EPSON

Epson RC+ 8.0 Ver.8.0
SPEL+語言參考
(RC800系列)

翻譯版

© Seiko Epson Corporation 2024

Rev.1
TCM247S6460F

目錄

1. 前言 23

1.1 前言 24
1.2 商標 24
1.3 關於標記 24
1.4 注意 24
1.5 製造商 24
1.6 聯絡資訊 24
1.7 閱讀本手冊之前 24
1.7.1 安全性注意事項 25
1.7.2 手冊中各符號的說明 25
1.7.3 關於Epson RC+ 8.0安裝資料夾 25

2. SPEL+ 命令一覽 26

2.1 SPEL+ 命令一覽 27

3. SPEL+ 語言參考 51

3.1 SPEL+ 語言參考 52
3.2 運算子 52
3.3 ! 54
3.3.1 !...! 平行處理 54
3.4 # 57
3.4.1 #define 57
3.4.2 #ifdef...#else...#endif 59
3.4.3 #ifndef...#endif 60
3.4.4 #include 61
3.4.5 #undef 62
3.5 A 63
3.5.1 AbortMotion 63
3.5.2 Abs函數 65
3.5.3 Accel 66
3.5.4 Accel函數 68
3.5.5 AccelMax函數 69
3.5.6 AccelR 70
3.5.7 AccelR函數 71

3.5.8 AccelS 72
3.5.9 AccelS函數 74
3.5.10 Acos函數 75
3.5.11 Agl函數 76
3.5.12 AglToPls函數 77
3.5.13 AIO_In函數 78
3.5.14 AIO_InW函數 79
3.5.15 AIO_Out 80
3.5.16 AIO_Out函數 81
3.5.17 AIO_OutW 82
3.5.18 AIO_OutW函數 83
3.5.19 AIO_Set 84
3.5.20 AIO_Set函數 86
3.5.21 AIO_TrackingSet 87
3.5.22 AIO_TrackingStart 91
3.5.23 AIO_TrackingEnd 93
3.5.24 AIO_TrackingOn函數 94
3.5.25 Align函數 95
3.5.26 AlignECP函數 96
3.5.27 And運算子 97
3.5.28 AOpen 99
3.5.29 Arc, Arc3 100
3.5.30 Arch 105
3.5.31 Arch函數 108
3.5.32 AreaCorrection函數 109
3.5.33 AreaCorrectionClr 111
3.5.34 AreaCorrectionDef函數 112
3.5.35 AreaCorrectionInv函數 113
3.5.36 AreaCorrectionOffset函數 114
3.5.37 AreaCorrectionSet 115
3.5.38 Arm 117
3.5.39 Arm函數 119
3.5.40 ArmCalib 120
3.5.41 ArmCalib函數 121
3.5.42 ArmCalibClr 122

3.5.43 ArmCalibDef 123
3.5.44 ArmCalibSet 124
3.5.45 ArmCalibSet函數 125
3.5.46 ArmClr 126
3.5.47 ArmDef函數 127
3.5.48 ArmSet 128
3.5.49 ArmSet函數 132
3.5.50 Asc函數 135
3.5.51 Asin函數 136
3.5.52 AtHome函數 137
3.5.53 Atan函數 138
3.5.54 Atan2函數 139
3.5.55 ATCLR 140
3.5.56 ATRQ 141
3.5.57 ATRQ函數 142
3.5.58 AutoLJM 143
3.5.59 AutoLJM函數 145
3.5.60 AutoOrientationFlag 146
3.5.61 AutoOrientationFlag函數 148
3.5.62 AvgSpeedClear 149
3.5.63 AvgSpeed 150
3.5.64 AvgSpeed函數 151
3.5.65 AvoidSingularity 152
3.5.66 AvoidSingularity函數 154
3.6 B 155
3.6.1 Base 155
3.6.2 BClr函數 156
3.6.3 BClr64函數 157
3.6.4 BGo 158
3.6.5 BMove 160
3.6.6 Boolean 162
3.6.7 BOpen 163
3.6.8 Box 164
3.6.9 Box函數 167
3.6.10 BoxClr 168

3.6.11 BoxDef函數 169
3.6.12 Brake 170
3.6.13 Brake函數 171
3.6.14 BSet函數 172
3.6.15 BSet64函數 173
3.6.16 BTst函數 174
3.6.17 BTst64函數 175
3.6.18 Byte 176
3.7 C 177
3.7.1 Calib 177
3.7.2 Call 179
3.7.3 CalPls 180
3.7.4 CalPls函數 182
3.7.5 ChDir 183
3.7.6 ChDisk 184
3.7.7 ChDrive 186
3.7.8 ChkCom函數 187
3.7.9 ChkNet函數 188
3.7.10 Chr$函數 189
3.7.11 ClearHistory 190
3.7.12 ClearPoints 191
3.7.13 Close 192
3.7.14 CloseCom 193
3.7.15 CloseDB 194
3.7.16 CloseNet 195
3.7.17 Cls 196
3.7.18 Cnv_AbortTrack 197
3.7.19 Cnv_Accel 198
3.7.20 Cnv_Accel函數 199
3.7.21 Cnv_AccelLim 200
3.7.22 Cnv_AccelLim函數 201
3.7.23 Cnv_Adjust 202
3.7.24 Cnv_AdjustClear 203
3.7.25 Cnv_AdjustGet函數 204
3.7.26 Cnv_AdjustSet 205

3.7.27 Cnv_Downstream 206
3.7.28 Cnv_Downstream函數 207
3.7.29 Cnv_Fine 208
3.7.30 Cnv_Fine函數 209
3.7.31 Cnv_Flag函數 210
3.7.32 Cnv_LPulse函數 211
3.7.33 Cnv_Mode 212
3.7.34 Cnv_Mode函數 213
3.7.35 Cnv_Name$函數 214
3.7.36 Cnv_Number函數 215
3.7.37 Cnv_OffsetAngle 216
3.7.38 Cnv_OffsetAngle函數 217
3.7.39 Cnv_Point函數 218
3.7.40 Cnv_PosErr函數 219
3.7.41 Cnv_PosErrOffset 220
3.7.42 Cnv_Pulse函數 221
3.7.43 Cnv_QueAdd 222
3.7.44 Cnv_QueGet函數 223
3.7.45 Cnv_QueLen函數 224
3.7.46 Cnv_QueList 225
3.7.47 Cnv_QueMove 226
3.7.48 Cnv_QueReject 227
3.7.49 Cnv_QueReject函數 228
3.7.50 Cnv_QueRemove 229
3.7.51 Cnv_QueUserData 230
3.7.52 Cnv_QueUserData函數 231
3.7.53 Cnv_RobotConveyor函數 232
3.7.54 Cnv_Speed函數 233
3.7.55 Cnv_Trigger 234
3.7.56 Cnv_Upstream 235
3.7.57 Cnv_Upstream函數 236
3.7.58 CollisionDetect 237
3.7.59 CollisionDetect函數 239
3.7.60 Cont 240
3.7.61 ContManualRecover 241

3.7.62 Copy 242
3.7.63 Cos函數 243
3.7.64 CP 244
3.7.65 CP函數 246
3.7.66 CP_Offset 247
3.7.67 CP_Offset函數 250
3.7.68 Ctr函數 251
3.7.69 CTReset 252
3.7.70 CtrlDev函數 253
3.7.71 CtrlInfo函數 254
3.7.72 CurDir$函數 256
3.7.73 CurDisk$函數 257
3.7.74 CurDrive$函數 258
3.7.75 CurPos函數 259
3.7.76 Curve 260
3.7.77 CVMove 265
3.7.78 CX、CY、CZ、CU、CV、CW、CR、CS、CT 267
3.7.79 CX、CY、CZ、CU、CV、CW、CR、CS、CT函數 268
3.8 D 269
3.8.1 Date 269
3.8.2 Date$函數 270
3.8.3 Declare 271
3.8.4 DegToRad函數 274
3.8.5 Del 275
3.8.6 DeleteDB 276
3.8.7 DiffPoint函數 277
3.8.8 DiffToolOrientation函數 278
3.8.9 DispDev 279
3.8.10 DispDev函數 280
3.8.11 Dist函數 281
3.8.12 Do...Loop 282
3.8.13 Double 284
3.9 E 285
3.9.1 ECP 285
3.9.2 ECP函數 286

3.9.3 ECPClr 287
3.9.4 ECPDef函數 288
3.9.5 ECPSet 289
3.9.6 ECPSet函數 290
3.9.7 ElapsedTime函數 291
3.9.8 Elbow 292
3.9.9 Elbow函數 293
3.9.10 EncTemper 294
3.9.11 EncTemper函數 295
3.9.12 Eof函數 296
3.9.13 Era函數 297
3.9.14 EResume 298
3.9.15 Erf$函數 299
3.9.16 Erl函數 300
3.9.17 Err函數 301
3.9.18 Errb函數 302
3.9.19 ErrMsg$函數 303
3.9.20 Error 304
3.9.21 ErrorOn函數 305
3.9.22 Ert函數 306
3.9.23 EStopOn函數 307
3.9.24 Eval函數 308
3.9.25 Exit 309
3.9.26 ExportPoints 310
3.10 F 311
3.10.1 FbusIO_GetBusStatus函數 311
3.10.2 FbusIO_GetDeviceStatus函數 312
3.10.3 FbusIO_SendMsg 313
3.10.4 FileDateTime$函數 315
3.10.5 FileExists函數 316
3.10.6 FileLen函數 317
3.10.7 Find 318
3.10.8 FindPos函數 320
3.10.9 Fine 321
3.10.10 Fine函數 323

3.10.11 FineDist 324
3.10.12 FineStatus函數 326
3.10.13 Fix函數 327
3.10.14 Flush 328
3.10.15 FmtStr 329
3.10.16 FmtStr$函數 332
3.10.17 FolderExists函數 335
3.10.18 For...Next 336
3.10.19 FreeFile函數 338
3.10.20 Function...Fend 339
3.11 G 341
3.11.1 GetRobotInsideBox函數 341
3.11.2 GetRobotInsidePlane函數 342
3.11.3 Global 343
3.11.4 Go 345
3.11.5 GoSub...Return 348
3.11.6 GoTo 349
3.12 H 350
3.12.1 Halt 350
3.12.2 Hand 351
3.12.3 Hand函數 353
3.12.4 HealthCalcPeriod 354
3.12.5 HealthCalcPeriod函數 355
3.12.6 HealthCtrlAlarmOn函數 356
3.12.7 HealthCtrlInfo 357
3.12.8 HealthCtrlInfo函數 358
3.12.9 HealthCtrlRateOffset 359
3.12.10 HealthCtrlReset 360
3.12.11 HealthCtrlWarningEnable 361
3.12.12 HealthCtrlWarningEnable函數 362
3.12.13 HealthRateCtrlInfo函數 363
3.12.14 HealthRateRBInfo函數 364
3.12.15 HealthRBAlarmOn函數 365
3.12.16 HealthRBAnalysis函數 366
3.12.17 HealthRBDistance 368

3.12.18 HealthRBDistance函數 369
3.12.19 HealthRBInfo 370
3.12.20 HealthRBInfo函數 372
3.12.21 HealthRBRateOffset 374
3.12.22 HealthRBReset 375
3.12.23 HealthRBSpeed 376
3.12.24 HealthRBSpeed函數 377
3.12.25 HealthRBStart 378
3.12.26 HealthRBAnalysis 379
3.12.27 HealthRBStop 381
3.12.28 HealthRBTRQ 382
3.12.29 HealthRBTRQ函數 383
3.12.30 HealthRBWarningEnable 384
3.12.31 HealthRBWarningEnable函數 385
3.12.32 Here 386
3.12.33 Here函數 388
3.12.34 Hex$函數 389
3.12.35 History 390
3.12.36 Hofs 391
3.12.37 Hofs函數 393
3.12.38 HofsJointAccuracy 394
3.12.39 Home 396
3.12.40 HomeClr 397
3.12.41 HomeDef函數 398
3.12.42 HomeSet 399
3.12.43 HomeSet函數 401
3.12.44 Hordr 402
3.12.45 Hordr函數 404
3.12.46 Hour 405
3.12.47 Hour函數 406
3.13 I 407
3.13.1 If...Then...Else...EndIf 407
3.13.2 ImportPoints 410
3.13.3 In函數 411
3.13.4 InBCD函數 413

3.13.5 Inertia 415
3.13.6 Inertia函數 416
3.13.7 InPos函數 417
3.13.8 Input 418
3.13.9 Input # 420
3.13.10 InputBox 422
3.13.11 InReal函數 423
3.13.12 InsideBox函數 424
3.13.13 InsidePlane函數 426
3.13.14 InStr函數 428
3.13.15 Int函數 429
3.13.16 Int32 430
3.13.17 Int64 431
3.13.18 Integer 432
3.13.19 InW函數 433
3.13.20 IODef函數 434
3.13.21 IOLabel$函數 435
3.13.22 IONumber函數 436
3.14 J 437
3.14.1 J1Angle 437
3.14.2 J1Angle函數 438
3.14.3 J1Flag 439
3.14.4 J1Flag函數 440
3.14.5 J2Flag 441
3.14.6 J2Flag函數 442
3.14.7 J4Angle 443
3.14.8 J4Angle函數 444
3.14.9 J4Flag 445
3.14.10 J4Flag函數 446
3.14.11 J6Flag 447
3.14.12 J6Flag函數 448
3.14.13 JA函數 449
3.14.14 Joint 450
3.14.15 JointAccuracy 451
3.14.16 JointAccuracy函數 452

3.14.17 JRange 453
3.14.18 JRange函數 454
3.14.19 JS函數 455
3.14.20 JT函數 456
3.14.21 JTran 457
3.14.22 Jump 458
3.14.23 Jump3, Jump3CP 463
3.14.24 JumpTLZ 468
3.15 L 471
3.15.1 LatchEnable 471
3.15.2 LatchState函數 472
3.15.3 LatchPos函數 473
3.15.4 LCase$函數 475
3.15.5 Left$函數 476
3.15.6 Len函數 477
3.15.7 LimitTorque 478
3.15.8 LimitTorque函數 480
3.15.9 LimitTorqueLP 481
3.15.10 LimitTorqueLP函數 483
3.15.11 LimitTorqueStop 484
3.15.12 LimitTorqueStop函數 485
3.15.13 LimitTorqueStopLP 486
3.15.14 LimitTorqueStopLP函數 487
3.15.15 LimZ 488
3.15.16 LimZ函數 489
3.15.17 LimZMargin 490
3.15.18 LimZMargin函數 491
3.15.19 Line Input 492
3.15.20 Line Input # 493
3.15.21 LJM函數 494
3.15.22 LoadPoints 499
3.15.23 Local 500
3.15.24 Local函數 503
3.15.25 LocalClr 504
3.15.26 LocalDef函數 505

3.15.27 Lof函數 506
3.15.28 LogIn 507
3.15.29 Long 508
3.15.30 LSet$函數 509
3.15.31 LShift函數 510
3.15.32 LShift64函數 511
3.15.33 LTrim$函數 512
3.16 M 513
3.16.1 Mask運算子 513
3.16.2 MCal 514
3.16.3 MCalComplete函數 515
3.16.4 MCordr 516
3.16.5 MCordr函數 519
3.16.6 MemIn函數 520
3.16.7 MemInW函數 522
3.16.8 MemOff 523
3.16.9 MemOn 525
3.16.10 MemOut 527
3.16.11 MemOutW 529
3.16.12 MemSw函數 530
3.16.13 MHour函數 531
3.16.14 Mid$函數 532
3.16.15 MkDir 533
3.16.16 Mod運算子 534
3.16.17 Motor 535
3.16.18 Motor函數 536
3.16.19 Move 537
3.16.20 MsgBox 540
3.16.21 MyTask函數 542
3.17 N 543
3.17.1 Next 543
3.17.2 Not運算子 545
3.18 O 546
3.18.1 Off 546
3.18.2 OLAccel 548

3.18.3 OLAccel函數 550
3.18.4 OLRate 551
3.18.5 OLRate函數 552
3.18.6 On 553
3.18.7 OnErr 555
3.18.8 OpBCD 556
3.18.9 OpenDB 559
3.18.10 OpenCom 561
3.18.11 OpenCom函數 562
3.18.12 OpenNet 563
3.18.13 OpenNet函數 564
3.18.14 Oport函數 565
3.18.15 Or運算子 567
3.18.16 Out 568
3.18.17 Out函數 571
3.18.18 OutReal 572
3.18.19 OutReal函數 573
3.18.20 OutW 574
3.18.21 OutW函數 575
3.19 P 576
3.19.1 P# (1. 定義點) 576
3.19.2 P# (2. 點指定) 578
3.19.3 PAgl函數 581
3.19.4 Pallet 582
3.19.5 Pallet函數 585
3.19.6 PalletClr 588
3.19.7 ParseStr/ParseStr函數 589
3.19.8 Pass 590
3.19.9 Pause 592
3.19.10 PauseOn函數 593
3.19.11 PDef函數 594
3.19.12 PDel 595
3.19.13 PDescription 596
3.19.14 PDescription$函數 597
3.19.15 PeakSpeedClear 598

