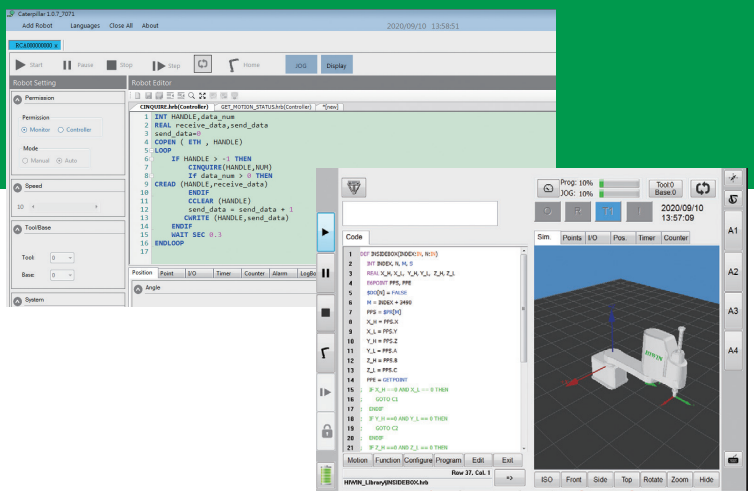
HIWIN®

机器人软件程式指令集
ROBOT Software Programming Instruction Set Manual

使用手册
User Manual



www.hiwin.cn

HIWIN® 工业4.0 优质伙伴

INDUSTRIE 4.0 Best Partner



多轴机器人
Multi-Axis Robot

取放作业/组装/整列与包装/半导体/光电业/汽车工业/食品业

- 关节式机器手臂
- 并联式机器手臂
- 史卡拉机器手臂
- 晶圆机器人
- 电动夹爪
- 整合型电动夹爪
- 旋转接头



单轴机器人
Single-Axis Robot

高精密产业/半导体/
医疗自动化/FPD面板搬运

- KK, SK
- KS, KA
- KU, KE, KC



Torque Motor 回转工作台
Torque Motor Rotary Table

医疗/汽车工业/工具机/产业机械

- RAB系列
- RAS系列
- RCV系列
- RCH系列



滚珠丝杠
Ballscrew

精密研磨/精密转造

- Super S 系列 (高Dm-N值/高速化)
- Super T 系列 (低噪音/低振动)
- 微小型研磨级
- E2 环保润滑模组
- R1 螺帽旋转式
- Cool Type 节能温控丝杠
- RD 高DN节能重负荷
- 滚珠花键



直线导轨
Linear Guideway

精密机械/电子半导体/生技医疗

- 滚珠式—
  HG重负荷型, EG低组装, WE宽幅型,
  MG微小型, CG扭矩型
- 静音式—
  QH重负荷型, QE低组装型,
  QW宽幅型, QR滚柱型
- 其他—
  RG滚柱型, E2自润型, PG定位型,
  SE金属端盖型, RC强化型



特殊轴承
Bearing

工具机产业/机械手臂

- 交叉滚柱轴承
- 滚珠丝杠轴承
- 精密线性轴承
- 轴承座



谐波减速机
DATORKER® Strain Wave Gear

机器人/自动化设备/半导体设备/工具机

- 调心输入轴组合式 DSC-PO
- 调心输入轴组件式 DSC-CO
- 中空输入轴组合式 DSH-PH
- 中空输入轴密封式 DSH-AH



AC伺服电机&驱动器
AC Servo Motor & Drive

半导体设备/包装机/SMT机台/
食品业机台/LCD设备

- 驱动器—D1, D2T/D2T-LM, E1
- 伺服电机—50W~2000W



医疗设备
Medical Equipment

医疗院所/复健中心/疗养中心

- 下肢康复训练机
- 内窥镜扶持机器手臂



直线电机平台
Linear Motor Stage

自动化搬运/AOI光学检测/
精密加工/电子半导体

- 铁心式直线电机
- 无铁心式直线电机
- 棒状直线电机
- 平面电机
- 空气轴承定位平台
- X-Y平台 • 龙门系统
- 单轴直线电机定位平台



力矩电机&直驱电机
Torque Motor &
Direct Drive Motor

工具机

- 力矩电机—TM-2/IM-2, TMRW系列

检测设备/机器人

- 直驱电机—DMS, DMY, DMN, DMT
  系列

目　录

0 保固范围与安全注意事项 7

1 手册入门使用说明 23

1.1 上银科技手册分类说明 23

1.2 如何使用指令架构说明 25

1.3 指令集总表 27

2 第一部份：指令集格式与应用说明 33

2.1 代码 A、变数&函式宣告指令 33

2.1.1 @REAL 浮点数变数指令 33
2.1.2 @INT 整数变数指令 35
2.1.3 @BOOL 宣告布林变数指令 37
2.1.4 @CHAR 宣告字元变数指令 38
2.1.5 @E6POS 宣告直角座标系位置变数指令 40
2.1.6 @E6AXIS 宣告关节座标系角度变数指令 42
2.1.7 @E6POINT 直角座标系点资讯位置变数指令 44
2.1.8 @FRAME 宣告(座标系)框架指令 46
2.1.9 @STRUC 宣告结构定义指令 48
2.1.10 @DEF... END 宣告副程式指令 50
2.1.11 @DEFFCT…ENDFCT 宣告功能函式指令 53
2.1.12 @EXT 宣告外部副程式 55
2.1.13 @EXTFCT 宣告外部函式 57
2.1.14 @STRING 宣告字串指令 59
2.1.15 ARRAY 宣告阵列指令 60
2.1.16 @UINT 宣告无符号(正)整数变数指令 63

2.2 代码 B、运动指令 65

2.2.1 @PTP 点到点移动指令 65
2.2.2 @PTP_REL 相对位置方式点到点移动指令 67
2.2.3 @LIN 直线移动指令 69
2.2.4 @LIN_REL 相对位置方式直线移动指令 71
2.2.5 @LIN_REL_TOOL 参考工具座标直线移动指令 73
2.2.6 @CIRC 圆弧移动指令 75

2.2.7 @CIRC_REL 相对位置方式圆弧移动指令 78
2.2.8 @SPLINE 点与点拟合样条指令 80
2.2.9 @STOP_MOTION 依讯触发停止上一个运动 81
2.2.10 @BRAKE 停止并清除制动中指令 82
2.2.11 @CHECK_LIN 直线路径奇异点检查指令 84
2.2.12 @P#_EXIST() 检查点位存在指令 85
2.2.13 @P#() 点位变数替换指令 87
2.2.14 @CT_A6 无限旋转最末端轴速率指令 89

2.3 代码 C、暂存器指令 91

2.3.1 @$C[] 计数暂存器指令 91
2.3.2 @$T[] 计时暂存器指令 92
2.3.3 @$T_STOP[] 停止计时暂存器指令 94
2.3.4 @$PR[] 位置暂存器指令 95
2.3.5 @$SRR[]、$SRW[] 现场总线(Fieldbus)暂存器 99

2.4 代码 D、输入&输出讯号控制指令 100

2.4.1 @$DI[ ] 数位输入讯号指令 101
2.4.2 @$DO[ ] 数位输出讯号指令 103
2.4.3 @$SI[ ] 现场总线输入讯号指令 104
2.4.4 @$SO[ ] 现场总线输出讯号指令 106
2.4.5 @$VO[ ] 电磁阀输出讯号指令 107
2.4.6 @$RO[ ] 机器人输出讯号指令 108
2.4.7 @$RI[ ] 机器人输入讯号指令 109
2.4.8 @SYN 输送带追踪的进行输出讯号开关调整时间指令 110
2.4.9 @PULSE 输出讯号脉冲时间控制指令 115
2.4.10 @$MI[ ] 组合输入讯号指令 117
2.4.11 @$MO[ ] 组合输出讯号指令 119

2.5 代码 E、流程控制功能指令 121

2.5.1 @IF 如果判断式 121
2.5.2 @FOR 对于回圈 123
2.5.3 @LOOP 循环回圈 125
2.5.4 @WHILE 当回圈 127
2.5.5 @REPEAT 重复回圈 129
2.5.6 @GOTO 去/跳跃指令 131
2.5.7 @SWITCH 开关逻辑指令 133
2.5.8 @WAIT SEC 等待秒数指令 136
2.5.9 @WAIT FOR 等待输入讯号指令 137
2.5.10 @WAIT FOR … TIMEOUT 输入讯号逾时指令 138

2.5.11 @AND 且逻辑闸指令 140
2.5.12 @OR 或逻辑闸指令 142
2.5.13 @INTERRUPT…WHEN…DO… 宣告中断条件指令 144
2.5.14 @INTERRUPT ON/OFF 启用/停止中断指令 146
2.5.15 @BREAK_POINT 程式中断点指令 148

2.6 代码 F、运动参数&系统设定指令 149

2.6.1 @SET_OVERRIDE_SPEED 复写总体速度指令 149
2.6.2 @SET_SPEED 设定速度指令 150
2.6.3 @SET_ACC 设定加速度指令 152
2.6.4 @SET_ROTATION_SPEED 设定姿态旋转速度指令 154
2.6.5 @SET_TOOL 设定工具座标系指令 155
2.6.6 @SET_BASE 设定基座座标系指令 157
2.6.7 @SET_PAYLOAD 设定末端负载重量资料指令 159
2.6.8 @GETPOINT 取得目前的座标值指令 160
2.6.9 @GET_MOTION_STATUS 读取目前运动状态指令 161
2.6.10 @EXT_TCP_START 启动外部 TCP 点位轨迹指令 163
2.6.11 @EXT_TCP_END 停止外部 TCP 点位轨迹指令 163
2.6.12 @TL_SET 设定扭力极限监控参数指令 165
2.6.13 @SERVO_ON 全轴激磁指令 166
2.6.14 @SERVO_OFF 全轴解激磁指令 167
2.6.15 @SERVO_ON(#) 单轴激磁指令 168
2.6.16 @SERVO_OFF(#) 单轴解激磁指令 169

2.7 代码 G、数学运算指令 170

2.7.1 @+ 数学运算相加指令 170
2.7.2 @- 数学运算相减指令 171
2.7.3 @* 数学运算相乘指令 172
2.7.4 @/ 数学运算相除指令 173
2.7.5 @sin 数学运算的正弦函数指令 174
2.7.6 @cos 数学运算的余弦函数指令 175
2.7.7 @tan 数学运算的正切函数指令 176
2.7.8 @ASIN 数学运算的反正弦函数指令 177
2.7.9 @ACOS 数学运算的反余弦函数指令 178
2.7.10 @ATAN 数学运算的反正切函数指令 179
2.7.11 @ATAN2 数学运算的反正切函数变化指令 180
2.7.12 @FWD 顺向机器人运动学转换指令 181
2.7.13 @% 求余数指令 183

2.8 代码 H、通讯功能指令 184

2.8.1 @COPEN 开启通讯指令.....184
2.8.2 @CCLOSE 关闭通讯指令.....186
2.8.3 @CCLEAR 清除通讯缓冲区资料指令.....187
2.8.4 @CREAD 接收资料指令.....188
2.8.5 @CWRITE 传送资料指令.....190
2.8.6 @CINQUIRE 取得资料笔数指令.....192
2.8.7 @CFORMAT 宣告自定义通讯格式指令.....194

2.9 代码 I、输送带指令 197

2.9.1 @CNV_START 启动输送带追踪程序指令.....197
2.9.2 @CNV_END 结束输送带追踪程序指令.....198
2.9.3 @CNV_PICK_QUANTITY 输送带追踪的抓取物件数量指令.....199
2.9.4 @CNV_TRIGGER_TIMES 输送带追踪的感测器触发次数指令.....200
2.9.5 @CNV_PICK 输送带追踪的抓取物件移动指令.....201
2.9.6 @CNV_PLACE 输送带追踪的摆放物件移动指令.....204
2.9.7 @CNV_OBJECT 输送带追踪的抓取之物件编号指令.....206
2.9.8 @CNV_FULL 输送带追踪的计数物件数量满额指令.....208
2.9.9 @CNV_EMPTY 输送带追踪的计数物件数量空指令.....210
2.9.10 @CNV_SET_DELAY_TIME 输送带追踪的延迟结束时间.....212
2.9.11 @CNV_QUEUE_SIZE 输送带追踪的已触发排序数量指令.....214
2.9.12 @CNV_OBJ_CNT_DIST 输送带追踪的编码器相差值指令.....215
2.9.13 @CNV_PLACE_BATCH 输送带追踪的可飞放的最大次数.....216
2.9.14 @CNV_RESET_ENC 输送带追踪的清除编码器的计数指令.....217
2.9.15 @CNV_QUEUE_REMOVE 输送带追踪的移除的序列指令.....218
2.9.16 @CNV_PICK_ACC 输送带追踪的下压加速度时间指令.....219
2.9.17 @CNV_OFFSET_X 输送带追踪的 X 轴位置偏移值指令.....220
2.9.18 @CNV_OFFSET_Y 输送带追踪的 Y 轴位置偏移值指令.....221
2.9.19 @CNV_OFFSET_Z 输送带追踪的 Z 轴位置偏移值指令.....222
2.9.20 @CNV_SPEED 输送带追踪的取得输送带速度指令.....223

2.10 代码 J、讯息提示功能指令 224

2.10.1 @USER_ALARM 触发使用者自定义警报.....224

2.11 代码 K、模拟环境指令 225

2.11.1 @ADDTOOL 模拟画面新增末端工具指令.....225
2.11.2 @ADDOBJ 模拟画面新增周边物件指令.....228
2.11.3 @SHOW_TOOL 模拟画面显示/隐藏末端工具指令.....230
2.11.4 @SHOW_OBJ 模拟画面显示显示/隐藏周边物件指令.....232
2.11.5 @MOVEFLOOR 模拟画面移动地板位置指令.....233
2.11.6 @AXISON 模拟画面显示座标系(轴)指令.....234

2.11.7 @AXISOFF 模拟画面隐藏座标系(轴)指令 235

2.12 代码 L、电动夹爪功能指令 236

2.12.1 @EG_OPEN 电动夹爪的开启连线指令 236
2.12.2 @EG_CLOSE 电动夹爪的关闭连线指令 238
2.12.3 @EG_RESET 电动夹爪的重置原点指令 239
2.12.4 @EG_GET_STATUS 电动夹爪的状态取得指令 240
2.12.5 @EG_RUN_MOVE 电动夹爪的位置移动指令 242
2.12.6 @EG_RUN_GRIP 电动夹爪的夹持动作指令 244
2.12.7 @EG_RUN_EXPERT 电动夹爪的专家模式 247
2.12.8 @EG_GET_POS 电动夹爪的当前位置取得指令 250

2.13 代码 M、外部轴功能指令 252

2.13.1 @EX_AX 外部轴的开启/关闭连接指令 252
2.13.2 @EX_AX_ASYNC 外部轴的设定同步轴/非同步轴指令 257
2.13.3 @ EX_AX_SET_ACC 外部轴的设定加/减速度指令 258
2.13.4 @EX_AX_SET_SPEED 外部轴的设定速度指令 259
2.13.5 @EX_AX_CT 外部轴的无限旋转速率指令 260
2.13.6 @ASYPTP 非同步运动外部轴点到点运动 262

2.14 代码 N、焊接功能指令 264

2.14.1 @WELD_START 启动焊接程序指令 264
2.14.2 @WELD_END 结束焊接程序指令 269
2.14.3 @WEAVE 焊接的设定摆焊动作指令 270
2.14.4 @WEAVE_END 结束焊接的摆焊动作指令 270
2.14.5 @RESUME_PROG 焊接的设定中断复归范围程式指令 277
2.14.6 @CLEAR_RESUME_PROG 清除焊接的中断复归程式范围指令 277

2.15 代码 O、碰撞侦测功能指令 281

2.15.1 @CD_START 启动碰撞侦测功能与灵敏度设定指令 281
2.15.2 @CD_END 关闭碰撞侦测功能指令 282

2.16 代码 P、字串操作功能指令 283

2.16.1 @STRCMP 字串比较指令 283
2.16.2 @STRPOS 搜寻字串位置指令 284
2.16.3 @STRCPY 字串复制指令 286
2.16.4 @STRLEN 取得字串长度指令 288
2.16.5 @INTTOSTR 整数型态转成字串型态指令 289
2.16.6 @REALTOSTR 浮点数型态转成字串型态指令 290
2.16.7 @STRTOINT 字串型态转成整数型态指令 291

2.16.8 @STRTOREAL 字串型态转成浮点数型态指令 292

2.17 代码 Q、专用功能指令 293

2.17.1 @TL_START 启用扭力极限监控指令 293
2.17.2 @TL_END 结束扭力极限监控指令 293

3 第二部份：功能性副程式与应用说明 294

3.1 使用功能性副程式方式说明 295

3.1.1 将副程式汇入资料夹 295
3.1.2 主程式呼叫副程式应用说明 299

3.2 代码 FA 数学运算功能性副程式说明 303

3.2.1 @ABS 数值取绝对值计算函式 303
3.2.2 @ROUND 四舍五入计算函式 304
3.2.3 @BIT 位元转换计算函式 305
3.2.4 @SQRT 开平方根计算函式 306

3.3 代码 FB 矩阵功能性副程式说明 307

3.3.1 @PAL_PR 定义矩阵功能副程式 307
3.3.2 @PAL_P 指定矩阵点位功能函式 310

3.4 代码 FC 区域设定功能性副程式说明 311

3.4.1 @BOX 设立检测区副程式 311
3.4.2 @INSIDEBOX 判定检测区副程式 313
3.4.3 @ATPOS 安全区域设定副程式 315

3.5 代码 FD 点位移动功能性副程式说明 316

3.5.1 @JUMP 门型移动副程式 316
3.5.2 @CIRCLE 圆周移动副程式 318
3.5.3 @OVAL 椭圆移动副程式 320

附录一、指令集预设连续(CONT)/不连续(FINE)轨迹总表 322

附录二、ASCII 可显示字元表 333

0 保固范围与安全注意事项

保固范围

本产品之保固时间为到货日起 12 个月，保固范围不包含以下原因所引起的任何故障：

1. 非本公司架设之生产线，与其他设备连接所导致之机器手臂损坏。
2. 超出产品手册定义之操作方式、操作环境及储存规范。
3. 由专业安装人员安装完毕后，因任何原因需移动安装处、改变使用环境或运送方式不当造成的损坏。
4. 因人为操作或安装不当所造成之碰撞及事故导致机器手臂损坏。
5. 机器手臂上安装非本公司之产品。

以下情况不在保固范围内：

1. 产品编号或生产日期（月和年）无法验证的产品。
2. 机器手臂本体及控制器元件使用 HIWIN 原厂外之产品。
3. 任意增加或移除机器手臂本体或控制器的任何元件。
4. 任意修改机器手臂本体或控制器间之线路或电缆。
5. 任意修改机器手臂及控制器外观以及任意拆卸机器手臂及控制器元件，例如：拆卸外壳、于产品上钻孔或切割等。
6. 任何天灾所造成之损毁或损坏，例如：火灾、地震、海啸、雷击、风灾以及洪水等。

在上述情况下产品发生损毁或损坏，HIWIN 不提供任何保固或赔偿，除非使用者分析证实为产品不良所导致。

有关保固期和条款的详细资讯，请联系购买产品之经销商或技术人员。

警告

1. 不当的修改或拆解可能导致机器手臂故障或降低机器手臂的性能、可靠性或寿命。
2. 末端工具或其他设备所需之电缆及管线应由专业人员进行安装、设计，避免影响机器手臂运动或引起机器手臂之损坏。
3. 如因产线配置问题，需特殊修改需求，请与技术人员连络。
4. 基于安全考量，严禁对 HIWIN 工业机器手臂进行修改。

安全注意事项

安全资讯

- 安全责任与效力
  1. 此章说明安全使用机械手臂需遵守的内容，在使用机械手臂之前，请务必详读此章内容。
  2. HIWIN 的工业机械手臂的使用者需设计与安装符合工业安全规范的安全装置，用以保护人身安全。
  3. 遵守本手册所讲述的任何有关工业机械手臂的安全资讯并不能被解读为 *HIWIN* 的工业机械手臂不会发生任何安全事故。
  4. 本机器被定义为部分完成的机器，相关的危险必须由系统整合商根据 ISO 10218-1 / ISO 10218-2 进行处理。
  5. 控制系统（SRP / CS）的安全相关部分应符合 ISO 13849-1 中性能等级 d 和类别 3 的要求。
  6. 额外新增紧停系统须由系统商定义，并遵照 ISO 10218-1/ISO 10218-2。

- 操作注意守则
  1. *HIWIN* 工业机械手臂启动装配程序连接电源前，应确认厂务输出电压规格与该产品的输入电压规格是否相符，若不相符，请务必使用对应变压器(建议使用 *HIWIN* 选配变压器)。
  2. 启动关机程序前，须先压下紧急停止开关(位于教导器上或外部紧急停止装置)，再开始关机程序。
  3. 在连接外部 I/O 或讯号时，请保持在电源关闭的状态下操作，防止过程中发生误触短路，造成损坏。

安全预防措施

i. 一般

所有与工业机械手臂相关之使用或设置工作的人员，必须详细阅读机械手臂系统安全相关文献以及说明手册，并在遵守规范条件下操作使用。

安全符号

危险

使用者必须严格遵守内容说明，否则将会造人员严重伤亡。

警告

使用者必须严格遵守内容说明，否则将可能造成人员受到轻伤或设备损坏。

注意

使用者必须严格遵守内容说明，否则将可能造成产品操作不良的状况。

使用限制

机械手臂禁止使用于下列所述之环境与用途

- 人员载运用途
- 爆炸危险的环境
- 未加装安全防护措施的环境
- 室外环境
- 油、水、尘等影响的环境

ii. 相关人员

工业机械手臂电气或机械方面的工作只允许由专业人员进行。

⚠ 警告

所有在工业机械手臂上工作的人员，必须阅读并理解含有机械手臂系统安全章节的文献。

系统整合者

系指将工业机械手臂按照安全规定集成到一套设备并进行投入运行调试的人员。

系统整合者负责以下工作：

- 安装工业机械手臂
- 工业机械手臂相关设备连接工作
- 整体系统的风险评估
- 使用安全防护装置
- 确认安全防护装置使用元件是否符合规定
- 安置、更换、设定、操作、保养和维修工作只允许经特殊培训过的人员按工业机械手臂各元件的操作指南来进行。

使用者

使用者必须受过专业训练、并具有该方面知识和经验，且熟知规定的标准，并由此能对准备从事的工作做出正确判断、能够辨别潜在危险的人员。

根据操作权限可将使用者定义成三种类别：

1. 操作人员 (Operator)
   - 系统的启动与关闭
   - 电源启闭
   - 警报系统状态恢复
2. 工程师 (Engineer)
   - 操作人员使用权限范围
   - 程式编写与更动
   - 手臂教导操作
3. 专家 (Expert)
   - 工程师使用权限范围
   - 机械手臂维修保养工作

上银机器人软体系统(HRSS)系统操作

未使用功能安全套件者，必须实施围篱外教导

人员的系统操作分为以下三个位阶

1. 操作人员
2. 工程师
3. 专家

其控制权限如下表所示

项次	功能	操作者 Operator	工程师 Engineer	专家 Expert
	功能表			
1	File	X	X	O
2	Configuration>User group	O	O	O
3	Display>Input/Output	X	O	O
4	Display>Variable	X	O	O
5	Display>Mileage	O	O	O
6	Display>Utilization	O	O	O
7	Display>Motor Torque	O	O	O
8	Display>Fieldbus	X	X	O
9	Diagnosis>Logbook	O	O	O
10	Start-up>Calibrate	X	X	O
11	Start-up>Master	X	X	O
12	Start-up>Robot data	X	O	O
13	Start-up>Network Config	X	X	O
14	Start-up>RS-232	X	X	O
15	Start-up>System Setting	X	X	O
16	Track>Setting	X	O	O
17	Track>Vision Setting	X	O	O
18	Track>Vision Object	X	O	O
19	Track>Calibration	X	O	O
20	Track>Monitor	O	O	O
21	Help>About	O	O	O
22	Help>Operating Time	O	O	O
23	Help>Update	X	X	O
24	Help>TP Calibration	O	O	O
25	Help>Manual	O	O	O

(续下表)

(续上表)

项次	功能	操作者 Operator	工程师 Engineer	专家 Expert
	介面			
26	讯息栏	O	O	O
27	速度设置	X	O	O
28	工具/基底座标	X	X	O
29	教导器位置设置	X	O	O
30	更改手动座标系	X	O	O
31	手动运行操作	X	O	O
32	小键盘	O	O	O
33	移除教导器	X	O	O
34	执行上下行程式	X	X	O
35	程式操作	O	O	O
36	程式选择	O	O	O
37	修改程式	X	X	O
38	校正点操作	X	O	O
39	IO 操作	X	O	O
40	Functional IO 程序变更	X	X	O

若为远端连线系统 Caterpillar 请参考下页表格。

远端连线系统(Caterpillar)系统操作

人员的系统操作分为以下两个位阶

1. 操作者
   - ■ 自动模式
   - ■ 手动模式
2. 监控者

其控制权限如下表所示

项次	功能	操作者		监控者
		手动模式	自动模式	
	功能表			
1	介面语言	O	O	O
2	权限选择	O	O	O
3	模式选择	X	O	X
4	速度调控	O	O	X
5	工具/基底选择	O	X	X
6	更新软体	O	X	X
7	储存资料库	O	X	X
8	载入资料库	O	X	X
9	程式执行	O	O	X
10	程式暂停	O	O	X
11	程式停止	O	O	X
12	原点	O	X	X
13	移动	O	X	X
14	程式编辑	O	X	X
15	程式上传/下载	O	X	X
16	点位增加	O	X	X
17	计数器	O	X	X
18	系统 I/O	O	O	X
19	RSR 设定	O	X	X
20	Fieldbus	O	O	X
21	清除警报	O	O	X
22	输送带追踪->设定	O	X	X
23	输送带追踪->视觉设定	O	X	X
24	输送带追踪->视觉物件设定	O	X	X
25	输送带追踪->感测器物件设定	O	X	X
26	输送带追踪->校正	O	X	X
27	输送带追踪->监控	O	O	O

(续下表)

(续上表)

项次	功能	操作者		监控者
		手动模式	自动模式	
28	定位校正	O	X	X
29	座标系校正	O	X	X
30	事件纪录簿	O	O	O
31	网路配置	O	X	X
32	原点设定	O	X	X
33	时间设定	O	X	X
34	RS-232	O	X	X
	介面			
35	加入手臂	O	O	O
36	全部断线	O	O	O

若为上银机器人软体 HRSS 请参考上页表格。

警告

工业机械手臂电气或机械方面的工作只允许由专业人员进行。

操作人员安全注意事项

在工作之前必须对相关人员就工作的方式和规模以及可能存在的危险进行说明，必须定期进行相关训练课程。若发生意外事故或进行技术修正后必须重新进行一次训练课程。

系统架设者的安全注意事项

系统架设者仅允许经过特殊训练的人员从事其工作，并按照原厂提供之安装、设置、操作等相关文件进行工作。

维修人员的注意事项

维修工作仅允许经过特殊训练的人员，按照各说明书与操作指南说明进行。

iii. 机械手臂工作范围定义

- 工作区域

  机械手臂工作区域定义系指其在运动限制下的运动区域，工作区域必须限定在需要的最小范围内。

- 防护区域

  防护区系指作业区中受到安全装置保护的区域。该区域范围需包含工作区域，并通过安全防护装置确保工作区域安全。

  任何人为的操作，必须在防护区域外。依据 ISO13849-1，控制系统内与安全相关的部分，必须符合效率等级 d 和范畴 3 的要求。

CAUTION

开启使用手臂前，请确认紧停开关已处于重置状态。

连接于紧停开关电路的外部装置，必须干接(没有充电)的开关。避免使用有电线路连接到控制紧停开关的电路。

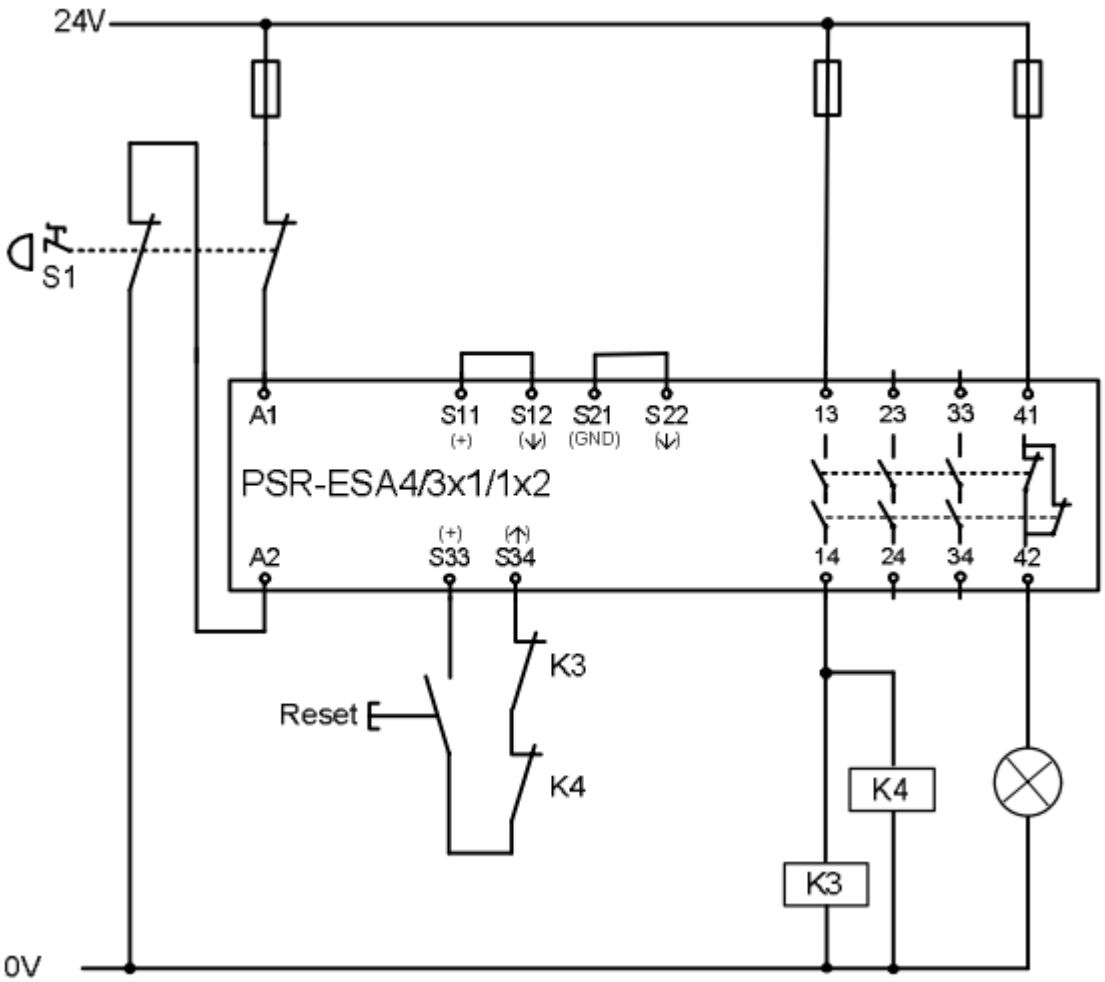


图 1 控制紧停开关的电路图

CAUTION

❖ EMO-紧停装置 (EN 60947-5-1 正面开启) 必须有安全模组，符合安规 ISO 13849-1 PL d 要求。



图 2 安全装置保护的区域图

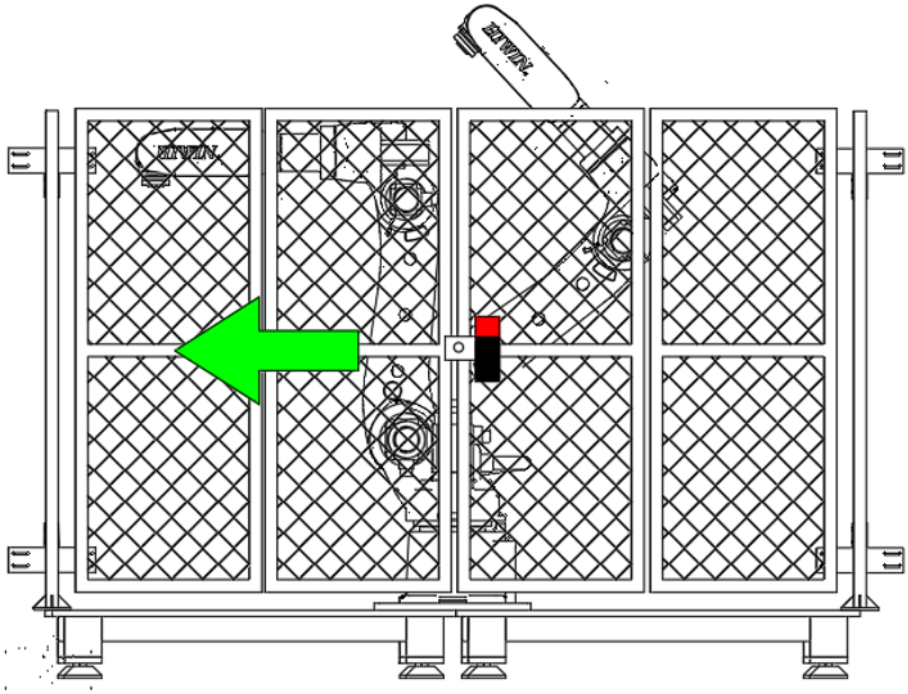


图 3 安全装置保护的装置示意图

iv. 安全功能说明

工业机械手臂需具有以下安全功能：

- 机械手臂的运转模式选择
- 安全防护装置
- 紧急停止装置
- 教导器致能开关

机械手臂系统的安全功能为防止人员或财产受到损失，若功能不整或是失效状态下的，必须禁止运行工业机械手臂。

手动运行模式

手动运行模式系用于程式设计、程式运行查验或示教等工作，进行手动运行时应注意如下事项：

- 所有动作都必须在防护区域内进行。
- 不可因工业机械手臂开动而造成相关设备的损坏或潜在危害。
- 必须尽可能在防护区域外进行。

除非手臂已搭载由原厂提供已认证的速度监控配件，否则防护区域内使用手动与自动运行模式都是不被允许的。

自动运行模式

自动运行模式启动应包含下列条件:

- 防护装置都已完成设置并确认其功能正常运作。
- 所有暂停的安全防护应恢复其全部功能。
- 确认防护区域内没有任何人员。
- 相关工作流程规定遵守。

若需在此模式下进入防护区域，必须先启动紧急停止功能才可进入。

安全防护装置说明

防护装置必须选用安规认可之元件，并依照相关法规要求设置与规划。

机械手臂系统需在收到安全防护讯号下自动运行方式才能被启动。自动模式运行期间若出现连线失效的情况时，必须触发紧急停止。断线后重新连线时不可直接自动启动装置，必须以人为方式启动。必须提供方法确认启动自动模式时，防护区域内没有人员停留。

使用者必须严格遵守内容说明，否则将会造人员严重伤亡。

系统架设期间可使用临时性围篱，可依 ISO 10218-2 法规相关内容设置。

停止功能

紧急停止说明

紧急停止相关注意事项

- 每半年确认功能是否做动正常。
- 系统整合者应提供紧急停止装置，确保机器作动或有危险发生可能性的工作状况可供使用。
- 至少安装有一个外部紧急停止装置。确保在没有或遗失教导器的情况下也需有额外的紧急停止装置可供使用。
- 提供介面连接外部紧急停止装置。
- 当与机械手臂连接的安全控制系统被切断时，可触发紧急停止功能。
- 风险评估中应评估，机械手臂控制系统关闭时未触发紧急停止的情况是否危险以及提供应对方法。
- 若与机器手相连的工具或其他装置如可能引发危险，则必须将其连入设备侧的紧急停止回路中。

v. 警告与注意事项

一般注意事项

危险

1. 所有作业程序必须遵守相关安全规范，并由专业人员进行相关评估工作。
2. 使用机械手臂的作业人员需配戴安全用具后再进行作业，如适合工作环境的工作服、安全鞋和安全帽。
3. 当遇到人员因机器手臂遭遇危险或其他紧急和异常情况时，请于第一时间按下紧急停止按钮，并用手动模式低速的将手臂远离危险状况。
4. 在考虑机器手臂安全性时，必须将手臂及系统一起考虑，使用机器手臂时，务必架起安全栅栏或其他安全设施，操作人员需在安全栅栏外才可操作手臂
5. 必须在机器手臂的工作范围之外设置一个安全区，并使用适当的安全装置，阻绝未经许可的人员进入。
6. 当安装或拆卸任何机械零件时，掉落的零件可能会对操作者造成伤害。
7. 确保工件重量，不得超过机器手臂的额定负载或可承受的扭矩，否则可能会导致驱动器警报或故障。
8. 禁止任何攀爬机器手臂的动作。
9. 请勿在存在腐蚀性、易燃性气体的环境内或靠近可燃物环境使用。
10. 请勿在潮湿或油水侵入之环境使用。
11. 请勿在振动或冲击激烈的地方使用。
12. 请勿将电线浸在油或水等液体中使用。
13. 请勿用湿手接线或操作。
14. 请勿在有潜在爆炸危险的环境使用。
15. 请确定控制器是否确实接地。
16. 控制器接上电源或运作时，切勿将手伸入到控制器内部。
17. 控制器的内部元件中,驱动器的散热器、回生电阻、电源供应器、电脑在运作时温度会上升，请勿碰触。
18. 控制器的移动、接线、检查、维护时要在切断电源后，并确认没有触电危险的前提下进行。
19. 请勿自行拆装控制器，若有需求请询问原厂工程。

警告

1. 安装机器手臂的人员必须受过相关的教育训练与许可。
2. 为了保护人身安全，必须遵守本手册安装程序以及相关工业安全规范。
3. 控制箱避免放置于高电压或其他会产生电磁场元件附近，以免发生电磁干扰并可能导致机器手臂动作偏差或故障。
4. 使用非原厂的维修零件。可能会导致机器手臂损毁或故障。
5. 需当心控制器和伺服电机产生的热源。
6. 不要过度弯曲缆线。否则可能会导致不可预期的危险。
7. 请勿站在产品上或在产品上放置重物。
8. 请勿堵塞散热孔，也不要放入异物。
9. 请确实将控制器固定在底座上。
10. 请勿用力拉扯接头或过度地卷曲电线。
11. 请勿频繁地开关电源开关与控制按钮。
12. 在开始作业前，请确认机器手臂、紧急停止开关、控制器等相关装置无异常状况。
13. 切勿在运作时，切断电源开关。
14. 请勿自行拆装、改造、分解、维修。
15. 长时间不使用时务必切断电源。
16. 机器手臂在示教时，请保持低速，并随时观察其运作状况。以避免导致工件掉落或造成操作者的危险。
17. 更改机器手臂控制器内部的程式或参数时，请勿关闭控制器的电源，否则会造成控制器内部资料的损坏。
18. 当伺服马达的刹车被释放后，机器手臂会因为重力的影响而移动，有可能对操作者造成伤害。
19. 工业机械手臂可以应用于许多不同的工业环境。
20. 当作业程序被中断时，需要人员进行故障排除时，需特别注意作业风险。

操作时的注意事项

危险

1. 编程时务必在安全栅栏外进行，如需进入安全栅栏内进行作业，必须按下紧急停止按钮。
2. 所有的操作必须由接受过教育训练的操作人员执行。

维护时的注意事项

危险

1. 如需进行非 HIWIN 指定的维修程序时，请与本公司联系。
2. 如需更换非 HIWIN 指定的零件时，请与本公司联系。
3. 务必进行定期维修，否则会影响机器手臂的寿命或其他不可预期的危险。
4. 进行维修及保养前，需先关闭所有电源。
5. 必须由合格的人员进行保养或维修，并清楚了解整个系统的安装程序与其他可能伴随的风险。
6. 更换零件时，避免其他异物进入手臂内。

使用末端效应器的注意事项

危险

末端效应器基本上可分为以下两类：

A. 夹具类：以取放作业为主，如气动、电动夹爪、真空吸盘等。
B. 工具类：以加工作业为主，如焊接、切割、表面处理等。

1. 当机器手臂出现动力或其他任何错误时所造成工件掉落或损毁，在设计时必须特别注意。
2. 末端效应器如果有高电压、高温或高速旋转处则需特别注意作业安全。
3. 末端效应器需确实安装在机器手臂上，避免在操作过程中工件掉落，可能会造成人员的受伤或危险。

警告

1. 末端效应器可能具有自身的控制单元，安装时须注意安装位置，避免与机器手臂的作业产生干涉。
2. 夹具类末端效应器为了防止，当机器手臂出现动力错误或其他任何错误时，所造成工件掉落或损毁，在设计时必须特别注意。

使用液压及气压的注意事项

危险

1. 使用液、气压系统进行作业时，可能会因为压力不足或重力而导致夹持的工件掉落。
2. 为提供紧急状况使用，液、气压系统需加装安全泄压阀。

警告

1. 气压、液压系统的压力值，在关闭动力后，依然会储存在系统内，需特别注意。
2. 在维修气压、液压系统前，需先释放系统内存压力。
3. 气压、液压系统内存的压力，通常为大气压力的数倍，需特别注意作业安全。

紧急停止开关的注意事项

危险

1. 机器手臂或其他控制元件需具有至少一个可以使进行中之程序立即停止的装置，如紧急停止开关。
2. 紧急停止开关必须放置于容易操作的位置，以便于快速地停止机器手臂。
3. 执行紧急停止时，会切断驱动器对马达的动力供给及停止所有的动作。若要恢复执行程序，需重置紧急停止开关。
4. 避免使用紧急停止开关取代正常的停止程序，以免机器手臂产生不必要的损耗。

警告

1. 执行紧急停止时，会切断驱动器的动力，停止所有的动作，并切断机器手臂的控制系统。
2. 若要恢复执行程序，需重置紧急停止开关。
3. 紧急停止为立即停止：立刻停止机器手臂的动作，并切断驱动器的动力。
4. 紧急停止开关仅供紧急停止使用。
5. HIWIN 的工业机械手臂具有两个紧急停止开关，其中一个紧急停止开关位于教导器上，另一个紧急停止开关透过专用连接线直接连接控制器。若有其他紧急停止开关的需求，可透过其他连接方式达到紧急停止的目的。
6. 基于相关之工业安全规范，紧急停止开关需透过实体连接线，直接连接机器手臂的控制箱。

1 手册入门使用说明

1.1 上银科技手册分类说明

在上银科技每一个手臂产品手册主要分为本体使用手册、控制器使用手册、软体使用手册、远端连线使用手册、软体开发套件使用手册、及软体指令集手册、机器人通讯手册等七本手册，本册为专叙述软体的指令集手册，若有其他需求请参考其他本手册。以下说明适用的产品对于其搭配手册：

手册名称 / 手臂规格	本体手册	控制器手册	软体手册	远端连线手册	软体开发套件手册	软体指令集手册	通讯手册
六轴关节式机器手臂系列 (RA605、RA610、RA620)	V	V	V	X	V	V	V
史卡拉机器手臂 SCARA 系列(RS405、RS410)	V	V	X	V	V	V	V
并联式机器手臂 Delta 产品 (RD403)	V	V	V	X	V	V	X
晶圆机器手臂 Wafer 产品 (RW 系列)	V	V	V	X	X	X	X
电动夹爪 X 系列	V	X	V	X	X	X	X
电动夹爪 S 系列	V	X	X	X	X	X	X

备注：V 代表适用，X 代表不适用。

※以 RA605-710-GC 六轴关节式机器手臂产品为范例：

手册序	1	2	3	4
图片				
名称	本体使用手册	控制器使用手册	机器人系统软体手册	远端操作介面软体手册
内容	手臂本体规格、选配件、保养/维修。	手臂的控制器规格、控制器选配件、保养/维修。	上银手臂系统软体 HRSS 软体安装、介面操作、应用软体范例。	远端连线软体 Caterpillar 安装、介面操作、应用软体范例。(注: SCARA-LU 系列适用，六轴不适用)
手册序	5	6	7	
图片				
名称	机器人软体开发套件手册	软体程式指令集手册	机器人通讯手册	
内容	提供手臂函式库 SDK 供使用者自行撰写软体介面来操控手臂。	上银手臂系统软体 HRSS 与远端连线软体 Caterpillar 共用，如何使用程式指令说明。	上银手臂系统软体 HRSS 与远端连线软体 Caterpillar 共用，如何使用(Modbus)通讯说明。	

1.2 如何使用指令架构说明

❖ 指令分类类别说明：

将 HRSS&Caterpillar 软体指令分成两个部分，第一部分为软体中的指令，只要在软体中输入指令即可使用，参考表 1 指令类别分类表，第二部分为软体中的功能性副程式，此部份为现有指令的延伸功能，需透过外部副程式/函式方式来达成特定功能，参考表 2 功能性副程式类别分类表。

将所有 HRSS&Caterpillar 软体内部指令依据使用不同功能方式来做分类，参考下表为所区分的类别并给予一个英文字代码，代码会应用在指令编号上做编号区分，使用者可根据分类方式做类别寻找，再寻找所需要的程式指令章节，或透过下个章节由总表中，寻找所需程式指令。

表 1 指令类别分类&代码表

项次	分类代码	指令分类
1	A	变数宣告
2	B	运动指令
3	C	暂存器
4	D	输入、输出讯号控制
5	E	流程控制功能
6	F	运动参数/系统设定
7	G	数学计算
8	H	通讯功能
9	I	输送带
10	J	讯息提示
11	K	模拟环境
12	L	电动夹爪功能
13	M	外部轴功能
14	N	焊接功能
15	O	碰撞侦测功能
16	P	字串操作

表 2 功能性副程式类别分类&代码表

编号	指令代码	功能性副程式分类
1	FA	数学运算功能
2	FB	矩阵功能
3	FC	区域设定功能
4	FD	点位移动功能

❖ 指令项目说明：

每个指令有 8 个项目说明，以下说明各项目内容：

1. **指令名称：**可以使用的输入指令名称及指令名称中文说明，前头有@为一个指令，用于快速指令寻找作用，并使用字体反黑(字元网底功能)做明显指令识别作用。

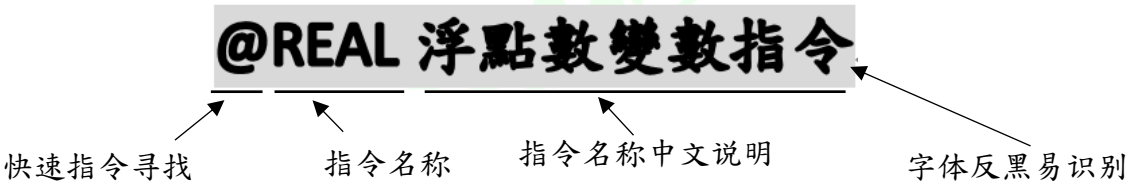


图 4 指令名称功能说明图

2. **指令说明：**简易描述此指令功用说明。
3. **前提条件：**使用此指令时，告知使用者应具备的条件补充说明。

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

在 HRSS 上銀機器人軟體系統可適用版本

在 Caterpillar 遠端連線軟體可適用版本

每個指令預設狀態，為預讀 CONT 或 不預讀 FINE。
不預讀 FINE 會讓程式等待執行完成目前指令後，再進行下一行指令，預讀則不會。總表參考
附录一、指令集预设连续(CONT)/不连续(FINE)轨迹总表

4. **指令格式：**说明指令完整格式，告知使用者程式制定指令格式方式。
5. **格式/参数说明：**以文字方式补充说明格式/参数指令内容，或说明区间范为数值。
6. **范例：**提供一个以上程式范例说明此指令使用。
7. **范例说明：**以文字方式补充说明范例程式表达内容。
8. **参考相关指令：**告知使用者相关指令，可参考类似指令应用。

1.3 指令集总表

第一部份指令集所有 HRSS 与 Caterpillar 所有的指令集表如下所示，快速连结请点页码编号。

表 3 指令集总表

分类代号	指令分类	指令编号	指令名称	说明	页码
A	变数宣告	A-1	REAL	宣告浮点数变数	33
		A-2	INT	宣告整数变数	35
		A-3	BOOL	宣告布林变数指令	37
		A-4	CHAR	宣告字元变数指令	37
		A-5	E6POS	宣告直角座标系位置变数指令	40
		A-6	E6AXIS	宣告关节座标系角度变数指令	42
		A-7	E6POINT	宣告直角座标系点资讯位置变数指令	44
		A-8	FRAME	宣告(座标系)框架指令	46
		A-9	STRUC	宣告结构定义指令	48
		A-10	DEF... END	宣告副程式指令	50
		A-11	DEFFCT…ENDFCT	宣告功能函式指令	53
		A-12	EXT	呼叫外部副程式指令	55
		A-13	EXTFCT…ENDFCT	呼叫外部功能函式指令	57
		A-14	STRING	宣告字串指令	59
		A-15	ARRAY 自定义名称	宣告阵列(Array)指令	60
		A-16	UINT	宣告无符号(正)整数变数指令	63
B	移动方式	B-1	PTP	点到点移动指令	65
		B-2	PTP_REL	相对位置方式点到点移动指令	67
		B-3	LIN	直线移动指令	69
		B-4	LIN_REL	相对位置方式直线移动指令	71
		B-5	LIN_REL_TOOL	参考工具座标相对位置方式直线移动指令	73
		B-6	CIRC	圆弧移动指令	75
		B-7	CIRC_REL	相对位置方式圆弧移动指令	78
		B-8	SPLINE	点与点拟合样条指令	80
		B-9	STOP_MOTION	触发上一个运动停止	81
		B-10	BRAKE	停止并清除制动中指令	82
		B-11	CHECK_LIN	直线路径奇异点检查指令	84
		B-12	P#_EXIST()	检查点位存在指令	85
		B-13	P#()	点位变数替换指令	87
		B-14	CT_A6	无限旋转第六轴速率指令	89

(表格 1/6)

表 3(续 1)

分类代号	指令分类	指令编号	指令名称	说明	页码
C	暂存器	C-1	$C[]	计数暂存器指令	91
		C-2	$T[]	计时暂存器指令	92
		C-3	$T_STOP[]	停止计时暂存器指令	94
		C-4	$PR[]	位置暂存器指令	95
		C-5	$SRR[]、$SRW[]	现场总线(Fieldbus)暂存器	99
D	输入、输出讯号控制	D-1	$DI[]	数位输入讯号指令	101
		D-2	$DO[]	数位输出讯号指令	103
		D-3	$SI[]	现场总线输入讯号指令	104
		D-4	$SO[]	现场总线输出讯号指令	106
		D-5	$VO[]	电磁阀输出讯号指令	107
		D-6	$RO[]	机器人(Robot)输出点位讯号指令	108
		D-7	$RI[]	机器人(Robot)输入点位讯号指令	109
		D-8	SYN	输送带追踪的进行输出讯号开关调整时间指令	110
		D-9	PULSE	输出讯号脉冲时间控制指令	115
		D-10	$MI[]	组合输入讯号指令	117
		D-11	$MO[]	组合输出讯号指令	119
E	流程控制功能	E-1	IF	如果判断式	121
		E-2	FOR	对于回圈	123
		E-3	LOOP	循环回圈	125
		E-4	WHILE	当回圈	127
		E-5	REPEAT	重复回圈	129
		E-6	GOTO	去；跳跃指令	131
		E-7	SWITCH	开关(switch)逻辑指令	133
		E-8	WAIT SEC	等待秒数指令	136
		E-9	WAIT FOR	等待输入讯号指令	137
		E-10	WAIT FOR ... TIMEOUT	输入讯号逾时指令	138
		E-11	AND	且逻辑闸指令	140
		E-12	OR	或逻辑闸指令	142
		E-13	INTERRUPT…WHEN…DO…	宣告中断条件指令	144
		E-14	INTERRUPT ON/OFF	启用/停止中断指令	146
		E-15	BREAK_POINT	程式中断点指令	148

(表格 2/6)

表 3(续 2)

分类代号	指令分类	指令编号	指令名称	说明	页码
F	运动参数/系统设定	F-1	SET_OVERRIDE_SPEED	设定复写总体速度指令	149
		F-2	SET_SPEED	设定速度指令	150
		F-3	SET_ACC	设定加速度指令	152
		F-4	SET_ROTATION_SPEED	设定姿态旋转速度指令	154
		F-5	SET_TOOL	设定工具座标系指令	155
		F-6	SET_BASE	设定基座座标系指令	157
		F-7	SET_PAYLOAD	设定末端负载重量资料	159
		F-8	GETPOINT	取得目前的座标位置或角度值指令	160
		F-9	GET_MOTION_STATUS	读取目前运动状态指令	161
		F-10	EXT_TCP_START	启动外部 TCP 点位轨迹指令	163
		F-11	EXT_TCP_END	停止外部 TCP 点位轨迹指令	163
		F-12	TL_SET	设定扭力极限监控参数	165
		F-13	SERVO_ON	全轴激磁指令	166
		F-14	SERVO_OFF	全轴解激磁指令	167
		F-15	SERVO_ON(#)	单轴激磁指令	168
		F-16	SERVO_OFF(#)	单轴解激磁指令	169
G	数学运算	G-1	+	数学运算的相加指令	170
		G-2	-	数学运算的相减指令	171
		G-3	*	数学运算的相乘指令	172
		G-4	/	数学运算的相除指令	173
		G-5	SIN	数学运算的正弦函数指令	174
		G-6	COS	数学运算的余弦函数指令	175
		G-7	TAN	数学运算的正切函数指令	176
		G-8	ASIN	数学运算的反正弦(arcsin)函数指令	177
		G-9	ACOS	数学运算的反余弦(arccos)函数指令	178
		G-10	ATAN	数学运算的反正切(arctan)函数指令	179
		G-11	ATAN2	数学运算的反正切函数变化指令	180
		G-12	FWD	顺向机器人运动学转换指令	181
		G-13	%	求余数指令	183

(表格 3/6)

表 3(续 3)

分类代号	指令分类	指令编号	指令名称	说明	页码
H	通讯功能	H-1	COPEN	开启通讯指令	184
		H-2	CCLOSE	关闭通讯指令	186
		H-3	CCLEAR	清除通讯缓冲区资料指令	187
		H-4	CREAD	接收资料指令	188
		H-5	CWRITE	传送资料指令	190
		H-6	CINQUIRE	取得资料笔数指令	192
		H-7	CFORMAT	宣告自定义通讯格式指令	194
I	输送带	I-1	CNV_START	启动输送带追踪程序指令	197
		I-2	CNV_END	结束输送带追踪程序指令	198
		I-3	CNV_PICK_QUANTITY	输送带追踪的抓取物件数量指令	199
		I-4	CNV_TRIGGER_TIMES	输送带追踪的感测器触发次数指令	200
		I-5	CNV_PICK	输送带追踪的抓取物件移动指令	201
		I-6	CNV_PLACE	输送带追踪的摆放物件移动指令	204
		I-7	CNV_OBJECT	输送带追踪的抓取之物件编号指令	206
		I-8	CNV_FULL	输送带追踪的计数物件数量满额指令	208
		I-9	CNV_EMPTY	输送带追踪的计数物件数量空指令	210
		I-10	CNV_SET_DELAY_TIME	输送带追踪的飞抓/飞放追纵延迟结束时间指令	212
		I-11	CNV_QUEUE_SIZE	输送带追踪的已触发排序但尚未抓取数量指令	214
		I-12	CNV_OBJ_CNT_DIST	输送带追踪的已触发物件位置之编码器相差值指令	215
		I-13	CNV_PLACE_BATCH	输送带追踪的可飞放的最大次数	216
		I-14	CNV_RESET_ENC	输送带追踪的清除外部编码器的计数指令	217
		I-15	CNV_QUEUE_REMOVE	输送带追踪的移除飞抓/飞放暂存区前面的序列指令	218
		I-16	CNV_PICK_ACC	输送带追踪的设定追踪下压加速度时间指令	219
		I-17	CNV_OFFSET_X	输送带追踪的飞抓/飞放的 X 轴位置偏移值指令	220
		I-18	CNV_OFFSET_Y	输送带追踪的飞抓/飞放的 Y 轴位置偏移值指令	221
		I-19	CNV_OFFSET_Z	输送带追踪的飞抓/飞放的 Z 轴位置偏移值指令	222
		I-20	CNV_SPEED	输送带追踪的取得输送带速度指令	223

(表格 4/6)

表 3(续 4)

分类代号	指令分类	指令编号	指令名称	说明	页码
J	讯息提示	J-1	USER_ALARM	触发使用者自定义警报指令	224
K	模拟环境	K-1	ADDTOOL	模拟画面新增末端工具指令	225
		K-2	ADDOBJ	模拟画面新增周边物件指令	228
		K-3	SHOW_TOOL	模拟画面显示/隐藏末端工具指令	230
		K-4	SHOW_OBJ	模拟画面显示/隐藏周边物件指令	232
		K-5	MOVEFLOOR	模拟画面移动地板位置指令	233
		K-6	AXISON	模拟画面显示座标系(轴)指令	234
		K-7	AXISOFF	模拟画面隐藏座标系(轴)指令	235
L	电动夹爪功能	L-1	EG_OPEN	电动夹爪的开启连线指令	236
		L-2	EG_CLOSE	电动夹爪的关闭连线指令	238
		L-3	EG_RESET	电动夹爪的重置原点指令	239
		L-4	EG_GET_STATUS	电动夹爪的状态取得指令	240
		L-5	EG_RUN_MOVE	电动夹爪的位置移动指令	242
		L-6	EG_RUN_GRIP	电动夹爪的夹持动作指令	244
		L-7	EG_RUN_EXPERT	电动夹爪的专家模式，位置移动及夹持动作组合指令	247
		L-8	EG_GOT_POS	电动夹爪的当前位置取得指令	250
M	外部轴功能	M-1	EX_AX	外部轴的开启/关闭连接指令	252
		M-2	EX_AX_ASYNC	外部轴的设定同步轴/非同步轴指令	257
		M-3	EX_AX_SET_ACC	外部轴的设定加/减速度指令	258
		M-4	EX_AX_SET_SPEED	外部轴的设定速度指令	259
		M-5	EX_AX_CT	外部轴的无限旋转速率指令	260
		M-6	ASYPTP	对异步外部轴进行点对点移动指令	262
N	焊接功能	N-1	WELD_START	启动焊接程序指令	264
		N-2	WELD_END	结束焊接程序指令	269
		N-3	WEAVE	焊接的设定摆焊动作指令	270
		N-4	WEAVE_END	结束焊接的摆焊动作指令	270
		N-5	RESUME_PROG	焊接的设定中断复归范围程式指令	277
		N-6	CLEAR_RESUME_PROG	清除焊接的中断复归程式范围指令	277

(表格 5/6)

表 3(续 5)

分类代号	指令分类	指令编号	指令名称	说明	页码
O	碰撞侦测功能	O-1	CD_START	启动碰撞侦测功能与灵敏度设定指令	281
		O-2	CD_END	关闭碰撞侦测功能指令	282
P	字串操作	P-1	STRCMP	字串比较指令	283
		P-2	STRPOS	搜寻字串位置指令	284
		P-3	STPCPY	字串复制指令	286
		P-4	STRLEN	取得字串长度指令	288
		P-5	INTTOSTR	整数型态转成字串型态指令	289
		P-6	REALTOSTR	浮点数型态转成字串型态指令	290
		P-7	STRTOINT	字串型态转成整数型态指令	291
		P-8	STRTOREAL	字串型态转成浮点数型态指令	292
Q	专用功能	Q-1	TL_START	启动扭力极限监控	293
		Q-2	TL_END	结束扭力极限监控	293

(表格 6/6)

将在下面第一部分章节做各指令详细说明。

第二部份功能性副程式所有 HRSS 与 Caterpillar 所扩充的功能性副程式表如下所示，快速连结请点页码编号。

表 4 功能性副程式总表

编号	指令分类	指令代码	功能性副程式名称	说明	页码
1	数学运算功能	FA-1	ABS	数值取绝对值计算	303
2		FA-2	ROUND	四舍五入计算	304
3		FA-3	BIT	位元转换计算	305
4		FA-4	SQRT	开平方根计算	306
5	矩阵功能	FB-1	PAL_PR	定义矩阵功能	307
6		FB-2	PAL_P	指定矩阵点位功能	310
7	区域设定功能	FC-1	BOX	设立检测区	311
8		FC-2	INSIDEBOX	判定检测区	313
9		FC-3	ATPOS	安全区域设定	315
10	点位移动功能	FD-1	JUMP	门型移动	316
11		FD-2	CIRCLE	圆周移动	318
12		FD-3	OVAL	椭圆移动	320

备注：功能性副程式名称可能为函式与副程式，标题统一为功能性副程式。

将在下面第二部分章节做各指令详细说明。

2 第一部份：指令集格式与应用说明

本章节说明程式内部已定义的程式指令作说明，绝大部份透过此章节即可完成基本应用；若进阶功能需求需要功能性副程式说明，请参考下一章节第二部份。

2.1 代码 A、变数&函式宣告指令

此章节针对于代码 A 的变数宣告指令做各别说明。

2.1.1@REAL 浮点数变数指令

■ **指令说明：**

可自行定义变数的指令名称，浮点数方式的储存型态，以自定义的名称来宣告储存设定值，供后续程式所需，程式关闭或断电后，变数数值不保存。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

格式(1)：

```
REAL [自定义变数名称 1]
[自定义变数名称 1] = [自定义的设定初始值或暂存器数值]
```

格式(2)：

```
REAL [自定义变数名称 1] = [自定义的设定初始值或暂存器内数值]
```

格式(3)：

```
REAL [自定义变数名称 1], [自定义变数名称 2],...
[自定义变数名称 1] = [自定义的设定初始值或暂存器内数值]
[自定义变数名称 2] = [自定义的设定初始值或暂存器内数值]
```

格式/参数说明：

⇨ 格式(1)：使用者只定义一个变数，次行再将定义自定义的设定初始值或计数暂存器数值。

⇨ 格式(2)：使用者只定义一个变数，并同时给予设定初始值。

⇨ 格式(3)：使用者定义二个以上变数，使用逗号「，」作分隔，次行后再将定义多行自定义的 设定初始值或计数暂存器数值。

说明：

1. [自定义变数名称 1]、[自定义变数名称 2]为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。

2. [自定义的设定初始值或暂存器内数值] 为浮点数，变数能记录小数点后 6 码，超过 6 位小数将无效数字。(暂存器可以使用计数暂存器、计时暂存器)

3.程式关闭后，变数定义之数值即消失，若需要保留特别值，需使用暂存器$C[]方式。

■ **范例 1：**

```
REAL G1
G1 = 10.2
$C[2] = G1*10
```

■ **范例 1 说明：**

宣告 REAL 变数 G1，并于次行定义 G1 的初始值为 10.2，
第 2 道计数暂存器数值定义为 G1 变数数值并乘上 10(计数暂存器为整数型态)，计数暂存器**结果**：102。

■ **范例 2：**

```
REAL G1 = 10.02
```

■ **范例 2 说明：**

宣告 REAL 变数 G1，并定义 G1 的初始值为 10.02。

■ **范例 3：**

```
REAL X1,Y1,Z1
REAL total
X1 = 0.6
Y1 = 12.2
Z1 = X1 * Y1
total = Z1 + X1
```

■ **范例 3 说明：**

宣告浮点数REAL变数 X1,Y1,Z1 ；宣告REAL 变数 total。
定义变数初始值X1 = 0.6 ; Y1 = 12.2；
Z1变数为X1乘上Y1的结果，(Z1 = 0.6 * 12.2 = 7.32)
total 变数为 Z1 加上 X1 的结果，(total = 7.32 + 0.6 = 7.92)
计算结果 7.92 值存于 total 变数内，供后续程式码使用。

■ **参考相关指令：**

INT、BOOL、$C[]

2.1.2 @INT 整数变数指令

■ **指令说明：**

可自行定义变数的指令名称，整数方式的储存型态，以自定义的名称来宣告储存设定值，供后续程式所需，程式关闭或断电后，变数数值不保存。也可以为十六进位数值(例如：十六进位 0X10 换算十进位为 16)。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

格式(1)：

```
INT [自定义变数名称 1]
[自定义变数名称 1] = [自定义的设定初始值或暂存器数值]
```

格式(2)：

```
INT [自定义变数名称 1] = [自定义的设定初始值或暂存器内数值]
```

格式(3)：

```
INT [自定义变数名称 1], [自定义变数 2],…
[自定义变数名称 1] = [自定义的设定初始值或暂存器内数值]
[自定义变数名称 2] = [自定义的设定初始值或暂存器内数值]
```

■ **格式/参数说明：**

⇨ [自定义变数名称 1]、[自定义变数名称 2]为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。

⇨ [自定义的设定初始值或暂存器内数值]为整数，小数点后数值无条件舍去。(暂存器可以使用计数暂存器、计时暂存器) 备注：计时、计数暂存器的范围数值(十进位)范围：- 2147483648(-2^{31}) ~ 2147483647(2^{31}-1)。

⇨ 程式关闭后，变数定义之数值即消失，若需要保留特别值，请使用暂存器$C[]。

■ **范例 1：**

```
INT G1
G1 = 10
$C[2] = G1
```

■ **范例 1 说明：**

宣告 INT 变数 G1，并于次行定义 G1 的初始值为 10，再次行计数暂存器[2]数值定义为 G1 变数数值。

■ **范例 2：**

INT G1 = 10

■ **范例 2 说明：**

宣告 INT 变数 G1，并定义 G1 的初始值为 10。

■ **范例 3：**

```
INT x1,y1,z1
INT total
x1 = 3
y1 = 2.2
z1 = x1 * y1
total = z1 + 3.14
```

■ **范例 3 说明：**

宣告INT变数 x1,y1,z1 ；宣告INT 变数 total。
定义变数初始值x1 = 3 ; y1 = 2.2
z1变数为x1乘上y1的结果，z1 = 3 * 2 = 6 (整数型态，小数点后数值无条件舍去)
total 变数为z1加上3.14的结果，total = 6 + 3.14 = 9 (整数型态，小数点后数值无条件舍去)
计算结果 9 值存于 total 变数内，供后续程式码使用。

■ **范例 4： (十六进位)**

```
UINT DATA = 0x10
$C[1] = DATA
$C[2] = DATA + 0x10
$C[3] = DATA - 0x10
$C[4] = DATA * 0x10
$C[5] = DATA / 0x10
```

■ **范例 4 说明：**

宣告INT整数变数 DATA。
以第 1 个计数暂存器显示 DATA 变数内数值，**结果为** 16(十进位)。
以第 2 个计数暂存器显示 DATA 变数 加上 0X10 数值，**结果为** 32(十进位)。
以第 3 个计数暂存器显示 DATA 变数 减去 0X10 数值，**结果为** 0(十进位)。
以第 4 个计数暂存器显示 DATA 变数 乘上 0X10 数值，**结果为** 256(十进位)。
以第 5 个计数暂存器显示 DATA 变数 除上 0X10 数值，**结果为** 1(十进位)。

■ **参考相关指令：**

REAL、BOOL、$C[]

2.1.3@BOOL 宣告布林变数指令

■ **指令说明：**

可自行定义变数的指令名称，以布林 BOOL 为真 TRUE 或假 FALSE 逻辑变数状态，二者(真/假)择一状态，以自定义的名称宣告储存真/假设定值，供后续程式所需，程式关闭或断电后，变数数值不保存。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
BOOL [自定义变数名称] =  [布林变数]
```

■ 格式/参数说明：

- [自定义变数名称]为使用者自定义宣告的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。
- [布林变数] 为真 TRUE 或假 FALSE，布林逻辑变数，二者择一状态。
- 程式关闭或断电后，变数定义之数值不保存。

■ **范例 1：**

```
BOOL K = TRUE
```

■ **范例 1 说明：**

布林变数宣告，宣告变数名称 K 且变数值为真 TRUE。

■ **范例 2：**

```
BOOL L = FALSE
IF L == FALSE THEN
    PTP P0
ENDIF
```

■ **范例 2 说明：**

宣告变数名称 L 且变数值为假 FALSE。如果 L 等于假 FALSE/0，点到点方式移动至 P0 点。

■ **参考相关指令：**

IF、INT、REAL

2.1.4@CHAR 宣告字元变数指令

■ **指令说明：**

可自行定义变数的指令名称，以字元变数型态，以自定义的名称宣告字元设定值，供后续程式所需，程式关闭或断电后，变数值不保存。。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
CHAR [自定义变数名称] = ' [自定义字元名称] '
```

■ 格式/参数说明：

⇨ [自定义变数名称]为使用者自定义宣告的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。

⇨ [自定义变数名称]的值由使用者自定义指定一个字元英文字母，并在前后各加上撇号「'」(apostrophe)，例如：'A'，A 为一个字元，英文字母大小写不限且不同；若超出一个字元为字串，目前不支援字串。

⇨ 字元在电脑中是以一个八位元的整数来储存(即 1 Byte)，而这个符号与数字的对应关系可参考 ASCII 码 (American Standard Code for Information Interchange)

⇨ 字元也是一个数字，因此也可以拿来做加减乘除等四则运算，计算后需比照 ASCII 码作对应，例如：ASCII 码数字 65 为 A 字元，ASCII 码数字 97 为 a 字元，请参考附录二、ASCII 可显示字元表。

⇨ 程式关闭或断电后，变数定义之数值不保存。

■ **范例 1：**

```
CHAR WORD = 'A'
IF WORD == 'A' THEN
   PTP P0
ENDIF
WORD = WORD +2
IF WORD == 'C' THEN
   PTP P1
ENDIF
```

■ **范例 1 说明：**

宣告 WORD 字元变数名称且定义初始字元'A'，使用如果判断式，如果 WORD 等于'A'字元，则点到点方式移动至 P0 点。

再来字元数字值增加 2，从'A' 字母 ASCII 码数字 65 变为'C'字母的 ASCII 码数字 67，使用如果判

断式，如果 WORD 等于'C'字元，则点到点方式移动至 P1 点。

■ **范例 2：**

```
CHAR WORD = 'A'
INT G
FOR G= 'A' TO 'Z' STEP 1
   $C[1] =G
   WAIT SEC 2
ENDFOR
```

■ **范例 2 说明：**

- 宣告字元型态 WORD 字元变数名称且定义初始字元'A'，'A' 字母 ASCII 码数字 65。
- 宣告整数型态 G 变数名称。
- FOR 回圈条件从'A' = 65 开始，至'Z' =90 结束，每次间隔距离为 1，执行动作内容为将 G 的数值汇入第一暂存器，并且等待 2 秒钟，共执行 26 次后就结束程式。

■ **参考相关指令：**

IF、FOR、INT

2.1.5@E6POS 宣告直角座标系位置变数指令

■ **指令说明：**

在直角/笛卡尔座标系下使用者自行宣告建立新点(POINT)名称，不在只是 P0~P9999，并善用于移动指令(例:PTP、LIN)移动至点(POINT)或作程式计算。用于宣告座标位置变数，当手臂执行动作时可以依设定座标移动到指定位置。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

