度，单位 mm/s。它用来代替速度数据中相应的速度。

[\T]：

时间

数据类型：num

该项目用来指定外部轴运动的总时间，单位秒。它代替相应的速度数据。

Zone：

数据类型：zonedata

运动的 zone 数据。它描述产生的转角路径的大小。

[\Z]：

Zone

数据类型：num

该项目用来在指令中直接指定机器人 TCP 的位置精度。转角路径的长度单位是 mm，它代替 zone 数据中相应的 zone。

[\Inpos]：

到位

数据类型：stoppointdata（停止点数据）

该项目用来指定机器人 TCP 在停止点位置的收敛性判别标准。该停止点数据代替在 zone 参数中指定的 zone。

Tool：

数据类型：tooldata

当机器人运动的时候使用的工具。TCP 是移动到指定的目标点的那个点。

[\Wobj]：

工作对象

数据类型：wobjdata

指令中机器人位置相关到的工作对象（坐标系）。该项目可以忽略，如果忽略的话，位置相关到世界坐标系。另一方面，如果使用了静态TCP或者并列了外部轴，该项目必须指定。

[\Corr]：

改正

数据类型：switch

如果使用该项目的话，通过 CorrWrite 指令写到改正入口的改正数据将被添加到路径和目标位置。

程序执行：

机器人和外部单元按照下列步骤运动：

- 工具的TCP按照程序中的速度匀速直线运动。
- 工具沿着路径以相等的间隔重新定向。
- 为了和机器人轴在同一时间到达目标位置，非并列的外部轴按照匀速度执行。

重新定向或者外部轴如果不能达到程序中的速度，TCP的速度将减小。

当运动路径转到下一段的时候，通常会产生转角路径。如果在zone数据中指定了停止点，只有当机器人和外部轴到达合适的位置的时候，程序执行才会继续。

更多范例：

如何使用改指令的更多范例说明如下：

例1 **MoveL *, v2000 \V:=2200, z40 \Z:=45, grip3;**

Grip3的TCP线性移动到存储在指令中的位置。该运动执行时的数据为v2000和z40。TCP的速度和zone大小分别是2200mm/s和45mm。

例2 **MoveL p5, v2000, fine \Inpos := inpos50, grip3;**

Grip3的TCP沿直线运动到停止点p5。当停止点fine的50%的位置条件和50%的速度条件满足的时候，机器人认为它到达了目标点。它等条件满足最多等两秒，参看stoppointdata数据类型的预定义数据inpos50。

例3 **MoveL \Conc, *, v2000, z40, grip3;**

Grip3的TCP直线运动到存储在指令中的位置。当机器人移动的时候，后续的逻辑指令开始执行。

例4 **MoveL start, v2000, z40, grip3 \WObj:=fixture;**

Grip3的TCP直线运动到位置start，位置在fixture的对象坐标系统中指定。

语法：

MoveL _

['\' Conc ',']_

[ToPoint ':='] < *robtarget类型的表达式*(**IN**) > ','

['\' ID ':=' < *identno类型的表达式*(**IN**) >]','_

[Speed ':='] < *speeddata类型的表达式*(**IN**) >_

['\' V ':=' < *num类型的表达式*(**IN**) >] _

| ['\' T ':=' < *num类型的表达式*(**IN**) >] ','_

[Zone ':='] < *zonedata类型的表达式*(**IN**) >_

['\' Z ':=' < *num类型的表达式*(**IN**) >]_

['\' Inpos ':=' < *stoppointdata类型的表达式*(**IN**) >] ','_

[Tool ':='] < *tooldata类型的恒量*(**PERS**) > _

['\' WObj ':=' < *wobjdata 类型的恒量*(**PERS**)>] _

['\' Corr]';'

相关信息：

相关信息	**参看**
其他位置指令	RAPID 参考手册－RAPID 概述，RAPID 摘要－运动部分
速度的定义	第 1010 页 speeddata—速度数据
Zone 数据的定义	第 1047 页 zonedata—zone 数据
停止点数据的定义	第 1014 页 stoppointdata－停止点数据
工具的定义	第 1031 页 tooldata—工具数据
工作对象的定义	第 1039 页 wobjdata—工作对象数据
写入一个改正入口	第 67 页 CorrWrite 写入一个改正发生器
运动综述	RAPID 参考手册—RAPID 概述，运动和 I/O 原理部分
坐标系	RAPID 参考手册－RAPID 概述，运动和 I/O 原理－坐标系部分

并发的程序执行	RAPID 参考手册—RAPID 概述，运动和 I/O 原理—用逻辑指令同步部分

1.97.MoveLDO－直线移动机器人并且在转角处设置数字输出

用途：

MoveLDO（直线运动数字输出）用来直线移动TCP到指定的目标点。在转角路径的中间位置，指定的数字输出信号被置位/复位。

当TCP仍旧固定的时候，该指令也可以用来给工具重新定向。

该指令只能用在主任务 T_ROB1 中，或者在多运动系统中的运动任务中。

基本范例：

该指令的基本范例说明如下：

例1　MoveLDO p1, v1000, z30, tool2, do1,1;

工具tool2的TCP直线运动到目标位置p1，速度数据v1000和zone数据z30。在p1的转角路径的中间位置，输出信号do1被置位。

项目：

MoveLDO ToPoint [\ID] Speed [\T] Zone Tool [\WObj] Signal Value

ToPoint：

数据类型：robtarget

机器人和外部轴的目标位置。定义为一个命名的位置或者直接存储在指令中（在指令中用*标记）。

[\ID]：

同步 ID

数据类型：identno

如果并列了同步运动，该项目必须使用在多运动系统中，并且不允许在其他任何情况下使用。

指定的 ID 号在所有协同的程序任务中必须相同。该 ID 号保证在 routine 中运动不会混乱。

Speed：

数据类型：speeddata

应用到运动中的速度数据。速度数据定义 TCP、工具重新定向或者外部轴的速度。

[\T]：

时间

数据类型：num

该项目用来指定外部轴运动的总时间，单位秒。它代替相应的速度数据。

Zone：

数据类型：zonedata

运动的 zone 数据。它描述产生的转角路径的大小。

Tool：

数据类型：tooldata

机器人运动时所使用的工具。TCP 就是移动到目标点的那个点。

[\Wobj]：

工作对象

数据类型：wobjdata

指令中机器人位置相关到的工作对象（坐标系）。该项目可以忽略，如果忽略的话，位置相关到世界坐标系。另一方面，如果使用了静态TCP或者并列了外部轴，该项目必须指定。

Signal：

数据类型：signaldo

要改变的数字输出信号的名称。

Value：

数据类型：dionum

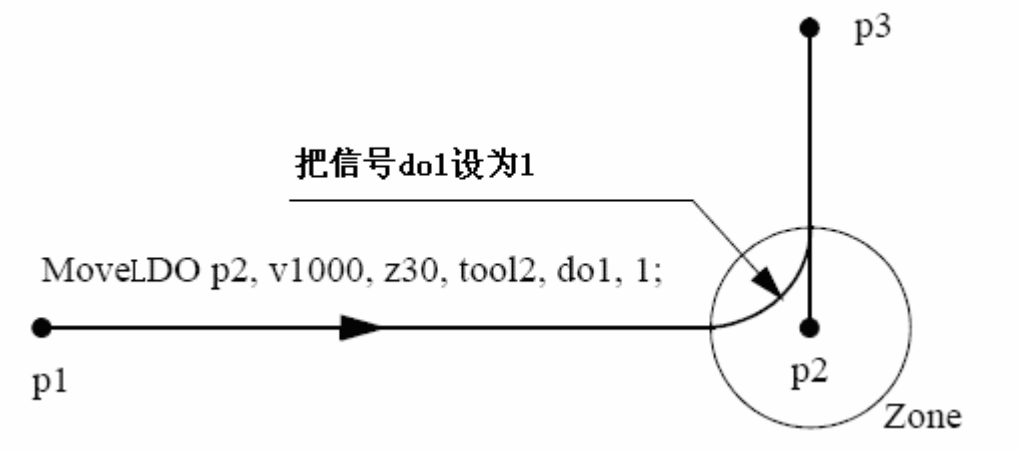
期望的信号数值（0或者1）。

程序执行：

参考指令MoveL，可以得到关节运动的更多信息。

在飞点的转角路径的中间位置，数字输出信号置位/复位，如下图所示。

下图说明在转角路径 MoveLDO 指令的数字输出信号的置位/复位。



对于停止点，我们推荐使用“正常”的编程顺序，即 MoveJ＋SetDO。但是当在指令 MoveLDO 中使用停止

点、当机器人到达停止点的时候，数字输出信号置位/复位。

在执行模式继续逐步向前而不是逐步向后时，指定的 I/O 信号被置位/复位。

语法：

MoveLDO _

[ToPoint ':='] < *robtarget类型的表达式*(**IN**) > ','_

['\' ID ':=' < *identno类型的表达式*(**IN**) >]','_

[Speed ':='] < *speeddata类型的表达式*(**IN**) >_

['\' T ':=' < *num类型的表达式*(**IN**) >] ','_

[Zone ':='] < *zonedata类型的表达式*(**IN**) > ','_

[Tool ':='] < *tooldata类型的恒量*(**PERS**) > _

['\' WObj ':=' < *wobjdata类型的恒量*(**PERS**) >] ','_

[Signal ':='] < *signaldo类型的变量*(**VAR**) >] ','_

[Value ':='] < *dionum类型的表达式*(**IN**) >] ';'