3.19.16 PeakSpeed 599
3.19.17 PeakSpeed函數 600
3.19.18 PerformMode 601
3.19.19 PerformMode函數 603
3.19.20 PG_FastStop 604
3.19.21 PG_LSpeed 605
3.19.22 PG_LSpeed函數 607
3.19.23 PG_Scan 608
3.19.24 PG_SlowStop 609
3.19.25 PLabel 610
3.19.26 PLabel$函數 611
3.19.27 Plane 612
3.19.28 Plane函數 614
3.19.29 PlaneClr 615
3.19.30 PlaneDef函數 616
3.19.31 PList 617
3.19.32 PLocal 619
3.19.33 PLocal函數 620
3.19.34 Pls函數 621
3.19.35 PNumber函數 622
3.19.36 PosFound函數 623
3.19.37 Power 624
3.19.38 Power函數 626
3.19.39 PPls函數 627
3.19.40 Print 628
3.19.41 Print # 630
3.19.42 ProjectName$函數 632
3.19.43 PTCLR 633
3.19.44 PTPBoost 634
3.19.45 PTPBoost函數 635
3.19.46 PTPBoostOK函數 636
3.19.47 PTPTime函數 637
3.19.48 PTran 638
3.19.49 PTRQ 639
3.19.50 PTRQ函數 640

3.19.51 Pulse 641
3.19.52 Pulse函數 643
3.20 Q 644
3.20.1 QP 644
3.20.2 QPDecelR 645
3.20.3 QPDecelR函數 646
3.20.4 QPDecelS 647
3.20.5 QPDecelS函數 648
3.20.6 Quit 649
3.21 R 651
3.21.1 RadToDeg函數 651
3.21.2 Randomize 652
3.21.3 Range 653
3.21.4 Read 655
3.21.5 ReadBin 656
3.21.6 Real 657
3.21.7 RealAccel函數 658
3.21.8 RealPls函數 659
3.21.9 RealPos函數 660
3.21.10 RealTorque函數 661
3.21.11 Recover 662
3.21.12 Recover函數 664
3.21.13 RecoverPos函數 666
3.21.14 Redim 667
3.21.15 Rename 668
3.21.16 RenDir 669
3.21.17 Reset 670
3.21.18 ResetElapsedTime 672
3.21.19 Restart 673
3.21.20 Resume 674
3.21.21 Return 675
3.21.22 Right$函數 676
3.21.23 RmDir 677
3.21.24 Rnd函數 678
3.21.25 Robot 679

3.21.26 Robot函數 680
3.21.27 RobotInfo函數 681
3.21.28 RobotInfo$函數 683
3.21.29 RobotModel$函數 684
3.21.30 RobotName$函數 685
3.21.31 RobotSerial$函數 686
3.21.32 RobotType函數 687
3.21.33 ROpen 688
3.21.34 ROTOK函數 689
3.21.35 RSet$函數 690
3.21.36 RShift函數 691
3.21.37 RShift64函數 692
3.21.38 RTrim$函數 693
3.21.39 RunDialog 694
3.22 S 695
3.22.1 SafetyOn函數 695
3.22.2 SavePoints 696
3.22.3 Seek 697
3.22.4 Select...Send 698
3.22.5 SelectDB 699
3.22.6 Sense 701
3.22.7 SetCom 703
3.22.8 SetLatch 705
3.22.9 SetIn 707
3.22.10 SetInReal 708
3.22.11 SetInW 709
3.22.12 SetNet 710
3.22.13 SetSw 711
3.22.14 SF_GetParam函數 712
3.22.15 SF_GetParam$函數 719
3.22.16 SF_GetStatus函數 720
3.22.17 SF_LimitSpeedS 726
3.22.18 SF_LimitSpeedS函數 728
3.22.19 SF_LimitSpeedSEnable 729
3.22.20 SF_LimitSpeedSEnable函數 731

3.22.21 SF_PeakSpeedS 732
3.22.22 SF_PeakSpeedS函數 733
3.22.23 SF_PeakSpeedSClear 734
3.22.24 SF_RealSpeedS 735
3.22.25 SF_RealSpeedS函數 736
3.22.26 SFree 737
3.22.27 SFree函數 739
3.22.28 Sgn函數 740
3.22.29 Short 741
3.22.30 Signal 742
3.22.31 SimGet 743
3.22.32 SimSet 746
3.22.33 Sin函數 751
3.22.34 SingularityAngle 752
3.22.35 SingularityAngle函數 753
3.22.36 SingularityDist 754
3.22.37 SingularityDist函數 755
3.22.38 SingularitySpeed 756
3.22.39 SingularitySpeed函數 757
3.22.40 SLock 758
3.22.41 SoftCP 759
3.22.42 SoftCP函數 760
3.22.43 Space$函數 761
3.22.44 Speed 762
3.22.45 Speed函數 764
3.22.46 SpeedFactor 765
3.22.47 SpeedFactor函數 766
3.22.48 SpeedR 767
3.22.49 SpeedR函數 768
3.22.50 SpeedRLimitation 769
3.22.51 SpeedRLimitation函數 770
3.22.52 SpeedS 771
3.22.53 SpeedS函數 773
3.22.54 Sqr函數 774
3.22.55 ST函數 775

3.22.56 StartMain 776
3.22.57 Stat函數 777
3.22.58 Str$函數 779
3.22.59 String 780
3.22.60 Sw函數 782
3.22.61 SyncLock 783
3.22.62 SyncUnlock 785
3.22.63 SyncRobots 786
3.22.64 SyncRobots函數 787
3.22.65 SysConfig 788
3.22.66 SysErr函數 790
3.23 T 792
3.23.1 Tab$函數 792
3.23.2 Tan函數 793
3.23.3 TargetOK函數 794
3.23.4 TaskDone函數 795
3.23.5 TaskInfo函數 796
3.23.6 TaskInfo$函數 798
3.23.7 TaskState函數 799
3.23.8 TaskWait 800
3.23.9 TC 801
3.23.10 TCLim 804
3.23.11 TCLim函數 806
3.23.12 TCPSpeed函數 807
3.23.13 TCSpeed 808
3.23.14 TCSpeed函數 809
3.23.15 TeachOn函數 810
3.23.16 TGo 811
3.23.17 Till 813
3.23.18 TillOn函數 815
3.23.19 Time 816
3.23.20 Time函數 817
3.23.21 Time$函數 818
3.23.22 TLClr 819
3.23.23 TLDef函數 820

3.23.24 TLSet 821
3.23.25 TLSet函數 825
3.23.26 TMOut 826
3.23.27 TMove 827
3.23.28 Tmr函數 829
3.23.29 TmReset 830
3.23.30 Toff 831
3.23.31 Ton 832
3.23.32 Tool 833
3.23.33 Tool函數 834
3.23.34 Trap(使用者定義觸發) 835
3.23.35 Trap(系統狀態觸發) 838
3.23.36 Trim$函數 841
3.23.37 TW函數 842
3.24 U 843
3.24.1 UBound函數 843
3.24.2 UByte 844
3.24.3 UCase$函數 845
3.24.4 UInt32 846
3.24.5 Uint64 847
3.24.6 UOpen 848
3.24.7 UpdateDB 849
3.24.8 UShort 850
3.25 V 851
3.25.1 Val函數 851
3.25.2 VSD 852
3.25.3 VSD函數 853
3.25.4 VxCalib 854
3.25.5 VxCalDelete 859
3.25.6 VxCalLoad 860
3.25.7 VxCalInfo函數 861
3.25.8 VxCalSave 862
3.25.9 VxTrans函數 863
3.26 W 864
3.26.1 Wait 864

3.26.2 WaitNet 867
3.26.3 WaitPos 868
3.26.4 WaitSig 869
3.26.5 Weight 870
3.26.6 Weight函數 872
3.26.7 Where 873
3.26.8 WindowsStatus函數 874
3.26.9 WOpen 875
3.26.10 WorkQue_Add 877
3.26.11 WorkQue_AutoRemove 878
3.26.12 WorkQue_AutoRemove函數 879
3.26.13 WorkQue_Get函數 880
3.26.14 WorkQue_Len函數 881
3.26.15 WorkQue_List 882
3.26.16 WorkQue_Reject 883
3.26.17 WorkQue_Reject函數 884
3.26.18 WorkQue_Remove 885
3.26.19 WorkQue_Sort 886
3.26.20 WorkQue_Sort函數 887
3.26.21 WorkQue_UserData 888
3.26.22 WorkQue_UserData函數 889
3.26.23 Wrist 890
3.26.24 Wrist函數 891
3.26.25 Write 892
3.26.26 WriteBin 893
3.27 X 894
3.27.1 Xor運算子 894
3.27.2 Xqt 895
3.27.3 XY函數 902
3.27.4 XYLim 903
3.27.5 XYLim函數 905
3.27.6 XYLimClr 906
3.27.7 XYLimDef函數 907
3.27.8 XYLimMode 908
3.27.9 XYLimMode函數 910

4. Appendix A：SPEL+命令 使用條件一覽 911

4.1 Appendix A：SPEL+命令 使用條件一覽 912

5. Appendix B：相容性相關注意事項 932

5.1 B-1：與Epson RC+ 6.0的相容性相關注意事項 933

5.1.1 概要 933

5.1.2 一般差異 933

5.1.3 命令相容性一覽 935

5.2 B-2：與Epson RC+ 5.0的相容性相關注意事項 947

5.2.1 概要 947

5.2.2 一般差異 947

5.2.3 命令相容性一覽 949

5.2.4 Epson RC+ Ver.4.*的轉移命令一覽(未支援Epson RC+5.0) 960

6. Appendix C：Epson RC+ 8.0的命令 962

6.1 C-1：Epson RC+ 4.0以後版本增加的命令一覽 963

6.2 C-2：Epson RC+ 8.0各版本增加的命令一覽 968

6.3 C-3：EPSON RC+ 7.0各版本增加的命令一覽 968

6.4 C-4：EPSON RC+ 7.0各版本刪除的命令一覽 974

7. Appendix D：常數 975

7.1 Appendix D：常數 976

1. 前言

1.1 前言

感謝您購買我們的機器人產品。
本手冊包含正確使用Epson RC+軟體的必要資訊。
在安裝機器人系統之前，請詳閱本手冊及其他相關手冊。
請妥善保管本手冊以供隨時取用。

所有機器人系統與其選配部件經嚴格的品質控管、測試與檢驗，以確保其符合我們的高效能標準，始能出貨給貴客戶。
請注意，若未依本手冊說明的使用條件與產品規格使用本機器人系統，將無法發揮產品的基本性能。

本手冊說明我們可預測的可能危險及後果。務必遵守本手冊的安全注意事項，確保安全及正確地使用機器人系統。

1.2 商標

Microsoft、Windows、Windows 圖標是美國 Microsoft Corporation 在美國及其他國家的註冊商標或商標。

Pentium是美國Intel公司的商標。

其他公司名稱, 品牌名稱, 產品名稱是各公司的註冊商標或商標。

1.3 關於標記

- Microsoft® WindowsR 10 operating system
- Microsoft® WindowsR 11 operating system

本使用說明書將上述操作系統分別標記爲Windows 10, Windows 11。另外，有時可能將Windows 10, Windows 11統一標記為Windows。

1.4 注意

如未獲授權，不得複製或重製本手冊中的任何部分。
本手冊內容如有變更，恕不另行通知。
若您在本手冊中發現任何錯誤或對相關內容有任何意見，請告知我們。

1.5 製造商

SEIKO EPSON CORPORATION

1.6 聯絡資訊

如需詳細資訊，請參閱下列手冊中的"供應商”。
「安全手冊」

1.7 閱讀本手冊之前

本節說明在閱讀本手冊前您應先瞭解的資訊。

1.7.1 安全性注意事項

安裝與運送機器人及其設備須由合格人員執行，且應遵守所有國家和當地法規。

在安裝機器人系統或連接電纜之前，請閱讀本手冊及其他相關手冊。

請妥善保管本手冊以供隨時取用。

1.7.2 手冊中各符號的說明

⚠ 警告

此符號代表若不正確遵守相關指示，可能會有重傷或死亡的危險。

⚠ 注意

此符號代表若不正確遵守相關指示，可能會有人員受傷或設備及設施受損的危險。

1.7.3 關於Epson RC+ 8.0安裝資料夾

Epson RC+ 8.0允許您將安裝資料夾路徑更改為任何位置。本手冊假定Epson RC+ 8.0安裝在 `C:\EpsonRC80` 上進行說明。

2. SPEL+ 命令一覽

2.1 SPEL+ 命令一覽

SPEL+ 命令一覽如下所述。

系統管理相關命令

命令	說明
Reset	用於將控制器重設為初始狀態
SysConfig	用於顯示系統設定參數
SysErr	用於傳回最新錯誤狀態或警告狀態
Date	用於顯示日期
Time	用於顯示時間
Date$	是以字串傳回日期的函數
Time$	是以字串傳回時間的函數
Hour	用於顯示控制器的累計通電時間
Stat	用於傳回控制器的狀態位元
CtrlInfo	用於傳回控制器資訊
RobotInfo	用於傳回機器人資訊
RobotInfo$	用於傳回機器人的文字資訊
TaskInfo	用於傳回工作資訊
TaskInfo$	用於傳回工作的文字資訊
DispDev	用於設定目前顯示裝置
EStopOn	用於傳回緊急停止狀態
CtrlDev	用於傳回目前控制裝置的編號
Cls	用於清除Run視窗、操作員視窗、命令視窗的文字區域，或者清除TP列印面板
Toff	用於關閉LCD上的執行行顯示
Ton	用於開啟LCD上的執行行顯示
SafetyOn	用於傳回安全門的狀態
Eval	用於執行命令視窗的陳述式，並傳回錯誤狀態
ShutDown	用於關閉Epson RC+，並關閉Windows或重新啟動
TeachOn	用於傳回教導模式的狀態

命令	說明
WindowsStatus	用於傳回Windows的啟動狀態
History	用於顯示命令的執行歷史記錄
ClearHistory	用於刪除所有命令執行歷史記錄

機器人控制相關命令

命令	說明
AIO_TrackingSet	用於設定距離追蹤功能
AIO_TrackingStart	用於啟用距離追蹤功能
AIO_TrackingEnd	用於停用距離追蹤功能
AIO_TrackingOn函數	是用於傳回距離追蹤功能狀態的函數
AtHome	是用於傳回目前機器人姿態是否為Home姿態的函數
Calib	用於以使用Calpls設定的脈衝值取代目前手臂姿態的脈衝值
CalPls	是用於設定和顯示用於校正的位置姿態脈衝值或傳回設定值的函數
Hofs	用於設定和顯示從編碼器原點到軟體原點的補償脈衝
JointAccuracy	用於設定和顯示關節精度補償之補償值
HofsJointAccuracy	設定與顯示從編碼器原點到軟體原點的補償脈衝，但不變更關節精度補償之補償值
MCal	用於帶增量型編碼器機器人的原點復歸操作(機械原點檢測)
MCalComplete	用於傳回MCal的狀態
MCordr	是用於指定和顯示以MCaL命令執行原點復歸時的關節動作順序，或傳回設定值的函數
EncTemper	顯示編碼器溫度
EncTemper函數	用於傳回編碼器溫度的函數
Power	用於將功率模式設為High或Low，並顯示目前模式
MHour函數	是用於傳回機器人馬達累計勵磁時間的函數
Motor	用於開啟或關閉機器人所有軸的馬達
SFree	用於設定指定關節的SFree狀態
SLock	用於解除指定關節的SFree狀態
SyncRobots	用於開始動作預約中的機器人動作
Jump	用於以門形動作移至目標坐標的動作(PTP動作)
Jump3	用於以三維門形動作移至目標坐標的動作(PTP動作+CP動作)
Jump3CP	用於以三維門形動作移至目標坐標的動作(CP動作)

命令	說明
JumpTLZ	用於以三維門形動作移至目標坐標的動作(PTP動作+CP動作)
Arch	是用於設定和顯示Jump命令下的Arch參數，並傳回設定值的函數
LimZ	是用於設定Jump命令時的第3關節高度(Z坐標值)初始值，並傳回設定值的函數
LimZMargin	是用於設定在高於LimZ設定值的位置開始動作時的錯誤檢測邊界，並傳回設定值的函數
Sense	用於設定和顯示以Jump命令指定Sense時要停在目標坐標上方的條件
JS	是執行Jump Sense後，用於傳回Sense輸入是否成立的函數
JT	是用於傳回在此之前Jump動作結果的函數
Go	用於從目前位置到指定位置之間進行PTP動作
Pass	用於通過(不停止)指定點附近的PTP動作
Pulse	用於透過PTP控制，將手臂移動到以各關節脈衝值指定的點
BGo	用於所選本地坐標系中的位移PTP動作
BMove	用於本地坐標系中的位移線性動作
TGo	用於工具坐標系中的位移PTP動作
TMove	用於工具坐標系中的位移線性動作
Till	用於設定和顯示以動作命令指定Till時在動作途中停止並結束處理的條件
TillOn	用於傳回Till的狀態
!…!	是在動作中平行執行I/O等輸入輸出處理的功能
Speed	是用於設定和顯示PTP動作速度，並傳回設定值的函數
Accel	是用於設定和顯示PTP動作的加減速度，並傳回設定值的函數
SpeedFactor	是用於設定和顯示PTP動作速度，並傳回設定值的函數
Inertia	用於設定機器人的負載慣性和偏心量
Weight	用於設定和顯示用於補償PTP動作時的速度/加減速度之參數
Arc	用於XY平面上的曲線動作(CP動作)
Arc3	用於三維曲線動作(CP動作)
Move	用於線性動作(CP動作)
Curve	用於建立以自由曲線進行CP控制的資料和點
CVMove	用於執行以Curve命令定義的自由曲線CP動作
SpeedS	是用於設定和顯示CP動作速度，並傳回設定值的函數
AccelS	是用於設定和顯示CP動作加減速度，並傳回設定值的函數