E6POS [变数名称]
E6POS [变数名称] = {X <数值 1>,Y <数值 2>,Z <数值 3>,A <数值 4>,B <数值 5>,C <数值 6>}

SCARA-LU 与 DELTA 系列无 AB 轴。

■ **格式/参数说明：**

⇨ E6POS 执行此程式码，即执行宣告位置座标名称。

⇨ [变数名称] 为使用者自定义位置座标变数，设定参数内容座标 X、Y、Z、A、B、C 轴位置座标变数。以{ }内制定格式可直接宣告 X、Y、Z、A、B、C 轴数值，赋与使用者自定义指定位置。(LU 系列无 A、B 轴值)

⇨ <数值 1>到<数值 6>为客户自行定义数值，距离单位为毫米 mm，轴名称与数值需有一格空格，二个轴以上以逗号”,”做分隔；也可以省略不宣告轴数值，只宣告[变数名称]，以利后续程式编辑使用。

⇨ E6POS 没有轴角度点的定义，故无法直接设定 A1~A6 的值，若需要设定 A1~A6 值，请使用 E6AXIS 宣告。(LU 系列无 A5、A6 值)

⇨ E6POS 只针对于设定/记录 X、Y、Z、A、B、C 轴 6 个变数，E6POINT 可以设定/记录 A1、A2、A3、A4、A5、A6、X、Y、Z、A、B、C 轴 12 个变数，可使用变数较多，但使用时尚需先定义每一轴初始值，若一轴没定义则会发生错误。

■ **范例 1：**

```
E6POS HOME = {Y 200,Z -100}
PTP HOME CONT=100% Vel=100% Acc=50% TOOL[0] BASE[0]
```

■ **范例 1 说明：**

- E6POS 执行此程式码，即做宣告位置座标名称。
- [{Y 200,Z -100}] = 宣告座标 Y、Z 位置座标 Y=200mm, Z=-100mm。
- 使用点到点(PTP)移动方式，以圆滑度 100%、速度 100%、加速度 50%、工具座标[0]、基座座标[0]方式，执行到 HOME 点位置。

备注：※座标 X 参数(或其他参数)未设定，旋转 A、B、C 参数未设定，则维持原值(或预设值)。

➢ **范例 2：** (P1 位置增加 10mm 并作移动至新位置)

```
E6POS NEWP1
NEWP1.X=P1.X+10
NEWP1.Y=P1.Y
NEWP1.Z=P1.Z
NEWP1.A=P1.A
NEWP1.B=P1.B
NEWP1.C=P1.C
LIN NEWP1 FINE=1 Vel=1000mm/s Acc=75% TOOL[0] BASE[0]
```

■ **范例 2 说明：**

⇨ E6POS 宣告 NEWP1 新位置座标名称。

⇨ NEWP1 X 轴的方向增加 10mm，NEWP1 位置值参考 P1 的 Y、Z、A、B、C，再使用直线(LIN)移动方式，移动到新 NEWP1 点位置。

■ **参考相关指令：**

E6POINT、E6AXIS

2.1.6@E6AXIS 宣告关节座标系角度变数指令

■ **指令说明：**

在关节座标系(JOINT 座标系)下使用者自行宣告建立新点(POINT)名称，不在只是 P0~P9999，并善用于 PTP 移动指令移动至新角度或作程式计算。通常用于宣告关节角度变数，当手臂执行动作时可以依设定座标移动到指定角度。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

E6AXIS [变数名称]
E6AXIS [变数名称] = {A1 <数值 1> ,A2 <数值 2> ,A3 <数值 3> ,A4 <数值 4> ,A5 <数值 5> ,A6 <数值 6>}

■ **格式/参数说明：**

⇨ E6AXIS 执行此程式码，即执行宣告位置座标名称。

⇨ [变数名称] 为使用者自定义变数名称，设定参数内容每一轴 A1、A2、A3、A4、A5、A6 轴关节座标系(JOINT 座标系)角度变数。以{}内制定格式可直接宣告 A1、A2、A3、A4、A5、A6 轴数值赋与使用者自定义指定角度。(LU 系列无 A5、A6 值)

⇨ <数值 1>到<数值 6>为客户自行定义数值，角度单位为度，轴座标与数值需有一格空格，二个轴以上以”,”(逗号)做分隔；也可以省略不宣告指令数值，只宣告[变数名称]，以利后续程式编辑使用。

⇨ E6AXIS 没有轴座标位置的定义，故无法直接设定 X、Y、Z、A、B、C 的值，若需要设定 X、Y、Z、A、B、C 值，请使用 E6POS 或 E6POINT 宣告。

■ **范例 1：**

```
E6AXIS HOME = {A1 0, A2 0, A3 0, A4 0, A5 -90}
PTP HOME CONT=100% Vel=100% Acc=50% TOOL[0] BASE[0]
```

■ **范例 1 说明：**

- **E6AXIS** 执行此程式码，即做宣告位置角度名称。
- [{**A1** 0, **A2** 0, **A3** 0, **A4** 0, **A5** -90}] = 宣告 A1、A2、A3、A4、A5 位置角度第一轴 0 度、第二轴 0 度、第三轴 0 度、第四轴 0 度、第五轴-90 度。
- 使用点到点(PTP)移动方式，以圆滑度 100%、速度 100%、加速度 50%、工具座标[0]、基座座标[0]方式，执行到 HOME 点位置。

备注：※角度 A6(或其他参数)未设定，则维持原值(或预设值)。

■ **范例 2： (P1 位置增加 10 度并作移动至新角度)**

```
E6AXIS NEWP1
NEWP1.A1=P1.A1+10
NEWP1.A2=P1.A2
NEWP1.A3=P1.A3
NEWP1.A4=P1.A4
NEWP1.A5=P1.A5
NEWP1.A6=P1.A6
PTP NEWP1 FINE=1 Vel=50% Acc=75% TOOL[0] BASE[0]
```

■ **范例 2 说明：**

- E6XIS 宣告 NEWP1 新角度座标名称。
- NEWP1 A1 轴的方向增加 10 度，NEWP1 位置值参考 P1 的 A2、A3、A4、A5、A6，再使用点到点(PTP)移动方式，移动到新 NEWP1 点位置。(假设原本 P1 点位 A1 轴为 5 度，NEWP1 点增加 10 度变为 15 度，其余 A2~A6 轴角度不变)

■ **参考相关指令：**

E6POS 、E6POINT

2.1.7@E6POINT 直角座标系点资讯位置变数指令

■ **指令说明：**

在直角/笛卡尔座标系下使用者自行宣告建立新点(POINT)名称，不在只是 P0~P9999，并善用于移动指令(例:PTP、LIN)移动至点(POINT)或作程式计算。通常用于宣告座标位置变数，当手臂执行动作时可以依设定座标移动到指定位置。E6POINT 为内部程式建立点位使用指令，参数较于 E6POS、E6AXIS 齐全，但建议直角/笛卡尔座标(XYZ 座标系)宣告点位以 E6POINT 或 E6POS 指令，关节座标系(JOINT 座标系)宣告点位以 E6AXIS 指令。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

E6POINT [变数名称]
E6POINT [变数名称] = {X <数值 1>,Y <数值 2>,Z <数值 3>,A <数值 4>,B <数值 5>,C <数值 6>}

SCARA-LU 与 DELTA 系列无 AB 轴。

■ **格式/参数说明：**

⇨ E6POINT 执行此程式码，即执行宣告位置座标名称。

⇨ [变数名称] 为使用者自定义位置座标变数，设定参数内容直角/笛卡尔座标 X、Y、Z、A、B、C 轴位置座标变数及关节座标系 A1~A6。以{ }内制定格式可直接宣告 X、Y、Z、A、B、C 轴数值为主，LU 系列无 A5、A6、A、B 轴值，赋与使用者自定义指定位置，各轴为独立，至少需宣告一个轴。

⇨ <数值 1>到<数值 6>为客户自行定义数值，距离单位为毫米 mm，角度单位为度，轴名称与数值需有一格空格，二个轴以上以逗号”,”做分隔；也可以省略不宣告轴数值，只宣告[变数名称]，以利后续程式编辑使用。

⇨ E6POINT 有轴角度点的定义，但使用时尚需先定义每一轴初始值，若一轴没定义则会发生错误，若需要设定 A1~A6 值，建议使用 E6AXIS 宣告点位。

⇨ E6POS 只针对于设定/记录 X、Y、Z、A、B、C 轴 6 个变数，E6POINT 可以设定/记录 A1、A2、A3、A4、A5、A6、X、Y、Z、A、B、C 轴 12 个变数，可使用变数较多，但使用时尚需先定义每一轴初始值，若一轴没定义则会发生错误。

■ **范例： (宣告一个原点并移动到原点位置)**

```
E6POINT HOME
HOME = {Y 200,Z 300,C 90}
PTP HOME CONT=100% Vel=100% Acc=50% TOOL[0] BASE[0]
```

或

```
E6POINT HOME = {Y 200,Z 300,C 90}
PTP HOME CONT=100% Vel=100% Acc=50% TOOL[0] BASE[0]
```

■ **范例说明：**

E6POINT 执行此程式码，即做宣告位置座标名称。

{Y 200,Z 300,C 90} = 宣告座标 Y、Z、C 位置座标 Y=200mm, Z=300mm, C=90 度。

使用点到点(PTP)移动方式，以圆滑度 100%、速度 100%、加速度 50%、工具座标[0]、基座座标[0]方式，执行到 HOME 点位置。

备注：※座标 X 参数(或其他参数)未设定，则维持原值(或预设值)。

■ **参考相关指令：**

E6POS 、E6AXIS 、GETPOINT

2.1.8@FRAME 宣告(座标系)框架指令

■ **指令说明：**

宣告一个座标系框架名称，框架名称包含有 X、Y、Z、A、B、C 资讯的变数，可用于替代变数值或是工具(TOOL)或基座(BASE)座标系，无法使用在移动点位上。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
FRAME [自定义变数名称]
[自定义变数名称].X = [自定义的设定值或暂存器内数值]
[自定义变数名称].Y = [自定义的设定值或暂存器内数值]
[自定义变数名称].Z = [自定义的设定值或暂存器内数值]
[自定义变数名称].A = [自定义的设定值或暂存器内数值]
[自定义变数名称].B = [自定义的设定值或暂存器内数值]
[自定义变数名称].C = [自定义的设定值或暂存器内数值]
```

■ **格式/参数说明：**

⇨ [自定义变数名称] 为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不能使用「-」。

⇨ [自定义的设定值或暂存器内数值]为浮点数，变数能记录小数点后 6 码，超过 6 位小数将无效数字。

⇨ X 为 X 轴值；Y 为 Y 轴值；Z 为 Z 轴值；A 为对于 X 轴旋转角度值，；B 为对于 Y 轴旋转角度值；C 为对于 Z 轴旋转角度值。

■ **范例：**

```
FRAME BASE_2
BASE_2.X = 0
BASE_2.Y = 200
BASE_2.Z = -100
BASE_2.A = 0
BASE_2.B = 0
BASE_2.C = 45
SET_BASE 2
SET_BASE BASE_2
```

■ **范例说明：**

宣告框架 BASE_2 变数，BASE_2 之 X 轴值 0mm；BASE_2 之 Y 轴值 200mm；BASE_2 之 Z 轴值-100mm；BASE_2 之 A 轴值 0 度；BASE_2 之 C 轴值 0 度；BASE_2 之 C 轴值 45 度。

设定要改变第 2 个 BASE 座标系，将 BASE_2 框架参数汇入至第 2 个 BASE 座标系。

■ **参考相关指令：**

TOOL、BASE。

2.1.9@STRUC 宣告结构定义指令

■ **指令说明：**

结构 (structure)是一种复合型别(derived data type)，用来表达由 1 个以上属性组成的型别，由使用者自订之资料型态，再透过宣告(Declaration) 来描述结构可应用的内容，让一个较为复杂的内容，可以容易区分。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

STRUC [自定义变数] [结构型态 1] [成员名称 1], [结构型态 2] [成员名称 2] …
DECL [自定义变数] [结构变数 1], [结构变数 1]变
[结构变数 1] ={[成员名称 1] <数值 1>, [成员名称 2] <数值 2>,>}
[结构变数 1] ={[成员名称 1] <数值 1>, [成员名称 2] <数值 2>,>}
…

■ **格式/参数说明：**

⇨ [自定义变数]为使用者自定义宣告的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。

⇨ [结构型态] 为搭配[成员名称]的格式型态，如 INT 正整数、REAL 浮点数型态。

⇨ [成员名称] 为使用者自定义宣告的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」

⇨ [结构变数 1] 为使用者自定义宣告的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」

■ **范例：**

```
STRUC GROUP INT Age, REAL Weight
DECL GROUP people1,people2
people1 = {Age 50, Weight 50.3 }
people2 = {Age 60, Weight 70.1}
$C[2] = people1.Weight
```

■ **范例说明：**

宣告 GROUP 结构式，由 Age 成员为整数型态与 Weight 成员为浮点数型态组成。

宣告GROUP结构式变数内容有people1与people2两位，

people1 变数是 Age 成员为 50，Weight 成员为 50.3

people2 变数是 Age 成员为 60，Weight 成员为 70.1

将 people1 变数的 Weight 成员参数值存入第 2 个暂存器，其结果为 50.3。

■ **参考相关指令：**

INT、REAL

2.1.10 @DEF... END 宣告副程式指令

■ **指令说明：**

在一个程式是个需要重复被使用的指令群组(1 行以上)，将可透过指令宣告方式将多个指令组成一个副程式当需要用到时，在主程式做呼叫指令群组即可；副程式只会作出输入并执行，并不会作数值回传，若有需要作回传需使用函式(DEFFCT…ENDFCT)。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
[自定义副程式名称]( [输入副程式参数值] )
DEF [自定义副程式名称]([输入副程式参数值])
    [动作指令]
END
```

备注：[自定义副程式名称]后的括号()为必要符号，不可省略，括号()内可不输入副程式参数值。

■ **格式/参数说明：**

⇨ [自定义副程式名称] 为使用者自行定义的副程式标签名称，需为英文字母或英文字母加数字的组合，不可为数字开头，且字母中间不能有空白，可用”_”(底线)来取代，例如：MySub 即为一个自定一标签，程式呼叫副程式直接输入自定义标签即可执行呼叫。

⇨ [输入副程式参数值] 为将使用者欲带入副程式作计算的数值或点位资讯，可忽略不作输入或 1 个以上的输入资讯，若多个输入值在中间以”,”(逗号)作分隔，最多五个数值；输入参数值需加上:IN，说明为输入数值，例如：MySub(num1:IN)。输入参数值可使用整数格式(INT)或点位格式(E6POINT)。如果参数为输入，则使用 parameter:IN 表示，作为输入的参数即使在函式中被修改，也不会影响传入的变数。如果为输出，则使用 parameter:OUT。作为输出的变数，如果在函式中被修改，则原本传入的变数也会随着改变。

⇨ [自定义标签]后的括号()为必要符号，不可省略，括号()内可不输入副程式参数值。

⇨ [动作指令] 为执行 副程式定义之陈述动作，逻辑程式、移动指令或讯号指令之组合。

■ **范例 1： (省略输入值的副程式范例)**

```
PTP P0   ;主程式
PICK()

DEF PICK () ;副程式
    PTP P1
    LIN P2 FINE = 1
    $DO[1] = TRUE
    WAIT SEC 1
END
```

■ **范例 1 说明：**

点到点移动至 P0 点(原点)，并执行副程式 PICK()内容。

副程式 PICK()内容如下：

点到点移动至P1点(取料点上方)，直线移动至P2点(取料点)，开启第1个数位输出讯号($DO[1] = TRUE)，等待1秒钟。 执行完副程式后，会再回到主程式，并继续后面的主程式内容。

■ **范例 2： (有输入值的副程式范例)**

```
;主程式
PTP P0
PICK(30)

DEF PICK(rise_Dis:IN) ;副程式
    INT rise_Dis
    PTP P1
    LIN P2 FINE = 1
    $DO[1] = TRUE
    WAIT SEC 1
    P2.Z = P2.Z + rise_Dis
    LIN P2
END
```

■ **范例 2 说明：**

点到点移动至 P0 点(原点)，并执行副程式 PICK()内容，其中代入副程式输入数值 30。

副程式 PICK(rise_Dis:IN)内容如下：

(rise_Dis:IN)为输入改变高度的数值变化，此范例为输入 30 数值

点到点移动至P1点(取料点上方)，直线移动至P2点(取料点)，开启第1个数位输出讯号($DO[1] = TRUE)，等待1秒钟。 将P2点Z轴方向迭加30mm的距离，再移动至新的P2点位。

执行完副程式后，会再回到主程式，并继续后面的主程式内容。

■ **范例 3： (有输入点位值的副程式范例)**

```
;主程式
PTP P0
changeposition1( P1 , -100)
changeposition1( P2 , 50)
LIN P1
LIN P2

DEF changeposition1( changepos:OUT, rise_Dis:IN) ;副程式
   E6POINT changepos
```

```
    INT rise_Dis
    changepos.Z = changepos.Z + rise_Dis
END
```

■ **范例 3 说明：**

此为透过副程式方式，将输入的点位，增加/减少距离的程式应用范例。

点到点方式移动至 P0 点后，执行 changeposition1 副程式对于 P1 点，距离为-100mm，在执行 changeposition1 副程式对于 P2 点，距离为 50mm，直线移动方式移动至新 P1 点位，直线移动方式移动至新 P2 点位。

副程式 changeposition1(changepos:OUT, rise_Dis:IN)内容如下：

(changepos:OUT, rise_Dis:IN)中 changepos 为输入&输出的资讯，rise_Dis 为输入上升距离，宣告点位资讯 changepos 变数，宣告整数资讯 rise_Dis 变数，将 changepos 点变数 Z 轴方向加上 rise_Dis 距离。

■ **参考相关指令：**

INT、$DO[]、DEFFCT…ENDFCT、E6POINT、EXT、EXTFCT。

2.1.11 @DEFFCT…ENDFCT 宣告功能函式指令

■ **指令说明：**

在一个程式是个需要重复被使用的指令群组(1 行以上)，将可透过指令宣告方式将多个指令组成一个函式，当需要用到时，在主程式做呼叫指令群组即可；函式可以执行多行指令，并且最后需作回传(RETURN)资料给主程式；若不需要作回传可使用副程式指令(DEF…END)。。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
DEFFCT [回传资料型态] [自定义函式名称] ([输入/输出函式参数值])
    [动作指令]
    RETURN [回传资料]
ENDFCT
```

备注：[自定义函式名称]后的括号()为必要符号，不可省略，括号()内可不输入/输出函式参数值。

■ **格式/参数说明：**

⇨ [回传资料型态] 为函式回传的资料型态定义宣告，例如：INT、E6POINT 等变数宣告格式。

⇨ [自定义函式名称] 为使用者自行定义函式的标签名称，需为英文字母或英文字母加数字的组合，不可为数字开头，且字母中间不能有空白，可用”_”(底线)来取代，例如：MySub 即为一个自定一标签，程式呼叫副程式直接输入自定义标签即可执行呼叫。

⇨ [输入/输出函式参数值] 为将使用者欲带入函式作计算的数值或点位资讯，可忽略不作输入或 1 个以上的输入资讯，若多个输入值在中间以”,”(逗号)作分隔，最多五个数值，函式也可以没有函式参数；参数的资料型态会参考宣告在[动作指令]里面的资料型态。输入参数值需加上:IN，说明为输入数值，例如：MyFunction(num1:IN)。输入参数值可使用整数格式(INT)或点位格式(E6POINT)。如果参数为输入，则使用 parameter:IN 表示，作为输入的参数即使在函式中被修改，也不会影响传入的变数。如果为输出，则使用 parameter:OUT。作为输出的变数，如果在函式中被修改，则原本传入的变数也会随着改变。

⇨ [动作指令] 为执行 副程式定义之陈述动作，逻辑程式、移动指令或讯号指令之组合。

⇨ [回传资料] 为执行完整个函式后，回传给主程式的资讯。可为数值或点资讯。

■ **范例 1： (两数值相加函式)**

```
INT SUM_value   ;主程式
SUM_value = SUM(2,3)
DEFFCT INT SUM(value1:IN, value2:IN)   ;函式
    INT value1, value2
    RETURN value 1+ value2
ENDFCT
```

■ **范例 1 说明：**

宣告整数变数 SUM_value，执行函式 SUM()，并将回传值传递 SUM_value 变数，输入数值为 2,3，结果变数 SUM_value 为 5。

函式 SUM(value1:IN, value2:IN)内容如下：

(value1:IN, value2:IN)为输入改变数值 1 与数值 2，此范例为输入 2 及 3 数值。

宣告整数变数 value1, value2。

回传资料为 value1 相加 value2 之值，回传给主程式之 SUM_value 变数。

■ **范例 2： (两点位相减函式)**

```
E6POINT Tolerance_point   ;主程式
Tolerance_point=change2(P1 ,P2)
IF Tolerance_point.Z > 10 THEN
    $DO[1] = TRUE
ENDIF

DEFFCT   E6POINT change2(POINT_A:IN, POINT_B:IN) ;函式
    E6POINT POINT_A,POINT_B,POINT_C
    POINT_C.Z =POINT_A.Z -POINT_B.Z
    RETURN POINT_C
ENDFCT
```

■ **范例 2 说明：**

宣告点位变数 Tolerance_point，执行函式 change2，计算两点位 P1 与 P2 点的 Z 轴距离差距，若差距超过 10mm 则开启第 1 个数位输出讯号($DO[1] = TRUE)。

函式 change2(POINT_A:IN, POINT_B:IN)内容如下：

(POINT_A:IN, POINT_B:IN)为输入点位 A 与点位 B，此范例为输入 P1 与 P2 点位。

宣告点位变数 POINT_A,POINT_B,POINT_C。

计算 POINT_A 与 POINT_B 的 Z 轴资讯相减，并将相减数值传给 POINT_C 变数。

最后回传 POINT_C 点位资讯给主程式，让 POINT_C 点位等于 Tolerance_point 点位。

■ **参考相关指令：**

INT、$DO[]、DEFFCT…ENDFCT、E6POINT、EXT、EXTFCT。

2.1.12 @EXT 宣告外部副程式

■ **指令说明：**

宣告外部副程式，代表使用者将此副程式内容写在另一个独立的档案中，当主程式需要时，在呼叫另一个档案的副程式内容作使用，可更容易确认程式分段架构，并容易寻找副程式内容。

宣告外部副程式的作法如下：

1) 主程式内当案需使用 EXT+副程式名称宣告格式内容，代表主程式可以呼叫副程式，一个主程式可以呼叫一个以上副程式，每个副程式皆需用 EXT 作宣告，宣告后，即可如同呼叫一般副程式。
2) 独立的档案名称需与副程式名称相同，并在此独立档案外的地方呼叫此副程式。
3) 外部副程式在程式码的第一行必须以 DEF 关键字为开头并 END 结尾，在副程式内容中写入动作指令，END 结尾以外不能在加入其他动作指令，一个档案只能定义一个外部副程式。
4) EXT 指令为执行程式后无回传任何数值(RETURN)，如须回传数值须使用 EXTFCT 指令。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

主程式内容：

EXT [自定义副程式名称]([输入副程式参数值])
[自定义副程式名称]

副程式内容(档案名称需与自定义副程式名称一样)：

DEF [自定义副程式名称]([输入副程式参数值])
    [动作指令]
END

备注：[自定义副程式名称]后的括号()为必要符号，不可省略，括号()内可不输入副程式参数值。

■ **格式/参数说明：**

⇨ [自定义副程式名称] 为使用者自行定义副程式的标签名称，需为英文字母或英文字母加数字的组合，不可为数字开头，且字母中间不能有空白，可用"_"(底线)来取代，例如：MySub 即为一个自定一标签，程式呼叫副程式直接输入自定义标签即可执行呼叫。

⇨ [输入副程式参数值] 为将使用者欲带入副程式作计算的数值或点位资讯，可忽略不作输入或 1 个以上的输入资讯，若多个输入值在中间以","(逗号)作分隔，最多五个数值；输入参数值需加上:IN，说明为输入数值，例如：MySub(num1:IN)。输入参数值可使用整数格式(INT)或点位格式(E6PINT)。如果参数为输入，则使用 parameter:IN 表示，作为输入的参数即使在函式中被修改，也不会影响传入的变数。如果为输出，则使用 parameter:OUT。作为输出的变数，如果在函式中被修改，则原本传入的变数也会随着改变。

⇨ [动作指令] 为执行 副程式定义之陈述动作，逻辑程式、移动指令或讯号指令之组合。

■ **范例：**

主程式内容：

```
EXT PICK(rise_Dis:IN)
PTP P0
PICK(30)
```

副程式内容(档案名称：PICK)：

```
DEF PICK(rise_Dis:IN)
    INT rise_Dis
    PTP P1
    LIN P2 FINE = 1
    $DO[1] = TRUE
    WAIT SEC 1
    P2.Z = P2.Z + rise_Dis
    LIN P2
END
```

■ **范例说明：**

主程式内容：

宣告会利用到外部副程式 PICK(rise_Dis:IN)

点到点移动至 P0 点(原点)，并执行副程式 PICK()内容，其中代入副程式输入数值 30。

副程式内容：

副程式档案名称为 PICK，宣告副程式 PICK(rise_Dis:IN)：

(rise_Dis:IN)为输入改变高度的数值变化，此范例为输入 30 数值

点到点移动至P1点(取料点上方)，直线移动至P2点(取料点)，开启第1个数位输出讯号($DO[1] = TRUE)，等待1秒钟。 将P2点Z轴方向迭加30mm的距离，再移动至新的P2点位。

执行完副程式后，会再回到主程式，并继续后面的主程式内容。

■ **参考相关指令：**

INT、$DO[]、DEF…END、DEFFCT…ENDFCT、E6POINT、EXTFCT。

2.1.13 @EXTFCT 宣告外部函式

■ **指令说明：**

宣告外部函式，代表使用者将此函式内容写在另一个独立的档案中，当主程式需要时，在呼叫另一个档案的函式内容作使用，可更容易确认程式分段架构，并容易寻找函式内容。

在一个程式是个需要重复被使用的指令群组(1 行以上)，将可透过指令宣告方式将多个指令组成一个函式，当需要用到时，在主程式做呼叫指令群组即可；函式可以执行多行指令，并且最后需作回传(RETURN)资料给主程式；若不需要作回传可使用副程式指令(DEF…END)。。

宣告外部函式的作法如下：

1) 主程式内当案需使用 EXTFCT+函式名称宣告格式内容，代表主程式可以呼叫函式，一个主程式可以呼叫一个以上函式，每个函式皆需用 EXTFCT 作宣告，宣告后，即可如同呼叫一般函式。
2) 独立的档案名称需与函式名称相同，并在此独立档案外的地方呼叫此函式。
3) 外部函式在程式码的第一行必须以 DEFFCT 关键字为开头并 ENDFCT 结尾，在函式内容中写入动作指令，ENDFCT 结尾以外不能在加入其他动作指令，一个档案只能定义一个外部函式。
4) EXTFCT 指令为执行程式后会回传数值(RETURN)，如无须回传数值须使用 EXT 指令。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

主程式内容：

EXTFCT [回传资料型态] [自定义函式名称]([输入/输出函式参数值])
[自定义函式名称]

函式内容(档案名称需与自定义函式名称一样)：

DEFFCT [回传资料型态] [自定义函式名称] ([输入/输出函式参数值])
    [动作指令]
    RETURN [回传资料]
ENDFCT

备注：[自定义函式名称]后的括号()为必要符号，不可省略，括号()内可不输入/输出函式参数值。

■ **格式/参数说明：**

⇨ [回传资料型态] 为函式回传的资料型态定义宣告，例如：INT、E6POINT 等变数宣告格式。
⇨ [自定义函式名称] 为使用者自行定义函式的标签名称，需为英文字母或英文字母加数字的组合，不可为数字开头，且字母中间不能有空白，可用"_"(底线)来取代，例如：MySub 即为一个自定一标签，程式呼叫副程式直接输入自定义标签即可执行呼叫。
⇨ [输入/输出函式参数值] 为将使用者欲带入函式作计算的数值或点位资讯，可忽略不作输入

或 1 个以上的输入资讯，若多个输入值在中间以”,”(逗号)作分隔，最多五个数值，函式也可以没有函式参数；参数的资料型态会参考宣告在[动作指令]里面的资料型态。输入参数值需加上:IN，说明为输入数值，例如：MyFunction(num1:IN)。输入参数值可使用整数格式(INT)或点位格式(E6PINT)。如果参数为输入，则使用 parameter:IN 表示，作为输入的参数即使在函式中被修改，也不会影响传入的变数。如果为输出，则使用 parameter:OUT。作为输出的变数，如果在函式中被修改，则原本传入的变数也会随着改变。

⇨ [动作指令] 为执行 副程式定义之陈述动作，逻辑程式、移动指令或讯号指令之组合。

⇨ [回传资料] 为执行完整个函式后，回传给主程式的资讯。可为数值或点资讯。

■ **范例：**

主程式内容：

```
EXTFCT E6POINT change2(POINT_A:IN, POINT_B:IN)
E6POINT Tolerance_point
Tolerance_point=change2(P1 ,P2)
IF Tolerance_point.Z > 10 THEN
    $DO[1] = TRUE
ENDIF
```

副程式内容(档案名称：change2)：

```
DEFFCT E6POINT change2(POINT_A:IN, POINT_B:IN)
    E6POINT POINT_A,POINT_B,POINT_C
    POINT_C.Z =POINT_A.Z -POINT_B.Z
    RETURN POINT_C
ENDFCT
```

■ **范例说明：**

主程式内容：

宣告会利用到外部函式 change2(POINT_A:IN, POINT_B:IN)。

宣告点位变数 Tolerance_point，执行函式 change2，计算两点位 P1 与 P2 点的 Z 轴距离差距，若差距超过 10mm 则开启第 1 个数位输出讯号($DO[1] = TRUE)。

函式内容：

函式档案名称为 change2，宣告函式 change2(POINT_A:IN, POINT_B:IN)。

(POINT_A:IN, POINT_B:IN)为输入点位 A 与点位 B，此范例为输入 P1 与 P2 点位。

宣告点位变数 POINT_A,POINT_B,POINT_C。

计算 POINT_A 与 POINT_B 的 Z 轴资讯相减，并将相减数值传给 POINT_C 变数。

最后回传 POINT_C 点位资讯给主程式，让 POINT_C 点位等于 Tolerance_point 点位。

■ **参考相关指令：**

INT、$DO[]、DEF…END、DEFFCT…ENDFCT、E6POINT、EXT。

2.1.14 @STRING 宣告字串指令

■ **指令说明：**

宣告字串文字之变数，并后续可做字串表达、分类应用，或做多个字串结合转换使用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.5 以上	Caterpillar1.0.4 以上	预读 CONT

■ **指令格式：**

```
STRING [字串名称] =  " [字串文字内容] "
```

■ **格式/参数说明：**

⇨ [字串名称]为使用者自订义字串名称，

⇨ [字串文字内容]为使用者自订义字串文字，例如 abc1，为自定义的文字内容；特别注意字串的内容前后需加上双引号「"」符号。(若用复制手册文字，部份产生错误格式，建议重新修改双引号「”」才不会导致有执行错误产生)。

■ **范例：**

```
STRING first_word = "Wang"
STRING second_word = "ming"
STRING Name
Name = first_word + second_word
```

■ **范例说明：**

宣告字串变数 first_word，字串变数内容文字为 "Wang"。

宣告字串变数 second_word，字串变数内容文字为 "ming"。

宣告字串变数 Name，Name 变数文字内容为 first_word 字串加上 second_word 字串内容文字组合而成。结果为"Wangming"。

■ **参考相关指令：**

STRCMP、STRPOS、STRCPY、STRLEN、INTTOSTR、REALTOSTR、STRTOINT、STRTOREAL。

2.1.15 ARRAY 宣告阵列指令

■ **指令说明：**

阵列(Array)是一组相同型态的连续变数，需先作阵列型态、阵列名称与阵列大小个数宣告，在同一个阵列名称下，透过一个索引值（Index）来指定使用第几个变数，而对应相对的储存变数值；此指令只有宣告格式与应用格式，无固定指令字传名称(故标题不加@)。

可分为一维阵列与多维阵列(2 维以上)，一维（或单维）阵列是一种线性阵列，例如：

INT K[4]，代表 K 阵列有 4 个整数变数值，分别是 K[1]、K[2]、K[3]、K[4]，参考图 5；多维阵列是平面或空间上的数值计算或图型应用可以使用，例如：INT F[2][3]，代表 2 维 F 阵列是 6 (2*3) 整数变数值，分别是 F[1][1]、F[1][2]、F[1][3]、F[2][1]、F[2][2]、F[2][3]，参考图 6，可更灵活双列计算应用。

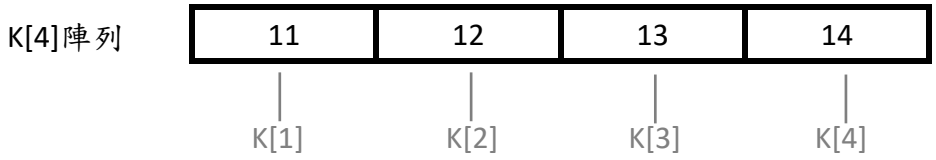


图 5 一维 K 阵列的示意图(储存变数值 11~14)

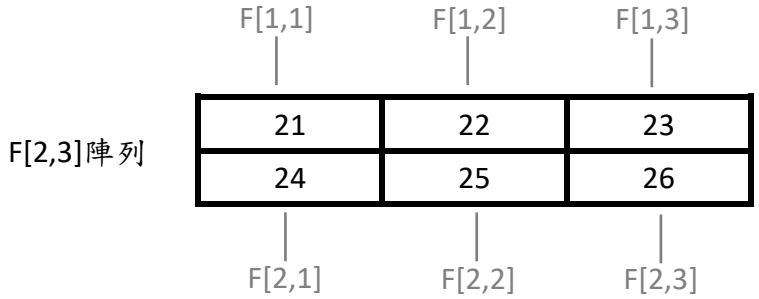


图 6 二维 F 阵列的示意图(储存变数值 21~26)

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.10 以上	Caterpillar1.0.8 以上	预读 CONT

■ **指令格式：**

格式(1) 一维矩阵

[变数型态] [阵列名称] [[索引值个数]]

[TYPE] [NAME] [[Index Quantity]]

阵列名称[[索引值]] = [自定义变数名称] 或 [自定义的设定数值]

格式(2) 多维矩阵

[变数型态] [阵列名称] [[索引值个数 1], [索引值个数 2],…]

[TYPE] [NAME] [[Index Quantity1], [Index Quantity2],…]

阵列名称[[索引值 1], [索引值 2], …] = [自定义变数名称] 或 [自定义的设定数值]

注： [ ]为阵列名称格式必要符号。

■ **格式/参数说明：**

⇨ [变数型态] TYPE 为使用阵列变数的型态规格，支援 INT、REAL、CHAR、E6POS、E6POINT、E6AXIS 等变数型态使用阵列。

⇨ [阵列名称] NAME 为自用者自定义阵列名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。

⇨ [自定义变数名称] 为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。

⇨ [自定义的设定数值] 为阿拉伯数字，范围为-2147483648(-2^{31})~2147483647。

⇨ [索引值个数]、[索引值个数 1]、[索引值个数 2] Index Quantity 为索引值个数量，代表阵列所宣告的变数数量，可填入阿拉伯数字或[自定义变数名称]，阿拉伯数字不可为 0。

⇨ [索引值]、[索引值 1]、[索引值 2] Index 为阵列对应索引数值，可填入阿拉伯数字或[自定义变数名称]，阿拉伯数字不可为 0。

⇨ 可以将阵列类似变数使用，当为赋值(阵列放置于等号左方)与取值(阵列放置于等号右方)使用。

⇨ 可透过引数传递的方式，将整个阵列传入副程式或副函数，无法透过函数 RETURN 的方式回传整个阵列。

■ **范例 1： (一维范例 1 Dimension，自定义的设定数值)**

```
REAL K [5]
K[1] = 9.8
```

■ **范例 1 说明：**

宣告浮点数型态阵列变数值，阵列名称 K 并有 5 个索引值个数，
K 阵列第 1 个索引值的变数值为 9.8。

■ **范例 2： (一维范例 1 Dimension，自定义的变数名称)**

```
INT K [5],U1,W1
U1 = 1
W1 = 9
K[U1] = W1
```

■ **范例 2 说明：**

宣告整数型态阵列变数值，阵列名称 K 并有 5 个索引值个数，宣告 U1、W1 整数型态变数。
变数 U1 初始值为 1，变数 W1 初始值为 9，
K 阵列第 U1 个索引值的变数值为 W。 **结果：**K 阵列第 1 个索引值的变数值为 9

■ **范例 3： (二维范例 2 Dimension)**

```
REAL F[3,4]
F[1,1] =10.1
$C[3] = F[1,1] *10
```

■ **范例 3 说明：**

宣告浮点数型态二维阵列变数值，阵列名称 F 为 3 列 4 行共有 12 个索引值个数，
F 阵列第 1 列 1 行索引值的变数值为 10.1。
第 3 个计数暂存器显示 F 阵列第 1 列 1 行索引值的变数值乘上 10。
第 3 个计数暂存器**结果**：101 (10.1*10)。

■ **参考相关指令：**

INT、REAL、CHAR、E6POS、E6POINT、E6AXIS。

2.1.16 @UINT 宣告无符号(正)整数变数指令

■ **指令说明：**

可自行定义变数的指令名称，以无符号整数(unsigned)方式的储存型态，32 位元的无符号整数，只能表示非负数数值，即是 0 及正数数，以自定义的名称来宣告储存设定值，供后续程式所需，也可以为十六进位数值(例如：十六进位 0X10 换算十进位为 16)，程式关闭或断电后，变数数值不保存，此指令常应用于在于其他设备通讯计算时使用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.12 以上	Caterpillar1.0.10 以上	预读 CONT

■ **指令格式：**

格式(1)：

```
UINT [自定义变数名称 1]
[自定义变数名称 1] = [自定义的设定初始值或暂存器数值]
```

格式(2)：

```
UINT [自定义变数名称 1] = [自定义的设定初始值或暂存器内数值]
```

格式(3)：

```
UINT [自定义变数名称 1], [自定义变数 2],...
[自定义变数名称 1] = [自定义的设定初始值或暂存器内数值]
[自定义变数名称 2] = [自定义的设定初始值或暂存器内数值]
```

■ **格式/参数说明：**

- ⇨ [自定义变数名称 1]、[自定义变数名称 2]为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。
- ⇨ [自定义的设定初始值或暂存器内数值] 为无符号(正)整数值，数值(十进位)范围：0 ~ 4294967295(2^{32}-1)，若为负数值程式将判定错误讯息，暂存器内数值若内小数点后数值无条件舍去。(暂存器可以使用计数暂存器、计时暂存器)。备注：计时、计数暂存器的范围数值(十进位)范围：- 2147483648(-2^{31}) ~ 2147483647(2^{31}-1)，无法全部显示 UINT 内数值，使用者需自行换算。
- ⇨ 自定义 Array 阵列也可以宣告为 16 进位表示，例如：UINT DATA[3]。
- ⇨ 程式关闭后，变数定义之数值即消失，若需要保留特别值，请使用$C[]计数暂存器。

■ **范例 1：**

```
UINT G1
G1 = 10
$C[2] = G1
```

■ **范例 1 说明：**

宣告 UINT 无符号(正)整数变数 G1，并于次行定义 G1 的初始值为 10，再次行第 2 个计数暂存器($C[2])数值定义为 G1 变数数值，**结果为** 10。

■ **范例 2：**

```
UINT G2 = 10
```

■ **范例 2 说明：**

宣告 UINT 无符号(正)整数变数 G2，并定义 G2 的初始值为 10。

■ **范例 3：**

```
UINT G3,H3,F3
G3 = 10
H3 = 5
F3 = G3 + H3
```

■ **范例 3 说明：**

宣告 UINT 无符号(正)整数变数 G3,H3,F3，并定义 G3 的初始值为 10、H3 的初始值为 5。
F3 为 G3 加上 H3 的数值，F3 **结果为** 15。

■ **范例 4： (十六进位)**

```
UINT DATA = 0x10
$C[1] = DATA
$C[2] = DATA + 0x10
$C[3] = DATA - 0x10
$C[4] = DATA * 0x10
$C[5] = DATA / 0x10
```

■ **范例 4 说明：**

宣告UINT无符号(正)整数变数 DATA。
以第 1 个计数暂存器显示 DATA 变数内数值，**结果为** 16(十进位)。
以第 2 个计数暂存器显示 DATA 变数 加上 0X10 数值，**结果为** 32(十进位)。
以第 3 个计数暂存器显示 DATA 变数 减去 0X10 数值，**结果为** 0(十进位)。
以第 4 个计数暂存器显示 DATA 变数 乘上 0X10 数值，**结果为** 256(十进位)。
以第 5 个计数暂存器显示 DATA 变数 除上 0X10 数值，**结果为** 1(十进位)。
备注：计时、计数暂存器的范围数值(十进位)范围:- 2147483648(2^{31}) ~ 2147483647(2^{31}-1)。

■ **参考相关指令：**

INT、REAL、$C[]、$T[]

2.2 代码 B、运动指令

此章节针对于代码 B 的移动方式指令做各别说明

2.2.1@PTP 点到点移动指令

■ **指令说明：**

当前位置为启始点，指定下一点后，以点到点绝对位置方式移动，各轴马达自行运算旋转角度，同时启动并同时到达，路径不为直线，此方式不会产生奇异点。

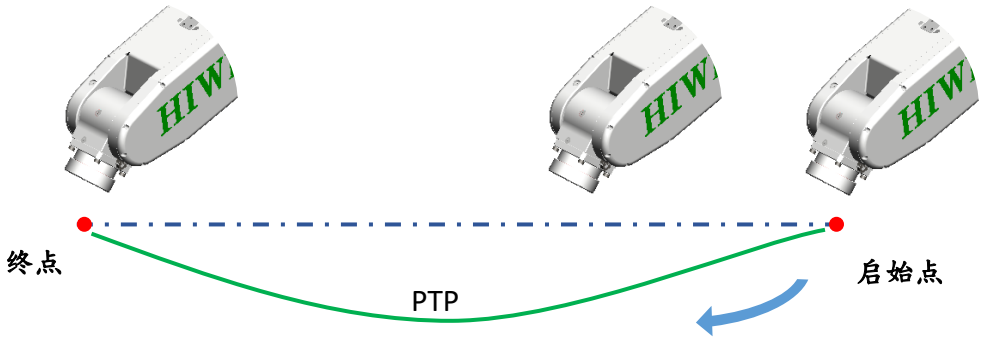


图 7 PTP 路径示意图(绿色实线路径)

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
PTP [点位名称] [轨迹形式        ] [速度 ] [加速度] [工具座标] [基座座标]
PTP [    P#       ] [FINE=#/CONT=#] [Vel=#] [Acc =#] [TOOL[#] ]    [BASE[#] ]
```

■ **格式/参数说明：**

⇨ #为数字，由使用者填入。

⇨ [点位名称]为点到点名称，配合点位视窗作教点位 P1~P9999，取得欲到达下一点点位，或根据使用者配合指令 E6POINT 命名方式自行定义点位、或直接给予各轴角度、位置数值。

⇨ [轨迹形式]为连续 CONT 或非连续 FINE 类型， 填入参数 CONT、FINE；并可在连续轨迹形式后面的百分比(1%~100%)为平滑的程度，决定连续参数；非连续型式 FINE 分成三个等级:。

- FINE (=0)，会预读接续的指令，不会执行到位检查。
- FINE = 1 ，不会预读接续的指令，不会执行到位检查。
- FINE = 2 ，不会预读接续的指令，会执行到位检查。

⇨ CONT、CONT = #%、CONT = #mm、FINE = #、省略轨迹形式等参数请择一使用。

⇨ [速度] Vel 为单行速度参数，填入参数 Vel=1~100%，可省略此参数，省略速度将参考上一行指令，或初始定义参数。

⇨ [加速度] Acc 为单行加速度参数，填入参数 Acc=1~100%，数字越大表示加速度越快， 可省略此参数，省略加速度将参考上一行指令，或初始定义参数。

⇨ [工具座标] TOOL[#]为工具座标参数，可设定不同的末端工具位置控制，#可输入范围为 0 至 15，可省略此参数，省略工具座标将参考上一行指令，或初始定义参数。

⇨ [基座座标] BASE[#]为 BASE 为基底座标参数，可设定为有进行输送带校正的基底编号，#可输入范围为 0 至 31，可省略此参数，省略基座座标将参考上一行指令，或初始定义参数。

⇨ 至少需撰写 PTP + [点位名称]，Vel、Acc、TOOL、BASE 参数可忽略输入。

■ **范例 1：**

```
PTP P0 CONT=100% Vel=100% Acc=50% TOOL[0] BASE[0]
```

■ **范例 1 说明：**

点到点方式移动到下一点位 P0，连续轨迹 100%方式移动，速度 100%，加速度 50%，第 0 组工具座标，第 0 组基座座标。

■ **范例 2：**

```
E6POS POINT = {X 0,Y 300,Z 200}
PTP POINT CONT=100% Vel=100% Acc=50% TOOL[0] BASE[0]
```

■ **范例 2 说明：**

使用 E6POS 宣告 POINT 点位为 X 轴 0mm、Y 轴 300mm、Z 轴 200mm 位置，点到点方式移动到下一点位 POINT 位置，连续轨迹 100%方式移动，速度 100%，加速度 50%，第 0 组工具座标，应用第 0 组基座座标。

■ **范例 3：**

```
PTP {X 100} FINE=1 Vel=100% Acc=50% TOOL[1] BASE[1]
```

■ **范例 3 说明：**

点到点方式移动到绝对位置 X 轴 100mm，不预读，不到位检查，速度 100%，加速度 50%，第 1 组工具座标，应用第 1 组基座座标。

■ **范例 4：**

```
PTP {A1 45} FINE=2 Vel=100% Acc=50% TOOL[0] BASE[0]
```

■ **范例 4 说明：**

点到点方式移动到下一点位，第一轴(A1 轴)增加 45 度，不预读，会执行到位检查，连续轨迹 100%方式移动，加速度 50%，第 0 组工具座标，应用第 0 组基座座标。

■ **参考相关指令：**

E6POINT、E6POS、E6AXISPTP、PTP_REL、LIN、LIN_REL、CIRC、CIRC_REL。

2.2.2@PTP_REL 相对位置方式点到点移动指令

■ 指令说明：

机器手臂沿点到点路径移动至相对增量值之位置。

例如：依前一点为位置相对点到点移动，令机器手臂参考 A1 轴 方向相对增加 10 度。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

PTP_REL{ [各轴增量参数] } [轨迹形式] [速度] [加速度] [工具座标] [基座座标]

PTP_REL{ [A1 num, A2 num,…]} [FINE=#/CONT=#] [Vel=#] [Acc =#] [TOOL[#]] [BASE[#]]

※各轴增量参数内容：{A1 num, A2 num, A3 num, A4 num, A5 num, A6 num}，num 表示数值，使用者自行填入，单位为度，LU 系列无 A5、A6 轴。

■ **格式/参数说明：**

⇨ #为数字，由使用者填入。

⇨ [各轴增量参数] 为增量参数数值，可选择填入各轴角度值(A1~A6)，单位为度，，LU 系列无 A5、A6 轴。

⇨ [轨迹形式]为连续 CONT 或非连续 FINE 类型， 填入参数 CONT、FINE；并可在连续轨迹形式后面的百分比(1%~100%)为平滑的程度，决定连续参数；非连续型式 FINE 分成三个等级:。

- FINE (=0)，会预读接续的指令，不会执行到位检查。
- FINE = 1 ，不会预读接续的指令，不会执行到位检查。
- FINE = 2 ，不会预读接续的指令，会执行到位检查。
- CONT (=100%) ，可依百分比(%)或转角半径长度(mm)决定平滑的程度。例如：CONT=10%、CONT=3mm

⇨ [速度] Vel 为单行速度参数，#填入参数 Vel=1~100%，数字越大表示速度越快

⇨ [加速度] Acc 为单行加速度参数，#填入参数 Acc=1~100%，数字越大表示加速度越快

⇨ [工具座标] TOOL[#] 为工具座标参数，可设定不同的末端工具，#可输入范围为 0 至 15

⇨ [基座座标]BASE[#] 为基底座标参数，可设定不同的基底编号，#可输入范围为 0 至 31

⇨ Vel、Acc、TOOL、BASE 可省略输入。省略时将参考上一行指令数值，或初始定义参数。

■ **范例 1：**

```
PTP_REL {A2 30} CONT=100% Vel=30% Acc=100% TOOL[0] BASE[0]
```

■ **范例 1 说明：**

机器手臂使用点到点运动方式 A2 轴相对增加 30 度移动。平滑轨迹设定值 100%，速度 30%，加速度 100%，第 0 组工具座标，第 0 个基座座标

■ **范例 2：**

PTP_REL {A3 -15, A4 5} FINE=2 Vel=100% Acc=50% TOOL[4] BASE[8]

■ **范例 2 说明：**

机器手臂使用点到点运动方式往 A3 轴相对减少 15 度移动 A4 轴相对增加 5 度移动。使用非连续平滑轨迹，速度 100%，加速度 50%，第 4 组工具座标，第 8 个基座座标

■ **参考相关指令：**

E6POINT、E6POS、E6AXISPTP、PTP、LIN、LIN_REL、CIRC、CIRC_REL。

2.2.3@LIN 直线移动指令

■ **指令说明：**

机器手臂由目前位置沿直线路径移动至目标点，以绝对位置方式移动，移动时末端姿态不变，呈直线运动(Linear Motion)。

例如：令机器手臂从 P0 启始点走直线到 P1 目标点，路径是直线方式。

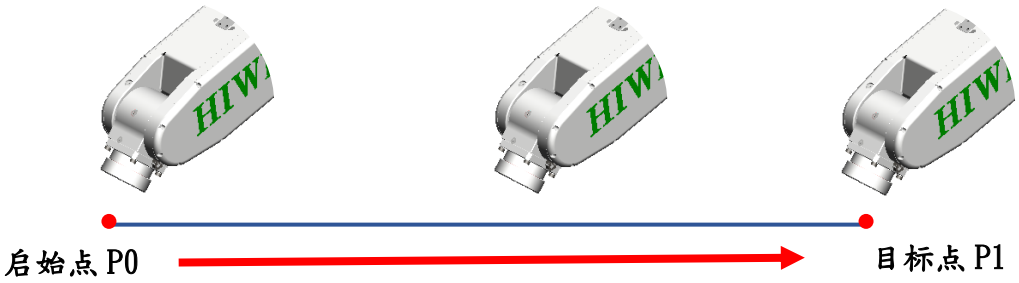


图 8 LIN 移动示意图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
LIN [点位名称] [轨迹形式        ] [速度 ] [加速度] [工具座标] [基座座标]
LIN [    P#       ] [FINE=#/CONT=#] [Vel=#] [Acc =#] [TOOL[#] ]    [BASE[#] ]
```

■ **格式/参数说明：**

⇨ #为数字，由使用者填入。

⇨ [点位名称]为目标点参数，可选择填入点位(P1~P9999)、自行定义点位(E6POINT、E6POS)、各轴角度值(A1~A6)、座标位置数值(XYZABC)，LU 系列无 A5、A6、A、B 轴。

⇨ [轨迹形式]为连续 CONT 或非连续 FINE 类型， 填入参数 CONT、FINE；并可在连续轨迹形式后面的百分比(1%~100%)为平滑的程度，决定连续参数；非连续型式 FINE 分成三个等级:。

1) FINE (=0)，会预读接续的指令，不会执行到位检查。
2) FINE = 1 ，不会预读接续的指令，不会执行到位检查。
3) FINE = 2 ，不会预读接续的指令，会执行到位检查。
4) CONT (=100%) ，可依百分比(%)或转角半径长度(mm)决定平滑的程度。例如：CONT=10%、CONT=3mm

⇨ [速度] Vel 为单行速度参数，#填入参数 Vel=1mm/s~2000mm/s，数字越大表示速度越快。(SCARA-LU 系列 Vel=1~4000 mm/s)

⇨ [加速度] Acc 为单行加速度参数，#填入参数 Acc=1~100%，数字越大表示加速度越快

⇨ [工具座标] TOOL[#] 为工具座标参数，可设定不同的末端工具，#可输入范围为 0 至 15

⇨ [基座座标]BASE[#] 为基底座标参数，可设定不同的基底编号，#可输入范围为 0 至 31

⇨ Vel、Acc、TOOL、BASE 可省略输入。省略时将参考上一行指令数值，或初始定义参数。

■ **范例 1：**

LIN P5 CONT=100% Vel=3000mm/s Acc=100% TOOL[0] BASE[0]

■ **范例 1 说明：**

机器手臂使用直线运动方式移动到目标点 P5。平滑轨迹设定值 100%，速度 3000 mm/s，加速度 100%，第 0 组工具座标，第 0 个基座座标。

■ **范例 2**

LIN {X -100} CONT=80% Vel=2500mm/s Acc=100% TOOL[0] BASE[0]

■ **范例 2 说明：**

机器手臂使用直线运动方式移动到 X -100 之位置。平滑轨迹设定值 80%，速度 2500 mm/s，加速度 100%，第 0 组工具座标，第 0 个基座座标。

■ **范例 3**

E6POINT AAA = {X 100, Y 90}

LIN AAA CONT=100% Vel=2000mm/s Acc=75% TOOL[1] BASE[1]

■ **范例 3 说明：**

机器手臂使用直线运动方式移动到自行定义点位位置 X 100mm、Y 90mm。平滑轨迹设定值 100%，速度 2000 mm/s，加速度 75%，第 1 组工具座标，第 1 个基座座标。

■ **参考相关指令：**

E6POINT、E6POS、E6AXISPTP、PTP_REL、LIN_REL、CIRC、CIRC_REL。

2.2.4@LIN_REL 相对位置方式直线移动指令

■ **指令说明：**

机器手臂沿直线路径移动至相对增量值之位置，移动时末端姿态不变。

例如：依前一点为位置相对直线移动，令机器手臂参考 X 轴 方向相对增加+100mm 距离。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

LIN_REL{ [各轴增量参数] } [轨迹形式] [速度] [加速度] [工具座标] [基座座标]

LIN_REL{ [X num, Y num,…]} [FINE=#/CONT=#] [Vel=#] [Acc =#] [TOOL[#]] [BASE[#]]

※各轴增量参数内容：{X num, Y num, Z num, A num, B num, C num}，num 表示数值。，使用者自行填入，SCARA-LU 与 DELTA 系列无 AB 轴。

■ **格式/参数说明：**

⇨ #为数字，由使用者填入。

⇨ [各轴增量参数] 为增量参数数值，可选择填入各轴座标位置数值(XYZABC)，SCARA-LU 与 DELTA 系列无 AB 轴。

⇨ [轨迹形式] 为连续 CONT 或非连续 FINE 类型， 填入参数 CONT、FINE；并可在连续轨迹形式后面的百分比(1%~100%)为平滑的程度，决定连续参数；非连续型式 FINE 分成三个等级:。

1) FINE (=0)，会预读接续的指令，不会执行到位检查。
2) FINE = 1 ，不会预读接续的指令，不会执行到位检查。
3) FINE = 2 ，不会预读接续的指令，会执行到位检查。
4) CONT (=100%) ，可依百分比(%)或转角半径长度(mm)决定平滑的程度。例如：CONT=10%、CONT=3mm

⇨ [速度] Vel 为单行速度参数，#填入参数 Vel=1mm/s~2000mm/s，数字越大表示速度越快。(SCARA-LU 系列 Vel=1~4000 mm/s)

⇨ [加速度] Acc 为单行加速度参数，#填入参数 Acc=1~100%，数字越大表示加速度越快

⇨ [工具座标] TOOL[#] 为工具座标参数，可设定不同的末端工具，#可输入范围为 0 至 15

⇨ [基座座标]BASE[#] 为基底座标参数，可设定不同的基底编号，#可输入范围为 0 至 31

⇨ Vel、Acc、TOOL、BASE 可省略输入。省略时将参考上一行指令数值，或初始定义参数。

■ **范例 1：**

LIN_REL {X -50} CONT=100% Vel=3000mm/s Acc=100% TOOL[0] BASE[0]

■ **范例 1 说明：**

机器手臂使用直线运动方式往 X-50 方向移动。平滑轨迹设定值 100%，速度 3000 mm/s，加速度 100%，第 0 组工具座标，第 0 个基座座标

■ **范例 2：**

LIN_REL {X 45, Z 15} FINE=2 Vel=1000mm/s Acc=50% TOOL[4] BASE[8]

■ **范例 2 说明：**

机器手臂使用直线运动方式往 X 轴+45 度、Z 轴+15 度之位置。使用非连续平滑轨迹，速度 1000 mm/s，加速度 50%，第 4 组工具座标，第 8 个基座座标

■ **参考相关指令：**

E6POINT、E6POS、E6AXISPTP、PTP_REL、LIN、CIRC、CIRC_REL。

2.2.5@LIN_REL_TOOL 参考工具座标直线移动指令

■ **指令说明：**

参考工具座标相对位置方式直线移动指令，为参考工具座标系作相对移动指令。

例如：参照工具座标系，往 X 轴方向做+100mm 直线移动。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 一次只能对一个轴作移动。

■ **指令格式：**

LIN_REL_TOOL { [各轴增量参数] } [轨迹形式] [速度] [加速度] [工具座标] [基座座标]

LIN_REL_TOOL { [X num, Y num,…]} [FINE=#/CONT=#] [Vel=#] [Acc =#] [TOOL[#]] [BASE[#]]

※各轴增量参数内容：{X num, Y num, Z num, A num, B num, C num}，num 表示数值。

使用者自行填入，SCARA-LU 与 DELTA 系列无 AB 轴。

■ **格式/参数说明：**

⇨ #为数字，由使用者填入。

⇨ [各轴增量参数]为增量参数数值，可选择填入各轴座标位置数值(XYZABC)，SCARA-LU 与 DELTA 系列无 AB 轴。

⇨ [轨迹形式]为连续 CONT 或非连续 FINE 类型， 填入参数 CONT、FINE；并可在连续轨迹形式后面的百分比(1%~100%)为平滑的程度，决定连续参数；非连续型式 FINE 分成三个等级:。

1) FINE (=0)，会预读接续的指令，不会执行到位检查。
2) FINE = 1 ，不会预读接续的指令，不会执行到位检查。
3) FINE = 2 ，不会预读接续的指令，会执行到位检查。
4) CONT (=100%) ，可依百分比(%)或转角半径长度(mm)决定平滑的程度。例如：CONT=10%、CONT=3mm

⇨ [速度]Vel 为单行速度参数，#填入参数 Vel=1mm/s~2000mm/s，数字越大表示速度越快。(SCARA-LU 系列 Vel=1~4000 mm/s)

⇨ [加速度]Acc 为单行加速度参数，#填入参数 Acc=1~100%，数字越大表示加速度越快

⇨ [工具座标] TOOL[#] 为工具座标参数，可设定不同的末端工具，#可输入范围为 0 至 15。

⇨ [基座座标] BASE[#] 为基底座标参数，可设定不同的基底编号，#可输入范围为 0 至 31。

⇨ Vel、Acc、TOOL、BASE 可省略输入。省略时将参考上一行指令数值，或初始定义参数，一次只能对一个轴作移动。

■ **范例 1：**

LIN_REL_TOOL {X -50} CONT=100% Vel=2000mm/s Acc=100% TOOL[0] BASE[0]

■ **范例 1 说明：**

机器手臂的工具坐标系使用直线运动方式往 X-50 方向移动。平滑轨迹设定值 100%，速度 2000 mm/s，加速度 100%，第 0 组工具座标，第 0 个基座座标

■ **范例 2：**

LIN_REL_TOOL {A3 45} FINE=2 Vel=1000mm/s Acc=50% TOOL[4] BASE[8]

■ **范例 2 说明：**

机器手臂的工具坐标系使用直线运动方式往 A3 轴+45 度之位置。使用非连续平滑轨迹，速度 1000 mm/s，加速度 50%，第 4 组工具座标，第 8 个基座座标

■ **参考相关指令：**

E6POINT、E6POS、E6AXISPTP、PTP、PTP_REL、LIN、LIN_REL、CIRC、CIRC_REL。。

2.2.6@CIRC 圆弧移动指令

■ **指令说明：**

使用圆弧方式移动指令，以当下位置与指令下的两点走圆弧路径，利用三点行成一圆弧路径，由当下手臂位置+过度点+终点三点组成一个圆弧。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

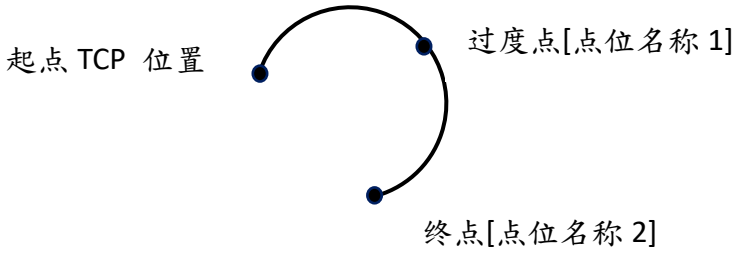


图 9 圆弧指令点位示意图

■ **指令格式：**

```
CIRC [点位名称 1] [点位名称 2] [轨迹形式        ] [速度 ] [加速度] [工具座标] [基座座标]
CIRC [P#          ] [P#          ] [FINE=#/CONT=#] [Vel=#] [Acc =#]  [TOOL[#] ]  [BASE[#] ]
```

■ **格式/参数说明：**

⇨ #为数字，由使用者填入。

⇨ [点位名称 1]、[点位名称 2]为点位名称，配合点位视窗作教点位 P1~P9999，取得欲到达下一点点位，或根据使用者配合指令 E6POINT 命名方式自行定义点位、或直接给予各轴角度、位置数值。

⇨ [轨迹形式]为连续 CONT 或非连续 FINE 类型， 填入参数 CONT、FINE；并可在连续轨迹形式后面的百分比(1%~100%)为平滑的程度，决定连续参数；非连续型式 FINE 分成三个等级:。

1) FINE (=0)，会预读接续的指令，不会执行到位检查。
2) FINE = 1 ，不会预读接续的指令，不会执行到位检查。
3) FINE = 2 ，不会预读接续的指令，会执行到位检查。
4) CONT (=100%) ，可依百分比(%)或转角半径长度(mm)决定平滑的程度。例如：CONT=10%、CONT=3mm

⇨ CONT、CONT = #%、CONT = #mm、FINE = #、省略轨迹形式等参数请择一使用。

⇨ [速度] 为单行速度参数，填入参数 Vel=1~2000mm/s，可省略此参数，省略速度将参考上一行指令，或初始定义参数。(SCARA-LU 系列 Vel=1~4000 mm/s)

⇨ [加速度] 为单行加速度参数，填入参数 Acc=1~100%，数字越大表示加速度越快， 可省略此参数，省略加速度将参考上一行指令，或初始定义参数。

⇨ [工具座标] 为工具座标参数，可设定不同的末端工具位置控制，表示 TOOL[#]，#可输入范围为 0 至 15，可省略此参数，省略工具座标将参考上一行指令，或初始定义参数。

⇨ [基座座标] 为 BASE 为基底座标参数，可设定为有进行输送带校正的基底编号，表示 BASE[#]，#可输入范围为 0 至 31，可省略此参数，省略基座座标将参考上一行指令，或初始定义参数。。

⇨ 至少需撰写 CIRC + [点位名称 1] [点位名称 2]，Vel、Acc、TOOL、BASE 参数可忽略输入。

■ **范例 1：**

CIRC P0 P1 CONT=100% Vel=2000mm/s Acc=100% TOOL[0] BASE[0]

Code

```
1  CIRC P0 P1 CONT=100% Vel=2000mm/s Acc=100% TOOL[0] BASE[0]
```

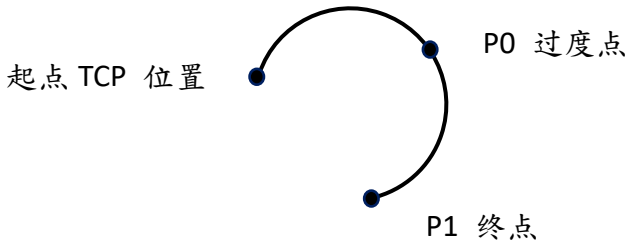


图 10 圆弧指令点位示意图

■ **范例 1 说明：**

当前位置 TCP 使用圆弧方式位经过 P0 到 P1 结束，应用连续轨迹 100%方式移动，应用速度 2000mm/s，应用加速度 100%，应用第 0 个工具座标，应用第 0 个基座座标。

■ **范例 2：**

E6POS POINTA = {X 20,Y 400,Z 800}
E6POS POINTB = {X 10,Y 200,Z 400}
CIRC POINTA POINTB CONT=100% Vel=2000mm/s Acc=100% TOOL[0] BASE[0]

Code

```
1  E6POS POINTA = {X 20,Y 400,Z 800}
2  E6POS POINTB = {X 10,Y 200,Z 400}
3  CIRC POINTA POINTB CONT=100% Vel=2000mm/s Acc=100% TOOOL[0] BASE[0]
```

■ **范例 2 说明：**

使用 E6POS 宣告 POINTA 点位为 X 轴 20mm、Y 轴 400mm、Z 轴 800mm 位置、POINTB 点位为 X 轴 10mm、Y 轴 200mm、Z 轴 400mm 位置，当前位置 TCP 使用圆弧方式位经过 POINTA 到 POINTB 结束，应用连续轨迹 100%方式移动，应用速度 2000mm/s，应用加速度 100%，应用第 0 个工具座标，应用第 0 个基座座标。

■ **参考相关指令：**

E6POINT、E6POS、E6AXIS、PTP、PTP_REL、LIN、LIN_REL、CIRC_REL。

2.2.7@CIRC_REL 相对位置方式圆弧移动指令

■ **指令说明：**

机器手臂沿弧线路径移动至相对增量值之位置，移动时末端姿态不变，与 CIRC 指令相似，但提供的过度点与终点采用相对位置。

例如：以原座标为起始点，移动至过度点再抵达至目地点

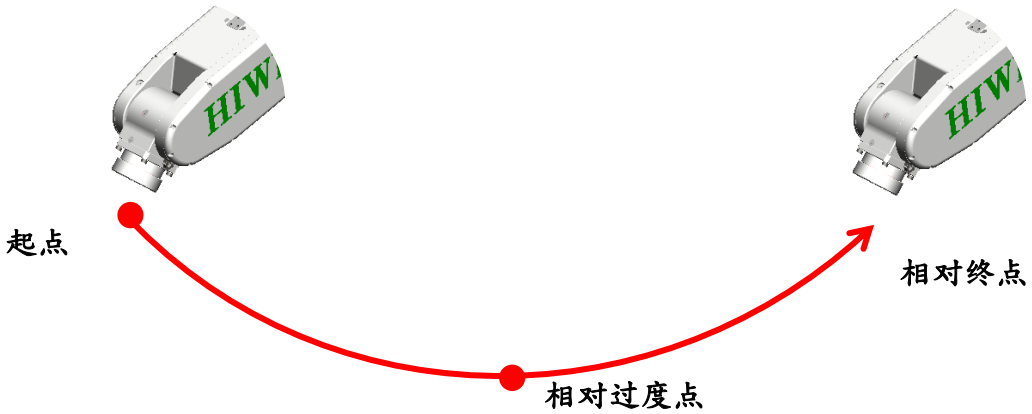


图 11 相对位置 CIRC_REL 示意图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

CIRC_REL { [过度点增量参数 1] } { [终点增量参数 2] } [轨迹形式] [速度] [加速度] [工具座标] [基座座标]

CIRC_REL { [X num, Y num,…] } { [X num, Y num,…] } [FINE=#/CONT=#] [Vel=#] [Acc =#] [TOOL[#]] [BASE[#]]

※各轴增量参数内容：{X num, Y num, Z num, A num, B num, C num}，num 表示数值。

使用者自行填入，SCARA-LU 与 DELTA 系列无 AB 轴。

■ **格式/参数说明：**

⇨ #为数字，由使用者填入。

⇨ [过度点增量参数 1]、[终点增量参数 2] 为过度点与终点的点位资讯资料，透过三点之相对数值行成一个弧线，[过度点增量参数 1]为起始点至中继点相对位置、[终点增量参数 2] 为中继点至终点相对位置，过度点与终点的点位各轴资讯可填入 X、Y、Z、A、B、C 等资讯，若无填写，则参考上一点或欲设数值。单位为 mm 或 Deg。

⇨ [轨迹形式]为连续 CONT 或非连续 FINE 类型， 填入参数 CONT、FINE；并可在连续轨迹形式后面的百分比(1%~100%)为平滑的程度，决定连续参数；非连续型式 FINE 分成三个等级:。

1) FINE (=0)，会预读接续的指令，不会执行到位检查。
2) FINE = 1 ，不会预读接续的指令，不会执行到位检查。
3) FINE = 2 ，不会预读接续的指令，会执行到位检查。
4) CONT (=100%) ，可依百分比(%)或转角半径长度(mm)决定平滑的程度。例如：CONT=10%、CONT=3mm

- ⇨ CONT、CONT = #%、CONT = #mm、FINE = #、省略轨迹形式等参数请择一使用。
- ⇨ [速度]为单行速度参数，填入参数 Vel=1~2000 mm/s，可省略此参数，省略速度将参考上一行指令，或初始定义参数。(SCARA-LU 系列 Vel=1~4000 mm/s)
- ⇨ [加速度]为单行加速度参数，填入参数 Acc=1~100%，数字越大表示加速度越快， 可省略此参数，省略加速度将参考上一行指令，或初始定义参数。
- ⇨ [工具座标]为工具座标参数，可设定不同的末端工具位置控制，表示 TOOL[#]，#可输入范围为 0 至 15，可省略此参数，省略工具座标将参考上一行指令，或初始定义参数。
- ⇨ [基座座标]为 BASE 为基底座标参数，可设定为有进行输送带校正的基底编号，表示 BASE[#]，#可输入范围为 0 至 31，可省略此参数，省略基座座标将参考上一行指令，或初始定义参数。。
- ⇨ 至少需撰写 CIRC_REL + { [过度点增量参数 1] }、{ [终点增量参数 2] }，Vel、Acc、TOOL、BASE 参数可忽略输入。

■ **范例 1：**

CIRC_REL {X -150, Y 150} {X -150, Y -150} CONT=100% Vel=2000mm/s Acc=50% TOOL[0] BASE[0]

■ **范例 1 说明：**

手臂末端以上一个座标位置为起始点，相对移动至辅助点{X -150, Y 150}再抵达至目地点{X -150, Y 150}，透过上述三个点的将画出一个圆弧，平滑轨迹设定值 100%，速度 2000mm/s，加速度 100%，第 0 组工具座标，第 0 个基座座标。

{X -150, Y 150} 为起始点至中继点相对位置、{X -150, Y 150} 为中继点至终点相对位置。

■ **范例 2：**

CIRC_REL {A1 5.0, A2 5.0, A3 5.0, A4 5.0} {A1 10.0, A2 10.0, A3 10.0, A4 10.0} CONT=100% Vel=2000mm/s Acc=50% TOOL[0] BASE[0]

■ **范例 2 说明：**

手臂末端以上一个座标位置为起始点，相对移动至辅助点{A1 5.0, A2 5.0, A3 5.0, A4 5.0}再抵达至目地点{A1 10.0, A2 10.0, A3 10.0, A4 10.0}，透过上述三个点的将画出一个圆弧，平滑轨迹设定值 100%，速度 2000mm/s，加速度 100%，第 0 组工具座标，第 0 个基座座标。

{A1 5.0, A2 5.0, A3 5.0, A4 5.0} 为起始点至中继点相对位置、{A1 10.0, A2 10.0, A3 10.0, A4 10.0} 为中继点至终点相对位置。

■ **参考相关指令：**

CIRC、PTP、PTP_REL、LIN、LIN_REL。

2.2.8@SPLINE 点与点拟合样条指令

■ **指令说明：**

将 1 个点以上的点位资讯，作点与点间的拟合成一样条线，当中使用 B-Spline 曲线运动方式作拟合，例如有多点的点胶轨迹预拟合成一条曲线轨迹即可应用此指令作结合。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
SPLINE
SPL [点位名称 1]
SPL [点位名称 2]
SPL [点位名称 3]
…
ENDSPLINE
```