相关信息：

相关信息	**参看**
其他位置指令	RAPID 参考手册－RAPID 概述，RAPID 摘要－运动部分
直线移动机器人	第 236 页 MoveL－直线移动机器人
速度的定义	第 1010 页 speeddata—速度数据
Zone 数据的定义	第 1047 页 zonedata—zone 数据
工具的定义	第 1031 页 tooldata—工具数据
工作对象的定义	第 1039 页 wobjdata—工作对象数据
运动综述	RAPID 参考手册—RAPID 概述，运动和 I/O 原理部分
坐标系	RAPID 参考手册－RAPID 概述，运动和 I/O 原理－坐标系部分
带 I/O 设定的运动	RAPID 参考手册—RAPID 概述，运动和 I/O 原理—用逻辑指令同步部分

1.98.MoveLSync－直线移动机器人并且执行一个 RAPID 程序

用途：

MoveLSync（同步直线移动）用来直线移动 TCP 到给定的目标位置。在目标点的转角路径的中间位置，指定的 RAPID 程序开始运行。

当 TCP 仍旧固定的时候，该指令也可以用来给工具重新定向。

该指令只能用在主任务 T_ROB1，或者多运动系统的运动任务中。

基本范例：

该指令的基本范例说明如下。

例1　MoveLSync p1, v1000, z30, tool2, “proc1”;

工具 tool2 的 TCP 沿线性移动到位置 p1，速度数据 v1000，zone 数据 z30。在 p1 的转角路径的中间位置程序 proc1 开始执行。

项目：

MoveLSync ToPoint [\ID] Speed [\T] Zone Tool [\WObj] ProcName

ToPoint：

数据类型：robtarget

机器人和外部轴的目标点。定义为一个命名的位置或者直接存储在指令中（在指令中用*标记）。

[\ID]：

同步 ID

数据类型：identno

该项目必须使用在多运动系统中，如果并列了同步运动，则不允许在其他任何情况下使用。

指定的 ID 号在所有协同的程序任务中必须相同。该 ID 号保证在 routine 中运动不会混乱。

如果并列了同步运动，不允许在其他任何情况下使用。

Speed：

数据类型：speeddata

应用到运动中的速度数据。速度数据定义 TCP、工具重新定向或者外部轴的速度。

[\T]：

时间

数据类型：num

该项目用来指定外部轴运动的总时间，单位秒。它代替相应的速度数据。

Zone：

数据类型：zonedata

运动的 zone 数据。它描述产生的转角路径的大小。

Tool：

数据类型：tooldata

机器人运动时所使用的工具。TCP 就是移动到目标点的那个点。

[\Wobj]：

工作对象

数据类型：wobjdata

指令中机器人位置相关到的工作对象（坐标系）。该项目可以忽略，如果忽略的话，位置相关到世界坐标系。另一方面，如果使用了静态TCP或者并列了外部轴，该项目必须指定。

ProcName：

程序名称

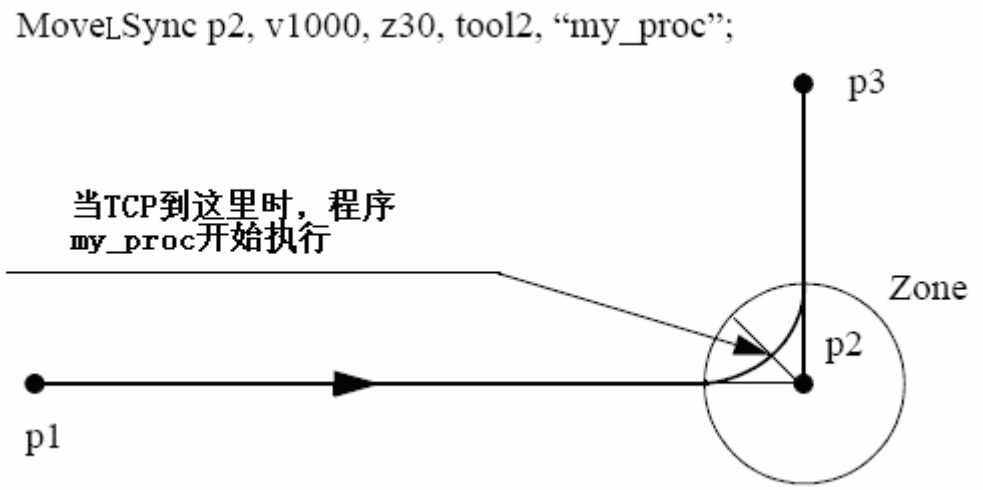
数据类型：string

在目标点的转角路径的中间位置要执行的RAPID程序的名称。

程序执行：

参考指令MoveJ，可以得到关节运动的更多信息。

当TCP到达MoveJSync指令的目标点的转角路径的中间位置时，指定的RAPID程序开始执行，如下图所示。



对于停止点，我们推荐使用“正常”的编程顺序，即 MoveL＋其他 RAPID 指令。

下表描述了在不同执行模式下指定的 RAPID 程序的执行：

执行模式	RAPID 程序的执行
继续或者循环	按照该描述
逐步向前	在停止点
逐步向后	一点也不执行

限制：

当程序停止后，从连续执行或循环执行切换到逐步向前或者向后将导致错误。该错误告诉用户模式切换将导致路径上的执行队列的 RAPID 程序的执行错误。

指令 MoveLSync 不能用在 TRAP 层次上。指定的 RAPID 程序不能用逐步执行测试。

语法：

MoveLSync

[ToPoint ':='] < robtarget类型的表达式(IN) > ','

['\' ID ':=' < identno类型的表达式(IN)>]','

[Speed ':='] < speeddata类型的表达式(IN) >

['\' T ':=' < num类型的表达式(IN) >] ','

[Zone ':='] < zonedata类型的表达式(IN) >

[Tool ':='] < tooldata类型的恒量(PERS) >

['\' WObj ':=' < wobjdata类型的恒量(PERS)>] ','

[ProcName':='] <string类型的表达式(IN) >] ';'

相关信息：

相关信息	参看
其他位置指令	RAPID 参考手册－RAPID 概述，RAPID 摘要－运动部分
直线移动机器人	第 236 页 MoveL－直线移动机器人
速度的定义	第 1010 页 speeddata—速度数据
Zone 数据的定义	第 1047 页 zonedata—zone 数据
工具的定义	第 1031 页 tooldata—工具数据
工作对象的定义	第 1039 页 wobjdata—工作对象数据
运动综述	RAPID 参考手册—RAPID 概述，运动和 I/O 原理部分
坐标系	RAPID 参考手册－RAPID 概述，运动和 I/O 原理－坐标系部分

World Zone：

最多可以在机器人的工作区域内定义 10 个不同的体积空间。他们可以用来：

- 指出机器人的 TCP 是工作区域中的一个明确的部分。
- 限制机器人的工作区域，阻止和工具的碰撞。
- 创建一个由两个机器人公用的区域，该区域在同一时间内只能由一个机器人使用。

1.227．WZBoxDef—定义一个箱体形状的 World Zone

用途：

WZBoxDef（World Zone 箱体定义）用来定义一个直立箱体形状的 World Zone，该箱体的所有边都和 World 坐标系的坐标轴平行。

基本范例：

该指令的基本范例说明如下：

例1

```
VAR shapedata volume;
CONST pos corner1:=[200, 100, 100];
CONST pos corner2 :=[600, 400, 400];
…
WZBoxDef \Inside, volume, corner1, corner2;
```