命令	說明
SpeedR	是用於設定和顯示工具姿態變化的速度，並傳回設定值的函數
AccelR	是用於設定和顯示工具姿態變化的加減速度，並傳回設定值的函數
SpeedRLimitation	是用於設定和顯示工具姿態變化速度的動作速度限制，並傳回設定值的函數
AccelMax	是用於傳回可以Accel設定的加減速度最大值的函數
Brake	用於開啟和關閉目前機器人所指定關節的制動器
Home	用於移至Home位置(等待姿態)(PTP動作)
HomeClr	用於清除Home位置的定義
HomeDef	用於傳回Home位置的定義
HomeSet	用於設定和顯示Home位置(等待位置)的脈衝
Hordr	是用於指定和顯示執行Home時的各關節動作順序，並傳回設定值的函數
InPos	是用於傳回機器人定位狀態的函數
CurPos	用於傳回機器人目前動作目標位置
TCPSpeed	用於傳回計算獲得的目前工具中心點速度
Pallet	用於定義棧板和顯示定義棧板
PalletClr	用於清除棧板的定義
Fine	用於設定和顯示結束目標位置定位的判斷範圍(單位：pulse)
FineDist	用於設定和顯示結束目標位置定位的判斷範圍(單位：mm)
FineStatus函數	用於以整數值傳回使用Fine或FineDist
QP	是用於設定/解除快速暫停功能，顯示目前設定，並傳回設定值的函數
QPDecelR	用於針對CP動作時的工具姿態變化設定快速暫停減速度
QPDecelS	用於設定CP動作時的快速暫停減速度
CP	用於設定路徑運動
Box	用於設定和顯示進入檢測區
BoxClr	用於清除設定的Box
BoxDef	用於傳回有無設定進入檢測區
Plane	用於設定和顯示進入檢測平面
PlaneClr	用於清除進入檢測平面(未定義)
PlaneDef	用於傳回進入檢測平面的設定狀態

命令	說明
InsideBox	用於傳回進入檢測區的檢測狀態
InsidePlane	用於傳回進入檢測平面的檢測狀態
GetRobotInsideBox	用於傳回進入到進入檢測區的機器人
GetRobotInsidePlane	用於傳回進入到進入檢測平面的機器人
Find	用於設定和顯示在動作命令中儲存坐標的條件
FindPos	用於在執行動作命令中傳回以Find儲存的坐標
PosFound	用於傳回執行Find命令時的狀態
WaitPos	即便路徑運動處於啟用狀態，在執行下一陳述式前仍等待機器人減速並停止
Robot	用於選擇機器人(以Robot函數傳回機器人編號)
RobotModel$	用於傳回機器人的型號
RobotName$	用於傳回機器人名稱
RobotSerial$	用於傳回機器人序號
RobotType	用於傳回機器人類型
TargetOK	用於傳回可否從目前位置朝目標坐標進行PTP(point to point)動作
ROTOK函數	用於傳回至目標坐標的動作命令時，可否附加ROT修飾參數
JRange	是用於以脈衝值設定指定關節容許動作區，並傳回設定值的函數
Range	用於以脈衝值設定和顯示容許動作區
XYLim	是用於設定和顯示XY平面容許動作區，並傳回設定值的函數
XYLimClr	用於清除設定的XYLim
XYLimDef	用於傳回是否設定XYLim
XYLimMode	適用於設定和顯示XYLim的監視方法，並傳回設定值的函數
XY	用於將指定的坐標值作為點資料傳回
Dist	用於傳回2個機器人坐標間的距離
DiffToolOrientation函數	用於傳回工具坐標系每個坐標軸的角度
DiffPoint函數	用於傳回指定的2個點之差值
PTPBoost	是用於進行PTP動作移動微小距離時的設定和顯示並傳回設定值的函數
PTPBoostOK	傳回從目前位置至目標座標的PTP(point to point)動作是否為微小距離的移動
PTPTime	用於傳回PTP動作命令的估計所需時間，而不執行PTP動作

命令	說明
CX	是用於設定指定點資料的X坐標值並傳回設定值的函數
CY	是用於設定指定點資料的Y坐標值並傳回設定值的函數
CZ	是用於設定指定點資料的Z坐標值並傳回設定值的函數
CU	是用於設定指定點資料的U坐標值並傳回設定值的函數
CV	是用於設定指定點資料的V坐標值並傳回設定值的函數
CW	是用於設定指定點資料的W坐標值並傳回設定值的函數
CR	是用於設定指定點資料的R坐標值並傳回設定值的函數
CS	是用於設定指定點資料的S坐標值並傳回設定值的函數
CT	是用於設定指定點資料的T坐標值並傳回設定值的函數
Pls	是用於傳回各關節脈衝值的函數
Agl	是用於傳回指定關節的角度或位置的函數
PAgl	是用於根據指定的坐標值計算關節位置(旋轉關節角度、直動關節位置)並進行傳回的函數
JA	用於傳回根據關節角度計算的機器人坐標
AglToPls	用於將機器人各關節角度轉換為脈衝
DegToRad	用於將角度轉換為弧度
RadToDeg	用於將弧度轉換為角度
Joint	用於在關節坐標系中顯示目前機器人位置
JTran	用於進行無須原點復歸的一個關節的PTP動作
PTran	用於以根據目前位置指定的脈衝量進行一個關節的PTP動作
RealPls	用於傳回指定關節的脈衝值
RealPos	用於傳回指定機器人的目前位置
RealAccel函數	是用於傳回以OLAccel自動調整的Accel值的函數
PPls	是用於根據指定的坐標值計算關節脈衝並進行傳回的函數
LJM函數	用於傳回對姿態旗標進行轉換的點資料，以確保最小的關節移動量(相對於指定點，從參照點來看的情況下)
AutoLJM	用於設定自動LJM
AutoLJM函數	是用於傳回自動LJM狀態的函數
AutoOrientationFlag	用於變更N6-A1000**的姿態旗標
AutoOrientationFlag函數	是用於傳回AutoOrientationFlag狀態的函數
AvoidSingularity	用於設定奇點迴避功能

命令	說明
AvoidSingularity函數	是用於傳回奇點迴避功能狀態的函數
SingularityAngle	用於設定奇點迴避功能的奇點附近角度
SingularityAngle函數	是用於傳回奇點迴避功能的奇點附近角度的函數
SingularitySpeed	用於設定奇點迴避功能的奇點附近速度
SingularitySpeed函數	是用於傳回奇點迴避功能的奇點附近速度的函數
SingularityDist	用於設定奇點通過功能所需的特定點附近距離
SingularityDist函數	是用於傳回奇點通過功能所需的特定點附近距離的函數
AbortMotion	用於中斷動作命令，並將正在執行動作的工作設為錯誤
Align函數	用於傳回能夠將機器人姿態對準本地坐標系中最近的坐標軸的點數據
AlignECP函數	是用於傳回已轉換的點資料，使機器人姿態在ECP坐標系的最接近坐標軸上排成一列之函數
SoftCP	用於設定和顯示SoftCP動作模式
SoftCP函數	是用於傳回SoftCP動作模式狀態的函數
Here	用於將目前位置作為機器人坐標進行教導
Where	用於顯示機器人目前位置資料
PerformMode	用於設定機器人動作模式
PerformMode函數	用於傳回機器人動作模式編號
VSD	用於設定SCARA機器人的變速CP動作功能
VSD函數	用於設定SCARA機器人的變速CP動作功能
CP_Offset	用於設定CP On時的後續動作命令之開始位移時間
CP_Offset函數	用於傳回CP On時的後續動作命令開始位移時間
AvgSpeedClear	用於清除關節平均速度並進行初始化
AvgSpeed	用於顯示關節平均速度
AvgSpeed函數	是用於傳回關節平均速度的函數
PeakSpeedClear	用於清除關節峰值速度並進行初始化
PeakSpeed	用於顯示關節峰值速度
PeakSpeed函數	是用於傳回關節峰值速度的函數

扭矩相關命令

命令	說明
TC	是用於設定扭矩控制模式並顯示目前模式的函數

命令	說明
TCSpeed	是用於設定扭矩控制中的速度限制值的函數
TCLim	是用於設定為扭矩控制模式的各關節扭矩限制值的函數
RealTorque	是用於傳回指定關節的目前扭矩命令值的函數
ATCLR	用於清除關節的有效扭矩並進行初始化
ATRQ	用於顯示指定關節的有效扭矩值
PTCLR	用於清除關節的峰值扭矩並進行初始化
PTRQ	用於顯示指定關節的峰值扭矩
OLAccel	用於設定與過載率相應的加減速度自動調整
OLRate	用於顯示指定關節的過載率
LimitTorque	本函數用於設定高功率模式時的扭矩上限值並傳回設定值。
LimitTorque函數	是用於傳回LimitTorque命令設定值的函數
LimitTorqueLP	本函數用於設定低功率模式時的扭矩上限值並傳回設定值。
LimitTorqueLP函數	是用於傳回LimitTorqueLP命令設定值的函數
LimitTorqueStop	是用於設定在高功率模式下達到扭矩上限時是否停止機器人並傳回設定值的函數
LimitTorqueStop函數	是用於傳回LimitTorqueStop命令設定值的函數
LimitTorqueStopLP	是用於設定在低功率模式下達到扭矩上限時是否停止機器人並傳回設定值的函數
LimitTorqueStopLP函數	是用於傳回LimitTorqueStopLP命令設定值的函數

輸入輸出相關命令

命令	說明
On	用於開啟輸出位元。
Off	用於關閉輸出位元。
Oport	用於載入輸出位元的狀態
Sw	用於進行I/O輸入或載入記憶體I/O的狀態並在I/O使用。
In	用於載入1位元組(8位元)的輸入資料並在I/O使用。
InW	用於載入1字組(16位元)的輸入資料並在I/O使用。
InBCD	用於以BCD(二進制編碼的十進制數)格式載入輸入連接埠的輸入狀態
Out	用於輸出1位元組(8位元)的輸出資料並在I/O使用。
OutW	用於輸出1字組(16位元)的輸出資料並在I/O使用。
OpBCD	用於以BCD格式向輸出連接埠輸出1位元組資料

命令	說明
MemOn	是開啟以位元編號指定的記憶體I/O的命令
MemOff	是關閉以位元編號指定的記憶體I/O的命令
MemSw	是傳回已指定記憶體I/O位元狀態的函數
MemIn	是以位元組為單位傳回記憶體I/O狀態的函數
MemOut	是以位元組為單位變更記憶體I/O狀態的命令
MemInW	用於以字組為單位傳回記憶體I/O連接埠的狀態，並以16個記憶體I/O位元構成字組連接埠
MemOutW	用於以字組為單位，同時以16位元設定記憶體I/O連接埠的狀態
Wait	等待條件或事件
TMOut	用於設定執行Wait命令時的超時時間
TW	是傳回是否已成立Wait命令條件運算式的函數
Input	用於接收來自顯示裝置的輸入，並儲存為變數
InReal	用於將2字組(32位元)的輸入資料作為32位元浮點數資料(符合IEEE754)進行載入。在I/O使用。
Print	用於在輸出畫面、命令畫面、操作員視窗上顯示資料
Line Input	用於載入1行輸入資料，並將該資料指派給字串變數
Input#	用於從檔案、通訊連接埠、資料庫或裝置輸入字串或數值資料，並儲存為變數
Print#	用於將資料輸出到指定的檔案、通訊連接埠、資料庫和裝置
Line Input#	用於從檔案、通訊連接埠、資料庫和裝置載入1行資料
Lof	用於傳回指定的RS-232C連接埠或TCP/IP連接埠的緩衝區所接收資料的行數
SetIn	用於設定虛擬I/O的輸入連接埠(8位元)
SetInW	用於設定虛擬I/O的輸入連接埠(16位元)
SetSw	用於設定虛擬I/O的輸入
IOLabel$	用於傳回指定的輸入/輸出位元、位元組、字組的I/O標籤
IONumber	用於傳回指定I/O標籤的I/O連接埠編號
IODef	用於傳回是否定義指定的I/O標籤
OpenCom	用於開啟RS-232C連接埠
OpenCom函數	是取得實施OpenCom的工作編號之函數
CloseCom	用於關閉以OpenCom開啟的RS-232C連接埠
SetCom	用於設定或顯示RS-232C連接埠的參數
ChkCom	用於傳回通訊連接埠接收緩衝區內的字元數

OpenNet	用於開啟TCP/IP網路連接埠
OpenNet函數	是取得實施OpenNet的工作編號之函數
OutReal	用於將實數值的輸出資料作為32位元浮點數資料(符合IEEE754)，輸出到輸出連接埠2字組(32位元)。在I/O使用。
CloseNet	用於關閉以OpenNet開啟的TCP/IP連接埠
SetNet	用於設定TCP/IP連接埠的參數
ChkNet	用於傳回網路連接埠接收緩衝區內的字元數
WaitNet	用於等待確立TCP/IP連接埠的連接
Read	用於從檔案或通訊連接埠載入指定的字元數
ReadBin	用於從檔案或通訊連接埠載入二進位資料
Write	用於將字串寫入檔案或通訊連接埠，而行尾不附加結束字元
WriteBin	用於將二進位資料寫出到檔案或通訊連接埠中
InputBox	用於顯示輸入對話方塊並傳回已輸入內容
MsgBox	用於顯示對話方塊訊息
RunDialog	用於以SPEL+程式啟動Epson RC+畫面
LatchEnable	用於啟用/停用以R-I/O輸入對機器人位置進行閂鎖的功能
LatchState函數	用於傳回以R-I/O對機器人位置進行閂鎖的狀態
LatchPos函數	用於傳回以R-I/O輸入信號進行閂鎖的機器人位置
SetLatch	用於設定以R-I/O輸入對機器人位置進行閂鎖的功能
AIO_In函數	用於從類比I/O輸入通道載入類比值
AIO_InW函數	用於從類比I/O輸入通道載入1字組的輸入資料
AIO_Out	用於對類比I/O輸出通道輸出類比值
AIO_Out函數	用於傳回類比I/O輸出通道的輸出狀態
AIO_OutW	用於將1字組的資料輸出到類比I/O輸出通道
AIO_OutW函數	用於以1字組傳回類比I/O輸出通道的輸出狀態
AIO_Set	用於向類比I/O輸出通道輸出速度資訊
AIO_Set函數	用於傳回選項的類比I/O輸出通道中所設定的機器人速度輸出設定資訊

點控制相關命令

命令	說明
ClearPoints	用於刪除記憶體中的位置資料

命令	說明
LoadPoints	用於將點檔案載入到機器人的點記憶區中
SavePoints	用於將主記憶體中的點資料儲存到目前機器人的磁碟檔案中
ImportPoints	用於將點檔案匯入目前選擇的機器人專案中
ExportPoints	用於將點檔案匯出到PC指定路徑中
P#	用於定義點資料
PDef	用於傳回是否定義指定點資料
PDel	用於刪除指定點資料
PLabel	用於設定指定點資料的標籤
PLabel$	用於傳回在指定點編號中定義的點標籤
PNumber	用於傳回與點標籤相應的點編號
PList	用於顯示目前機器人記憶體中的點資料
PLocal	用於設定指定點資料的本地屬性
PDescription	用於設定指定點資料的註解
PDescription$	用於傳回在指定點編號中定義的點之註解
WorkQue_Add	用於在指定工作佇列上新增工作佇列資料(點資料和使用者資料)
WorkQue_AutoRemove	用於在指定工作佇列上設定自動刪除功能
WorkQue_AutoRemove函數	用於傳回在工作佇列上設定的自動刪除功能的狀態
WorkQue_Get函數	用於從指定工作佇列傳回點資料
WorkQue_Len函數	用於傳回在指定工作佇列上註冊的啟用工作佇列資料數量
WorkQue_List	用於顯示指定工作佇列的工作佇列資料一覽(點資料和使用者資料)
WorkQue_Reject	用於在指定工作佇列上設定和顯示防止重複註冊點資料的最小距離
WorkQue_Reject函數	用於傳回指定工作佇列上設定的防止重複註冊的距離
WorkQue_Remove	用於從指定工作佇列中刪除工作佇列資料(點資料和使用者資料)
WorkQue_Sort	用於在指定工作佇列上設定和顯示Sort方法
WorkQue_Sort函數	用於傳回指定工作佇列上設定的Sort方法
WorkQue_UserData	用於重新設定和顯示指定工作佇列上註冊的使用者資料(實數)
WorkQue_UserData函數	用於傳回指定工作佇列上註冊的使用者資料(實數)