■ **格式/参数说明：**

⇨ SPLINE 为拟合点位起头； ENDSPLINE 为拟合点位结尾。SPL + [点位名称]决定点位组合顺序，最多拟合 200 个点位。

⇨ [点位名称] 为各点位置的点位名称，名称内具备点位资讯(X,Y,Z,A,B,C；LU 系列无 AB 资讯)。

■ **范例： (以 RA605-710 手臂为范例)**

```
E6POINT P1 ={ X 0 , Y 368 , Z 293.5 }
E6POINT P2 ={ X 145 , Y 358 , Z 293.5 }
E6POINT P3 ={ X 185 , Y 420 , Z 255 }

PTP P0 CONT Vel=100% Acc=100% TOOL[0] BASE[0]
SPLINE
SPL P1
SPL P2
SPL P3
ENDSPLINE
```

■ **范例说明：**

使用点位变数宣告 P1、P2、P3 点位资讯。

点到点方式移动到 P0 位置后，开始执行 P1、P2、P3 点 SPLINE 拟合线作业并作拟合线运动。

■ **参考相关指令：**

E6POINT、E6POS、E6AXIS

2.2.9@STOP_MOTION 依讯触发停止上一个运动

■ **指令说明：**

此指令用于依讯号触发后停止前一行运动指令动作中，当指定的$DI、$SI 或$MI 触发开启(ON)时，将停止当前运动，并以 1mm/s 的速度依反方向后退慢速移动，直到至指定的$DI、$SI 或$MI 为触发关闭(OFF)，后续将停止前一行的动作，往下一个点位继续执行程式。可搭配于感测器用于测高度动作，类似停止并清除制动中指令(BRAKE)指令应用，例如：取堆迭板金工件时未知堆迭高度，针对第一个取板高度作侦测距离，即可使用此指令。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
[动作指令 1]
STOP_MOTION [输入讯号指令]
[动作指令 2]
```

■ **格式/参数说明：**

⇨ [动作指令] 为执程式陈述动作，逻辑程式、移动指令或讯号指令之组合。

⇨ [输入讯号指令] 为输入讯号控制指令格式，例如：$DI[]数位讯号输入、$SI[] 现场总线输入讯号 或$MI[]组合输入讯号，当输入讯号触发开启(ON)时，将停止当前运动，并以 1mm/s 的速度依反方向后退慢速移动，直到至指定的输入讯号为触发关闭(OFF)。

■ **范例：**

```
E6POINT OBJ_Position
LIN P0 CONT Vel=200mm/s Acc=100% TOOL[0] BASE[0]
STOP_MOTION $DI[1]
WAIT SEC 0.1
OBJ_Position = GETPOINT
```

■ **范例说明：**

说明：手臂末端挂载一个感测器，感测器讯号接到$DI[1]讯号，向 P0 方向测量物件高度，待感测器讯号触发后，就反向动作直至感测器讯号关闭后，记录当下位置。

宣告点位变数 OBJ_Position，直线运动慢速移动至 P0 点位，当移动途中末端感测器被触发开启时，立刻停止向前 P0 点方向动作，停止后反向 P0 点方向动作，直至末端感测器被关闭讯号，即等待 0.1 秒，并记录当下位置至 OBJ_Position 点位变数中。

■ **参考相关指令：**

E6POINT、GETPOINT。

2.2.10 @BRAKE 停止并清除制动中指令

■ **指令说明：**

停止并清除运动指令，包含在运动列中的指令并需搭配前一个移动指令使用连续轨迹(CONT)，通常搭配 IF 如果判断式来作停止条件触发。

可应用案例举例：若手臂应用在测试 Z 轴平面距离，末端位置从上往下慢速移动，若因手臂末端装设红外线感测器讯号触发条件成立后，表示手臂已经侦测到 Z 轴平面位置，手臂就执行跳出目前的侦测动作。后续以开发中断指令(INTERRUPT)。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
[动作指令]
IF [条件式] THEN
        BRAKE
 ENDIF
```

备注：BRAKE 不一定需要搭配 IF 回圈判断式，可以使用其他回圈判断式例如：WHILE、LOOP。

■ **格式/参数说明：**

⇨ [动作指令] 为执程式陈述动作，逻辑程式、移动指令或讯号指令之组合。

⇨ [条件式]为使用者设定条件，例如变数条件设定、讯号条件设定，可为以下逻辑运算子，等条件判断式。

运算子名称	语法	运算子名称	语法
小于	A < B	不等于	A != B
小于或等于	A <= B	等于	A == B
大于	A > B	逻辑取反	!A
大于或等于	A >= B		

⇨ IF … THEN 为IF判断式，判定条件是否成立。

⇨ 当[条件式]成立时，执行到BRAKE时，会停止运动，并搭配跳出。

⇨ 需搭配前一个移动指令使用连续轨迹(CONT)。

⇨ 通常外部会搭配回圈指令，例如：LOOP、WHILE…等，执行到BRAKE时，并搭配EXIT跳出外部回圈指令。

■ **范例：**

```
LOOP
    PTP P0 FINE=1 Vel=80% Acc=100% TOOL[0] BASE[0]
    PTP P1 CONT Vel=80% Acc=100% TOOL[0] BASE[0]
    IF  $DI[1]==1 THEN
        BRAKE
        EXIT
    ENDIF
    WAIT SEC 0.1
ENDLOOP
PTP P2 FINE=1 Vel=80% Acc=100% TOOL[0] BASE[0]
```

■ **范例说明：**

当 DI[1]的讯号没有被触发时，机器手臂在 P0 与 P1 两点间来回运动，

如果 DI[1]的讯号被触发，手臂从 P0 移动到 P1 后会停止运动，跳出回圈后走到 P2 点。

■ **参考相关指令：**

IF、LOOP、WHILE 、INTERRUPT … WHEN … DO…、INTERRUPT ON/OFF 。

2.2.11 @CHECK_LIN 直线路径奇异点检查指令

■ **指令说明：**

用于预先检查两点之间是否发生奇异点状态，避免发生中途移动至奇异点而停机的情况，让使用者可以透过软体先行确认，不需等到实机测试时，才发生点位问题，减少验证所花费时间，指令仅供预先检查作软体验证，尚需要实际测试后，才能确定不会发生奇异点。

机器手臂在运动中除了 PTP 点到点运动指令外(正向机器人运动学计算方式)，而 LIN 直线运动指令可能会移动至奇异点位置，此时机器手臂将发警报停止运行。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 此指令仅供 RA、RT 六轴系列使用，不适用 RS 史卡拉(SCARA)系列。

■ **指令格式：**

```
[自定义变数名称] =CHECK_LIN ([起始点位名称], [终止点位名称])
```

■ **格式/参数说明：**

⇨ [自定义变数名称]为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」；也可以使用$C[]计数暂存器。

⇨ [起始点位名称], [终止点位名称] 为开始检查直线路径奇异点点位与终止检查直线路径奇异点点位名称，例如 P1、P2…等，或者是使用者自行宣告的点位变数名称。

■ **范例 1：**

```
$C[2] =CHECK_LIN(P1, P2)
```

■ **范例 1 说明：**

在第 2 个计数暂存器可获得 P1 点与 P2 点直线路径奇异点检查后的数值，若数值为 1 则为有发生奇异点路径；若数值为 0 则为无发生奇异点路径。

■ **范例 2：**

```
IF CHECK_LIN(P1, P2) == TRUE THEN
    LIN P1
    LIN P2
ENDIF
```

■ **范例 2 说明：**

透过 IF 如果判断式，判断条件为若 P1 点与 P2 点直线路径奇异点检查值等于真(TRUE)时，确定没有发生奇异点路径，才执行直线运动 P1 点与 P2 点指令 。

■ **参考相关指令：**

IF、$C[]。

2.2.12 @P#_EXIST() 检查点位存在指令

■ **指令说明：**

确认点位是否存在指令。可用于判断点位资讯，若点位删除后，仍可继续执行剩下点位回圈。可以用在判定点位存在性后，在作执行应用的逻辑判断上使用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.10 以上	预读 CONT

■ **指令格式：**

格式(1)：

```
[布林变数] = P#_EXIST( [点位编号] )
```

格式(2)：

```
IF P#_EXIST( [点位编号] ) == TRUE THEN
    [动作指令]
ENDIF
```

■ **格式/参数说明：**

⇨ [布林变数] 为真 TRUE 或假 FALSE，布林逻辑变数，二者择一状态。

⇨ [点位编号] 为点位清单的点位数值，为阿拉伯数值。例如：P0 点，数值即为 0。

⇨ [动作指令] 为执行 副程式定义之陈述动作，逻辑程式、移动指令或讯号指令之组合。

⇨ 格式(1)为单行检查一个点位编号，格式(2)为P#_EXIST搭配上如果IF判断式使用。

⇨ 常配点位变数替换 P#()使用。

■ **范例 1：**

```
BOOL P_Status
P_Status = P#_EXIST(1)
```

■ **范例 1 说明：**

宣告布林变数 P_Status。

P_Status 变数为确认第 1 个点位是否存在的布林变数，若存在第 1 个点位，则为真(TURE)，反之，若第 1 个点位不存在，则为假(FALSE)。

■ **范例 2：**

```
INT G = 0
FOR G = 0 TO 3
    IF P#_EXIST(G) == TRUE THEN
        LIN P#(G)
    ENDIF
ENDFOR
```

■ **范例 2 说明：**

宣告整数变数 G。

FOR 回圈变数 G 从 0 到 3 共执行 4 次(0、1、2、3)，如果第 G 个点位存在为真(TRUE)则执行直线移动至第 G 个点；若点位不存在，就不执行该点。

■ **参考相关指令：**

IF、BOOL、FOR、P#()。

2.2.13 @P#() 点位变数替换指令

■ **指令说明：**

点位变数替换指令，适用于回圈内多点位动作重复带入使用，例如 1~20 个 Point 点位要作执行，若正常方式需从 P1、P2、P3…至 P20 点有 20 个程式行数，透过此指令后，将执行点位校正完成或通讯指派后，可透过 FOR 回圈从 1 顺续带入点位变数替换指令中，执行完 20 点位，可节省程式行数与重复撰写的时间，更精减程式并有效率达成期望的顺序点位。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.10 以上	预读 CONT

■ **指令格式：**

```
PTP P#( [点位编号] )
LIN P#( [点位编号] )
```

■ **格式/参数说明：**

⇨ [点位编号] 为点位清单(Pointlist)的点位编号，为阿拉伯数值，例如：P0 点，编号即为 0。可用自定义整数变数作编号更换。

⇨ 不支援使用者在程式中自行宣告的点位、PR 点位和其他变数。

⇨ 前方程式常配点检查点位存在指令 P#_EXIST()。

■ **范例 1：**

```
INT G = 0
FOR G = 0 TO 10
    IF P#_EXIST(G) == TRUE THEN
        PTP P#(G)
    ENDIF
ENDFOR
```

■ **范例 1 说明：**

宣告整数变数 G。

FOR 回圈变数 G 从 0 到 10 共执行 11 次，如果第 G 个点位存在为真(TRUE)则执行点到点方式移动至第 G 个点(从 P0、P1、P2…P10)；若点位不存在，就不执行该点。

■ **范例 2：**

```
INT G = 0
FOR G = 0 TO 10
    IF P#_EXIST(G) == TRUE THEN
        LIN P#(G)
    ENDIF
ENDFOR
```

■ **范例 2 说明：**

宣告整数变数 G。

FOR 回圈变数 G 从 0 到 10 共执行 11 次，如果第 G 个点位存在为真(TRUE)则执行直线方式移动至第 G 个点(从 P0、P1、P2…P10)；若点位不存在，就不执行该点。

■ **参考相关指令：**

IF、FOR、P#_EXIST()。

2.2.14 @CT_A6 无限旋转最末端轴速率指令

■ **指令说明：**

执行机器手臂最末端轴的无限旋转运动并决定旋转速率指令，关节式(六轴)手臂为第六轴旋转，史卡拉(SCARA)手臂为第四轴旋转。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.1 以上	Caterpillar1.0.6	预读 CONT

⇨ 关节式(六轴)手臂为第六轴旋转，史卡拉(SCARA)手臂为第四轴旋转。

⇨ 关节式(六轴)手臂需先先至主选单 >> 启动设定 Start-up >> 系统设定 Systems Setting >> 下一页(Next) >> 无线旋转 Continuous Turn 的分页上开启无限旋转功能设定。

⇨ 史卡拉(SCARA)手臂需先先至主选单 >>启动设定 Start-up >> 无线旋转 Continuous Turn。

⇨ 史卡拉(SCARA)机器手臂在开启无限旋转功能时，同时也会开启点位检查警报设置(Home point check)功能。

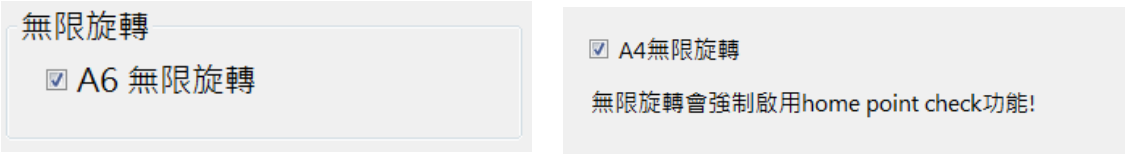


图 12 无限旋转设定图 (左图：关节式机器手臂，右图：史卡拉机器手臂)

■ **指令格式：**

```
CT_A6 [最末端轴旋转速率]
```

■ **格式/参数说明：**

⇨ CT_A6 为使用手臂最末端轴的无限旋转运动功能指令。

⇨ [最末端轴旋转速率] 为无限旋转速度比例(%)，可输入范围为-100 至 100，正负号决定顺时钟/逆时钟旋转方向，输入 0 则停止无限旋转运动。

■ **范例：**

```
CT_A6 88
WAIT SEC 3
PTP P0 CONT
PTP P1 CONT
CT_A6 0
```

■ **范例说明：**

开启关节式手臂第六轴的无限旋转运动并使用顺时钟 88%速率无限旋转，等待 3 秒钟。
点到点方式移动至 P0、P1 点，连续路径，无限旋转进行中。

到达 P1 点后，将无限旋转速度归零(停止第六轴旋转)。

■ **参考相关指令：**

无

2.3 代码 C、暂存器指令

此章节针对于代码 C 的暂存器指令做各别说明

2.3.1@$C[] 计数暂存器指令

■ **指令说明：**

计数暂存器为一变数空间，可储存整数，用于计数或计算。永久保存，关闭程式或断电不遗失。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
$C [ [计数暂存器编号] ] = <数值>
$C [ [计数暂存器编号] ] = $C [计数暂存器编号] + <数值>
```

■ **格式/参数说明：**

⇨ [计数暂存器编号] 为填入欲使用计数暂存器编号，可填入数值 1~20。

⇨ <数值> 为常数整数数值，可填入负值，储存容量为 32bit，
储存范围为-2147483648(-2^{31})~2147483647(2^{31} -1)。

⇨ 计数暂存器可搭配运算符号(加+、减-、乘*、除/)作数值计算。

■ **范例 1：**

```
$C[1] = 10
```

■ **范例 1 说明：**

常数 10 存入第 1 个计数暂存器。程式关闭后，变数定义之数值依旧暂存着。

編號	值	說明
1	10	COUNTER 1

■ **范例 2：**

```
$C[1] = 10
$C[1] = $C[1] + 1
```

■ **范例 2 说明：**

常数 10 存入第 1 个计数暂存器。

第 1 个计数暂存器值增加 1 并存入第 1 个暂存器，第 1 个计数暂存器值变为 11。程式关闭后，变数定义之数值依旧暂存着。

■ **参考相关指令：**

$PR[]、$T[]、$T_STOP[]

2.3.2@$T[] 计时暂存器指令

■ **指令说明：**

计时暂存器为一变数空间，英文代号 Timer，可储存时间变化整数值，用于计数时间或时间计算使用，计数时间跳动最小单位为 1 毫秒(ms,0.001 秒)。永久保存，断电不遗失。

使用者可用于计算手臂执行的运动时间，或使用者个用此计时暂存器当为条件式，应用在时间的判定。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
$T [ [计时暂存器编号] ] = <数值>
$T[ [计时暂存器编号] ] = $T [计时暂存器编号] + <数值>
```

■ **格式/参数说明：**

⇨ [计时暂存器编号] 为填入欲使用计时暂存器编号，可填入数值 1~20。

⇨ <数值> 为常数整数数值，可填入负值，储存容量为 32bit，
储存范围为-2147483648(-2^{31})~2147483647(2^{31} -1)。

⇨ 计时暂存器可搭配运算符号(加+、减-、乘*、除/)作数值计算。

⇨ 计时器计数时间跳动 1 毫秒(0.001 秒))为 1 个单位，为工程电脑软体计数时间。

■ **范例 1：**

```
$T[1] = 1001
```

■ **范例 1 说明：**

常数 1001 存入第 1 个计时暂存器，表示 1.001 秒。程式关闭后，变数定义之数值依旧暂存着。

■ **范例 2：**

```
REAL TIME_error
$T[1] = 1000
$T[2] = 3000
TIME_error = $T[2] -$T[1]
```

■ **范例 2 说明：**

宣告浮点数形式 Time_error 变数。

时间常数 1000 存入第 1 个计时暂存器，时间常数 2000 存入第 2 个暂存器。

Time_error 变数值为第 2 个计时暂存器值相减第 1 个计时暂存器，结果为 2000ms(2 秒)的误差时间。

■ **范例 3：**

```
REAL NEED_TIME
$T[1] = 0
PTP P0 FINE = 1
$T_STOP[1] = FALSE
PTP P1    FINE = 1
$T_STOP[1] = TRUE
NEED_TIME = $T[1]
```

■ **范例 3 说明：**

宣告浮点数形式 NEED_TIME 变数，并将第一个计时暂存器时间归零。
点到点方式移动至 P0 点，开启非连续轨迹，开启第一个计时暂存器时间计数。
点到点方式移动至 P1 点，开启非连续轨迹，关闭第一个计时暂存器时间计数。
将第一个计时暂存器时间转换至 NEED_TIME 变数，已利后续程式使用，同时也可以到计时暂存器介面观看执行 P1 点所需的时间资讯。

■ **参考相关指令：**

$PR[]、$C[]、$T_STOP[]

2.3.3@$T_STOP[] 停止计时暂存器指令

■ **指令说明：**

用于开启/停止计时暂存器的时间累加值，类似码秒的开关按钮动作指令，当停止计时暂存器为真(TRUE)则停止时间计数；相反的，停止计时暂存器为假(FALSE)则开启时间计数，时间以工业电脑软体时间为主作(毫)秒数增加，计算出来的时间资讯仅供参考，非与实际一定符合。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
$T_STOP [ [计时暂存器编号] ] = [布林变数]
```

■ **格式/参数说明：**

⇨ [计时暂存器编号] 为填入欲使用暂存器编号，可填入数值 1~20。

⇨ [布林变数] 为真 TRUE 或假 FALSE，布林逻辑变数，二者择一状态。当停止计时暂存器为真(TRUE)则停止时间计数；相反的，停止计时暂存器为假(FALSE)则开启时间计数。

⇨ 计时暂存器数值可到介面上观看计算数值，或搭配$T[]指令使用。

⇨ $T_STOP 预设为预读状态，故前面点位需选择 FINE=1 格式，或加入等待 0.05 秒，计算时间较为准确。

■ **范例 1：**

```
REAL NEED_TIME
$T[1] = 0
PTP P0 FINE = 1
$T_STOP[1] = FALSE
PTP P1    FINE = 1
$T_STOP[1] = TRUE
NEED_TIME = $T[1]
```

■ **范例 1 说明：**

宣告浮点数形式 NEED_TIME 变数，并将第一个计时暂存器时间归零。
点到点方式移动至 P0 点，开启非连续轨迹，开启第一个计时暂存器时间计数。
点到点方式移动至 P1 点，开启非连续轨迹，关闭第一个计时暂存器时间计数。
将第一个计时暂存器时间转换至 NEED_TIME 变数，已利后续程式使用，同时也可以到计时暂存器介面观看执行 P1 点所需的时间资讯。

■ **参考相关指令：**

$PR[]、$C[]、$T[]。

2.3.4@$PR[] 位置暂存器指令

■ **指令说明：**

位置暂存器 PR(Position Register)能将预设位置储存于暂存器里，也可以将手臂当下点位储存在暂存器里，可以有角度(DEG)格式与位置(POS)格式两种类型。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ HRSS3.3.16 之后版本才有 E1、E2、E3 外部轴；SCARA-LU 与 DELTA 系列无 AB 轴。

■ **指令格式：**

✧ 格式 1：

```
$PR [#] = { X <数值 1>,Y <数值 2>,Z <数值 3>,A <数值 4>,B <数值 5>,C <数值 6>
          ,E1 <数值 7> ,E2 <数值 8> ,E3 <数值 9>}
```

✧ 格式 2：

```
$PR [#] = { A1 <数值 1>, A2 <数值 2>,A3 <数值 3>,A4 <数值 4>,A5 <数值 5>,A6 <数值 6>
          ,E1 <数值 7> ,E2 <数值 8> ,E3 <数值 9> }
```

✧ 格式 3：

```
$PR [#] = P#
或
$PR [#] = [点位变数名称]
```

✧ 格式 4：

```
$PR [#] = GETPOINT
```

※SCARA-LU 与 DELTA 系列无 AB 轴。

■ **格式/参数说明：**

✧ 格式 1：

⇨ #为数字，使用者欲使用位置暂存器 PR 编号值。

⇨ 设定参数内容座标，以{ }内制定格式可直接宣告 X、Y、Z、A、B、C、E1、E2、E3 轴数值，SCARA-LU 与 DELTA 系列无 AB 轴，E1 ~ E3 为外部轴(HRSS3.3.16 之后版本才有外部轴)，赋与使用者自定义$PR[]内直角座标位置。

⇨ <数值 1>至<数值 9>为客户自行定义数值，距离单位为毫米 mm，角度单位为度，轴座标与数值需有一格空格，二个轴以上以”,”(逗号)做分隔。

⇨ 宣告完$PR 值，以利后续程式编辑使用，例如移动或作位置计算或通讯传输使用。

✧ 格式 2：

⇨ #为数字，使用者欲使用位置暂存器 PR 编号值。

⇨ 设定参数内容座标，以{ }内制定格式可直接宣告 A1~A6 轴与 E1~E3 轴数值，LU 系列无 A5、A6 轴，E1 ~ E3 为外部轴(HRSS3.3.16 之后版本才有外部轴)，赋与使用者自定义 $PR[]内关节座标角度位置。

⇨ <数值 1>~<数值 9>为客户自行定义数值，角度单位为度，外部轴单位为毫米 mm，轴座标与数值需有一格空格，二个轴以上以”,”(逗号)做分隔。

⇨ 宣告完$PR 值，以利后续程式编辑使用，例如移动或作位置计算或通讯传输使用。

✧ 格式 3：

⇨ #为数字，使用者欲使用位置暂存器 PR 编号值。

⇨ P#为在点位分页的使用者教点的点位名称，例如：P0、P1，$PR 暂存器将点位置资讯相同。

⇨ 使用者也可使用 **E6POS、E6AXIS** 自定义[点位变数名称]，再将点位资讯转移至$PR[]暂存器。

✧ 格式 4：

⇨ $PR 暂存器搭配 **GETPOINT** 指令，将现今的位置值，输入至暂存器。

■ **范例 1：**

```
$PR[2] = {X 0,Y 150,Z 300,A 0,B 0,C 0, E1 10,E2 5,E3 20}
LIN $PR[2]
```

■ **范例 1 说明：**

将手臂位置X=0mm，Y=150mm，Z=300mm，A=0°，B=0°，C=0°，E1 =10mm，E2 =5mm，E3 =20mm，数值储存在位置暂存器2里，再使用直线移动指令移动位置暂存器2数值。

■ **范例 2：**

```
$PR[1] = {A1 1,A2 2,A3 3,A4 4,A5 5,A6 6, E1 7,E2 8,E3 9}
PTP $PR[1]
```

■ **范例 2 说明：**

将手臂角度A1=1°，A2=2°，A3=3°，A4=4°，A5=5°，A6=6°，E1 =7mm ，E2 =8mm，E3 =9mm 数值储存在位置暂存器1里，再使用点到点指令移动位置暂存器1数值。



或

N..	Com...	TYPE	Value1	Value2	Value
1		DEG	1.000	2.000	3.00
2		POS	0.000	500.000	200.0
3					
4					

■ **范例 3：**

```
E6POS POINTA = {X 20,Y 400,Z 800}
$PR[4] = POINTA
```

■ **范例 3 说明：**

使用E6POS宣告POINTA点位为X轴20mm、Y轴400mm、Z轴800mm位置，将数值储存在位置暂存器4里。

■ **范例 4：**

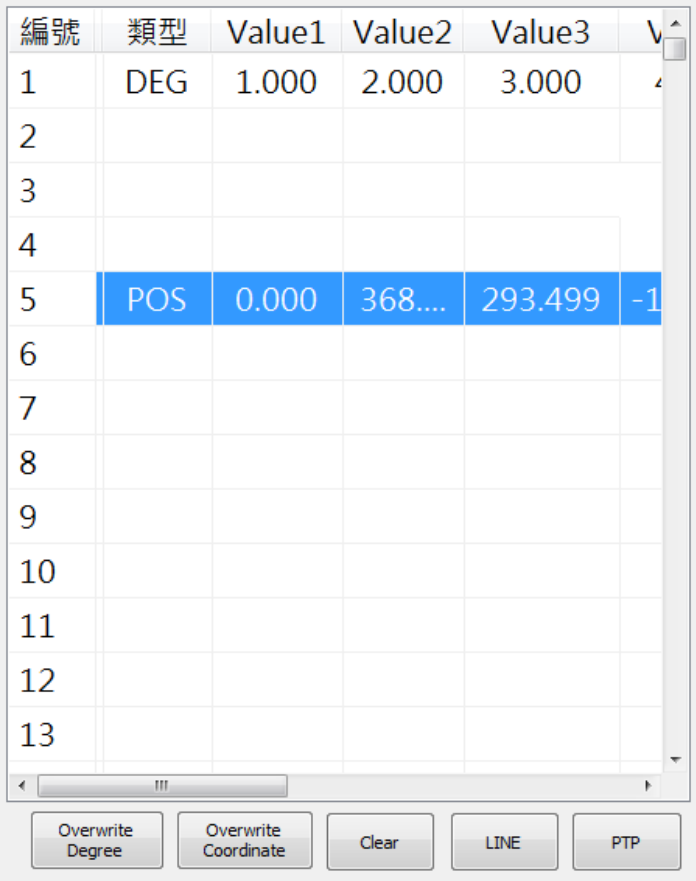
```
$PR[5] =GETPOINT
```

■ **范例 4 说明：**

将手臂当下位置资讯储存在暂存器5里。

模擬畫面 | 點位 | I/O | 位置 | 計時器 | 暫存器

參數名稱	數值	單位
X	0.000	毫米
Y	368.000	毫米
Z	293.500	毫米
A	-180.000	度
B	0.000	度
C	90.000	度



现在位置转移至$PR[5]位置资讯

■ **参考相关指令：**

GETPOINT、E6POINT、E6POS、E6AXIS。

2.3.5@$SRR[]、$SRW[] 现场总线(Fieldbus)暂存器

■ **指令说明：**

现场总线(Fieldbus)暂存器，SRR 为 Register for Read 的缩写，表示手臂暂存器数值等待读取外部设备(输入值)，SRW 为 Register for Write 的缩写，表示手臂暂存器数值写入外部设备(输出值)。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 需安装现场总线(Fieldbus)通讯硬体(例如：CC-Link 卡、Profinet 卡，选配项目)与相关设定才可使用，并连线成功后才可使用现场总线(Fieldbus)暂存器指令。

⇨ 先至主选单 >> 显示资讯 Display >> 现场总线 Fieldbus 的暂存器分页上作设定，可设定暂存器编号所代表的参数定义，参数例如轴 1 角度、X 值、错误码、TCP 速度、加速度时间…等资讯，每个参数所需要的现场总线暂存器数量不同。

■ **指令格式：**

```
[自定义变数名称] = $SRR[#]
$SRW[#] = <数值>
```

■ **格式/参数说明：**

⇨ $SRR 为表示手臂暂存器数值等待读取外部设备(输入值)，$SRW 表示手臂暂存器数值写入外部设备(输出值)。

⇨ [自定义变数名称]为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」；也可以使用$C[]计数暂存器。

⇨ #为数字，使用者欲使用现场总线暂存器编号值，可输入数值为 1~32。

⇨ <数值> 为常数整数数值，可填入负值，储存容量为 16bit，
储存范围为- 32768 (-2^{15}) ~ 32767(2^{15})。

■ **范例：**

```
$SRW[1] = 10
$C[2] =$SRR[1]
```

■ **范例说明：**

手臂第 1 个暂存器数值 10 写入给外部设备。
手臂第 1 个暂存器数值等待读取外部设备数值后，再将值传给第 2 个计数暂存器内。

■ **参考相关指令：**

$C[]

2.4 代码 D、输入&输出讯号控制指令

此章节针对于代码 D 的讯号通讯指令做各别说明。

I/O(输入/输出)指令，是读出输入讯号的状态，或改变外部设备输出讯号状态的指令。

编号	类型	I/O 名称	说明
1	DI/O	数位(Digital) I/O	通讯 PLC、通讯设备、感测器判定用
2	SI/O	现场总线(FieldBus) I/O	CC-Link/Profinet 网路通讯 I/O，需加装通讯卡才可使用。
3	FI/O	功能性(Function) I/O	外部控制，指定功能
4	RI/O	机器人(Robot) I/O	末端电气配置 I/O，机器人本体内线。(Delta、SCARA-LU 无)
5	MI/O	模组(Module) I/O	同时模组监控数个同类型 I/O。
6	VO	气压(Valve)I/O	控制末端内装电磁阀，用于控制末端气压控制。(RA6XX 系列才有)

后续小节对于各指令作说明。

2.4.1@$DI[ ] 数位输入讯号指令

■ **指令说明：**

开启或关闭已连接的数位输入讯号，使用者可利用此指令开启或关闭作为判断条件式。藉此讯号判断确定后，才执行后续程式，常与 WAIT FOR 指令搭配，或者使用在 IF 条件式判定。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	不预读 FINE

通常运用在判断式或等待讯号,使用键盘方式撰写指令，指令前面需有”$”作判定。

■ **指令格式：**

```
$DI[#] == [布林变数]
```

■ **格式/参数说明：**

⇨ 指令前面需有”$”作判定。

⇨ DI [#]数位讯号闸道，#可填入讯号数字 1~24(根据手臂机种有所不同)。

⇨ [布林变数] 为真 TRUE 或假 FALSE，布林逻辑变数，二者择一状态。或者可填入替代数字：1 或 0；开启或关闭讯号。

■ **范例 1：**

```
WAIT FOR $DI[1] == TRUE
WAIT FOR $DI[3] == 0
```

■ **范例 1 说明：**

当执行以下程式码时，等待数位讯号输入闸道($DI[]) 1 执行开启讯号,再等待数位讯号输入闸道($DI[]) 3 执行关闭讯号。

■ **范例 2：**

```
IF $DI[1] == TRUE THEN
   PTP P0
ENDIF
```

或者是

```
IF $DI[1] == 1 THEN
   PTP P0
ENDIF
```

■ **范例 2 说明：**

使用$DI[]讯号作判定程式，如果数位讯号闸道 1 开启时，执行点到点至 P0 点位程式，当数位讯号闸道 1 未开启时结束程式。

■ **参考相关指令：**

$DO[]、IF、WAIT FOR、BOOL。

2.4.2@$DO[ ] 数位输出讯号指令

■ **指令说明：**

数位输出讯号，使用者可利用此指令开启或关闭数位输出讯号，使用者可利用此指令开启或关闭周边通讯设备(例如：PLC、工具机、输送带、自动门…等)作交握讯号。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	不预读 FINE

■ **指令格式：**

```
$DO[#] == [布林变数]
```

■ **格式/参数说明：**

⇨ 指令前面需有”$”作判定。

⇨ DO [#]数位讯号闸道，#可填入讯号数字 1~24(根据手臂机种有所不同)。

⇨ [布林变数] 为真 TRUE 或假 FALSE，布林逻辑变数，二者择一状态。或者可填入替代数字：1 或 0；开启或关闭讯号。

■ **范例：**

```
$DO[1] = TRUE
```

或者是

```
$DO[1] = 1
```

■ **范例说明：**

开启数位讯号输出闸道($DO[])1 外部输出数位讯号。

■ **参考相关指令：**

$DI[]、BOOL

2.4.3@$SI[ ] 现场总线输入讯号指令

■ **指令说明：**

开启或关闭已连接的现场总线(Fieldbus)输入讯号，使用者可利用此指令开启或关闭作为判断条件式。藉此讯号判断确定后，才执行后续程式，常与 WAIT FOR 指令搭配，或者使用在 IF 条件式判定。

现场总线输入讯号 1~8 为系统预设的功能 I/O 指令，使用者无法做更改。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	不预读 FINE

■ **指令格式：**

```
$SI[#] == [布林变数]
```

■ **格式/参数说明：**

- ⇨ 指令前面需有”$”作判定。
- ⇨ SI [#]现场总线讯号闸道，#可填入讯号数字 1~128(根据手臂机种选配卡片有所不同)。
- ⇨ [布林变数] 为真 TRUE 或假 FALSE，布林逻辑变数，二者择一状态。或者可填入替代数字：1 或 0；开启或关闭讯号。
- ⇨ 现场总线输入讯号 1~8 为系统预设的功能 I/O 指令，使用者无法做更改，使用者应用从现场总线闸道第 9 道后开始自行定义使用。

■ **范例 1：**

```
WAIT FOR $SI[10] == TRUE
WAIT FOR $SI[13] == 0
```

■ **范例 1 说明：**

当执行以下程式码时，等待现场总线输入闸道($SI[]) 第 10 道执行开启讯号,再等待现场总线输入闸道($SI[]) 第 13 道执行关闭讯号。

■ **范例 2：**

```
IF $RI[10] == TRUE THEN
   PTP P0
ENDIF
或者是
IF $RI[10] == 1 THEN
   PTP P0
ENDIF
```

■ **范例 2 说明：**

使用$RI[]讯号作判定程式，如果现场总线讯号闸道第 10 道开启时，执行点到点至 P0 点位程式，当现场总线讯号闸道第 10 道未开启时结束程式。

■ **参考相关指令：**

$SO[]、IF、WAIR FOR、BOOL。

2.4.4@$SO[ ] 现场总线输出讯号指令

■ **指令说明：**

开启或关闭已连接的现场总线(Fieldbus)输出讯号，使用者可利用此指令开启或关闭现场总线输出讯号，使用者可利用此指令开启或关闭周边通讯设备(例如：PLC、工具机…等)作交握讯号，周边设备也同样为现场总线通讯方式才可。

现场总线输出讯号 1~8 为系统预设的功能 I/O 指令，使用者无法做更改。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	不预读 FINE

■ **指令格式：**

```
$DO[#] == [布林变数]
```

■ **格式/参数说明：**

⇨ 指令前面需有”$”作判定。

⇨ SO [#]现场总线讯号闸道，#可填入讯号数字 1~128(根据手臂机种选配卡片有所不同)。

⇨ [布林变数] 为真 TRUE 或假 FALSE，布林逻辑变数，二者择一状态。或者可填入替代数字：1 或 0；开启或关闭讯号。

⇨ 现场总线输入讯号 1~8 为系统预设的功能 I/O 指令，使用者无法做更改，使用者应用从现场总线闸道第 9 道后开始自行定义使用。

■ **范例：**

```
$SO[11] = TRUE
```

或者是

```
$SO[11] = 1
```

■ **范例说明：**

开启现场总线讯号输出闸道($SO[])第 11 道外部输出现场总线讯号。

■ **参考相关指令：**

$SO[]、IF、WAIR FOR、BOOL。

2.4.5@$VO[ ] 电磁阀输出讯号指令

■ **指令说明：**

电磁阀输出讯号，使用者可利用此指令开启或关闭电磁阀讯号，使用者可利用此指令开启或关闭电磁阀内部设备输出讯号。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	不预读 FINE

■ **指令格式：**

```
$VO[#] == [布林变数]
```

■ **格式/参数说明：**

⇨ 指令前面需有”$”作判定。

⇨ VO [#]机器人末端讯号闸道，#可填入讯号数字 1~3。

⇨ [布林变数] 为真 TRUE 或假 FALSE，布林逻辑变数，二者择一状态。或者可填入替代数字：1 或 0；开启或关闭讯号。

■ **范例：**

```
$VO[1] = TRUE
```

或者是

```
$VO[1] = 1
```

■ **范例说明：**

开启电磁阀讯号输出闸道($RO[])1 外部输出电磁阀讯号。

■ **参考相关指令：**

BOOL。

2.4.6@$RO[ ] 机器人输出讯号指令

■ **指令说明：**

机器人末端输出讯号，使用者可利用此指令开启或关闭机器人末端讯号，使用者可利用此指令开启或关闭手臂末端外部设备输出讯号(例如：气动夹爪或气管)。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	不预读 FINE

■ **指令格式：**

```
$RO[#] == [布林变数]
```

■ **格式/参数说明：**

⇨ 指令前面需有”$”作判定。

⇨ RO [#]机器人末端讯号闸道，#可填入讯号数字 1~4。

⇨ [布林变数] 为真 TRUE 或假 FALSE，布林逻辑变数，二者择一状态。或者可填入替代数字：1 或 0；开启或关闭讯号。

■ **范例：**

```
$RO[1] = TRUE
```

或者是

```
$RO[1] = 1
```

■ **范例说明：**

开启机器人末端讯号输出闸道($RO[])1 外部输出机器人末端讯号。

■ **参考相关指令：**

$RI[]、BOOL

2.4.7@$RI[ ] 机器人输入讯号指令

■ **指令说明：**

开启或关闭已连接的机器人末端输入讯号，使用者可利用此指令开启或关闭作为判断条件式。藉此讯号判断确定后，才执行后续程式，常与 WAIT FOR 指令搭配，或者使用在 IF 条件式判定。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	不预读 FINE

■ **指令格式：**

```
$RI[#] == [布林变数]
```

■ **格式/参数说明：**

⇨ 指令前面需有”$”作判定。

⇨ RI [#]机器人末端讯号闸道，#可填入讯号数字 1~6。

⇨ [布林变数] 为真 TRUE 或假 FALSE，布林逻辑变数，二者择一状态。或者可填入替代数字：1 或 0；开启或关闭讯号。

■ **范例 1：**

```
WAIT FOR $RI[1] == TRUE
WAIT FOR $RI[3] == 0
```

■ **范例 1 说明：**

当执行以下程式码时，等待机器人末端输入闸道($RI[]) 1 执行开启讯号,再等待机器人末端输入闸道($RI[]) 3 执行关闭讯号。

■ **范例 2：**

```
IF $RI[1] == TRUE THEN
   PTP P0
ENDIF
或者是
IF $RI[1] == 1 THEN
   PTP P0
ENDIF
```

■ **范例 2 说明：**

使用$RI[]讯号作判定程式，如果机器人末端讯号闸道 1 开启时，执行点到点至 P0 点位程式，当机器人末端讯号闸道 1 未开启时结束程式。

■ **参考相关指令：**

$RO[]、IF、WAIR FOR、BOOL。

2.4.8@SYN 输送带追踪的进行输出讯号开关调整时间指令

■ **指令说明：**

输送带追踪的进行输出($DO/$SO)讯号开关调整时间指令，手臂运动过程中设定输出讯号的开关调整时间指令，SYN 设定可一次多个状态，最多 8 笔。应用在手臂调整给外部配件/设备的讯号延迟时间指令，例如调整开启/关闭末端吸盘吸取动作时间，调整胶枪开启/关闭供应胶的动作时间等应用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

- 使用 2 点以上的程式点位，并将 SYN 放在 2 点之间。

■ **指令格式：**

```
SYN [输出讯号]        = [布林变数] [计算开头/结尾/路径设定] DELAY = [调整时间] ms
SYN [$DO[#]/$SO[#]] = [布林变数] [    START/ END /PATH       ] DELAY = [调整时间] ms
```

■ **格式/参数说明：**

⇨ SYN 非单独指令，需搭配点位移动动作指令使用，单一移动指令中，可不只一个 SYN 设定，最多 8 笔 SYN 设定。

⇨ [输出讯号] 为输送带飞抓手臂输出给外部设备的讯号指令，例如开启/关闭末端吸盘动作，可用输出讯号为数位输出($DO[#])与现场总线输出讯号($SO[#])，#表示数字，代表选用第#个输出讯号。

⇨ [布林变数] 为真 TRUE 表示启动或假 FALSE 表示关闭，布林逻辑变数，二者择一状态。

⇨ [计算开头/结尾设定] 为输出讯号以开头或结尾开始计算延迟时间的设定，选择开头输入「START」文字，或选择结尾输入「END」文字，或选择中间路径输入「PATH」文字，三者择一状态，若为「PATH」需要再加上距离单位毫米(mm)，距离从上一点开始计算。

⇨ [调整时间] 为输出讯号提前或延迟的调整时间数值，若提前时间以负值表示，若延迟时间以正值表示，可以输入范围为-1000~1000，单位毫秒 ms。

■ **范例 1：**

```
LIN P1 FINE Vel=100% Acc=50% TOOL[0] BASE[0]
SYN $DO[1] = TRUE START DELAY = 50 ms
SYN $DO[2] = TRUE END DELAY = -50 ms
LIN P2 FINE Vel=100% Acc=50% TOOL[0] BASE[0]
```

■ **范例 1 说明： (P1 与 P2 都使用 FINE 轨迹模式)**

如图下所示，从 P1 移动到 P2 加入 SYN 指令，P1、P2 在没有平滑的情况下，START 的范围为精确的点位 P1 位置到精确的点位 P2 位置，End 的范围为 P2 点到 P1 点的距离；SYN 下 START 指令 Delay 50ms，即从 P1 开始经过延迟 50ms 后，执行 DO[1]=True 的指令，SYN 下 END 指令 Delay -50ms，即从 P2 往回推提前 50ms 执行 DO[2]=True，藉此指令来调整输出的讯号时间。

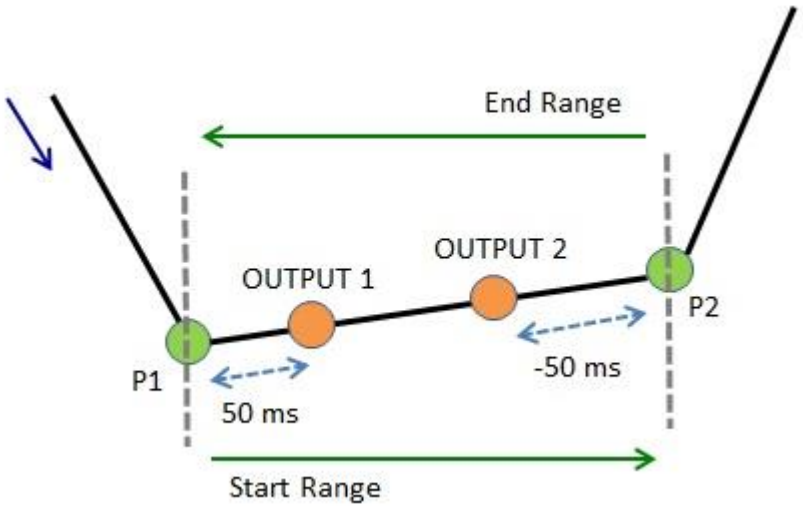


图 13 范例 1 的示意图

➢ 范例 2： (P1 使用 FINE 轨迹模式，P2 使用 CONT 轨迹模式，START& END 范例)

```
LIN P1 FINE Vel=100% Acc=50% TOOL[0] BASE[0]
SYN $DO[1] = TRUE START DELAY = 50 ms
SYN $DO[2] = TRUE END DELAY = -50 ms
LIN P2 CONT=100% Vel=100% Acc=50% TOOL[0] BASE[0]
```

■ **范例 2 说明：**

如图下所示，从 P1 移动到 P2 加入 SYN 指令，P2 有使用平滑的情况下，START 的范围为精确的点位 P1 位置到 P2 点的平滑起始位置，END 的范围为 P2 点的平滑起始位置到 P2 点的平滑终止位置；SYN 下 START 指令 Delay 50ms，即从 P1 开始经过延迟 50ms 后，执行 DO[1]=True 的指令，SYN 下 END 指令 Delay -50ms，即从 P2 平滑范围的中心点开始提前 50ms，执行 DO[2]=True 的指令。

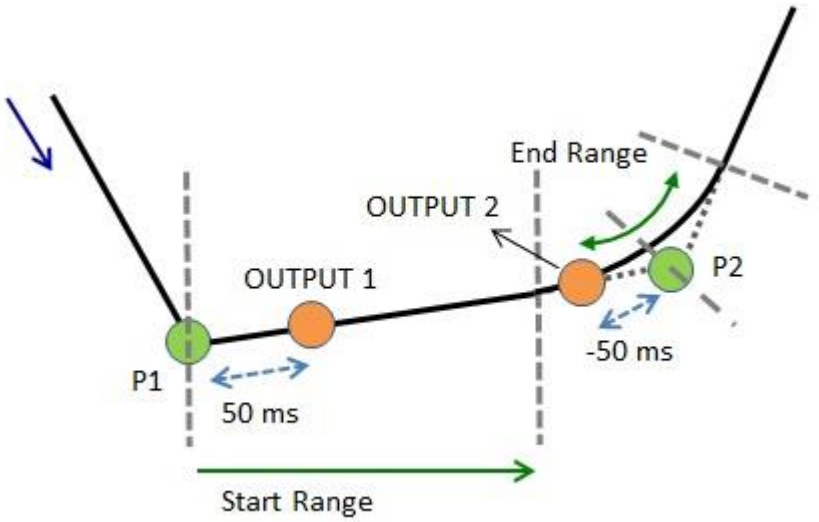


图 14 范例 2 的示意图

■ **范例 3： (P1 与 P2 都使用 CONT 轨迹模式，START& END 范例)**

```
LIN P1 CONT=100% Vel=100% Acc=50% TOOL[0] BASE[0]
SYN $DO[1] = TRUE START DELAY = 50 ms
SYN $DO[2] = TRUE END DELAY = -50 ms
LIN P2 CONT=100% Vel=100% Acc=50% TOOL[0] BASE[0]
```

■ **范例 3 说明：**

如图下所示，从 P1 移动到 P2 加入 SYN 指令，P1、P2 皆有平滑的情况下，START 的范围为 P1 点的平滑终止位置到 P2 点的平滑开始位置，END 的范围为 P2 点的平滑起始位置到 P2 点的平滑终止位置；SYN 下 START 指令 Delay 50ms，即从 P1 平滑终止位置开始经过延迟 50ms 后，执行 DO[1]=True 的指令，SYN 下 END 指令 Delay -50ms，即从 P2 平滑范围的中心点开始提前 50ms，执行 DO[2]=True 的指令。

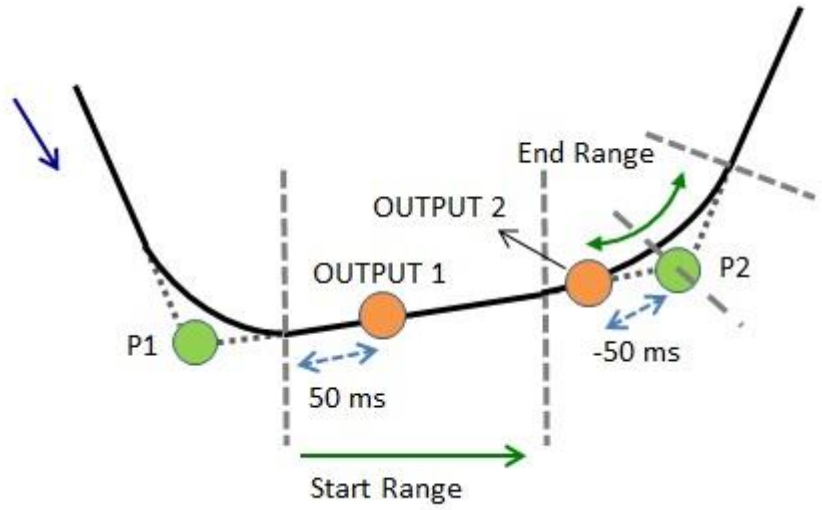


图 15 范例 3 的示意图

■ **范例 4： (P1 与 P4 都使用 FINE 轨迹模式，PATH 范例)**

```
LIN P1 FINE Vel=100% Acc=50% TOOL[0] BASE[0]
SYN $DO[1] = FALSE PATH = 50 mm DELAY = -50 ms
LIN P2 CONT=100% Vel=100% Acc=50% TOOL[0] BASE[0]
LIN P3 CONT=100% Vel=100% Acc=50% TOOL[0] BASE[0]
LIN P4 FINE Vel=100% Acc=50% TOOL[0] BASE[0]
```

■ **范例 4 说明：**

如下图所示，从 P1 移动到 P2 加入 SYN 指令，有使用 PATH 且 P2、P3 皆有平滑的情况下，START 的范围为精确的点位 P1 位置到精确的点位 P4 位置；SYN 下 PATH=50mm DELAY = -50ms，即从 P1 点开始的 50mm 位置，再提前 50ms 处，执行 DO[1]=FALSE；如果 P3 点为精确的点位未使用平滑的情况，则 START 的范围为精确的点位 P1 位置到精确的点位 P3 位置。

范例 4 的示意图

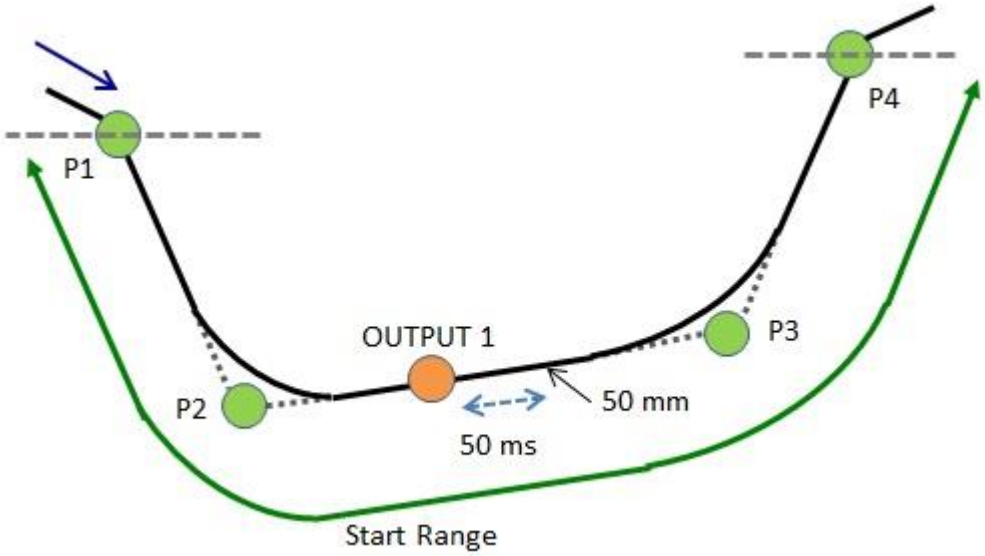


图 16、图范例 4 的示意图

■ **范例 5： (P1 使用 CONT 轨迹模式，P4 使用 FINE 轨迹模式，PATH 范例)**

```
LIN P1 CONT=100% Vel=100% Acc=50% TOOL[0] BASE[0]
SYN $DO[1] = FALSE PATH = 50 mm DELAY = -50 ms
LIN P2 CONT=100% Vel=100% Acc=50% TOOL[0] BASE[0]
LIN P3 CONT=100% Vel=100% Acc=50% TOOL[0] BASE[0]
LIN P4 FINE Vel=100% Acc=50% TOOL[0] BASE[0]
```

■ **范例 5 说明：**

如图下所示，从 P1 移动到 P2 下 SYN 指令，有使用 PATH 且 P1、P2、P3 皆有平滑的情况下，START 的范围为 P1 点的平滑开始位置到精确的点位 P4 位置；SYN 下 PATH=50mm DELAY = -50ms，即从 P1 点的平滑开始位置起算 50mm，再提前 50ms 处，执行 DO[1]=FALSE；如果 P2 点为精确的点位未使用平滑的情况，则 START 的范围为 P2 点的平滑开始位置到精确的点位 P3 位置。

范例 5 的示意图

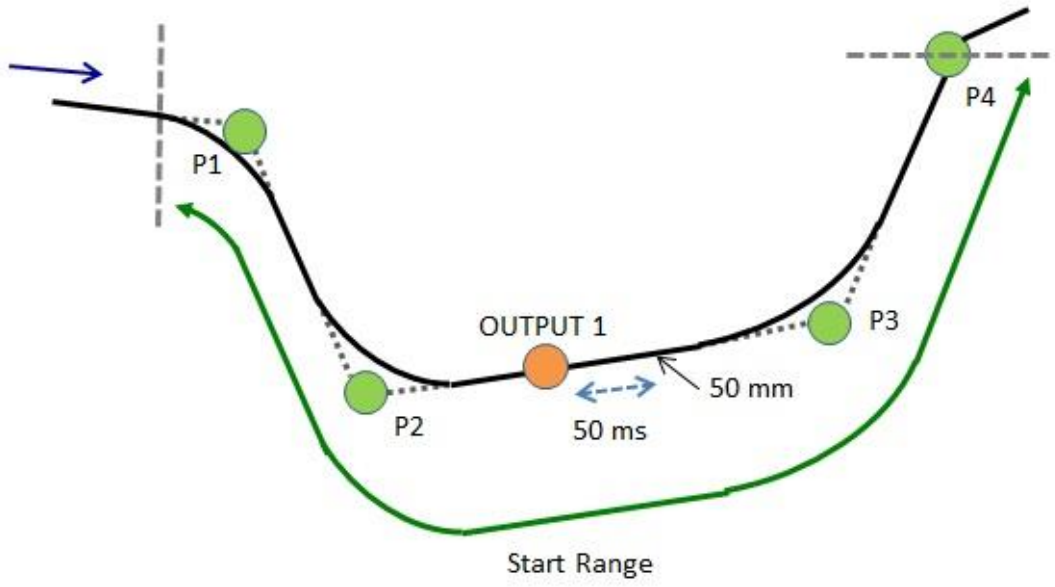


图 17 范例 5 的示意图

■ **参考相关指令：**

PTP、LIN、PICK、PLACE。

2.4.9@PULSE 输出讯号脉冲时间控制指令

■ **指令说明：**

输出讯号脉冲时间控制指令，可使$DO[]讯号开启后，可控制延时关闭讯号时间，可应用在掌握外部交握的时间控制与开关时间控制使用，例如控制指示灯号，可以控制灯号开启时间为 5 秒后即关闭灯号。

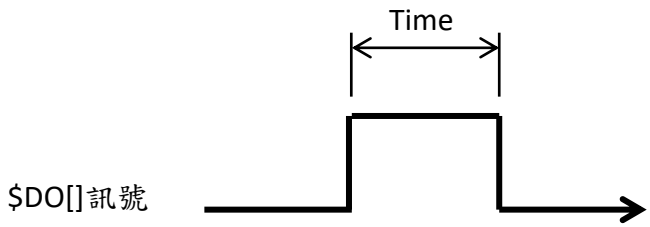


图 18 脉冲 PULSE 数位输出示意图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.10 以上	不预读 FINE

■ **指令格式：**

[数位输出讯号] = PULSE ([脉冲输出时间])

■ **格式/参数说明：**

⇨ [数位输出讯号] 为数位输出讯号指令，如$DO[1]为第一道数位输出讯号。只支援$DO 输出脉冲讯号，$SO[]现场总现输出讯号、$MO[]组合输出讯号不适用。

⇨ [脉冲输出时间]：脉冲输出时间，预设 0.1 秒

1) 最小 0.1 秒单位值，最大数值 30 秒。
2) 最多同时支援 3 组 PULSE 指令。
3) 程式取消(QUIT 指令) 会取消 PULSE 输出。
4) 中断触发指令不会取消 PULSE 输出。
5) 警急停止、使用者停止、Alarm 不会取消 PULSE 输出。

⇨ 此指令会挡预读，类同 FINE=1 设定功能，让程式不会提前读取下面程式。

■ **范例：**

```
PTP P1
$DO[1] = PULSE(3.2)
WAIT SEC 5
LIN P2
```

■ **范例说明：**

点到点移动至 P1 点位后，开启第一道数位输出讯号，持续输出 3.2 秒脉冲后，关闭第一道数位输出讯号。输出讯号的同时也在执行等待 5 秒指令程式，最后直线移动至 P2 点位。

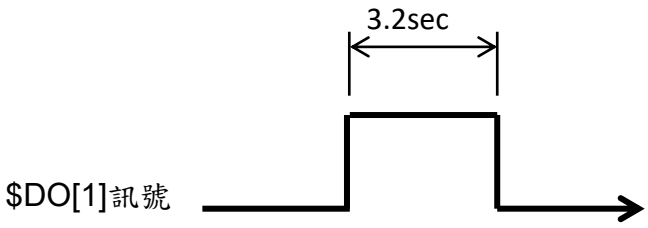


图 19 脉冲 PULSE 3.2 秒数位输出$DO[1]示意图

■ **参考相关指令：**

$DO[]、WAIT SEC。

2.4.10 @$MI[ ] 组合输入讯号指令

■ **指令说明：**

开启或关闭的组合输入讯号，为多个连续的数位输入(DI)或现场总线输入(SI)讯号组合而成，需透过介面先行设定起头讯号与终止讯号，讯号之间需为连续编号。

使用者可利用此指令开启或关闭作为判断条件式。藉此多个讯号判断确定后，才执行后续程式，常与 WAIT FOR 指令搭配，或者使用在 IF 条件式判定。

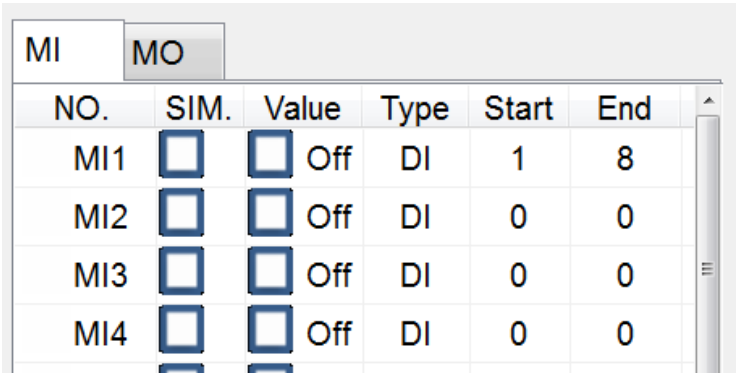
■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	不预读 FINE

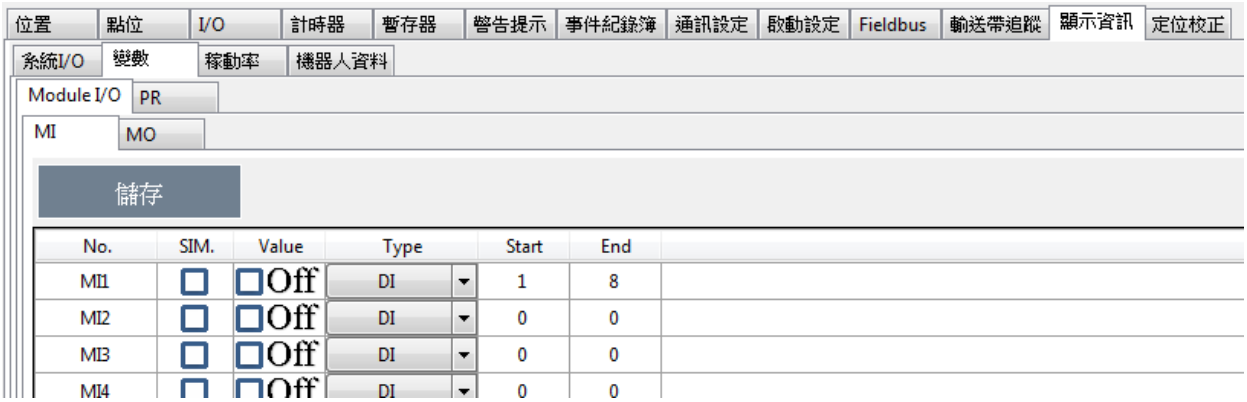
⇨ 通常运用在判断式或等待讯号,使用键盘方式撰写指令，指令前面需有”$”作判定。

⇨ 需在软体的介面做讯号来源类型、起头与结尾讯号设定后才可使用

HRSS 路径：主选单 >> 显示资讯 Display >> 变数 Variable >> Module I/O>> MI。



Caterpillar 路径：下方式窗 >> 显示资讯 Display >> Variable >> Module I/O >> MI。



■ **指令格式：**

```
$MI[#] == [布林变数]
```

■ **格式/参数说明：**

⇨ 指令前面需有”$”作判定。

⇨ MI [#]数位讯号闸道，#可填入讯号数字 1~32。

⇨ [布林变数] 为真 TRUE 或假 FALSE，布林逻辑变数，二者择一状态。或者可填入替代数字：1 或 0；开启或关闭讯号。

■ **范例 1：**

```
WAIT FOR $MI[1] == TRUE
WAIT FOR $MI[3] == 0
```

■ **范例 1 说明：**

当执行以下程式码时，等待组合讯号输入闸道($MI[]) 1 执行开启讯号,再等待组合讯号输入闸道($MI[]) 3 执行关闭讯号。

■ **范例 2：**

```
IF $MI[1] == TRUE THEN
   PTP P0
ENDIF
```

或者是

```
IF $MI[1] == 1 THEN
   PTP P0
ENDIF
```

■ **范例 2 说明：**

使用$MI[]讯号作判定程式，如果组合讯号闸道 1 开启时，执行点到点至 P0 点位程式，当组合讯号闸道 1 未开启时结束程式。

■ **参考相关指令：**

$MO[]、$DI[]、$DO[]、$SI[]、$SO[]、IF、WAIT FOR、BOOL。

2.4.11 @$MO[ ] 组合输出讯号指令

■ **指令说明：**

开启或关闭组合输出讯号，为多个连续的数位输出(DO)或现场总线输出(SO)讯号组合而成，使用者可利用此指令开启或关闭组合输出讯号，对象有可能为多个周边通讯设备或吸盘讯号作交握讯号，可节省程式指令行数。

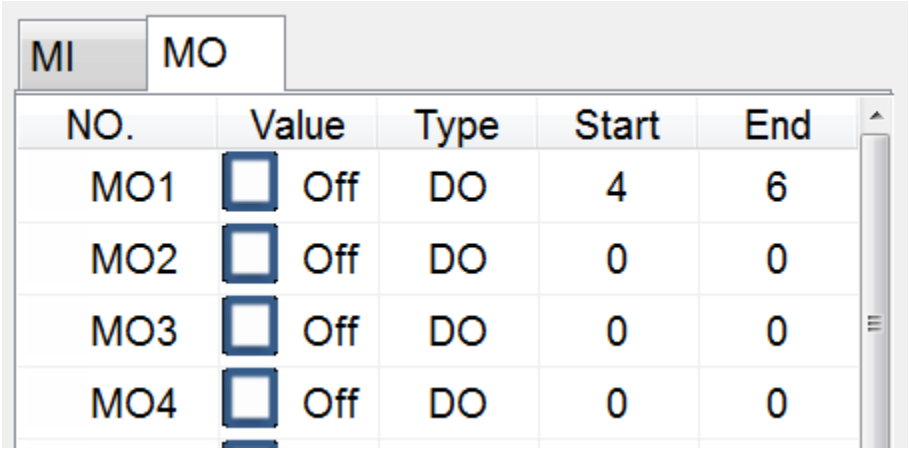
■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	不预读 FINE

⇨ 通常运用在判断式或等待讯号,使用键盘方式撰写指令，指令前面需有”$”作判定。

⇨ 需在软体的介面做讯号来源类型、起头与结尾讯号设定后才可使用

HRSS 路径：主选单 >> 显示资讯 Display >> 变数 Variable >> Module I/O >> MO。



Caterpillar 路径：下方式窗 >> 显示资讯 Display >> Variable >> Module I/O >> MO。



■ **指令格式：**

$MO[#] == [布林变数]

■ **格式/参数说明：**

⇨ 指令前面需有”$”作判定。

⇨ MO [#]组合讯号闸道，#可填入讯号数字 1~32。

⇨ [布林变数] 为真 TRUE 或假 FALSE，布林逻辑变数，二者择一状态。或者可填入替代数字：1 或 0；开启或关闭讯号。

■ **范例：**

$MO[1] = TRUE

或者是

$MO[1] = 1

■ **范例说明：**

开启组合讯号输出闸道($MO[])1。

■ **参考相关指令：**

$MI[]、$DI[]、$DO[]、$SI[]、$SO[]、BOOL

2.5 代码 E、流程控制功能指令

此章节针对于代码 E 的逻辑功能指令做各别说明

2.5.1@IF 如果判断式

■ **指令说明：**

此指令是判断型指令，使用在判断单个或复数条件，当达成条件，执行成立条件的后续动作，若所有条件都不成立，则执行判断式以后的程式码动作。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

✧ 格式 1：

```
IF [条件] THEN
   [动作指令]
ENDIF
```

✧ 格式 2：

```
IF [条件] THEN
   [动作指令1]
ELSE
   [动作指令2]
ENDIF
```

✧ 格式 3：

```
IF [条件A] AND [条件B] THEN
   [动作指令]
ENDIF
```

■ **格式/参数说明：**

⇨ [条件]、[条件 A]、[条件 B]为使用者设定条件，例如变数条件设定、讯号条件设定，可为以下逻辑运算子，等条件判断式。提醒：目前无法将两个比较运算子直接连接(例如. 1< 变数 <10)，否则软体会回报比较操作错误讯息("comparison operator error!")。

运算子名称	语法	运算子名称	语法
小于	A < B	不等于	A != B
小于或等于	A <= B	等于	A == B
大于	A > B	逻辑取反	!A
大于或等于	A >= B		

- ⇨ [动作指令] 为执行 IF判断式之陈述动作，逻辑程式、移动指令或讯号指令之组合。
- ⇨ 格式 1 为单一条件判断式，并执行单一动作指令。
- ⇨ 格式 2 为单一条件判断式，透过判定条件执行动作指令 1 或执行动作指令 2，两者只能二择一。
- ⇨ 格式 2 为两个以上条件判断式，并执行单一动作指令。

■ **范例 1：**

```
IF   $DI[1] == TRUE   THEN
     PTP P0 CONT Vel=100% Acc=100% TOOL[0] BASE[1]
ENDIF
```

■ **范例 1 说明：**

判断如果数位输入$D[1]为真，然后执行以点到点方式移动至点 P0，如果数位输入$D[1]为假，则忽略此判断式，继续执行 ENDIF 后指令码。

■ **范例 2：**

```
IF   $DI[1] == TRUE   THEN
     PTP P0
ELSE
     LIN P1
ENDIF
```

■ **范例 2 说明：**

判断如果数位输入$D[1]为真，然后执行以点到点方式移动至点 P0；否则数位输入$D[1]为假，则执行以直线方式移动至点 P1。

■ **范例 3：**

```
IF   $DI[1] == TRUE   AND   $DI[2] == TRUE   THEN
     PTP P1 CONT Vel=100% Acc=100% TOOL[0] BASE[1]
ENDIF
```

■ **范例 3 说明：**

判断如果数位输入$D[1] 和 数位输入$D[2] 两者同时为真，然后执行以点到点方式移动至点 P1，如果数位输入$D[1]为假 或 数位输入$D[2]为假，条件式无法同时成立时，则忽略此判断式，继续执行 ENDIF 后指令码。

■ **参考相关指令：**

$DI[]、WHILE、LOOP、FOR、REPEAT。

2.5.2@FOR 对于回圈

■ **指令说明：**

FOR 回圈是一种常见的控制流程。可能会连续执行多次的程式码或执行特定的次数，等条件成立时结束回圈，或者将程式内所有项目都执行一次。简单来说能够让程式码反复的执行且拥有计数器功能的回圈。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
FOR [变数名称] =[起始值] TO [最终值] STEP [递增值]
    [运动指令]
ENDFOR
```

■ **格式/参数说明：**

⇨ [变数名称] 为使用者所定义的名称，需在 FOR 回圈前作宣告。
注:只能是英文(大小写不限)或英文搭配数字。例:INT H、INT h、INT H1、INT h1。

⇨ [起始值] 为所要执行的回圈的起始数值。

⇨ [最终值] 为所要执行的回圈的最终数值。

⇨ [递增值] 为每一次执行 FOR 回圈所要递增值，为整数值；STEP [递增值]也可以省略不写，预设值递增值为 1。

⇨ ([最终值]- [起始值])/ [递增值] 为所需动作回圈次数。

⇨ [运动指令] 为执行 FOR 判断式之陈述动作，逻辑程式、移动指令或讯号指令之组合。

■ **范例 1：**

```
INT H
FOR H = 0 TO 2 STEP 1
    PTP P0 CONT = 100% Vel=100% Acc=50% TOOL[0]BASE[0]
    PTP P1 CONT = 100% Vel=100% Acc=50% TOOL[0]BASE[0]
ENDFOR
```

■ **范例 1 说明：**

宣告一个整数变数为 H，起始值为 0，最终值为 2，递增值为 1，开始 FOR 回圈 0 计数到 3，共执行 2 次运动指令，运动指令执行点到点 P0，再点到点 P1 移动指令，结束 FOR 回圈。

※程式叙述：从 P0 点到 P1 点来回三次(0、1、2)。

■ **范例 2：**

```
INT H
FOR H = 0 TO 4 STEP 2
   PTP P0 CONT = 100% Vel=100% Acc=50% TOOL[0]BASE[0]
   PTP P1 CONT = 100% Vel=100% Acc=50% TOOL[0]BASE[0]
ENDFOR
```

■ **范例 2 说明：**

宣告一个整数变数为 H，

FOR H = [起始值 2] TO [最终值 4] STEP [递增值 2]

运动指令执行点到点 P0，再点到点 P1 移动指令

结束 FOR 回圈。

※程式叙述：从 P0 点到 P1 点来回三次(0、2、4)。

■ **参考相关指令：**

INT、$DI[]、WHILE、LOOP、REPEAT。

2.5.3@LOOP 循环回圈

■ **指令说明：**

LOOP 回圈内的内容会被重复执行使用者设定的动作内容。

若 LOOP 回圈内根据使用者设定条件符合后，使用离开”EXIT”指令时，程式将会跳离 LOOP 回圈，继续后续其他程式动作。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

✧ 格式 1：

```
LOOP
    [动作指令]
ENDLOOP
```

✧ 格式 2：

```
LOOP
    [动作指令 1]
    IF [条件] THEN
       EXIT
    ENDIF
     [动作指令 2]
ENDLOOP
```

■ **格式/参数说明：**

⇨ “LOOP”为无线回圈开头文字，”ENDLOOP “为无线回圈结尾文字。

⇨ EXIT 可跳离一层 LOOP

⇨ [条件] 为使用者设定条件，例如变数条件设定、讯号条件设定，可为以下逻辑运算子，等条件判断式。提醒：目前无法将两个比较运算子直接连接(例如. 1< 变数 <10)，否则软体会回报比较操作错误讯息("comparison operator error!")