定义一个直立的箱体，该箱体的所有边都和 World 坐标系的轴平行，该箱体由两个对角点 corner1 和 corner2 定义。

项目：

```
WZBoxDef [\Inside] | [\Outside] Shape LowPoint HighPoint
```

[\Inside]:

数据类型：switch

定义箱体内部的体积

[\OutSide]:

数据类型：switch

定义箱体外部的体积（反体积）。

必须指定\Inside 和\Outside 两个项目中的一个。

Shape：

数据类型：shapedata

定义的体积的存储的变量（系统的私有（private）数据）。

LowPoint：

数据类型：pos

定义箱体的一个较低的角点的位置（x, y, z）以毫米为单位。

HighPoint:

数据类型：pos

定义箱体的另一个相对的角点的位置（x, y, z）以毫米为单位。

程序执行：

箱体的定义存储在 shapedata 类型（Shape 项目）的变量中，用于将来在 WZLimSup 和 WZDOSet 指令中使用。

限制：

LowPoint 和 HighPoint 的位置必须是有效的相对角点（x，y 和 z 的坐标值都不相同）。如果用机器人来指出 LowPoint 和 HighPoint，工作对象（wobj0）必须激活（在 robtarget 中使用 trans 组件，即 p1.trans 作为项目）。

语法：

WZBoxDef

[[‘\’Inside] | [‘\’Outside]’,’

[LowPoint’:=’]<pos 类型的表达式（IN）>’,’

[Shape’:=’]<shapedata 类型的变量（VAR）>’,’

[HighPoint’:=’]<pos 类型的表达式（IN）>’;’

相关信息：

相关信息	参看
World Zones	《RAPID 参考手册—RAPID 概述》运动和 I/O 原理—World Zone 部分
World Zone 形状	第 1004 页 shapedata—World Zone 形状数据部分
定义球形 World Zone	第 636 页 WZSphDef—定义球形 World Zone。
定义圆柱形 World Zone	第 613 页 WZCylDef—定义圆柱形 World Zone。
定义关节 home 位的 World Zone	第 625 页 WZHomeJointDef—定义关节 home 位的 World Zone。
定义关节限位的 World Zone	第 629 页 WZLimJointDef—定义关节限位的 World Zone。
激活 World Zone 限位管理	第 633 页 WZLimSup—激活 World Zone 限位管理。
激活 World Zone 数字输出设置	第 617 页 WZDOSet—激活 World Zone 来设置数字输出。

1.228．WZCylDef—定义一个圆柱形的 World Zone

用途：

WZCylDef（World Zone 圆柱定义）用来定义一个圆柱形状的 World Zone，该圆柱的轴线平行于 World 坐标系的 z 轴。

基本范例：

该指令的基本范例说明如下：

例 1

```
VAR shapedata volume;
CONST pos C2:= [300, 200, 200];
CONST num R2:= 100;
CONST num H2:=200;
```

…

WZCylDef \Inside, volume, C2, R2, H2;

定义一个圆柱，底面圆心为 C2，半径 R2，高度 H2。

项目：

WZCylDef [\Inside] | [\Outside] Shape CenterPoint Radius Height

[\Inside]:

数据类型：switch

定义圆柱内部的体积。

[\Outside]:

数据类型：switch

定义圆柱外部的体积（反体积）。

必须指定两个项目\Inside 和\Outside 中的一个。

Shape:

数据类型：shapedata

用来存储定义的体积的变量（系统的私有（private）数据）。

CentrePoint:

数据类型：pos

定义圆柱的一个底面圆的圆心位置（x，y，z），单位是毫米。

Radius：（半径）

数据类型：num

圆柱的半径，单位是毫米。

Height：

数据类型：num

圆柱的高度，单位是毫米。如果是正的（+z 方向），CentrePoint 项目是圆柱较低底面的圆心（如以上例子）。Height 如果是负的（-z 方向），CentrePoint 项目是圆柱上底面的圆心。

程序执行：

圆柱的定义存储在 shapedata 类型的变量中（项目 Shape），将来在 WZLimSup 或者 WZDOSet 指令中使用。

限制：

如果用机器人指出 CentrePoint，工作对象 wobj0 必须被激活（使用 robtarget 中的 trans 组件，即 p1.trans 作为项目。

语法：

WZCylDef

['\'Inside] | ['\'Outside]','

[Shape':=']<shapedata 类型的变量（VAR）>','

[CentrePoint':=']<pos 类型的表达式（IN）>','

[Radius':=']<num 类型的表达式（IN）>','

[Height':=']<num 类型的表达式（IN）>';'

相关信息：

相关信息	参看
World Zones	《RAPID 参考手册—RAPID 概述》运动和 I/O 原理—World Zone 部分
World Zone 形状	第 1004 页 shapedata—World Zone 形状数据部分
定义球形 World Zone	第 636 页 WZSphDef—定义球形 World Zone。
定义箱体形状的 World Zone	第 611 页 WZBoxDef—定义箱体形状的 World Zone。
定义关节 home 位的 World Zone	第 625 页 WZHomeJointDef—定义关节 home 位的 World Zone。
定义关节限位的 World Zone	第 629 页 WZLimJointDef—定义关节限位的 World Zone。
激活 World Zone 限位管理	第 633 页 WZLimSup—激活 World Zone 限位管理。
激活 World Zone 数字输出设置	第 617 页 WZDOSet—激活 World Zone 来设置数字输出。

1.229．WZDisable—解除临时 World Zone 监视

用途：

WZDisable（解除 World Zone）用来解除对临时 World Zone 的监视，该监视原先用来停止运动或者设置一个输出。

基本范例：

该指令的基本范例说明如下：

例 1　VAR wztemporary wzone;

…

PROC …

WZLimSup \Temp, wzone, volume;

MoveL p_pick, v500, z40, tool1;

WZDisable wzone;

MoveL p_place, v200, z30, tool1;

ENDPROC

当移动到 p_pick 的时候，机器人 TCP 的位置被检测到，这样机器人将不能够进入指定的体积 wzone 内部。

当移动到 p_place 的时候，该监视没有执行。

项目：

WZDisable WorldZone

WorldZone：

数据类型：wztemporary

Wztemporary 类型的变量或者恒量，包含要解除的 WorldZone 的标识符。

程序执行：

临时 WorldZone 被解除。也就是说对机器人 TCP 在相应体积空间内的监视被临时停止。它可以通过 WZEnable 指令被再次激活。

限制：

只有临时 WorldZone 可以被解除。一个静态的 WorldZone 总是激活的。

语法：

WZDisable

[WorldZone':=']<wztemporaty 类型的变量或者恒量（INOUT）>';'

相关信息：

相关信息	参看
World Zones	《RAPID 参考手册—RAPID 概述》运动和 I/O 原理—World Zone 部分
定于圆柱形状 World Zone	第 613 页 WZCylDef—定义圆柱形状的 World Zone。
临时 World Zone 数据	第 1045 页 wztemporary—临时 WorldZone 数据
激活 WorldZone 限位监视	第 633 页 WZLimSup—激活 WorldZone 限位监视
激活 World Zone 数字输出设置	第 617 页 WZDOSet—激活 World Zone 来设置数字输出。
激活 WorldZone	第 621 页 WZEnable—激活临时监视
擦除 WorldZone	第 623 页 WZFree—擦除临时 WorldZone 监视

1.230．WZDOSet—激活 WorldZone 来设置数字输出

用途：

WZDOSet（WorldZone 数字输出设置）用来定义动作并且激活一个 WorldZone 来监视机器人运动。

在该指令执行以后，当机器人的 TCP 或机器人/外部轴（关节中的区域）在定义的 WorldZone 内部或者接近 WorldZone 时，一个数字输出信号被设为一个特定的数值。

基本范例：

该指令的基本范例说明如下：

例 1　　VAR wztemporary service;

```
PROC zone_output( )
    VAR shapedata volume;
    CONST pos p_service:= [500, 500, 700];
    …
    WZSphDef \Inside, volume, p_service, 50;
    WZDOSet \Temp, service \Inside, volume, do_service, 1;
ENDPROC
```