坐標轉換相關命令

命令	說明
AreaCorrection函數	回傳以補償區域進行補償的點

命令	說明
AreaCorrectionClr	清除補償區域
AreaCorrectionDef	回傳補償區域的設定值
AreaCorrectionInv函數	復原完成補償的點
AreaCorrectionOffset函數	回傳從完成補償的點進行相對移動的點
AreaCorrectionSet	設定與顯示補償區域
Arm	是用於選擇手臂和顯示目前選擇的手臂編號並傳回設定值的函數
ArmSet	用於設定和顯示增設型手臂
ArmDef	用於傳回手臂設定
ArmClr	用於清除手臂設定
ArmCalib	用於啟用或停用目前選擇的機器人的手臂長度補償
ArmCalibClr	用於清除手臂長度補償設定
ArmCalibDef	用於傳回手臂長度補償的設定
ArmCalibSet	用於設定手臂長度補償的設定
Tool	是用於選擇工具、顯示選擇工具編號或傳回設定值的函數
TLSet	用於設定和顯示工具坐標系
TLDef	用於傳回工具坐標系設定
TLClr	用於清除工具坐標系設定
ECP	是用於選擇ECP、顯示選擇的ECP編號或傳回設定值的函數
ECPSet	用於定義和顯示外部控制點(ECP)
ECPDef	用於傳回ECP設定
ECPClr	用於清除ECP設定
Base	用於定義和顯示基本坐標系
Local	用於定義和顯示本地坐標系資料
LocalDef	用於傳回本地坐標系設定
LocalClr	用於清除本地坐標系(未定義)
Elbow	是用於設定點的臂肘姿態並傳回設定的函數
Hand	是用於設定點的手臂姿態並傳回設定的函數
Wrist	是用於設定點的腕部姿態並傳回設定的函數
J4Flag	是用於設定點的J4Flag值並傳回設定值的函數
J6Flag	是用於設定點的J6Flag值並傳回設定值的函數

命令	說明
J1Flag	是用於設定點的J1Flag值並傳回設定值的函數
J2Flag	是用於設定點的J2Flag值並傳回設定值的函數
J1Angle	是用於設定點的J1Angle值並傳回設定值的函數
J4Angle	是用於設定點的J4Angle值並傳回設定值的函數
VxCalib	用於建立Vision Guide以外視覺系統的校正資料
VxTrans	是用於從像素坐標轉換為機器人坐標，並傳回已轉換點資料的函數
VxCalInfo	是用於傳回Vision Guide以外視覺系統的校正結束狀態和校正資料的函數
VxCalDelete	用於刪除Vision Guide以外視覺系統的校正資料
VxCalSave	用於將Vision Guide以外視覺系統的校正資料存檔
VxCalLoad	用於從檔案載入Vision Guide以外視覺系統的校正資料

程式控制相關命令

命令	說明
Function	用於函式的定義
For...Next	用於重覆執行For...Next之間的一系列陳述式
GoSub	用於執行副程式
Return	用於從副程式返回程式
GoTo	用於將程式控制移至指定標籤
Call	用於叫用作為副程式的函式
If… Then...Else...EndIf	用於程式條件的分歧
Else	以If…Then…Else形式使用後，若不符合條件運算式，則在Else之後繼續執行陳述式，而得以省略Else
Select ... Send	用於依據比較值選擇要執行的陳述式
Do...Loop	用於在迴圈的第一行或最終行執行條件判斷，並在符合或不符合條件時在Do...Loop之間重複
Declare	用於叫用DLL(動態連結程式庫)定義的外部函式
Trap	用於定義插斷以及發生插斷時的處理
OnErr	用於定義錯誤處理常式的位置
Era	是用於傳回發生錯誤的關節編號之函數
Erf$	用於傳回發生錯誤的函數之名稱
Erl	是用於傳回發生錯誤的行編號之函數

命令	說明
Err	是用於傳回錯誤代碼的函數
Ert	是用於傳回發生錯誤的工作編號之函數
Errb	是用於傳回發生錯誤的機器人編號之函數
ErrMsg$	是用於傳回與錯誤代碼相應的錯誤訊息的函數
Signal	用於向正在執行WaitSig命令的工作傳送信號
SyncLock	使用互斥鎖，對多項工作進行同步
SyncUnlock	用於解除事先用SyncLock鎖定的信號編號鎖定狀態
WaitSig	用於等待由來自其它工作的Signal命令發出之同步信號
ErrorOn	用於傳回控制器的錯誤狀態
Error	用於發生使用者定義錯誤
EResume	用於結束錯誤處理常式後重新執行程式
PauseOn	用於傳回暫停狀態(Pause狀態)
Exit	用於強制終止迴圈或函式。

程式執行相關命令

命令	說明
Xqt	用於執行以函式名稱指定的工作
Pause	用於暫停所有可暫停的工作
Cont	用於重新執行所有暫停的工作
Halt	用於暫停執行中的工作
Quit	用於結束工作
Resume	用於繼續執行暫停的工作
MyTask	用於傳回目前程式的工作編號
TaskDone	是用於確認結束工作的函數
TaskState	是用於取得目前工作狀態的函數
TaskWait	用於等待指定工作結束
Restart	用於中斷所有執行中的工作，並重新執行已執行的最後程式群組
Recover	用於執行朝開啟安全門時的位置的復歸動作，並傳回狀態
RecoverPos	用於傳回開啟安全門時的位置
StartMain	用於從背景工作執行Main函式

模擬命令

命令	說明
#define	用於以指定字串取代識別碼的定義
#ifdef ... #endif	用於條件式編譯
#ifndef ... #endif	用於條件式編譯
#include	用於插入指定的包含檔案
#undef	#用於清除以#define陳述式定義的識別碼

檔案管理相關命令

命令	說明
ChDir	用於變更指定驅動器的目前目錄
ChDisk	用於設定檔案操作對象的磁碟
MkDir	用於建立子目錄
RmDir	用於刪除子目錄
RenDir	用於變更目錄名稱
FileDateTime$	用於傳回檔案的日期與時間
FileExists	用於檢查是否有檔案
FileLen	用於傳回檔案的大小
FolderExists	用於檢查是否有資料夾
FreeFile	用於傳回並預約目前未使用的檔案編號
Del	用於刪除檔案
Copy	用於複製檔案
Rename	用於變更檔名
AOpen	用於為附加而開啟檔案
BOpen	用於以二進位存取模式開啟檔案
ROpen	用於開啟要載入的檔案
Uopen	用於開啟要讀寫的檔案
WOpen	用於開啟要寫入的檔案
Input#	用於從檔案、通訊連接埠、資料庫和裝置輸入字串或數值資料，並儲存為變數
Print#	用於將資料輸出到指定的檔案、通訊連接埠、資料庫和裝置
Line Input#	用於從檔案、通訊連接埠、資料庫和裝置載入1行資料
Read	用於從檔案或通訊連接埠載入指定的字元數

命令	說明
ReadBin	用於從檔案或通訊連接埠載入二進位資料
Write	用於將字串寫入檔案或通訊連接埠，而行尾不附加結束字元
WriteBin	用於將二進位資料寫出到檔案或通訊連接埠中
Seek	用於設定目前檔案指標
Close	用於關閉檔案
Eof	用於傳回檔案的EOF(已開啟檔案的指標位於端點)
ChDrive	用於變更目前驅動器
CurDir$	用於以字串傳回目前目錄
CurDrive$	用於以字串傳回目前的驅動器名稱
CurDisk$	用於以字串傳回目前的磁碟名稱
Flush	用於將緩衝區寫入檔案

現場匯流排相關命令

命令	說明
FbusIO_GetBusStatus	用於傳回指定現場匯流排的狀態
FbusIO_GetDeviceStatus	用於傳回指定現場匯流排裝置的狀態
FbusIO_SendMsg	用於將訊息傳送到現場匯流排I/O裝置並傳回回覆

數值相關命令

命令	說明
Ctr	是用於傳回計數器計數值的函數
CTReset	用於定義和重設計數器
ElapsedTime	用於測量生產節拍時間的函數
ResetElapsedTime	用於重設和啟動生產節拍時間測量用計時器
Tmr	用於計時器的函數
TmReset	用於重設和啟動計時器
Sin	是用於傳回指定角度正弦的函數
Cos	是用於傳回指定角度餘弦的函數
Tan	是用於傳回指定角度正切的函數
Acos	是用於傳回指定數值的反餘弦之函數
Asin	是用於傳回指定數值的反正弦之函數
Atan	是用於傳回指定數值的反正切之函數

命令	說明
Atan2	是用於傳回連接坐標原點和指定點的直角角度之函數
Sqr	是用於傳回平方根的函數
Abs	是用於傳回絕對值的函數
Sgn	是用於傳回數值符號的函數
Int	是用於傳回無條件捨去小數部分後的數值之函數
BClr	用於清除指定數值的1位元，並傳回該值
BSet	用於設定指定數值的1位元，並傳回該結果
BTst	用於傳回數值的1位元狀態
BClr64	用於清除指定數值的1位元，並傳回該值
BSet64	用於設定指定數值的1位元，並傳回該結果
BTst64	用於傳回數值的1位元狀態
Fix	用於從實數值取出整數部分
Hex	用於將以十六進位表示的數值進行字元化後傳回
Randomize	用於亂數系列的初始化
Redim	用於在執行階段變更陣列的最大元素編號
Rnd	用於傳回亂數
UBound	用於傳回可在指定陣列中設定的元素編號之最大值

字串相關命令

命令	說明
Asc	用於以十進位傳回字串開頭字元的字元碼
Chr$	用於傳回字元碼相關字元
Left$	用於從字串左側取出指定數量的字串
Mid$	用於從指定字串，自開始位置取出指定數量的字元
Right$	用於從字串的末尾取出指定數量的字串
Len	用於傳回字串長度(字元數)
LSet$	用於在指定字串的末尾加入空格，並傳回指定長度的字串
RSet$	用於在字串的開頭加入空格，並傳回指定長度的字串
Space$	用於傳回指定數量的空格字串
Str$	用於將數值轉換為字串

命令	說明
Val	用於將由數字構成的字串轉換為數值
LCase$	用於以小寫形式傳回指定字串
UCase$	用於以大寫形式傳回指定字串
LTrim$	用於刪除字串開頭的空格並傳回
RTrim$	用於刪除字串末尾的空格並傳回
Trim$	用於刪除字串開頭和末尾的空格並傳回
ParseStr	用於在Token陣列上對字串進行語法分析
FmtStr	用於對數字或日期、時間的標記進行格式化
FmtStr$	用於對數字或日期、時間的標記進行格式化
InStr	用於從字串中搜尋字串，並傳回發現的位置
Tab$	用於傳回指定數量的定位字串

邏輯運算相關命令

命令	說明
And	用於邏輯AND運算
Or	用於邏輯OR運算
LShift	用於數值資料的邏輯左移
LShift64	用於數值資料的邏輯左移
Mod	用於整數的模數運算
Not	用於NOT運算
RShift	用於數值資料的邏輯右移
RShift64	用於數值資料的邏輯右移
Xor	用於互斥運算
Mask	用於以Wait陳述式執行位元AND運算

變數相關命令

命令	說明
Boolean	用於宣告Boolean變數
Byte	用於宣告Byte變數
Double	用於宣告Double變數
Global	用於宣告全域變數
Int32	用於宣告4位元組整數變數

命令	說明
Integer	用於宣告2位元組整數變數
Long	用於宣告Long變數
Int64	用於宣告8位元組整數變數
Real	用於宣告實數變數
Short	用於宣告2位元組整數變數
String	用於宣告字串變數
UByte	用於宣告無符號整數變數
UInt32	用於宣告無符號4位元組整數變數
UShort	用於宣告無符號2位元組整數變數
UInt64	用於宣告無符號8位元組整數變數

安全相關命令

命令	說明
Login	執行時，以其他使用者身份登入Epson RC+

輸送帶追蹤相關命令

命令	說明
Cnv_AbortTrack	用於中斷對輸送帶佇列點的動作命令
Cnv_Accel函數	用於傳回輸送帶追蹤前的加速度、減速度設定值
Cnv_Accel	用於設定輸送帶追蹤前的加速度、減速度設定值
Cnv_AccelLim	用於設定輸送帶追蹤後的加速度、減速度設定值
Cnv_AccelLim函數	用於傳回輸送帶追蹤後的加速度、減速度設定值
Cnv_Adjust	用於設定是否執行取得輸送帶的追蹤延遲補償值之動作
Cnv_AdjustClear	用於刪去輸送帶的追蹤的追蹤延遲補償值
Cnv_AdjustGet函數	用於傳回輸送帶的追蹤的追蹤延遲補償值
Cnv_AdjustSet	用於設定輸送帶的追蹤之追蹤延遲補償值
Cnv_Downstream函數	用於傳回輸送帶的下游極限設定值
Cnv_Downstream	用於設定輸送帶的下游極限設定值
Cnv_Fine函數	用於傳回指定輸送帶完成追蹤判斷範圍的設定
Cnv_Fine	用於設定和顯示對於指定輸送帶的完成追蹤判斷範圍
Cnv_Flag函數	用於傳回追蹤停止線的追蹤狀態
Cnv_Mode函數	用於傳回輸送帶的設定模式設定值
Cnv_Mode	用於設定輸送帶的設定模式設定值

命令	說明
Cnv_Name$函數	用於傳回指定輸送帶的名稱
Cnv_Number函數	用於傳回指定輸送帶名稱的輸送帶編號
Cnv_OffsetAngle	用於設定輸送帶佇列資料的位移值
Cnv_OffsetAngle函數	用於傳回輸送帶佇列資料的位移值
Cnv_Point函數	用於將感測器坐標值轉換為輸送帶坐標值並傳回
Cnv_PosErr函數	用於傳回目前追蹤位置與目標之間的位置偏差
Cnv_PosErrOffset	設置一個值，以補償當前追蹤位置與目標之間的位置偏差
Cnv_Pulse函數	用於傳回輸送帶的目前位置脈衝
Cnv_QueAdd	用於在輸送帶佇列資料中新增點資料
Cnv_QueGet函數	用於從指定的輸送帶佇列資料傳回點資料
Cnv_QueLen函數	用於傳回指定輸送帶佇列的資料數量
Cnv_QueList	用於顯示指定輸送帶佇列資料的一覽
Cnv_QueMove	用於將上游輸送帶的佇列資料移至下游輸送帶佇列
Cnv_QueReject	設定和顯示用於防止重複註冊輸送帶的最小距離
Cnv_QueReject函數	用於傳回防止重複註冊輸送帶佇列的距離
Cnv_QueRemove	用於從輸送帶佇列資料中刪除佇列資料
Cnv_QueUserData	用於顯示和設定佇列項目相關使用者資料
Cnv_QueUserData函數	用於傳回佇列項目相關使用者資料
Cnv_RobotConveyor函數	用於傳回追蹤中的輸送帶編號
Cnv_Speed函數	用於傳回輸送帶的動作速度
Cnv_Trigger	用於為下一個Cnv_QueAdd陳述式而對輸送帶的目前位置進行閂鎖
Cnv_Upstream函數	用於傳回輸送帶的上游極限設定值
Cnv_Upstream	用於設定輸送帶的上游極限設定值

DB相關命令

命令	說明
CloseDB	用於關閉以OpenDB命令開啟的資料庫，並釋放資料庫編號
DeleteDB	用於從開啟的資料庫內的表格中刪除資料
OpenDB	用於開啟資料庫、Excel活頁簿
SelectDB	是用於從開啟的資料庫內的表格中搜尋資料的函數
UpdateDB	用於更新開啟的資料庫內的表格資料

PG相關命令

命令	說明
PG_FastStop	用於緊急停止連續旋轉的脈衝輸出軸
PG_LSpeed	用於設定脈衝輸出軸開始加速時的脈衝速度以及結束減速時的脈衝速度
PG_Scan	用於開始脈衝輸出軸的連續旋轉動作
PG_SlowStop	用於使連續旋轉的脈衝輸出軸減速並停止

碰撞檢測相關命令

命令	說明
CollisionDetect	用於設定碰撞檢測功能的啟用/停用
CollisionDetect函數	是用於傳回CollisionDetect命令設定值的函數