运算子名称	语法	运算子名称	语法
小于	A < B	不等于	A != B
小于或等于	A <= B	等于	A == B
大于	A > B	逻辑取反	!A
大于或等于	A >= B		

⇨ [动作指令] 为执行逻辑程式、移动指令或讯号指令之组合。
⇨ 格式 1 为无限回圈逻辑指令，并执行无限循环作动作指令内容。
⇨ 格式 2 为有条件式的无限回圈逻辑指令，若使用者设定如果(IF)条件符合后，利用离开"EXIT"指令时，程式将会跳离/结束 LOOP 回圈，不再执行(动作指令 2)内容。

■ **范例：**

```
LOOP
    PTP P1 FINE=1 Vel=100% Acc=100% TOOL[0] BASE[0]
    PTP P2 FINE=1 Vel=100% Acc=100% TOOL[0] BASE[0]
ENDLOOP
```

■ **范例 1 说明：**

手臂不断重复 P1 ➔ P2 ➔ P1 ➔ P2 点位移动的无限回圈。

■ **范例 2：**

```
LOOP
    PTP P1 FINE=1 Vel=100% Acc=100% TOOL[0] BASE[0]
    PTP P2 FINE=1 Vel=100% Acc=100% TOOL[0] BASE[0]
    IF $DI[1] == TRUE THEN
        EXIT
    ENDIF
ENDLOOP
```

■ **范例 2 说明：**

手臂不断重复 P1 ➔ P2 ➔ P1 ➔ P2 点位移动的无限回圈，若 DI[1]讯号为真(True)，则跳离 LOOP 回圈结束程式; (若 DI[1]没讯号，则手臂不断重复 P1 ➔ P2 ➔ P1 ➔ P2 点位移动的无限回圈)。

■ **参考相关指令：**

IF、$DI[]、WHILE

2.5.4@WHILE 当回圈

■ **指令说明：**

当(while)回圈为控制流程。利用附加之条件判断式决定是否结束回圈，当这个附加之条件判断式的返回值为 TRUE / 1 时，则反复执行回圈体内的程式码；若附加之条件判断式的返回值为 FALSE / 0，则不再执行回圈内程式，继续执行回圈外之程式码。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
WHILE [条件式]
    [运动指令]
ENDWHILE
```

■ **格式/参数说明：**

⇨ [条件式]为附加之条件判断式，可为以下逻辑运算子，等条件判断式。提醒：目前无法将两个比较运算子直接连接(例如. 1< 变数 <10)，否则软体会回报比较操作错误讯息("comparison operator error!")

运算子名称	语法	运算子名称	语法
小于	A < B	不等于	A != B
小于或等于	A <= B	等于	A == B
大于	A > B	逻辑取反	!A
大于或等于	A >= B		

⇨ [运动指令] 为执行 WHILE 回圈内之陈述动作，逻辑程式、移动指令或讯号指令之组合。

■ **范例：**

```
INT N=0
WHILE N <= 10
    LIN P0 FINE=1
    LIN P1 FINE=1
    N=N+1
    WAIT SEC 0.5
ENDWHILE
```

■ **范例说明：**

宣告一个整数变数N为0值，执行回圈WHILE附加判断式；假设N小于等于10 符合条件式为真(TRUE)，则执行内部程式运动指令，运动指令内容为直线移动P0，再直线移动P1，作完一次后，N值累加数值1，并等待0.5秒后，再次进行WHILE回圈条件式判定；假设N累加至超过10符合条件式为假(FALSE)，则不执行内部程式运动指令，程式继续执行ENDWHILE程式码。

■ **参考相关指令：**

INT、FOR、LOOP、REPEAT、GOTO。

2.5.5@REPEAT 重复回圈

■ **指令说明：**

重复回圈为控制流程。先会执行完一次动作指令后，再根据条件判断式决定是否结束回圈，当这个附加之条件判断式的返回值为 TRUE / 1 时，则反复执行回圈体内的程式码；若附加之条件判断式的返回值为 FALSE / 0，则不再执行回圈内程式，继续执行回圈外之程式码。

重复回圈与当(WHILE)回圈相似，会先全部执行完一次动作指令再作条件式判断，当(WHILE)回圈会一开始就执行条件式判定后，才会执行动作指令

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
REPEAT
    [运动指令]
UNTIL [条件式]
```

■ **格式/参数说明：**

⇨ [条件式]为附加之条件判断式，可为以下逻辑运算子，等条件判断式。提醒：目前无法将两个比较运算子直接连接(例如. 1< 变数 <10)，否则软体会回报比较操作错误讯息("comparison operator error!")

运算子名称	语法	运算子名称	语法
小于	A < B	不等于	A != B
小于或等于	A <= B	等于	A == B
大于	A > B	逻辑取反	!A
大于或等于	A >= B		

⇨ [运动指令] 为执行 REPEAT 回圈内之陈述动作，逻辑程式、移动指令或讯号指令之组合。

■ **范例：**

```
INT N=1
REPEAT
    LIN P0 FINE=1
    LIN P1 FINE=1
    N=N+1
    WAIT SEC 0.5
UNTIL N > 3
```

■ **范例说明：**

宣告一个整数变数N为1值，执行回圈REPEAT；执行内部程式运动指令，运动指令内容为直线移动P0，再直线移动P1，作完一次后，N值累加数值1，并等待0.5秒后，进行REPEAT回圈条件式判定；假设N小于等于3 符合条件式为真(TRUE)，则重复执行内部程式运动指令；假设N累加至超过3符合条件式为假(FALSE)，程式停止执行REPEAT回圈。以上述指令，将从P0点到P1点位执行3次(1、2、3，最后执行一次才作条件判定)。

■ **参考相关指令：**

INT、FOR、LOOP、WHILE、GOTO。

2.5.6@GOTO 去/跳跃指令

■ **指令说明：**

当使用者想在执行程式过程中，透过设定标签方式，使程式从指定位置任意跳跃至目标标签位置时，所使用之指令。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
GOTO [标签名称]

[标签名称]：
[运动指令]
```

■ **格式/参数说明：**

⇨ [标签名称] 为使用者自定义名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，GOTO + [标签名称] 为程式码启动跳曜指令的位置。

⇨ [标签名称]： 为跳曜至目标标签位置，后续再执行(运动指令)。在[标签名称]后需加上”:”(冒号)，辨别为目标标签位置。

⇨ [运动指令] 为执行 GOTO 跳曜指令内之陈述动作，逻辑程式、移动指令或讯号指令之组合。

■ **范例：**

```
INT n = 0
WHILE n < 2
    IF n == 0 THEN
        GOTO STEP0
    ENDIF
    IF n == 1 THEN
        GOTO STEP1
    ENDIF
    NEXT:
    n = n + 1
ENDWHILE

STEP0:
PTP P0 FINE=1 Vel=100% Acc=100% TOOL[0] BASE[0]
GOTO NEXT
STEP1:
```

```
PTP P1 FINE=1 Vel=100% Acc=100% TOOL[0] BASE[0]
GOTO NEXT
```

■ **范例说明：**

当自定整数变数 n 小于 2 时，进入 WHILE 回圈，接着使用 IF 指令判断整数变数 n 为 0 或者为 1，当整数变数 n 为 0 时，利用 GOTO 跳跃至 STEP0 标签执行底下程式，STEP0 标签底下程式为利用 PTP 运动指令将手臂移动至点位 P0 位置，接着利用 GOTO 跳跃至 NEXT 标签执行底下程式，NEXT 标签底下程式则是使整数变数 n 加 1；当整数变数 n 为 1 时，利用 GOTO 跳跃至 STEP1 标签执行底下程式，STEP0 标签底下程式为利用 PTP 运动指令将手臂移动至点位 P1 位置，接着利用 GOTO 跳跃至 NEXT 标签执行底下程式，NEXT 标签底下程式则是使整数变数 n 加 1，当整数变数 n 累加至不小于 2 时，则跳出 WHILE 回圈并结束程式。

■ **参考相关指令：**

INT、WHILE、FOR、REPEAT、LOOP、IF。

2.5.7@SWITCH 开关逻辑指令

■ **指令说明：**

开关逻辑指令 SWITCH 是一个选取范围陈述式，可根据使用「比对运算式」的模式比对，从候选项清单中选择要执行的单一「参数区段」。

开关逻辑指令 SWITCH 陈述式包含一个或多个参数区段。每个参数区段都包含一或多个「case 标签」 (case 或预设标签)，后面接着一或多个陈述式，使用者可以根据当下情况来选择判定所需要的陈述式内容。SWITCH 陈述式在任何参数区段中最多只能放置一个预设标签。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
SWITCH  [条件变数名称]
CASE [变数 1]
    [运动指令 1]
CASE [变数 2]
    [运动指令 2]
CASE [变数 3]
    [运动指令 3]
DEFAULT
    [运动指令 4]
 ENDSWITCH
```

■ **格式/参数说明：**

⇨ [条件变数名称] 为开关条件变数，使用者自定义名称，整数、长整数或字元变数型态。

⇨ [变数 1~3] 为对应已知案件 CASE 数值/字母。

⇨ [运动指令 1~4] 为执行 CASE 指令内之陈述动作，逻辑程式、移动指令或讯号指令之组合。

⇨ ”SWITCH” 为开头字母，并作条件判断式，”ENDSWITCH”为结束字母，”CASE” 为变数条件设定，若[条件变数名称]等于[变数 n]则进行[运动指令 n]，”DEFAULT” 为排他条款，若[条件变数名称]全不等于[变数 n]，则进行预设[运动指令 4]内容，上述 n 为任意数值。

⇨ SWITCH 判断式的[变数]无对应 CASE 时，又无设定预设排他条款”DEFAULT”时，直接 ENDSWITCH。

■ **范例 1：SWITCH 无预设值**

```
INT n =0
LOOP
    SWITCH n
    CASE 0
```

```
        PTP P0 FINE=1 Vel=100% Acc=50% TOOL[0] BASE[0]
    CASE 1
        PTP P1 FINE=1 Vel=100% Acc=50% TOOL[0] BASE[0]
    CASE 2
        EXIT
    ENDSWITCH
n = n + 1
ENDLOOP
```

■ **范例 1 说明：**

宣告整数变数 n，执行 LOOP 回圈，使用 SWITCH 回圈以判断 n 条件变数，当 n = 0，开关至案件 0 (CASE 0)时，末端 TCP 点到点移动至 P0 位置，n 累加数值 1；当 n = 1，开关至案件 1 (CASE 1)时，末端 TCP 点到点移动至 P1 位置，n 累加数值 1；当 n = 2，开关至案件 2 (CASE 2)时，接着执行跳出程式 EXIT 指令，结束 LOOP 回圈。

■ **范例 2：SWITCH 有预设值 DEFAULT**

```
INT n =0
LOOP
    SWITCH n
    CASE 0
        PTP P0 FINE=1 Vel=100% Acc=50% TOOL[0] BASE[0]
    CASE 1
        PTP P1 FINE=1 Vel=100% Acc=50% TOOL[0] BASE[0]
    DEFAULT
        EXIT
    ENDSWITCH
n = n + 1
ENDLOOP
```

■ **范例 2 说明：**

宣告整数变数 n，执行 LOOP 回圈，使用 SWITCH 回圈以判断 n 条件变数，当 n = 0，开关至案件 0 (CASE 0)时，末端 TCP 点到点移动至 P0 位置，n 累加数值 1；当 n = 1，开关至案件 1 (CASE 1)时，末端 TCP 点到点移动至 P1 位置，n 累加数值 1；当 n 值不等于 0 或 1 时，则会运行预设排他条件 (DEFAULT)，执行跳出程式 EXIT 指令，结束 LOOP 回圈。

※SWITCH 的[条件变数名称]对应 CASE [变数]，若无对应则执行 DEFAULT 之陈述。

范例 3：SWITCH 延伸运用-多个变数条件

```
INT n =0
LOOP
    SWITCH n
    CASE 0,2,4
        PTP P0 FINE=1 Vel=100% Acc=50% TOOL[0] BASE[0]
    CASE 1,3
        PTP P1 FINE=1 Vel=100% Acc=50% TOOL[0] BASE[0]
    CASE 5
        EXIT
    ENDSWITCH
n = n + 1
ENDLOOP
```

■ **范例 3 说明：**

宣告整数变数 n，执行 LOOP 回圈，使用 SWITCH 回圈以判断 n 条件变数，当 n = 0,2,4，开关至案件 0,2,4 (CASE 0,2,4)时，末端 TCP 点到点移动至 P0 位置，n 累加数值 1,3；当 n = 1，开关至案件 1,3 (CASE 1,3)时，末端 TCP 点到点移动至 P1 位置，n 累加数值 1；当 n = 5，开关至案件 5 (CASE 5)时，接着执行跳出程式 EXIT 指令，结束 LOOP 回圈。

■ **参考相关指令：**

INT、WHILE、FOR、REPEAT、LOOP。

2.5.8@WAIT SEC 等待秒数指令

■ **指令说明：**

等待秒数讯号指令，让程式执行中让使用者自行设定欲等待的一段时间指令。例如：真实打开气动夹爪时会有时间，可透过此指令，让程式软体暂时停止执行程式，等时间秒数读取完后，再继续进行程式运行。

若加此程式预读功能也会停止，常会在重要讯号或点位前，先作等待讯号。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	不预读 FINE

■ **指令格式：**

```
WAIT SEC [时间数值]
```

■ **格式/参数说明：**

⇨ “WAIT SEC” 为等待秒数指令，后面需填入欲等待秒数数值，等待秒数为参考系统软体时间，可用来档程式系统的底层预先读取程式的功能。

⇨ [时间数值]为等待时间数值，常数正浮点数值，可输入范围为 0~2147483648，单位为秒(Second)，最小增加单位 0.01 秒。(不建议写入 WAIT SEC 0，物理现象无法解释)

⇨ 自 HRSS3.3.23 版本后，WAIT SEC 数值可使用变数做切换。

■ **范例 1：**

```
PTP P1
WAIT SEC 1.15
LIN P2
```

■ **范例 1 说明：**

以点到点方式移动至 P1 点，等待 1.15 秒后，在执行以直线方式移动至 P2 点。

■ **范例 2：**

```
$DO[1] = TRUE
WAIT SEC 2
$DO[1] = FALSE
```

■ **范例 2 说明：**

开启数位输出讯号 DO[1]，等待 2 秒后，再关闭数位输出讯号 DO[1]。
(此$DO[1]可能为控制气动夹爪，故需给予实际气爪开合动作时间)

■ **参考相关指令：**

WAIT FOR、$DO

2.5.9@WAIT FOR 等待输入讯号指令

■ **指令说明：**

等待外部输入讯号指令，让程式执行中让使用者自行设定欲等待的输入讯号指令。例如：等待入料盘输入讯号，程式确定有入料盘讯号完成后才会计续后续动作。若等待的输入讯号一值没有给予(例：感测器损坏)则程式会一值等待，不会有任何动作，故需要人工检查输入讯号来源问题。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	不预读 FINE

■ **指令格式：**

```
WAIT FOR  [输入讯号] = = [布林变数]
```

■ **格式/参数说明：**

⇨ “WAIT FOR” 为等待讯号指令，后面需填入欲等待输入讯号格式。

⇨ [输入讯号] 为输入来源，有四个讯号来源可以选择，Digital 数位输入讯号($DI[])、Robot 机器人输入讯号($RI[])、FieldBus 现场总线输入讯号($SI[])或组合输入讯号($MI[])。

⇨ [布林变数] 为真 TRUE 表示启动或假 FALSE 表示关闭，布林逻辑变数，二者择一状态。

■ **范例 1：**

```
PTP P1
WAIT FOR $DI[1] == TRUE
LIN P2
```

■ **范例 1 说明：**

以点到点方式移动至 P1 点，等待数位输入讯号为真(TRUE)后，在执行以直线方式移动至 P2 点。

■ **范例 2：**

```
$SO[1] = TRUE
WAIT FOR $SI[1] == FALSE
PTP P1
```

■ **范例 2 说明：**

开启编号 1 现场总线输出讯号($SO[1])，等待编号 1 现场总线输入讯号($SI[1])后，再开始以点到点方式移动到 P1 点。

(此$SO[1]可能为通知上位控制器 PLC 等通知讯号，需由上位给予手臂输入讯号才能动作)

■ **参考相关指令：**

WAIT SEC、$DI[]、$RI[]、$SI[]、$MI[]。

2.5.10 @WAIT FOR … TIMEOUT 输入讯号逾时指令

■ **指令说明：**

在等待 IO 讯号时，可以指定等待时间，当等待逾时后程式会跳到指定标签(LABLE)做特定处理。

如下图流程图，程式先判定讯号是否条件式成立，再判定是否逾时，若逾时则往使用者设定的标签作后续程式 B 的动作运行。

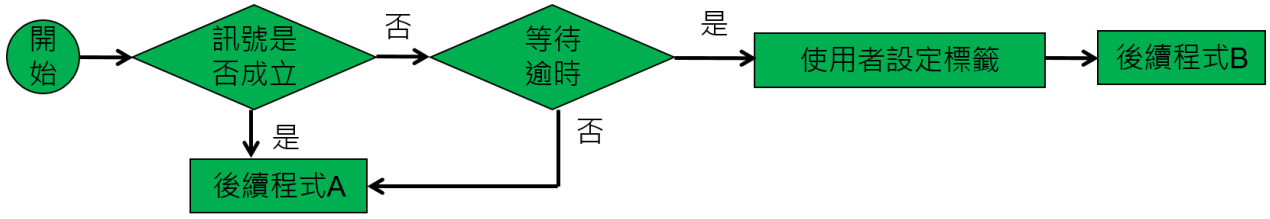


图 20 WAIT FOR … TIMEOUT 流程图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.10 以上	不预读 FINE

■ **指令格式：**

```
WAIT FOR [数位输入讯号] == [布林变数] TIMEOUT( [等待时间],[使用者设定标签] )
```

■ **格式/参数说明：**

- ⇨ [数位输入讯号]：等待数位输入($DI[])讯号。
- ⇨ [布林变数] 为真 TRUE 或假 FALSE，布林逻辑变数，二者择一状态。
- ⇨ [等待时间]：等待逾时时间。预设 0.1 秒，最小 0.1 秒单位值，最大数值 86400 秒(一天)
- ⇨ [使用者设定标签]：等待时间逾时后，就切换执行标签(lable)动作，需由使用者自行定义标签名称，并决定标签放置位置，当时间逾时后，符合使用者自行定义程式逻辑运行。

■ **范例：**

```
LOOP
    PTP P0 FINE = 1
    WAIT FOR $DI[1] == TRUE TIMEOUT(10,Safetypoint)
    LIN P1
ENDLOOP

Safetypoint:    ;标签
LIN P2 Vel= 500 mm/s
USER_ALARM[1]
```

■ **范例说明：**

宣告 LOOP …ENDLOOP 回圈，重复执行以下 3 行程式:

点到点移动至 P0 位置，等待第一个数位输入讯号为真(TURE)后，直线移动至 P1 点位。

若等待第一个数位输入讯号时间超过10秒中，则执行标签Safetypoint:之程式，会改为直线移动至P2点位速度调整500每秒毫米，并发出第一个使用者自定义错误(USER_ALARM[1])。

■ **参考相关指令：**

LOOP、LIN、WAITFOR、USER_ALARM。

2.5.11 @AND 且逻辑闸指令

■ **指令说明：**

逻辑运算子之一，且(AND)逻辑闸，若有 A 条件与 B 条件，需要两个条件同时符合才会作执行动作，如下表与下图为不同条件会发生的 AND 逻辑闸判定结果。

表 5 且(AND)逻辑闸判定结果表

A 条件	B 条件	R 结果
0	0	0
0	1	0
1	0	0
1	1	1

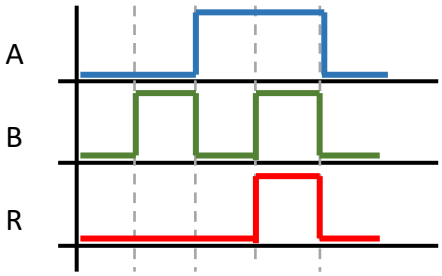


图 21 且(AND)逻辑闸判定结果时序图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

[条件 A] AND [条件 B]

■ **格式/参数说明：**

⇨ AND 为且(AND)逻辑闸指令，需搭配其他逻辑功能使用，如 IF、WHILE。。

⇨ [条件 A]、[条件 B]为使用者设定条件，例如变数条件设定、讯号条件设定，可为以下逻辑运算子，等条件判断式。提醒：目前无法将两个比较运算子直接连接(例如. 1< 变数 <10)，否则软体会回报比较操作错误讯息("comparison operator error!")

运算子名称	语法	运算子名称	语法
小于	A < B	不等于	A != B
小于或等于	A <= B	等于	A == B
大于	A > B	逻辑取反	!A
大于或等于	A >= B		

⇨ 需分别不同使用情况状态可搭配小括号()作分类，()内建议判断式数量不超出 30 个条件。

■ **范例 1：**

```
PTP P0 FINE = 1
IF $DI[2] == TRUE AND $DI[3] == TRUE THEN
    PTP P1
ENDIF
LIN P3
```

■ **范例 1 说明：**

以点到点方式移动至 P0 点位，

IF 判断式，当第 2 道数位输入($DI[2])与第 3 道数位输入($DI[3])讯号皆为真(TRUE)时，条件成立才会点到点移动至 P1 点位，再直线移动至 P3 点位。

若条件无法成立则不移动至 P1 点位。

结果： $DI[2] 且$DI[3]条件皆成立，移动路径 P0→P1→P3 点位；$DI[2]与$DI[3]条件其一皆不成立，移动路径 P0→P3 点位。

■ **范例 2：**

```
INT X1 = 0
PTP P0 FINE=1
WHILE (X1<3) AND ($C[3] == 1)
    PTP P1
    PTP P0
    X1 = X1 +1
ENDWHILE
LIN P3
```

■ **范例 2 说明：**

宣告整数变数 X1，初始值为 0，以点到点方式移动至 P0 点位，

当(WHILE)判断式，当(X1<3)且($C[3] == 1)皆成立时，点到点方式移动至 P0 点位，点到点方式移动至 P1 点位，再作完一次将 X1 数值加 1，直至当(WHILE)条件式不成立后，再直线移动至 P3 点位。

结果：(X1<3)且($C[3] == 1)条件皆成立时，移动路径 P0→(P1→P0)*3 次→P3 点位；

(X1<3)且($C[3] == 1)条件其一皆不成立，移动路径 P0→P3 点位。

■ **参考相关指令：**

IF、WHILE、OR、$C[]、INT。

2.5.12 @OR 或逻辑闸指令

■ **指令说明：**

逻辑运算子之一，或(OR)逻辑闸，若有 A 条件与 B 条件，需要两个条件只要其中一个符合就会作执行动作，如下表与下图为不同条件会发生的 OR 逻辑闸判定结果。

表 6 且(OR)逻辑闸判定结果表

A 条件	B 条件	R 结果
0	0	0
0	1	1
1	0	1
1	1	1

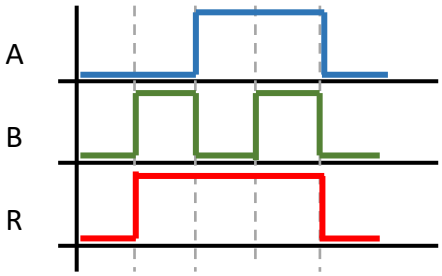


图 22 或(OR)逻辑闸判定结果时序图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

[条件 A] OR [条件 B]

■ **格式/参数说明：**

⇨ OR 为或(OR)逻辑闸指令，需搭配其他逻辑功能使用，如 IF、WHILE。。

⇨ [条件 A]、[条件 B]为使用者设定条件，例如变数条件设定、讯号条件设定，可为以下逻辑运算子，等条件判断式。提醒：目前无法将两个比较运算子直接连接(例如. 1< 变数 <10)，否则软体会回报比较操作错误讯息("comparison operator error!")

运算子名称	语法	运算子名称	语法
小于	A < B	不等于	A != B
小于或等于	A <= B	等于	A == B
大于	A > B	逻辑取反	!A
大于或等于	A >= B		

⇨ 需分别不同使用情况状态可搭配小括号()作分类，()内建议判断式数量不超出 30 个条件。

■ **范例 1：**

```
PTP P0 FINE = 1
IF $DI[2] == TRUE OR $DI[3] == TRUE THEN
    PTP P1
ENDIF
LIN P3
```

■ **范例 1 说明：**

以点到点方式移动至 P0 点位，

IF 判断式，当第 2 道数位输入($DI[2])与第 3 道数位输入($DI[3])讯号其中一个为真(TRUE)时，条件成立才会点到点移动至 P1 点位，再直线移动至 P3 点位。

若条件无法成立则不移动至 P1 点位。

结果： $DI[2] 或 $DI[3]条件其中一个成立，移动路径 P0→P1→P3 点位；

只有在$DI[2]与$DI[3]条件同时皆不成立，移动路径 P0→P3 点位。

■ **范例 2：**

```
INT X1 = 0
PTP P0 FINE=1
WHILE (X1<3) OR ($C[3] == 1)
    PTP P1
    PTP P0
    X1 = X1 +1
ENDWHILE
LIN P3
```

■ **范例 2 说明：**

宣告整数变数 X1，初始值为 0，以点到点方式移动至 P0 点位，

当(WHILE)判断式，当(X1<3)且($C[3] == 1) 其中一个成立时，点到点方式移动至 P0 点位，点到点方式移动至 P1 点位，再作完一次将 X1 数值加 1，直至当(WHILE)条件式皆不成立后，再直线移动至 P3 点位。

结果：(X1<3)或($C[3] == 1)条件其中一个成立时，移动路径 P0→(P1→P0)*3 次→P3 点位；

只有在(X1<3)或($C[3] == 1)条件同时皆不成立，移动路径 P0→P3 点位。

■ **参考相关指令：**

IF、WHILE、AND、$C[]、INT。

2.5.13 @INTERRUPT…WHEN…DO… 宣告中断条件指令

■ **指令说明：**

中断指令优先顺序、触发条件与副程式设定指令宣告，此指令只用于做中断指令宣告，若需开启需参考中断开启/关闭指令。 中断指令需宣告触发讯号条件、讯号的优先度顺序并指定中断后执行的副程式名称，中断副程式执行完后，会接续原本尚未执行完的动作。

可应用案例举例：若手臂应用在搬移 A 工件循环动作时，若因感测器讯号触发条件成立后，手臂就执行另一个副程式去搬移 B 工件，并执行完搬移 B 工件后，会在回到原本搬移 A 工件循环动作。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.5 以上	Caterpillar1.0.4 以上	预读 CONT

⇨ 宣告完成中断指令设定，需再使用开启中断讯号指令，才会开启中断任务功能。

■ **指令格式：**

INTERRUPT [优先顺序值] WHEN [讯号条件判定] DO [副程式名称]

■ **格式/参数说明：**

⇨ [优先顺序值] 为定义中断讯号动作的优先顺序值，正整数值，可输入 1~8 数值，1 为最优先度最高。

⇨ [讯号条件判定] 为(数位$DI)输入讯号，例如：$DI[1] == TRUE 。讯号开关一次，只会执行一次中断；讯号一直开启不会一直执行中断指定的副程式。

⇨ [副程式名称] 为使用者自定义副程式名称，例如：PRO_1()，当中断讯号条件判定成立时，就会执行副程式的内容；执行完副程式后，就会再继续原本的未完成程式动作。

■ **范例：**

```
INTERRUPT 1 WHEN $DI[1] == TRUE DO Subprogram1()
INTERRUPT ON 1
PTP P0 FINE=1 Vel=100% Acc=100% TOOL[0] BASE[0]
PTP P1 FINE=1 Vel=10% Acc=100% TOOL[0] BASE[0]

DEF Subprogram1()
    PTP P2 CONT Vel=100% Acc=100% TOOL[0] BASE[0]
    WAIT FOR $DI[1] == FALSE
END
```

■ **范例说明：**

宣告中断指令，优先顺序为 1，判断触发讯号条件为数位输入讯号开启($DI[1] == TRUE)，条件成立后将执行副程式 Subprogram1。并开启优先顺序为 1 的中断指令动作。

以点到点移动方式 P0 点至 P1 点，当移动途中，发生数位输入第 1 个讯号开启($DI[1] == TRUE)时，将中断现在 P0 至 P1 动作，而改执行 Subprogram1 副程式动作。

副程式动作内容为以点到点移动方式 P2 点(安全点)，并等待数位输入第 1 个讯号关闭($DI[1] == FALSE) 。

■ **参考相关指令：**

BRAKE、$DI[]、DEF…END、INTERRUPT ON/OFF 。

2.5.14 @INTERRUPT ON/OFF 启用/停止中断指令

■ **指令说明：**

启用/停止中断指令；需先透过宣告中断条件指令宣告完后(INTERRUPT … WHEN … DO…)，才能有效应用开启/停止中断指令。 此指令主要为段落间控制开关中断指令动作。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.5 以上	Caterpillar1.0.4 以上	预读 CONT

■ **指令格式：**

```
INTERRUPT 开 [优先顺序值]
INTERRUPT ON [优先顺序值]
或
INTERRUPT 关 [优先顺序值]
INTERRUPT OFF [优先顺序值]
```

■ **格式/参数说明：**

⇨ 开 ON /关 OFF 为中断指令开启/关闭控制，ON 为开启中断指令，OFF 为关闭中断指令，指令只能选择开或关，二择一。

⇨ [优先顺序值] 为定义中断讯号动作的优先顺序值，正整数值，可输入 1~8 数值，1 为最优先度最高；或可以输入 ALL 指令，让中断指令一次全开启或一次全关闭。

■ **范例 1：**

```
INTERRUPT 1 WHEN $DI[1] == TRUE DO Subprogram1()
INTERRUPT ON 1
PTP P0 FINE=1 Vel=100% Acc=100% TOOL[0] BASE[0]
PTP P1 FINE=1 Vel=10% Acc=100% TOOL[0] BASE[0]
INTERRUPT OFF 1

DEF Subprogram1()
    PTP P2 CONT Vel=100% Acc=100% TOOL[0] BASE[0]
    WAIT FOR $DI[1] == FALSE
END
```

■ **范例 1 说明：**

宣告中断指令，优先顺序为 1，判断触发讯号条件$DI[1] == TRUE，条件成立后将执行副程式 Subprogram1。并开启优先顺序为 1 的中断指令动作。

以点到点移动方式 P0 点至 P1 点，当移动途中，发生$DI[1] == TRUE 时，将中断现在 P0 至

P1 动作，而改执行 Subprogram1 副程式动作。

副程式动作内容为以点到点移动方式 P2 点(安全点)，并等待$DI[1] == FALSE。

待副程式动作完成，且会到 P1 点位完成后，将关闭优先顺序为 1 的中断指令动作。

■ **范例 2：**

```
INTERRUPT 1 WHEN $DI[1] == TRUE DO Subprogram1()
INTERRUPT 2 WHEN $DI[2] == TRUE DO Subprogram2()
INTERRUPT ON ALL
PTP P0 FINE=1 Vel=100% Acc=100% TOOL[0] BASE[0]
PTP P1 FINE=1 Vel=10% Acc=100% TOOL[0] BASE[0]
INTERRUPT OFF ALL

DEF Subprogram1()
    PTP P2 CONT Vel=100% Acc=100% TOOL[0] BASE[0]
    WAIT FOR $DI[1] == FALSE
END
DEF Subprogram2()
    PTP P3 CONT Vel=100% Acc=100% TOOL[0] BASE[0]
    WAIT FOR $DI[2] == FALSE
END
```

■ **范例 2 说明：**

宣告中断指令，优先顺序为 1，判断触发讯号条件$DI[1] == TRUE，条件成立后将执行副程式 Subprogram1。

宣告中断指令，优先顺序为 2，判断触发讯号条件$DI[2] == TRUE，条件成立后将执行副程式 Subprogram2。

将开启所有优先顺序(1&2)的中断指令动作。

以点到点移动方式 P0 点至 P1 点，当移动途中，发生$DI[1] == TRUE 时，将中断现在 P0 至 P1 动作，而改执行 Subprogram1 副程式动作。；当移动途中，发生$DI[2] == TRUE 时，将中断现在 P0 至 P1 动作，而改执行 Subprogram2 副程式动作。其中$DI[1]优先顺序大于$DI[2]。

Subprogram1 副程式动作内容为以点到点移动方式 P2 点(安全点)，并等待$DI[1] == FALSE。

Subprogram2 副程式动作内容为以点到点移动方式 P3 点(安全点)，并等待$DI[2] == FALSE。

待副程式动作完成，且会到 P1 点位完成后，将关闭所有优先顺序(1&2)的中断指令动作。

■ **参考相关指令：**

INTERRUPT … WHEN … DO… , BRAKE, $DI[], DEF…END。

2.5.15 @BREAK_POINT 程式中断点指令

■ **指令说明：**

程式中断点指令，运行到该指令时将暂停程式，可让程式停下，供使用者作确认、检查使用，藉由程式运转键可再继续执行中断点后程式。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.20 以上	Caterpillar1.0.4 以上	不预读 FINE

■ **指令格式：**

```
BREAK_POINT
```

■ **格式/参数说明：**

⇨ “BREAK_POINT” 为等程式中断点指令，可让程式停下，作确认、检查使用，若确认完必须再需由操作者再次点选程式运转键，可接续后续的程式运转，同一个程式内可以设定多个程式中断点。

■ **范例：**

```
PTP P1
$C[1] = 5
BREAK_POINT
LIN P2
```

■ **范例说明：**

以点到点方式移动至 P1 点，第 1 个计数暂存器数值等于 5 后，程式新增一个中断点，需由操作者再次点选程式运转键后，程式继续执行，直线方式移动至 P2 点。

■ **参考相关指令：**

WAIT SEC、BRAKE、INTERRUPT … WHEN … DO…、INTERRUPT ON/OFF。

2.6 代码 F、运动参数&系统设定指令

此章节针对于代码 F 的运动参数设定指令做各别说明

2.6.1@SET_OVERRIDE_SPEED 复写总体速度指令

■ **指令说明：**

在程式中使用此指令，用于设定动作中改变程式整体速度百分比，调整速度 SPEED 调整线条。设置后，可放于程式初始运行速度或任一行程式中，即为设定速度数值百分比。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

- 使用指令编辑，不限制放在主程式内位置，执行至该行程式码程式才开始做速度切换。

■ **指令格式：**

```
SET_OVERRIDE_SPEED [参数 1]
```

■ **格式/参数说明：**

⇨ 参数变数型别为正整数，[参数 1]可输入范围为 1 至 100，表示手臂运动执行时的速度为 0%~100%，[参数 1]不可为 0，输入的参数代表运行的整体速度百分比。若手动/自动执行程式后，预设执行复写总体速度为 10%。

■ **范例：**

```
SET_OVERRIDE_SPEED 100
PTP P0
WAIT SEC 1
SET_OVERRIDE_SPEED 55
LIN P1
```

■ **范例说明：**

当执行手臂程式后，手臂整体速度会先至 100%，点到点移动至 P0 点，等后 1 秒后，手臂整体速度会更改至 55%，直线移动至 P1 点。

■ **参考相关指令：**

SET_SPEED、SET_ACC。

2.6.2@SET_SPEED 设定速度指令

■ **指令说明：**

在程式中使用此指令，可以于动作中设定直线轨迹或圆形轨迹的移动速度；通常使用在速度需多段作调整时，且程式单行点位中速度(Vel=)通常忽略不作定义时使用。



1. 使用 SET_SPEED 指令可使机器手臂获得更高的运行速度，但是过高的设定值，可能会超出机器手臂的负载，导致发生错误警报。
2. 请根据实际使用条件，适当地调整参数，避免因过高的运行速度造成设备毁损。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
SET_SPEED [速度参数]
```

■ **格式/参数说明：**

⇨ [速度参数]为设定速度的正整参数值，不可为 0，依机型不同而有不同的预设值。

⇨ 输入的参数代表设定的速度，单位为 mm/s，六轴(RA6)范围介于 1 到 2000 之间、Delta(RD4)范围介于 1 到 6000 之间，SCARA(RS4)范围 1 到 4000 之间(除了 RS406 范围 1 到 3200 之间)。

■ **范例 1：**

```
SET_SPEED 2000
LIN P0
CIRC P1 P2
```

■ **范例 1 说明：**

设定一般运动的进给速度，为2000mm/s。

机器手臂使用直线运动方式移动到目标点P0。

以P0点为起点、P1点为圆弧过度点、P2点终止点做圆弧运动。

※Vel、Acc、TOOL、BASE 可省略输入。省略时将参考上一行指令数值，或初始定义参数。

■ **范例 2： (HRSS3.3 以后速度可填入变数)**

```
INT High_SPEED = 2000
INT Low_SPEED = 500
SET_SPEED Low_SPEED
LIN P0
SET_SPEED High _SPEED
LIN P1
```

■ **范例 2 说明：**

宣告整数型态变数 High_SPEED 高速，初始值为 2000mm/s。
宣告整数型态变数 Low_SPEED 低速，初始值为 500mm/s。
更改速度为低速 Low_SPEED (500mm/s)，并直线移动至 P0 点。
更改速度为高速 High_SPEED (2000mm/s)，并直线移动至 P1 点。

■ **参考相关指令：**

SET_ROTATION_SPEED。

2.6.3@SET_ACC 设定加速度指令

■ **指令说明：**

设定程式执行的加速度与减速度指令，可以设定动作的 100%加/减速度时间。通常使用在加速度需多段作调整时，且程式单行点位中加速度(Acc=)通常忽略不作定义时使用。此为加速至所需的动作速度的所用时间，单位为 ms，范围介于 20 到 1000 之间。

若无使用此设定，依机型不同而有不同的预设值。

⚠ 警告	1.使用 SET_ACC 指令可使机器手臂获得更高的运行度，但是过低的设定值，可能会超出机器手臂的负载，导致发生错误警报。 2. 请根据实际使用条件，适当地调整参数，避免因过高的运行速度造成设备毁损。

例如：RA605-710-GB Payload 为 0 kg 的加速度时间预设值：176 ms(预设值由原厂根据不同机种给予其值)。

根据 V-T 曲线图，此指令设定值即为△t 值，如下图中的第(1)、(3)区间加速度设定，决定手臂动作的每个动作初始加速度、结尾加速度设定((2)等加速度等区间)。

T：时间。

V：速度。

△t：加速度时间。

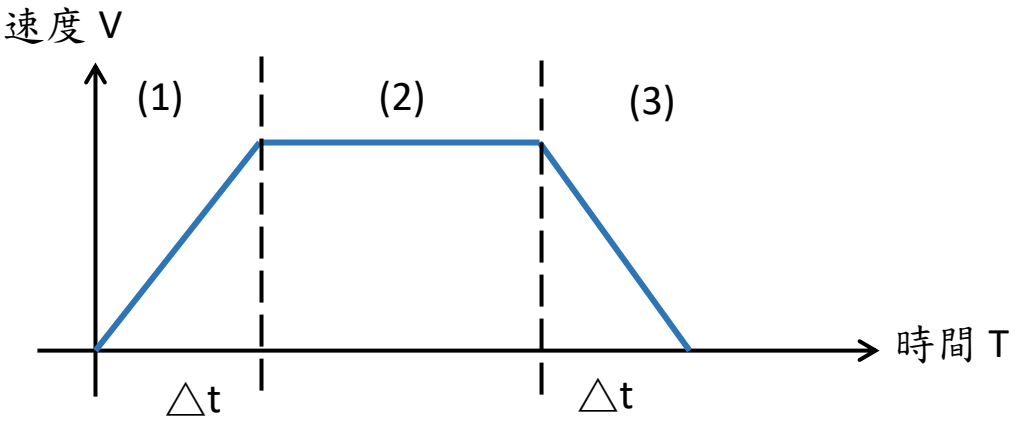


图 23 速度-时间(V-T)曲线图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

SET_ ACC [加速度参数值]

■ **格式/参数说明：**

⇨ [加速度参数值]变数型别为正整数值，不可为 0，范围介于 20 到 1000 之间，数值越小代表加/减速度越快。若遇到单行设定 Acc 值，加速度值会被取代，通常使用者省略单行程式码加速度设定值(Acc)，利用 SET_ ACC 来做区间加/减速度调整，建议修改参数值逐步调整勿一次调整超过 50ms，可能有突然加速或顿点等现象产生，需特别注意。

表 7 加速度参数预设数值表

手臂规格 / 参数名称	RS405	RS410	RA605	RA610	RA620	RD403
加速度设定(毫秒) SET_ACC (ms)	120	110~170	55~176	265~468	280~315	222

■ **范例 1：**

```
SET_ACC 500
PTP P0 CONT Vel=100% Acc=50% TOOL[0] BASE[0]
```

■ **范例 1 说明：**

设定动作的加速度时间为 500 ms。再使用点到点(PTP)移动方式，以圆滑轨迹、速度 100%、加速度 50%、工具座标[0]、基座座标[0]方式，执行到 P0 点位置，此时第二行已更改加速度为 50%，非再是加速度时间为 500 ms，而是 500/ (50%)=1000ms。

■ **范例 2：**

```
SET_ACC 500
PTP P0
WAIT SEC 1
SET_ACC 200
PTP P1
```

■ **范例 2 说明：**

设定动作的加速度时间为 500 ms，再使用点到点(PTP)移动方式至 P0 点位置，等待 1 秒钟，设定动作的加速度时间为 200 ms，再使用点到点移动方式至 P1 点位置，

因PTP省略加速度参数，将参考上一行指令，或初始定义参数。使用者可以在程式中区段自行调整所想要的加速度数值。

注：行程式码省略CONT/FINE、Vel、Acc、TOOL[]、BASE[]设定值，以便后续整体调整速度用。

■ **参考相关指令：**

SET_SPEED 、SET_ROTATION_SPEED。

2.6.4@SET_ROTATION_SPEED 设定姿态旋转速度指令

■ **指令说明：**

在程式中使用此指令，可以于动作中设定姿态旋转的速度。



警告

1. 使用 SET_ROTATION_SPEED 指令可使机器手臂获得更高的运行速度，但是过高的设定值，可能会超出机器手臂的负载，导致发生错误警报。
2. 请根据实际使用条件，适当地调整参数，避免因过高的运行速度造成设备毁损。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

SET_ROTATION_SPEED [旋转速度参数]

■ **格式/参数说明：**

⇨ [旋转速度参数]为设定旋转速度的正整数参数值，单位为 deg/sec，可以限制或提升马达转速，[旋转速度参数]无法使用变数替换，只能给予数值。

⇨ 六轴(RA6)范围介于 1 到 500 之间，DELTA (RD4)范围介于 1 到 2400 之间，SCARA (RS4)范围介于 1 到 2400 之间。

■ **范例：**

```
SET_ROTATION_SPEED 100
PTP P0
```

■ **范例说明：**

设定姿态旋转的速度，单位 100 deg/sec，并点到点方式移动至 P0 点。

■ **参考相关指令：**

SET_SPEED。

2.6.5@SET_TOOL 设定工具座标系指令

■ **指令说明：**

使用此指令可以选取指定的 TOOL 设定，或是改变目前已选取的 TOOL 设定参数。通常使用在多个末端工具切换作调整的指令，且程式单行点位中工具座标(TOOL[#])需忽略不作定义。

工具座标系更变示意图(左图为更变前；右图为更变后)

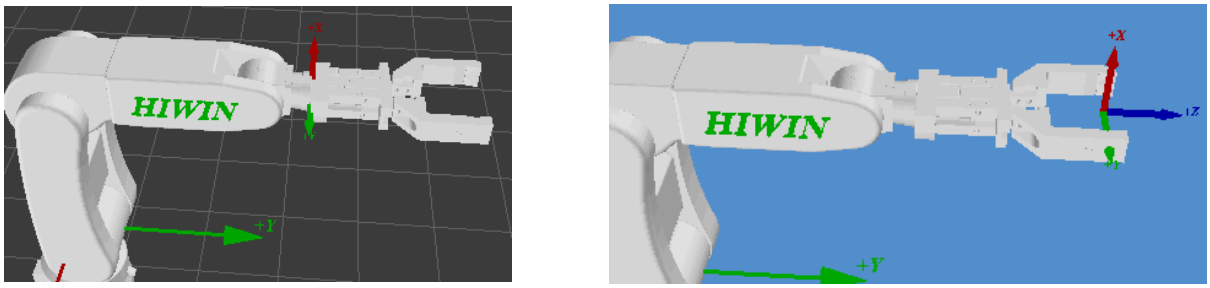


图 24 工具座标系更变示意图(左图为更变前；右图为更变后)图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	不预读 FINE

■ **指令格式：**

```
SET_TOOL  [工具参数值]
```

■ **格式/参数说明：**

⇨ [工具参数值] 为设定工具座标系编号数值，可输入正整数值及框架(FRAME)两种型式。

⇨ 正整数值可输入范围为 0 至 15，使手臂选取该编号的工具 Tool 设定，右上角介面的工具 Tool 编号也会改变，预设为 TOOL[0]。

⇨ 若输入框架 FRAME，则会使目前已选取的 Base 设定参数改变为 FRAME 的数值，需先再程式前端宣告 FRAME 变数。

⇨ 当使用此指令时要将程式内每行点位预设的格式 TOOL[#]删除。

■ **范例 1：**

```
SET_TOOL 1
LIN P1 FINE=1 Vel=3000mm/s Acc=100%
```

■ **范例 1 说明：**

切换工具 TOOL 座标为第一组参数，直线运动移动到 P1 点(第二行后方指令需省略 TOOL[#])。

■ **范例 2：**

```
FRAME T_ONE
T_ONE.X = 100
SET_TOOL T_ONE
LIN P1 FINE=1 Vel=3000mm/s Acc=100%
```

■ **范例 2 说明：**

宣告框架(FRAME)变数 T_ONE，T_ONE 参数 X 轴为 100mm，将目前使用的座标系 TOOL 变更为 T_ONE 变数，直线运动移动至 P1 点。(第二行后方指令需省略 TOOL[#])。

■ **参考相关指令：**

FRAME、SET_BASE。

2.6.6@SET_BASE 设定基座座标系指令

■ **指令说明：**

使用此指令可以选取指定的 Base 设定，或是改变目前已选取的 Base 设定参数。同时介面右上角的 Base 小画面也会更变为使用者指定基座 Base 编号。通常使用在多个基座座标切换作调整的指令，且程式单行点位中基座座标(BASE[#])需忽略不作定义。

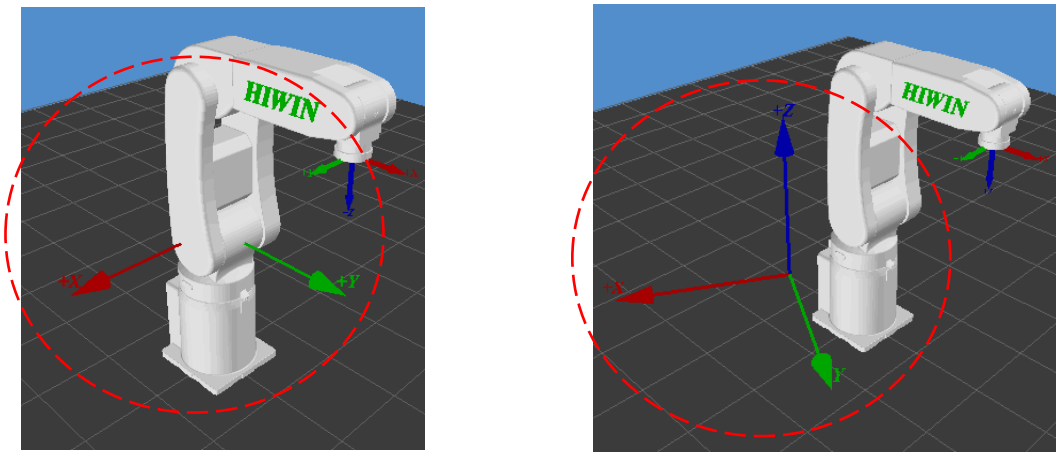


图 25 基座座标系更变示意图(左图为更变前；右图为更变后)图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	不预读 FINE

■ **指令格式：**

SET_BASE [基座参数值]

■ **格式/参数说明：**

⇨ [基座参数值] 为设定基座座标系编号数值，可输入正整数值及框架 FRAME 两种型式。

⇨ 正整数可输入范围为 0 至 31，使手臂选取该编号的 BASE 设定，右上角介面的 BASE 编号也会改变，预设为 BASE[0]。

⇨ 若输入框架 FRAME，则会使目前已选取的 Base 设定参数改变为 FRAME 的数值。需先再程式前端宣告 FRAME 变数。

⇨ 当使用此指令时要将程式内每行点位预设的格式 BASE[#]删除。

■ **范例 1：**

```
SET_BASE 1
LIN P1 FINE=1 Vel=3000mm/s Acc=100%
```

■ **范例 1 说明：**

切换基座 BASE 座标为第一组参数，直线运动移动到 P1 点(第二行后方指令需省略 BASE[#])。

■ **范例 2：**

```
FRAME B_ONE
B_ONE.Y = 100
SET_BASE B_ONE
LIN P1 FINE=1 Vel=3000mm/s Acc=100%
```

■ **范例 2 说明：**

宣告框架(FRAME)变数 B_ONE，B_ONE 参数 Y 轴为 100mm，将目前使用的座标系 BASE 变更为 B_ONE 变数，直线运动移动到 P1 点。

■ **参考相关指令：**

FRAME、SET_TOOL。

2.6.7@SET_PAYLOAD 设定末端负载重量资料指令

■ **指令说明：**

切换所设定完成的末端负载重量资料指令，用于程式中改变不同负载设定数值。
输入的参数代表负载资料组数，需在负载设定参数页面完成设定，指令才有意义。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
SET_PAYLOAD[负载设定编号]
```

■ **格式/参数说明：**

⇨ [负载设定编号] 为末端负载质量值之设定编号值，数值型别为正整数，输入范围编号为 1 至 20 组。

⇨ 末端负载重量值包含为手臂第六轴法兰面之后，所承受的重量总合，含欲夹取工件、夹爪与板金等治具；根据不同手臂规格可输入不同范围，例如：5 公斤六轴建议范围为 0 至 5，其重量设定会影响加/减加速度值。

➢ 范例：

```
SET_PAYLOAD[1]
LIN P1
$DO[1] = TRUE

SET_PAYLOAD[2]
LIN P2
```

■ **范例说明：**

- 设定为第1组负载质量编号；(假设为2公斤，只有夹爪的负载尚未加上物件重量)。
- 以直线方式移动至P1点，开启第1数位输出讯号$DO[1]，抓取物件(增加重量后)；
- 切换设定为第2组负载质量编号，再以直线方式移动至P5点；(假设为3公斤，夹爪的负载且加上物件重量)。

■ **参考相关指令：**

$DO[]

2.6.8@GETPOINT 取得目前的座标值指令

■ **指令说明：**

取得当前手臂位置的座标(X, Y, Z, A, B, C)及角度值(A1, A2, A3, A4, A5, A6)，并将手臂位置资讯放置使用者想要的位置变数值(可利用 E6POINT 来宣告位置变数)，距离单位为 mm，角度单位为度。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

[位置变数名称] = **GETPOINT**

■ **格式/参数说明：**

⇨ [位置变数名称]为 E6POINT 使用者自定义宣告位置座标变数；或也可以使用点位视窗的点位资讯，例如：P0、P1…Pn，n 为任意整数，当为位置变数。

⇨ **GETPOINT** 执行此程式码，即取得目前位置的座标值及角度值，后续可用来运算使用。

■ **范例：**

```
E6POINT Newpoint1
Newpoint1 = GETPOINT
Newpoint1.Z = Newpoint1.Z - 10
PTP Newpoint1
```

■ **范例说明：**

使用 **E6POINT** 宣告位置变数 Newpoint1，将当前手臂位置值转移给 Newpoint1，Newpoint1 Z 轴位置累加-10mm，并使用点到点方式移动至新 Newpoint1 位置；每此执行以上程序，以当下位置为参考让 Z 轴能够移动 10mm。

■ **参考相关指令：**

E6POINT、E6POS、E6AXIS

2.6.9@GET_MOTION_STATUS 读取目前运动状态指令

■ **指令说明：**

读取目前运动状态，取得手臂现今使用状态。读取数值是 0 时为闲置状态(Idle)、-1 时为运动状态(Running)、2 时为暂停状态(Hold)。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
[变数名称] = GET_MOTION_STATUS
```

■ **格式/参数说明：**

⇨ [变数名称]为 REAL 或 INT 使用者自定义宣告变数，并记录手臂状态变数值。

变数名称用于记录数值，手臂各状态之代号如下：

状态 / 代码	手臂状态
0	闲置(Idle)
-1	运动中(Running)
2	暂停 Hold

备注：目前 HRSS 为单执行序列方式，无法在暂存器中显示出动作中与暂停状态(Hold)，但在远端连线(SDK)时才具备其功能性。

■ **范例 1：**

```
PTP P0 CONT
$C[2]   = GET_MOTION_STATUS
PTP P1
WAIT SEC 1
$C[2]   = GET_MOTION_STATUS
```

■ **范例 1 说明：**

点到点移动方式至 P0 点，连续轨迹(CONT)，故会先预独程式，在 P0 点尚在移动途中，读取目前运动状态并将数值传送给第 2 个计时暂存器内，此时状态值为-1(运动中)。

点到点移动方式至 P1 点，等待 1 秒钟后，再读取目前运动状态并将数值传送给第 2 个计时暂存器内，此时状态值为 0(闲置)。

■ **范例 2：**

```
PTP P0 CONT
   LOOP
        IF GET_MOTION_STATUS == 0 THEN
                PTP P1
                EXIT
      ENDIF
ENDLOOP
```

■ **范例 2 说明：**

点到点移动方式移动至 P0 点，连续轨迹(CONT)，进入 LOOP 回圈，

IF 如果 读取目前运动状态数值为 0 时，才进行点到点移动方式至 P1 点，并退出 LOOP 回圈。

■ **参考相关指令：**

$C[]。

2.6.10 @EXT_TCP_START 启动外部 TCP 点位轨迹指令

2.6.11 @EXT_TCP_END 停止外部 TCP 点位轨迹指令

■ **指令说明：**

当机器人拿工件对外部工具点执行 LIN 或 CIRC 运动时，例如抛光作业，若 需改变姿态，则需要使用这指令。

若未使用 EXT_TCP，仅在起点和终点时，工件与工具才会有接触。 如下图所示

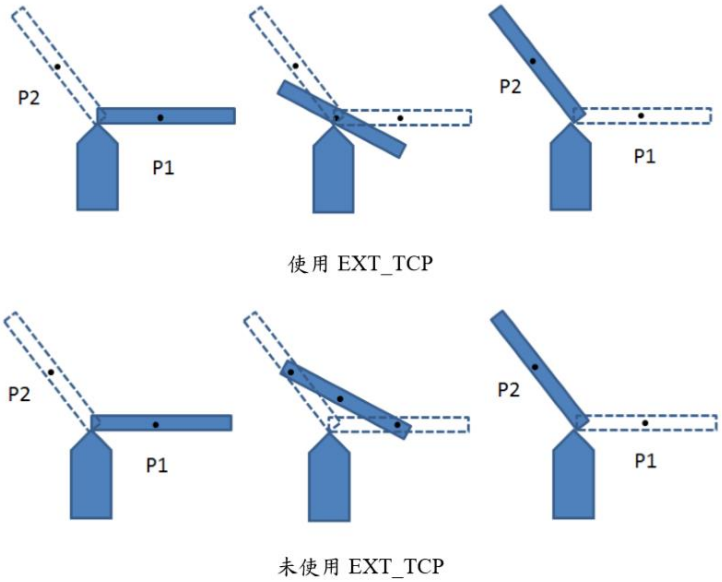


图 26 EXT_TCP 示意图(使用后为上图，未使用为下图)

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	不预读 FINE

⇨ 于外部工具点(下方图面红色尖点位置)，教导一个 BASE 座标系，BASE 座标系原点位于工具加工点上。

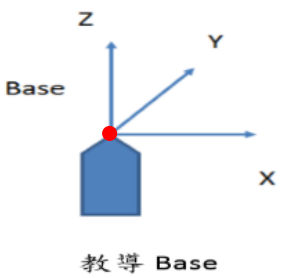


图 27 教导 BASE 座标系原点图

⇨ 教导起点 P1 和终点 P2(红点处)。



图 28 教导起点 P1(左)图教导终点 P2(右)图

■ **指令格式：**

```
EXT_TCP_START
    [运动指令]
EXT_TCP_END
```

■ **格式/参数说明：**

⇨ 在 EXT_TCP_START 和 EXT_TCP_END 之间的运动指令，会以外部工具点参考基座座标点的方式运动。

⇨ [运动指令] 为执行陈述动作，逻辑程式、移动指令或讯号指令之组合。

■ **范例：**

```
EXT_TCP_START
LIN P1 FINE=1 Vel=1000mm/s Acc=100% TOOL[0] BASE[1]
LIN P2 FINE=1 Vel=1000mm/s Acc=100% TOOL[0] BASE[1]
EXT_TCP_END
```

■ **范例说明：**

机器人走到 P1 后，沿着工件的直线进行(抛光)加工，运动到 P2，过程中，工件与外部工具点保持接触。

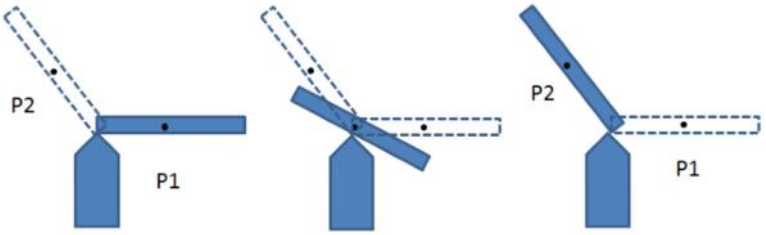


图 29 使用 EXT_TCP 示意图

■ **参考相关指令：**

LIN、CIRC

2.6.12 @TL_SET 设定扭力极限监控参数指令

■ **指令说明：**

分别可设定各轴的扭力极限值，在开启扭力监控功能下，该轴扭力达所设定的极限值，停止机器人运动，设定「-1」表示该轴不启用扭力极限监控，各轴预设为不开启此功能，由使用者进行设定。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.19 以上	Caterpillar1.1.2 以上	预读 CONT

■ **指令格式：**

TL_SET([第 1 轴数值], [第 2 轴数值], [第 3 轴数值], [第 4 轴数值], [第 5 轴数值], [第 6 轴数值], [第 E1 轴数值], [第 E2 轴数值], [第 E3 轴数值])

■ **格式/参数说明：**

⇨ [第 1 轴数值]~[第 6 轴数值] 为手臂第 1 轴到第 6 轴的扭力极限数值，整数(INT)形式，单位为%，可输入范围为 1~300 ，若其中一轴输入数值「-1」代表不启用该轴扭力极限监控；各轴预设监控功能未开启(-1)。

⇨ [第 E1 轴数值]~[第 E3 轴数值] 为外部轴第 1 轴到第 3 轴的扭力极限数值，整数(INT)形式，单位为%，可输入范围为 1~300 ，若其中一轴输入数值「-1」代表不启用该轴扭力极限监控；各轴预设监控功能未开启(-1)。

■ **范例：**

```
TL_SET(50, -1, 20, 30, -1, -1, 100, -1, -1)
TL_START
PTP P0
PTP P1
TL_END
```

■ **范例说明：**

设定扭力极限监控功能，第 1 轴设定为 50%，第 3 轴设定为 20%，第 4 轴设定为 30%，第 E1 轴设定为 100%，第 2、5、6、E2、E3 轴设定为-1 表示不启用监控。
启动扭力极限监控功能。
点到点方式移动至 P0，点到点方式移动至 P1。
结束扭力极限监控功能。

■ **参考相关指令：**

TL_START、TL_END。

2.6.13 @SERVO_ON 全轴激磁指令

■ **指令说明：**

全轴激磁指令，将会使手臂的马达做全轴(第一与第二轴)变为激磁状态，让手臂可进行操作状态，目前仅适用史卡拉机器手臂(SCARA)，六轴机器手臂安全性问题暂不开放。

注意：若靠近机器手臂操作，请务必一手拿持紧急停止开关，可随时掌握手臂的状态。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.23 以上	Caterpillar1.1.6 以上	预读 CONT

⇨ 仅适用史卡拉机器手臂(SCARA) 第一与第二轴马达。

■ **指令格式：**

```
SERVO_ON
```

■ **格式/参数说明：**

⇨ 使用程式方式使马达激磁状态

■ **范例：**

```
SERVO_ON
```

■ **范例说明：**

开启 SCARA 系列第一轴与第二轴马达激磁。

■ **参考相关指令：**

、SERVO_OFF、SERVO_ON(#)、SERVO_OFF(#)。

2.6.14 @SERVO_OFF 全轴解激磁指令

■ **指令说明：**

全轴解激磁指令，将会使手臂的马达做全轴(第一与第二轴)变为解激磁状态，让手臂可进行操作状态，目前仅适用史卡拉机器手臂(SCARA)，六轴机器手臂安全性问题暂不开放，可让使用者进入手动方式推动手臂模式做到校正点位或移动期望位置，再将马达做激磁，可另类调整的方式。

注意：若靠近机器手臂操作，请务必一手拿持紧急停止开关，可随时掌握手臂的状态。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.23 以上	Caterpillar1.1.6 以上	预读 CONT

⇨ 仅适用史卡拉机器手臂(SCARA) 第一与第二轴马达。

■ **指令格式：**

```
SERVO_OFF
```

■ **格式/参数说明：**

⇨ 使用程式方式使马达解激磁状态

■ **范例：**

```
SERVO_OFF
```

■ **范例说明：**

解激磁第一轴与第二轴马达 (SCARA 系列)。

■ **参考相关指令：**

SERVO_ON、SERVO_ON(#)、SERVO_OFF(#)。

2.6.15 @SERVO_ON(#) 单轴激磁指令

■ **指令说明：**

全轴激磁指令，将会使手臂的马达做单轴(第一或第二轴)变为激磁状态，让手臂可进行操作状态，目前仅适用史卡拉机器手臂(SCARA)，六轴机器手臂安全性问题暂不开放。

注意：若靠近机器手臂操作，请务必一手拿持紧急停止开关，可随时掌握手臂的状态。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.23 以上	Caterpillar1.1.6 以上	预读 CONT

⇨ 仅适用史卡拉机器手臂(SCARA) 第一或第二轴马达。

■ **指令格式：**

```
SERVO_ON
```

■ **格式/参数说明：**

⇨ 使用程式方式使马达激磁状态

■ **范例：**

```
SERVO_ON(2)
```

■ **范例说明：**

开启第二轴马达激磁功能 (SCARA 系列)。

■ **参考相关指令：**

SERVO_ON、SERVO_OFF、SERVO_OFF(#)。

2.6.16 @SERVO_OFF(#) 单轴解激磁指令

■ **指令说明：**

全轴解激磁指令，将会使手臂的马达做单轴(第一或第二轴)变为解激磁状态，让手臂可进行操作状态，目前仅适用史卡拉机器手臂(SCARA)，六轴机器手臂安全性问题暂不开放，可让使用者进入手动方式推动手臂模式做到校正点位或移动期望位置，再将马达做激磁，可另类调整的方式。

注意：若靠近机器手臂操作，请务必一手拿持紧急停止开关，可随时掌握手臂的状态。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.23 以上	Caterpillar1.1.6 以上	预读 CONT

⇨ 仅适用史卡拉机器手臂(SCARA) 第一或第二轴马达。

■ **指令格式：**

```
SERVO_OFF(#)
```

■ **格式/参数说明：**

⇨ 使用程式方式使马达解激磁状态

⇨ # 表示为欲解激磁的马达轴别，可输入范围 1~2。

■ **范例：**

```
SERVO_OFF(1)
```

■ **范例说明：**

解激磁第一轴马达(SCARA 系列)。

■ **参考相关指令：**

SERVO_ON、SERVO_OFF、SERVO_ON(#)。

2.7 代码 G、数学运算指令

此章节针对于代码 G 的输送带指令做各别说明。

2.7.1@+ 数学运算相加指令

■ **指令说明：**

用于基本数学运算相加(+)功能，数值、变数、计时暂存器、计时暂存器或 PR 位置暂存器的同性质数学运算使用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

[自定义变数名称] = [自定义变数名称] + [数值]
[暂存器] = [暂存器] + [数值]

■ **格式/参数说明：**

⇨ [自定义变数名称] 为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，例如： X1 。

⇨ [暂存器] 为各种不同的暂存器用于数学运算，例如：计时暂存器$T[]、计时暂存器$C[]或 PR 位置暂存器$PR[]。

■ **范例 1：**

```
INT X1 = 10
X1 = X1 + 1
```

■ **范例 1 说明：**

宣告整数变数 X1，初始值为 10。

将 X1 变数相加数值 1，再传回至 X1 变数中。 **结果：**10+1 = 11。

■ **范例 2：**

```
$C[1] = 0
$C[1] = $C[1] + 1
```

■ **范例 2 说明：**

第一个计数暂存器初始值为 0。

将第一个计数暂存器($C[1])相加数值 1，再传回至第一个计数暂存器中。**结果：**$C[1]=0+1 = 1。

■ **参考相关指令：**

INT、REAL、$C[]、$T[]、$PR[]。

2.7.2@- 数学运算相减指令

■ **指令说明：**

用于基本数学运算相减(-)功能，数值、变数、计时暂存器、计时暂存器或 PR 位置暂存器的同性质数学运算使用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

[自定义变数名称] = [自定义变数名称] - [数值]
[暂存器] = [暂存器] - [数值]

■ **格式/参数说明：**

⇨ [自定义变数名称] 为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，例如： X1 。

⇨ [暂存器] 为各种不同的暂存器用于数学运算，例如：计时暂存器$T[]、计时暂存器$C[]或 PR 位置暂存器$PR[]。

■ **范例 1：**

```
INT X1 = 10
X1 = X1 - 1
```

■ **范例 1 说明：**

宣告整数变数 X1，初始值为 10。

将 X1 变数相减数值 1，再传回至 X1 变数中。 **结果：**10-1 = 9。

■ **范例 2：**

```
$C[1] = 0
$C[1] = $C[1] - 1
```

■ **范例 2 说明：**

第一个计数暂存器初始值为 0。

将第一个计数暂存器($C[1])相减数值 1，再传回至第一个计数暂存器中。**结果：**$C[1]=0-1 = -1。

■ **参考相关指令：**

INT、REAL、$C[]、$T[]、$PR[]。

2.7.3@* 数学运算相乘指令

■ **指令说明：**

用于基本数学运算相乘(*)功能，数值、变数、计时暂存器、计时暂存器或 PR 位置暂存器的同性质数学运算使用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

[自定义变数名称] = [自定义变数名称] * [数值]
[暂存器] = [暂存器] * [数值]

■ **格式/参数说明：**

⇨ [自定义变数名称] 为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，例如： X1 。

⇨ [暂存器] 为各种不同的暂存器用于数学运算，例如：计时暂存器$T[]、计时暂存器$C[]或 PR 位置暂存器$PR[]。

■ **范例 1：**

```
INT X1 = 10
X1 = X1 * 2
```

■ **范例 1 说明：**

宣告整数变数 X1，初始值为 10。

将 X1 变数相乘数值 2，再传回至 X1 变数中。 **结果：**10*2 = 20。

■ **范例 2：**

```
$C[1] = 4
$C[1] = $C[1] * 2
```

■ **范例 2 说明：**

第一个计数暂存器初始值为 4。

将第一个计数暂存器($C[1])相乘数值 2，再传回至第一个计数暂存器中。**结果：**$C[1]=4*2 = 8。

■ **参考相关指令：**

INT、REAL、$C[]、$T[]、$PR[]。

2.7.4@/ 数学运算相除指令

■ **指令说明：**

用于基本数学运算相除(/)功能，数值、变数、计时暂存器、计时暂存器或 PR 位置暂存器的同性质数学运算使用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

[自定义变数名称] = [自定义变数名称] / [数值]
[暂存器] = [暂存器] / [数值]

■ **格式/参数说明：**

⇨ [自定义变数名称] 为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，例如： X1 。

⇨ [暂存器] 为各种不同的暂存器用于数学运算，例如：计时暂存器$T[]、计时暂存器$C[]或 PR 位置暂存器$PR[]。

■ **范例 1：**

```
INT X1 = 10
X1 = X1 / 2
```

■ **范例 1 说明：**

宣告整数变数 X1，初始值为 10。

将 X1 变数相除数值 2，再传回至 X1 变数中。 **结果：**10/2 = 5。

■ **范例 2：**