在应用程序中定义临时 WorldZone service，当机器人 TCP 在程序执行过程中或者点动过程中进入定义的球体时，设定信号 do_service。

项目：

WZDOSet [\Temp] | [\Stat] WorldZone [\Inside] | [\Before] Shape Signal SetValue

[\Temp]：

临时的

数据类型：switch

要定义的 WorldZone 是一个临时的 WorldZone。

[\Stat]：

静态的

数据类型：switch

要定义的 WorldZone 是一个静态的 WorldZone。

必须指定[\Temp]和[\Stat]两个项目中的一个。

WorldZone：

数据类型：wztemporary 或者 wzstationary

可以根据 WorldZone 的特性（数字数值）进行更新的变量或者恒量。

如果使用可选项目\Temp，数据类型必须是 wztemporary。如果使用了\Stat，数据类型必须是 wzstationary。

[\Inside]：

数据类型：switch

当机器人的 TCP 或者某一个轴进入定义的体积空间内的时候，将设定数字输出信号。

[\Before]：

数据类型：switch

当机器人的 TCP 或者某一个轴进入定义的体积空间之前（马上就要进入空间），将设定数字输出信号。

两个项目[\Inside]和[\Before]必须选定一个。

Shape：

数据类型：shapedata

定义 WorldZone 空间的变量。

Signal：

数据类型：signaldo

将要改变的数字输出信号的名称。

如果使用了静态 WorldZone，信号必须写保护，防止用户进入（RAPID，FP 示教器）。在系统参数或者指定的轴上设定用户进入等级。

SetValue:

数据类型：dionum

当机器人 TCP 进入体积空间或者恰好在进入之前，期望的信号输出的数值（1 或者 0）。

在机器人 TCP 在外面或者正好在空间外面，信号输出为相反的数值。

程序执行：

定义的 WorldZone 被激活。从这时开始，机器人 TCP 位置（或者机器人/外部轴位置）将被监视，当机器人 TCP 位置（或者机器人/外部轴位置）在空间内（\Inside）或者接近空间的边界（\Before），将被设置输出。

如果和 WZDOSet 同时使用了 WZHomeJointDef 或者 WZLimJointDef 指令，只有在带空间监视的所有激活的轴即将进入或者已经进入关节空间时，才能够设置数字输出信号。

更多范例：

有关该指令如何使用的更多范例说明如下：

例 1

```
VAR wztemporary home;
VAR wztemporary service;
PERS wztemporary equip1:=[0];

PROC main( )
  …
  ! 定义所有临时的 WorldZone
  Zone_output;
  …
  ! equip1 在机器人工作区域
  WZEnable equip1;
```

```
    …
    ! equip1 在机器人工作区域之外
    WZDisable equip1;
    …
    ! 不再使用 equip1
    WZFree equip1;
    …
ENDPROC

PROC zone_output( )
    VAR shapedata volume;
    CONST pos p_home:=[800, 0, 800];
    CONST pos p_service:=[800, 800, 800];
    CONST pos p_equip1:=[-800,-800, 0];
    …
    WZSphDef \Inside, volume, p_home, 50;
    WZDOSet \Temp, home \Inside, volume, do_home, 1;
    WZSphDef |Inside, volume, p_service, 50;
    WZDOSet \Temp, service \Inside, volume, do_service, 1;
    WZCylDef \Inside, volume, p_equip1, 300, 1000;
    WZLimSup \Temp, equip1, volume;
    ! equip1 不在机器人工作区域。
    WZDisable equip1;
ENDPROC
```

在应用程序中定义临时 WorldZone home 和 service， 当机器人在程序执行或者点动过程中分别进入球体 home 或者 service 时，这两个 WorldZone 用来设定信号 do_home 和 do_service。

同时，定义一个临时 WorldZone quip1，equip1 只有在机器人程序中、当 equip1 在机器人工作区域以内的时候才会被激活。这时候，无论在程序执行或者手动的时候，机器人在进入 equip1 区域之前停止。通过使用恒量 equip1 的数值，equip1 可以从其它程序任务中使能或者解除。

限制：

WorldZone 不能通过使用项目 WorldZone 中的相同的变量重复定义。

Grip3的TCP沿直线运动到停止点p5。当停止点fine的50%的位置条件和50%的速度条件满足的时候，机器人认为它到达了目标点。它等条件满足最多等两秒，参看stoppointdata数据类型的预定义数据inpos50。

例3 **MoveL \Conc, *, v2000, z40, grip3;**

Grip3的TCP直线运动到存储在指令中的位置。当机器人移动的时候，后续的逻辑指令开始执行。

例4 **MoveL start, v2000, z40, grip3 \WObj:=fixture;**

Grip3的TCP直线运动到位置start，位置在fixture的对象坐标系统中指定。

语法：

MoveL _

['\' Conc ',']_

[ToPoint ':='] < *robtarget类型的表达式*(**IN**) > ','

['\' ID ':=' < *identno类型的表达式*(**IN**) >]','_

[Speed ':='] < *speeddata类型的表达式*(**IN**) >_

['\' V ':=' < *num类型的表达式*(**IN**) >] _

| ['\' T ':=' < *num类型的表达式*(**IN**) >] ','_

[Zone ':='] < *zonedata类型的表达式*(**IN**) >_

['\' Z ':=' < *num类型的表达式*(**IN**) >]_

['\' Inpos ':=' < *stoppointdata类型的表达式*(**IN**) >] ','_

[Tool ':='] < *tooldata类型的恒量*(**PERS**) > _

['\' WObj ':=' < *wobjdata 类型的恒量*(**PERS**)>] _

['\' Corr]';'

相关信息：

相关信息	**参看**
其他位置指令	RAPID 参考手册－RAPID 概述，RAPID 摘要－运动部分
速度的定义	第 1010 页 speeddata—速度数据
Zone 数据的定义	第 1047 页 zonedata—zone 数据
停止点数据的定义	第 1014 页 stoppointdata－停止点数据
工具的定义	第 1031 页 tooldata—工具数据
工作对象的定义	第 1039 页 wobjdata—工作对象数据
写入一个改正入口	第 67 页 CorrWrite 写入一个改正发生器
运动综述	RAPID 参考手册—RAPID 概述，运动和 I/O 原理部分
坐标系	RAPID 参考手册－RAPID 概述，运动和 I/O 原理－坐标系部分

并发的程序执行	RAPID 参考手册—RAPID 概述，运动和 I/O 原理—用逻辑指令同步部分

1.97.MoveLDO－直线移动机器人并且在转角处设置数字输出

用途：

MoveLDO（直线运动数字输出）用来直线移动TCP到指定的目标点。在转角路径的中间位置，指定的数字输出信号被置位/复位。

当TCP仍旧固定的时候，该指令也可以用来给工具重新定向。

该指令只能用在主任务 T_ROB1 中，或者在多运动系统中的运动任务中。

基本范例：

该指令的基本范例说明如下：

例1　MoveLDO p1, v1000, z30, tool2, do1,1;