零件消耗管理相關命令

命令	說明
HealthCalcPeriod	用於設定零件消耗管理的運算期間
HealthCalcPeriod函數	是用於傳回零件消耗管理運算期間的函數
HealthCtrlAlarmOn函數	是用於傳回控制器零件的消耗警報狀態的函數
HealthCtrlInfo	用於顯示控制器零件的建議更換時期前的剩餘月數
HealthCtrlInfo函數	是用於傳回控制器零件的建議更換時期前的剩餘月數的函數
HealthCtrlRateOffset	用於設定控制器零件消耗率的位移值
HealthCtrlReset	用於對控制器零件的消耗率進行初始化
HealthCtrlWarningEnable	用於設定控制器零件消耗警報通知的啟用/停用
HealthCtrlWarningEnable函數	是用於傳回控制器零件消耗警報通知的啟用/停用狀態之函數
HealthRateCtrlInfo函數	是用於傳回控制器零件消耗率的函數
HealthRateRBInfo函數	是用於傳回機器人零件消耗率的函數
HealthRBAlarmOn函數	是用於傳回機器人零件的消耗警報狀態之函數
HealthRBAnalysis	用於顯示機器人零件消耗相關分析結果(建議更換時期前的剩餘月數)
HealthRBAnalysis函數	是用於傳回機器人零件消耗相關分析結果(建議更換時期前的剩餘月數)的函數
HealthRBDistance	用於顯示指定關節的驅動量
HealthRBDistance函數	是用於傳回指定關節驅動量的函數
HealthRBInfo	用於顯示機器人零件的建議更換時期前的剩餘月數
HealthRBInfo函數	是用於傳回機器人零件的建議更換時期前的剩餘月數之函數
HealthRBRateOffset	用於設定機器人零件消耗率的位移值
HealthRBReset	用於對機器人零件消耗率進行初始化
HealthRBSpeed	用於顯示指定關節的平均速度

命令	說明
HealthRBSpeed函數	是用於傳回指定關節平均速度的函數
HealthRBStart	用於開始機器人零件消耗相關分析
HealthRBStop	用於結束機器人零件消耗相關分析
HealthRBTRQ	用於顯示指定關節的扭矩值
HealthRBTRQ函數	是用於傳回指定關節扭矩值的函數
HealthRBWarningEnable	用於設定機器人零件消耗警報通知的啟用/停用
HealthRBWarningEnable函數	是用於傳回機器人零件消耗警報通知的啟用/停用狀態的函數

模擬器相關命令

命令	說明
SimSet	用於設定模擬器的物件並對操作及機器人動作進行設定
SimGet	用於取得模擬器物件的設定值

末端夾具相關命令

有關詳細資訊，請參閱以下手冊。
「Hand功能」

命令	說明
Hand_On	夾持機構 : 執行握持動作 電動螺絲起子：執行末端夾具的螺絲鎖付動作
Hand_On函數	夾持機構 : 當末端夾具處在握持狀態，則傳回“True” 電動螺絲起子：當末端夾具處在完成螺絲鎖付狀態，則傳回“True”
Hand_Off	夾持機構 : 執行放開動作 電動螺絲起子：執行末端夾具的鬆開螺絲動作
Hand_Off函數	夾持機構 : 當末端夾具處在放開狀態，則傳回“True” 電動螺絲起子：當末端夾具處在完成鬆開螺絲狀態，則傳回“True”
Hand_TW函數	當上一個Hand_On命令或Hand_Off命令逾時，則傳回“True”
Hand_Def函數	當抓手已被設置，則傳回“True”
Hand_Type函數	傳回末端夾具的類型編號
Hand_Label$函數	傳回末端夾具的標籤
Hand_Number函數	傳回末端夾具的編號

關於安全功能的命令

有關詳細資訊，請參閱以下手冊。
「機器人控制器 安全功能手冊」

命令	說明
SF_GetParam函數	回傳安全功能參數資訊

命令	說明
SF_GetParam$函數	回傳安全功能參數的文字資訊
SF_GetStatus函數	回傳安全功能的狀態位元
SF_LimitSpeedS	啟用SLS時，針對調整透過Tool指令所設定位置的速度的功能，設定、顯示速度調整值。
SF_LimitSpeedS函數	啟用SLS時，會送回對透過Tool指令設定位置的速度進行調整的功能的速度調整值。
SF_LimitSpeedSEnable	啟用SLS時，針對調整透過Tool指令設定位置的速度的功能，設定On/Off，並顯示設定狀態。
SF_LimitSpeedSEnable函數	啟用SLS時，會送回調整透過Tool指令設定位置的速度的功能狀態。
SF_PeakSpeedS	顯示速度監視點的峰值速度
SF_PeakSpeedS函數	回傳速度監視點的峰值速度
SF_PeakSpeedSClear	清除速度監視點的峰值速度，並進行初始化
SF_RealSpeedS	顯示速度監視點的目前速度
SF_RealSpeedS函數	回傳速度監視點的目前速度

VRT相關命令

有關詳細資訊，請參閱以下手冊。
「Vibration Reduction Technology」

命令	說明
VRT	用於選擇VRT或顯示選擇的VRT編號
VRT函數	傳回目前設定的VRT編號的函數
VRT_Clr	用於清除VRT功能
VRT_CPMotion	用於指定CP動作期間啟用或停用VRT功能
VRT_CPMotion函數	傳回CP動作期間啟用或停用VRT功能的函數
VRT_Def函數	傳回所選VRT編號的設定狀態的函數
VRT_Description	設定所選VRT編號的註解
VRT_Description$函數	傳回所選VRT編號的註解的函數
VRT_Label	用於設定所選VRT編號的標籤
VRT_Label$函數	傳回所選VRT編號的標籤的函數
VRT_Number函數	傳回與VRT標籤相應的VRT編號
VRT_Set	用於設定各VRT編號的VRT功能之VRTParam1、VRTParam2
VRT_Set函數	傳回各VRT編號設定的VRTParam1、VRTParam2設定值的函數
VRT_Trigger	用於輸出測量觸發至VR軟體

3. SPEL+ 語言參考

3.1 SPEL+ 語言參考

本章節將根據以下項目來說明SPEL+命令。

格式

表示使用該命令時的格式。部分命令同時顯示多種格式與參照編號，參照編號表示與「說明」欄的關聯性。
格式中使用的符號說明如下所示。

符號：[ ]
可省略的參數。
以下參數是使用此括弧的參數。

- 以空格區隔的參數
  以空格區隔時，可改變參數的順序或省略。

- 以逗號區隔的參數
  以逗號區隔時，順序固定無法改變。
  此外，逗號後面的所有參數都可以省略，但最後以逗號區隔的參數前面的參數無法各別省略。
  例：
  Inertia [負載慣性 [, 偏心量]]
  此範例如下所示。
  OK: Inertia [負載慣性]
  OK: Inertia
  NG: Inertia [, 偏心量]

符號：{ }
可選擇的參數。
可選擇1個此括弧中以 | 區隔的參數。

例：
Exit { Do | For | Function }
上述情況，請由Do, For , Function之中選擇1個。

參數

是關於該命令參數的說明。

結果

是關於該命令傳回值的說明。

傳回值

是關於該函數傳回值的說明。

說明

對命令的功能進行詳細說明。

注意

針對該命令，說明重要事項。

參照

是與該命令相關的其他命令清單。若要參照相關命令，可從目錄尋找相關頁面。

範例

記載使用該命令的範例。

3.2 運算子

SPEL+ 語言所使用的運算子如下所示。

運算子	格式範例	說明
+	A+B	加法
-	A-B	減法
*	A*B	乘法
/	A/B	除法
**	A**B	指數
=	A=B	A等於B
>	A>B	A大於B
<	A<B	A小於B
>=	A>=B	A大於或等於B
<=	A<=B	A小於或等於B
<>	A<>B	A不等於B
And	A And B	邏輯AND
Mod	A Mod B	整數的餘數
Not	Not A	否定
Or	A Or B	邏輯OR
Xor	A Xor B	互斥

運算子的優先順序

在程式中，按下列順序處理運算子。

優先順序	運算子	格式範例	說明
1	()	(A+B)	括號
2	**	A**B	指數
3	*	A*B	乘法
	/	A/B	除法
4	Mod	A Mod B	整數的餘數
5	+	A+B	加法
	-	A-B	減法
6	=	A=B	A等於B
	<>	A<>B	A不等於B
	<	A<B	A小於B
	>	A>B	A大於B
	<=	A<=B	A小於或等於B

優先順序	運算子	格式範例	說明
	>=	A>=B	A大於或等於B
7	Not	Not A	否定
8	And	A And B	邏輯AND
9	Or	A Or B	邏輯OR
10	Xor	A Xor B	互斥

3.3 !

3.3.1 !...! 平行處理

在動作中平行執行I/O等輸入輸出處理。

格式

動作命令 !平行處理陳述式!

參數

動作命令

描述以下任一個可進行平行處理的命令。

Arc、Arc3、Go、Jump、Jump3、Jump3CP、Move、BGo、BMove、TGo、TMove

平行處理陳述式

描述在動作中可進行平行處理的I/O陳述式。(參照下表)

說明

在開始動作的同時，開始執行被夾在「!」標記中的陳述式。例如，I/O無須在機器人停止運作前等待執行，而可在手臂動作時執行。備有手臂動作時對I/O催促執行的陳述式。(參照下表「Dn」)

下表匯總所有可使用的平行處理陳述式。表中的陳述式皆可單獨使用。此外，還可組合多項，在1個動作命令中執行多項I/O陳述式。

Dn	用於執行下一個平行處理陳述式前指定移動量的%值並與動作命令取得同步。「n」是以0~100的整數指定的%值，用於指定開始位置(從該動作的哪個位置開始執行平行處理陳述式)。也就是說，當動作達到所有移動量的n%時，執行Dn以後的陳述式。 與Jump命令組合使用時，移動量的%值中不包含第3關節(上下軸)的垂直移動動作。進行Jump動作時，只對非第3關節(上下軸)的平行移動動作取得同步。完成第3關節垂直移動動作之後，請在陳述式的開頭加入D0(零)，以執行陳述式。 與Jump3命令組合使用時，移動量的%值中不包含閃避/接近動作。執行Jump3動作時，只對Span動作取得同步。請在陳述式的開頭輸入D0(零)，以便在完成閃避動作時執行陳述式。 在1次平行處理陳述式中，最多可使用16次「Dn」。
On / Off n	用於開啟或關閉輸出位元編號「n」。
MemOn / MemOff n	用於開啟或關閉記憶體I/O位元編號「n」。
Out p,d OpBCD p,q OutW p,d	用於向輸出連接埠「p」輸出資料「d」。

MemOut p, d MemOutW p,d	用於向記憶體I/O連接埠「p」輸出資料「d」。
Signal s	用於發出同步信號。
WaitSig s	用於在等待信號「s」後處理下一個陳述式。
Wait t	用於等待「t」秒鐘後，執行下一個平行處理陳述式。
Wait Sw(n) = j	用於等待執行下一個平行處理陳述式，直到輸入位元「n」與以「j」定義的條件(On或Off)一致。
Wait MemSw(n) = j	用於等待執行下一個平行處理陳述式，直到記憶體I/O位元「n」與以「j」定義的條件(On或Off)一致。
Wait 其它條件	也可執行上述2種以外的Wait。詳細內容請參閱Wait。
Print	用於向顯示裝置輸出資料。
Print#	用於向指定的通訊連接埠輸出資料。
外部函式	執行Declare陳述式所宣告的外部函式。
Hand_On n Hand_Off n	執行末端夾具編號“n”的Hand_On/Hand_Off動作。

注意

- 在結束所有I/O命令之前動作就結束時

  即便完成針對特定命令的動作，但未結束所有平行處理的陳述式時，則在全部完成後執行下一個程式。尤其在必須對多個I/O命令進行平行處理的短距離移動動作時就預測到這種情況。

- 依據停止手臂的Till陳述式，在中途結束動作時

  倘若在移動中途使用停止手臂的Till陳述式，即視為已100%完成動作，並執行到D100。在完成執行所有平行處理陳述式之前，不會移到動作命令的下一個陳述式。

- 依據AbortMotion陳述式或Trap，在中途結束動作時

  不執行結束動作以後的D陳述式。

- 設定為接近100% 的「n」值時的路徑運動的減速

  已在路徑運動中設定較高的「n」值時，機器人有可能減速，以便完成執行中的動作。CP On通常用於在開始減速的同時開始下一個動作命令的加速。然而，減速中若有指定Dn，則在尚未結束該命令前，無法開始下一個加速。為了避免減速，將處理陳述式留在動作命令之後。以下列為例，將On 1陳述式位置從Jump P1動作中的平行處理移到Jump P1之後。

```
CP On
Jump P1 !D96; On 1!
Go P2
```

```
CP On
Jump P1
On 1
Go P2
```

- Jump陳述式和平行處理

  在結束上升移動後，開始執行連同Jump陳述式一起使用的平行處理陳述式，在開始下降移動之前結束。

  在結束閃避移動後，開始執行連同Jump3陳述式一起使用的平行處理陳述式，在開始接近移動之前結束。

- Here陳述式和平行處理

  1個動作命令中不可同時存在Here陳述式和平行處理。

```
Go Here :Z(0) ! D10; MemOn 1 !
```

  不接受上述用法。

```
P999 = Here
Go P999 Here :Z(0) ! D10; MemOn 1 !
```

  請變更為上述程式。

參照

Arc、Arc3、Go、Jump、Jump3、Jump3CP、Move、BGo、BMove、TGo、TMove

!...! 平行處理範例

如下所述是連同動作命令一起使用平行處理功能的範例。

連同Jump命令一起使用平行處理。在第3關節結束上升移動，第1、2、4關節開始動作的階段，開啟輸出位元1。在完成50%Jump動作的階段，再度關閉輸出位元1。

```
Function test
   Jump P1 !D0; On 1; D50; Off 1!
Fend
```

連同Move命令一起使用平行處理。在已完成10%移至P1的階段，開啟輸出位元5，0.5秒過後關閉輸出位元5。

```
Function test2
   Move P1 !D10; On 5; Wait 0.5; Off 5!
Fend
```

3.4

3.4.1 #define

用於以指定字串取代識別碼的定義。

格式

#define 識別碼 [(參數 [, 參數])] 取代字串

參數

識別碼

是用於省略字串參數的關鍵字。如下所述為識別碼的規則。

- 以字母開頭並用英文數字和下劃線(_)建立。
- 識別碼不可使用空格和定位字元。

參數

指定用於取代字串的1個或多個變數，然後提供如同巨集指令的動態定義機制。#在#define命令中，最多可使用8個參數。請用逗號分隔每個參數，然後將參數清單放在括弧內。

取代字串

是程式被編譯時用於取代識別碼的字串。如下所述為取代字串相關規則。

- 在取代字串中可使用空格和定位字元。
- 連同陳述式一起使用的識別碼不可用作取代字串。
- 一旦使用註解記號「'」，其後續字元則被判斷為註解，而不包含在取代字串中。
- 可省略取代字串。此時，已指定的識別碼會以空白或NULL字串取代。藉此，該識別碼會在程式中被刪除。

說明

使用#define命令在程式中取代已指定的識別碼。每發現一次指定的識別碼，都以取代字串取代並進行編譯。然而，原始碼本身並不是取代字串，而是以包含識別碼的型態被保留下來的。這是因為在讀取代碼本身時，讀取識別碼比起讀取以字串取代的代碼更為容易的緣故。

已定義的識別碼可搭配#ifdef或#ifndef命令，作為是否編譯的條件使用。

若指定參數，識別碼便可用作巨集。

注意

- #若將#define用於變數宣告或標籤取代，則會發生錯誤。

  請注意，若為了變數宣告而使用#define命令，則會發生錯誤。

參照

#ifdef、#ifndef

#define範例

```
'偵錯模式時，會對下一行進行非註解處理
' #define DEBUG

Input #1, A$
 #ifdef DEBUG
    Print "A$ = ", A$
 #endif
Print "The End"
```

```
 #define SHOWVAL(x) Print "var = ", x

Integer a

a = 25

SHOWVAL(a)
```

3.4.2 #ifdef...#else...#endif

進行條件式編譯。

格式

#ifdef識別碼 ...用於條件式編譯的原始碼
[#else]...用於偽條件的原始碼
#endif

參數

識別碼
　　#是編譯#ifdef和#else或#endif之間的原始碼的使用者定義關鍵字。此識別碼作為編譯條件發揮作用。

說明

#ifdef...#else...#endif用於對選擇的原始碼進行條件式編譯。是否編譯的條件取決於識別碼。首先，#ifdef用於檢查是否依目前#define定義指定的識別碼。#可省略#else陳述式。

[已定義時] 若使用#else陳述式，則編譯#ifdef和#else之間的陳述式。#若未使用#else陳述式，即編譯#ifdef和#endif之間的陳述式。

[未定義時] 若使用#else陳述式，則編譯#else和#endif之間的陳述式。#若未使用#else陳述式，則不編譯並直接跳過#ifdef和#endif之間的陳述式。

參照

#define、#ifndef

#ifdef範例

以下為使用#ifdef的程式範例。在以下範例中，可依照有無#define DEBUG模擬命令定義，來控制是否輸出變數A$的值。倘若在此範例記載的部分之前在原始碼中使用#define DEBUG模擬命令，則編譯Print A$的行，並在執行程式時進行輸出。然而，無論#define DEBUG模擬命令如何，皆輸出字串「The End」。

```
'偵錯模式時，會對下一行進行非註解處理
' #define DEBUG

Input #1, A$
 #ifdef DEBUG
    Print "A$ = ", A$
 #endif
Print "The End"
```