```
$C[1] = 4
$C[1] = $C[1] / 2
```

■ **范例 2 说明：**

第一个计数暂存器初始值为 4。

将第一个计数暂存器($C[1])相除数值 2，再传回至第一个计数暂存器中。**结果：**$C[1]=4/2 = 2。

■ **参考相关指令：**

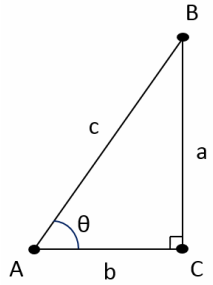
INT、REAL、$C[]、$T[]、$PR[]。

2.7.5@sin 数学运算的正弦函数指令

■ **指令说明：**

数学运算的正弦函数(sine)指令，用于程式中作数学式运算使用，若得知三角形角度值欲求得边长的比值数值可使用此指令。

$$\sin\theta = \frac{a}{c}$$



■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
[自定义变数名称] = SIN(#)
```

■ **格式/参数说明：**

⇨ [自定义变数名称] 为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。

⇨ # 为三角形的角度数值，可为浮点数值，由使用者输入数值，可输入区间范围介于 0 至 180 度之间。

■ **范例：**

```
REAL dis_ratioA
dis_ratioA =SIN(30)
```

■ **范例说明：**

宣告浮点数值名称 dis_ratioA，dis_ratioA 内数值为 30 度的正弦函数之值，计算数值结果为 0.5。

■ **参考相关指令：**

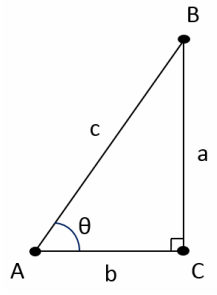
ACOS、ASIN、ATAN、ATAN2、ACOS、COS、TAN

2.7.6@cos 数学运算的余弦函数指令

■ **指令说明：**

数学运算的余弦函数(cosine)指令，用于程式中作数学式运算使用，若得知三角形角度值欲求得边长的比值数值可使用此指令。

$$\cos\theta = \frac{b}{c}$$



■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

[自定义变数名称] = COS(#)

■ **格式/参数说明：**

⇨ [自定义变数名称] 为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。

⇨ # 为三角形的角度数值，可为浮点数值，由使用者输入数值，可输入区间范围介于 0 至 180 度之间。

■ **范例：**

```
REAL dis_ratioA
dis_ratioA =COS(60)
```

■ **范例说明：**

宣告浮点数值名称 dis_ratioA，dis_ratioA 内数值为 60 度的余弦函数之值，计算数值结果为 0.5。

■ **参考相关指令：**

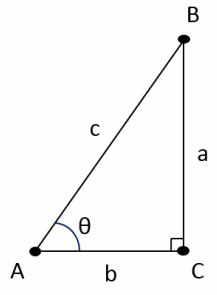
ACOS、ASIN、ATAN、ATAN2、ACOS、SIN、TAN。

2.7.7@tan 数学运算的正切函数指令

■ **指令说明：**

数学运算的正切函数(Tangent)指令，用于程式中作数学式运算使用，若得知三角形角度值欲求得边长的比值数值可使用此指令。

$$\tan\theta = \frac{a}{b}$$



■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
[自定义变数名称] = TAN(#)
```

■ **格式/参数说明：**

⇨ [自定义变数名称] 为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。

⇨ # 为三角形的角度数值，可为浮点数值，由使用者输入数值，可输入区间范围介于 0 至 180 度之间。

■ **范例：**

```
REAL dis_ratioA
dis_ratioA =TAN(45)
```

■ **范例说明：**

宣告浮点数值名称 dis_ratioA，dis_ratioA 内数值为 45 度的正切函数之值，计算数值结果为 1。

■ **参考相关指令：**

ACOS、ASIN、ATAN、ATAN2、ACOS、COS、SIN。

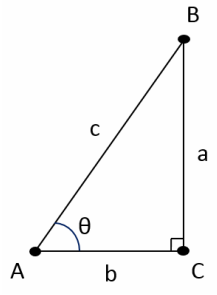
2.7.8@ASIN 数学运算的反正弦函数指令

■ **指令说明：**

数学运算的反正弦函数(arcsine,$\sin^{-1}$)指令，用于程式中作数学式运算使用，若得知三角形边长的比值数值欲求得角度值可使用此指令。

$$\sin\theta = \frac{a}{c} = x$$

$$\theta = \sin^{-1}x$$



■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

[自定义变数名称] = ASIN(#)

■ **格式/参数说明：**

⇨ [自定义变数名称] 为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。

⇨ # 为三角形边长的比值数值，可为浮点数值，由使用者输入数值，可输入区间范围介于 -1 至 1 之间，计算出的角度值为-90 至 90 度之间($-\frac{\pi}{2}\sim\frac{\pi}{2}$)。

■ **范例：**

```
REAL Degree_A
Degree_A =ASIN(0.5)
```

■ **范例说明：**

宣告浮点数值名称 Degree_A，Degree_A 内数值为 0.5 三角形边长的比值的反正弦函数之值，计算数值结果为 30 度。

■ **参考相关指令：**

ACOS、SIN、ATAN、ATAN2、ACOS、COS、TAN。

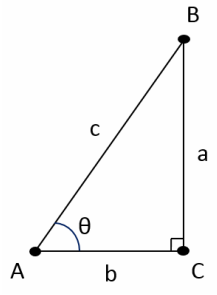
2.7.9@ACOS 数学运算的反余弦函数指令

■ **指令说明：**

数学运算的反余弦函数(arccosine,$\cos^{-1}$)指令，用于程式中作数学式运算使用，若得知三角形边长的比值数值欲求得角度值可使用此指令。

$$\cos\theta = \frac{b}{c} = x$$

$$\theta = \cos^{-1}x$$



■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

[自定义变数名称] = ACOS(#)

■ **格式/参数说明：**

⇨ [自定义变数名称] 为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。

⇨ # 为三角形边长的比值数值，可为浮点数值，由使用者输入数值，可输入区间范围介于 -1 至 1 之间，计算出的角度值为 0 至 180 度之间(0~π)。

■ **范例：**

```
REAL Degree_A
Degree_A =ACOS(0.5)
```

■ **范例说明：**

宣告浮点数值名称 Degree_A，Degree_A 内数值为 0.5 三角形边长的比值的反正弦函数之值，计算数值结果为 60 度。

■ **参考相关指令：**

ASIN、SIN、ATAN、ATAN2、ACOS、COS、TAN。

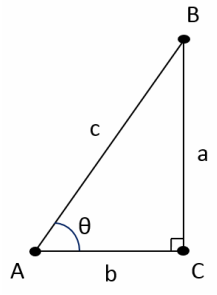
2.7.10 @ATAN 数学运算的反正切函数指令

■ **指令说明：**

数学运算的反正切函数(arctangent,$\tan^{-1}$)指令，用于程式中作数学式运算使用，若得知三角形边长的比值数值欲求得角度值可使用此指令。

$$\tan\theta = \frac{a}{b} = x$$

$$\theta = \tan^{-1}x$$



■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
[自定义变数名称] = ATAN(#)
```

■ **格式/参数说明：**

⇨ [自定义变数名称] 为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。

⇨ # 为三角形边长的比值数值，可为浮点数值，由使用者输入数值，可输入区间范围介于 0 至 1 之间，计算出的角度值为-90 至 90 度之间($-\frac{\pi}{2} \sim \frac{\pi}{2}$)。。

■ **范例：**

```
REAL Degree_A
Degree_A =ATAN(1)
```

■ **范例说明：**

宣告浮点数值名称 Degree_A，Degree_A 内数值为 1 三角形边长的比值的反正弦函数之值，计算数值结果为 45 度。

■ **参考相关指令：**

ASIN、SIN、ACOS、ATAN2、ACOS、COS、TAN。

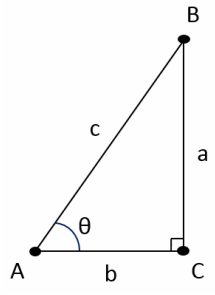
2.7.11 @ATAN2 数学运算的反正切函数变化指令

■ **指令说明：**

数学运算的反正切函数(arctangent,$\tan^{-1}$)指令的变种方式，用于程式中作数学式运算使用，若得知三角形 a、b 值(x、y 坐标)数值欲求得角度值可使用此指令。

$$\tan\theta = \frac{a}{b} = x$$

$$\theta = \tan^{-1}x = \tan^{-1}\frac{a}{b}$$



■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

[自定义变数名称] = ATAN2(y 坐标值, x 坐标值)

■ **格式/参数说明：**

⇨ [自定义变数名称] 为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。

⇨ y 坐标值& x 坐标值 为距离数值，可为浮点数值，由使用者输入数值，可输入区间范围介于 0 至 1 之间，计算出的角度值为-90 至 90 度之间($-\frac{\pi}{2}\sim\frac{\pi}{2}$)。

■ **范例：**

```
REAL dis_X, dis_Y, Degree_A
dis_X = 5
dis_Y = 5
Degree_A =ATAN2(dis_Y, dis_X)
```

■ **范例说明：**

宣告浮点数值名称dis_X, dis_Y,Degree_A，dis_X 与dis_X 边长距离为5，Degree_A内数值为dis_Y与dis_X边长的比值的反正弦函数之值，计算数值结果为45度。

■ **参考相关指令：**

ASIN、SIN、ACOS、ATAN、ACOS、COS、TAN。

2.7.12 @FWD 顺向机器人运动学转换指令

■ **指令说明：**

由关节座标系角度透过顺向机器人运动学转换为 XYZ 直角座标系(笛卡尔座标系)，作位置切换运算。在使用者仅知道点的各轴角度，需求得该点在当前的座标下末端效果器刀具中心点(TCP)的位置与姿态座标值，可使用此指令转换求得资讯，并应用作座标系判定。

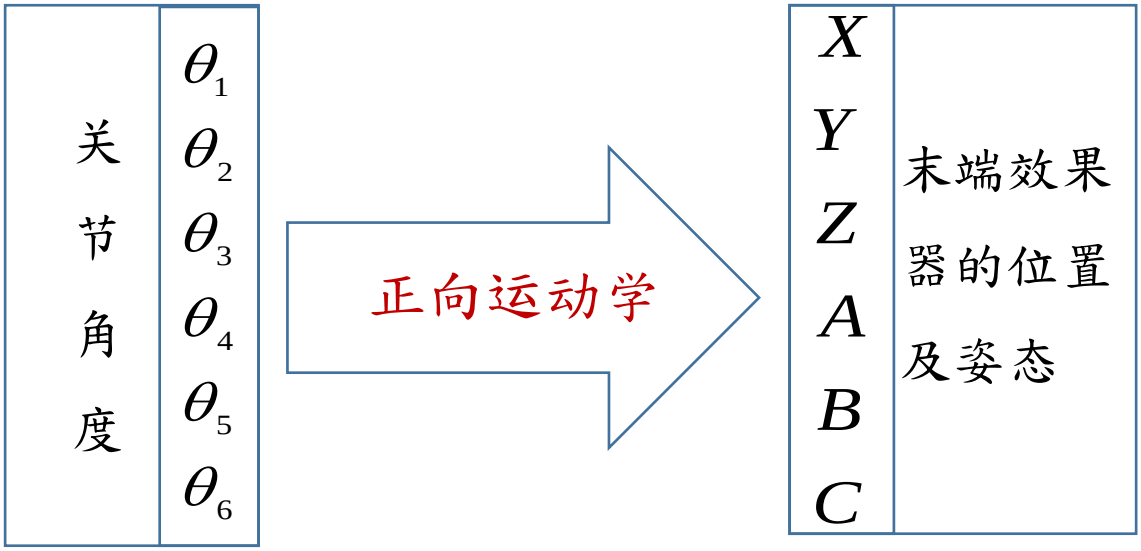


图 30 正向运动学示意图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

[自定义点位变数] = FWD([关节座标系角度点位/变数])

■ **格式/参数说明：**

⇨ [自定义点位变数] 为使用者自定义位置座标变数，设定参数内容直角/笛卡尔座标 X、Y、Z、A、B、C 轴位置座标变数。SCARA-LU 与 DELTA 系列无 AB 轴。需先透过 E6POINT、E6POS、E6AXIS 等指令先作点变数宣告。

⇨ [关节座标系角度点位/变数] 为代有关节座标系 JOINT 角度的点位或变数，具备 A1、A2、A3、A4、A5、A6 角度值资讯，LU 系列无 A5、A6 轴资讯。

■ **范例：**

```
E6AXIS P1_AXIS = {A1 0, A3 0, A4 0, A5 -90, A6 0}
E6POS P1_POS
P1_POS = FWD(P1_AXIS)
```

■ **范例说明：**

宣告 P1_AXIS 关节座标系角度变数，定义第 1 轴为 0 度、第 2 轴为 0 度、第 3 轴为 0 度、第 4 轴为 0 度、第 5 轴为-90 度、第 6 轴为 0 度。

宣告 P1_POS 直角座标系角度变数。

将 P1_AXIS 变数透过顺向机器人运动学转换为 XYZ 直角座标系，并存到 P1_POS 变数，P1_POS 为 P1_AXIS 的直角坐标系数值。(若为 RA605-710-GC 机型，P1_POS 为(0,368,293.5,-180,0,90)；若为 RS405-500-200-LU 机型，P1_POS 为(0,500,0,0)，每个机型计算出来的结果不同)。

■ **参考相关指令：**

E6POINT、E6POS、E6AXIS。

2.7.13 @% 求余数指令

■ **指令说明：**

数学运算求余数指令(remainder)，用于程式中作数学式运算使用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.10 以上	预读 CONT

■ **指令格式：**

[自定义变数] = [被除数值] % [除数值]
[计数暂存器] = [被除数值] % [除数值]

■ **格式/参数说明：**

⇨ [自定义变数名称] 为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。

⇨ [被除数值] 与[除数值] 为自定义数值或变数，除数不得为 0。

⇨ [计数暂存器] 为计数/显示使用的暂存器，显示在暂存器(Counter)画面中，整数型态，例如 $C[1] 为第一个计数暂存器。

⇨ 例如：13 除以 10，[被除数值]为 13，[除数值]为 10，商为 1，余数为 3 (13=1×10+3)。

■ **范例：**

```
INT remainder_1
remainder_1 = 13%10
$C[1] = remainder_1
```

■ **范例说明：**

宣告整数变数 remainder_1。
计算 13 除以 10 的余数，并将余数存放至 remainder_1 变数内。
使用第一个计数暂存器将 remainder_1 变数内值作显示。

■ **参考相关指令：**

INT、$C[]。

2.8 代码 H、通讯功能指令

此章节说明手臂与外部的通讯指令，通讯指令可用于串列通讯 RS232 与网路 EtherNet(TCP/IP)通讯架构，可应用于跟外部设备如可逻辑语言控制 PLC 或工业电脑之间的通讯交握，终端应用可搭配视觉或人机介面 HMI 沟通使用。(Modbus 通讯不在此指令说明)

2.8.1@COPEN 开启通讯指令

■ **指令说明：**

开启串列通讯 RS232 或网路 EtherNet(TCP/IP)通讯，通常与程式开头初始化时就开启网路通讯以作后续资料交握，支援单一连线与多重连线通讯方式。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 需先至串列通讯 RS232 或网路 EtherNet(TCP/IP)通讯的参数页面作设定，RS232 需设串列鲍率设定；EtherNet 需选择伺服端 Server 或客户端 Client 连线架构并作 IP 设定。

■ **指令格式：**

✧ 格式 1： (单一连线)

```
COPEN( [通讯方式] , [自定义变数名称])
```

✧ 格式 2： (多重连线)

```
[自定义变数名称] = COPEN( [通讯方式] ,[连线 IP],[通讯埠] )
```

■ **格式/参数说明：**

⇨ [通讯方式] 为选择手臂对外通讯方式，串列通讯 RS232 或网路 EtherNet(TCP/IP)通讯，若为 RS232 时需填写代号「SER」；若为 EtherNet 时需填写代号「ETH」。

⇨ [自定义变数名称]为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，在 COPEN 指令前，需先宣告[自定义变数名称]，通常为整数 INT 型态，以利使用变数于开通指令内；若开通不成功，[自定义变数名称]之数值将为-1。

⇨ 使用多重连线连线通道若 IP 为 192.168.0.5，port 为 5000 则写法如下:

⇨ [连线 IP] 为通讯连线 IP，需输入四个值，中间用逗号「,」隔开，范例：192, 168, 0, 5。

⇨ [通讯埠] 为需要连线设定的通讯埠(PORT)，以辨别指定的频道，例如：PORT: 5000。

■ **范例 1： (单一连线)**

```
INT HANDLE
COPEN(SER,HANDLE)
```

■ **范例 1 说明：**

宣告整数变数 HANDLE。

开启串列通讯 RS232 通讯。

(注:若 SER 换成 ETH，则开启网路 EtherNet(TCP/IP)通讯。)

■ **范例 2： (多重连线)**

```
INT HANDLE1,HANDLE2
HANDLE1 = COPEN(ETH, 192, 168, 0, 5, 5000)
HANDLE2 = COPEN(ETH, 192, 168, 0, 25, 5000)
```

■ **范例 2 说明：**

宣告整数变数 HANDLE1、HANDLE2。

开启网路 EtherNet(TCP/IP)通讯 1，IP192.168.0.5，Port:5000；开启网路 EtherNet(TCP/IP)通讯 2，IP192.168.0.25，Port:5000。

■ **参考相关指令：**

INT、CCLOSE、CCLEAR、CREAD、CWRITE、CINQUIRE。

2.8.2@CCLOSE 关闭通讯指令

■ **指令说明：**

关闭串列通讯 RS232 或网路 EtherNet(TCP/IP)通讯指令。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 需先至串列通讯 RS232 或网路 EtherNet(TCP/IP)通讯的参数页面作设定。

⇨ 在 CCLOSE 指令前，需作执行 COPEN 指令。

■ **指令格式：**

```
CCLOSE( [通讯变数名称] )
```

■ **格式/参数说明：**

⇨ [通讯变数名称]为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，在 CCLOSE 指令前，需作执行 COPEN 指令，且对应相对的 [通讯变数名称]，才可作通讯关闭。

■ **范例：**

```
INT HANDLE
COPEN( ETH ,HANDLE)
WAIT SEC 3
IF $DI[5] == TRUE THEN
    CCLOSE (HANDLE)
ENDIF
```

■ **范例说明：**

宣告整数变数 HANDLE。

开启，则开启网路 EtherNet(TCP/IP)通讯。

等待 3 秒后，判定如果第五个数位输入讯号($DI[5])为真(TRUE)，则关闭网路 EtherNet(TCP/IP)通讯。

■ **参考相关指令：**

INT、COPEN、CCLEAR、CREAD、CWRITE、CINQUIRE、IF。

2.8.3@CCLEAR 清除通讯缓冲区资料指令

■ **指令说明：**

清除手臂内通讯暂存缓冲区资料指令，在通讯期间，会发生不断输入资料给手臂至暂存缓冲区，手臂会依接收顺序作业，透过 CCLEAR 指令可将已接收到尚未执行的暂存缓冲区资料作清除动作，可确保下一笔资料为最新资讯。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 需先至串列通讯 RS232 或网路 EtherNet(TCP/IP)通讯的参数页面作设定。

⇨ 在 CCLEAR 指令前，需作执行 COPEN 指令。

■ **指令格式：**

```
CCLEAR( [通讯变数名称] )
```

■ **格式/参数说明：**

⇨ [通讯变数名称]为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，在 CCLEAR 指令前，需作执行 COPEN 指令，且对应相对的 [通讯变数名称]，才可作暂存缓冲区资料清除动作。

■ **范例：**

```
INT HANDLE
REAL receive_data
COPEN( ETH ,HANDLE)
CREAD (HANDLE,receive_data)
IF $DI[5] == TRUE THEN
    CCLEAR (HANDLE)
ENDIF
```

■ **范例说明：**

宣告整数变数 HANDLE，宣告浮点数变数 receive_data(通讯接收的资料)。

开启，则开启网路 EtherNet(TCP/IP)通讯。

等待读取由外部设备通讯的数值至 receive_data 变数内，

判定如果第五个数位输入讯号($DI[5])为真(TRUE)，则清除手臂内通讯暂存缓冲区资料(可能传很多的资料，但没传输至 receive_data 变数暂存器内资料全部清除)。

■ **参考相关指令：**

INT、REAL、COPEN、CCLOSE、CREAD、CWRITE、CINQUIRE。

2.8.4@CREAD 接收资料指令

■ **指令说明：**

手臂通讯方式接收外部设备的输入资料指令；需由使用者决定后续在程式上应用的数值，将由上位设备与手臂使用者协调传输/接收资料代表意义。

在执行 CREAD 指令至程式时，会需由上位设备传输资料，若无传输资料成功，手臂会一直停在 CREAD 指令，若上位传输值数量不符合时，会以 0 为值或忽略不计，不会产生错误讯息。例如传输 2 个数值给手臂接收，手臂需要接收 3 个数值资讯，则手臂第 3 个数值为 0。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 需先至串列通讯 RS232 或网路 EtherNet(TCP/IP)通讯的参数页面作设定。

⇨ 在 CREAD 指令前，需作执行 COPEN 指令。

■ **指令格式：**

CREAD([通讯变数名称] , [输入资料接收变数 1], [输入资料接收变数 2], …)

■ **格式/参数说明：**

⇨ [通讯变数名称]为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，在 CREAD 指令前，需作执行 COPEN 指令，且对应相对的 [通讯变数名称]，才可作接收资料动作。

⇨ [输入资料接收变数] 为由外部设备通讯至手臂的输入资料接收变数，需使用者先自定义的变数名称。

⇨ 宣告变数需要使用浮点数(REAL)型态。

⇨ 可接收多个[输入资料接收变数]，每个中间以逗号「,」作区隔(最多 400 笔资料)。

⇨ [输入资料接收变数] 将决定后续在程式上应用的数值，将由上位设备传输值与使用者协调[输入资料接收变数]代表变数内数值意义。

⇨ 读取资料时会一串封包，当[输入资料接收变数]与上位传输值数量不符合时，会以 0 为值或忽略不计。

⇨ 手臂通讯接收资料开头与结尾都需有大括号符号「{}」，例如{ 100,200 }。

⇨ 在 HRSS3.3.8 或 Caterpillar1.0.6 以上版本可以通讯传输字元资讯。

■ **范例 1：**

```
INT HANDLE
REAL receive_data
COPEN( ETH ,HANDLE)
CREAD (HANDLE,receive_data)
```

■ **范例 1 说明：**

宣告整数变数 HANDLE，宣告浮点数变数 receive_data(通讯接收的资料)。
开启，则开启网路 EtherNet(TCP/IP)通讯。
等待读取由外部设备通讯的数值至 receive_data 变数内。

■ **范例 2： (透过外部设备传输视觉座标 X,Y 范例)**

例如： Camera 视觉传送座标{100,200}
手臂指令如下：

```
INT HANDLE
REAL X_data,Y_data
E6POINT OBJ_P
COPEN( ETH ,HANDLE)
PTP P1 FINE=1 Vel=40% Acc=75% TOOL[0] BASE[0]
OBJ_P = P1
LOOP
    IF HANDLE > -1 THEN
        CREAD (HANDLE, X_data,Y_data)
        OBJ_P.X = X_data
        OBJ_P.Y = Y_data
        LIN OBJ_P FINE = 1
    ENDIF
    WAIT SEC 0.3
ENDLOOP
```

■ **范例 2 说明：**

宣告整数变数 HANDLE，宣告浮点数变数 X_data,Y_data (通讯接收的外部设备传输视觉座标 X,Y 资料)。宣告位置变数 OBJ_P，开启，则开启网路 EtherNet(TCP/IP)通讯。
点到点方式移动至 P1 点，并让 OBJ_P 位置等于 P1(为了避免其他 A,B,C 等资讯不同步)
使用LOOP…ENDLOOP回圈，让接收资讯与移动动作无限的回圈。
IF如果 网路通讯使有开启(HANDLE > -1)，才执行接收资料作业。
等待读取由外部设备通讯的数值至 X_data,Y_data 变数内(接收到数值 X_data=100，Y_data=200)。
将接收到的 X_data,Y_data 变数数值取代 OBJ_P.X,OBJ_P.Y 位置值，并使用直线方式移动至新的待取料 OBJ_P 点位(即移动至 X 为 100mm，Y 为 200mm 位置)。
等待 0.3 秒后，继续下一轮接收资料动作。

■ **参考相关指令：**

INT、REAL、COPEN、CCLOSE、CCLEAR、CWRITE、CINQUIRE。

2.8.5@CWRITE 传送资料指令

■ **指令说明：**

手臂通讯方式给予外部设备的输出资料指令；将由上位设备与手臂使用者协调传输资料代表意义，以确保传输资料的数值应用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	不预读 FINE

⇨ 需先至串列通讯 RS232 或网路 EtherNet(TCP/IP)通讯的参数页面作设定。

⇨ 在 CWRITE 指令前，需作执行 COPEN 指令。

■ **指令格式：**

CREAD([通讯变数名称] , [输出资料接收变数 1], [输出资料接收变数 2], …)

■ **格式/参数说明：**

⇨ [通讯变数名称]为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，在 CWRITE 指令前，需作执行 COPEN 指令，且对应相对的 [通讯变数名称]，才可作传输资料动作。

⇨ [输出资料传输变数] 为由手臂通讯至外部设备的输出资料传输变数，需使用者先自定义的变数名称。

⇨ 宣告变数需要使用浮点数(REAL)型态。

⇨ 可传输多个[输出资料传输变数]，每个中间以逗号「,」作区隔，最多 400 笔资料。

⇨ [输出资料传输变数] 将决定后续在程式上应用的数值，将由上位设备传输值与使用者协调[输出资料传输变数]代表变数内数值意义。

⇨ 手臂通讯传出资料开头与结尾都会增加大括号符号「{}」，例如{ 100,200 }。

⇨ 在 HRSS3.3.1 或 Caterpillar1.0.4 以上版本可以通讯传输字元资讯。

■ **范例 1：**

```
INT HANDLE
REAL send_data = 10
COPEN( ETH ,HANDLE)
CWRITE (HANDLE,send_data)
```

■ **范例 1 说明：**

宣告整数变数 HANDLE，宣告浮点数变数 send_data(通讯传输的资料)数值为 10。
开启，则开启网路 EtherNet(TCP/IP)通讯。
通讯传输 send_data 变数内数值 10 给由外部设备。(外部将接收到数值{10} ，手臂通讯传出资料开头与结尾都会增加大括号符号「{}」)

■ **范例 2：**

手臂指令如下：

```
INT HANDLE
REAL long_size = 30
REAL width_size = 20
REAL tall_size = 10
COPEN( ETH ,HANDLE)
CWRITE (HANDLE,long_size,width_size,tall_size)
```

例如： 外部设备工业电脑接收座标{30,20,10}

■ **范例 2 说明：**

宣告整数变数 HANDLE，宣告浮点数变数 long_size 数值为 30，width_size 数值为 20，tall_size 数值为 10。
开启，则开启网路 EtherNet(TCP/IP)通讯。
通讯传输变数内数值 30,20,10 给由外部设备。(外部将接收到数值{30,20,10}，手臂通讯传出资料开头与结尾都会增加大括号符号「{}」)

■ **参考相关指令：**

INT、REAL、COPEN、CCLOSE、CCLEAR、CREAD、CINQUIRE。

2.8.6@CINQUIRE 取得资料笔数指令

■ **指令说明：**

读取通讯传输缓冲区接收到的资料数量指令，用于确认外部设备传输数量的正确性。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 需先至串列通讯 RS232 或网路 EtherNet(TCP/IP)通讯的参数页面作设定。

⇨ 在 CINQUIRE 指令前，需作执行 COPEN 指令。

■ **指令格式：**

CINQUIRE([通讯变数名称] , [自定义变数名称])

■ **格式/参数说明：**

⇨ [通讯变数名称]为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，在 CINQUIRE 指令前，需作执行 COPEN 指令，且对应相对的 [通讯变数名称]，才可作传输资料动作。

⇨ [自定义变数名称]为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，用于读取通讯传输接收到的资料数量

■ **范例：**

```
INT HANDLE,data_num
REAL receive_data,send_data
send_data=0
COPEN ( ETH , HANDLE)
LOOP
    IF HANDLE > -1 THEN
        CINQUIRE(HANDLE,NUM)
        If data_num > 0 THEN
          CREAD (HANDLE,receive_data)
        ENDIF
        CCLEAR (HANDLE)
        send_data = send_data + 1
       CWRITE (HANDLE,send_data)
    ENDIF
    WAIT SEC 0.3
ENDLOOP
```

■ **范例说明：**

宣告整数变数HANDLE,data_num，宣告浮点数变数receive_data,send_data，send_data变数初始直为0。开启，则开启网路EtherNet(TCP/IP)通讯。

使用LOOP…ENDLOOP回圈，让接收资讯与移动动作无限的回圈。

IF如果 网路通讯使有开启(HANDLE > -1)，才执行接收资料作业。

读取通讯传输缓冲区接收到的资料数量。IF 如果接收 data_num 资料数量大于 1，才执行 CREAD 接收资料。

等待读取由外部设备通讯的数值至 receive_data 变数内。

CCLEAR 清除手臂内通讯暂存缓冲区资料。

每执行一次 send_data 数量加 1，并将 send_data 数值传送给外部设备通讯。

等待 0.3 秒后，继续下一轮接收资料动作。

■ **参考相关指令：**

INT、REAL、COPEN、CCLOSE、CCLEAR、CREAD、CWRITE。

2.8.7@CFORMAT 宣告自定义通讯格式指令

■ **指令说明：**

新增自定义通讯格式指令，可透过 Hex ASCII 码方式(参考 0)，自行定义开头、结尾及分割字元之符号，应用在不同通讯格式切换符号使用，例如：视觉系统配合开头码非预设”{”格式，可以自行作宣告定义，可补足通讯介面选项不够的问题。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.16 以上	Caterpillar1.1.0 以上	预读 CONT

⇨ 在 CFORMAT 指令前，需作执行 COPEN 指令。

⇨ 在网路连接设定(Network Config)页面不能勾选取消格式(non format)功能。

■ **指令格式：**

CFORMAT ([通讯变数名称] , [自定义起始符号] , [自定义分隔符号] , [自定义结束符号])

■ **格式/参数说明：**

⇨ [通讯变数名称] 为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，在 CCLOSE 指令前，需作执行 COPEN 指令，且对应相对的 [通讯变数名称]，才可作通讯关闭。

⇨ [自定义起始符号] 为用在通讯的起始封包辨别符号，类别为字串(STRING)类型，原始预设符号为”{” (0x7B)，和结束符号至少其一需要设定，不可以都不使用 。。

⇨ [自定义分隔符号] 为用在通讯的每字串间辨别符号，类别为字串(STRING)类型，原始预设符号为”,” (0x2C)，分隔符号不可以不设定。

⇨ [自定义结束符号] 为用在通讯的结束封包辨别符号，类别为字串(STRING)类型，原始预设符号为”}” (0x7D) ，和起始符号至少其一需要设定，不可以都不使用 。。

⇨ 若没有使用起始符号或结束符号则以「””」。

⇨ 定义符号的字串起始必须以”0x”开头，例如：常见结尾字元<CR><LF>为”0x0D0A” ，且自订符号最多使用两个字元。

⇨ CFORMAT 会将通讯格式同时套用在 CREAD 和 CWRITE 上。

⇨ CFORMAT 自定义通讯格式仅在程式执行间有效，程式结束后会回复介面(UI)上的通讯格式。

■ **范例 1：**

```
INT HANDLE
REAL receive_data
COPEN(ETH, HANDLE)
CFORMAT(HANDLE,"0x02","0x2C","0x0D0A")
CREAD (HANDLE,receive_data)
```

■ **范例 1 说明：**

宣告整数变数 HANDLE。

宣告浮点数变数 receive_data(通讯接收的资料)。

开启网路 EtherNet(TCP/IP)通讯。

自行宣告通讯格式起头符号为 "0x02"，代表意义为<STX>；分隔符号为 "0x2C"，代表意义为”,”(逗号)；结尾符号为 "0x0D0A"，代表意义为<CR><LF>。

程式执行时手臂只接收其他通讯设备传送<STX>开头符号，<CR><LF>结尾符号，”,” 分隔符号的格式。

等待读取由外部设备通讯的数值至 receive_data 变数内。

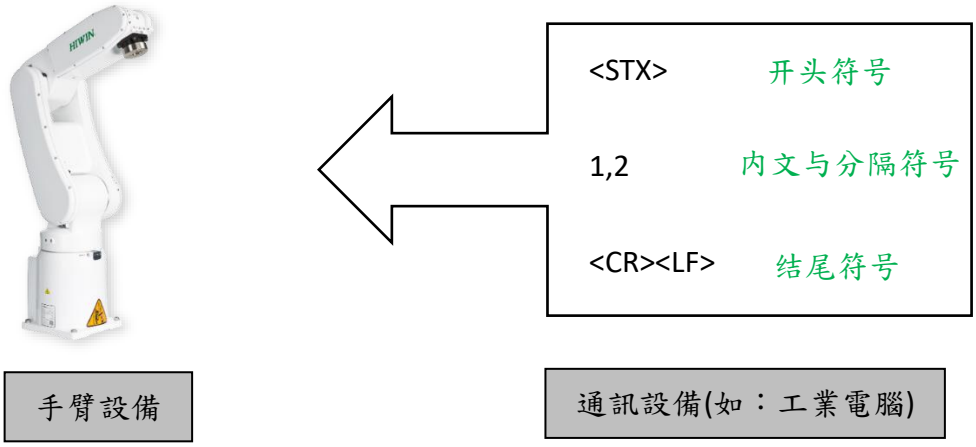


图31 自定义通讯格式通讯传输示意图

■ **范例 2：**

```
INT HANDLE
STRING receive_data1,send_data1,send_data2
COPEN(ETH, HANDLE)
CFORMAT(HANDLE,"0x00","0x01","0x0D0A")
send_data1 = "Robot1"
send_data2 = "Robot2"
CWRITE (HANDLE,send_data1,send_data2)
CREAD(HANDLE, receive_data1)
```

■ **范例 2 说明：**

宣告整数变数 HANDLE。

宣告字串变数receive_data1与send_data1(通讯接收的资料)。

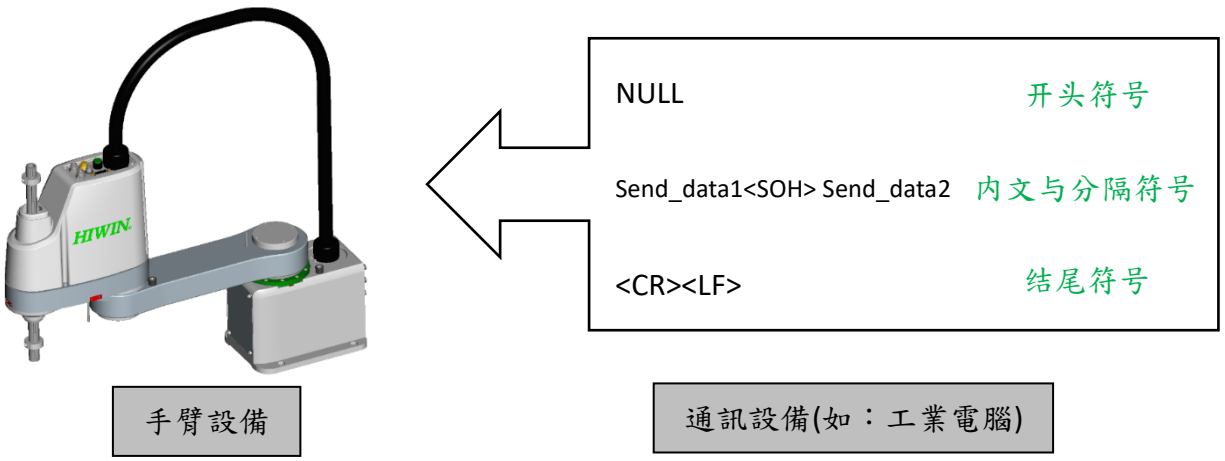
开启网路 EtherNet(TCP/IP)通讯。

自行宣告通讯格式起头符号为 "0x00"，代表意义为NULL(也可使用「""」取代)；分隔符号为 "0x01"，代表意义为<SOH>；结尾符号为 "0x0D0A"，代表意义为<CR><LF>。

程式执行时手臂只接收其他通讯设备传送NULL开头符号，<CR><LF>结尾符号，”<SOH>” 分隔符号的格式。

send_data1 资料为字串”Robot1”资讯，send_data2 资料为字串”Robot2”资讯，并通讯传送资讯给外部设备 send_data1, send_data2 资料。

等待读取由外部设备通讯的数值至 receive_data1 变数内。



■ **参考相关指令：**

COPEN、CCLOSE、CCLEAR、CREAD、CWRITE。

2.9 代码 I、输送带指令

此章节针对于代码 I 的输送带指令做各别说明

2.9.1@CNV_START 启动输送带追踪程序指令

■ **指令说明：**

启动输送带追踪功能，使对应的输送带编号 CNV_相关指令可使用，在此指令之后其他输送带追踪指令才会动作，用于输送带追踪程式的起头。。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 设定输送代追踪 Track 内各项动态抓取(飞抓)、输送带参数皆需完成设定，使用键盘方式撰写指令。

■ **指令格式：**

```
CNV_START CNV = [输送带编号值]
```

■ **格式/参数说明：**

⇨ CNV_START 为开始输送带追踪指令；CNV 为输送带的英文缩写，需使用者给予输送带数值；通常程式结尾搭配 CNV_END (结束输送带追踪指令)使用。

⇨ [输送带编号值] 为指定启动的输送带编号值，可输入范围为 1~4。

■ **范例：**

```
CNV_START CNV = 1
CNV_START CNV = 4
```

■ **范例说明：**

启动第 1 号输送带追踪功能。

启动第 4 号输送带追踪功能。

■ **参考相关指令：**

CNV_END。

2.9.2@CNV_END 结束输送带追踪程序指令

■ **指令说明：**

结束输送带追踪功能，使对应的输送带编号 CNV_相关指令不可使用，用于输送带追踪程式的结束。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 设定输送代追踪 Track 内各项动态抓取(飞抓)、输送带参数皆需完成设定，使用键盘方式撰写指令。

■ **指令格式：**

```
CNV_END CNV = [输送带编号值]
```

■ **格式/参数说明：**

⇨ CNV_END 为结束输送带追踪指令；CNV 为输送带的英文缩写，需使用者给予输送带数值；通常程式开头搭配 CNV_ START (结束输送带追踪指令)使用。

⇨ [输送带编号值] 为指定结束的输送带编号值，可输入范围为 1~4。

■ **范例：**

```
CNV_START CNV = 1
CNV_END CNV = 1
```

■ **范例说明：**

开启第 1 号输送带追踪功能。

结束第 1 号输送带追踪功能。

■ **参考相关指令：**

CNV_START。

2.9.3@CNV_PICK_QUANTITY 输送带追踪的抓取物件数量指令

■ **指令说明：**

设定手臂每次一趟动态抓取(飞抓)的物件最大数量变数，当抓取数量达到此设定值，便不再执行抓取动作指令，若未设定，预设值为 1 个物件。

常搭配 CNV_FULL(输送带抓取满额数量)指令使用。此变数设定同时会影响到 CNV_FULL 与 CNV_EMPTY 的状态。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用、使用键盘方式撰写指令。

■ **指令格式：**

CNV_PICK_QUANTITY = [工件数量数值参数值]

■ **格式/参数说明：**

⇨ CNV_PICK_QUANTITY 为输送带追踪抓取物件数量设定指令。

⇨ [工件数量数值参数值] 为抓取工件数量数值，可输入范围为 1 至 8 件的物件，整数型态。

■ **范例：**

CNV_PICK_QUANTITY = 2

■ **范例说明：**

设定手臂每次一趟飞抓的物件最大数量为 2。(适用在双吸嘴，每趟飞抓 2 个物件)

■ **参考相关指令：**

CNV_START、CNV_END、CNV_FULL。

2.9.4@CNV_TRIGGER_TIMES 输送带追踪的感测器触发次数指令

■ **指令说明：**

输送带追踪的感测器触发次数指令，机器手臂才会得到工作任务 QUEUE 进行抓取。此指令仅适用于感测器触发类型使用。可应用在多次感测器判断后才有一次工作任务，工作任务才能执行后续飞抓或飞放的动作。

※此功能需在介面设定，也可在程式中使用指令。若在程式中使用指令会以程式指令设定为优先。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用、使用键盘方式撰写指令。

⇨ 只限 SENSOR 触发，不可使用在视觉辨识系统上。

■ **指令格式：**

CNV_TRIGGER_TIMES[输送带编号] = [触发参数数值]

■ **格式/参数说明：**

⇨ [输送带编号] 为输送带的编号值，输入范围为 1 至 4，代表 CNV1 到 CNV4。

⇨ [触发参数数值] 为触发次数整数数值，输入范围为 1 至 100 次。

■ **范例：**

CNV_TRIGGER_TIMES[1] = 1

■ **范例说明：**

当编号 1 的输送带的感测器触发 1 次时，手臂会得到一个进行飞抓或飞放的工作任务。

■ **参考相关指令：**

CNV_START、CNV_END、CNV_QUEUE_SIZE 。

2.9.5@CNV_PICK 输送带追踪的抓取物件移动指令

■ **指令说明：**

输送带追纵抓取物件动作指令，简称飞抓，手臂等待系统派工后进行一次动态抓取(飞抓)动作。在此指令中，可能有 11 个参数需作设定，才能作输送带追纵抓取物件动作。

抓取成功手臂回到准备向下抓取的高度位置，再待下一个移动/摆放指令；抓取失败手臂回到开始动作时的位置，可根据输送带设定方式决定重抓 Retry 或者移除 Remove 等方式。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用。

■ **指令格式：**

```
CNV_PICK [输送带编号] [物件编号] [输出讯号      ] [点位名称] [下压高度距离     ]
[轨迹形式       ] [速度          ] [加速度  ] [输入讯号] [工具座标  ] [基座座标]
CNV_PICK [    CNV=#    ] [OBJ =#     ] [$DO[#] / $SO[#] ] [ P#          ] [Down=# mm      ]
[FINE=#/CONT=#] [Vel=# mm/s] [Acc =# %] [$DI[#]    ] [TOOL[#]    ] [BASE[#]    ]
```

■ **格式/参数说明：**

参数名称	参数格式	参数说明
[输送带编号]	CNV=#	指定启动的输送带 CNV 编号参数，#可输入范围为 1 至 4 的整数数值。若省略输送带编号，则输入不追踪。
[物件编号]	OBJ=#	抓取的物件 OBJ 编号参数，#可输入范围为 1 至 8 整数数值。或者省略物件编号，就会不指定物件编号，将所有 ALL 物件视为可抓取物件。
[输出讯号]	$DO[#] 或$SO[#] 或$MO[#]	抓取时，给与吸嘴开启讯号或上位控制系统的输出讯号，#可输入$DO 数位输出范围为 1 至 24 的数值。或使用$SO 现场总线输出范围为 9 至 128 的数值。或使用$MO 组合输出讯号。
[点位名称]	P# 或 Point	位置点位参数。位置的值为物件触发感测器讯号时的起始位置座标；若是使用影像/视觉触发时需忽略此点位，则不写点位名称。#可对应教导时纪录位置编号，例如 P1、P3…等，或者为使用者自行宣告 Point 名称。
[下压高度距离]	Down=# mm	手臂抓取时下压吸取物件之高度距离，以达到抓取物件；飞抓时手臂会先停在物件上方特定距离再向下压高度距离抓取，#可输入正整数数值，单位公厘 mm。
[轨迹形式]	FINE=# 或 CONT=#	为连续 CONT 或非连续 FINE 类型；并可在连续轨迹形式后面的百分比 1%~100%为平滑的程度，决定连续参数；连续型式 FINE 分成三个等级 0、1、2。

[速度]	Vel= #mm/s	速度参数，DELTA 规格#可输入范围为 1 至 3000 的整数数值。(六轴规格范围为 1 至 2000 数值、SCARA-LU 系列 Vel=1~4000 数值)。可省略此参数，省略速度将参考上一行指令，或初始定义参数。
[加速度]	Acc=#%	加速度参数，#可输入范围为 1 至 100 的整数数值。数字越大表示加速度越快， 可省略此参数，省略加速度将参考上一行指令，或初始定义参数。
[输入讯号]	$DI[#] 或$SI[#] 或$MI[#]	抓取时，从吸嘴讯号或上位控制系统获得的输入讯号，判断是否成功抓取物件讯号，#可输入数位输入范围为 1 至 24 的数值。或使用$SI 现场总线输出范围为 9 至 128 的数值。或使用$MI 组合输出讯号。
[工具座标]	TOOL[#]	工具座标参数，#可输入范围为 0 至 15 的整数。可省略此参数，省略工具座标将参考上一行指令，或初始定义参数。
[基座座标]	BASE[#]	基座座标参数，#可输入范围为 0 至 31 的整数。可省略此参数，省略基座座标将参考上一行指令，或初始定义参数。

CNV_PICK 指令参数撰写顺序：

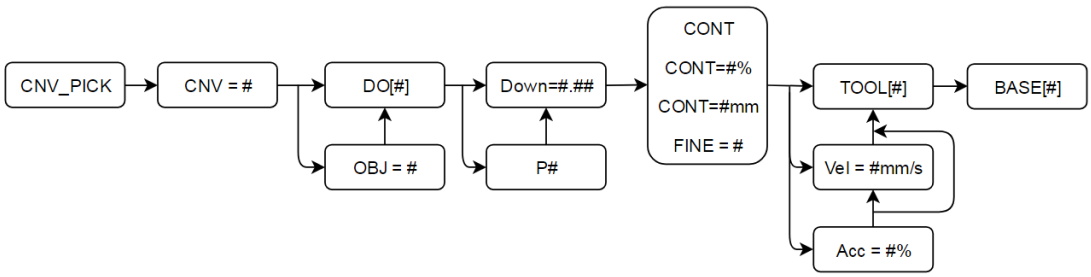


图 32 CNV_PICK 指令参数撰写顺序图

顺序图说明：

1) #为数字
2) CNV、OBJ、P、Vel、Acc 参数可视使用情况忽略参数项目。
3) (轨迹形式) CONT、CONT = #%、CONT = #mm、FINE 参数请择一使用。

■ **范例 1：**

```
CNV_START CNV = 1
CNV_PICK CNV=1 OBJ=1 $DO[1] P1 Down=5.000mm FINE=1 Vel=2500mm/s Acc=50% $DI[1] TOOL[0]
BASE[0]
```

■ **范例 1 说明：**

⇨ 启动第 1 号输送带追踪功能。

⇨ 输送带追纵抓取物件动作指令，适用于感测器触发方式，手臂到编号 1 的输送带上，设定飞抓起始位置为 P1 点，下压 5mm 后输出$DO[1]讯号抓取编号 1 的物件，读取 DI[1]输入讯号确认抓取是否成功。手臂移动过程中，非连续轨迹参数为 FINE=1、速度参数为 2500mm/s、加速度参数为 50%、第 0 个工具座标参数、第 0 个基座座标参数。

■ **范例 2：**

```
CNV_START CNV = 1
CNV_PICK CNV=1 OBJ=1 $DO[1] Down=5.000mm FINE=1 Vel=2500mm/s Acc=50% $DI[1] TOOL[0]
BASE[0]
```

■ **范例 2 说明：**

⇨ 启动第 1 号输送带追踪功能。

⇨ 输送带追纵抓取物件动作指令，适用于影像/视觉触发方式，手臂到编号 1 的输送带上，下压 5mm 后输出$DO[1]讯号抓取编号 1 的物件，读取 DI[1]输入讯号确认抓取是否成功。手臂移动过程中，非连续轨迹参数为 FINE=1、速度参数为 2500mm/s、加速度参数为 50%、第 0 个工具座标参数、第 0 个基座座标参数。

■ **参考相关指令：**

CNV_START、CNV_END、CNV_PLACE。

2.9.6@CNV_PLACE 输送带追踪的摆放物件移动指令

■ **指令说明：**

输送带追纵摆放物件动作指令，简称飞放，手臂等待系统派工后进行一次动态抓取(飞抓)动作，在此指令中，可能有 11 个参数需作设定，才能作输送带追纵摆放物件动作。

摆放成功手臂回到准备向下抓取的高度位置，再待下一个移动/摆放指令；摆放失败手臂回到开始动作时的位置，可根据输送带设定方式决定重抓 Retry 或者移除 Remove 等方式。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用。

■ **指令格式：**

CNV_PLACE [输送带编号] [物件编号] [输出讯号] [点位名称] [下压高度距离]
[轨迹形式] [速度] [加速度] [工具座标] [基座座标]
CNV_PLACE [CNV=#] [OBJ =#] [$DO[#] / $SO[#]] [P#] [Down=# mm]
[FINE=#/CONT=#] [Vel=# mm/s] [Acc =# %] [TOOL[#]] [BASE[#]]

■ **格式/参数说明：**

参数名称	参数格式	参数说明
[输送带编号]	CNV=#	指定启动的输送带 CNV 编号参数，#可输入范围为 1 至 4 的整数数值。若省略输送带编号，则输入不追踪。
[物件编号]	OBJ=#	抓取的物件 OBJ 编号参数，#可输入范围为 1 至 8 整数数值。或者省略物件编号，就会不指定物件编号，将所有 ALL 物件视为可被摆放物件。
[输出讯号]	$DO[#] 或$SO[#] 或$MO[#]	摆放时，给与吸盘关闭讯号或上位控制系统的输出讯号，#可输入$DO 数位输出范围为 1 至 24 的数值。或使用$SO 现场总线输出范围为 9 至 128 的数值。或使用$MO 组合输出讯号。
[点位名称]	P# 或 Point	位置点位参数。位置的值为物件触发感测器讯号时的起始位置座标；若是使用影像/视觉触发时需忽略此点位，则不写点位名称。#可对应教导时纪录位置编号，例如 P1、P3…等，或者为使用者自行宣告 Point 名称。
[下压高度距离]	Down=# mm	手臂摆放时下压吸取物件之高度距离，以达到摆放物件；飞抓时手臂会先停在物件上方特定距离再向下压高度距离摆放，#可输入正整数数值，单位公厘 mm。
[轨迹形式]	FINE=# 或 CONT=#	为连续 CONT 或非连续 FINE 类型；并可在连续轨迹形式后面的百分比 1%~100%为平滑的程度，决定连续参数；连续型式 FINE 分成三个等级 0、1、2。
[速度]	Vel= #mm/s	速度参数，DELTA 规格#可输入范围为 1 至 3000 的整数数值。(六轴规格范围为 1 至 2000 数值、SCARA-LU 系列

		Vel=1~4000 数值)。可省略此参数，省略速度将参考上一行指令，或初始定义参数。
[加速度]	Acc=#%	加速度参数，#可输入范围为 1 至 100 的整数数值。数字越大表示加速度越快， 可省略此参数，省略加速度将参考上一行指令，或初始定义参数。
[工具座标]	TOOL[#]	工具座标参数，#可输入范围为 0 至 15 的整数。可省略此参数，省略工具座标将参考上一行指令，或初始定义参数。
[基座座标]	BASE[#]	基座座标参数，#可输入范围为 0 至 31 的整数。可省略此参数，省略基座座标将参考上一行指令，或初始定义参数。

CNV_PLACE 指令参数撰写顺序：

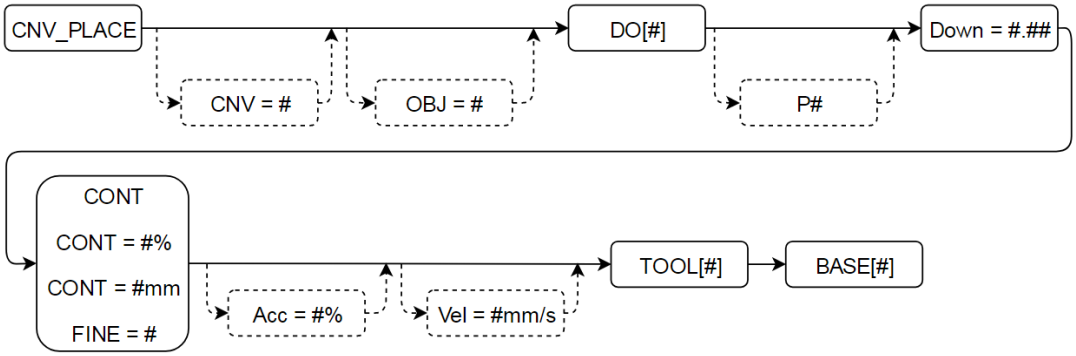


图 33 CNV_PLACE 指令参数撰写顺序图

顺序图说明：

1) #为数字
2) CNV、OBJ、P、Vel、Acc 参数可视使用情况忽略参数项目。
3) (轨迹形式) CONT、CONT = #%、CONT = #mm、FINE 参数请择一使用。

■ **范例：**

CNV_PLACE CNV=1 OBJ=1 $DO[1] P1 Down=5.000mm CONT=100% Vel=2000mm/s Acc=50% TOOL[0] BASE[0]

■ **范例说明：**

输送带追纵摆放物件动作指令，手臂到编号 1 的输送带上，设定摆放起始位置为 P1 点，下压 5mm 后输出$DO[1]讯号摆放编号 1 的物件。手臂移动过程中，连续轨迹参数为 CONT 100%、速度参数为 2000mm/s、加速度参数为 50%、第 0 个工具座标参数、第 0 个基座座标参数。

■ **参考相关指令：**

CNV_START、CNV_END、CNV_PICK。

2.9.7@CNV_OBJECT 输送带追踪的抓取之物件编号指令

■ **指令说明：**

读取手臂当下抓取的物件编号，可用于判断目前物件编号条件并依此执行特定动作，最多取 8 个物件，故最多可有 8 种物件编号分类 8 种动作状态，需搭配 CNV_PICK(飞抓)、CNV_PLACE(飞放)指令使用，CNV_PICK 抓取物件后更新状态，CNV_PLACE 摆放物件后状态归零。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用。

⇨ 使用 CNV_PICK(飞抓)、CNV_PLACE(飞放)后 CNV_OBJECT 物件编号才会变更。

■ **指令格式：**

```
IF CNV_OBJECT == <数值> THEN
    [动作指令]
ENDIF
```

■ **格式/参数说明：**

⇨ [变数名称] 为使用者只定义一个变数，例如：X1、Y1 者等。

⇨ 使用 IF 条件式判断，以物件编号数值当成条件判断式，<数值>为 1 到 8 的整数，当条件成立就执行以下动作指令内容。

⇨ [动作指令] 为IF判断式成立后之陈述动作，逻辑程式、移动指令或讯号指令之组合。

■ **范例： (以感测器方式动态抓取)**

```
CNV_START CNV = 1
CNV_START CNV = 2
CNV_START CNV = 3
CNV_PICK_QUANTITY = 2
WHILE CNV_FULL == FALSE
CNV_PICK CNV=1 $DO[1] P1 Down=5.000mm FINE=1 Vel=2500mm/s Acc=50% $DI[1]
ENDWHILE
IF   CNV_OBJECT==1   THEN
    CNV_PLACE CNV=2 OBJ=1 $DO[1] P2 Down=5.000mm FINE=1 Vel=2500mm/s Acc=50%
ENDIF
IF   CNV_OBJECT==2   THEN
    CNV_PLACE CNV=3 OBJ=2 $DO[1] P3 Down=5.000mm FINE=1 Vel=2500mm/s Acc=50%
ENDIF
CNV_END CNV = 1
CNV_END CNV = 2
CNV_END CNV = 3
```

■ **范例说明：**

- 启动第 1、2、3 号输送带追踪功能。
- 手臂到编号 1 的输送带上飞抓取物件后(使用$DO[1]输出讯号，下压 5mm 抓取，判断$DI[1]输出讯号是否抓料成功)，
- 再依照 IF 判断式条件，将物件摆放至对应的输送带上，若确认输送带物件(CNV_OBJECT)编号为 2，则飞放到编号 2 的输送带上(使用$DO[1]输出讯号，下压 5mm 摆放)；若确认输送带物件(CNV_OBJECT)编号为 3，则飞放到编号 3 的输送带上(使用$DO[1]输出讯号，下压 5mm 摆放)。
- 结束第 1、2、3 号输送带追踪功能。

■ **参考相关指令：**

CNV_START、CNV_END、CNV_PICK、CNV_PLACE。

2.9.8@CNV_FULL 输送带追踪的计数物件数量满额指令

■ **指令说明：**

输送带追踪的物件数量判定满额状态变数指令，确认目前动态抓取数量是否等于设定的最大数量变数(CNV_PICK_QUANTITY)，飞抓状态为布林变数；根据已飞抓/非放达到满额之状态作真 TRUE 或假 FALSE 的判定。

为条件判断式之状态指令，需搭配 WHILE…ENDWHILE 作条件设定。相对之状态指令为 CNV_EMPTY。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用。

⇨ 条件判断成立后，搭配 CNV_PICK(飞抓)、CNV_PLACE(飞放)等指令使用。

■ **指令格式：**

```
[回圈判断式开头] CNV_FULL [比较运算子] [布林变数]
    [动作指令]
[回圈判断式结尾]
```

格式范例：

```
WHILE CNV_FULL == FLASE
        [动作指令]
ENDWHILE
```

■ **格式/参数说明：**

⇨ [回圈判断式开头] 为使用回圈判断指令开头字母，可用于 WHILE 指令或 IF…THEN 指令判断式作开头。

⇨ [回圈判断式结尾] 为使用回圈判断指令结尾字母，呼应开头字母的 ENDWHILE 指令或 ENDIF 指令判断式作结尾。

⇨ [比较运算子] 为 CNV_FULL 与布林变数是否成立条件判断式的符号，例如”==”或”!=” 两个比较运算子较为常用。

⇨ [布林变数] 为真 TRUE 表示启动或假 FALSE 表示关闭，布林逻辑变数，二者择一状态。

⇨ [动作指令] 为回圈判断式成立后之陈述动作，逻辑程式、移动指令或讯号指令之组合。

■ **范例：**

```
CNV_START CNV = 1
CNV_PICK_QUANTITY = 2
WHILE CNV_FULL == FALSE
    CNV_PICK CNV=1 OBJ=1 $DO[1] Down=5.000mm FINE=1 Vel=2500mm/s Acc=50% $DI[1] TOOL[0]
    BASE[0]
ENDWHILE
```

■ **范例说明：**

- 启动第 1 号输送带追踪功能。
- 设定手臂每次一趟飞抓的物件最大数量为 2。
- 当手臂飞抓物件数量未满额 2 时(FALSE)，将进行物件飞抓。

(飞抓条件:下压 5mm 后输出$DO[1]讯号抓取编号 1 的物件，读取 DI[1]输入讯号确认抓取是否成功。手臂移动过程中，非连续轨迹参数为 FINE=1、速度参数为 2500mm/s、加速度参数为 50%、第 0 个工具座标参数、第 0 个基座座标参)

■ **参考相关指令：**

CNV_START、CNV_END、CNV_PICK_QUANTITY、CNV_EMPTY。

2.9.9@CNV_EMPTY 输送带追踪的计数物件数量空指令

■ **指令说明：**

输送带追踪的物件数量判定空状态变数指令，确认目前物件数量是否为空/零(0)，飞抓状态为布林变数；根据已飞抓/非放达到空/零之状态作真 TRUE 或假 FALSE 的判定。

为条件判断式之状态指令，需搭配 WHILE…ENDWHILE 作条件设定。相对之状态指令为 CNV_FULL。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用。

⇨ 条件判断成立后，搭配 CNV_PICK(飞抓)、CNV_PLACE(飞放)等指令使用。

■ **指令格式：**

```
[回圈判断式开头] CNV_EMPTY [比较运算子] [布林变数]
    [动作指令]
[回圈判断式结尾]
```

格式范例：

```
WHILE CNV_EMPTY == FLASE
        [动作指令]
ENDWHILE
```

■ **格式/参数说明：**

⇨ [回圈判断式开头] 为使用回圈判断指令开头字母，可用于 WHILE 指令或 IF…THEN 指令判断式作开头。

⇨ [回圈判断式结尾] 为使用回圈判断指令结尾字母，呼应开头字母的 ENDWHILE 指令或 ENDIF 指令判断式作结尾。

⇨ [比较运算子] 为 CNV_EMPTY 与布林变数是否成立条件判断式的符号，例如”==”或”!=” 两个比较运算子较为常用。

⇨ [布林变数] 为真 TRUE 表示启动或假 FALSE 表示关闭，布林逻辑变数，二者择一状态。

⇨ [动作指令] 为回圈判断式成立后之陈述动作，逻辑程式、移动指令或讯号指令之组合。

■ **范例：**

```
CNV_START CNV = 1
CNV_PICK_QUANTITY = 2
WHILE CNV_EMPTY == FALSE
    CNV_PLACE CNV=1 OBJ=1 $DO[1] Down=5.000mm FINE=1 Vel=2500mm/s Acc=50%    TOOL[0]
    BASE[0]
ENDWHILE
```

■ **范例说明：**

- 启动第 1 号输送带追踪功能。
- 设定手臂每次一趟飞抓的物件最大数量为 2。
- 当手臂飞抓物件数量未达到为空/零(0)时(FALSE)，将进行物件摆放。

(摆放条件:下压 5mm 后输出$DO[1]讯号抓取编号 1 的物件。手臂移动过程中，非连续轨迹参数为 FINE=1、速度参数为 2500mm/s、加速度参数为 50%、第 0 个工具座标参数、第 0 个基座座标参)

■ **参考相关指令：**

CNV_START、CNV_END、CNV_FULL。

2.9.10 @CNV_SET_DELAY_TIME 输送带追踪的延迟结束时间

■ **指令说明：**

输送带追踪的飞抓/飞放追纵延迟结束时间指令。藉由设定此变数，能够使手臂在到达进行飞抓或飞放的定位之后，继续跟着物件移动特定的时间，再进行离开的动作；让整个飞抓/飞放有调整离开的时间，让使用者对于飞抓/飞放动作可以完整完成。

如图下所示，在飞抓或飞放到定位后，跟着物件移动 50ms 后，才会离开。

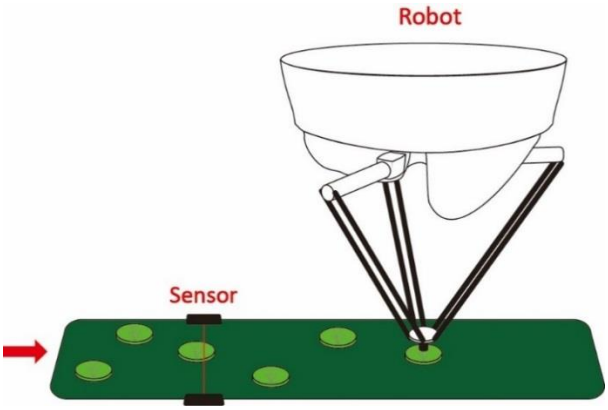


图 34 Delta 到定位示意图(到定位就立即离开)

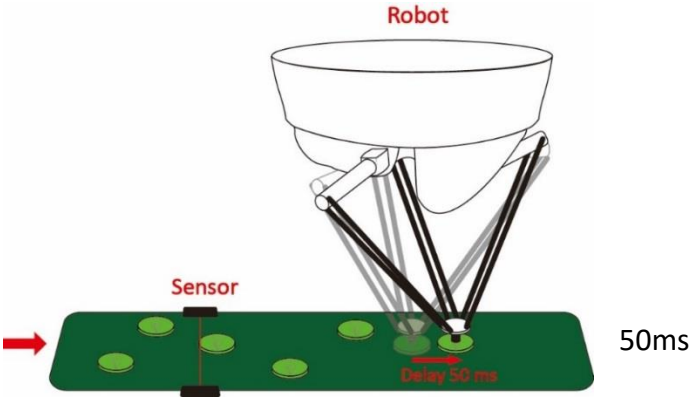


图 35 Delta 延迟 50 ms 示意图(跟随 50ms 才离开)

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用。

■ **指令格式：**

CNV_SET_DELAY_TIME[输送带编号] = [延迟时间]

■ **格式/参数说明：**

⇨ [输送带编号] 为欲延迟时间的输送带编号，输入范围为 1 至 4 的整数数值。

⇨ [延迟时间] 为设定跟随输送带移动的延迟时间，输入范围为 0 至 1500 的整数数值，单位为 ms(0.001 秒)。

■ **范例：**

CNV_SET_DELAY_TIME[1] = 100

■ **范例说明：**

设定手臂于第 1 号输送带，在到达定位点后跟随 100ms 延迟时间后，才离开定位位置，继续后面程式指令。

■ **参考相关指令：**

CNV_START、CNV_END。

2.9.11 @CNV_QUEUE_SIZE 输送带追踪的已触发排序数量指令

■ **指令说明：**

输送带追踪的已触发排序但尚未抓取数量变数，此变数代表在输送带上，已被感测器触发，但尚未被抓取的物件数量，获取数量后可作后续程序的判定使用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用。

■ **指令格式：**

```
CNV_QUEUE_SIZE [输送带编号]
```

■ **格式/参数说明：**

⇨ [输送带编号] 为输送带的编号值，输入范围为 1 至 4，代表 CNV1 到 CNV4。变数型别为正整数，可用于 WHILE 指令或 IF 指令等逻辑判断式使用。

■ **范例：**

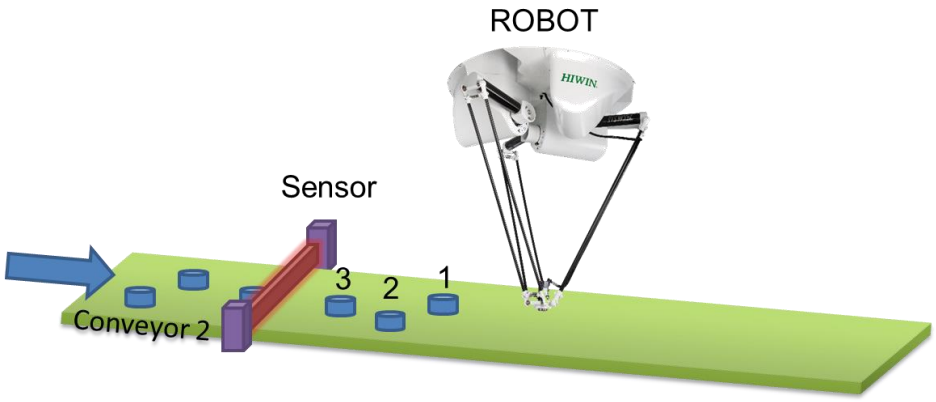


图 36 CNV_QUEUE_SIZE 输送带示意图

```
IF CNV_QUEUE_SIZE[4] > 2 THEN
    $C[3] = 0
    $DO[3] = TRUE
ENDIF
```

■ **范例说明：**

如图所示，输送带 4 的感测器已触发到三个物件，程式达成大于 2 的条件，将执行 IF 判断式内的内容，第 3 个计数器归零且第三个数位输出讯号为真(TRUE)。

■ **参考相关指令：**

CNV_START、CNV_END、CNV_PICK、CNV_PLACE、CNV_PICK_QUANTITY。

2.9.12 @CNV_OBJ_CNT_DIST 输送带追踪的编码器相差值指令

■ **指令说明：**

输送带追踪的已触发物件位置之编码器相差值变数，当 CNV_QUEUE_SIZE[NUM]变数值大于等于 2 时(输送带上已被触发的物件有 2 个以上)，即可使用。

此变数可显示感测器触发的第一个物件位置及第二个物件位置之 Encoder 相差值。通常用于判断触发的物件是否连续。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用。

■ **指令格式：**

```
CNV_OBJ_CNT_DIST[输送带编号]
```

■ **格式/参数说明：**

⇨ [输送带编号] 为输送带的编号值，输入范围为 1 至 4，代表 CNV1 到 CNV4。变数型别为正整数，可用于 WHILE 指令或 IF 指令。

■ **范例：**

```
IF CNV_QUEUE_SIZE[2] > 1 THEN
    IF CNV_OBJ_CNT_DIST[2] < 2600 THEN
        $C[3] = 0
        $DO[3] = TRUE
    ELSE
        $C[3] = 1
        $DO[3] = FLASE
    ENDIF
ENDIF
```

■ **范例说明：**

如果侦测到输送带 2 上的物件数量大于 1 个时，判断第一个侦测及第二个侦测的编码器数值是否小于 2600(2600 是自行量测出来的值)，若是则为连续，否则为不是连续的物件。将执行 IF 判断式内的内容，第 3 个计数器归零且第三个数位输出讯号为真(TRUE)；否则第 3 个计数器归 1 且第三个数位输出讯号为假(FALSE)。。

■ **参考相关指令：**

CNV_START、CNV_END、CNV_QUEUE_SIZE、CNV_PICK、CNV_PLACE、CNV_PICK_QUANTITY。

2.9.13 @CNV_PLACE_BATCH 输送带追踪的可飞放的最大次数

■ **指令说明：**

飞放状态变数，用于需要在同一个工作任务位置飞放复数物件的状况。
当释放物件输送带的感测器触发时，手臂会得到一个可以执行飞放的工作任务位置，而手臂在该工作任务位置最大的飞放次数，可由此变数宣告设定。可用于产线在多个物件堆迭数量的设定使用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用。

■ **指令格式：**

CNV_PLACE_BATCH[输送带编号] = [最大的飞放次数]

■ **格式/参数说明：**

⇨ [输送带编号] 为输送带的编号值，输入范围为 1 至 4，代表 CNV1 到 CNV4。

⇨ [最大的飞放次数] 为正整数值，输入范围为 1 至 100 次； 若无指定数量，预设值为 1。

■ **范例：**

CNV_PLACE_BATCH[1] = 3

■ **范例说明：**

设定手臂会得到一个可以执行飞放的工作任务位置，而手臂在该工作任务位置宣告最大的飞放次数为 3 次。

■ **参考相关指令：**

CNV_START、CNV_END、CNV_TRIGGER_TIMES。

2.9.14 @CNV_RESET_ENC 输送带追踪的清除编码器的计数指令

■ **指令说明：**

输送带追踪的清除外部编码器的计数指令。使用者能够在撰写程式的流程当中，藉由使用此指令， 清除全部外部编码器的计数数值，使编码器数值为零，重新计数。 使用此指令，效果与在输送带校正页面按下清除”CLEAR”效果相同。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

```
CNV_RESET_ENC
```

■ **格式/参数说明：**

⇨ CNV_RESET_ENC 为让全部外部编码器数值归零的程式指令。

■ **范例：**

CNV_RESET_ENC

■ **范例说明：**

程式读到此指令全部外部编码器数值清除为零。

■ **参考相关指令：**

CNV_START、CNV_END。

2.9.15 @CNV_QUEUE_REMOVE 输送带追踪的移除的序列指令

■ **指令说明：**

输送带追踪的移除飞抓/飞放暂存区前面的序列指令。抓取/摆放状态变数，使用者能够在撰写程式的流程当中，藉由使用此指令 ，将放在等待伫列最前面的暂存值移除。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