工具tool2的TCP直线运动到目标位置p1，速度数据v1000和zone数据z30。在p1的转角路径的中间位置，输出信号do1被置位。

项目：

MoveLDO ToPoint [\ID] Speed [\T] Zone Tool [\WObj] Signal Value

ToPoint：

数据类型：robtarget

机器人和外部轴的目标位置。定义为一个命名的位置或者直接存储在指令中（在指令中用*标记）。

[\ID]：

同步 ID

数据类型：identno

如果并列了同步运动，该项目必须使用在多运动系统中，并且不允许在其他任何情况下使用。

指定的 ID 号在所有协同的程序任务中必须相同。该 ID 号保证在 routine 中运动不会混乱。

Speed：

数据类型：speeddata

应用到运动中的速度数据。速度数据定义 TCP、工具重新定向或者外部轴的速度。

[\T]：

时间

数据类型：num

该项目用来指定外部轴运动的总时间，单位秒。它代替相应的速度数据。

Zone：

数据类型：zonedata

运动的 zone 数据。它描述产生的转角路径的大小。

Tool：

数据类型：tooldata

机器人运动时所使用的工具。TCP 就是移动到目标点的那个点。

[\Wobj]：

工作对象

数据类型：wobjdata

指令中机器人位置相关到的工作对象（坐标系）。该项目可以忽略，如果忽略的话，位置相关到世界坐标系。另一方面，如果使用了静态TCP或者并列了外部轴，该项目必须指定。

Signal：

数据类型：signaldo

要改变的数字输出信号的名称。

Value：

数据类型：dionum

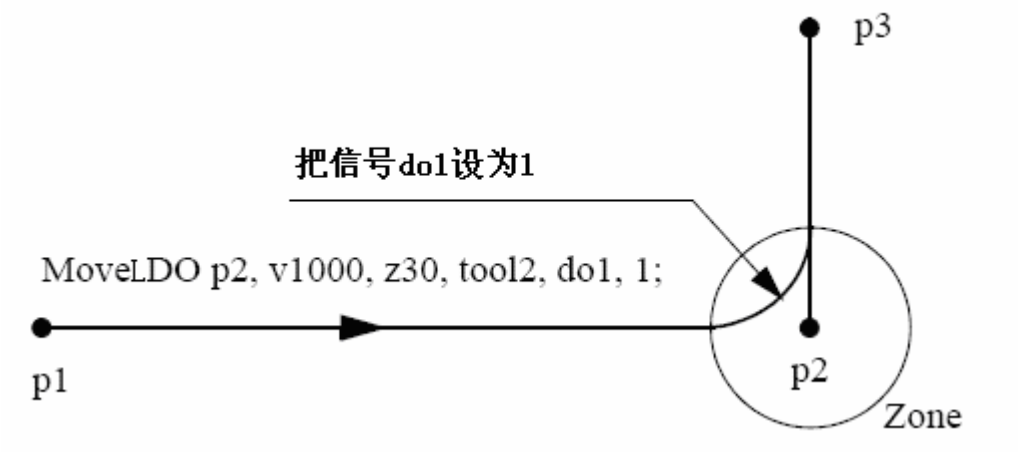
期望的信号数值（0或者1）。

程序执行：

参考指令MoveL，可以得到关节运动的更多信息。

在飞点的转角路径的中间位置，数字输出信号置位/复位，如下图所示。

下图说明在转角路径 MoveLDO 指令的数字输出信号的置位/复位。



对于停止点，我们推荐使用“正常”的编程顺序，即 MoveJ＋SetDO。但是当在指令 MoveLDO 中使用停止

点、当机器人到达停止点的时候，数字输出信号置位/复位。

在执行模式继续逐步向前而不是逐步向后时，指定的 I/O 信号被置位/复位。

语法：

MoveLDO _

[ToPoint ':='] < *robtarget类型的表达式*(**IN**) > ','_

['\' ID ':=' < *identno类型的表达式*(**IN**) >]','_

[Speed ':='] < *speeddata类型的表达式*(**IN**) >_

['\' T ':=' < *num类型的表达式*(**IN**) >] ','_

[Zone ':='] < *zonedata类型的表达式*(**IN**) > ','_

[Tool ':='] < *tooldata类型的恒量*(**PERS**) > _

['\' WObj ':=' < *wobjdata类型的恒量*(**PERS**) >] ','_

[Signal ':='] < *signaldo类型的变量*(**VAR**) >] ','_

[Value ':='] < *dionum类型的表达式*(**IN**) >] ';'

相关信息：

相关信息	**参看**
其他位置指令	RAPID 参考手册－RAPID 概述，RAPID 摘要－运动部分
直线移动机器人	第 236 页 MoveL－直线移动机器人
速度的定义	第 1010 页 speeddata—速度数据
Zone 数据的定义	第 1047 页 zonedata—zone 数据
工具的定义	第 1031 页 tooldata—工具数据
工作对象的定义	第 1039 页 wobjdata—工作对象数据
运动综述	RAPID 参考手册—RAPID 概述，运动和 I/O 原理部分
坐标系	RAPID 参考手册－RAPID 概述，运动和 I/O 原理－坐标系部分
带 I/O 设定的运动	RAPID 参考手册—RAPID 概述，运动和 I/O 原理—用逻辑指令同步部分

1.98.MoveLSync－直线移动机器人并且执行一个 RAPID 程序

用途：

MoveLSync（同步直线移动）用来直线移动 TCP 到给定的目标位置。在目标点的转角路径的中间位置，指定的 RAPID 程序开始运行。

当 TCP 仍旧固定的时候，该指令也可以用来给工具重新定向。

该指令只能用在主任务 T_ROB1，或者多运动系统的运动任务中。

基本范例：

该指令的基本范例说明如下。

例1　MoveLSync p1, v1000, z30, tool2, “proc1”;

工具 tool2 的 TCP 沿线性移动到位置 p1，速度数据 v1000，zone 数据 z30。在 p1 的转角路径的中间位置程序 proc1 开始执行。

项目：

MoveLSync ToPoint [\ID] Speed [\T] Zone Tool [\WObj] ProcName

ToPoint：

数据类型：robtarget

机器人和外部轴的目标点。定义为一个命名的位置或者直接存储在指令中（在指令中用*标记）。

[\ID]：

同步 ID

数据类型：identno

该项目必须使用在多运动系统中，如果并列了同步运动，则不允许在其他任何情况下使用。

指定的 ID 号在所有协同的程序任务中必须相同。该 ID 号保证在 routine 中运动不会混乱。

如果并列了同步运动，不允许在其他任何情况下使用。

Speed：

数据类型：speeddata

应用到运动中的速度数据。速度数据定义 TCP、工具重新定向或者外部轴的速度。

[\T]：

时间

数据类型：num

该项目用来指定外部轴运动的总时间，单位秒。它代替相应的速度数据。

Zone：

数据类型：zonedata

运动的 zone 数据。它描述产生的转角路径的大小。

Tool：

数据类型：tooldata

机器人运动时所使用的工具。TCP 就是移动到目标点的那个点。

[\Wobj]：

工作对象

数据类型：wobjdata

指令中机器人位置相关到的工作对象（坐标系）。该项目可以忽略，如果忽略的话，位置相关到世界坐标系。另一方面，如果使用了静态TCP或者并列了外部轴，该项目必须指定。

ProcName：

程序名称

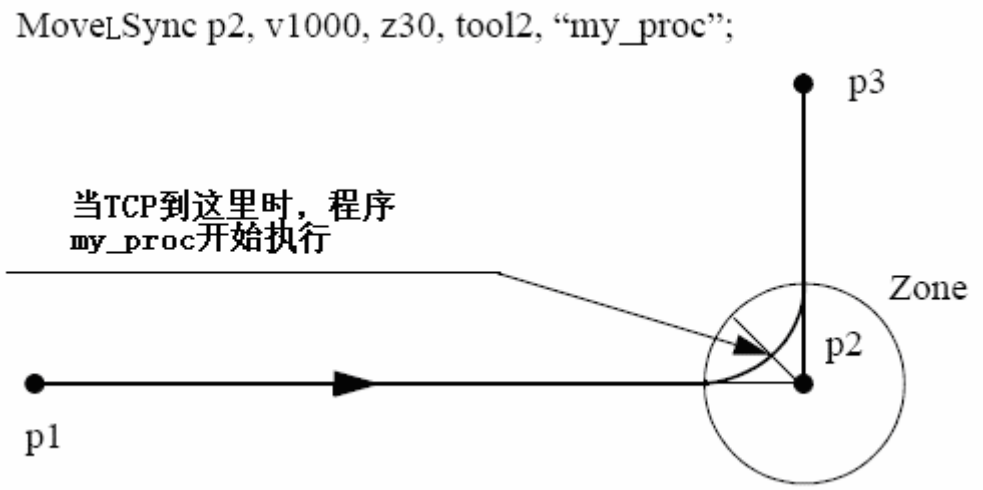
数据类型：string

在目标点的转角路径的中间位置要执行的RAPID程序的名称。

程序执行：

参考指令MoveJ，可以得到关节运动的更多信息。

当TCP到达MoveJSync指令的目标点的转角路径的中间位置时，指定的RAPID程序开始执行，如下图所示。



对于停止点，我们推荐使用“正常”的编程顺序，即 MoveL＋其他 RAPID 指令。

下表描述了在不同执行模式下指定的 RAPID 程序的执行：

执行模式	RAPID 程序的执行
继续或者循环	按照该描述
逐步向前	在停止点
逐步向后	一点也不执行

限制：

当程序停止后，从连续执行或循环执行切换到逐步向前或者向后将导致错误。该错误告诉用户模式切换将导致路径上的执行队列的 RAPID 程序的执行错误。

指令 MoveLSync 不能用在 TRAP 层次上。指定的 RAPID 程序不能用逐步执行测试。

语法：

MoveLSync

[ToPoint ':='] < robtarget类型的表达式(IN) > ','

['\' ID ':=' < identno类型的表达式(IN)>]','

[Speed ':='] < speeddata类型的表达式(IN) >

['\' T ':=' < num类型的表达式(IN) >] ','

[Zone ':='] < zonedata类型的表达式(IN) >

[Tool ':='] < tooldata类型的恒量(PERS) >

['\' WObj ':=' < wobjdata类型的恒量(PERS)>] ','