3.4.3 #ifndef...#endif

進行條件式編譯。

格式

#ifndef識別碼 .. 用於條件式編譯的原始碼
[#else]... 用於真條件的原始碼
#endif

參數

識別碼

是使用者定義的關鍵字。未定義時，則編譯#ifndef到#else或#endif為止的原始碼。此識別碼作為執行編譯的條件發揮作用。

說明

此命令也稱之為「若未定義」命令，負責檢查與#ifdef相反的條件。#ifndef...#else...#endif用於對選擇的原始碼進行條件式編譯。#可省略#else陳述式。

[已定義時] 若使用#else陳述式，則編譯#else和#endif之間的陳述式。#若未使用#else陳述式，則不編譯#ifndef和#endif之間的陳述式。

[未定義時] 若使用#else陳述式，則編譯#else和#endif之間的陳述式。#若未使用#else陳述式，即編譯#ifndef和#endif之間的陳述式。

注意

- #ifdef和#ifndef的差異

  #已定義識別碼時，#ifdef用於編譯指定的原始碼。另一方面，未定義識別碼時，#ifndef用於編譯指定的原始碼。

參照

#define、#ifdef

#ifndef範例

以下為使用#ifndef的程式範例。在以下範例中，可依照有無#define NODELAY模擬命令定義，來控制是否輸出變數A$的值。倘若在此例記載的部分之前在原始碼中使用#define NODELAY模擬命令，則在編譯之時，Wait 1的行不會與此程式原始碼的其它部分一起編譯。倘若在此範例記載的部分之前不使用#define NODELAY模擬命令(換句話說，若未定義NODELAY)，則編譯Wait 1的行，並在隨後被程式執行。無論#define NODELAY模擬命令如何，皆輸出字串「The End」。

```
'延遲時，會對下一行進行註解處理
 #define NODELAY 1

Input #1, A$
 #ifndef NODELAY
    Wait 1
 #endif
Print "The End"
```

3.4.4 #include

在使用#include陳述式的位置，插入指定的包含檔案。

格式

#include "包含檔案名.INC "

參數

包含檔案名

請指定目前專案中啟用的包含檔案名。副檔名皆為「.inc」。包含檔案名則指定插入目前檔案的檔名。

說明

#include用於將指定包含檔案的內容插入到使用#include陳述式的位置。

請將#define陳述式和全域變數的宣告加入包含檔案中。

請在函式定義之外部分使用#include陳述式。

可在包含檔案中記述其它包含檔案。例如，在FILE1中可包含FILE2；在FILE2中可包含FILE3。將此稱為檔案巢狀。

參照

#define、#ifdef、#ifndef

#include範例

```
Include File (Defs.inc)

 #define DEBUG 1
 #define MAX_PART_COUNT 20

Program File (main.prg)

 #include "defs.inc"

Function main
    Integer i

    Integer Parts(MAX_PART_COUNT)

Fend
```

3.4.5 #undef

將利用#define陳述式定義的識別碼設為未定義。

格式

#undef 識別碼名

參數

識別碼名

#用於指定define陳述式所使用的關鍵字。

參照

#define、#ifdef、#ifndef

3.5 A

3.5.1 AbortMotion

用於中斷動作命令，並將執行動作的工作設為錯誤。

本命令為適合高階專業人員使用的命令。請充分理解命令內容後使用。

格式
AbortMotion

參數

機器人編號
　　用於指定中斷動作的機器人編號。
All
　　用於中斷所有機器人動作。

說明
依照執行AbortMotion時的機器人狀態，會出現如下情形。

無論在任何情況之下，若要繼續處理，都必須以OnErr攔截錯誤，記述錯誤處理。

錯誤2999可使用常數ERROR_DOINGMOTION。

錯誤2998可使用常數ERROR_NOMOTION。

請在執行繼續執行(Cont)之前，建立避免執行2次以上AbortMotion的程式。

- 機器人正在執行動作命令時

  機器人立即暫停(快速暫停)手臂動作，放棄剩餘的動作。

  在對機器人執行動作命令的工作中發生錯誤2999(ERROR_DOINGMOTION)。

  下一個動作命令用於從暫停位置直接向下一個目標位置進行動作。

- 機器人處於暫停(快速暫停)狀態時

  雖然在執行AbortMotion時未發生任何情況，但放棄了內部的剩餘動作。

  發出繼續執行(Cont)命令時，在對機器人執行動作命令的工作中發生錯誤2999(ERROR_DOINGMOTION)。

  下一個動作命令用於從暫停位置直接向下一個目標位置進行動作。

- 機器人處於WaitRecover狀態(打開安全門狀態)時

  雖然在執行AbortMotion時未發生任何情況，但放棄了內部的剩餘動作。

  可以Recover命令的旗標(Flag)選擇接下來的動作。

  - 執行「Recover 機器人編號, WithMove」，即恢復機器人的勵磁，並執行復歸動作。發出繼續執行(Cont)命令時，在對機器人執行動作命令的工作中發生錯誤2999(ERROR_DOINGMOTION)。下一個動作命令用於從完成復歸動作的位置直接向下一個目標位置進行動作。
  - 執行「Recover 機器人編號, WithoutMove」，即恢復機器人的勵磁。發出繼續執行(Cont)命令時，在對機器人執行動作命令的工作中發生錯誤2999(ERROR_DOINGMOTION)。下一個動作命令用於從恢復勵磁的位置直接向下一個目標位置進行動作。(無復歸動作。)

- 機器人正在執行非動作命令時

  在此之前，在對機器人執行動作命令的工作中發生錯誤2998(ERROR_NOMOTION)。因Wait或Input命令而處於等待狀態時，立即中斷工作並發生錯誤2998。

  利用CP On設定執行動作命令時，動作命令中若缺少程式，則即便機器人處於動作狀態，仍會發生錯誤2998。

- 指定的機器人未以程式(工作)進行動作時

  會發生錯誤。

注意

- 支援的控制器型號

  不支援T/VT系列。

參照

OnErr、Recover、Till

AbortMotion範例

若記憶體I/O的0號為On，則執行AbortMotion，機器人向Home位置移動。

```
Function main
  Motor On
  Xqt sub, NoEmgAbort
  OnErr GoTo errhandle

  Go P0
  Wait Sw(1)
  Go P1

  Quit sub
  Exit Function

errstart:
  Home
  Quit sub
  Exit Function

errhandle:
  Print Err
  If Err = ERROR_DOINGMOTION Then
    Print "機器人處於動作狀態"    'Go P0或Go P1執行中
    EResume errstart
  ElseIf Err = ERROR_NOMOTION Then
    Print "機器人處於非動作狀態"    'Wait Sw (1) 執行中
    EResume errstart
  EndIf

  Print "Error Stop"    '發生其它錯誤
  Quit All
Fend

Function sub
  MemOff 0
  Wait MemSw(0)
  AbortMotion 1
  MemOff 0
Fend
```

3.5.2 Abs函數

是用於傳回絕對值的函數。

格式
Abs (數值)

參數

數值
以運算式或數值指定。

傳回值
用於傳回指定數值的絕對值。

說明
絕對值是指無符號的數值。例如，對於Abs (-1) 和Abs (1) 兩者，皆傳回1。

參照
Atan、Atan2、Cos、Int、Mod、Not、Sgn、Sin、Sqr、Str$、Tan、Val

Abs函數範例
下列範例表示的是，在命令視窗中使用Print命令執行的情形。

```
> print abs(1)
1
> print abs(-1)
1
> print abs(-3.54)
3.54
>
```

3.5.3 Accel

設定和顯示Go、Jump、Pulse等PTP動作的加減速度。

格式

Accel 加速設定值, 減速設定值 [, 閃避加速設定值, 閃避減速設定值, 接近加速設定值, 接近減速設定值]

參數

加速設定值
　　用1以上的整數指定對最大加速度的比例。(單位：%)
減速設定值
　　用1以上的整數指定對最大減速度的比例。(單位：%)
閃避加速設定值
　　用1以上的整數指定Jump時的閃避加速度。可省略。唯有Jump命令時方可設定。
閃避減速設定值
　　用1以上的整數指定Jump時的閃避減速度。可省略。唯有Jump命令時方可設定。
接近加速設定值
　　用1以上的整數指定Jump時的接近加速度。可省略。唯有Jump命令時方可設定。
接近減速設定值
　　用1以上的整數指定Jump時的接近減速度。可省略。唯有Jump命令時方可設定。

結果

省略參數時，傳回目前Accel參數。

說明

Accel用於設定各種PTP動作(以Go、Jump、Pulse等命令產生的動作)的加減速度。

用1以上的整數值設定Accel所設定的加減速度參數。此數值表示對最大加速度(或減速度)的比例。通常100為最大值，但也有可設為超過100的機器人。AccelMax函數用於傳回可進行Accel設定的最大值。

Accel用於新設定加減速度時或僅輸出目前設定值等情況。在新設定加速度或減速度並使用Accel時，需要最初的2個參數(加速設定值和減速設定值)。

僅在Jump命令時，方可設定並省略閃避加速設定值、閃避減速設定值、接近加速設定值、接近減速設定值這4種參數。這些參數用於指定Jump動作開始時的閃避動作和Jump動作結束時的接近動作之各加速設定值和減速設定值。

在以下任一情況下，Accel值會被初始化。

- 控制器電源 開啟
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

注意

- 在低功率模式(Power Low)下執行Accel命令時

  在低功率模式(Power Low)下執行Accel時，雖然會儲存新的值，但目前值會受限制。在TEACH模式為OFF、功率為High的情況下，Accel會處於啟用狀態。

- Accel和AccelS的差異

  Accel命令並非用於設定直線或圓弧動作的加減速度。另一方面，AccelS命令則用於設定直線或圓弧動作的加減速度。

- 超過100的Accel設定

  Accel設定通常100為最大值，但也有可設為100以上數值的機種。在正常使用情況下，Accel設定值100是指在加減速度和定位時的振動之間取得平衡的最佳設定。然而，可能會有依動作條件，透過降低定位時的振動以提高加減速度，將提升週期時間作為優先的情況。在這種情況下，請在Accel設定值上設定100以上的數值。但是，也可能會有依動作條件，即便設定100以上的數值，也未能在週期時間上看到變化。

參照

AccelR、AccelS、Go、Jump、Jump3、Power、Pulse、Speed、TGo

Accel範例

以下為使用Accel和Speed的簡易動作程式範例。在Accel和Speed中使用事先定義的變數。

[例1]

```
Function acctest
  Integer slow, accslow, decslow, fast, accfast, decfast

  slow = 20      '低速度的設定
  fast = 100     '高速度的設定
  accslow = 20   '低加速度的設定
  decslow = 20   '低減速度的設定
  accfast = 100 '高加速度的設定
  decfast = 100 '高減速度的設定

  Accel accslow, decslow
  Speed slow
  Jump pick
  On gripper
  Accel accfast, decfast
  Speed fast
  Jump place
  .
  .
  .
Fend
```

[例2]

這是在Jump命令時減慢第3關節下降減速度以便小心處理零件的範例。在此範例中，設定Accel時，需要將第3關節下降減速設定值的參數設定為低值。

```
>Accel 100,100,100,100,100,35

>Accel
    100     100
    100     100
    100     35
>
```

3.5.4 Accel函數

是用於傳回目前加減速度設定值的函數。

格式

Accel (設定值編號)

參數

設定值編號

以整數值指定以下各值。

- 1：加速設定值
- 2：減速設定值
- 3：Jump動作時的閃避加速設定值
- 4：Jump動作時的閃避減速設定值
- 5：Jump動作時的接近加速設定值
- 6：Jump動作時的接近減速設定值

傳回值

用於傳回1以上的整數(%)。

參照

Accel

Accel函數範例

是在程式中使用Accel函數的範例。

```
Integer currAccel, currDecel

'取得目前加減速度
currAccel = Accel(1)
currDecel = Accel(2)
Accel 50, 50
SRVJump pick
'還原之前的設定
Accel currAccel, currDecel
```

3.5.5 AccelMax函數

是用於傳回可以Accel設定的加減速度最大值的函數。

格式

AccelMax (最大值編號)

參數

最大值編號

以整數值指定以下各值。

- 1：加速度最大值
- 2：減速度最大值
- 3：Jump動作時的閃避加速度最大值
- 4：Jump動作時的閃避減速度最大值
- 5：Jump動作時的接近加速度最大值
- 6：Jump動作時的接近減速度最大值

傳回值

用於傳回1以上的整數(%)。

參照

Accel

AccelMax函數範例

是在程式中使用AccelMax函數的範例。

```
'顯示最大加減速度
Print AccelMax(1), AccelMax(2)
```

3.5.6 AccelR

用於設定和顯示針對CP動作的工具姿態變化之加減速度。

格式

(1) AccelR 加速設定值 [, 減速設定值]

(2) AccelR

參數

加速設定值

以實數值(0.1~5000)指定加速設定值。(單位：deg/sec^2)

減速設定值

以實數值(0.1~5000)指定減速設定值。(單位：deg/sec^2)

參數的有效設定值

	加速設定值/減速設定值
VT系列	0.1～1000
C系列、N系列、T系列、G系列、GX系列、RS系列、LS-B系列	0.1～5000

(單位：deg/sec^2)

結果

若省略參數，即顯示目前AccelR設定值。

說明

在Move、Arc、Arc3、BMove、TMove、Jump3CP上使用ROT修飾參數時，AccelR會處於啟用狀態。

在以下任一情況下，AccelR值會被初始化。

- 控制器電源 開啟
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

參照

Arc、Arc3、BMove、Jump3CP、Power、SpeedR、TMove

AccelR範例

```
AccelR 360, 200
```

3.5.7 AccelR函數

用於傳回指定工具姿態變化的加減速度設定值。

格式

AccelR (設定值編號)

參數

設定值編號

以運算式或數值指定以下任一值。

- 1：加速度設定值
- 2：減速度設定值

傳回值

用實數值(單位：deg/sec^2)傳回加速或減速設定值。

參照

AccelR

AccelR函數範例

```
Real currAccelR, currDecelR

'取得目前加減速度
currAccelR = AccelR(1)
currDecelR = AccelR(2)
```

3.5.8 AccelS

設定機器人的直線動作和CP動作之加減速度。(請參閱Move、Arc、Arc3、Jump3、CVMove。)

格式

(1) AccelS 加速設定值 [,減速設定值] [, 閃避加速設定值, 閃避減速設定值, 接近加速設定值, 接近減速設定值]

(2) AccelS

參數

加速設定值

以實數值指定直線動作或CP動作時的加速度(單位：mm/sec^2)。若省略減速設定值，加速設定值則適用於加速和減速之時。

減速設定值

以實數值指定直線動作或CP動作時的減速度(單位：mm/sec^2)。可省略。

閃避加速設定值

以實數值指定Jump3和Jump3CP時的閃避動作之閃避加速度(單位：mm/sec^2)。可省略。

閃避減速設定值

以實數值指定Jump3和Jump3CP時的閃避動作之閃避減速度(單位：mm/sec^2)。可省略。

接近加速設定值

以實數值指定在Jump3和Jump3CP時的接近動作之接近加速度(單位：mm/sec^2)。可省略。

接近減速設定值

以實數值指定在Jump3和Jump3CP時的接近動作之接近減速度(單位：mm/sec^2)。可省略。

參數的有效設定值 (單位：mm/sec^2)

	加速設定值/減速設定值 閃避加速設定值/閃避減速設定值 接近加速設定值/接近減速設定值
N2	0.1～5000
LS20-B, T系列, VT系列	0.1～10000
C4-*901**	0.1～15000
C4-*601**, C8-*1401**, G系列, GX系列, RS系列, LS3-B, LS6-B, LS10-B, C8-*701**W, C8-*901**W, N6, C12	0.1～25000
C8-*701**, C8-*701**R, C8-*901**, C8-*901**R	0.1～35000

結果

若省略參數，即顯示加速值和減速值。

顯示加速值與減速值時，對於加速設定值、減速設定值、退避加速設定值、退避減速設定值、接近加速設定值、接近減速設定值，也顯示根據目前設定的末端夾具重量補償的加速值和減速值。

說明

AccelS用於設定包括直線及圓弧內插在內的各種曲線動作之加速和減速。亦包括Move和Arc命令下的動作。

在以下任一情況下，AccelS值會被初始化，導致減慢加減速度。

- 控制器電源 開啟
- 執行Motor On

- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

注意

- 在低功率模式(Power Low)下執行AccelS命令時

  在低功率模式(Power Low)下執行AccelS時，雖然會儲存新的值，但目前值會受限制。

  將TEACH模式為OFF、功率為High的情況下，AccelS會處於啟用狀態。

- Accel和AccelS的差異

  AccelS命令並非用於設定PTP動作(Go和Jump命令的動作)的加減速度。另一方面，Accel命令則用於設定PTP動作的加減速度。

- 上限值

  SCARA機器人(包括RS系列)AccelS的上限值會因Weight設定和花鍵裝置的位置而異。有關詳細資訊，請參閱以下手冊。
  「機械手手冊 - 設定CP動作時ACCELS」