■ **指令格式：**

CNV_QUEUE_REMOVE[输送带编号]

■ **格式/参数说明：**

⇨ [输送带编号] 为输送带的编号值，输入范围为 1 至 4，代表 CNV1 到 CNV4。

■ **范例：**

CNV_QUEUE_REMOVE[1]

■ **范例说明：**

移除第一个输送带的伫列暂存值。

■ **参考相关指令：**

CNV_START、CNV_END、CNV_QUEUE_SIZE、CNV_RESET_ENC。

2.9.16 @CNV_PICK_ACC 输送带追踪的下压加速度时间指令

■ **指令说明：**

需搭配输送带追踪功能指令(CNV_START)，设定输送带追踪下压的加速度时间。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用。

■ **指令格式：**

CNV_PICK_ACC[输送带编号] = [加速度参数值]

■ **格式/参数说明：**

⇨ [输送带编号] 为输送带的编号值，输入范围为 1 至 4，代表 CNV1 到 CNV4。

⇨ [加速度参数值]变数型别为正整数值，输入范围为 10 至 100 之间，单位为毫秒 ms；若无指定数量，预设值为 30 ms。

■ **范例：**

```
CNV_START CNV = 1
CNV_PICK_ACC[1] = 50
```

■ **范例说明：**

启动第 1 号输送带追踪功能。

将输送带追踪时的下压的加速度时间设定为 50ms。

■ **参考相关指令：**

CNV_START、CNV_END、SET_ACC。

2.9.17 @CNV_OFFSET_X 输送带追踪的 X 轴位置偏移值指令

■ **指令说明：**

需搭配输送带追踪功能指令(CNV_START)，使用者能够在撰写程式的流程当中，在动态抓取，设定输送带上 X 轴的偏移 Offset 距离值。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用。

■ **指令格式：**

```
CNV_OFFSET_X[输送带的编号] = [偏移距离值]
```

■ **格式/参数说明：**

⇨ [输送带的编号] 为输送带的编号，输入范围为 1~4，代表 CNV1 到 CNV4，若不输入为指定全部。

⇨ [偏移距离值] 为轴的偏移距离值，单位为公厘 mm。

■ **范例 1：**

```
CNV_START CNV = 1
CNV_OFFSET_X[1] = 10
```

■ **范例 1 说明：**

启动第 1 号输送带追踪功能。

输送带 CNV1 中，X 轴方向偏移 10mm 距离。

■ **范例 2：**

```
CNV_START CNV = 1
CNV_OFFSET_X = -20
```

■ **范例 2 说明：**

启动第 1 号输送带追踪功能。

全部输送带(CNV1~CNV4)中，X 轴方向偏移-20mm 距离。

■ **参考相关指令：**

CNV_START、CNV_END、CNV_OFFSET_Y、CNV_OFFSET_Z、CNV_START。

2.9.18 @CNV_OFFSET_Y 输送带追踪的 Y 轴位置偏移值指令

■ **指令说明：**

需搭配输送带追踪功能指令(CNV_START)，使用者能够在撰写程式的流程当中，在动态抓取，设定输送带上 Y 轴的偏移 Offset 距离值。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用。

■ **指令格式：**

```
CNV_OFFSET_Y[输送带的编号] = [偏移距离值]
```

■ **格式/参数说明：**

⇨ [输送带的编号] 为输送带的编号，输入范围为 1~4，代表 CNV1 到 CNV4，若不输入为指定全部。

⇨ [偏移距离值] 为轴的偏移距离值，单位为公厘 mm。

■ **范例 1：**

```
CNV_START CNV = 1
CNV_OFFSET_Y[1] = 10
```

■ **范例 1 说明：**

启动第 1 号输送带追踪功能。

输送带 CNV1 中，Y 轴方向偏移 10mm 距离。

■ **范例 2：**

```
CNV_START CNV = 1
CNV_OFFSET_Y = -20
```

■ **范例 2 说明：**

启动第 1 号输送带追踪功能。

全部输送带(CNV1~CNV4)中，Y 轴方向偏移-20mm 距离。

■ **参考相关指令：**

CNV_START、CNV_END、CNV_OFFSET_X、CNV_OFFSET_Z、CNV_START。

2.9.19 @CNV_OFFSET_Z 输送带追踪的 Z 轴位置偏移值指令

■ **指令说明：**

需搭配输送带追踪功能指令(CNV_START)，使用者能够在撰写程式的流程当中，在动态抓取，设定输送带上 Z 轴的偏移 Offset 距离值。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用。

■ **指令格式：**

```
CNV_OFFSET_Z[输送带的编号] = [偏移距离值]
```

■ **格式/参数说明：**

⇨ [输送带的编号] 为输送带的编号，输入范围为 1~4，代表 CNV1 到 CNV4，若不输入为指定全部。

⇨ [偏移距离值] 为轴的偏移距离值，单位为公厘 mm。

■ **范例 1：**

```
CNV_START CNV = 1
CNV_OFFSET_Z[1] = 10
```

■ **范例 1 说明：**

启动第 1 号输送带追踪功能。

输送带 CNV1 中，Z 轴方向偏移 10mm 距离。

■ **范例 2：**

```
CNV_START CNV = 1
CNV_OFFSET_Z = -20
```

■ **范例 2 说明：**

启动第 1 号输送带追踪功能。

全部输送带(CNV1~CNV4)中，Z 轴方向偏移-20mm 距离。

■ **参考相关指令：**

CNV_START、CNV_END、CNV_OFFSET_X、CNV_OFFSET_Y、CNV_START。

2.9.20 @CNV_SPEED 输送带追踪的取得输送带速度指令

■ **指令说明：**

输送带状态变数，使用者可读取目前输送带的速度。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.0.1 以上	Caterpillar1.0.1 以上	预读 CONT

⇨ 使用 CNV_START 启动指定输送带后才能使用。

■ **指令格式：**

[自定义变数名称] = CNV_SPEED[输送带编号]

■ **格式/参数说明：**

⇨ [自定义变数名称]为自定义的变数名称，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」。

⇨ [输送带编号] 为输送带的编号值，输入范围为 1 至 4，代表 CNV1 到 CNV4。

⇨ 或采用暂存器作为输入暂存值。

⇨ 透过 CNV_SPEED 指令获得输送带速度后，可应用于后续程式作条件判定。

■ **范例 1：**

```
INT G1
G1 =CNV_SPEED[1]
```

■ **范例 1 说明：**

宣告 INT 变数 G1， 输送带 1 当前速度指定 G1 变数值，作为后续程序判定使用。

■ **范例 2：**

```
$C[1]=CNV_SPEED[1]
```

■ **范例 2 说明：**

输送带 1 当前速度指定暂存器 1 作为显示。

■ **参考相关指令：**

CNV_START、CNV_END、$C[1]、INT 。

2.10 代码J、讯息提示功能指令

2.10.1 @USER_ALARM 触发使用者自定义警报

■ **指令说明：**

若使用者有需要在程式执行中，自行判断状况而发布警报，可使用此指令功能在程式运行中发布警报，警报记录的文字内容为使用于介面上所设定的文字内容。发布警报后，会停止程式执行，并在显示视窗显示错误码(Err01-09-XX)及设定的文字内容，同时也会记录在历史记录之中。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS2.0.1 以上	Caterpillar1.0.1 以上	不预读 FINE

■ **指令格式：**

```
USER_ALARM[#]
```

■ **格式/参数说明：**

⇨ #为数字，正整数型别，表示为触发第#个自定义警报，可输入范围为 1 至 10，不可为 0。

■ **范例：**

```
IF $DI[2] == TRUE AND $DI[3] == TRUE THEN
    USER_ALARM[3]
ENDIF
```

■ **范例说明：**

IF 如果数位输入讯号 2 与 3 同时间都为真(TRUE)，将触发第 3 个自定义警报，错误码 Err01-09-12 +使用者定义警报文字内容，且手臂会停止运行 。

■ **参考相关指令：**

IF。

2.11 代码 K、模拟环境指令

此章节针对于代码 K 的模拟环境指令做各别说明

2.11.1 @ADDTOOL 模拟画面新增末端工具指令

■ **指令说明：**

在离线版本软体增加模拟工具显示画面，新增使用者绘制 3D 工具模型，显示于模拟显示画面中，当手臂(法兰面)移动时，模拟工具画面也会跟随于手臂移动各种姿态，让使用者能更符合与更容易体会手臂装上工具后的模拟动作/姿态等资讯。

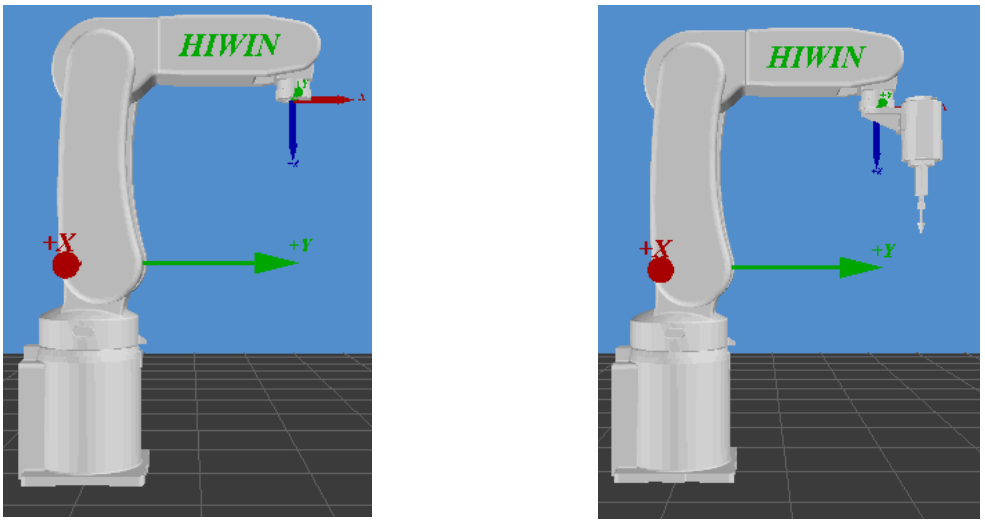


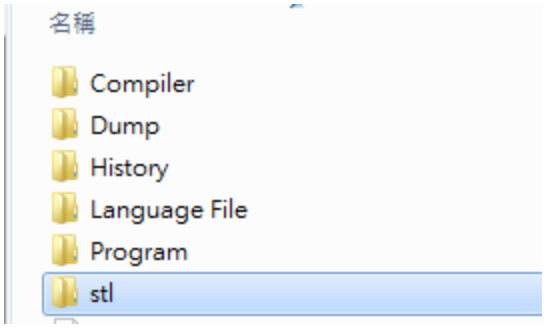
图 37 增加修毛边末端工具模拟画面图(左：未增加；右：已增加)

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.1 以上	Caterpillar 不适用	预读 CONT

⇨ 使用绘图软体 3D 图档转成.stl 格式(档案请勿太大，会无法运作)。

⇨ 将 stl 档案放到离线软体资料夹 stl 的资料夹内。



⇨ HRSS3.3.1 以下或 HRSS3.2 的版本，不能调整位置与颜色。

■ **指令格式：**

ADDTOOL [档案名称] [偏移法兰面距离资讯] [颜色资讯]
ADDTOOL [档案名称] [P:X, Y, Z, A, B, C] [C: R, G, B]

■ **格式/参数说明：**

⇨ [档案名称] 为 3D 图档转为.stl 的图档档案名称，可接受的档案名称为英文大小写开头或加底线的档案名称。

⇨ 不需输入副档名称(.stl)，以 ee.stl 为例，只需键入档案名称"ee"即可。

⇨ 3D 图档转 stl 档需为 binary 格式，且档案坐标系与法兰面末端坐标系方向一致。

■ **范例 1：** (工具档案名称：XEG_FINGER.stl)

ADDTOOL XEG_FINGER

■ **范例 1 说明：**

如下图，在离线模拟画面中，增加一组 XEG 电动夹爪并含指部(XEG_FINGER)的模拟工具画面。

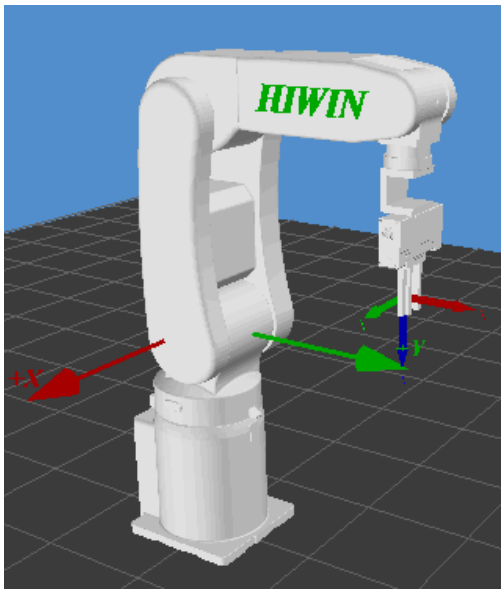


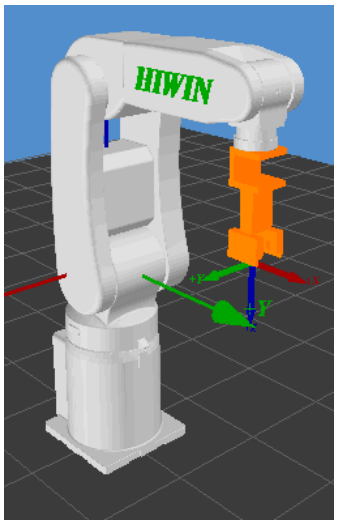
图 38 含电动夹爪与指部模拟图

■ **范例 2：** (工具档案名称：gripper_open.stl)

ADDTOOL gripper_open P:-25,0,0,0,0,0 C:250,80,0

■ **范例 2 说明：**

如下图，在离线模拟画面中，增加一组气动夹爪(gripper_open)的模拟工具画面，物件距离原始法兰面工具座标系 X 轴-25mm，Y、Z 轴 0mm，对 X、Y、Z 轴旋转角为 0 度，颜色资讯配置(R,G,B)为(250,80,0)。



■ **参考相关指令：**

SHOW_TOOL、ADDOBJ、SHOW_OBJ、MOVEFLOOR、AXISON、AXISOFF。

2.11.2 @ADDOBJ 模拟画面新增周边物件指令

■ **指令说明：**

在离线版本软体增加模拟周边物件、设备显示画面，新增使用者绘制 3D 物件、设备模型，显示于模拟显示画面中，让使用者能更符合与更容易体会手臂物件、设备周边配置的模拟动作/姿态等资讯，确认是否手臂可以达置期望范围，并可搭配计时器功能，计算手臂来回动作的周期时间。

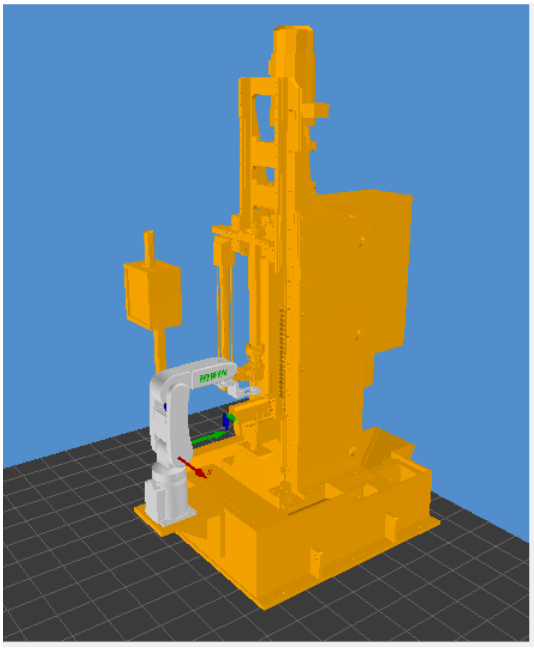


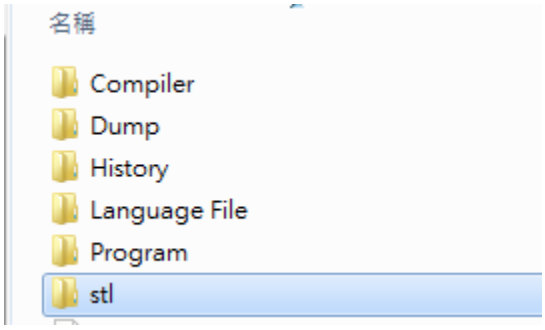
图 39 增加工具机物件模拟画面图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar 不适用	预读 CONT

⇨ 使用绘图软体 3D 图档转成.stl 格式(档案请勿太大，会无法运作)。

⇨ 将 stl 档案放到离线软体资料夹 stl 的资料夹内。



■ **指令格式：**

ADDOBJ [档案名称] [偏移手臂中心距离资讯] [颜色资讯]
ADDOBJ [档案名称] [P:X, Y, Z, A, B, C] [C: R, G, B]

■ **格式/参数说明：**

⇨ [档案名称] 为 3D 图档转为.stl 的图档档案名称，可接受的档案名称为英文大小写开头或加底线的档案名称。

⇨ 不需输入副档名称(.stl)，以 QW.stl 为例，只需键入档案名称"QW"即可。

⇨ 3D 图档转 stl 档需为 binary 格式，且档案坐标系与原始基座座标系方向一致。(原本座标系为置在于第二轴旋转中心点)

⇨ [偏移手臂中心距离资讯] 为由原始基座座标系为中心点使用者偏移距离资讯，格式为 P:X, Y, Z, A, B, C，X 表示为 X 轴距离，Y 表示为 Y 轴距离，Z 表示为 Z 轴距离，单位为公厘 mm；A 表示为 X 轴旋转角度，B 表示为 Y 轴旋转角度，C 表示为 Z 轴旋转角度，单位为度(degree)。

⇨ [颜色资讯] 为输入的 3D 物件颜色搭配，格式为 C:R,G,B，采用红、绿、蓝颜色搭配，可输入范围数值为 0~255，例如 C:0,0,0 为黑色；C:255,255,255 为白色。

■ **范例：** (输送带物件档案名称：conveyor.stl)

ADDOBJ conveyor P: 0,470,-130,0,0,30 C: 0,130,0

■ **范例说明：**

在离线模拟画面中，增加一输送带物件(conveyor)的模拟物件画面，物件距离原始基座座标系 X 轴 0mm，Y 轴 470mm，Z 轴-130mm，对 Z 轴旋转角为 30 度，颜色资讯配置(R,G,B)为(0,130,0)，参考下图。

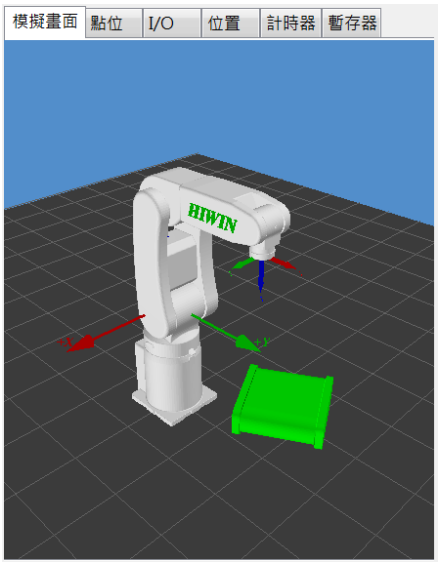


图 40 增加一输送带模拟图

■ **参考相关指令：**

ADDTOOL、SHOW_TOOL、SHOW_OBJ、MOVEFLOOR、AXISON、AXISOFF。

2.11.3 @SHOW_TOOL 模拟画面显示/隐藏末端工具指令

■ **指令说明：**

在离线模拟画面中，可透过增加末端工具指令 ADDTOOL 后，进一步透过此指令来显示或隐藏末端工具，在模拟画面更贴近于使用者欲模拟的流程情境。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar 不适用	预读 CONT

■ **指令格式：**

SHOW_TOOL [档案名称] [布林变数]

■ **格式/参数说明：**

⇨ [档案名称] 为 3D 图档转为.stl 的图档档案名称，可接受的档案名称为英文大小写开头或加底线的档案名称。

⇨ 不需输入副档名称(.stl)，以 ee.stl 为例，只需键入档案名称”ee”即可。

⇨ 3D 图档转 stl 档需为 binary 格式，且档案坐标系与法兰面末端坐标系方向一致。

⇨ [布林变数] 为真 TRUE 表示启动或假 FALSE 表示关闭，布林逻辑变数，二者择一状态。

■ **范例： (工具档案名称：XEG_FINGER.stl)**

```
ADDTOOL XEG_FINGER
WAIT SEC 2
SHOW_TOOL XEG_FINGER FALSE
```

■ **范例说明：**

如下图，在离线模拟画面中，增加一组 XEG 电动夹爪并含指部(XEG_FINGER)的模拟工具画面。等待 2 秒后，将模拟画面的电动夹爪工具隐藏。

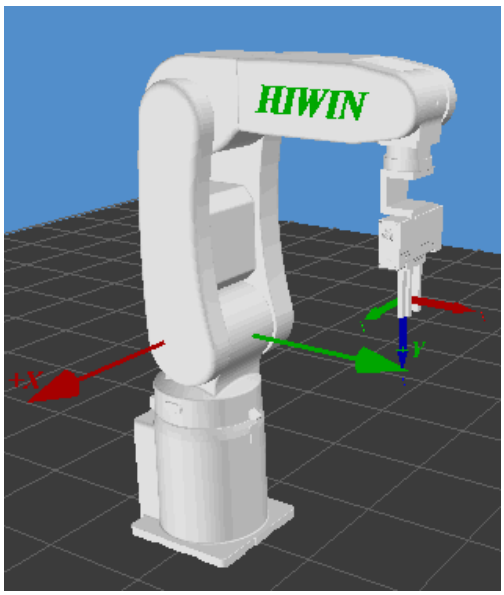


图 41 增加一组 XEG 电动夹爪并含指部模拟图

■ **参考相关指令：**

ADDOBJ、SHOW_OBJ、MOVEFLOOR、AXISON、AXISOFF。

2.11.4 @SHOW_OBJ 模拟画面显示显示/隐藏周边物件指令

■ **指令说明：**

在离线模拟画面中，可透过增加周边环境物件指令 ADDOBJ 后，进一步透过此指令来显示或隐藏周边环境物件，在模拟画面更贴近于使用者欲模拟的流程情境。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar 不适用	预读 CONT

■ **指令格式：**

SHOW_OBJ [档案名称] [布林变数]

■ **格式/参数说明：**

⇨ [档案名称] 为 3D 图档转为.stl 的图档档案名称，可接受的档案名称为英文大小写开头或加底线的档案名称。

⇨ 不需输入副档名称(.stl)，以 ee.stl 为例，只需键入档案名称"ee"即可。

⇨ 3D 图档转 stl 档需为 binary 格式，且档案坐标系与法兰面末端坐标系方向一致。

⇨ [布林变数] 为真 TRUE 表示启动或假 FALSE 表示关闭，布林逻辑变数，二者择一状态。

■ **范例： (物件档案名称：conveyor.stl)**

```
ADDOBJ conveyor P: 0,470,-130,0,0,30 C: 0,130,0
WAIT SEC 1
SHOW_OBJ conveyor FALSE
```

■ **范例说明：**

在离线模拟画面中，增加一输送带物件(conveyor)的模拟物件画面，物件距离原始基座座标系 X 轴 0mm，Y 轴 470mm，Z 轴-130mm，对 Z 轴旋转角为 30 度，颜色资讯配置(R,G,B)为(0,130,0)。等待 1 秒后，将模拟画面的输送带物件(conveyor)隐藏。

■ **参考相关指令：**

ADDOBJ、SHOW_TOOL、MOVEFLOOR、AXISON、AXISOFF。

2.11.5 @MOVEFLOOR 模拟画面移动地板位置指令

■ **指令说明：**

在离线模拟画面中，移动地板位置指令，可用来配合机器手臂安装的位置，若手臂安装在一个手臂底座上，就可透过移动地板指令来调整移动距离，让手臂离线画面更符合现在真实环境。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar 不适用	预读 CONT

■ **指令格式：**

MOVEFLOOR [移动距离]

■ **格式/参数说明：**

⇨ [移动距离] 为模拟画面地板移动距离，以手臂底座为原始基准距离零，相较于基准的距离可由使用者来调整，单位为毫米(mm)，数字正值为原始世界座标 Z 轴向上；数字负值为原始世界座标 Z 轴向下。

■ **范例：**

MOVEFLOOR -1000

■ **范例说明：**

在离线模拟画面中，以手臂底座为基础，将地板平面位置向下移动 1000mm 距离。

■ **参考相关指令：**

ADDOBJ、SHOW_TOOL、SHOW_OBJ、AXISON、AXISOFF。

2.11.6 @AXISON 模拟画面显示座标系(轴)指令

■ **指令说明：**

在离线模拟画面中，可将笛卡尔座标系 XYZ 座标系座标轴与工具 TOOL 座标系座标轴打开，如下图为开启显示座标轴状态。

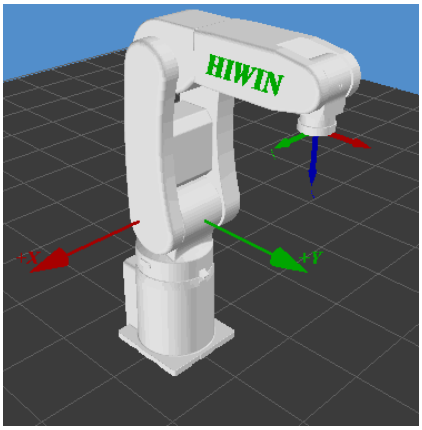


图 42 座标轴打开模拟图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar 不适用	预读 CONT

■ **指令格式：**

AXISON

■ **格式/参数说明：**

⇨ 开启显示笛卡尔座标系 XYZ 座标系座标轴与工具 TOOL 座标系座标轴。

■ **范例：**

AXISON

■ **范例说明：**

显示笛卡尔座标系 XYZ 座标系座标轴与工具 TOOL 座标系座标轴。

■ **参考相关指令：**

ADDOBJ、SHOW_TOOL、SHOW_OBJ、MOVEFLOOR、AXISOFF。

2.11.7 @AXISOFF 模拟画面隐藏座标系(轴)指令

■ **指令说明：**

在离线模拟画面中，可将笛卡尔座标系 XYZ 座标系座标轴与工具 TOOL 座标系座标轴关闭，如下图为开启隐藏座标轴状态。

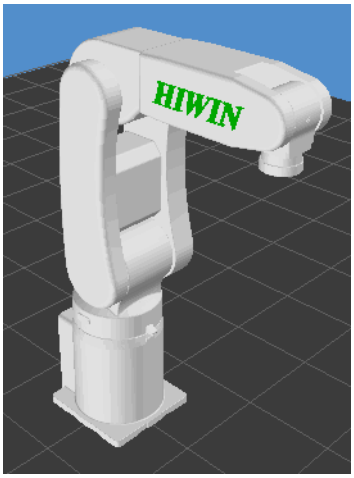


图 43 座标轴关闭模拟图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar 不适用	预读 CONT

■ **指令格式：**

AXISOFF

■ **格式/参数说明：**

⇨ 隐藏笛卡尔座标系 XYZ 座标系座标轴与工具 TOOL 座标系座标轴。

■ **范例：**

AXISOFF

■ **范例说明：**

隐藏笛卡尔座标系 XYZ 座标系座标轴与工具 TOOL 座标系座标轴。

■ **参考相关指令：**

ADDOBJ、SHOW_TOOL、SHOW_OBJ、MOVEFLOOR、AXISON。

2.12 代码 L、电动夹爪功能指令

此章节针对于代码 L 的电动夹爪功能指令做各别说明

2.12.1 @EG_OPEN 电动夹爪的开启连线指令

■ **指令说明：**

手臂搭配控制上银科技 XEG 系列电动夹爪适用指令，为电动夹爪开启连线指令，若使用电动夹爪需先使用开启连线指令，才能应用其他控制电动夹爪的指令。一台手臂只能同时间连线一台电动夹爪。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	不预读 FINE

⇨ 手臂透过 USB 通讯方式控制上银科技 XEG 系列的电动夹爪，为特定指令。

相同于软体电动夹爪设定介面功能「连线」按钮，如下图所示，路径：主选单>>启动设定 Start-up>>电动夹爪 ElectricGripper。



图 44 电动夹爪设定介面图

■ **指令格式：**

EG_OPEN(电动夹爪型号代码)

■ **格式/参数说明：**

⇨ EG_OPEN() 为开启电动夹爪连线指令。

⇨ (电动夹爪型号代码) 为使用者欲开启电动夹爪型号简易代码，对应的代码如下:

上银科技电动夹爪型号	代码
XEG-16	X16
XEG-32	X32
XEG-64	X64

■ **范例：**

```
EG_OPEN(X16)
WAIT SEC 1
EG_CLOSE
EG_OPEN(X32)
WAIT SEC 1
EG_CLOSE
EG_OPEN(X64)
WAIT SEC 1
EG_CLOSE
```

■ **范例说明：**

开启连线电动夹爪为 XEG-16 的款式，等待 1 秒后，再关闭 XEG-16 电动夹爪连线。
开启连线电动夹爪为 XEG-32 的款式，等待 1 秒后，再关闭 XEG-32 电动夹爪连线。
开启连线电动夹爪为 XEG-64 的款式，等待 1 秒后，再关闭 XEG-64 电动夹爪连线。
一台手臂只能同时间连线一台电动夹爪。

■ **参考相关指令：**

EG_CLOSE。

2.12.2 @EG_CLOSE 电动夹爪的关闭连线指令

■ **指令说明：**

可使用此指令来关闭当下连线中的 XEG 系列电动夹爪的连线，连线关闭后才可切换开启其他型号的电动夹爪的连线。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	不预读 FINE

⇨ 手臂透过 USB 通讯方式控制上银科技 XEG 系列的电动夹爪，为特定指令。

■ **指令格式：**

```
EG_CLOSE
```

■ **格式/参数说明：**

EG_ CLOSE 为关闭当前电动夹爪连线指令

■ **范例：**

```
EG_OPEN(X16)
WAIT SEC 1
EG_CLOSE
EG_OPEN(X64)
```

■ **范例说明：**

⇨ 开启连线电动夹爪为 XEG-16 的款式，等待 1 秒后，再关闭 XEG-16 电动夹爪连线。

⇨ 因要更换连线的电爪类型，故先关闭与 XEG-16 电动夹爪的连线

⇨ 再开启连线电动夹爪为 XEG-64 的款式

■ **参考相关指令：**

EG_OPEN、WAIT。

2.12.3 @EG_RESET 电动夹爪的重置原点指令

■ **指令说明：**

上银科技 XEG 系列电动夹爪重置原点动作指令，此指令会让电动夹爪开合零点位置数次，作建置原点位置与力矩模型计算。

在与电动夹爪完成连线后，需先使用此指令校正原点，才可进行电动夹爪的开关精准位置/力量控制，并确保各项控制参数读取与写入的正确性，建议使用者电动夹爪每次送电后，一定要进行原点校正；或使用次数过多时，建议也进行一次重置原点。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	不预读 FINE

⇨ 手臂透过 USB 通讯方式控制上银科技 XEG 系列的电动夹爪，为特定指令。

■ **指令格式：**

✧ 格式 1：

```
EG_RESET
```

✧ 格式 2：

```
EG_RESET CONT
```

■ **格式/参数说明：**

⇨ EG_RESET 为重置电动夹爪原点指令。

⇨ 格式 2，CONT 为连续轨迹指令，表示程式会直接往下预读执行，不会等电动夹爪重置移动完成，即开始下面程式执行。

■ **范例：**

```
EG_OPEN(X16)
EG_RESET
EG_RUN_MOVE(10, 40)
```

■ **范例说明：**

⇨ 开启与 XEG-16 电动夹爪的连线。

⇨ 再执行电动夹爪原点重置。

⇨ 再以速度 40mm/s，移到绝对位置 10mm 的位置。

■ **参考相关指令：**

EG_OPEN、EG_RUN_MOVE。

2.12.4 @EG_GET_STATUS 电动夹爪的状态取得指令

■ **指令说明：**

上银科技 XEG 系列电动夹爪目前使用状态显示，以数值代码方式回馈，用于程式或上位控制系统判断电动夹爪目前是否有夹持物件或为作动中状态，以作为物件辨识依据之一。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	不预读 FINE

⇨ 手臂透过 USB 通讯方式控制上银科技 XEG 系列的电动夹爪，为特定指令。

■ **指令格式：**

```
[变数名称] = EG_GET_STATUS
```

■ **格式/参数说明：**

[变数名称]为 REAL 或 INT 使用者自定义宣告变数，并记录电动夹爪状态变数值。

变数名称用于记录数值，电动夹爪各状态之代号如下：

状态 / 代码	电动夹爪状态	对应状态
0	闲置	Ready
2	夹持	Hold
-1	位置异常	Alarm1
-2	过行程	Alarm2
-3	原点复归异常	Alarm3

备注：目前 HRSS 为单执行序列方式，无法在暂存器中显示出动作中 Busy 讯号。

■ **范例 1：**

```
EG_OPEN(X32)
EG_RESET
EG_RUN_MOVE(25, 20)
$C[3]   = EG_GET_STATUS
EG_RUN_GRIP(C, 13, L, L)
$C[3]   = EG_GET_STATUS
```

■ **范例 1 说明：**

⇨ 宣告 X1 浮点数变数名称，开启与 XEG-32 电动夹爪的连线，重置电动夹爪原点。

⇨ 再以速度 20mm/s，移到绝对位置 25mm 的位置。

⇨ 记录目前电动夹爪状态，当执行完绝对位置移动目前为闲置 Ready 状态，并将状态存入至第 3 个暂存器位置，数值为 0。

⇨ 再向内方向相对位置移动 13mm，低速度，低夹持力量。

- ⇨ 记录目前电动夹爪状态，执行完相对位置夹持动作完成，若已有正确夹持物件条件下，将呈现为夹持 Hold 状态，并将状态存入至第 3 个暂存器位置，数值为 2；若不正确夹持(空夹)到物件条件下，因已完成动作为闲置 Ready 状态，并将状态存入至第 3 个暂存器位置，数值为 0。
- ⇨ PS: 若有位置异常 Alarm1 状态(电动夹爪需跳出红色警示灯)，并将状态存入至第 3 个暂存器位置，数值为-1。

■ **范例 2：**

```
EG_OPEN(X32)
EG_RESET
EG_RUN_MOVE(25, 20)
EG_RUN_GRIP(C, 13, L, L)
IF EG_GET_STATUS == 2 THEN
        PTP P0 FINE=1 Vel=10% Acc=100% TOOL[0] BASE[0]
ENDIF
```

■ **范例 2 说明：**

- ⇨ 宣告 X1 浮点数变数名称，开启与 XEG-32 电动夹爪的连线，重置电动夹爪原点。
- ⇨ 再以速度 20mm/s，移到绝对位置 25mm 的位置。
- ⇨ 再向内方向相对位置移动 13mm，低速度，低夹持力量。
- ⇨ 以如果判断式作判定，确认电动夹爪状态数值为 2 夹持 Hold 状态，就以点对点方式移动至 P0 点位置。

■ **参考相关指令：**

EG_OPEN 、EG_RESET、EG_RUN_MOVE 、EG_RUN_GRIP 、IF。

2.12.5 @EG_RUN_MOVE 电动夹爪的位置移动指令

■ **指令说明：**

XEG 系列电动夹爪绝对位置方式移动指令，使用者自行设定的速度、位置参数，以绝对位置方式移动到使用者指定的绝对位置。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	不预读 FINE

⇨ 手臂透过 USB 通讯方式控制上银科技 XEG 系列的电动夹爪，为特定指令。

■ **指令格式：**

```
EG_RUN_MOVE([绝对位置移动距离],[移动速度])
EG_RUN_MOVE([绝对位置移动距离],[移动速度]) CONT
```

■ **格式/参数说明：**

⇨ [绝对位置移动距离] 为使用者自定义下一个移动绝对位置，以绝对位置方式移动，绝对位置设定的最小移动单位为 0.01mm。

⇨ 各款电动夹爪可设定绝对位置的参数范围如下：

参数种类 / 电动夹爪款式	绝对位置 (mm)	移动速度 (mm/s)
XEG-16	0~16	1~60
XEG-32	0~32	1~80
XEG-64	0~64	1~100
最小调整单位	0.01	1

⇨ [移动速度] 为使用者自定义下一个移动速度参数，移动速度设定的最小速度单位为 1mm/s。

⇨ CONT 为连续轨迹指令，表示程式会直接往下预读执行，不会等电动夹爪重置移动完成，即开始下面程式执行。

■ **范例：**

```
EG_OPEN(X16)
EG_RESET
EG_RUN_MOVE(10, 40)
EG_RUN_MOVE(8.23, 5) CONT
PTP P0 FINE=1 Vel=100% Acc=100% TOOL[0] BASE[0]
```

■ **范例说明：**

⇨ 开启与 XEG-16 电动夹爪的连线。

⇨ 再执行电动夹爪原点重置。

⇨ 再以速度 40mm/s，移到绝对位置 10mm 的位置。

⇨ 再以速度 5mm/s，移到绝对位置 8.23mm 的位置，并使用 CONT 为连续轨迹指令，预读程式，电动夹爪开始移动后，就执行下面那行点到点移动程式。

⇨ 点到点方式移动至 P0 点位。

■ **参考相关指令：**

EG_OPEN、EG_RESET。

2.12.6 @EG_RUN_GRIP 电动夹爪的夹持动作指令

■ **指令说明：**

XEG 系列电动夹爪夹持动作指令，附有夹持力量控制，根据使用者设定的夹持移动方向(Direction)、相对移动夹持距离(Gripping stroke)、夹持速度(Gripping speed)、夹持力量(Gripping force)来移动，比照电动夹爪的介面参数设定。

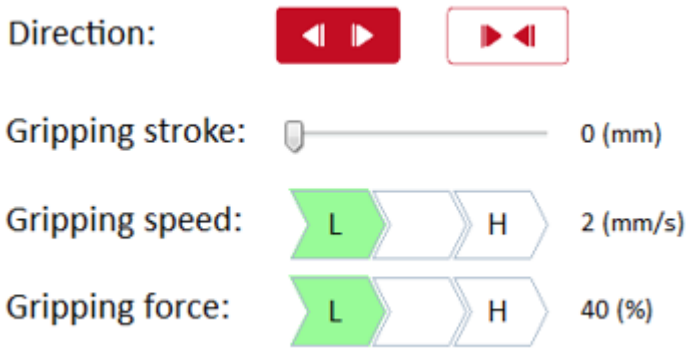


图 45 电动夹爪软体夹持模式介面图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	不预读 FINE

⇨ 手臂透过 USB 通讯方式控制上银科技 XEG 系列的电动夹爪，为特定指令。

■ **指令格式：**

EG_RUN_GRIP([移动方向],[夹持距离],[夹持速度],[夹持力量])
EG_RUN_GRIP([移动方向],[夹持距离],[夹持速度],[夹持力量]) CONT

■ **格式/参数说明：**

各参数组合：

参数种类 / 电爪款式	移动方向	相对位置 夹持距离 (mm)	夹持速度	夹持力量
XEG-16	C O	0 ~ 16	L M H	L M H
XEG-32		0 ~ 32		
XEG-64		0 ~ 64		
最小调整单位	-	1	-	-

备注：高(H) / 中(M) / 低 (L)；开(O) 或 关(C)。

高(H) / 中(M) / 低 (L)对应数值如下

Electric Gripper Type	Grip Speed (mm/s)			Grip Force(%)		
	L	M	H	L	M	H
XEG-16	2	5	10	50	75	100
XEG-32	2	5	10	40	70	100
XEG-64	2	8	15	40	70	100

⇨ [移动方向]与对应的代码如下:

移动方向	代码
向内移动	C
向外移动	O

⇨ 各款电动夹爪可设定的相对位置[夹持距离]参数范围如下:

参数种类 / 电动夹爪款式	绝对位置 (mm)
XEG-16	0~16
XEG-32	0~32
XEG-64	0~64

※夹持距离的最小移动单位为 1mm

⇨ [夹持速度]可设定的代码如下 :

夹持速度	代码
低	L
中	M
高	H

⇨ [夹持力量]可设定的代码如下 :

夹持力量	代码
低比率	L
中比率	M
高比率	H

⇨ CONT 为连续轨迹指令，表示程式会直接往下预读执行，不会等电动夹爪重置移动完成，即开始下面程式执行。

■ **范例：**

```
EG_OPEN(X16)
EG_RESET
EG_RUN_MOVE(10, 40)
EG_RUN_GRIP(C, 3, L, H)
EG_RUN_GRIP(O, 5, M, L) CONT
PTP P0 FINE=1 Vel=100% Acc=100% TOOL[0] BASE[0]
```

■ **范例说明：**

- ⇨ 开启与 XEG-16 电动夹爪的连线。
- ⇨ 再执行电动夹爪原点重置。
- ⇨ 再以速度 40mm/s，移到绝对位置 10mm 的位置。
- ⇨ 再向内方向相对移动 3mm，低速度，高夹持力量。
- ⇨ 再向外移动 5mm，中速，低夹持力量，并使用 CONT 为连续轨迹指令，预读程式，电动夹爪开始移动后，就执行下面那行点到点移动程式。
- ⇨ 点到点方式移动至 P0 点位。

■ **参考相关指令：**

EG_OPEN、EG_RESET、EG_RUN_MOVE。

2.12.7 @EG_RUN_EXPERT 电动夹爪的专家模式

■ **指令说明：**

执行 XEG 系列电动夹爪的专家模式动作组合，根据使用者设定的夹持移动方向(Direction)、相对位置移动距离(Move stroke)、移动速度(Move speed)、相对移动夹持距离(Gripping stroke)、夹持速度(Gripping speed)、夹持力量(Gripping force)来移动，比照电动夹爪的介面参数设定，专业模式下精确控制六项参数组合指令。

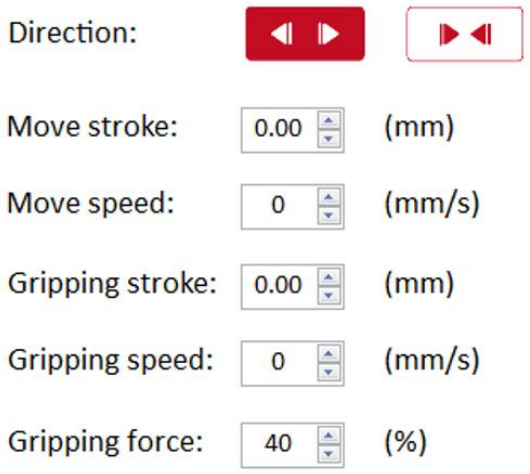


图 46 电动夹爪软体专家模式介面图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	不预读 FINE

⇨ 手臂透过 USB 通讯方式控制上银科技 XEG 系列的电动夹爪，为特定指令。

■ **指令格式：**

EG_RUN_EXPERT([移动方向],[相对位置移动距离],[相对位置移动速度],[夹持距离],[夹持速度],[夹持力量])

EG_RUN_EXPERT([移动方向],[相对位置移动距离],[相对位置移动速度],[夹持距离],[夹持速度],[夹持力量]) CONT

■ **格式/参数说明：**

⇨ 各款电动夹爪可设定的夹持距离参数范围如下:

参数种类 / 电爪款式	移动方向	相对位置移动距离 (mm)	相对位置移动速度 (mm/s)	夹持距离 (mm)	夹持速度 (mm/s)	夹持力量 (%)
XEG-16	C O	0~16	1~60	0~16	1~20	50~100
XEG-32		0~32	1~80	0~32		40~100
XEG-64		0~64	1~100	0~64		
最小调整单位	-	0.01	1	0.01	1	5

⇨ [移动方向]与对应的代码如下:

移动方向	代码
向内移动	C
向外移动	O

⇨ 各款电动夹爪可设定的[相对位置移动距离]、[相对位置移动速度]、[夹持距离]参数范围如下:

参数种类 / 电动夹爪款式	相对位置移动距离 (mm)	相对位置移动速度 (mm/s)	夹持距离 (mm)
XEG-16	0~16	1~60	0~16
XEG-32	0~32	1~80	0~32
XEG-64	0~64	1~100	0~64

※[相对位置移动距离]、[夹持距离]的最小移动单位为 0.01mm。

※[相对位置移动速度]的最小移动单位为 1mm/s。

⇨ [夹持速度]可设定的 0~20 mm/s，最小移动单位为 1mm/s。

⇨ [夹持力量] XEG-16 可设定的 50~100%；XEG-32、XEG-64 可设定的 40~100%，最小力量单位为 5%。

⇨ CONT 为连续轨迹指令，表示程式会直接往下预读执行，不会等电动夹爪重置移动完成，即开始下面程式执行。

■ **范例：**

```
EG_OPEN(X16)
EG_RESET
EG_RUN_MOVE(10, 40)
EG_RUN_EXPERT(C, 8, 40, 3, 10, 85)
```

■ **范例说明：**

⇨ 开启与 XEG-16 电动夹爪的连线。

⇨ 再执行电动夹爪原点重置。

⇨ 再以速度 40mm/s，移到绝对位置 10mm 的位置。

⇨ 电动夹爪向内移动到相对位置 8mm 的位置后，使用速度 40mm/s；再向内移动 3mm，使用 10mm/s 的速度、85%力道。

■ **参考相关指令：**

EG_OPEN、EG_RESET、EG_RUN_MOVE、EG_GRIP。

2.12.8 @EG_GET_POS 电动夹爪的当前位置取得指令

■ **指令说明：**

读取 XEG 系列电动夹爪当前所在的位置资讯指令，通常取得位置资讯后可用于程式当为判断条件。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.2.1 以上	Caterpillar1.0.1 以上	不预读 FINE

- 手臂透过 USB 通讯方式控制上银科技 XEG 系列的电动夹爪，为特定指令。

■ **指令格式：**

[电动夹爪位置变数名称] = **EG_GET_POS**

■ **格式/参数说明：**

⇨ [电动夹爪位置变数名称]为 REAL 或 INT 使用者自定义宣告变数。

⇨ **EG_GET_POS** 执行此程式码，即取得目前电动夹爪位置值，后续可用来运算使用。

⇨ 读取位置数值到小数后两位，单位为公厘 mm。

■ **范例 1：**

```
REAL X1
EG_OPEN(X16)
EG_RUN_MOVE(5.43, 40)
X1 = EG_GET_POS
$C[3] = X1*100
```

■ **范例 1 说明：**

⇨ 宣告 X1 浮点数变数名称，开启与 XEG-16 电爪的连线

⇨ 再以速度 40mm/s，移到绝对位置 10mm 的位置。

⇨ 将当下电动夹爪位置记录至 X1 变数值。

⇨ 将 X1 变数值乘 100 后，存入第 3 个计数暂存器，使用者可以在计数暂存器介面看到现在电动夹爪位置数值(计数暂存器只能存入整数，故将浮点数乘上 100)。

■ **范例 2：**

```
EG_OPEN(X16)
IF EG_GET_POS < 7 THEN
    PTP P0 FINE=1 Vel=100% Acc=100% TOOL[0] BASE[0]
    EG_RUN_MOVE(15, 40)
ENDIF
```

■ **范例 2 说明：**

⇨ 开启与 XEG-16 电爪的连线

⇨ 当电动夹爪当下所在位置小于 7 时，手臂移动至 P0 点(原点)位置。

⇨ 再以速度 40mm/s，移到绝对位置 15mm 的位置 作打开夹爪准备抓取状态。

■ **参考相关指令：**

EG_OPEN、IF。

2.13 代码 M、外部轴功能指令

此章节针对于代码 M 的外部轴功能指令做各别说明

2.13.1 @EX_AX 外部轴的开启/关闭连接指令

■ **指令说明：**

启用或关闭已连接的外部轴，可利用此指令启用对应的外部轴，或关闭对应的外部轴。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.1 以上	Caterpillar1.0.1 以上	不预读 FINE

⇨ 使用前需先设定外部轴相关参数，可由主功能选单->Start-up->(Next)->External Axis->Setting 进入设定页面，勾选”Enable”后可开始设定相关参数，根据所连接的外部轴形式(直线轴/旋转轴)设定相关参数。

⇨ 直线型式(Linear)外部轴的设定页面如图 47 所示，参数说明如表 8 所示。

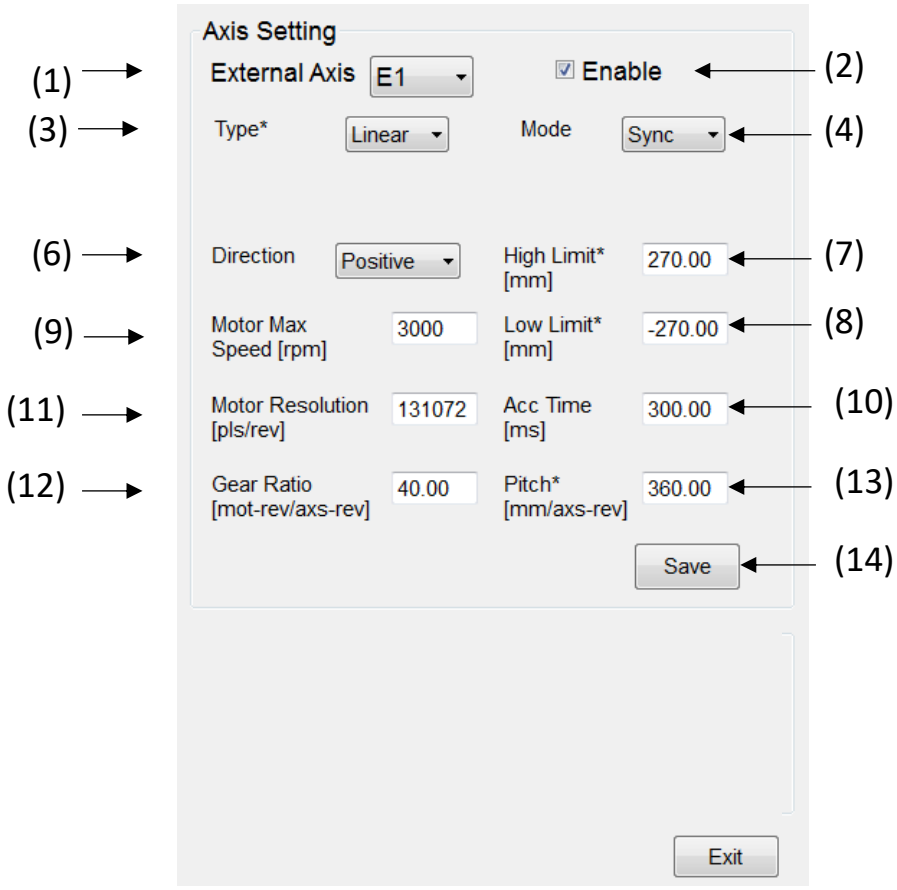


图 47 “Linear”外部轴设定页面图

表 8 外部轴形式(直线轴/旋转轴)设定相关参数表

项次	说明
1	最大可控制 3 组外部轴，选定 E1~E3 设定相关对应参数
2	启用/停用所选取的外部轴
3	设定外部轴的型式，可为直线轴(Linear)/旋转轴(Rotary)
4	设定外部轴的运动模式: **(1)同步运动:** 表示包括机器人及外部轴的运动为同时开始且同时结束。 **(2)非同步运动:** 表示机器人与外部轴的运动非同时开始或非同时结束，机器人与外部轴彼此的运动不具相依性。
6	设定外部轴马达的方向为正转(Positive)/反转(Negative)
7	设定外部轴运动范围的上极限
8	设定外部轴运动范围的下极限
9	设定外部轴马达的最大转速
10	设定外部轴马达的最大加速时间
11	设定外部轴马达的编码器解析度
12	设定外部轴马达配置减速机的减速比
13	设定外部轴的导程(仅对应直线轴)
14	储存设定值

⇨ 旋转型式(Rotary)外部轴的设定页面如图 48 图 47 所示，参数说明如表 9 所示。

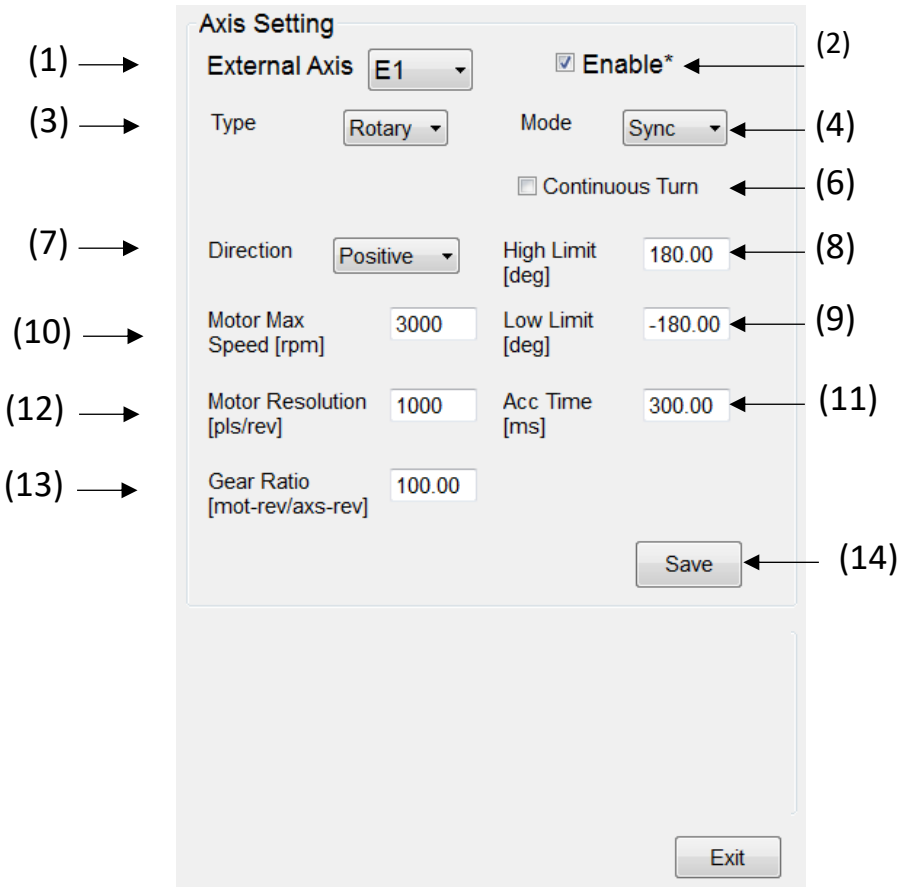


图 48 “Rotary”外部轴设定页面图

表 9 “Rotary”外部轴设定页面参数表

项次	说明
1	最大可控制 3 组外部轴，选定 E1~E3 设定相关对应参数
2	启用/停用所选取的外部轴
3	设定外部轴的型式，可为直线轴(Linear)/旋转轴(Rotary)
4	设定外部轴的运动模式: ⇨ **同步运动:** 表示包括机器人及外部轴的运动为同时开始且同时结束。 ⇨ **非同步运动:** 表示机器人与外部轴的运动非同时开始或非同时结束，机器人与外部轴彼此的运动不具相依性。
6	设定外部轴为连续旋转，当设定为连续旋转时，则运动范围没有上下限的限制(仅对应旋转轴)
7	设定外部轴马达的方向为正转(Positive)/反转(Negative)
8	设定外部轴运动范围的上极限
9	设定外部轴运动范围的下极限
10	设定外部轴马达的最大转速
11	设定外部轴马达的最大加速时间
12	设定外部轴马达的编码器解析度
13	设定外部轴马达配置减速机的减速比
14	储存设定值

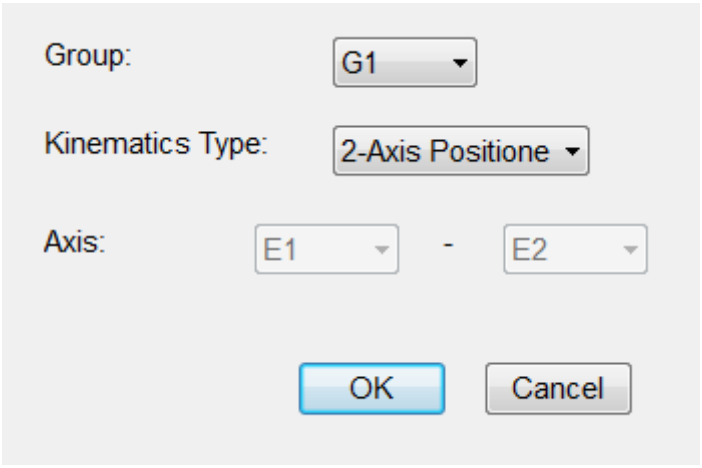


图 49 外部轴群组设定页面图(目前只对应 2 轴定位器)

E2

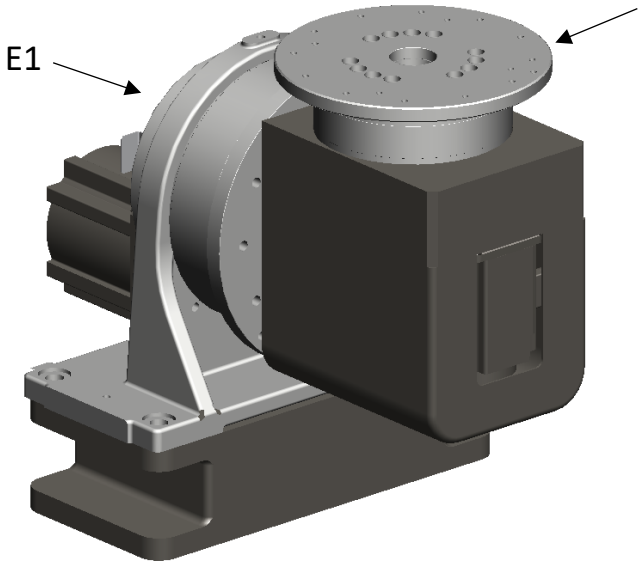


图 50　2 轴定位器示意图

■ **指令格式：**

EX_AX[外部轴编号] = [布林变数]

■ **格式/参数说明：**

⇨ [外部轴编号] 为外部轴编号，正整数值，可输入范围为 1 至 3 之间。

⇨ [布林变数] 为真 TRUE 表示启动或假 FALSE 表示关闭，布林逻辑变数，二者择一状态。

■ **范例 1：**

EX_AX[1] = TRUE

■ **范例 1 说明：**

启用第 1 组外部轴使用。

■ **范例 2：**

EX_AX[2] = FALSE

■ **范例 2 说明：**

关闭第 2 组外部轴使用。

■ **参考相关指令：**

EX_AX_ASYNC, EX_AX_SET_ACC, EX_AX_SET_SPEED, EX_AX_CT, ASYPTP。

2.13.2 @EX_AX_ASYNC 外部轴的设定同步轴/非同步轴指令

■ **指令说明：**

设定指定的外部轴为同步运动或非同步运动，当设定为同步运动时，机器人与外部轴的运动会同时开始，并同时到达目标位置，若设定为非同步运动时，则机器人与外部轴的运动不具有相依性，会分别各自到达目标位置

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.1 以上	Caterpillar1.0.1 以上	预读 CONT

使用前需先设定外部轴相关参数，可由主功能选单->Start-up->(Next)->External Axis->Setting 进入设定页面(参照 EX_AX 的使用说明)，勾选”Enable”后可开始设定相关参数，根据所连接的外部轴设定相关参数。

■ **指令格式：**

```
EX_AX_ASYNC[外部轴编号] = [布林变数]
```

■ **格式/参数说明：**

⇨ [外部轴编号] 为外部轴编号值，可输入范围数值为 1~3，表示第 1~3 编号外部轴。

⇨ [布林变数] 为真 TRUE 或假 FALSE，布林逻辑变数，二者择一状态。 TRUE 为设定为非同步运动；FALSE 为设定为同步运动。

■ **范例 1：**

```
EX_AX_ASYNC[1] = TRUE
```

■ **范例 1 说明：**

设定第 1 组外部轴为非同步运动。

■ **范例 2：**

```
EX_AX_ASYNC[2] = FALSE
```

■ **范例 2 说明：**

设定第 2 组外部轴为同步运动。

■ **参考相关指令：**

EX_AX, EX_AX_SET_ACC, EX_AX_SET_SPEED, EX_AX_CT, ASYPTP。

2.13.3 @ EX_AX_SET_ACC 外部轴的设定加/减速度指令

■ **指令说明：**

设定外部轴的加减速时间，可使用此指令修改指定外部轴的加减速度

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.1 以上	Caterpillar1.0.1 以上	预读 CONT

使用前需先设定外部轴相关参数，可由主功能选单->Start-up->(Next)->External Axis->Setting 进入设定页面(参照 EX_AX 的使用说明)。

■ **指令格式：**

EX_AX_SET_ACC[外部轴编号] [加减速时间值]

■ **格式/参数说明：**

⇨ [外部轴编号]为外部轴编号值，可输入范围数值为 1~3，表示第 1~3 编号外部轴。

⇨ [加减速时间值] 为加减速时间，单位为毫秒 msec(ms)，代表 0.001 秒，

■ **范例 1：**

EX_AX_SET_ACC[1] 300

■ **范例 1 说明：**

设定第 1 组外部轴的加减速时间为 300 毫秒 ms。

■ **参考相关指令：**

EX_AX, EX_AX_ASYNC, EX_AX_SET_SPEED, EX_AX_CT, ASYPTP。

2.13.4 @EX_AX_SET_SPEED 外部轴的设定速度指令

■ **指令说明：**

设定指定外部轴一般运动的进给速度，线性轴单位为 mm/sec，旋转轴单位为 deg/sec。在程式中使用此指令，不会改变 PTP 与无限旋转指令的速度设定。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.1 以上	Caterpillar1.0.1 以上	预读 CONT

使用前需先设定外部轴相关参数，可由主功能选单->Start-up->(Next)->External Axis->Setting 进入设定页面(参照 EX_AX 的使用说明)。

■ **指令格式：**

EX_AX_SET_SPEED[外部轴编号] [速度参数值]

■ **格式/参数说明：**

⇨ [外部轴编号]为外部轴编号值，可输入范围数值为 1~3，表示第 1~3 编号外部轴。

⇨ [速度参数值] 为速度值，单位为毫米/秒(mm/s) 或 角度/秒(deg/s)。

■ **范例：**

EX_AX_SET_SPEED[1] 1000

■ **范例说明：**

设定第 1 组外部轴(E1)的速度上限为 1000 毫米/秒(mm/s)。

■ **参考相关指令：**

EX_AX, EX_AX_ASYNC, EX_AX_SET_ACC, EX_AX_CT, ASYPTP。

2.13.5 @EX_AX_CT 外部轴的无限旋转速率指令

■ **指令说明：**

指定的外部轴依照设定的速率往同一方向进行连续旋转运动，本功能可对应外部轴应用于输送带或砂轮机等需要连续运转的需求中(当使用连续运转的情况下，将不执行运动命令中外部轴的到位检查)。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.1 以上	Caterpillar1.0.1 以上	预读 CONT

使用前需先设定外部轴相关参数，可由主功能选单->Start-up->(Next)->External Axis->Setting 进入设定页面(参照 EX_AX 的使用说明) ，且需于"连续旋转"启用的情况下可使用，此指令仅对应"Rotary"类型的外部轴，其他待软体改进中。

■ **指令格式：**

EX_AX_CT[外部轴编号] [旋转速率值]

■ **格式/参数说明：**

⇨ [外部轴编号]为外部轴编号值，可输入范围数值为 1~3，表示第 1~3 编号外部轴。

⇨ [旋转速率值] 为欲旋转速率值(VEL_RATIO)，可输入范围数值为-100(%)~100(%)，当[旋转速率值]大于 0 时往正方向运转，当[旋转速率值] 小于 0 时往反方向运转，当[旋转速率值] 等于 0 时强制停止外部轴运动。

■ **范例 1：**

```
EX_AX_CT[1] 80
PTP P0 CONT Vel=100% Acc=100% TOOL[0] BASE[0]
PTP P1 CONT Vel=100% Acc=100% TOOL[0] BASE[0]
EX_AX_CT[1] 0
```

■ **范例 1 说明：**

设定第 1 组外部轴以 80%的速率正向连续运转，手臂由 P0 运动到 P1 后终止连续旋转

■ **范例 2：**

```
EX_AX_CT[2] -100
LIN P0 CONT Vel=600mm/s Acc=100% TOOL[0] BASE[0]
LIN P1 FINE=1 Vel=600mm/s Acc=100% TOOL[0] BASE[0]
```

■ **范例 2 说明：**

设定第 2 组外部轴以 100%的速率反向连续运转，当手臂由 P0 运动到 P1 后终止连续旋转。

■ **参考相关指令：**

EX_AX, EX_AX_ASYNC, EX_AX_SET_ACC, EX_AX_SET_SPEED。

2.13.6 @ASYPTP 非同步运动外部轴点到点运动

■ **指令说明：**

当外部轴设定为非同步运动时，可使用此指令使外部轴执行点到点的运动

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.1 以上	Caterpillar1.0.1 以上	预读 CONT

使用前需先设定外部轴相关参数，可由主功能选单->Start-up->(Next)->External Axis->Setting 进入设定页面(参照 EX_AX 的使用说明)，并将对应的外部轴设定为非同步运动。

■ **指令格式：**

```
ASYPTP { [外部轴增量参数]          } [轨迹形式       ] [速度  ] [加速度]
ASYPTP { [E1 num, E2 num, E3 num] } [FINE=#/CONT=#] [Vel=#] [Acc =#]
```

■ **格式/参数说明：**

⇨ #为数字，由使用者填入。

⇨ [各轴增量参数] 为增量参数数值，可选择填入 E1、E2、E3 轴角度值，单位为度，外轴增量参数内容：[E1 num, E2 num, E3 num]，num 表示数值。

⇨ [轨迹形式]为连续 CONT 或非连续 FINE 类型， 填入参数 CONT、FINE；并可在连续轨迹形式后面的百分比(1%~100%)为平滑的程度，决定连续参数；非连续型式 FINE 分成三个等级:。

(1)FINE (=0)，会预读接续的指令，不会执行到位检查。

(2)FINE = 1 ，不会预读接续的指令，不会执行到位检查。

(3)FINE = 2 ，不会预读接续的指令，会执行到位检查。

(4)CONT (=100%) ，可依百分比(%)或转角半径长度(mm)决定平滑的程度。例如：CONT=10%、CONT=3mm

⇨ [速度] Vel 为单行速度参数，#填入参数 Vel=1%~100%，数字越大表示速度越快

⇨ [加速度] Acc 为单行加速度参数，#填入参数 Acc=1~100%，数字越大表示加速度越快

● Vel、Acc 可省略输入。省略时将参考上一行指令数值，或初始定义参数。

■ **范例 1：**

```
ASYPTP {E1 20} FINE Vel=100% Acc=100%
```

■ **范例 1 说明：**

第 1 组外部轴以非平滑运动到达角度 20 的位置，速度 100%，加速度 100%。

■ **范例 2：**

```
ASYPTP {E1 60, E2 120} CONT Vel=100% Acc=100%
```

■ **范例 2 说明：**

第 1 组外部轴与第 2 组外部轴分别以平滑运动到达角度 60 与 120 的位置，速度 100%，加速度 100%。

■ **参考相关指令：**

EX_AX, EX_AX_ASYNC, EX_AX_SET_SPEED, EX_AX_SET_ACC。

2.14 代码 N、焊接功能指令

此章节针对于代码 N 的焊接功能指令做各别说明

2.14.1 @WELD_START 启动焊接程序指令

■ **指令说明：**

启动焊接程序，依照指定的焊接程序与焊接条件执行焊接动作，此指令用以告知机器手何时开始以什么条件执行焊接任务

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.1 以上	Caterpillar 不适用	不预读 FINE

使用前需先与焊接电源(Weld Power Source)连接，并由主功能选单->Start-up->(Next)->Arc Weld->Weld Setting 进入设定页面设定焊机相关参数，并由主功能选单->Start-up->(Next)->Arc Weld->Weld Procedure 中设定焊接程序与焊接条件

⇨ 焊接程序与焊接条件设定页面如图 51 所示，参数说明如表 10 表 11 所示。

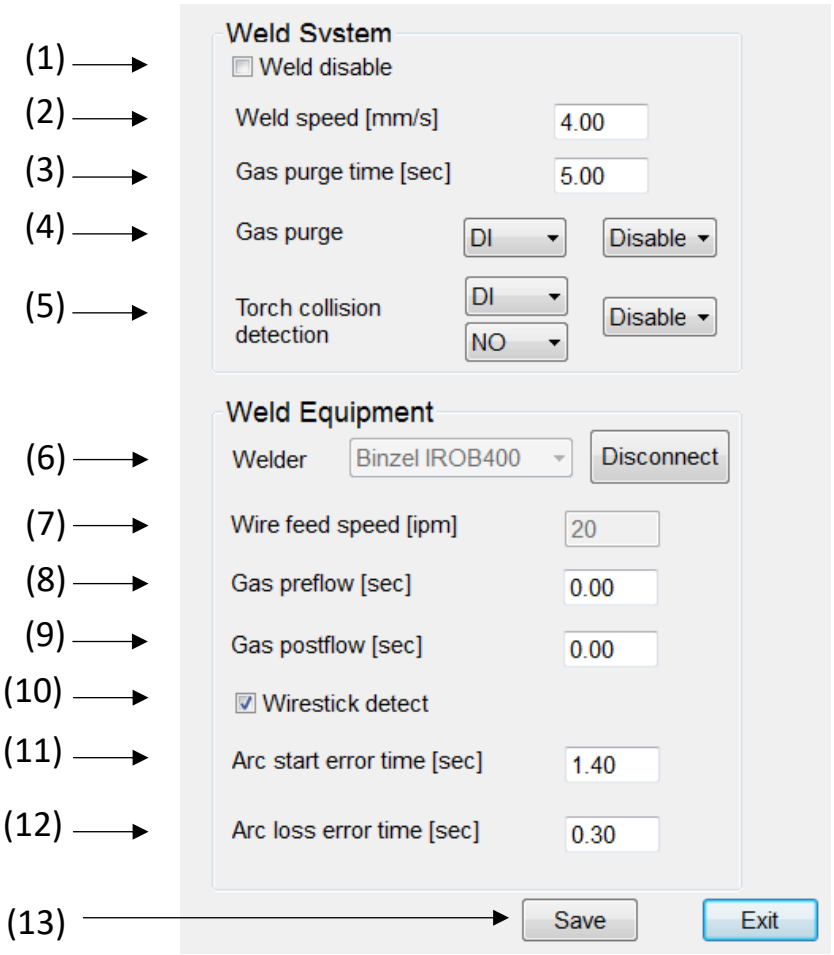


图 51 焊接系统及装置设定图

表 10 焊接系统及装置设定参数表

项次	说明
1	起弧功能关闭，勾选后于程式运行时会跳过起弧的动作，用于确认焊接示教路径
2	焊接速度设定，需搭配 WELD_SPEED 使用 EX: LIN P1 FINE=1 WELD_SPEED Acc=100%
3	焊枪气洗时间设定
4	焊枪气洗输入讯号的设定，可设为 DI 或 SI
5	焊枪撞枪讯号设定，可设为 DI 或 SI，并可针对感测器的类型设为常开讯号(NO)或常闭讯号(NC)
6	焊接电源设定，选择焊机并进行连线 目前对应 Lincoln S350 与 Binzel IROB400
7	送丝速度设定(对应 Lincoln S350)
8	焊接前送气时间设定(对应 Lincoln S350)
9	焊接后送气时间设定(对应 Lincoln S350)
10	焊丝溶敷检测启用(对应 Lincoln S350)
11	起弧失败检测时间设定，于设定时间内没有检测到电弧的产生则判定为起弧失败(对应 Lincoln S350)
12	电弧消失检测时间，于焊接过程终，电弧消失的时间超出设定时间时判定为电弧消失(对应 Lincoln S350)
13	储存设定