[ProcName':='] <string类型的表达式(IN) >] ';'

相关信息：

相关信息	参看
其他位置指令	RAPID 参考手册－RAPID 概述，RAPID 摘要－运动部分
直线移动机器人	第 236 页 MoveL－直线移动机器人
速度的定义	第 1010 页 speeddata—速度数据
Zone 数据的定义	第 1047 页 zonedata—zone 数据
工具的定义	第 1031 页 tooldata—工具数据
工作对象的定义	第 1039 页 wobjdata—工作对象数据
运动综述	RAPID 参考手册—RAPID 概述，运动和 I/O 原理部分
坐标系	RAPID 参考手册－RAPID 概述，运动和 I/O 原理－坐标系部分

World Zone：

最多可以在机器人的工作区域内定义 10 个不同的体积空间。他们可以用来：

- 指出机器人的 TCP 是工作区域中的一个明确的部分。
- 限制机器人的工作区域，阻止和工具的碰撞。
- 创建一个由两个机器人公用的区域，该区域在同一时间内只能由一个机器人使用。

1.227．WZBoxDef—定义一个箱体形状的 World Zone

用途：

WZBoxDef（World Zone 箱体定义）用来定义一个直立箱体形状的 World Zone，该箱体的所有边都和 World 坐标系的坐标轴平行。

基本范例：

该指令的基本范例说明如下：

例1

```
VAR shapedata volume;
CONST pos corner1:=[200, 100, 100];
CONST pos corner2 :=[600, 400, 400];
…
WZBoxDef \Inside, volume, corner1, corner2;
```

定义一个直立的箱体，该箱体的所有边都和 World 坐标系的轴平行，该箱体由两个对角点 corner1 和 corner2 定义。

项目：

```
WZBoxDef [\Inside] | [\Outside] Shape LowPoint HighPoint
```

[\Inside]:

数据类型：switch

定义箱体内部的体积

[\OutSide]:

数据类型：switch

定义箱体外部的体积（反体积）。

必须指定\Inside 和\Outside 两个项目中的一个。

Shape：

数据类型：shapedata

定义的体积的存储的变量（系统的私有（private）数据）。

LowPoint：

数据类型：pos

定义箱体的一个较低的角点的位置（x, y, z）以毫米为单位。

HighPoint:

数据类型：pos

定义箱体的另一个相对的角点的位置（x, y, z）以毫米为单位。

程序执行：

箱体的定义存储在 shapedata 类型（Shape 项目）的变量中，用于将来在 WZLimSup 和 WZDOSet 指令中使用。

限制：

LowPoint 和 HighPoint 的位置必须是有效的相对角点（x，y 和 z 的坐标值都不相同）。如果用机器人来指出 LowPoint 和 HighPoint，工作对象（wobj0）必须激活（在 robtarget 中使用 trans 组件，即 p1.trans 作为项目）。

语法：

WZBoxDef

[[‘\’Inside] | [‘\’Outside]’,’

[LowPoint’:=’]<pos 类型的表达式（IN）>’,’

[Shape’:=’]<shapedata 类型的变量（VAR）>’,’

[HighPoint’:=’]<pos 类型的表达式（IN）>’;’

相关信息：

相关信息	参看
World Zones	《RAPID 参考手册—RAPID 概述》运动和 I/O 原理—World Zone 部分
World Zone 形状	第 1004 页 shapedata—World Zone 形状数据部分
定义球形 World Zone	第 636 页 WZSphDef—定义球形 World Zone。
定义圆柱形 World Zone	第 613 页 WZCylDef—定义圆柱形 World Zone。
定义关节 home 位的 World Zone	第 625 页 WZHomeJointDef—定义关节 home 位的 World Zone。
定义关节限位的 World Zone	第 629 页 WZLimJointDef—定义关节限位的 World Zone。
激活 World Zone 限位管理	第 633 页 WZLimSup—激活 World Zone 限位管理。
激活 World Zone 数字输出设置	第 617 页 WZDOSet—激活 World Zone 来设置数字输出。

1.228．WZCylDef—定义一个圆柱形的 World Zone

用途：

WZCylDef（World Zone 圆柱定义）用来定义一个圆柱形状的 World Zone，该圆柱的轴线平行于 World 坐标系的 z 轴。

基本范例：

该指令的基本范例说明如下：

例 1

```
VAR shapedata volume;
CONST pos C2:= [300, 200, 200];
CONST num R2:= 100;
CONST num H2:=200;
```

…

WZCylDef \Inside, volume, C2, R2, H2;

定义一个圆柱，底面圆心为 C2，半径 R2，高度 H2。

项目：

WZCylDef [\Inside] | [\Outside] Shape CenterPoint Radius Height

[\Inside]:

数据类型：switch

定义圆柱内部的体积。

[\Outside]:

数据类型：switch

定义圆柱外部的体积（反体积）。

必须指定两个项目\Inside 和\Outside 中的一个。

Shape:

数据类型：shapedata

用来存储定义的体积的变量（系统的私有（private）数据）。

CentrePoint:

数据类型：pos

定义圆柱的一个底面圆的圆心位置（x，y，z），单位是毫米。

Radius：（半径）

数据类型：num

圆柱的半径，单位是毫米。

Height：

数据类型：num

圆柱的高度，单位是毫米。如果是正的（+z 方向），CentrePoint 项目是圆柱较低底面的圆心（如以上例子）。Height 如果是负的（-z 方向），CentrePoint 项目是圆柱上底面的圆心。

程序执行：

圆柱的定义存储在 shapedata 类型的变量中（项目 Shape），将来在 WZLimSup 或者 WZDOSet 指令中使用。

限制：

如果用机器人指出 CentrePoint，工作对象 wobj0 必须被激活（使用 robtarget 中的 trans 组件，即 p1.trans 作为项目。

语法：

WZCylDef

['\'Inside] | ['\'Outside]','

[Shape':=']<shapedata 类型的变量（VAR）>','

[CentrePoint':=']<pos 类型的表达式（IN）>','

[Radius':=']<num 类型的表达式（IN）>','

[Height':=']<num 类型的表达式（IN）>';'

相关信息：

相关信息	参看
World Zones	《RAPID 参考手册—RAPID 概述》运动和 I/O 原理—World Zone 部分
World Zone 形状	第 1004 页 shapedata—World Zone 形状数据部分
定义球形 World Zone	第 636 页 WZSphDef—定义球形 World Zone。
定义箱体形状的 World Zone	第 611 页 WZBoxDef—定义箱体形状的 World Zone。
定义关节 home 位的 World Zone	第 625 页 WZHomeJointDef—定义关节 home 位的 World Zone。
定义关节限位的 World Zone	第 629 页 WZLimJointDef—定义关节限位的 World Zone。
激活 World Zone 限位管理	第 633 页 WZLimSup—激活 World Zone 限位管理。
激活 World Zone 数字输出设置	第 617 页 WZDOSet—激活 World Zone 来设置数字输出。

1.229．WZDisable—解除临时 World Zone 监视

用途：

WZDisable（解除 World Zone）用来解除对临时 World Zone 的监视，该监视原先用来停止运动或者设置一个输出。

基本范例：

该指令的基本范例说明如下：

例 1　VAR wztemporary wzone;

…

PROC …

WZLimSup \Temp, wzone, volume;

MoveL p_pick, v500, z40, tool1;

WZDisable wzone;

MoveL p_place, v200, z30, tool1;

ENDPROC

当移动到 p_pick 的时候，机器人 TCP 的位置被检测到，这样机器人将不能够进入指定的体积 wzone 内部。

当移动到 p_place 的时候，该监视没有执行。

项目：

WZDisable WorldZone

WorldZone：

数据类型：wztemporary

Wztemporary 类型的变量或者恒量，包含要解除的 WorldZone 的标识符。

程序执行：

临时 WorldZone 被解除。也就是说对机器人 TCP 在相应体积空间内的监视被临时停止。它可以通过 WZEnable 指令被再次激活。

限制：

只有临时 WorldZone 可以被解除。一个静态的 WorldZone 总是激活的。

语法：

WZDisable

[WorldZone':=']<wztemporaty 类型的变量或者恒量（INOUT）>';'