  垂直六軸機器人的AccelS的上限值會因Weight設定而異。有關詳細資訊，請參閱以下手冊。
  「機械手手冊 - 規格表」

參照

Accel、Arc、Arc3、Jump3、Jump3CP、Power、Move、TMove、SpeedS

AccelS範例

以下為使用事先定義的變數設定Move命令的AccelS和SpeedS的簡易動作程式範例。

```
Function acctest
  Integer slow, accslow, fast, accfast

  slow = 20         '低速度的設定
  fast = 100        '高速度的設定
  accslow = 200     '低加速度的設定
  accfast = 5000    '高加速度的設定
  AccelS accslow
  SpeedS slow
  Move P1
  On 1
  AccelS accfast
  SpeedS fast
  Jump P2
  .
  .
  .
Fend
```

3.5.9 AccelS函數

是用於傳回CP動作加減速度設定值的函數。

格式

AccelS (設定值編號)

參數

設定值編號

以整數值或運算式指定以下任一值。

- 1：加速設定值
- 2：減速設定值
- 3：Jump3和Jump3CP時的閃避加速設定值
- 4：Jump3和Jump3CP時的閃避減速設定值
- 5：Jump3和Jump3CP時的接近加速設定值
- 6：Jump3和Jump3CP時的接近減速設定值
- 7：根據夾具重量補償的加速值
- 8：根據夾具重量補償的減速值
- 9：根據夾具重量補償的Jump3和Jump3CP時的閃避加速值
- 10：根據夾具重量補償的Jump3和Jump3CP時的閃避減速值
- 11：根據夾具重量補償的Jump3和Jump3CP時的接近加速值
- 12：根據夾具重量補償的Jump3和Jump3CP時的接近減速值

傳回值

用於傳回加速設定值或減速設定值(0~5000的實數值、單位：mm/s^2)。

參照

AccelS、Arc3、SpeedS、Jump3、Jump3CP

AccelS函數範例

```
Real savAccelS

savAccelS = AccelS(1)
```

3.5.10 Acos函數

用於傳回指定數值的反餘弦。

格式

Acos (數值)

參數

數值

以實數值指定角度的餘弦。

傳回值

以實數值(單位：弧度)傳回指定數值的反餘弦值。

說明

Acos用於傳回指定數值的反餘弦。指定數值範圍為-1~1。傳回值範圍為0~ π 。數值小於-1或大於1時，會發生錯誤。

使用RadToDeg函數，以便將弧度值轉換為度。

參照

Abs、Asin、Atan、Atan2、Cos、DegToRad、RadToDeg、Sgn、Sin、Tan、Val

Acos函數範例

```
Function acostest
  Double x

  x = Cos(DegToRad(30))
  Print "Acos of ", x, " is ", Acos(x)
Fend
```

3.5.11 Agl函數

是用於傳回指定旋轉關節角度或直動關節位置的函數。

格式

Agl (關節編號)

參數

關節編號

以整數值指定關節編號。範圍：1~機器人的關節數附加軸的S軸為8，T軸為9。

傳回值

用於傳回指定旋轉關節角度或直動關節位置。

說明

Agl函數用於獲得指定旋轉關節角度或直動關節位置。

指定關節為旋轉關節時，則用以「度」為單位的實數值傳回從指定關節角度0開始的旋轉角度。

指定關節為直動關節時，則用以「mm」為單位的實數值傳回從該關節位置0開始的移動量。

依Arm陳述式設定增設型手臂並選擇該增設型手臂時，Agl函數則用於傳回該增設型手臂的從標準手臂脈衝0位置開始的角度或位置。

參照

PAgl、Pls、PPls

Agl函數範例

以下是在命令視窗中使用Print命令的範例。

```
> print agl(1), agl(2)
 17.234  85.355
```

3.5.12 AglToPls函數

用於將機器人各關節角度轉換為脈衝。

格式

AglToPls (關節位置1, 關節位置2, 關節位置3, 關節位置4 [, 關節位置5, 關節位置6] [, 關節位置7] [, 關節位置8, 關節位置9])

參數

關節位置1 - 6
　　以實數值指定關節的角度。
關節位置7
　　以實數值指定第7關節的角度。僅用於關節型7軸機器人。
關節位置8
　　以實數值指定附加軸S關節的角度。
關節位置9
　　以實數值指定附加軸T關節的角度。

傳回值

用於傳回以各關節位置計算的關節脈衝。

說明

要從關節角度轉換為關節脈衝時，則使用AglToPls函數。

注意

在用AglToPls函數指定的關節位置上，機器人的型態屬於奇點時，若將AglToPls的結果指派給點變數，則喪失指定關節位置的部分資訊。這樣的話，若用指派的點變數進行動作，機器人則向不同於用AglToPls指定位置的關節位置上進行動作。在以下範例中，P1向關節位置(0, 0, 0, 0, 0, 90)上進行動作。

```
P1 = AglToPls(0, 0, 0, 90, 0, 0)
Go P1
```

同樣，直接將AglToPls指定為CP動作命令引數時，則向不同於指定位置的關節位置進行動作。在以下範例中，向關節位置(0, 0, 0, 0, 0, 90)進行動作。

```
Move AglToPls(0, 0, 0, 90, 0, 0)
```

作為PTP動作命令的引數使用AglToPls函數時，並無這種奇點的問題。

參照

Agl、JA、Pls

使用AglToPls函數範例

```
Go AglToPls(0, 0, 0, 90, 0, 0)
```

3.5.13 AIO_In函數

用於從選項的類比I/O輸入通道載入類比值。

格式

AIO_In (通道編號)

參數

通道編號
　　指定類比I/O的通道編號。

傳回值

以實數傳回以通道編號指定的類比I/O通道的類比輸入值。傳回值的範圍因類比I/O基板的輸入範圍而定。

說明

In函數

參照

AIO_InW函數、AIO_Out、AIO_OutW、AIO_Out函數、AIO_OutW函數、AIO_Set、Wait

AIO_In函數範例

```
Function main
    Real var1
    var1 = AIO_In(2)  '取得類比輸入通道2的輸入狀態
    If var1 > 5.0 Then
        Go P1
        Go P2
        '在此處執行其它動作命令
        '.
        '.
    Else
        Print "Error in initialization!"
        Print "Sensory Inputs not ready for cycle start"
        Print "Please check analog inputs 2."
    EndIf
Fend
```

3.5.14 AIO_InW函數

用於從選項的類比I/O輸入通道載入類比值。

格式

AIO_InW (通道編號)

參數

通道編號

指定類比I/O的通道編號。

傳回值

用於傳回指定類比I/O通道的輸入狀態(0~65535的Long整數)。

依照類比I/O基板的輸入範圍設定，各輸入通道的輸入電壓(電流)與傳回值的對應關係可分為如下。

輸入資料		設定輸入範圍				
十六進位	十進位	±10.24(V)	±5.12(V)	0-5.12(V)	0-10.24(V)	0-24(mA)
0xFFFF	65535	10.23969	5.11984	5.12000	10.24000	24.00000
0x8001	32769	0.00031	0.00016	2.56008	5.12016	12.00037
0x8000	32768	0.00000	0.00000	2.56000	5.12000	12.00000
0x0000	0	-10.24000	-5.12000	0.00000	0.00000	0.00000

參照

AIO_In函數、AIO_Out、AIO_OutW、AIO_Out函數、AIO_OutW函數、AIO_Set、Wait

AIO_In函數範例

```
Long word0

word0 = AIO_InW(1)
```

3.5.15 AIO_Out

用於從選項的類比I/O輸出通道輸出類比值。

格式

AIO_Out 通道編號, 輸出資料 [, Forced]

參數

通道編號
　　指定類比I/O的通道編號。
輸出資料
　　以運算式或數值指定表示欲輸出的電壓[V]或電流值[mA]的實體l型實數。
Forced
　　可省略。通常會省略。

說明

向以通道編號指定的類比輸出連接埠輸出表示指定電壓[V]或電流[mA]的Real值。透過I/O板上的開關設定類比輸出連接埠的電壓輸出範圍並選擇電壓/電流輸出。指定超出類比I/O板輸出範圍的數值時，則輸出不超過範圍的邊界值(最大值、最小值)。

透過指定通道輸出機器人的速度資訊時，AIO_Out命令會發生錯誤。請於停止輸出速度資訊，並執行AIO_Out。

注意

- Forced旗標

  在緊急停止時或打開安全門而執行類比I/O輸出時，從NoPause工作、NoEmgAbort工作(執行Xqt時，指定NoPause或NoEmgAbort開始的特別工作)指定此旗標。

  在緊急停止時或打開安全門時，類比I/O輸出會發生變化，因此在系統設計上需加以注意。

參照

AIO_In函數、AIO_OutW、AIO_Out函數、AIO_OutW函數、AIO_Set

AIO_Out範例

用於從類比I/O通道#1輸出7.0[V]。

```
AIO_Out 1, 7.0
```

3.5.16 AIO_Out函數

以實數值傳回透過選項的類比I/O輸出通道輸出的類比值。

格式

AIO_Out (通道編號)

參數

通道編號

指定類比I/O的通道編號。

傳回值

以實數值傳回指定類比I/O通道的電壓/電流輸出狀態。電壓輸出時的單位為[V]，電流輸出時的單位為[mA]。

即使通過指定通道輸出機器人的速度資訊，也可執行本函數。

參照

AIO_In函數、AIO_Out、AIO_OutW、AIO_OutW函數、AIO_Set、Wait

AIO_Out函數範例

```
Real rdata01

rdata01 = AIO_Out(1)
```

3.5.17 AIO_OutW

用於從選項的類比I/O輸出通道輸出16位元類比值。

格式

AIO_OutW 通道編號, 輸出資料 [, Forced]

參數

通道編號
　　指定類比I/O的通道編號。
輸出資料
　　以運算式或數值指定輸出資料(0~65535的整數)。
Forced
　　可省略。通常會省略。

說明

用於向以通道編號指定的類比I/O通道輸出。以運算式或數值指定的輸出資料是0~65535的整數。

如下所述為依照I/O板上開關設定的輸出範圍輸出的電壓(電流)。

輸出資料		設定輸出範圍					
十六進位	十進位	±10(V)	±5(V)	0-5(V)	0-10(V)	4-20(mA)	0-20(mA)
0xFFFF	65535	9.99970	4.99985	5.00000	10.00000	20.00000	20.00000
0x8001	32769	0.00031	0.00015	2.50008	5.00015	12.00024	10.00031
0x8000	32768	0.00000	0.00000	2.50000	5.00000	12.00000	10.00000
0x0000	0	-10.00000	-5.00000	0.00000	0.00000	4.00000	0.00000

注意

- Forced旗標

  在緊急停止時或打開安全門而執行類比I/O輸出時，從NoPause工作, NoEmgAbort工作(執行Xqt時，指定NoPause或NoEmgAbort開始的特別工作)以及背景工作指定此旗標。

  在緊急停止時或打開安全門時，類比I/O輸出會發生變化，因此在系統設計上需加以注意。

參照

AIO_In函數、AIO_Out、AIO_Out函數、AIO_OutW函數、AIO_Set、Wait

AIO_OutW範例

```
AIO_OutW 1, &H8000
```

3.5.18 AIO_OutW函數

以0~65535的Long整數傳回透過選項的類比I/O輸出通道輸出的類比值。

格式

AIO_OutW (通道編號)

參數

通道編號

指定類比I/O的通道編號。

傳回值

以0~65535的Long整數傳回指定類比I/O通道的輸出狀態。

依照類比I/O基板的輸出範圍設定，各輸出通道的輸出電壓(電流)與傳回值的對應關係可分為如下。

輸出資料		設定輸出範圍					
十六進位	十進位	±10(V)	±5(V)	0-5(V)	0-10(V)	4-20(mA)	0-20(mA)
0xFFFF	65535	9.99970	4.99985	5.00000	10.00000	20.00000	20.00000
0x8001	32769	0.00031	0.00015	2.50008	5.00015	12.00024	10.00031
0x8000	32768	0.00000	0.00000	2.50000	5.00000	12.00000	10.00000
0x0000	0	-10.00000	-5.00000	0.00000	0.00000	4.00000	0.00000

即使通過指定通道輸出機器人的速度資訊，也可執行本函數。

參照

AIO_In函數、AIO_Out、AIO_OutW、AIO_Out函數、AIO_Set、Wait

AIO_OutW函數範例

```
Long word0

word0 = AIO_OutW(1)
```

3.5.19 AIO_Set

用於向選項的類比I/O輸出通道輸出機器人的速度資訊。

格式

(1) AIO_Set 通道編號, On, {RefTCPSpeed | RealTCPSpeed | RefECPSpeed | RealECPSpeed }, 最大輸出時速度 [, 最小輸出時速度] [, Cnv , 傳送帶編號]
(2) AIO_Set 通道編號, Off
(3) AIO_Set [通道編號]

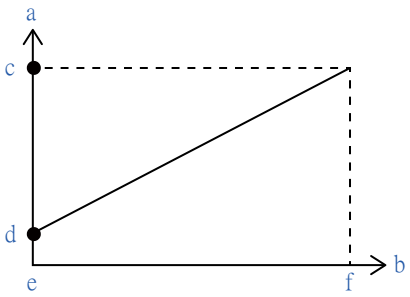
參數

通道編號
　　指定類比I/O的通道編號。
On
　　以運算式或數值指定輸出資料(0~65535的整數)。
Off
　　結束速度資訊的類比輸出，並初始化為輸出0。
RefTCPSpeed
　　輸出目前選擇的TCP命令速度。
RealTCPSpeed
　　輸出目前選擇的TCP實際速度。
RefECPSpeed
　　輸出目前選擇的ECP命令速度。
RealECPSpeed
　　輸出目前選擇的ECP實際速度。
最大輸出時速度
　　以運算式或數值指定表示輸出範圍最大值時速度的實體型實數值(單位[mm/s])。
最小輸出時速度
　　以運算式或數值指定表示輸出範圍最小值時速度的實體型實數值(單位[mm/s])。可省略。省略時為「0」[0mm/s]。
Cnv
　　輸出與傳送帶相對的TCP速度。和輸送帶編號一起指定。
傳送帶編號
　　指定用於計算相對TCP速度的傳送帶編號。

說明

以類比電壓和電流即時向以通道編號指定的類比I/O通道輸出選擇的機器人TCP(工具中心點)或ECP(外部控制點)的速度。以類比I/O基板上的開關及跳線器選擇類比電壓/電流並設定輸出範圍。

根據指定的最小輸出時速度以及最大輸出時速度，按下圖所示的線性內插確定與輸出範圍的最小值/最大值相應的機器人速度。



符號	說明
a	尖端速度 [mm/s]
b	輸出 [V or mA]

符號	說明
c	最大輸出時速度
d	最小輸出時速度
e	最小輸出
f	最大輸出

若指定了命令速度(RefTCPSpeed或RefECPSpeed)，則基於賦予機器人的命令值輸出理想的速度波形。

若指定了實際速度(RealTCPSpeed或RealECPSpeed)，則輸出基於機器人的實際動作(各關節編碼器值)算出的速度波形。

指定TCP(RefTCPSpeed或RealTCPSpeed)時，則輸出目前選擇的工具(預設Tool 0)的中心點速度。

指定ECP(RefECPSpeed或RealECPSpeed)時，則輸出目前選擇的外部控制點(ECP)速度。若未在此選擇ECP(ECP = 0時)，則始終進行最小值。

不能同時指定 ECP 和Cnv。

指定的輸送帶編號，只能是配置在機械臂上並且進行了輸送帶校準的輸送帶編號。

僅指定通道編號時，則顯示指定類比I/O通道的輸出設定資訊。已省略所有引數時，則顯示所有類比I/O通道的輸出設定資訊。

參照

AIO_In函數、AIO_Out、AIO_Out函數、AIO_Out、AIO_OutW函數、AIO_Set、Wait

AIO_Set範例

在類比輸出通道1上設定機器人1、工具1的TCP實際速度輸出。

對機器人動作中的速度進行類比輸出後，解除速度輸出設定。

```
Robot 1
Tool 1
Motor On
Power High
SpeedS 2000
AccelS 5000
AIO_Set 1, On, RealTCPSpeed, 2000.0, 0.0
Move P1
AIO_Set 1, Off
```

3.5.20 AIO_Set函數

用於傳回選項的類比I/O輸出通道所設定的機器人速度輸出的設定資訊。

格式

AIO_Set (通道編號、指數)

參數

通道編號

指定類比I/O的通道編號。

指數

以整數值指定要讀取的設定資訊指數。

傳回值

如下所示為可用AIO_Set函數取得的資訊。

指數	資訊
1	On(1) / Off(0)
2	RefTCPSpeed(0)/ RealTCPSpeed(1)/ RefECPSpeed(2)/ RealECPSpeed(3)
3	最大輸出時速度 [mm/s]
4	最小輸出時速度 [mm/s]
5	輸送帶編號 未設置(0)/輸送帶編號 (1~16)