⇨ Lincoln 焊接程序与焊接条件设定页面图 52 如所示，参数说明如表 11 所示。

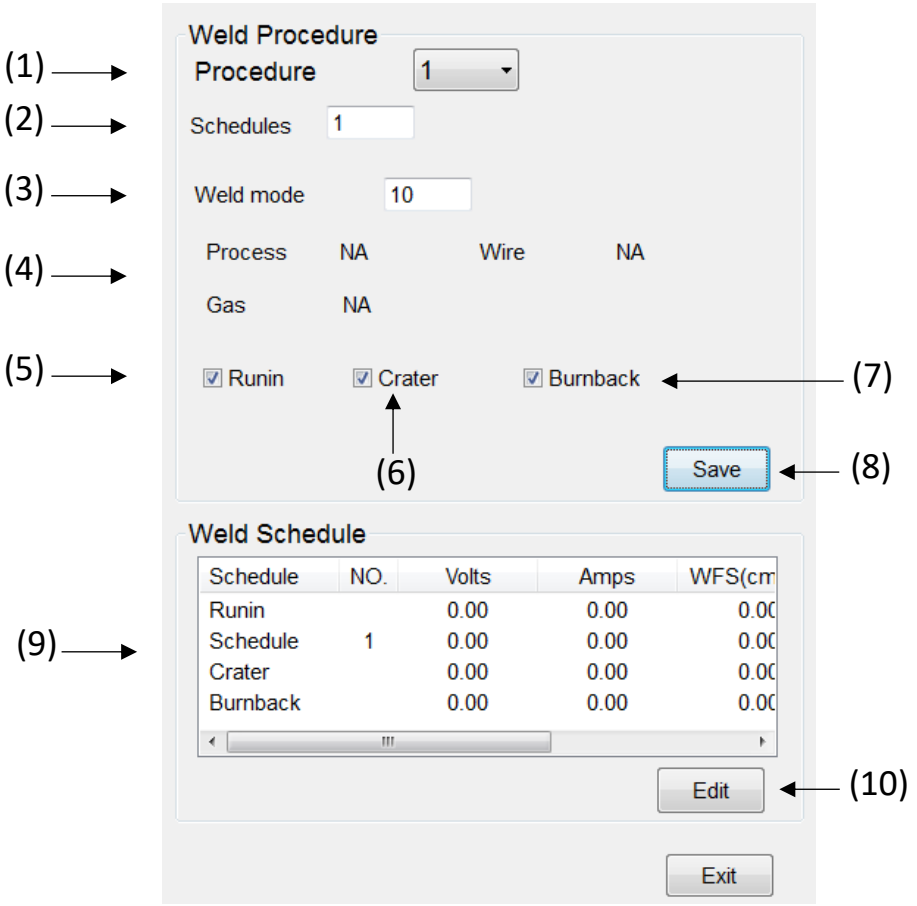


图 52 Lincoln 焊接程序与焊接条件设定图

表 11　Lincoln 焊接程序与焊接条件设定参数表

项次	说明
1	焊接程序
2	焊接程序中的焊接条件数量
3	焊接模式设定
4	选定焊接模式所对应的金属种类、焊丝规格与适用保护气体
5	启动处理：为了使焊接开始时可以顺畅进行，可于启动处理时设定高于焊接时的焊接条件(勾选后启用)
6	弧坑处理，用于焊接结束时在一定时间内指定较低的电压与电流，避免急剧下降的电压造成弧坑孔
7	回烧处理，用于在送丝结束后透过在适当时间施加较高电压来防止焊丝与工件发生溶敷
8	储存焊接程序设定
9	焊接条件参数，一般包括电压、电流、电弧长度、送丝速度及执行时间等参数
10	编辑焊接条件参数

⇨ Binzel 焊接程序与焊接条件设定页面图 53 如所示，参数说明如表 12 所示。

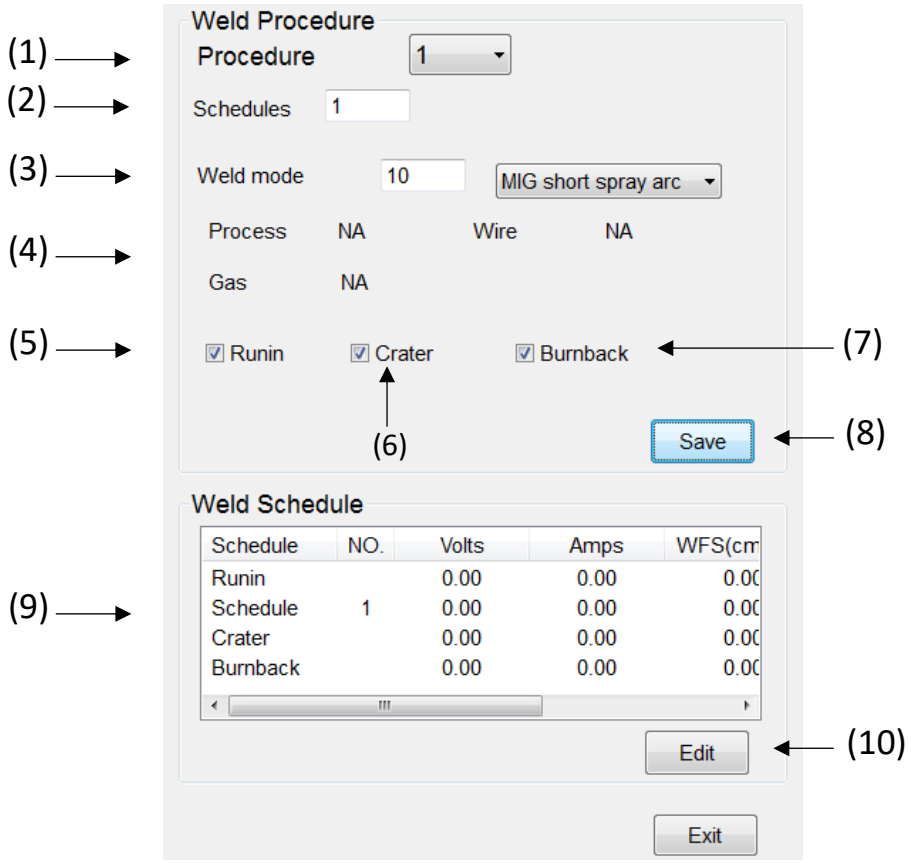


图 53 Binzel 焊接程序与焊接条件设定图

表 12　Binzel 焊接程序与焊接条件设定参数表

项次	说明
1	焊接程序
2	焊接程序中的焊接条件数量
3	焊接模式设定: MIG short/spray arc: 短路电弧/射流电弧 Pulse MIG: 脉冲电弧 Saved jobs: 自定义焊接程式
4	选定焊接模式所对应的金属种类、焊丝规格与适用保护气体 (Saved jobs 不对应)
5	仅对应 Lincoln S350
6	仅对应 Lincoln S350
7	仅对应 Lincoln S350
8	储存焊接程序设定
9	仅对应 Lincoln S350
10	仅对应 Lincoln S350

■ **指令格式：**

WELD_START WP = [焊接程序编号] WS = [焊接条件编号]

■ **格式/参数说明：**

⇨ [焊接程序编号] 为使用的焊接程序编号，对应数值为 1 至 20 之间。

⇨ [焊接条件编号] 为使用的焊接条件编号，对应数值为 1 至 32 之间。

■ **范例 1：**

```
LIN P1 FINE=1 Vel=600mm/s Acc=100% TOOL[0] BASE[0]
WELD_START WP=1 WS=2                                    (开始焊接任务)
LIN P2 CONT Vel=5mm/s Acc=100% TOOL[0] BASE[0] (使用焊接程序 1 中的焊接条件 2 焊接)
WELD_START WP=1 WS=3
LIN P3 FINE=1 Vel=5mm/s Acc=100% TOOL[0] BASE[0] (使用焊接程序 1 中的焊接条件 3 焊接)
WELD_END WP=1 WS=2          (使用焊接程序 1 中的焊接条件 2 进行填坑并结束焊接)
```

■ **范例 1 说明：**

移动至 P1 点以焊接程序 1 中的焊接条件 2 进行焊接，移动经过 P2 点时切换为焊接程序 1 中的焊接条件 3 进行焊接，到 P3 点后结束焊接程序

* 焊接起始点运动指令必须使用 FINE=1(2)
* 焊接过程点运动指令必须使用 CONT
* 焊接结束点运动指令必须使用 FINE=1(2)
* 焊接过程点位与结束点位请勿使用 PTP 运动

(焊接结束指令，参照 WELD_END 说明)

■ **参考相关指令：**

WELD_END, WEAVE, WEAVE_END。

2.14.2 @WELD_END 结束焊接程序指令

■ **指令说明：**

结束焊接程序，依照指定的焊接程序与焊接条件执行填坑处理并结束焊接动作

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.1 以上	Caterpillar 不适用	不预读 FINE

使用前需先与焊接电源(Weld Power Source)连接，并由主功能选单->Start-up->(Next)->Arc Weld->Weld Setting 进入设定页面设定焊机相关参数，并由主功能选单->Start-up->(Next)->Arc Weld->Weld Procedure 中设定焊接程序与焊接条件(参照 WELD_START 的使用说明)

■ **指令格式：**

WELD_END WP = [焊接程序编号] WS = [焊接条件编号]

■ **格式/参数说明：**

⇨ [焊接程序编号] 为使用的焊接程序编号，对应数值为 1 至 20 之间。

⇨ [焊接条件编号] 为使用的焊接条件编号，对应数值为 1 至 32 之间。

■ **范例：**

```
LIN P1 FINE=1 Vel=600mm/s Acc=100% TOOL[0] BASE[0]
WELD_START WP=2 WS=1                                    (开始焊接任务)
LIN P2 CONT Vel=5mm/s Acc=100% TOOL[0] BASE[0]
LIN P3 FINE=1 Vel=5mm/s Acc=100% TOOL[0] BASE[0]        (使用焊接程序 1 中的焊接條件 2 焊接)
WELD_END WP=2 WS=1          (使用焊接程序 2 中的焊接条件 1 进行填坑并结束焊接)
```

■ **范例说明：**

移动至 P1 点以焊接程序 2 中的焊接条件 1 进行焊接，移动经过 P2 到 P3 点后以焊接程序 2 中的焊接条件 1 进行填坑处理并结束焊接

(焊接开始指令，参照 WELD_START 说明)

■ **参考相关指令：**

WELD_START, WEAVE, WEAVE_END。

2.14.3 @WEAVE 焊接的设定摆焊动作指令

2.14.4 @WEAVE_END 结束焊接的摆焊动作指令

■ **指令说明：**

适用软体版本		指令预设状态
HRSS3.3.1 以上	Caterpillar 不适用	预读 CONT

摆焊运动是于焊接过程中使焊枪依据特定左右摆动运动来执行焊接任务，藉由增加焊道宽度的方式来增加焊皆强度。此指令使机器手依照预先设定好的摆焊条件进行摆焊运动，或依照指定的频率、振幅、左停留、右停留…等条件进行摆焊运动。

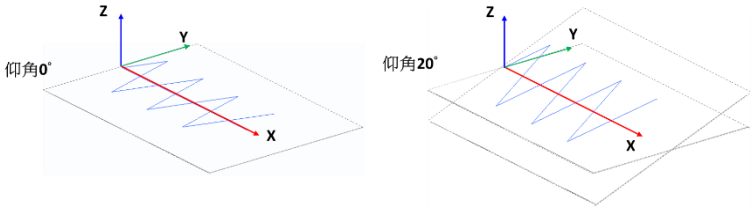
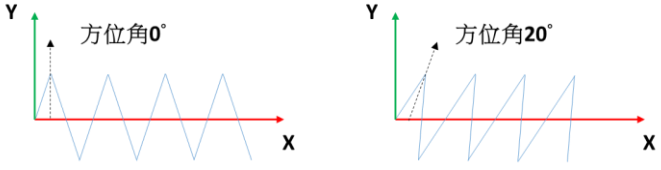
■ **前提条件：**

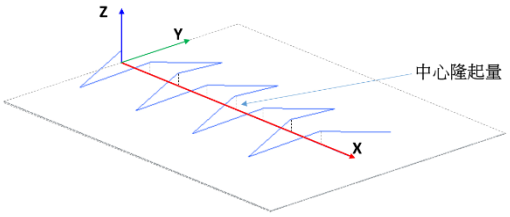
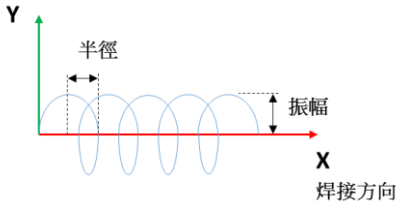
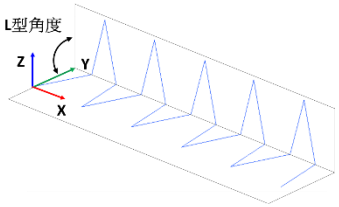
使用前需先由主功能选单->Start-up->(Next)->Arc Weld->Weaving 进入页面设定摆焊相关参数。在摆焊设定(Weave Setting)介面中，停留延迟类型(Dwell delay type)、摆焊连接(Blend weave end)、端点输出(Peak output port, Peak output pulse, Peak output shift)等相关设定，会套用至所有的摆焊运动中，而于摆焊设定介面中设定的仰角、方位角、中心隆起量与半径等参数只在使用(**WEAVE** PAT **Freq= # Hz Amp= # mm LD= # sec RD = # sec)**此型式的指令时才会被套用。而当在使用(**WEAVE** PAT **WS=** SCH_NUM)型式的指令时，则会套用下方摆焊条件(Weave Schedule)中的设定。



图 54 摆焊设定页面图

表 13 摆焊设定页面参数表

项次	说明
1	设定停留延迟类型: 停止(Stop): 在摆焊的过程中于两端点处完全停止 移动(Move): 在摆焊的两端点处仅停止摆焊动作，但手臂于依然会往焊接行进方向动作 (停止时间于摆焊条件设定的"左停留"与"右停留"中设定)
2	设定仰角，相对摆焊座标系使摆动角倾斜指定角度 
3	设定方位角，在摆焊平面上摆焊方位角的倾斜度 
4	设定中心隆起量，指定摆焊中心处焊炬的隆起量，只有在正弦摆焊才会套用

	
5	设定半径，设置圆型摆焊相对于焊接方向的振幅 
6	设定 L 角度，于 L 型摆焊中摆焊左右与平面所成角度 
7	设定摆焊连接，可将动作指令的教导点前后的摆焊轨迹平顺连接
8	设定端点输出 DO，在摆焊过程中，焊炬至端点时，输出指定 DO 讯号
9	设定端点输出 DO 的脉冲宽度
10	设定端点输出 DO 的延迟时间
11	储存摆焊设定
12	摆焊条件列表 对应(**WEAVE** PAT **WS=** SCH_NUM)型指令
13	编辑指定摆焊条件 对应(**WEAVE** PAT **WS=** SCH_NUM)型指令 (点选「编辑 Edit」按钮，参考图 55)

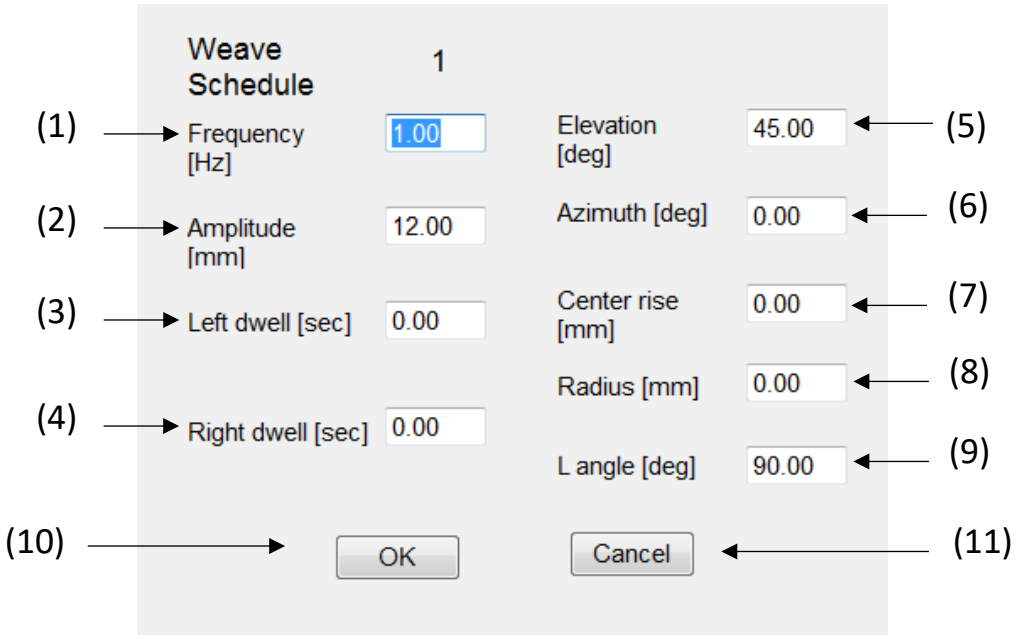
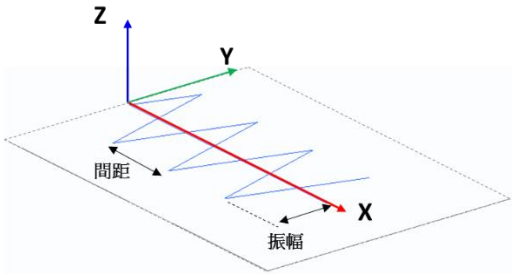
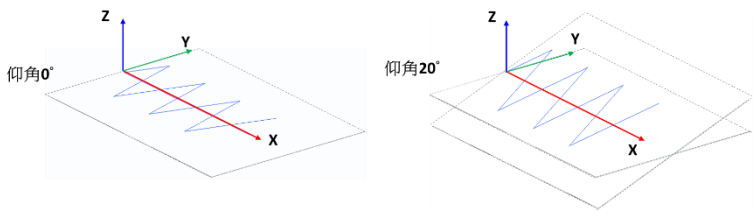
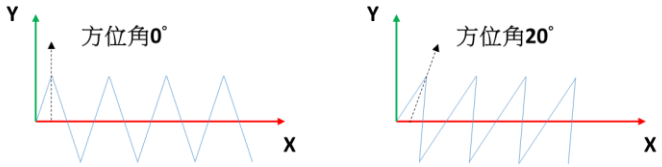
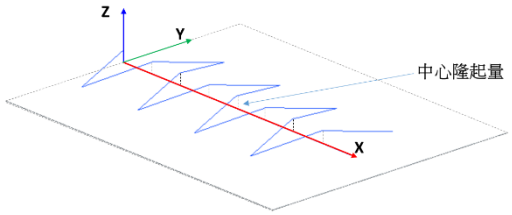
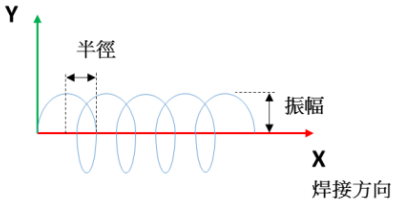
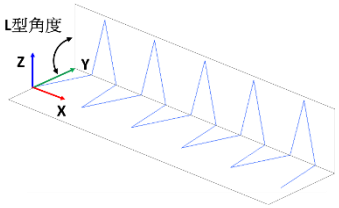


图 55 摆焊条件设定页面图

表 14 摆焊条件设定页面参数表

项次	说明
1	设定频率，指定摆焊每秒中的循环数
2	设定振福，指定焊接生产线到端点的距离 
3	设定左侧停留时间，指定在摆焊左端点的停止时间，停留延迟类型为移动时，机器人向焊接方向移动 (圆型摆焊时本设置无效)
4	设定右侧停留时间，指定在摆焊右端点的停止时间，停留延迟类型为移动时，机器人向焊接方向移动 (圆型摆焊时本设置无效)
5	设定仰角，相对摆焊座标系使摆动角倾斜指定角度 
6	设定方位角，在摆焊平面上摆焊方位角的倾斜度

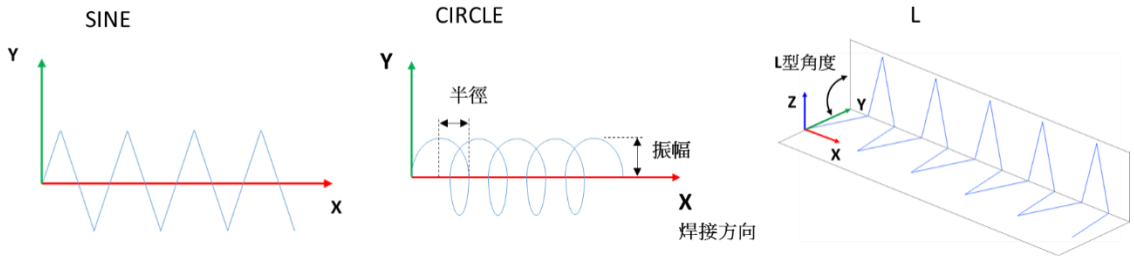
	
7	设定中心隆起量，指定摆焊中心处焊炬的隆起量，只有在正弦摆焊才会套用 
8	设定半径，设置圆型摆焊相对于焊接方向的振福值 
9	设定 L 角度，于 L 型摆焊中摆焊左右与平面所成角度 
10	确认并储存条件设定
11	取消条件设定

■ **指令格式：**

```
WEAVE_START
WEAVE PAT WS = SCH_NUM
WEAVE PAT Freq = # Hz Amp= # mm LD = # sec RD= # sec
WEAVE_END
```

■ **格式/参数说明：**

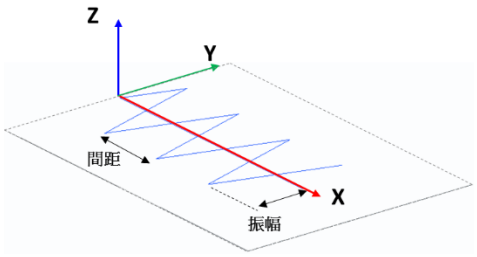
⇨ PAT: 摆焊模式，对应 SINE(正弦)、CIRCLE(圆型)与 L(L 型)三种模式



⇨ SCH_NUM: 焊接条件编号，对应数值为 1~10

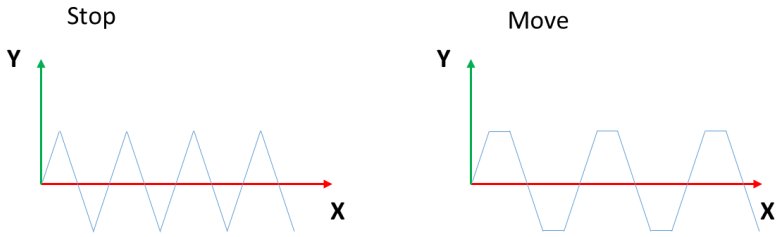
⇨ Freq = # Hz: 摆焊频率(Hz)，于 1 秒内执行摆动的次数(摆焊间距 = 焊接速度/频率)

⇨ Amp = # mm: 摆焊振幅(mm)，指焊道到摆动端点的距离



⇨ LD = # sec: 左侧停留时间(sec)，摆焊左侧端点停止时间，对应 Stop 或 Move 两种型式

⇨ RD = # sec: 右侧停留时间(sec)，摆焊右侧端点停止时间，对应 Stop 或 Move 两种型式



⇨ WEAVE_END: 结束摆焊运动，于启用摆焊运动后，透过此指令停止摆焊运动

■ **范例 1：**

```
LIN P2 FINE=1 Vel=600mm/s Acc=100% TOOL[2] BASE[4]
WEAVE SINE WS = 1
LIN P3 CONT Vel=5mm/s Acc=100% TOOL[2] BASE[4]
LIN P4 FINE=1 Vel=5mm/s Acc=100% TOOL[2] BASE[4]
WEAVE_END
```

■ **范例 1 说明：**

手臂移动至 P2 点后，以 SINE(正弦)摆焊经过 P3 到 P4 点后结束摆焊运动

■ **范例 2：**

```
LIN P2 FINE=1 Vel=600mm/s Acc=100% TOOL[2] BASE[4]
WEAVE L Freq = 1Hz Amp = 6mm LD = 0sec RD = 0sec
LIN P3 CONT Vel=5mm/s Acc=100% TOOL[2] BASE[4]
LIN P3 FINE=1 Vel=5mm/s Acc=100% TOOL[2] BASE[4]
WEAVE_END
```

■ **范例 2 说明：**

手臂移动至 P2 点后，以频率 1Hz、振幅 6mm 的 L(L 型)摆焊经过 P3 到 P4 点后结束摆焊运动 (L 型摆焊的角度需于摆焊设定页面中设定)

■ **参考相关指令：**

WELD_START, WELD_END。

2.14.5 @RESUME_PROG 焊接的设定中断复归范围程式指令

2.14.6 @CLEAR_RESUME_PROG 清除焊接的中断复归程式范围指令

■ **指令说明：**

RESUME_PROG 用于设定中断程式起始标签并以 CLEAR_RESUME_PROG 设定中断程式结束标签，在此两指令间的 LIN 运动若发生撞枪警报或人为停止程式时，于执行指定的覆归程式或以手动方式排除故障后，再次执行"启动"，便可回到原程式中的中断位置再继续执行。
(此功能不对应 EXT 模式，指令需使用键盘输入)

中断覆归功能执行程序图

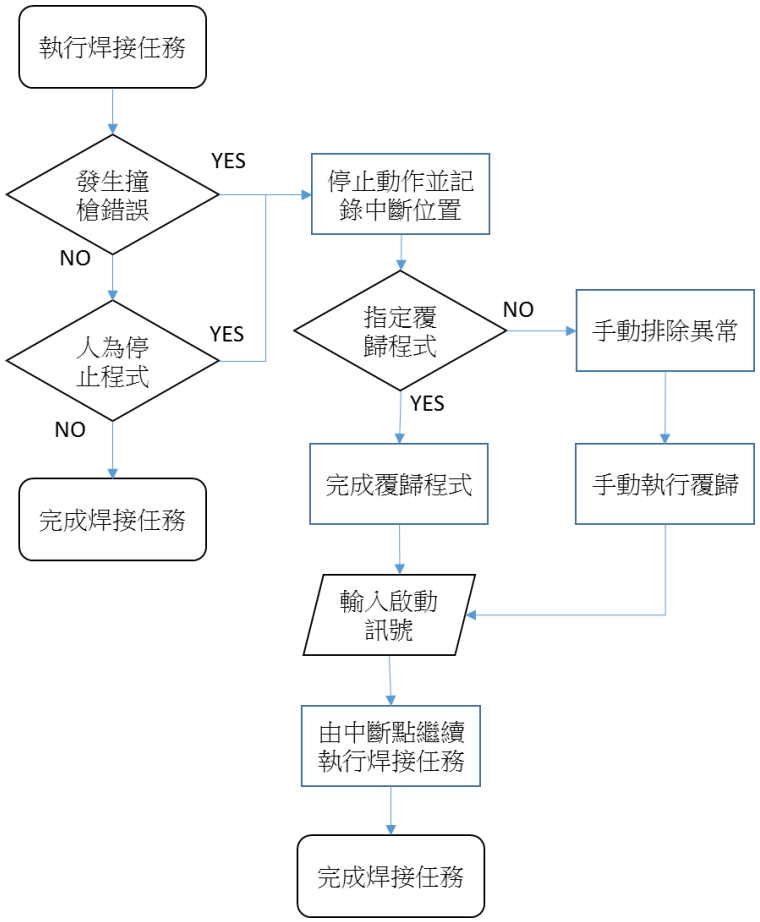


图 56 中断复归功能执行程序图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.1 以上	Caterpillar 不适用	预读 CONT

使用前需先由主功能选单->Start-up->(Next)->Arc Weld->Weld Setting 进入页面设定撞枪讯号对应的 In put 编号，并于主功能选单->Start-up->System Setting->(Next)->Resume Program 进入中断覆归设定(Resume Program Settings)中启用此功能。

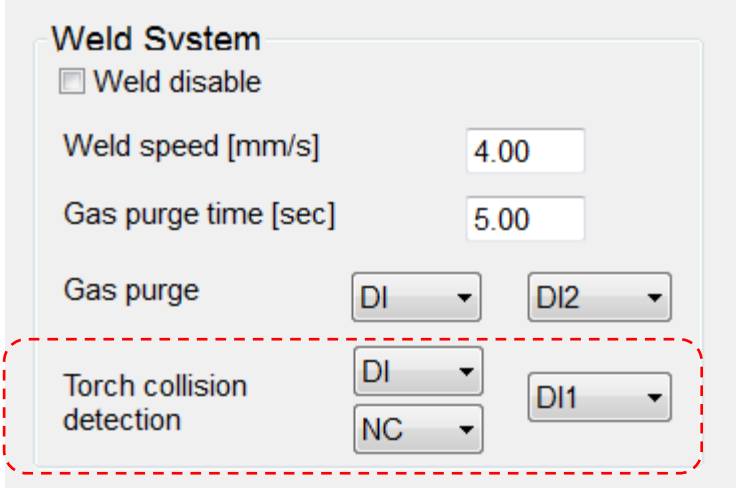


图 57 焊接设定页面图

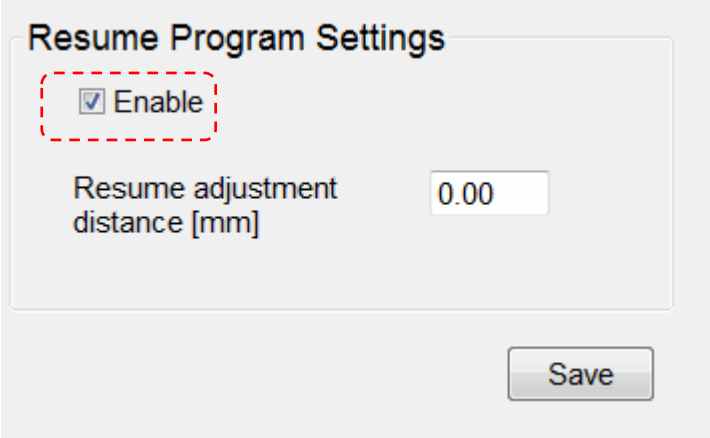


图 58 中断覆归设定页面图

■ **指令格式：**

```
RESUME_PROG
RESUME_PROG = PROG_NAME
CLEAR_RESUME_PROG
```

■ **格式/参数说明：**

⇨ PROG_NAME: 覆归程式名称，可指定放置于同一资料夹位置中的程式作为覆归程式

⇨ RESUME_PROG 后若没有指定覆归程式则不会自动执行覆归动作，可切换为 T1，并以手动的方式进行故障排除与覆归动作

⇨ CLEAR_RESUME_PROG: 设定要执行中断复归程式的范围的结束标签，在此行指令之后的程式将不会再有中断复归功能，除非再编写 RESUME_PROG 指令。

■ **范例 1：**

```
PTP P0 FINE=1 Vel=100% Acc=100% TOOL[2] BASE[3]
LIN P1 CONT Vel=600mm/s Acc=100% TOOL[2] BASE[3]
LIN P2 FINE=1 Vel=600mm/s Acc=100% TOOL[2] BASE[4]
RESUME_PROG = WIRE_CUT
WELD_START WP = 1 WS = 2
LIN P3 CONT Vel=5mm/s Acc=100% TOOL[2] BASE[4]
LIN P4 FINE=1 Vel=5mm/s Acc=100% TOOL[2] BASE[4]
WELD_END WP = 1 WS = 2
CLEAR_RESUME_PROG
LIN P5 FINE=1 Vel=600mm/s Acc=100% TOOL[2] BASE[4]
```

■ **范例 1 说明：**

设定于焊接启动到焊接结束的过程中(P3~P4 过程中)若发生撞枪或人为停止时，手臂会自动执行指定覆归程式 WIRE_CUT。

■ **范例 2：**

```
RESUME_PROG
PTP P0 FINE=1 Vel=100% Acc=100% TOOL[2] BASE[3]
LIN P1 CONT Vel=600mm/s Acc=100% TOOL[2] BASE[3]
LIN P2 FINE=1 Vel=600mm/s Acc=100% TOOL[2] BASE[4]
RESUME_PROG = WIRE_CUT
WELD_START WP = 1 WS = 2
LIN P3 CONT Vel=5mm/s Acc=100% TOOL[2] BASE[4]
LIN P3 FINE=1 Vel=5mm/s Acc=100% TOOL[2] BASE[4]
WELD_END WP = 1 WS = 2
CLEAR_RESUME_PROG
LIN P4 FINE=1 Vel=600mm/s Acc=100% TOOL[2] BASE[4]
```

■ **范例 2 说明：**

设定于焊接启动前(P0~P2 过程中)若发生撞枪或人为停止时，可手动进行覆归动作，并设定于焊接启动到焊接结束的过程中(P3~P4 过程中)若发生撞枪或人为停止时，手臂会自动执行指定覆归程式 WIRE_CUT。

■ **参考相关指令：**

WELD_START, WELD_END。

2.15 代码 O、碰撞侦测功能指令

此章节针对于代码 O 的其他功能指令做各别说明

2.15.1 @CD_START 启动碰撞侦测功能与灵敏度设定指令

■ **指令说明：**

开启碰撞侦测功能并调整碰撞侦测调整敏感度设定值指令；此指令专用于碰撞侦测功能开启时才能使用，请需要先行进行手臂关于碰撞侦测参数设定专属页面设定完后，在程式中可弹性根据手臂环境区段来调整敏感度。

碰撞侦测功能因软体侦透过电流值侦测到实体碰撞发生，透过软体方式停止手臂运行，可避免外力碰撞后电控方式导致手臂直接超过最高电流值，而产生马达驱动器故障情状产生，容易发生在教导时候人员，不正确操作时的碰撞产生。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.1 以上	Caterpillar 不适用	不预读 FINE

■ **指令格式：**

```
CD_START [敏感度数值]
```

■ **格式/参数说明：**

⇨ [敏感度数值]为碰撞侦测调整敏感度设定数值，数值为正整数，可输入范围为 1 至 100，不可为 0 值，单位为百分比(%)。

■ **范例：**

```
CD_START 80
PTP P1 FINE=1
```

■ **范例说明：**

执行到此行指令时，将会开启碰撞功能并将碰撞侦测调整敏感度调整为 80(%)的数值。
使用点到点移动方式移动至 P1(待有碰撞侦测，若感测到 80%碰撞侦测电流值，将会停下手臂来保护手臂功能)。

■ **参考相关指令：**

CD_END。

2.15.2 @CD_END 关闭碰撞侦测功能指令

■ **指令说明：**

使用此指令可关闭碰撞侦测功能；后续动作将不会因侦测到碰撞，软体方式停止手臂运行。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.1 以上	Caterpillar 不适用	不预读 FINE

■ **指令格式：**

```
CD_END
```

■ **格式/参数说明：**

⇨ CD_END 为关闭碰撞侦测功能指令。

■ **范例：**

```
CD_START 80
PTP P1 FINE=1
WAIT SEC 2
CD_END
```

■ **范例说明：**

将会开启碰撞功能并将碰撞侦测调整敏感度调整为 80(%)的数值。
使用点到点移动方式移动至 P1(待有碰撞侦测，若感测到 80%碰撞侦测电流值，将会停下手臂来保护手臂功能)，等待两秒后，
关闭碰撞侦测功能。(后续动作将不会因侦测到碰撞，软体方式停止手臂运行)

■ **参考相关指令：**

CD_START。

2.16 代码 P、字串操作功能指令

2.16.1 @STRCMP 字串比较指令

■ **指令说明：**

比较两组字串之内容是否相同字串，若相同则回传 TRUE，不相同则回传 FALSE。

<<字串指令常用于通讯使用，结果建议由通讯传输方式后确认结果正确性>>

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.5 以上	Caterpillar1.0.4 以上	预读 CONT

⇨ 先做使用前须做字串宣告，才能使用。

■ **指令格式：**

[布林变数] = STRCMP([字串变数 1] , [字串变数 2])

■ **格式/参数说明：**

⇨ [布林变数] 为真 TRUE 或假 FALSE，布林逻辑变数型态，二者择一状态，需先由使用者自行宣告定义变数名称。

⇨ [字串变数 1]、[字串变数 2] 为需要被比较的两个字串变数，变数为字串型态，需先由使用者自行宣告字串变数内容。

■ **范例：**

```
STRING word = "abc"
STRING right_word = "abc"
STRING wrong_word = "abcd"
BOOL check1,check2
check1 = STRCMP (word, right_word)
check2 = STRCMP (word, wrong_word)
```

■ **范例说明：**

宣告字串变数 word，其内容文字为"abc"。

宣告字串变数 right_word，其内容文字为"abc"；

宣告字串变数 wrong_word，其内容文字为"abcd"。

宣告布林变数 check1,check2。check1 变数之数值为比较 word 与 right_word 是否相同字串，结果为真 TRUE。Check2 变数之数值为比较 word 与 wrong_word 是否相同字串，结果为真 FALSE。

■ **参考相关指令：**

STRING、STRPOS、STRCPY、STRLEN、INTTOSTR、REALTOSTR、STRTOINT、STRTOREAL。

2.16.2 @STRPOS 搜寻字串位置指令

■ **指令说明：**

搜寻字串位置指令，有两个字串，从字串 1 中被字串 2 搜寻找相同之文字内容，并回传找到的字串 2 文字开头位置数值。

若从字串 1 无搜寻到字串 2 的文字，回传数值将为 0。

<<字串指令常用于通讯使用，结果建议由通讯传输方式后确认结果正确性>>

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.5 以上	Caterpillar1.0.4 以上	预读 CONT

■ **指令格式：**

```
[自定义变数名称] = STRCMP( [字串变数 1] , [字串变数 2] )
```

■ **格式/参数说明：**

⇨ [自定义变数名称]为自定义的变数名称，变数为整数型态，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，[自定义变数名称] 需在 STRCMP 指令前先行宣告。

⇨ [字串变数 1]为需要被搜寻的字串变数，变数为字串型态，通常为较长字串文字，需先由使用者自行宣告字串变数内容。

⇨ [字串变数 2] 为需要搜寻的字串变数或字串文字，变数为字串型态，通常为较短字串文字，需先由使用者自行宣告字串变数内容，若为字串文字前后需加上双引号「"」符号。

⇨ 从字串变数 1 中被字串变数 2 搜寻找相同之文字内容，并回传找到的字串变数 2 文字开头位置数值；若从字串变数 1 无搜寻到字串变数 2 的文字，回传数值将为 0。

■ **范例：**

```
STRING word1 = "abcde"
STRING word2 = "cd"
INT word_pos1, word_pos2
word_pos1 = STRPOS (word1, word2)
word_pos2 = STRPOS (word1," f ")
```

■ **范例说明：**

宣告字串变数 word1，其内容文字为"abcde"。

宣告字串变数 word2，其内容文字为"cd"。

宣告整数变数 word_pos1, word_pos2

从 word1 变数上("abcde")被搜寻 word2 变数("cd")之文字位置，结果为 3，将传输给 word_pos1 变数。

从 word1 变数上("abcde")被搜寻"f"之文字位置，结果没搜寻到，回传数值 0，将传输给

word_pos2 变数。

■ **参考相关指令：**

STRING、STRCMP、STRCPY、STRLEN、INTTOSTR、REALTOSTR、STRTOINT、STRTOREAL。

2.16.3 @STRCPY 字串复制指令

■ **指令说明：**

从参考来源字串中，在指定的位置开始复制特定长度到目标字串变数内。

<<字串指令常用于通讯使用，结果建议由通讯传输方式后确认结果正确性>>

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.5 以上	Caterpillar1.0.4 以上	预读 CONT

■ **指令格式：**

STRCPY([目标字串变数], [被参考字串来源变数], [参考字串起始位置变数], [参考字串复制长度变数])

■ **格式/参数说明：**

⇨ [目标字串变数] 为最终复制完的储存目标变数，用于最后变数文字结果呈现变数，变数为字串型态。

⇨ [被参考字串变数] 为被参考的字串文字内容变数，由此变数内容之文字来取得要复制的内容，经由文字起始位置与复制长度来决定，变数为字串型态；也可直接填入字串。

⇨ [参考字串起始位置变数] 为字串文字开始复制的起始位置，变数为整数型态，例如变数之数值为 2 时，代表从第 2 的文字开始；也可直接填入数值。

⇨ [参考字串复制长度变数] 为字串文字要复制的长度变数，变数为字串型态，例如变数之数值为 5 时，代表从复制 5 个文字长度；也可直接填入数值。

■ **范例 1：**

```
STRING word1 = "a"
STRCPY (word1, "abcdefg", 2, 3)
```

■ **范例 1 说明：**

宣告字串变数 word1，其内容文字为"a"。

从"abcdefg"之文字中，参考字串起始位置数值为 2，参考字串复制长度数值为 3，再将复制后的文字传输给 word1 目标字串变数，其结果为" bcd " 。

■ **范例 2：**

```
STRING word1 = "a"
STRING word2 = "abcdefg"
INT word_pos,word_length
word_pos = 2
word_length = 3
STRCPY (word1, word2, word_pos, word_length)
```

■ **范例 2 说明：**

宣告字串变数 word1，其内容文字为"a"。

宣告字串变数word2，其内容文字为"abcdefg"。

宣告整数变数word_pos1,word_length，word_pos1变数之数值为2，word_length变数之数值为3。

从 Word2 变数之文字中，参考字串起始位置数值为 2，参考字串复制长度数值为 3，再将复制后的文字传输给 word1 目标字串变数，其结果为" bcd " 。

■ **参考相关指令：**

STRING、STRCMP、STRPOS、STRLEN、INTTOSTR、REALTOSTR、STRTOINT、STRTOREAL。

2.16.4 @STRLEN 取得字串长度指令

■ **指令说明：**

取得字串长度之字元数量，回传值为整数型态。可用于判断通讯传输字元数量正确性。
<<字串指令常用于通讯使用，结果建议由通讯传输方式后确认结果正确性>>

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.5 以上	Caterpillar1.0.4 以上	预读 CONT

■ **指令格式：**

```
[自定义变数名称] = STRLEN( [字串变数名称] )
```

■ **格式/参数说明：**

⇨ [自定义变数名称]为自定义的变数名称，变数为整数型态，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，[自定义变数名称] 需在 STRLEN 指令前先行宣告。

⇨ [字串变数名称]为需要被取得字串长度的字串变数，变数为字串型态，需先由使用者自行宣告字串变数内容。

■ **范例：**

```
STRING word = "abcde"
INT word_Length
word_Length = STRLEN(word)
```

■ **范例说明：**

宣告字串变数 word，其内容文字为"abcde"
宣告整数变数 word_Length。
取得 word 变数之字串长度并传输给 word_Length 变数，其结果为 5 个字元。

■ **参考相关指令：**

STRING、STRCMP、STRPOS、STRCPY、INTTOSTR、REALTOSTR、STRTOINT、STRTOREAL。

2.16.5 @INTTOSTR 整数型态转成字串型态指令

■ **指令说明：**

整数型态转成字串型态指令，回传整数型态(INT)转型成字串型态(STRING)的结果。
<<字串指令常用于通讯使用，结果建议由通讯传输方式后确认结果正确性>>

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.5 以上	Caterpillar1.0.4 以上	预读 CONT

■ **指令格式：**

[字串变数名称] = INTTOSTR ([自定义变数名称])

■ **格式/参数说明：**

⇨ [字串变数名称] 为从整数型态转换后的字串型态变数，变数为字串型态，[自定义变数名称] 需在 INTTOSTR 指令前先行宣告。

⇨ [自定义变数名称]为自定义的变数名称，变数为整数型态(INT)，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，[自定义变数名称] 需在 INTTOSTR 指令前先行宣告，并宣告变数之整数数值。

■ **范例：**

```
INT num = 520
STRING word1
word1 = INTTOSTR(num)
```

■ **范例说明：**

宣告整数变数 num，其数值为 520。
宣告整数变数 word1。
从整数变数 num 之数值 520，从整数型态转换为字串型态并将结果传至 word1 变数，结果为 "520"。

■ **参考相关指令：**

STRING、STRCMP、STRPOS、STRCPY、STRLEN、REALTOSTR、STRTOINT、STRTOREAL。

2.16.6 @REALTOSTR 浮点数型态转成字串型态指令

■ **指令说明：**

浮点数型态转成字串型态指令，回传浮点数型态(REAL)转型成字串型态(STRING)的结果。<< 字串指令常用于通讯使用，结果建议由通讯传输方式后确认结果正确性>>

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.5 以上	Caterpillar1.0.4 以上	预读 CONT

■ **指令格式：**

[字串变数名称] = REALTOSTR([自定义变数名称])

■ **格式/参数说明：**

⇨ [字串变数名称] 为从整数型态转换后的字串变数，变数为字串型态，[自定义变数名称] 需在 REALTOSTR 指令前先行宣告。

⇨ [自定义变数名称]为自定义的变数名称，变数为浮点数型态(REAL)，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，[自定义变数名称] 需在 REALTOSTR 指令前先行宣告，并宣告变数之整数数值。

■ **范例：**

```
REAL num = 1.23
STRING word1
word1 = REALTOSTR(num)
```

■ **范例说明：**

宣告浮点数变数 num，其数值为 1.23。

宣告整数变数 word1。

从浮点数变数 num 之数值 1.23，从浮点数型态转换为字串型态并将结果传至 word1 变数，结果为"1.230000"。

■ **参考相关指令：**

STRING、STRCMP、STRPOS、STRCPY、STRLEN、INTTOSTR、STRTOINT、STRTOREAL。

2.16.7 @STRTOINT 字串型态转成整数型态指令

■ **指令说明：**

字串型态转成整数型态指令，回传字串型态(STRING)转型成整数型态(INT)的结果。

<<字串指令常用于通讯使用，结果建议由通讯传输方式后确认结果正确性>>

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.5 以上	Caterpillar1.0.4 以上	预读 CONT

■ **指令格式：**

```
[自定义变数名称] = STRTOINT ([字串变数名称])
```

■ **格式/参数说明：**

⇨ [自定义变数名称]为自定义的变数名称，变数为整数型态(INT)，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，[自定义变数名称] 需在 STRTOINT 指令前先行宣告。

⇨ [字串变数名称] 为从字串型态转换后的整数型态变数，变数为字串型态，[字串变数名称]需在 STRTOINT 指令前先行宣告，并宣告字串变数之字串内容。

■ **范例：**

```
STRING word1 = "200"
INT num
num = STRTOINT(word1)
```

■ **范例说明：**

宣告字串变数 word1，其字串之文字内容为"200"。

宣告整数变数 num。从字串变数 word1 之数值"200"，从字串型态转换为整数型态并将结果传至 num 变数，结果为 200。 (若需要为通讯使用，num 变数应该为 REAL 变数)

■ **参考相关指令：**

STRING、STRCMP、STRPOS、STRCPY、STRLEN、INTTOSTR、REALTOSTR、STRTOREAL。

2.16.8 @STRTOREAL 字串型态转成浮点数型态指令

■ **指令说明：**

字串型态转成浮点数型态指令，回传字串型态(STRING)转型成浮点数型态(REAL)的结果。<<字串指令常用于通讯使用，结果建议由通讯传输方式后确认结果正确性>>

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.5 以上	Caterpillar1.0.4 以上	预读 CONT

■ **指令格式：**

```
[自定义变数名称] = STRTOREAL ([字串变数名称])
```

■ **格式/参数说明：**

⇨ [自定义变数名称]为自定义的变数名称，变数为浮点数型态(REAL)，可以是英文字母或英文字母+数字，但是数字不能是开头字元，不可使用「-」，[自定义变数名称] 需在 STRTOREAL 指令前先行宣告。

⇨ [字串变数名称] 为从字串型态转换后的浮点数型态变数，变数为字串型态，[字串变数名称]需在 STRTOREAL 指令前先行宣告，并宣告字串变数之字串内容。

■ **范例：**

```
STRING word1 = "2.34"
REAL num
num = STRTOREAL(word1)
```

■ **范例说明：**

宣告字串变数 word1，其字串之文字内容为"2.34"。

宣告整数变数 num。从字串变数 word1 之数值"2.34"，从字串型态转换为浮点数型态并将结果传至 num 变数，结果为 2.340。

■ **参考相关指令：**

STRING、STRCMP、STRPOS、STRCPY、STRLEN、INTTOSTR、REALTOSTR、STRTOINT。

2.17 代码 Q、专用功能指令

2.17.1 @TL_START 启用扭力极限监控指令

2.17.2 @TL_END 结束扭力极限监控指令

■ **指令说明：**

开始执行扭力极限监控功能(TL_START)，若要停止扭力极限监控功能(TL_END)，此指令须先使用 TL_SET 设定扭力极限值。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.19 以上	Caterpillar1.1.2 以上	预读 CONT

⇨ 需利用 TL_SET 指令设定扭力极限监控数值。

■ **指令格式：**

```
TL_START
…
TL_END
```

■ **格式/参数说明：**

⇨ 在程式中，输入 TL_START 即可启动执行扭力极限监控功能。

⇨ 在程式中，输入 TL_END 即可停止扭力极限监控功能。

■ **范例：**

```
TL_SET(50, -1, 20, 30, -1, -1, 100, -1, -1)
TL_START
PTP P0
PTP P1
TL_END
```

■ **范例说明：**

设定扭力极限监控功能，第 1 轴设定为 50%，第 3 轴设定为 20%，第 4 轴设定为 30%，第 E1 轴设定为 100%，第 2、5、6、E2、E3 轴设定为-1 表示不启用监控。

启动扭力极限监控功能。

点到点方式移动至 P0，点到点方式移动至 P1。

结束扭力极限监控功能。

■ **参考相关指令：**

TL_SET。

3 第二部份：功能性副程式与应用说明

本章节说明程式由上银提供特定功能性副程式使用说明，例如： 门型移动 JUMP(跳跃)、矩阵 Matrix、设定安全点(ATPOS)，为特别应用程式由上银提供相关使用资料库，使用者只需要呼教副程式即可以应用，并且若使用者有更进阶需求也可以自行修改；若基本功能需求需要基本指令说明，请参考上一章节第一部份。

表 4

编号	指令分类	指令代码	功能性副程式名称	说明	页码
1	数学运算功能	FA-1	ABS	数值取绝对值计算	303
2		FA-2	ROUND	四舍五入计算	304
3		FA-3	BIT	位元转换计算	305
4		FA-4	SQRT	开平方根计算	306
5	矩阵功能	FB-1	PAL_PR	定义矩阵功能	307
6		FB-2	PAL_P	指定矩阵点位功能	310
7	区域设定功能	FC-1	BOX	设立检测区	311
8		FC-2	INSIDEBOX	判定检测区	313
9		FC-3	ATPOS	安全区域设定	315
10	点位移动功能	FD-1	JUMP	门型移动	316
11		FD-2	CIRCLE	圆周移动	318
12		FD-3	OVAL	椭圆移动	320

备注：功能性副程式名称可能为函式与副程式，标题统一为功能性副程式。

3.1 使用功能性副程式方式说明

功能性副程式为透过副程式方式呼叫已撰写的资料库，本章节说明如何应用功能性副程式。

3.1.1 将副程式汇入资料夹

想要使用副程式，需确认同一个资料夹内有主程式档案与副程式档案，官网下载档案内皆附加功能性副程式档案，不需在额外汇入，若没有此资料夹时，需在额外汇入资料夹，以下步骤教学如何将副程式档案汇入资料夹。介面说明分成上银软体系统 HRSS 3.3.XX 与远端连线系统 Caterpillar(毛毛虫)两种方式，使用针对所使用程式作应用选择。

HRSS 方式：

确认 HRSS 软体内有「HIWIN_Library」资料库档案，并将需要使用的功能性副程式放置在共同一个资料夹内，才可以呼叫副程式。

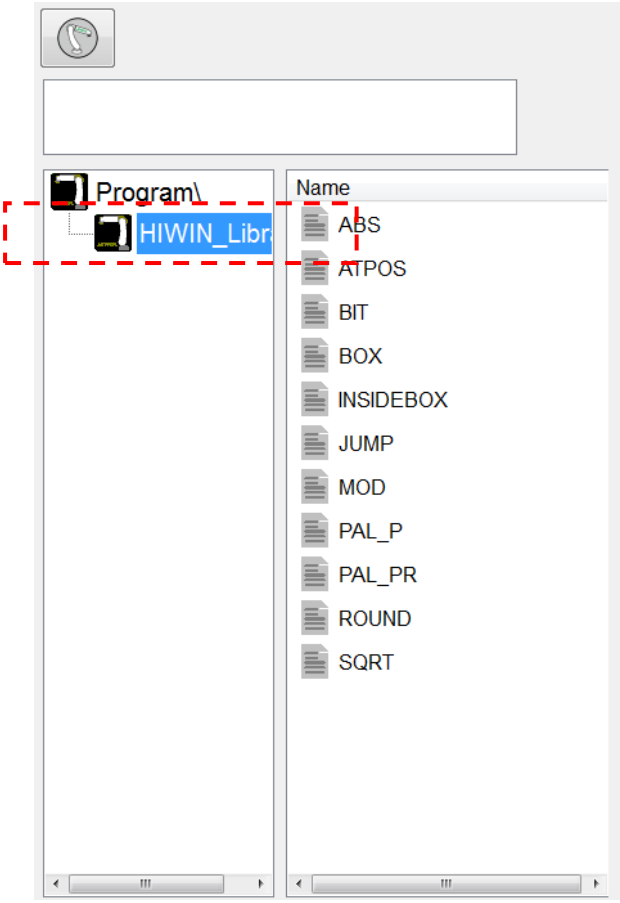


图 59 HRSS 资料库的资料夹图

可将应用主程式写在「HIWIN_Library」内即可；或者在另外程式开资料夹，将需使用的副程式复制到新资料夹。

步骤说明：(1) 创造新资料夹(NEW_File)。 (2) 点选期望使用的副程式，并按「复制(Operate File)」。 (3) 点选复制到(COPY TO)。 (4) 选择预复制资料夹名称(NEW_File)。 (5)点选「OK」按钮即完成档案复制。 复制完成后继下面章节说明如何应用。

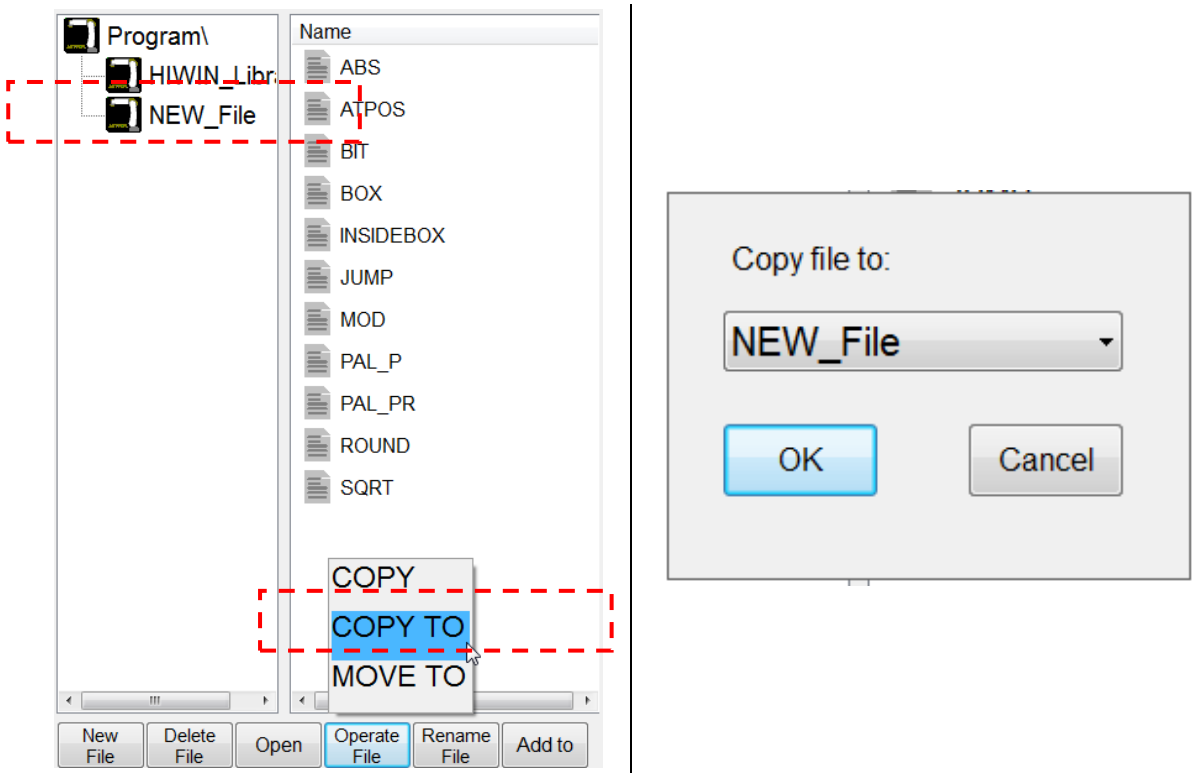


图 60 创建新资料夹并复制期望副程式档案图

若下载离线版本，无附加程式资料库(需先至官网重新下载离线软体)，依照路径将副程式汇入 HRSS 的程式档案资料夹内 HRSS→ Program→HIWIN_Library。

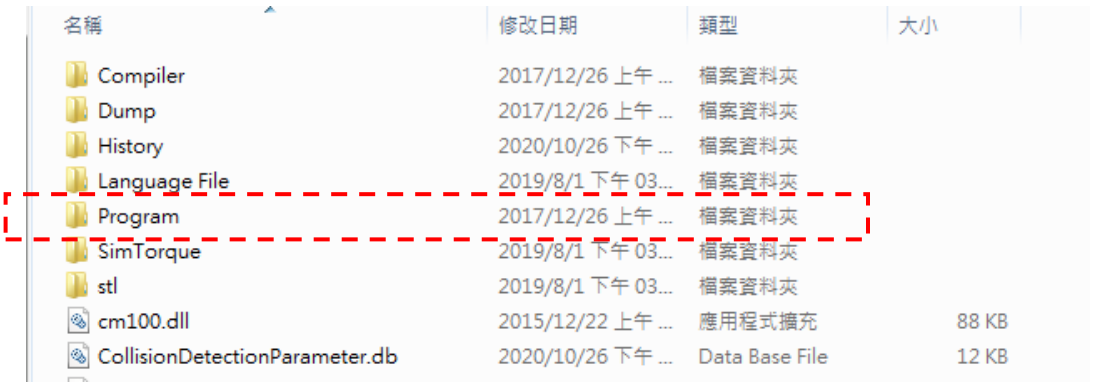


图 61 HRSS 程式档案资料夹图

Caterpillar 方式：

确认 Caterpillar 软体内有「HIWIN_Library」资料库档案，并将需要使用的功能性副程式放置在共同一个资料夹内，才可以呼叫副程式。

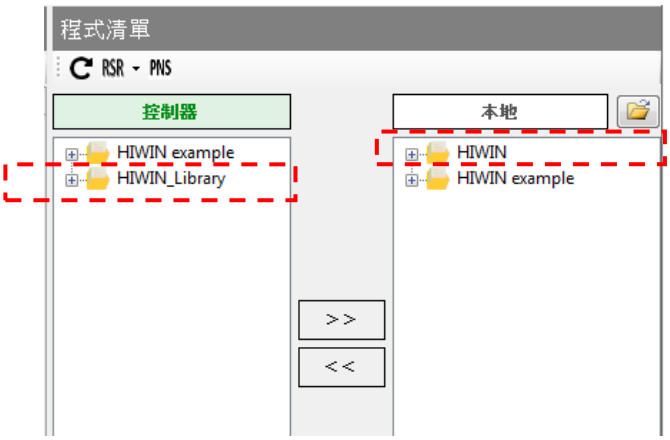


图 62 官网下载功能性副程式档案图

开启毛毛虫(Caterpillar)的资料夹，依照路径将副程式汇入毛毛虫(Caterpillar)的 Local 端资料夹内 Caterpillar→HRL_files。

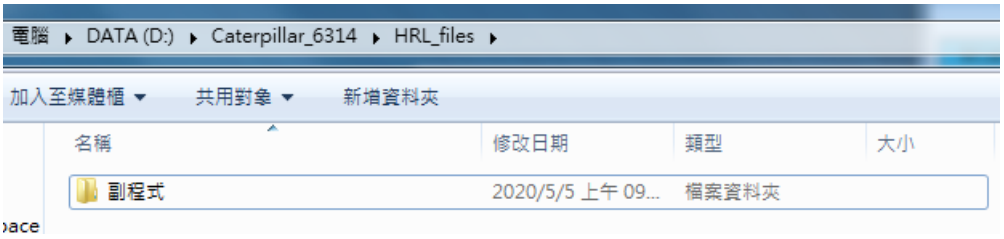


图 63 功能性副程式汇入档案路径图

透过毛毛虫(Caterpillar)程式进行副程式汇入设定。此时在 Local 本地端可以看见刚刚放进资料夹的副程式，但无法汇入 Controller 控制器端，因此需先将权限切换为控制模式并切换成手动模式后，在 Local 端选择要汇入的程式或资料夹，按下圈选的按钮程式将自动汇入 Controller 端。

欲使用功能性副程式，需让使用者执行主程式的路径需与副程式相同资料夹才能呼叫副程式运用，故请善用上述汇入功能或按右键资料夹档案复制功能，将主程式与副程式在同一资料夹内，以利呼叫功能性副程式使用。

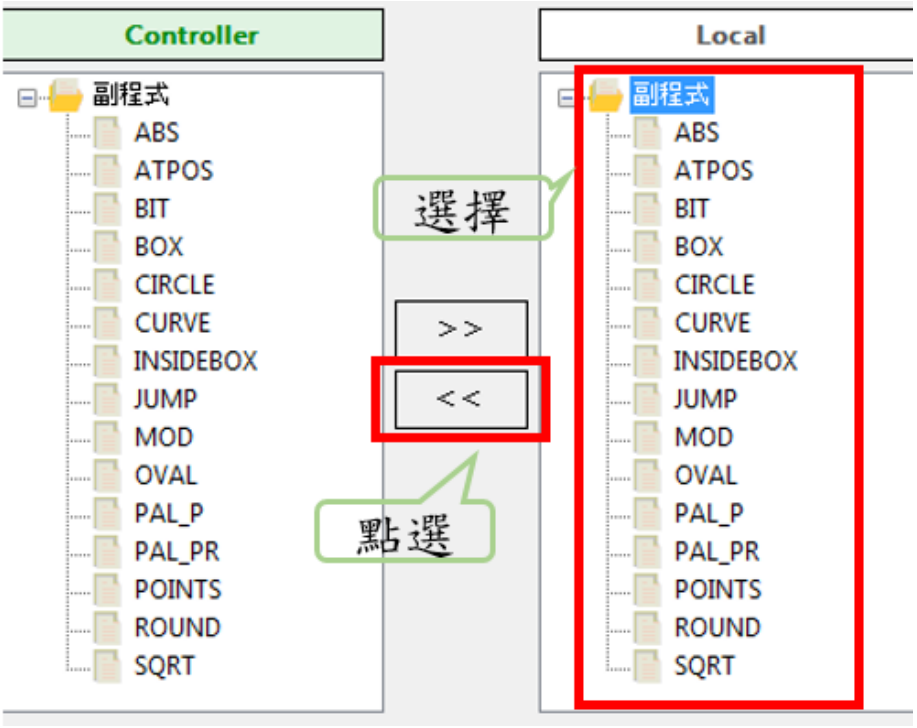


图 64 功能性副程式档案本地端汇入控制器图

复制完成后继下面章节说明如何应用。

3.1.2 主程式呼叫副程式应用说明

上银工程师已利用外部函式与副程式将功能性副程式制定功能与格式，若使用者在(主)程式内需使用到副程式内容，只要作呼叫副程式就可以应用该功能；简单说明外部函式 EXTFCT 需对函式定义参数计算并作回传数值，而外部副程式 EXT 只作参数计算，不需定义也不需回传数值(若对外部函式与副函式想更了解可参考软体手册函式与副函式章节)，本章节后续将说明如何应用的方法，以下先说明外部函式与外部副程式宣告的定义。

◎外部函式宣告的定义：

```
EXTFCT [回传值型态] [函式名称] ( [函式参数] )
EXTFCT [return_type] [function_name] ( [parameter list])
```

说明如下：

[回传值型态] 为宣告回传值型态，支援整数(INT)、浮点数(REAL)、点位(E6POINT、E6AXIS、E6POS)结构。

[函式名称]：欲使用的函式名称，参考　　表 4。

[函式参数] ：函式参数，请注意函式参数名称的宣告(位在欲呼叫的程序中)与定义(位在被呼叫的程序)必须一致，最多五个参数，支援点位型态。

◎外部副程式宣告的定义：

```
EXT [副程式名称　　] [副程式参数]
EXT [subprogram_name] ( [parameter list] )
```

说明如下：

[副程式名称]：欲使用的副程式名称，　　表 4。

[副程式参数]：副程式参数设定。副程式参数名称的宣告(位在欲呼叫的程序)与定义(位在被呼叫的程序)必须一致，最多五个参数，支援点位型态，不支援结构型态。

如何应用范例说明，这边举例外部函式范例如 MOD 与外部副程式范例如 JUMP 作说明，其他以此类推。HRSS 与 Caterpillar 共同适用。

HRSS 方式： & **Caterpillar 方式：**

✧ **如何使用 MOD 外部函式范例步骤说明：**

步骤 1.已将欲使用的外部函式/副程式汇入相同路径资料夹内。

步骤 2.开立一个新(主)程式，例如：MAIN_ABS。

步骤 3.在档案开头输入欲使用的外部函式宣告格式，以 MOD 为例需输入「EXTFCT REAL ABS(R:IN) 」，让(主)程式可知道会利用到外部函式功能，如下图第一行处；其他函式内容请参考给予的各函式第一行的程式指令，并在「DEFFCT」更换为「EXTFCT」即可。

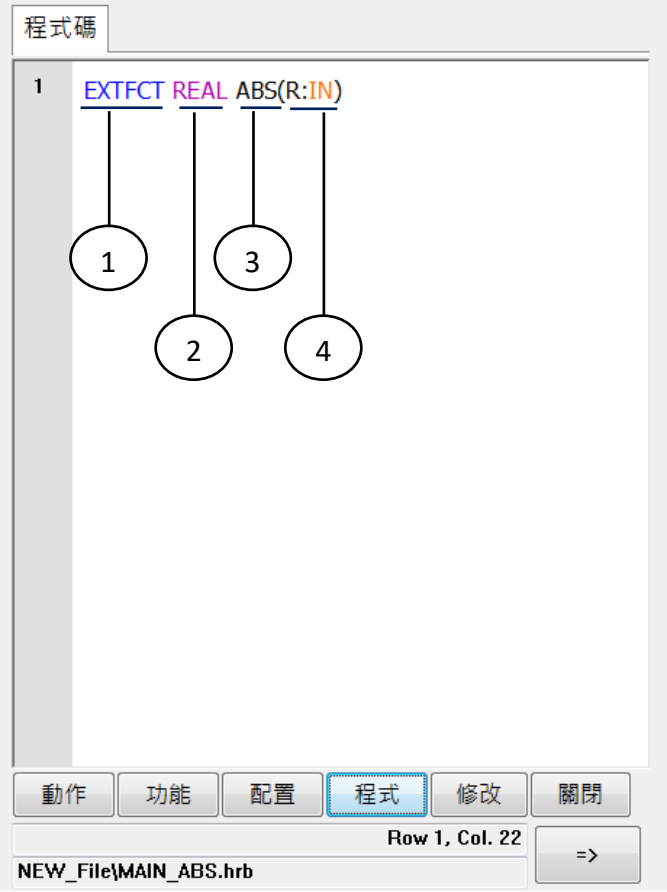


图 65 外部函式宣告格式输入图

表 15、MOD 功能性副程式宣告说明表

项次	项目名称	说明
1	外部函式宣告	EXTFCT 代表为宣告会使用到函式。
2	回传值型态	REAL 定义回传值使用整数型态。
3	函式名称	ABS 余数计算功能。
4	函式参数	R 两个为需输入的函式数值。

步骤 4. 在(主)程式内输入要使用函式，即可利用并回传数值，以下图范例，第二行输入「REAL GY1 = ABS(-1.5)」，第三行输入「$C[1] = GY1 *10」 (计数暂存器只能显示整数，故乘上 10 倍)，在暂存器**$C**[1]可以看到-1.5 的乘上 10 倍绝对数值为 15，即成功呼叫 ABS 函式。

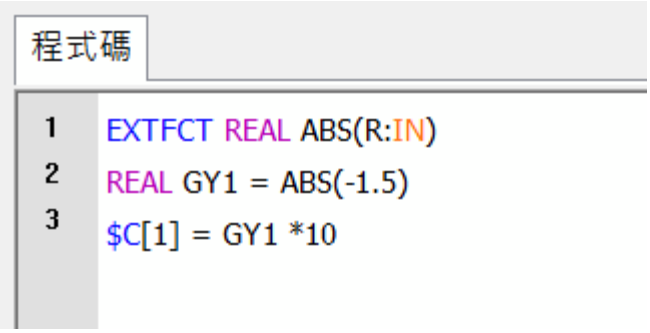


图 66 输入函式应用程式图

✧ **如何使用 JUMP 外部副程式范例步骤说明：**

步骤 1.已将欲使用的外部函式/副程式汇入相同路径资料夹内。

步骤 2.开立一个新(主)程式。

步骤 3.在档案开头输入欲使用的外部副程式宣告格式，以 JUMP 为例需输入「**EXT** JUMP(PPE:**IN**, **H**:**IN**)」，让(主)程式可知道会利用到副程式功能，如下图第一行处；其他副程式内容请参考给予的各副程式第一行的程式指令，并在「**DEF**」更换为「**EXT**」即可。