相关信息：

相关信息	参看
World Zones	《RAPID 参考手册—RAPID 概述》运动和 I/O 原理—World Zone 部分
定于圆柱形状 World Zone	第 613 页 WZCylDef—定义圆柱形状的 World Zone。
临时 World Zone 数据	第 1045 页 wztemporary—临时 WorldZone 数据
激活 WorldZone 限位监视	第 633 页 WZLimSup—激活 WorldZone 限位监视
激活 World Zone 数字输出设置	第 617 页 WZDOSet—激活 World Zone 来设置数字输出。
激活 WorldZone	第 621 页 WZEnable—激活临时监视
擦除 WorldZone	第 623 页 WZFree—擦除临时 WorldZone 监视

1.230．WZDOSet—激活 WorldZone 来设置数字输出

用途：

WZDOSet（WorldZone 数字输出设置）用来定义动作并且激活一个 WorldZone 来监视机器人运动。

在该指令执行以后，当机器人的 TCP 或机器人/外部轴（关节中的区域）在定义的 WorldZone 内部或者接近 WorldZone 时，一个数字输出信号被设为一个特定的数值。

基本范例：

该指令的基本范例说明如下：

例 1　　VAR wztemporary service;

```
PROC zone_output( )
    VAR shapedata volume;
    CONST pos p_service:= [500, 500, 700];
    …
    WZSphDef \Inside, volume, p_service, 50;
    WZDOSet \Temp, service \Inside, volume, do_service, 1;
ENDPROC
```

在应用程序中定义临时 WorldZone service，当机器人 TCP 在程序执行过程中或者点动过程中进入定义的球体时，设定信号 do_service。

项目：

WZDOSet [\Temp] | [\Stat] WorldZone [\Inside] | [\Before] Shape Signal SetValue

[\Temp]：

临时的

数据类型：switch

要定义的 WorldZone 是一个临时的 WorldZone。

[\Stat]：

静态的

数据类型：switch

要定义的 WorldZone 是一个静态的 WorldZone。

必须指定[\Temp]和[\Stat]两个项目中的一个。

WorldZone：

数据类型：wztemporary 或者 wzstationary

可以根据 WorldZone 的特性（数字数值）进行更新的变量或者恒量。

如果使用可选项目\Temp，数据类型必须是 wztemporary。如果使用了\Stat，数据类型必须是 wzstationary。

[\Inside]：

数据类型：switch

当机器人的 TCP 或者某一个轴进入定义的体积空间内的时候，将设定数字输出信号。

[\Before]：

数据类型：switch

当机器人的 TCP 或者某一个轴进入定义的体积空间之前（马上就要进入空间），将设定数字输出信号。

两个项目[\Inside]和[\Before]必须选定一个。

Shape：

数据类型：shapedata

定义 WorldZone 空间的变量。

Signal：

数据类型：signaldo

将要改变的数字输出信号的名称。

如果使用了静态 WorldZone，信号必须写保护，防止用户进入（RAPID，FP 示教器）。在系统参数或者指定的轴上设定用户进入等级。

SetValue:

数据类型：dionum

当机器人 TCP 进入体积空间或者恰好在进入之前，期望的信号输出的数值（1 或者 0）。

在机器人 TCP 在外面或者正好在空间外面，信号输出为相反的数值。

程序执行：

定义的 WorldZone 被激活。从这时开始，机器人 TCP 位置（或者机器人/外部轴位置）将被监视，当机器人 TCP 位置（或者机器人/外部轴位置）在空间内（\Inside）或者接近空间的边界（\Before），将被设置输出。

如果和 WZDOSet 同时使用了 WZHomeJointDef 或者 WZLimJointDef 指令，只有在带空间监视的所有激活的轴即将进入或者已经进入关节空间时，才能够设置数字输出信号。

更多范例：

有关该指令如何使用的更多范例说明如下：

例 1

```
VAR wztemporary home;
VAR wztemporary service;
PERS wztemporary equip1:=[0];

PROC main( )
  …
  ! 定义所有临时的 WorldZone
  Zone_output;
  …
  ! equip1 在机器人工作区域
  WZEnable equip1;
```

```
    …
    ! equip1 在机器人工作区域之外
    WZDisable equip1;
    …
    ! 不再使用 equip1
    WZFree equip1;
    …
ENDPROC

PROC zone_output( )
    VAR shapedata volume;
    CONST pos p_home:=[800, 0, 800];
    CONST pos p_service:=[800, 800, 800];
    CONST pos p_equip1:=[-800,-800, 0];
    …
    WZSphDef \Inside, volume, p_home, 50;
    WZDOSet \Temp, home \Inside, volume, do_home, 1;
    WZSphDef |Inside, volume, p_service, 50;
    WZDOSet \Temp, service \Inside, volume, do_service, 1;
    WZCylDef \Inside, volume, p_equip1, 300, 1000;
    WZLimSup \Temp, equip1, volume;
    ! equip1 不在机器人工作区域。
    WZDisable equip1;
ENDPROC
```

在应用程序中定义临时 WorldZone home 和 service， 当机器人在程序执行或者点动过程中分别进入球体 home 或者 service 时，这两个 WorldZone 用来设定信号 do_home 和 do_service。

同时，定义一个临时 WorldZone quip1，equip1 只有在机器人程序中、当 equip1 在机器人工作区域以内的时候才会被激活。这时候，无论在程序执行或者手动的时候，机器人在进入 equip1 区域之前停止。通过使用恒量 equip1 的数值，equip1 可以从其它程序任务中使能或者解除。

限制：

WorldZone 不能通过使用项目 WorldZone 中的相同的变量重复定义。

在 RAPID 程序中，静态的 WorldZone 不能被解除并再次激活，或者进行擦除。

在 RAPID 程序中，临时的 WorldZone 可以被解除（WZDisable），再次激活（WZEnable）或者擦除（WZFree）。

语法：

WZDOSet

[[‘\’Temp] | [‘\’Stat]’,’]

[WorldZone’:=’]<wztemporary 类型的变量或者恒量（INOUT）>

[‘\’Inside] | [‘\’Before] ‘,’

[Shape’:=’]<shapedata 类型的变量（VAR）>’,’

[Signal’:=’]<signaldo 类型的变量（VAR）>’,’

[SetValue’:=’]<dionum 类型的表达式（IN）>’;’

相关信息：

相关信息	参看
World Zones	《RAPID 参考手册—RAPID 概述》运动和 I/O 原理—World Zone 部分
World Zone 形状	第 1004 页 shapedata—World Zone 形状数据部分
临时 WorldZone	第 1045 页 wztemporary—临时 WorldZone 数据
静态 WorldZone	第 1043 页 wzstationary—静态 WorldZone 数据
定义球形 World Zone	第 636 页 WZSphDef—定义球形 World Zone。
定义箱体形状的 World Zone	第 611 页 WZBoxDef—定义箱体形状的 World Zone。
定义圆柱形状 World Zone	第 613 页 WZCylDef—定义圆柱形状的 World Zone。
定义关节 home 位的 World Zone	第 625 页 WZHomeJointDef—定义关节 home 位的 World Zone。
激活 WorldZone 限位监视	第 633 页 WZLimSup—激活 WorldZone 限位监视
信号进入水平	《技术参考手册—系统参数》I/O 主题—信号类型—进入等级部分

1.231．WZEnable—激活临时 WorldZone 监视

用途：

WZEnable（WorldZone 使能）用来重新激活对临时 WorldZone 的监视，该 WorldZone 之前定义用来停止运动或者设定输出。

基本范例：

该指令的基本范例说明如下：

例 1　VAR wztemporary wzone;

…

PROC …

```
WZLimSup \Temp, wzone, volume;
MoveL p_pick, v500, z40, tool1;
WZDisable wzone;
MoveL p_place, v200, z30, tool1;
WZEnable wzone;
MoveL p_home, v200, z30, tool1;
ENDPROC
```

当往 p_pick 移动的时候，检测到机器人 TCP 的位置，这样它就不能进入特定的空间 wzone。当进入 p_place 的时候没有进行这个监视，但是在移动到 p_home 的时候重新激活。

项目：

WZEnable WorldZone

WorldZone：

数据类型：wztemporary

wztemporary 类型的变量或者恒量，包含要激活的 WorldZone 的标识符。

程序执行：

临时 WorldZone 被再次激活。注意，当创建 WorldZone 的时候，它是自动激活的。如果被 WZDisable 指令解除后，它只是需要重新激活。

限制：

只有临时的 WorldZone 可以被解除或者再激活，静态的 WorldZone 总是激活的。

语法：

WZEnable

[WorldZone':=']<wztemporary 类型的变量或者恒量（INOUT）>';'

相关信息：

相关信息	参看
World Zones	《RAPID 参考手册—RAPID 概述》运动和 I/O 原理—World Zone 部分
World Zone 形状	第 1004 页 shapedata—World Zone 形状数据部分
临时 WorldZone 数据	第 1045 页 wztemporary—临时 WorldZone 数据
激活 WorldZone 限位监视	第 633 页 WZLimSup—激活 WorldZone 限位监视
激活 World Zone 数字输出设置	第 617 页 WZDOSet—激活 World Zone 来设置数字输出。
解除 WorldZone	第 615 页 WZDisable—解除临时 WorldZone
擦除 WorldZone	第 623 页 WZFree—擦除临时 WorldZone 监视

1.232．WZFree—擦除临时 WorldZone 监视

用途：

WZFree（WorldZone 释放）用来擦除临时 WorldZone 的定义，之前定义用来停止运动或者设定输出的。

基本范例：

该指令的基本范例说明如下：

例 1

```
VAR wztemporary wzone;
…
PROC …
    WZLimSup \Temp, wzone, volume;
    MoveL p_pick, v500, z40, tool1;
    WZDisable wzone;
    MoveL p_Place, v200, z30, tool1;
    WZEnable wzone;
    MoveL p_home, v200, z30, tool1;
    WZFree wzone;
ENDPROC
```

当往 p_pick 移动的时候，检测到机器人 TCP 的位置，所以它不能够进入指定的 wzone 空间。当往 p_place 移动的时候，该监视没有执行，但是在往 p_home 移动之前重新激活该监视。到达 p_home 之后，WorldZone 的定义被擦除。

项目：

WZFree WorldZone

WorldZone：

数据类型：wztemporary

wztemporary 类型的变量或者恒量，包含要擦除的 world zone 的标识符。

程序执行：

临时 world zone 首先被解除，然后它的定义被擦除。一旦擦除，临时 world zone 不能被重新激活也不能被解除。

限制：

只有临时 world zone 可以被解除、重新激活或者擦除，静态 world zone 通常都是激活的。

语法：

WZFree [WorldZone':=']<wztemporary 类型的变量或者恒量（INOUT）>';'

相关信息：

相关信息	参看
World Zones	《RAPID 参考手册—RAPID 概述》运动和 I/O 原理—World Zone 部分
World Zone 形状	第 1004 页 shapedata—World Zone 形状数据部分
临时 WorldZone 数据	第 1045 页 wztemporary—临时 WorldZone 数据
激活 WorldZone 限位监视	第 633 页 WZLimSup—激活 WorldZone 限位监视
激活 World Zone 数字输出设置	第 617 页 WZDOSet—激活 World Zone 来设置数字输出。
解除 WorldZone	第 615 页 WZDisable—解除临时 WorldZone
激活 WorldZone	第 621 页 WZEnable—激活临时 WorldZone 监视

1.233．WZHomeJointDef—定义关节 home 位的 WorldZone

用途：

WZHomeJointDef（World Zone home 关节定义）用来为机器人和外部轴在关节坐标中定义一个 world zone，用来作为一个 Home 或者服务位置。

基本范例：

该指令的基本范例说明如下：

例 1

```
VAR wzstationary home;
…
PROC power_on( )
    VAR shapedata joint_space;
    CONST jointtarget home_pos := [ [ 0, 0, 0, 0, 0, -45], [0, 9E9, 9E9, 9E9, 9E9, 9E9] ];
    CONST jointtarget delta_pos := [ [2, 2, 2, 2, 2, 2], [5, 9E9, 9E9, 9E9, 9E9, 9E9] ];
    …
    WZHomeJointDef \Stat, home \Inside, joint_space, do_home, 1;
ENDPROC
```

定义并激活一个静态 world zone home，当所有机器人轴和外部轴 extax.eax_a 在程序执行或者点动过程中处于关节位置 home_pos（每一个轴在+/-delta_pos 范围内），home 把信号 do_home 设为 1。Shapedata 类型的数据变量 joint_space 用来从指令 WZHomeJointDef 指令传送到 WZDOSet 指令。

项目：

WZHomeJointDef [\Inside] | [\Outside] Shape MiddleJointVal DeltaJointVal

[\Inside]：

数据类型：switch

定义 MiddleJointVal +/- DeltaJointVal 范围内的关节空间。

[\Outside]：

数据类型：switch

定义 MiddleJointVal +/- DeltaJointVal 范围外的关节空间（反关节空间）。

Shape：

数据类型：shapedata

存储定义的关键空间的变量（系统的私有数据 private）。

MiddleJointVal：

数据类型：jointtarget

关节坐标系中即将定义的关节空间的中心位置。为每一个机器人轴和外部轴指定（对于旋转轴来说单位是度数，对于线形轴来说单位是毫米）。在绝对关节中指定（对于外部轴来说，不是在偏移坐标系 EoffsSet-EoffsOn 中）。对于某些轴来说 9E9 就是说轴不应该去理会。在编程中，不激活的外部轴也会给出 9E9。

DeltaJointVal：

数据类型：jointtarget

在关节坐标系中，从关节空间的中心位置算起的+/-偏移位置。对于每一个要管理的轴来说，该数值必须大于 0。

下图说明旋转轴的关节空间的定义。

下图说明了线性轴的关节空间的定义。

程序执行：

关节空间的定义存储在 shapedata 类型的变量中（项目 Shape），以后在 WZLimSup 或者 WZDOSet 指令中使用。

如果和 WZHomeJointDef 指令一起还使用了 WZDOSet，当所有带关节空间监视的、激活的轴即将进入或者已经进入关节空间的时候，才能设置数字输出信号。

如果和关节空间以外 WZHomeJointDef（项目\Outside）指令一起还使用了 WZLimSup，当任何一个带关节空间监视的、激活的轴到达关节空间时，机器人立即停止。

如果和关节空间内 WZHomeJointDef（项目\Inside）指令一起还使用了 WZLimSup，当最后一个带关节空间监视的、激活的轴到达关节空间时，机器人立即停止。这就是说，一个或者几个轴，但不是所有监视的、激活的轴可以同时位于关节空间内。

当执行 ActUnit 或者 DeactUnit 指令来激活或者解除机械单元的时候，HOME 位或者工作区域限制位的管理

状态将被更新。

限制：

只有激活的机械单元和他的激活的轴在 world zone 激活的时间段内

3.44．shapedata—World Zone 形状数据

用途：

Shapedata 用来描述 World Zone 的几何形状。

描述：

World Zone 可以定义为四个不同的几何形状：

- 直立的箱体，所有的边都平行于 world 坐标系，由 WZBoxDef 指令定义。
- 一个球体，由 WZSphDef 指令定义。
- 一个圆柱体，平行于 world 坐标系的 z 轴，由 WZCylDef 指令定义。
- 机器人和/或外部轴的关节的一个空间区域，由指令 WZHomeJointDef 或者 WZLimJointDef 定义。

World Zone 的几何形状由预览指令中的一个定义，World Zone 的动作由指令 WZLimSup 和 WZDOSet 定义。

基本范例：

该数据类型的基本范例说明如下：

例1

```
VAR wzstationary pole;
VAR wzstationary conveyor;
…
PROC …
  VAR shapedata volume;
  …
  WZBoxDef \Inside, volume, p_corner1, p_corner2;
  WZLimSup \Stat, conveyor, volume;
  WZCylDef \Inside, volume, p_center, 200, 2500;
  WZLimSup \Stat, pole, volume;
ENDPROC
```

定义了一个箱体形状的 conveyor，并且该区域的监视已经激活。一个圆柱形状的 pole 也已经定义，此区域的管理也已经激活。如果机器人到达以上区域中的一个，运动将被停止。

特征：

Shapedata 是一个非数值的数据类型。

相关信息：

相关信息	参看
World Zones	《RAPID 参考手册—RAPID 概述》运动和 I/O 原理—World Zone 部分
定义圆柱形状 World Zone	第 613 页 WZCylDef—定义圆柱形状的 World Zone。
定义球形 World Zone	第 636 页 WZSphDef—定义球形 World Zone。
定义箱体形状的 World Zone	第 611 页 WZBoxDef—定义箱体形状的 World Zone。
定义关节 home 位的 World Zone	第 625 页 WZHomeJointDef—定义关节 home 位的 World Zone。
定义关节限位的 World Zone	第 629 页 WZLimJointDef—定义关节限位的 World Zone。
激活 World Zone 限位管理	第 633 页 WZLimSup—激活 World Zone 限位管理。
激活 World Zone 数字输出设置	第 617 页 WZDOSet—激活 World Zone 来设置数字输出。