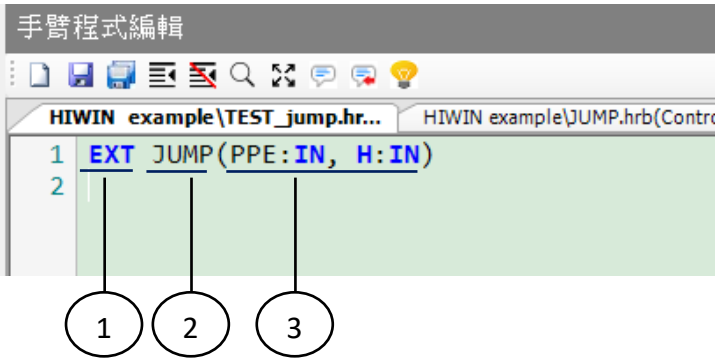


图 67 外部副程式宣告格式输入图

表 16、JUMP 功能性副程式宣告说明表

项次	项目名称	说明
1	外部副程式宣告	**EXT** 代表为宣告会使用到副程式。
2	函式名称	**JUMP** 余数计算功能。
3	函式参数	**PPE,H** 两个为需输入的副程式数值。

步骤 4. 在(主)程式内输入要使用副程式，即可利用并回传数值，以下图范例，第四行输入「JUMP(P1, -100.0) 」，在 P0 点位移动后使用门型路径移动，即成功呼叫 JUMP 副程式。

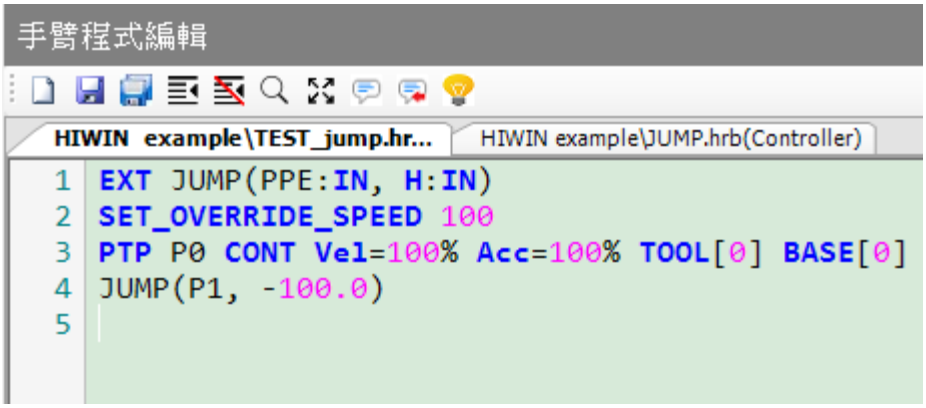


图 68 输入副程式应用程式图

3.2 代码 FA 数学运算功能性副程式说明

本章节介绍数学运算功能性副程式说明。

3.2.1@ABS 数值取绝对值计算函式

■ **指令说明：**

宣告一组数值取绝对值计算的 ABS 函式。将正数或负数数值都变为正数绝对值计算，最后回传结果传至变数或暂存器内，供后续程式运算使用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.8 以上	预读 CONT

⇨ 已将汇入 ABS 函式之程式于主程式相同路径资料夹内。

■ **指令格式：**

```
EXTFCT REAL ABS(R:IN)                          ←開頭檔
…
[自定义变数名称] = ABS( [欲绝对值变数] )          ←呼叫函式指令
```

■ **格式/参数说明：**

⇨ REAL 为浮点数(含小数点)型态宣告，对应输入值(R:IN)需给予浮点数，例如：5 需输入 5.0 数值。

⇨ [自定义变数名称]为使用者定义绝对值后储存的变数，需先宣告变数或使用计时暂存器($C[])。

⇨ [欲绝对值变数]为使用者定义欲绝对值变数(或数值)。

■ **范例：**

```
EXTFCT REAL ABS(R:IN)
REAL final_ABS_value,To_ABS_value
To_ABS_value = -3.14
final_ABS_value = ABS(To_ABS_value)
```

■ **范例说明：**

宣告外部函式 ABS，宣告 final_ABS_value,To_ABS_value 浮点数型式。

To_ABS_value 初始数值为-3.14。

final_ABS_value 为 To_ABS_value 经过绝对值后的数值，结果为 3.14。

■ **参考相关指令：**

REAL、EXTFCT。

3.2.2@ROUND 四舍五入计算函式

■ **指令说明：**

宣告一组数值取四舍五入计算的 ROUND 函式，将判定数值若小于 5，如 0~4，则往前一位数值保持原本数值，判定数值若大于 4，如 5~9，则往前一位数值将数值加 1，供后续程式运算使用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.8 以上	预读 CONT

⇨ 已将汇入 ROUND 函式之程式于主程式相同路径资料夹内。

■ **指令格式：**

```
EXTFCT REAL ROUND(R:IN, N:IN)                          ←開頭檔
...
[自定义变数名称] = ROUND ( [欲四舍五入变数] , [小数点位数] )   ←呼叫函式指令
```

■ **格式/参数说明：**

⇨ REAL 为浮点数(含小数点)型态宣告，对应输入值(R:IN, N:IN)需给予浮点数，例如：5 需输入 5.0 数值

⇨ [自定义变数名称]为使用者定义四舍五入后储存的变数，需先宣告变数或使用暂存器($C[ ])。

⇨ [欲四舍五入变数]为使用者定义欲四舍五入变数。

⇨ [小数点位数]为欲作小数点后位数数值，取下一个位置作四舍五入变数的判定，例如 1.45，若数值 1 表示为小数点后 1 位，则对 5 作四舍五入，对 4 作数值加 1，四舍五入后变为 1.5。

■ **范例：**

```
EXTFCT REAL ROUND(R:IN, N:IN)
REAL final_ROUND_value,To_ROUND_value
To_ROUND_value = 1.45
final_ROUND_value = ROUND(To_ROUND_value,1)
```

■ **范例说明：**

宣告外部函式 ROUND，宣告 final_ ROUND _value, To_ ROUND _value 浮点数型式。

To_ROUND_value 初始数值为 1.45。

final_ROUND_value 为 To_ROUND_value 小数点后第一位经过四舍五入后的数值，**结果为** 1.5。

■ **参考相关指令：**

REAL、EXTFCT。

3.2.3@BIT 位元转换计算函式

■ **指令说明：**

宣告一组数位输入 DI 讯号转换为位元数值的 BIT 函式，从二进制方式转换为十进制。可善用于感测器总合计算应用，供后续程式运算使用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.8 以上	预读 CONT

⇨ 已将汇入 BIT 函式之程式于主程式相同路径资料夹内。

■ **指令格式：**

```
EXTFCT INT BIT(A:IN, B:IN)                                    ←開頭檔
…
[自定义变数名称] = BIT ( [数位输入起始讯号数] , [数位输入终止讯号数])    ←呼叫函式指令
```

■ **格式/参数说明：**

⇨ INT 为整数型态宣告。

⇨ [变数名称]为使用者定义位元转换后储存的变数，需先宣告变数或使用暂存器(**$C**[ ])。

⇨ [数位输入起始讯号数]为数位输入讯号$DI[]的起始通道设定值，可填入数值为 1~24。

⇨ [数位输入终止讯号数]为数位输入讯号$DI[]的终止通道设定值，可填入数值为 1~24。

■ **范例：**

```
EXTFCT INT BIT(A:IN, B:IN)
$C[1] = BIT(1,3)
```

■ **范例说明：**

宣告外部函式 BIT，将数位输入讯号从 1 至 5 通道数值作位元转换，从二进制方式转换为十进制；如下图，此时若$DI[1]、$DI[2]、$DI[3] 有讯号为开 ON，则暂存器$C[1]数值结果为 7。

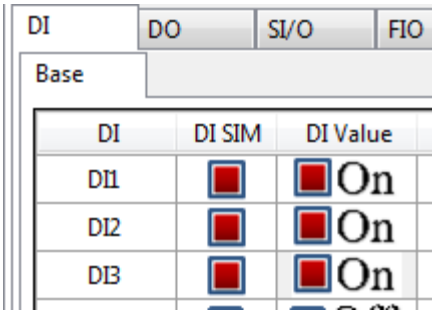


图 69 数位输入讯号 DI[1]~ DI[3]开启图

■ **参考相关指令：**

INT、EXTFCT

3.2.4@SQRT 开平方根计算函式

■ **指令说明：**

宣告一组数值作开平方根的 SQRT 函式，同等数学符号"√ "，作数学式开更根计算后，供后续程式运算使用。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.8 以上	预读 CONT

⇨ 已将汇入 SQRT 函式之程式于主程式相同路径资料夹内。

■ **指令格式：**

```
EXTFCT REAL SQRT(I:IN)                          ←開頭檔
…
[自定义变数名称] = SQRT( [欲开根号变数] )          ←呼叫函式指令
```

■ **格式/参数说明：**

⇨ REAL 为浮点数(含小数点)型态宣告，对应输入值(R:IN)需给予浮点数，例如：5 需输入 5.0 数值。

⇨ [自定义变数名称]为使用者定义开根号后储存的变数，需先宣告变数。

⇨ [欲开根号变数] 为欲开根号变数。

■ **范例：**

```
EXTFCT REAL SQRT(I:IN)
REAL final_SQRT_value,To_SQRT_value
To_SQRT_value = 9.0
final_SQRT_value = (SQRT(To_SQRT_value) )
```

■ **范例说明：**

宣告外部函式 SQRT，宣告 final_ROUND_value,To_ROUND_value 浮点数型式。

定义 To_ROUND_value 数值为 9.0， final_ROUND_value 为 To_ROUND_value 作开根号计算后数值，结果为 3。

■ **参考相关指令：**

REAL、EXTFCT。

3.3 代码 FB 矩阵功能性副程式说明

若使用多数量行列排列整列、堆迭应用，矩阵为常见的应用功能，此章节描述如何定义矩阵功能与指定矩阵点位功能。

3.3.1@PAL_PR 定义矩阵功能副程式

■ **指令说明：**

。宣告一组矩阵参数规格定义的 PAL_PR 副程式，使用者需要先告知所期望的矩阵资讯。需宣告一个为 PAL_PR 的副程式的六个参数，矩阵编号 INDEX, 行数 ROW,列数 COL,矩阵的三个代表边点 PP1, PP2,PP3，作完定义矩阵完成的座标资讯，将点位输入于位置暂存器**$PR**[ ]，记录点位会利用在位置暂存器**$PR**[3501]点位之后，HRSS 位置暂存器路径在显示资讯 Display >> 变数 Variable >> 位置暂存器 PR 可以查到介面使用。(暂定可设定 5 组不同的矩阵)

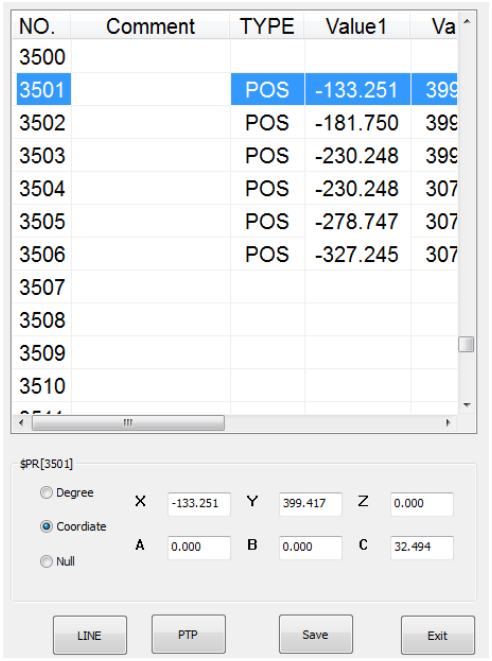


图 70 定义矩阵点位参数在位置暂存器内图(HRSS 介面)

Caterpillar 位置暂存器路径在显示资讯 >> 变数 >> 位置暂存器 PR 可以查到介面使用。(暂定可设定 5 组不同的矩阵)

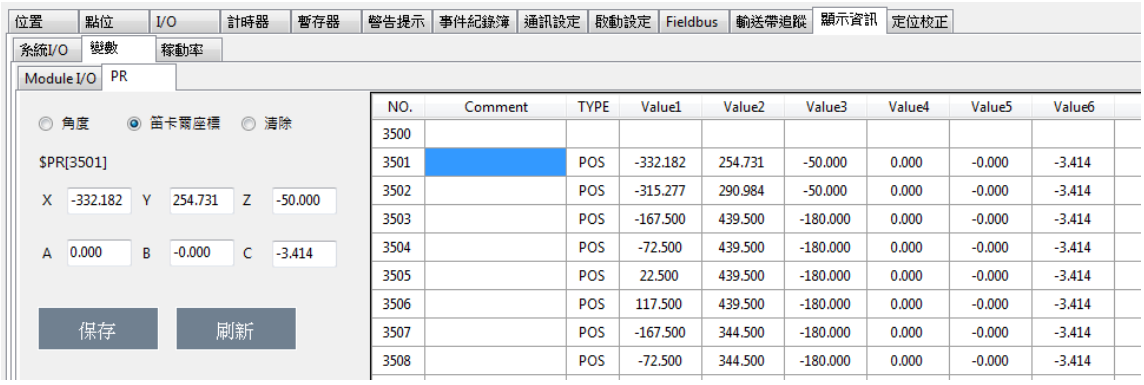


图 71 定义矩阵点位参数在位置暂存器内图(Caterpillar 介面)

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.8 以上	预读 CONT

⇨ 已将汇入 PAL_PR 副程式之程式于主程式相同路径资料夹内。

■ **指令格式：**

```
EXT PAL_PR(INDEX:IN, ROW:IN, COL:IN, PP1:IN, PP2:IN, PP3:IN)        ←開頭檔
…
PAL_PR( [矩阵编号],[行数],[列数],[定义点 1],[定义点 2],[定义点 3] )   ←呼叫副程式指令
```

■ **格式/参数说明：**

⇨ [矩阵编号] 为输入目前使用的矩阵号码，每使用一个 PAL_PR 即为一个新矩阵号码，暂定可设定 5 组不同的矩阵。

⇨ [行数] 为矩阵的行数(ROW)定义。

⇨ [列数] 为矩阵的列数(COLUMN)定义。

⇨ [定义点 1],[定义点 2],[定义点 3] 为定义矩阵起始点、横向最远点位与直向最远点位三个点位，如下图示意图 PP1、PP2、PP3 三点，藉由三点定义其矩阵位置、行和列距离资讯，也可算出每个点位的资讯，点位资讯将记录在位置暂存器内($PR[3501]之后)。

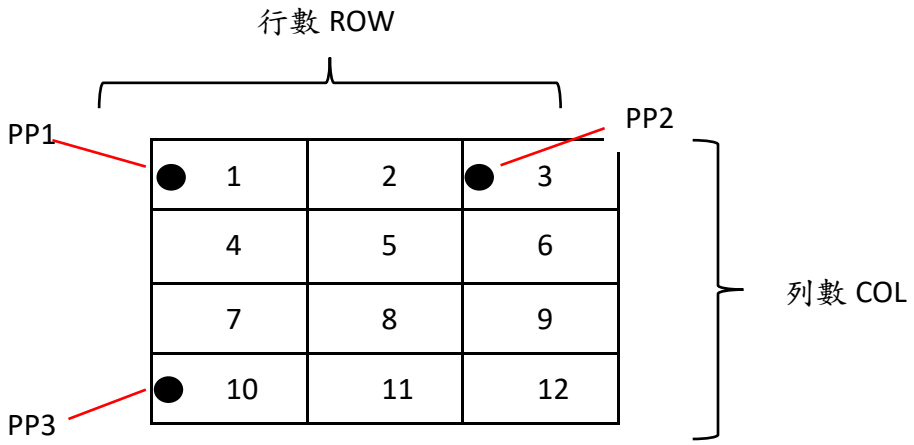


图 72 定义矩阵点位参数示意图

■ **范例：**

EXT PAL_PR(INDEX:IN, ROW:IN, COL:IN, PP1:IN, PP2:IN, PP3:IN)

PAL_PR(1, 3, 2, P1, P2, P3)

■ **范例说明：**

宣告外部副程式 PAL_PR。

定义第 1 个矩阵编号，2 行 3 列矩阵参数，根据 P1、P2、P3 定义点求得 2X3 矩阵点位资讯，如下图案例，P1(100,400,-50,-52.679)、P2(200,400,-50,-52.679)、P1(100,450,-50,-52.679)，可求得 6 个点位记录于$PR[3501]~ $PR[3506]。

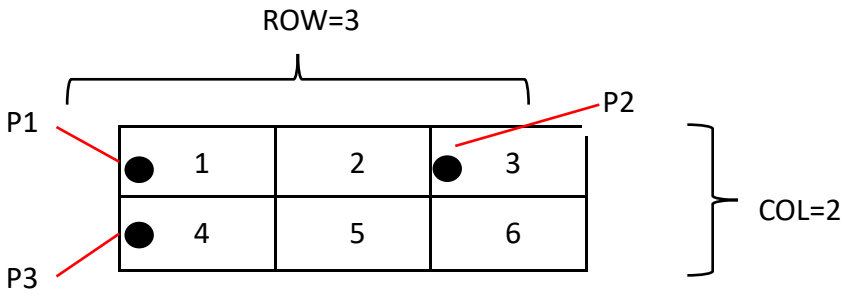


图 73 定义矩阵点位参数示意图

NAME	COMMENT	X	Y	Z	C
P0	HOME	0.000	500.000	0.000	0.000
P1	PP1	100.000	400.000	-50.000	-52.679
P2	PP2	200.000	400.000	-50.000	-52.679
P3	PP3	100.000	450.000	-50.000	-52.679

图 74 定义 P1、P2、P3 矩阵点位位置图

NO.	Comment	TYPE	Value1	Value2	Value3	Value4	Value5	Value6
3500								
3501	第1點	POS	100.000	400.000	-50.000	0.000	0.000	-52.679
3502	第2點	POS	150.000	400.000	-50.000	0.000	0.000	-52.679
3503	第3點	POS	200.000	400.000	-50.000	0.000	0.000	-52.679
3504	第4點	POS	100.000	450.000	-50.000	0.000	0.000	-52.679
3505	第5點	POS	150.000	450.000	-50.000	0.000	0.000	-52.679
3506	第6點	POS	200.000	450.000	-50.000	0.000	0.000	-52.679
3507								

图 75 显示矩阵点位参数位置暂存器示意图

■ **参考相关指令：**

$PR[]、EXT、PAL_P 函式、ARRAY 一维至三维阵列(第 2.1.15 章节)。

3.3.2@PAL_P 指定矩阵点位功能函式

■ **指令说明：**

宣告一组位置数值作指定矩阵点位 PAL_P 函式，也就是汇出下一步欲移动的位置资讯，会搭配 PAL_PR 指令使用，先由 PAL_PR 定义完成矩阵参数与点位，再透过 PAL_P 指令，将指定的 PR 位置值汇出至指定位置变数中，才能移动手臂至期望位置。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.8 以上	预读 CONT

⇨ 已将汇入 PAL_P、PAL_PR 函式之程式于主程式相同路径资料夹内，并已作完 PAL_PR 矩阵参数设定(必要)。

■ **指令格式：**

```
EXTFCT E6POINT PAL_P(INDEX:IN, N:IN)
...
[位置变数名称] = PAL_P( [矩阵编号],[点位编号] )
```

■ **格式/参数说明：**

⇨ E6POINT 为点位位置变数型态宣告。

⇨ [位置变数名称] 为使用者定义矩阵点位资讯汇入的位置变数，需先宣告位置变数或使用位置暂存器($PR[ ])。

⇨ [矩阵编号] 为输入目前使用的矩阵号码。

⇨ [点位编号] 为在矩阵内部的点位编号排序，输入的编号即为下一步要移动的资讯。最小为 1，最大数值为行数*列数数值。

■ **范例：**

```
EXT PAL_PR(INDEX:IN, ROW:IN, COL:IN, PP1:IN, PP2:IN, PP3:IN)
EXTFCT E6POINT PAL_P(INDEX:IN, N:IN)
E6POINT arrive_position
PAL_PR(1, 3, 2, P1, P2, P3)
arrive_position = PAL_P(1,2)
LIN arrive_position FINE=1
```

■ **范例说明：**

宣告外部副程式 PAL_PR、函式 PAL_P。宣告位置变数 arrive_position。

定义第 1 个矩阵编号，2 行 3 列矩阵参数，根据 P1、P2、P3 定义点求得 2X3 矩阵点位资讯。

将第 1 个矩阵编号，第 2 个点位编号的位置数值汇出给 arrive_position。以 PAL_PR 范例，编号 2 点位为(150,400,-50)，再以 **LIN** 直线移动方式移至 arrive_position 点位位置。

■ **参考相关指令：**

E6POINT、REAL、$PR[]、EXT、EXTFCT、PAL_PR 函式、ARRAY 一维至三维阵列(第 2.1.15 章节)。

3.4 代码 FC 区域设定功能性副程式说明

此章节说明相关区预设定的功能副程式如 BOX 设立检测区域、INSIDEBOX 判定检测区域、ATPOS 安全区预设定。

3.4.1@BOX 设立检测区副程式

■ **指令说明：**

宣告一组手臂区域检测空间的 BOX 副程式，只作宣告手臂能移动的空间范围，若要判定手臂是否在空间范围内，需搭配 INSIDEBOX 副程式使用。宣告一个为 BOX 的副程式的 7 个参数，区域编号 INDEX、X 轴值上限 X_H、 X 轴值下限 X_L、 Y 轴值上限 Y_H、 Y 轴值下限 Y_L、 Z 轴值上限 Z_H、Z 轴值下限 Z_L，作完定义检测区域完成的参数资讯，将区域参数输入于位置暂存器**$PR**[ ]，记录区域参数会在位置暂存器**$PR**[3491] **~$PR**[3450]参数，位置暂存器路径在显示资讯 >> 变数 >> PR 可以查到介面使用。(可设定 10 组不同的检测区域)

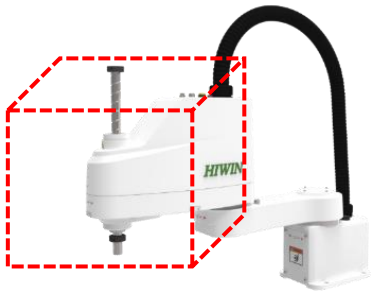


图 76 界定手臂检测区域示意图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.8 以上	预读 CONT

⇨ 已将汇入 BOX 副程式之程式于主程式相同路径资料夹内。

■ **指令格式：**

```
DEF BOX(INDEX:IN, X_H:IN, X_L:IN, Y_H:IN, Y_L:IN, Z_H:IN, Z_L:IN)   ←開頭檔
…
                                                                    ←呼叫副程式指令
BOX([区域编号], [X 轴上限值], [X 轴下限值], [Y 轴上限值], [Y 轴下限值], [Z 轴上限值], [Z 轴下限值] )
```

■ **格式/参数说明：**

⇨ [区域编号] 为定义检测区域编号数值，输入 1~10，共 10 组不同的检测区域编号。

⇨ [X 轴上限值][X 轴下限值] 为定义世界座标 X 轴方向的上、下限数值。

⇨ [Y 轴上限值][Y 轴下限值] 为定义世界座标 Y 轴方向的上、下限数值。

⇨ [Z 轴上限值][Z 轴下限值] 为定义世界座标 Z 轴方向的上、下限数值。

⇨ 数值皆为浮点数 REAL 型态。

■ **范例：**

EXT BOX(INDEX:IN, X_H:IN, X_L:IN, Y_H:IN, Y_L:IN, Z_H:IN, Z_L:IN)
BOX(1, 454.0,304.0,15.0,-185.0,-60.0,-173.5)

■ **范例说明：**

宣告外部副程式 BOX，定义第一个检测区域范围 X 轴介于 454.0~304.0mm 之间，Y 轴介于 15.0~-185.0mm 之间，Z 轴介于-60.0~-173.5mm 之间。

NO.	Comment	TYPE	Value1	Value2	Value3	Value4	Value5	Value6
3490								
3491	第1個檢測區域	POS	454.000	304.000	15.000	-185.000	-60.000	-173.500

图 77 显示检测区域参数位置暂存器示意图

■ **参考相关指令：**

REAL、 INSIDEBOX 副程式、EXT。

3.4.2@INSIDEBOX 判定检测区副程式

■ **指令说明：**

宣告一组判定手臂在区域检测空间内的 INSIDEBOX 副程式，判定 BOX 设定的 X、Y、Z 轴区域皆需在范围内，判定结果后将结论输出于使用者设定的数位输出讯号中，若在范围内输出讯号会为开 ON，在透过判定出输讯号来判定是否能作后续程式动作(提醒一下，非随时都在作区域判定，需使用者动作前先判定)；此判定检测副程式需先将 BOX 副程式作检测区域参数设定才可以使用。

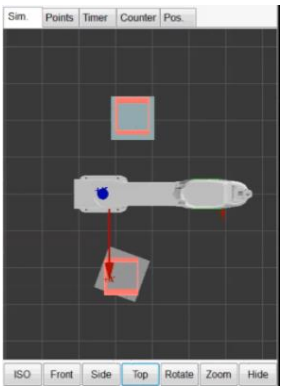


图 78 判定检测区域内显示示意图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.8 以上	预读 CONT

⇨ 已将汇入 BOX、INSIDEBOX 副程式之程式于主程式相同路径资料夹内，并已作完 BOX 检测区域设定。

■ **指令格式：**

```
EXT INSIDEBOX(INDEX:IN, N:IN)                    ←開頭檔
...
INSIDEBOX( [区域编号],[数位输出讯号编号] )          ←呼叫副程式指令
```

■ **格式/参数说明：**

⇨ [区域编号] 为定义检测区域编号数值，可输输入数值 1~10。

⇨ [数位输出讯号编号] 为数位输出讯号编号，例如数值填 1，即表示**$DO**[1] = **TRUE**。

■ **范例：**

```
EXT BOX(INDEX:IN, X_H:IN, X_L:IN, Y_H:IN, Y_L:IN, Z_H:IN, Z_L:IN)
EXT INSIDEBOX(INDEX:IN, N:IN)
BOX(1, 454.0,304.0,15.0,-185.0,-60.0,-173.5)
WAIT SEC 0.05
INSIDEBOX(1, 3)
```

■ **范例说明：**

宣告外部副程式 BOX、INSIDEBOX。

定义第一个检测区域范围 X 轴介于 454.0~304.0mm 之间，Y 轴介于 15.0~-185.0mm 之间，Z 轴介于-60.0~-173.5mm 之间。

等待 0.5 秒后判定手臂是否在第一个检测区域范围空间之内，若有在范围内，在第 3 数位输出讯号开启**$DO[3] = TRUE**；若有在范围内在第 3 数位输出讯号关闭**$DO[3] = FALSE**。

(若不使用等待时间，需在前面动作中使用非连续轨迹 FINE = 1，程式才不会预读)

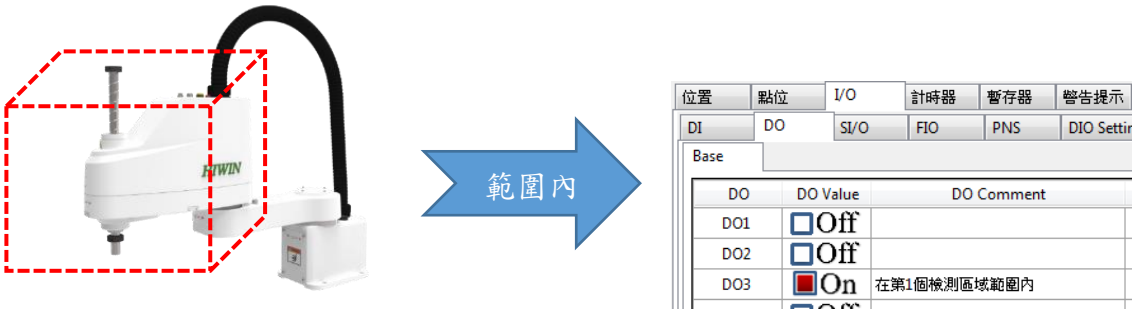


图 79 判定手臂在检测区域内显示示意图

■ **参考相关指令：**

REAL、EXT、BOX 副程式。

3.4.3@ATPOS 安全区域设定副程式

■ **指令说明：**

宣告一组安全区域判定的 ATPOS 函式，欲判定点位与被判定点位若为同一点位上，则数位输出讯号为开启 ON，公差范围为 1 mm 或 1 度。

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.8 以上	预读 CONT

⇨ 已将汇入 ATPOS 副程式之程式于主程式相同路径资料夹内。

■ **指令格式：**

```
EXT ATPOS(PPE:IN, N:IN)                          ←開頭檔
…
ATPOS( [被判定点位],[ 数位输出讯号编号] )          ←呼叫副程式指令
```

■ **格式/参数说明：**

⇨ [被判定点位] 为使用者预判定的点位或点位变数，例如：P1 或 PP1。

⇨ [数位输出讯号编号] 为数位输出讯号编号，例如数值填 1，即表示$DO[1] = TRUE；若欲判定点位与被判定点位在同一点位上，则数位输出讯号为开启 ON，公差范围为 1 mm 或 1 度。

■ **范例：**

```
EXT ATPOS(PPE:IN, N:IN)
ATPOS(P1,2)
```



图 80 被判定手臂点位资讯示意图

■ **范例说明：**

宣告外部副程式 ATPOS，

若欲判定点位与被判定点位 P1 在同一点位上，则数位输出讯号第二通道为开启 ON($DO[2] = TRUE)。

■ **参考相关指令：**

EXT。

3.5 代码 FD 点位移动功能性副程式说明

本章节描述为在点位与点位间互相移动使用的特殊功能 JUMP 门型移动、CIRCLE 圆周移动、OVEL 椭圆移动等功能。

3.5.1@JUMP 门型移动副程式

■ **指令说明：**

宣告一组门型移动的 JUMP 副程式，参考图 81 中门型移动轨迹，此门型移动副程式较适用于 SCARA 手臂机型，若使用在六轴手臂机型，请特别注意是否导致多关节动作异常或撞机现象。(门型移动路径上方采用为点到点 PTP 移动方式，特别为了避免直线 LIN 移动时的奇异点或姿态问题，若有特别需求需要更改原参考副程式指令)。

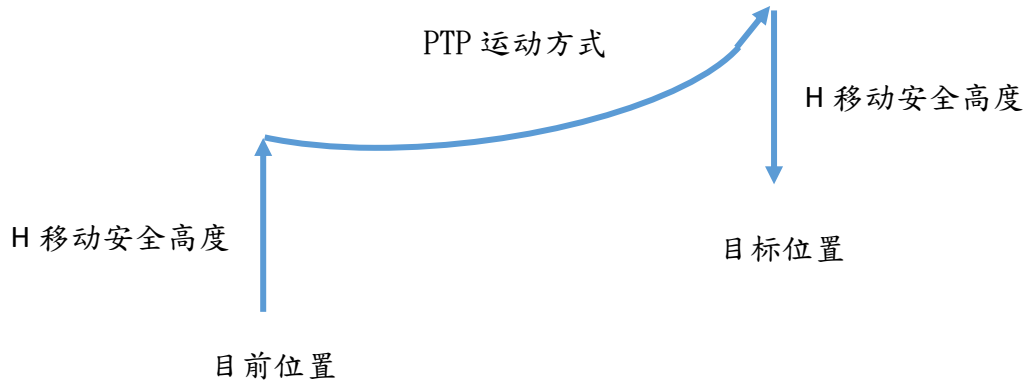


图 81 JUMP 门型移动轨迹示意图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.8 以上	预读 CONT

⇨ 已将汇入 JUMP 副程式之程式于主程式相同路径资料夹内。

■ **指令格式：**

```
EXT JUMP(PPE:IN, H:IN)                          ←開頭檔
...
JUMP( [目标点位],[拉升安全高度] )                ←呼叫副程式指令
```

■ **格式/参数说明：**

⇨ [目标点位] 为经过门型移动后到达的位置点。

⇨ [拉升安全高度] 为防止撞机产生的一个拉升安全高度距离设定，单位为毫米 mm，参考图 81 中，变数 H 距离，变数 H 需要给定 REAL 型态的数值，比如距离 5mm 需要给 5.0 mm，经确位置计算，若只输入整数数值会发生错误。

⇨ 拉升高度后所需使用的移动方式为 PTP 点到点的方式，故需特别小心是否也会导致撞到周边设备，若需更改移动方式，至 JUMP 副程式档案作修改为 LIN 直线方式。

■ **范例：**

```
EXT JUMP(PPE:IN, H:IN)
PTP P0 FINE=1 Vel=100% Acc=100% TOOL[0] BASE[0]
JUMP(P1, 100.0)
```

■ **范例说明：**

宣告外部副程式 JUMP。

PTP 方式移动到 P0 点位后，再以门型移动方式移动至 P1 点位，移动拉升安全高度为 100.0mm 距离。

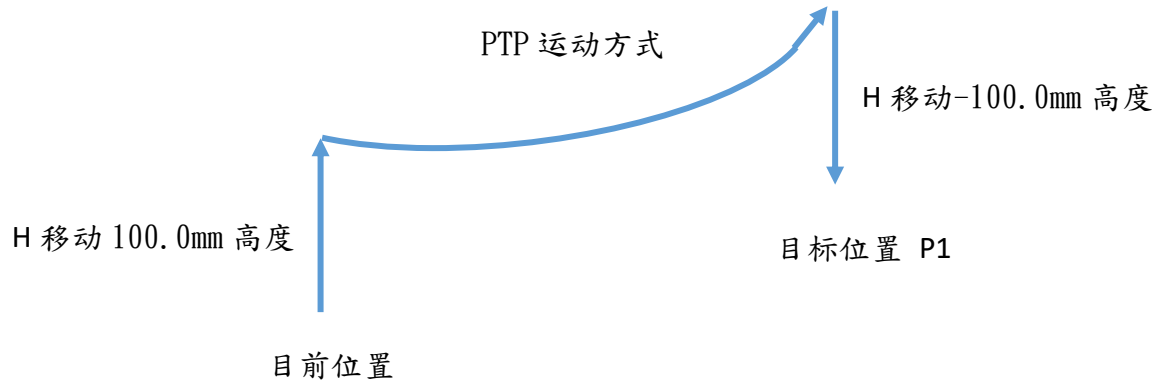


图 82 图 JUMP 门型移动 P1 轨迹示意图

■ **参考相关指令：**

EXT。

3.5.2@CIRCLE 圆周移动副程式

■ **指令说明：**

宣告一组圆周移动的 CIRCLE 副程式，让手臂行走一个圆的动作，需给予手臂圆心位置、圆半径、及输出讯号编号等资讯，圆的起始位置需在 CIRCLE 副程式前，使用者先移动至起始位置点位。此副程式可适用于圆形工件作点胶作业。

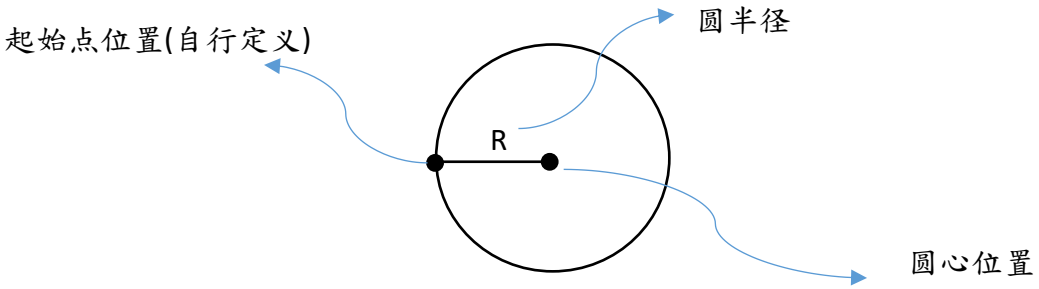


图 83　CIRCLE 圆周移动示意图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.8 以上	预读 CONT

⇨　已将汇入 CIRCLE 副程式之程式于主程式相同路径资料夹内。

■ **指令格式：**

```
EXT CIRCLE(PPC:IN, R:IN, N:IN)                    ←開頭檔
...
CIRCLE( [圆心位置], [圆半径], [数位输出讯号编号] )    ←呼叫副程式指令
```

■ **格式/参数说明：**

⇨　[圆心位置] 为画圆的圆心位置。

⇨　[圆半径] 为从圆心至周边的半径距离，单位为毫米 mm。

⇨　[数位输出讯号编号] 为数位输出讯号编号，例如数值填 1，即表示**$DO**[1] = **TRUE**。

⇨　圆的起始位置需在 CIRCLE 副程式前，先移动至起始位置点位。

■ **范例：**

```
EXT CIRCLE(PPC:IN, R:IN, N:IN)
PTP P1 FINE=1 Vel=50% Acc=50% TOOL[0] BASE[0]
CIRCLE(P0, 100.0, 2)
```

■ **范例说明：**

宣告 CIRCLE 外部副程式，并以点到点方式移动至 P1 点位，
以 P0 点为圆心位置，再画半径 100mm 圆周，再画圆的轨迹同时，会输出数位输出讯号编号第 2 通道开启**$DO**[2] = **TRUE**。

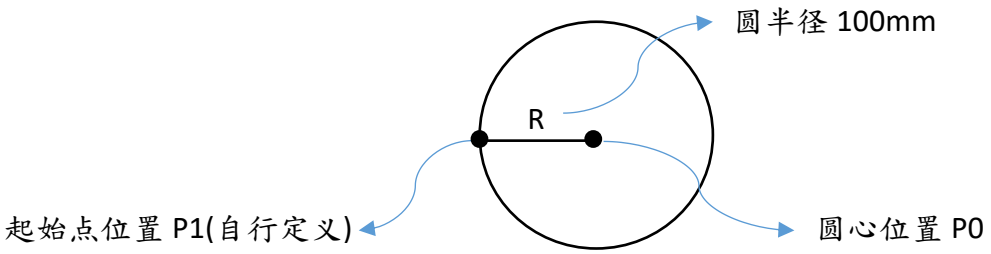


图 84　CIRCLE 圆周运动 P0 示意图

■ **参考相关指令：**

SYN、EXT、OVAL 副程式。

3.5.3@OVAL 椭圆移动副程式

■ **指令说明：**

宣告一组椭圆运动的 OVAL 副程式，让手臂行走一个椭圆的动作，需给予手臂圆心位置、长边半径、短边半径、及输出讯号编号等资讯，椭圆的起始位置需在 OVAL 副程式前，使用者先移动至起始位置点位。此副程式可适用于椭圆形工件作点胶作业。

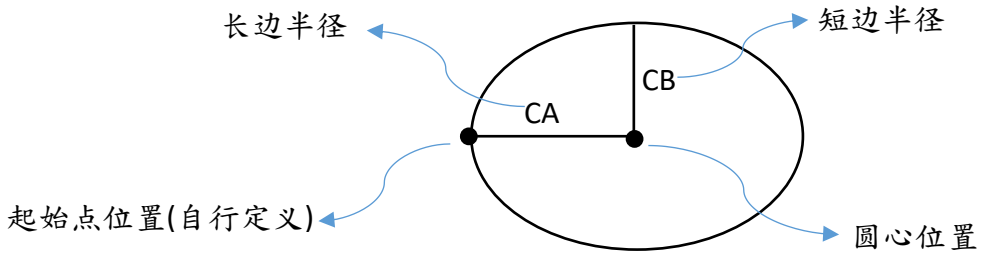


图 85　OVAL 椭圆运动示意图

■ **前提条件：**

适用软体版本		指令预设状态
HRSS3.3.8 以上	Caterpillar1.0.8 以上	预读 CONT

⇨　已将汇入 OVAL 副程式之程式于主程式相同路径资料夹内。

■ **指令格式：**

```
EXT OVAL(PPC:IN, CA:IN, CB:IN, N:IN)                                    ←開頭檔
...
OVAL( [椭圆圆心位置], [椭圆长边半径], [椭圆短边半径], [数位输出讯号编号] )   ←呼叫副程式指令
```

■ **格式/参数说明：**

⇨　[椭圆圆心位置] 为画圆的圆心位置。

⇨　[椭圆长边半径] 为从圆心至长边的半径距离，单位为毫米 mm。

⇨　[椭圆短边半径] 为从圆心至短边的半径距离，单位为毫米 mm。

⇨　[数位输出讯号编号] 为数位输出讯号编号，例如数值填 1，即表示**$DO**[1] = **TRUE**。

⇨　椭圆的起始位置需在 OVAL 副程式前，先移动至起始位置点位。

■ **范例：**

```
EXT OVAL(PPC:IN, CA:IN, CB:IN, N:IN)
PTP P1 FINE=1 Vel=50% Acc=50% TOOL[0] BASE[0]
OVAL(P0, 100.0, 50.0, 5)
```

■ **范例说明：**

宣告 OVAL 外部副程式，并以点到点方式移动至 P1 点位，
以 P0 点为椭圆圆心位置，再画长边半径 100mm、短边半径 50mm 椭圆，再画椭圆的轨迹同时，会输出数位输出讯号编号第 5 通道开启**$DO**[5] = **TRUE**。

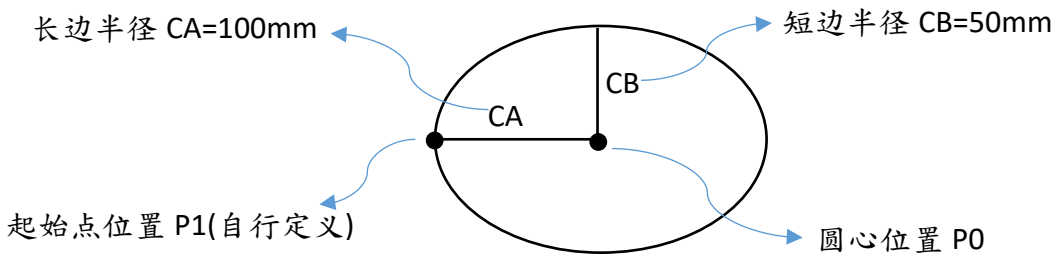


图 86　OVAL 椭圆 P0 运动示意图

■ **参考相关指令：**

SYN、EXT、CIRCLE 副程式。

附录一、指令集预设连续(CONT)/不连续(FINE)轨迹总表

此附录为指令预设为连续轨迹(预读)CONT/非连续轨迹(不预读)FINE 的整理表格，方便使用者能预先了解该指令的特性，并了解是否能透过程式指令来修改。

若原本为连续轨迹 CONT 状态，需请使用者特别注意程式预读状态，才不会导致有动作提前动作或动作不符的现象产生；改善方式可以透过程式使用非连续轨迹(FINE)指令或者在预读前一行增加等待时间(WAIT SEC)指令来停止(卡住)预读状态。

下页表中，「O」表示预设设定状态；「-」 表示无、空白；「△」 表示可由程式修改，需使用 CONT/FINE 功能。

表 17 指令集预读预设状态表

分类代号	指令分类	指令编号	指令名称	连续 CONT (预读)	精确 FINE (不预读)
A	变数/函式宣告	A-1	REAL	O	-
		A-2	INT	O	-
		A-3	BOOL	O	-
		A-4	CHAR	O	-
		A-5	E6POS	O	-
		A-6	E6AXIS	O	-
		A-7	E6POINT	O	-
		A-8	FRAME	O	-
		A-9	STRUC	O	-
		A-10	DEF... END	O	-
		A-11	DEFFCT...ENDFCT	O	-
		A-12	EXT	O	-
		A-13	EXTFCT	O	-
		A-14	STRING	O	-
		A-15	ARRAY 自定义矩阵名称	O	-
		A-16	UINT	O	-

备注：「O」表示预设设定状态；「-」 表示无、空白；
「△」 表示可由程式修改，需使用 CONT/FINE 功能。

表 17(续 1)

分类代号	指令分类	指令编号	指令名称	连续 CONT (预读)	精确 FINE (不预读)
B	运动指令	B-1	PTP	O	△
		B-2	PTP_REL	O	△
		B-3	LIN	O	△
		B-4	LIN_REL	O	△
		B-5	LIN_REL_TOOL	O	△
		B-6	CIRC	O	△
		B-7	CIRC_REL	O	△
		B-8	SPLINE	O	△
		B-9	STOP_MOTION	O	-
		B-10	BRAKE	O	-
		B-11	CHECK_LIN	O	-
		B-12	P#_EXIST()	O	-
		B-13	P#()	O	-
		B-14	CT_A6	O	-
C	暂存器	C-1	$C[]	O	-
		C-2	$T[]	O	-
		C-3	$T_STOP[]	O	-
		C-4	$PR[]	O	-
		C-5	$SRR[]、$SRW[]	O	-
D	输入、输出讯号控制	D-1	$DI[]	-	O
		D-2	$DO[]	△	O
		D-3	$SI[]	-	O
		D-4	$SO[]	△	O
		D-5	$VO[]	△	O
		D-6	$RO[]	△	O
		D-7	$RI[]	-	O
		D-8	SYN	O	-
		D-9	PULSE	-	O
		D-10	$MI[]	-	O
		D-11	$MO[]	△	O

备注：「O」表示预设设定状态；「-」 表示无、空白；
「△」 表示可由程式修改，需使用 CONT/FINE 功能。

表 17(续 2)

分类代号	指令分类	指令编号	指令名称	连续 CONT (预读)	精确 FINE (不预读)
E	流程控制功能	E-1	IF	O	-
		E-2	FOR	O	-
		E-3	LOOP	O	-
		E-4	WHILE	O	-
		E-5	REPEAT	O	-
		E-6	GOTO	O	-
		E-7	SWITCH	O	-
		E-8	WAIT SEC	-	O
		E-9	WAIT FOR	-	O
		E-10	WAIT FOR ... TIMEOUT	-	O
		E-11	AND	-	O
		E-12	OR	-	O
		E-13	INTERRUPT…WHEN…DO…	O	-
		E-14	INTERRUPT ON/OFF	O	-
		E-15	BREAK_POINT	O	-
F	运动参数/系统设定	F-1	SET_OVERRIDE_SPEED	O	-
		F-2	SET_SPEED	O	-
		F-3	SET_ACC	O	-
		F-4	SET_ROTATION_SPEED	O	-
		F-5	SET_TOOL	-	O
		F-6	SET_BASE	-	O
		F-7	SET_PAYLOAD	O	-
		F-8	GETPOINT	O	-
		F-9	GET_MOTION_STATUS	O	-
		F-10	EXT_TCP_START	-	O
		F-11	EXT_TCP_END	-	O
		F-12	TL_SET	O	-
		F-13	SERVO_ON	O	-
		F-14	SERVO_OFF	O	-
		F-15	SERVO_ON(#)	O	-
		F-16	SERVO_OFF(#)	O	-

备注：「O」表示预设设定状态；「-」 表示无、空白；
「△」 表示可由程式修改，需使用 CONT/FINE 功能。

表 17(续 3)

分类代号	指令分类	指令编号	指令名称	连续 CONT (预读)	精确 FINE (不预读)
G	数学计算	G-1	+	O	-
		G-2	-	O	-
		G-3	*	O	-
		G-4	/	O	-
		G-5	SIN	O	-
		G-6	COS	O	-
		G-7	TAN	O	-
		G-8	ASIN	O	-
		G-9	ACOS	O	-
		G-10	ATAN	O	-
		G-11	ATAN2	O	-
		G-12	FWD	O	-
		G-13	%	O	-
H	通讯功能	H-1	COPEN	O	-
		H-2	CCLOSE	O	-
		H-3	CCLEAR	O	-
		H-4	CREAD	O	-
		H-5	CWRITE	△	O
		H-6	CINQUIRE	O	-

备注：「O」表示预设设定状态；「-」 表示无、空白；
「△」 表示可由程式修改，需使用 CONT/FINE 功能。

表 17(续 4)

分类代号	指令分类	指令编号	指令名称	连续 CONT (预读)	精确 FINE (不预读)
I	输送带	I-1	CNV_START	O	-
		I-2	CNV_END	O	-
		I-3	CNV_PICK_QUANTITY	O	-
		I-4	CNV_TRIGGER_TIMES	O	-
		I-5	CNV_PICK	O	△
		I-6	CNV_PLACE	O	△
		I-7	CNV_OBJECT	O	-
		I-8	CNV_FULL	O	-
		I-9	CNV_EMPTY	O	-
		I-10	CNV_SET_DELAY_TIME	O	-
		I-11	CNV_QUEUE_SIZE	O	-
		I-12	CNV_OBJ_CNT_DIST	O	-
		I-13	CNV_PLACE_BATCH	O	-
		I-14	CNV_RESET_ENC	O	-
		I-15	CNV_QUEUE_REMOVE	O	-
		I-16	CNV_PICK_ACC	O	-
		I-17	CNV_OFFSET_X	O	-
		I-18	CNV_OFFSET_Y	O	-
		I-19	CNV_OFFSET_Z	O	-
		I-20	CNV_SPEED	O	-
J	讯息提示	J-1	USER_ALARM	-	O
K	模拟环境	K-1	ADDTOOL	O	-
		K-2	ADDOBJ	O	-
		K-3	SHOW_TOOL	O	-
		K-4	SHOW_OBJ	O	-
		K-5	MOVEFLOOR	O	-
		K-6	AXISON	O	-
		K-7	AXISOFF	O	-

备注：「O」表示预设设定状态；「-」 表示无、空白；
「△」 表示可由程式修改，需使用 CONT/FINE 功能。

表 17(续 5)

分类代号	指令分类	指令编号	指令名称	连续 CONT (预读)	精确 FINE (不预读)
L	电动夹爪功能	L-1	EG_OPEN	-	O
		L-2	EG_CLOSE	-	O
		L-3	EG_RESET	△	O
		L-4	EG_GET_STATUS	△	O
		L-5	EG_RUN_MOVE	△	O
		L-6	EG_RUN_GRIP	△	O
		L-7	EG_RUN_EXPERT	△	O
		L-8	EG_GOT_POS	△	O
M	外部轴功能	M-1	EX_AX	-	O
		M-2	EX_AX_ASYNC	O	-
		M-3	EX_AX_SET_ACC	O	-
		M-4	EX_AX_SET_SPEED	O	-
		M-5	EX_AX_CT	O	-
		M-6	ASYPTP	O	△
N	焊接功能	N-1	WELD_START	-	O
		N-2	WELD_END	-	O
		N-3	WEAVE	O	-
		N-4	WEAVE_END	O	-
		N-5	RESUME_PROG	O	-
		N-6	CLEAR_RESUME_PROG	O	-
O	碰撞侦测功能	O-1	CD_START	-	O
		O-2	CD_END	-	O
P	字串操作	P-1	STRCMP	O	-
		P-2	STRPOS	O	-
		P-3	STPCPY	O	-
		P-4	STRLEN	O	-
		P-5	INTTOSTR	O	-
		P-6	REALTOSTR	O	-
		P-7	STRTOINT	O	-
		P-8	STRTOREAL	O	-
Q	专用功能	Q-1	TL_START	O	-
		Q-2	TL_END	O	-

备注：「O」表示预设设定状态；「-」 表示无、空白；
「△」 表示可由程式修改，需使用 CONT/FINE 功能。

§连续运动轨迹 CONT (预读)使用详细说明：

程式执行时，执行到运动参数为 CONT 时，程式会预先读入 CONT 之后的程式资讯，以利预先规划 CONT 程式之移动路径，使 TCP 运动路径为圆滑轨迹。

如该点只是 TCP 移动的路径点，可使用 CONT 使运作顺畅，若该点为 TCP 必须精准抵达之点，则需使用 FINE。CONT 的路径会随着速度及加速度的不同而有所变化，但是如果 CONT 后面有接贝兹曲线平滑的百分比或转角的半径长度，例如：CONT = 50%或 CONT = 30mm，移动路径则为固定路径。

CONT 有 CONT, CONT = #%, CONT = #mm 三种使用情形(#符号代表数字)。

■ **指令格式：**

格式(1)：CONT **格式(2)：CONT = #%** **格式(3)：CONT = # mm**

⇨ **格式(1)：CONT**

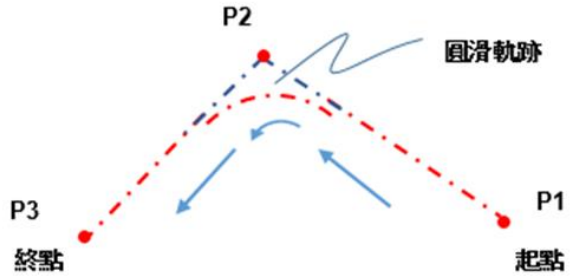
单纯指令，无任何指定圆滑比例与圆滑距离，预设为参数 100%圆滑比例。

LIN P1 CONT

LIN P2 CONT

LIN P3 CONT

上述指令会在 P2 点有圆滑轨迹，P1 和 P3 分别为起点和终点，所以没有圆滑轨迹。



CONT連續運動軌跡

⇨ **格式(2)：CONT = #%**

由使用者指定圆滑百分比例。

LIN P1 CONT

LIN P2 CONT = 50%

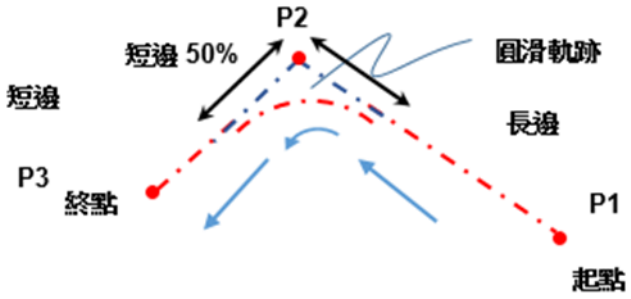
LIN P3 CONT

P1 和 P2 之间距离较长，称之为长边。

P2 和 P3 之间距离较短，称之为短边。

上述指令，运动轨迹会从 P2 点回推，短边长度 50%的距离处，开始圆滑轨迹。

注意	使用 CONT= # %时，为指定轨迹使用，机器人速度可能会因轨迹限制而减速。



CONT = 50%連續運動軌跡

⇨ **格式(3)：CONT = # mm**

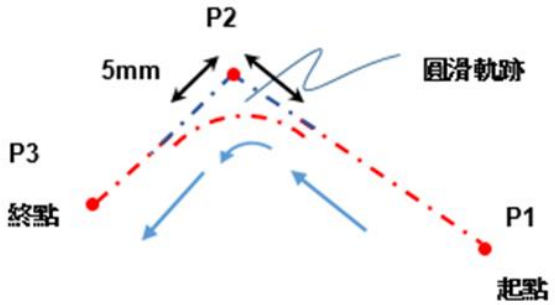
由使用者指定圓滑距离。

LIN P1 CONT

LIN P2 CONT = 5mm

LIN P3 CONT

上述指令，运动轨迹会从 P2 点回推 5mm 的距离处，开始圆滑轨迹。



CONT = 5mm連續運動軌跡

注意	使用 CONT= # mm 时，为指定轨迹使用，机器人速度可能会因轨迹限制而减速。

注意	使用 CONT= # mm 时，并不意味着与用户指定的值完全一致。但是系统将以用户指定的距离进行尝试。

若在有设定 CONT 的 P 点之前有使用 DO 讯号，那运动将会先停止运动再重新启动运动。

例如：

LIN P1 CONT

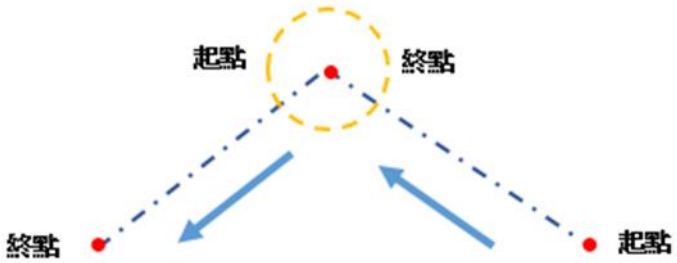
$DO[1] = TRUE

LIN P2 CONT

LIN P3 CONT

在 P1 到 P2 本来会是平滑曲线，但是中间有一个 DO，以致于变成 P1 到位后停止，再开始移动到 P2 点，CONT 就无作用(即无预读功能)。

§非连续运动轨迹 FINE(不预读)使用详细说明：



FINE非連續運動軌跡

如该点只是 TCP 移动的路径点，可使用 CONT 使运作顺畅，若该点为 TCP 必须精准抵达之点，则需使用 FINE，FINE 指令又分成三种情况，可供使用者因任务状况自行作选择，预设状态为 FINE =1。

■ **指令格式：**

格式(1)：FINE (=0)
格式(2)：FINE = 1
格式(3)：FINE = 2

FINE 有以下 三种运用情况

⇨ FINE (= 0)， 会预读接续的指令，不会执行到位检查，等同 CONT 指令，可节省时间。

⇨ FINE = 1 ，不会预读接续的指令，不会执行到位检查，建议使用。

⇨ FINE = 2 ，不会预读接续的指令， 会执行到位检查，精准到位。

※预读功能说明

在电脑程式运算中，会在程式底层先预先读取程式码，用于多个动作后续作编译结合后，再作整合输出的动作，我们称之为预先读取的功能，简称预读功能。预读的功能优点可以让程式预先了解下一步需动作内容，并作调整、精简时间的动作，但缺点就有可能多种类型程序时(如计数、判断式与计算类型)，时序会有可能发生不正确性状态，CONT 指令都具备预读功能。

举例说明：如下图，手臂有顺序移动 P1 至 P5 等五个点位，当手臂在进行 P2 点位时，程式已经读取到 P5 点位(上方绿色虚线箭头处)，就可以更确定后面的动作。

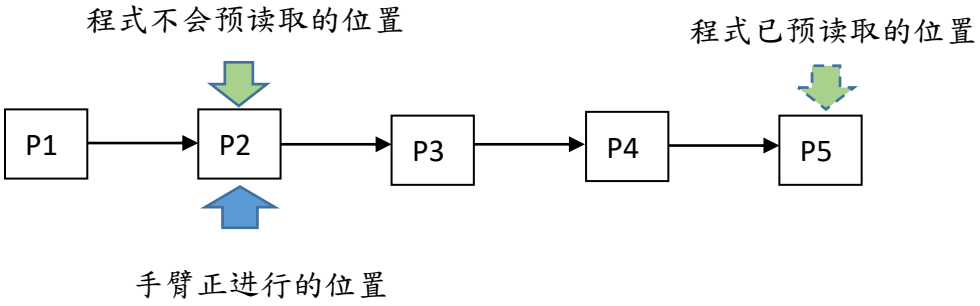


图 87 预读功能举例示意图

※到位功能说明

所谓的到位功能是指程式判定到达目标位置的准确程度，误差值极低为精准到位，误差值高为粗略到位/不到位，如下图，将其到位位置放大来看，虚线圆圈内为点位的误差范围(马达到位误差值)，当 P1 点位与到达点位与精准有三的等级，FINE(=0)相等于连续 CONT 功能，不到位且不精准；佳的为 FINE=1，到位且不精准；最佳的为 FINE=2，到位且精准，因需要准确会比较耗时间。

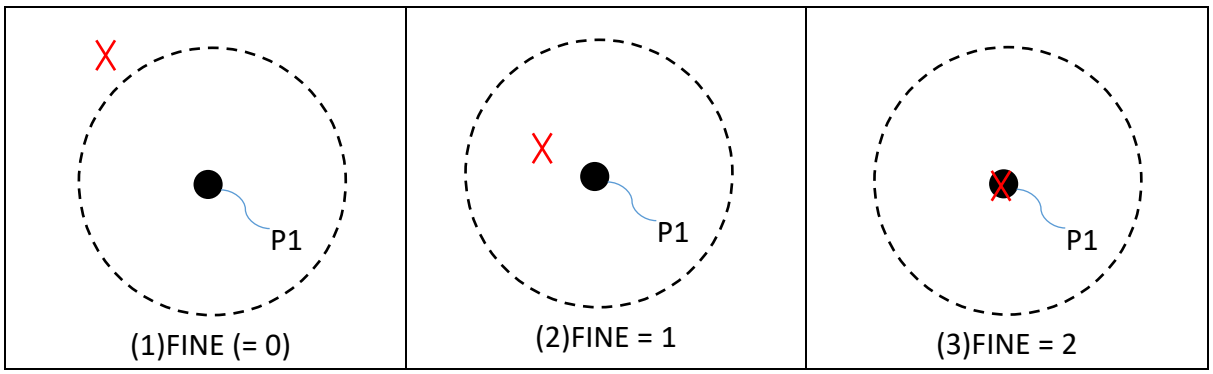


图 88 到位功能举例示意图(1)不到位 (2)到位且不精准 (3) 到位且精准

● 表示 P1 点 ， ╳ 表示手臂到位点，虚线圆圈内为点位的误差范围。

■ **应用程式范例 1： (使用 CONT 说明，同等 FINE=0)**

```
PTP P1 CONT
LIN P2 CONT
$C[1] = $C[1] + 1
```

■ **应用程式范例 1 说明：**

执行点到点移动方式至 P1 点，采连续轨迹方式(CONT)。

执行点到点移动方式至 P2 点，采连续轨迹方式(CONT)。

第一个计数器执行执行加 1 的计算。

结果描述：可以顺序从 P1 点移动至 P2 点并完成计数动作，P1 与 P2 之间动作衔接的是连续轨迹，可节省时间，但在刚开始移动 P1 动作时，程式已作预读，已经先作完计数器的计算+1，计数动作已偷跑未等待执行完 P2 才作计数动作。

■ **应用程式范例 2：(使用 FINE = 1 说明)**

```
PTP P1 FINE = 1
LIN P2 FINE = 1
$C[1] = $C[1] + 1
```

■ **应用程式范例 2 说明：**

执行点到点移动方式至 P1 点，采非连续轨迹方式(FINE=1)。。

执行点到点移动方式至 P2 点，采非连续轨迹方式(FINE=1)。

第一个计数器执行执行加 1 的计算。

结果描述：可以顺序从 P1 点移动至 P2 点并完成计数动作，P1 与 P2 之间动作衔接的是非连续轨迹，P1&P2 为到位方式是不完全精准的位置，程式不会预读，计数动作等待执行完 P2 才作计数动作。

■ **应用程式范例 3：(使用 FINE = 2 说明)**

```
PTP P1 FINE = 1
LIN P2 FINE = 2
$C[1] = $C[1] + 1
```

■ **应用程式范例 3 说明：**

执行点到点移动方式至 P1 点，采非连续轨迹方式(FINE=1)。。

执行点到点移动方式至 P2 点，采非连续轨迹方式(FINE=2)。

第一个计数器执行执行加 1 的计算。

结果描述：可以顺序从 P1 点移动至 P2 点并完成计数动作，P1 与 P2 之间动作衔接的是非连续轨迹， P2 为到位方式是完全精准的位置，程式不会预读，计数动作等待执行完 P2 才作计数动作。

附录二、ASCII 可显示字元表

表 18　ASCII 可显示字元表(ASCII printable characters)-1

十进制 DEC	十六进制 HEX	图形 Symbol	十进制 DEC	十六进制 HEX	图形 Symbol	十进制 DEC	十六进制 HEX	图形 Symbol
0	00	NULL	32	20		64	40	@
1	01	SOH	33	21	!	65	41	A
2	02	STX	34	22	"	66	42	B
3	03	ETX	35	23	#	67	43	C
4	04	EOT	36	24	$	68	44	D
5	05	ENQ	37	25	%	69	45	E
6	06	ACK	38	26	&	70	46	F
7	07	BEL	39	27	'	71	47	G
8	08	BS	40	28	(	72	48	H
9	09	HT	41	29	)	73	49	I
10	0A	LF	42	2A	*	74	4A	J
11	0B	VT	43	2B	+	75	4B	K
12	0C	FF	44	2C	,	76	4C	L
13	0D	CR	45	2D	-	77	4D	M
14	0E	SO	46	2E	.	78	4E	N
15	0F	SI	47	2F	/	79	4F	O
16	10	DLE	48	30	0	80	50	P
17	11	DC1	49	31	1	81	51	Q
18	12	DC2	50	32	2	82	52	R
19	13	DC3	51	33	3	83	53	S
20	14	DC4	52	34	4	84	54	T
21	15	NAK	53	35	5	85	55	U
22	16	SYN	54	36	6	86	56	V
23	17	ETB	55	37	7	87	57	W
24	18	CAN	56	38	8	88	58	X
25	19	EM	57	39	9	89	59	Y
26	1A	SUB	58	3A	:	90	5A	Z
27	1B	ESC	59	3B	;	91	5B	[
28	1C	FS	60	3C	<	92	5C	\
29	1D	GS	61	3D	=	93	5D	]
30	1E	RS	62	3E	>	94	5E	^
31	1F	US	63	3F	?	95	5F	_

备注：参考资料来源：https://www.ascii-code.com/

表 18 ASCII 可显示字元表(ASCII printable characters)-2

十进制 DEC	十六进制 HEX	图形 Symbol	十进制 DEC	十六进制 HEX	图形 Symbol	十进制 DEC	十六进制 HEX	图形 Symbol
96	60	`	107	6B	k	117	75	u
97	61	a	108	6C	l	118	76	v
98	62	b	109	6D	m	119	77	w
99	63	c	110	6E	n	120	78	x
100	64	d	111	6F	o	121	79	y
101	65	e	112	70	p	122	7A	z
102	66	f	113	71	q	123	7B	{
103	67	g	114	72	r	124	7C	|
104	68	h	115	73	s	125	7D	}
105	69	i	116	74	t	126	7E	~
106	6A	j						

备注：参考资料来源：https://www.ascii-code.com/

HIWIN® 工业4.0 优质伙伴

INDUSTRIE 4.0 Best Partner



多轴机器人
Multi-Axis Robot

取放作业/组装/整列与包装/半导体/光电业/汽车工业/食品业

- 关节式机器手臂
- 并联式机器手臂
- 史卡拉机器手臂
- 晶圆机器人
- 电动夹爪
- 整合型电动夹爪
- 旋转接头



单轴机器人
Single-Axis Robot

高精密产业/半导体/
医疗自动化/FPD面板搬运

- KK, SK
- KS, KA
- KU, KE, KC



Torque Motor 回转工作台
Torque Motor Rotary Table

医疗/汽车工业/工具机/产业机械

- RAB系列
- RAS系列
- RCV系列
- RCH系列



滚珠丝杠
Ballscrew

精密研磨/精密转造

- Super S 系列 (高Dm-N值/高速化)
- Super T 系列 (低噪音/低振动)
- 微小型研磨级
- E2 环保润滑模组
- R1 螺帽旋转式
- Cool Type 节能温控丝杠
- RD 高DN节能重负荷
- 滚珠花键



直线导轨
Linear Guideway

精密机械/电子半导体/生技医疗

- 滚珠式—
  HG重负荷型, EG低组装, WE宽幅型,
  MG微小型, CG扭矩型
- 静音式—
  QH重负荷型, QE低组装型,
  QW宽幅型, QR滚柱型
- 其他—
  RG滚柱型, E2自润型, PG定位型,
  SE金属端盖型, RC强化型



特殊轴承
Bearing

工具机产业/机械手臂

- 交叉滚柱轴承
- 滚珠丝杠轴承
- 精密线性轴承
- 轴承座



谐波减速机
DATORKER® Strain Wave Gear

机器人/自动化设备/半导体设备/工具机

- 调心输入轴组合式 DSC-PO
- 调心输入轴组件式 DSC-CO
- 中空输入轴组合式 DSH-PH
- 中空输入轴密封式 DSH-AH



AC伺服电机&驱动器
AC Servo Motor & Drive

半导体设备/包装机/SMT机台/
食品业机台/LCD设备

- 驱动器—D1, D2T/D2T-LM, E1
- 伺服电机—50W~2000W



医疗设备
Medical Equipment

医疗院所/复健中心/疗养中心

- 下肢康复训练机
- 内窥镜扶持机器手臂



直线电机平台
Linear Motor Stage

自动化搬运/AOI光学检测/
精密加工/电子半导体

- 铁心式直线电机
- 无铁心式直线电机
- 棒状直线电机
- 平面电机
- 空气轴承定位平台
- X-Y平台 • 龙门系统
- 单轴直线电机定位平台



力矩电机&直驱电机
Torque Motor &
Direct Drive Motor

工具机

- 力矩电机—TM-2/IM-2, TMRW系列

检测设备/机器人

- 直驱电机—DMS, DMY, DMN, DMT
  系列

机器人软件程式指令集使用手册

出版日期：2022年07月

1. HIWIN为上銀科技的注册商标，请勿购买来路不明之仿冒品以维护您的权益。
2. 本型录所载规格、照片有时会与实际产品有所差异，包括因为改良而导致外观或规格等发生变化的情况。
3. HIWIN产品专利清单查询网址：http://www.hiwin.tw/Products/Products_patents.aspx
4. 凡受”贸易法”等法规限制之相关技术与产品，HIWIN将不会违规擅自出售。若要出口HIWIN受法律规范限制出口的产品，应根据相关法律向主管机关申请出口许可，并不得供作生产或发展核子、生化、飞弹等军事武器之用。

Copyright © HIWIN Technologies Corp.

HIWIN® 上银®

全球销售暨服务据点

上銀科技(中国)有限公司
HIWIN TECHNOLOGIES (CHINA) CORP.
江苏省苏州市苏州工业园区夏庄路2号
Tel : (0512) 8068-5599
Fax: (0512) 8068-9858
www.hiwin.cn
bussiness@hiwin.cn

德国 欧芬堡
HIWIN GmbH
OFFENBURG, GERMANY
www.hiwin.de
www.hiwin.eu

日本 神户・东京・名古屋・长野・东北・静冈・北陆・广岛・福冈・熊本
HIWIN JAPAN
KOBE・TOKYO・NAGOYA・NAGANO・TOHOKU・SHIZUOKA・HOKURIKU・HIROSHIMA・FUKUOKA・KUMAMOTO, JAPAN
www.hiwin.co.jp

美国 芝加哥
HIWIN USA
CHICAGO, U.S.A.
www.hiwin.us

意大利 米兰
HIWIN Srl
BRUGHERIO, ITALY
www.hiwin.it

瑞士 优纳
HIWIN Schweiz GmbH
JONA, SWITZERLAND
www.hiwin.ch

捷克 布尔诺
HIWIN s.r.o.
BRNO, CZECH REPUBLIC
www.hiwin.cz

法国 史特拉斯堡
HIWIN FRANCE
STRASBOURG, FRANCE
www.hiwin.fr

新加坡
HIWIN SINGAPORE
SINGAPORE
www.hiwin.sg

韩国 水原・昌原
HIWIN KOREA
SUWON・CHANGWON, KOREA
www.hiwin.kr

以色列 海法
Mega-Fabs Motion Systems, Ltd.
HAIFA, ISRAEL
www.mega-fabs.com

本型录的内容规格若有变更，恕不另行通知。

全球营运总部

上銀科技股份有限公司
HIWIN TECHNOLOGIES CORP.
台湾40852台中市精密机械园区精科路7号
Tel: +886-4-23594510
Fax: +886-4-23594420
www.hiwin.tw
business@hiwin.tw (销售)
robotservice@hiwin.tw (客服)

Copyright © HIWIN Technologies Corp.

©2022 FORM C25US907-2207 (PRINTED IN TAIWAN)