參照

AIO_In函數、AIO_Out、AIO_OutW、AIO_Out函數、AIO_OutW函數、AIO_Set、Wait

AIO_Set函數範例

```
Print "Analog Ch#1 speed output is: ", AIO_Set(1, 1)
```

3.5.21 AIO_TrackingSet

用於設定距離追蹤功能。

格式

(1) AIO_TrackingSet
通道編號, 測量值和距離的轉換係數, 距離0mm的測量值, 可追蹤範圍的下限值, 可追蹤範圍的上限值 [,超出可追蹤範圍時的動作 [,進行距離追蹤的軸]]
(2) AIO_TrackingSet
通道編號

參數

通道編號
以1~8的整數值指定連接所使用距離感測器的類比I/O通道編號。
測量值和距離的轉換係數
是將距離感測器的測量值(V、mA)轉換為距離(mm)的係數。以除0之外的-500~500的實數進行指定。(單位：mm/V、mm/mA)
距離0mm的測量值
用以下範圍的實數指定距離(位移計時：位移量)為0mm時的電壓或電流值。(單位：V、mA)
設定類比I/O基板輸入範圍內的設定值。

設定輸入範圍	最小值	最大值
±10.24 V	-10.24 V	10.24 V
±5.12 V	-5.12 V	5.12 V
0-5.12 V	0 V	5.12 V
0-10.24 V	0 V	10.24 V
0-24 mA	0 mA	24 mA

可追蹤範圍的下限值
可追蹤範圍的下限值是執行距離追蹤功能時容許的位移量下限值。以-300~300的實數指定下限值。(單位：mm)
請指定大於距離感測器可測量範圍下限值的值。可追蹤範圍的下限值用於指定小於可追蹤範圍上限值的值。
可追蹤範圍的上限值
可追蹤範圍的上限值是執行距離追蹤功能時容許的位移量上限值。以-300~300的實數指定上限值。(單位：mm)
請指定小於距離感測器可測量範圍上限值的值。可追蹤範圍的上限值用於指定大於可追蹤範圍下限值的值。
超出可追蹤範圍時的動作
以0~1的整數指定在可追蹤範圍(上述情形為下限值~上限值之間)外時停止或繼續機器人動作。可省略。省略時被設為「0」。如下所述為常數。

常數	值	內容
AIOTRACK_ERRSTOP	0	超出可追蹤範圍時，機器人因錯誤而停止。
AIOTRACK_CONTINUE	1	超出可追蹤範圍時，機器人繼續進行動作。

進行距離追蹤的軸
以0~5的整數指定要進行距離追蹤的軸。請指定與使用的距離感測器測量方向一致的軸。
可省略。省略時被設為「2」。如下所述為常數。

常數	值	內容
AIOTRACK_TOOL_X	0	Tool坐標X軸
AIOTRACK_TOOL_Y	1	Tool坐標Y軸
AIOTRACK_TOOL_Z	2	Tool坐標Z軸

常數	值	內容
AIOTRACK_ECP_X	3	ECP坐標X軸
AIOTRACK_ECP_Y	4	ECP坐標Y軸
AIOTRACK_ECP_Z	5	ECP坐標Z軸

唯有啟用外部控制點動作(ECP)選項時，方可指定3~5。

結果

若為格式2，則在控制台上顯示目前設定值。

以下為上述參數名稱與控制台上顯示的參數名稱的對應表。

參數名稱	控制台顯示名稱
測量值和距離的轉換係數	ScaleFactor
距離0mm的測量值	RefVoltage
可追蹤範圍的下限值	ThresholdMin
可追蹤範圍的上限值	ThresholdMax
超出可追蹤範圍時的動作	OutOfRangeMode
進行距離追蹤的軸	TrackingAxis

如下所述為顯示範例。

例1：設定通道編號1時
Ch1: ScaleFactor 1.000 [V/mm or mA/mm] RefVoltage 0.000 [V or mA] ThresholdMin -10.000 [mm] ThresholdMax 10.000 [mm] OutOfRangeMode AIOTRACK_ERRSTOP TrackingAxis AIOTRACK_TOOL_Z

例2：未設定通道編號1時
Ch1: Undefined

說明

AIO_TrackingSet用於設定距離追蹤功能的參數。設定參數取決於使用的距離感測器和實施本功能的環境。在接通控制器電源後，執行AIO_TrackingStart之前，務必執行AIO_TrackingSet。設定的參數會被保持到關閉機器人控制器電源或重新啟動為止。

以下將對參數的詳細內容進行說明。

測量值和距離的轉換係數

若為表示每+1V位移+2mm的距離感測器，轉換係數則為2。屆時，+2mm是向長距離方向進行位移的長度。有些位移計可能會對於短距離方向的位移將電壓設為正電壓。屆時，轉換係數則為負值。

距離0mm的測量值

距離感測器(特別是位移計)的距離0mm時的電壓或電流值因各產品而異。此外，也會依使用者的設定而得以自由設定距離0mm時的電壓或電流值。請依使用的距離感測器設定，指定數值。距離(或位移量)為0mm且距離感測器輸出電壓為0V時，本參數即為「0」。

可追蹤範圍的上、下限值

依應用程式所容許的偏差設定上/下限值。請務必將要設定的值設為距離感測器可測量範圍內的值。距離感測器的可測量範圍因各感測器或使用者的設定而異。啟用距離追蹤功能前，請務必進行確認。若將本參數設為超出距離感測器可測量範圍的數值，便無法正確啟用距離追蹤功能，而且機器人有可能進行非預期的動作。

超出可追蹤範圍時的動作

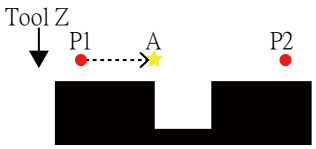
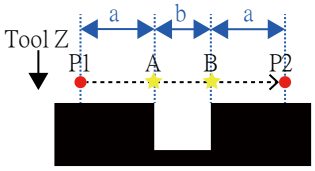
下圖表示，在朝Tool的Z方向啟用距離追蹤功能的情況下，將「超出可追蹤範圍時的動作」參數設定為「0」和「1」時呈現出之機器人的移動軌跡。

- P1：距離追蹤開始位置
- P2：目標位置

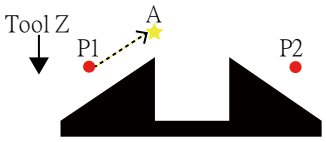
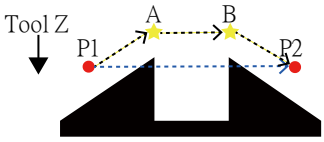
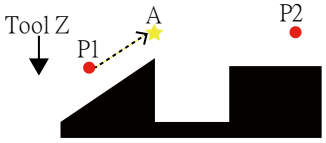
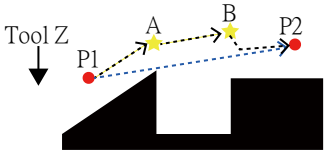
該圖所示為在A點超出計測範圍、在B點返回計測範圍內的標的物之情形。

距離追蹤功能啟用時，以啟用功能的位置P1的Tool Z方向測量值(位移量)為基準值，對機器人進行控制，以便使測量值始終成為基準值。因此，從P1移至P2時，機器人會進行移動，以便使P1到A點的測量值保持恆定。

到達A點時，若被設定為「0」，機器人則在A點因錯誤而停止。若設為「1」，則從A點朝P2繼續移動。但是，由於超出可追蹤範圍，因此不進行追蹤。由於從B點開始進入到可追蹤範圍內，因此機器人同樣會進行移動，以便P1到A點的測量值保持恆定。

0：機器人因超出範圍而停止動作	1：機器人超出範圍但繼續動作
	
	■ a：距離追蹤功能 啟用 ■ b：超出範圍，距離追蹤功能 停用

設定「1」時，超出可追蹤範圍的機器人動作是指，從開始位置(P1)向目標位置(P2)進行CP動作的軌跡動作。下圖所示為，超出可追蹤範圍的A點-B點間的軌跡與P1-P2間的軌跡平行的情形。到達B點後，由於在可追蹤範圍內，因此可能會啟用以計測值為基準值的機器人控制功能，導致機器人突然移動。

0：機器人因超出範圍而停止動作	1：機器人超出範圍但繼續動作
	
	

⚠ 注意

若未正確設定各參數，在執行AIO_TrackingStart時，機器人有可能進行非預期的動作。

請依使用的設備和實施環境進行適當的設定。

如有異常動作，請立即按下緊急停止按鈕。

參照

AIO_TrackingStart、AIO_TrackingEnd、AIO_TrackingOn函數

AIO_TrackingSet範例

這是將P1作為動作開始位置，將P2作為動作結束位置，然後利用距離追蹤功能移動機器人的程式範例。

> ⚠ 注意
>
> 範例中所設定的參數為參考值。
>
> 有可能因設定的參數或動作環境而發生作業不成功或振動動作的情況。
>
> 如有異常動作，請立即按下緊急停止按鈕。

```
Function Main
  Motor On
  Power High
  SpeedS 30
  AccelS 300300

  Go P1    ' 移至開始位置P1
  AIO_TrackingSet 1,1,0,-5,5,0,2    ' 設定距離追蹤功能
  AIO_TrackingStart 1,5,5,5    ' 啟用距離追蹤功能
  Move P2    ' 在執行距離追蹤功能的同時移至P2
  AIO_TrackingEnd    ' 停用距離追蹤功能
  Motor Off
Fend
```

3.5.22 AIO_TrackingStart

用於啟動距離追蹤功能。

格式

AIO_TrackingStart 通道編號,ProportionalGain [,IntegralGain [,DifferentialGain]]

參數

通道編號

以1~8的整數值指定連接所使用距離感測器的類比I/O通道編號。

ProportionalGain

以除了0以外之50以下正實數指定距離追蹤功能的比例增益。由於最佳值會因機器人移動速度或工件形狀等而發生變化，因此，需依據使用環境進行設定。

IntegralGain

以100以下的正實數指定距離追蹤功能的積分增益。可省略。省略時被設為「0」。若要提高距離追蹤精度，請調整積分增益。

DifferentialGain

以100以下的正實數指定距離追蹤功能的微分增益。可省略。省略時被設為「0」。若要提高距離追蹤精度，請調整微分增益。

說明

距離追蹤功能的目的在於，用連接於類比I/O的距離感測器的計測值，對機器人進行控制，以便使機器人和工件之間的距離保持恆定。

控制機器人的軸方向為，以AIO_TrackingSet的參數「進行距離追蹤的軸」指定的1個軸之方向。若將要保持的距離作為「基準值」，執行本命令時，距離感測器所測量的距離即為基準值。

透過執行AIO_TrackingStart啟用距離追蹤功能；透過執行AIO_TrackingEnd停用該功能。在執行AIO_TrackingEnd之前，持續啟用距離追蹤功能。不使用距離追蹤功能時，請立即執行AIO_TrackingEnd，以停用距離追蹤功能。

在執行AIO_TrackingSet之前已執行AIO_TrackingStart時，會發生錯誤。請務必在執行AIO_TrackingSet後執行AIO_TrackingStart。

可使用距離追蹤功能的機器人機種有水平多關節型(包含RS系列)和垂直6軸型(包含N系列)。

機器人在執行距離追蹤功能時雖可進行動作，但僅限於CP動作。無法執行PTP動作。

在執行距離追蹤功能時，若通過特定點附近，會發生錯誤。

不能在執行距離追蹤功能時執行以下命令。

變為MOTOR OFF的命令	Motor off, SFree
PTP動作命令	BGo, Go, JTran, Jump, Jump3, Jump3CP, JumpTLZ, Pass, Ptran, Pulse, TGo
壓力控制執行命令	FCKeep, 附帶FC的動作命令, FS#.Reset, FS.Reboot
扭矩控制執行命令	TC
輸送帶追蹤執行命令	動作命令 + Cnv_QueGet
VRT執行命令	VRT, VRT_CPMotion

設定變更命令	AIO_TrackingSet, Arm, ArmSet, Base, Calib, CalPls, ECP, ECPSet, Hofs, Inertia, MCal, Power, TLSet, Tool, Weight (唯有在進行與使用中編號相關的變更時，AIO_TrackingSet, ArmSet, ECPSet, TLSet才會發生錯誤。)
其它	Brake, Here, Home, VCal, WaitPos

ProportionalGain、IntegralGain、DifferentialGain的設定

ProportionalGain時，設定值越大，機器人越是快速追蹤。但數值過大時，有可能會因機器人的快速動作而發生錯誤。

可省略IntegralGain和DifferentialGain。若要提高補償精度，需進行設定。

若未設定適當的值，機器人有可能進行高速動作或發生振蕩。

有關各增益的設定方法，請參閱以下手冊。
「Epson RC+ 使用者指南 - 距離追蹤功能」

⚠ 注意

ProportionalGain、IntegralGain、DifferentialGain時，若設定過大的值，機器人有可能進行非預期的動作。

各參數值請逐漸變更為較大的數值。若一次變更為較大數值，則機器人有可能進行十分危險的非預期動作。

如有異常動作，請立即按下緊急停止按鈕。

參照

AIO_TrackingSet、AIO_TrackingEnd、AIO_TrackingOn函數

AIO_TrackingStart範例

這是將P1作為動作開始位置，經由P2，然後將P3作為動作結束位置，最後利用距離追蹤功能移動機器人的程式範例。

⚠ 注意

範例中所設定的參數為參考值。

有可能因設定的參數或動作環境而發生作業不成功或振動動作的情況。

此外，如有異常動作，請立即按下緊急停止按鈕。

```
Function Main
  Motor On
  Power High
  SpeedS 30
  AccelS 300300

  Go P1     ' 移至開始位置P1
  AIO_TrackingSet 1,1,0,-5,5,0,2     ' 設定距離追蹤功能
  AIO_TrackingStart 1,1,0,0     ' 啟用距離追蹤功能
  Move P2     ' 在執行距離追蹤功能的同時移至P2
  Move P3     ' 在執行距離追蹤功能的同時移至P3
  AIO_TrackingEnd     ' 停用距離追蹤功能
  Motor Off
Fend
```

3.5.23 AIO_TrackingEnd

停用距離追蹤功能。

格式

AIO_TrackingEnd

說明

停用以AIO_TrackingStart啟用的距離追蹤功能。

參照

AIO_TrackingSet、AIO_TrackingStart、AIO_TrackingOn函數

AIO_TrackingEnd範例

這是將P1作為動作開始位置，經由P2，然後將P3作為動作結束位置，最後利用距離追蹤功能移動機器人的程式範例。

> ⚠ 注意
>
> 範例中所設定的參數為參考值。
>
> 有可能因設定的參數或動作環境而發生作業不成功或振動動作的情況。
>
> 此外，如有異常動作，請立即按下緊急停止按鈕。

```
Function Main
  Integer ChNo
  Motor On
  Power High
  SpeedS 30
  AccelS 300300
  ChNo=1

  Go P1    ' 移至開始位置P1
  AIO_TrackingSet ChNo,10,0,-3,3,0,2    ' 設定距離追蹤功能
  AIO_TrackingStart ChNo,1,0,0    ' 啟用距離追蹤功能
  Move P2    ' 在執行距離追蹤功能的同時移至P2
  Move P3    ' 在執行距離追蹤功能的同時移至P3
  AIO_TrackingEnd    ' 停用距離追蹤功能
  Motor Off
Fend
```

3.5.24 AIO_TrackingOn函數

用於傳回指定機器人是否執行距離追蹤功能。

格式

AIO_TrackingOn (機器人編號)

參數

機器人編號

以運算式或數值指定要讀取狀態的機器人編號。

傳回值

用於在執行距離追蹤功能時傳回True (-1)，在停止時傳回False (0)。

參照

AIO_TrackingSet、AIO_TrackingStart、AIO_TrackingEnd

AIO_TrackingOn範例

```
Function Main
  Integer i
  i = AIO_TrackingOn(1)
  print i
Fend
```

命令視窗中的範例

```
> print AIO_TrackingOn(1)
0
```

3.5.25 Align函數

用於傳回點資料，該點資料透過轉換將在指定點上的機器人工具坐標系姿態(U、V、W)排列在指定本地坐標系之最近的坐標軸或指定的坐標軸上。

格式

Align (點指定[, 本地坐標系編號[, 指定坐標軸]])

參數

點指定

指定作為對象的點數據。

本地坐標系編號

指定要用作姿態排列標準的本地坐標系編號。一旦省略，則指定基本坐標系。

坐標軸指定

指定欲排列的坐標軸。一旦省略，則排列成最近的坐標軸。

常數	值	
COORD_X_PLUS	1：	+X軸
COORD_Y_PLUS	2：	+Y軸
COORD_Z_PLUS	3：	+Z軸
COORD_X_MINUS	4：	-X軸
COORD_Y_MINUS	5：	-Y軸
COORD_Z_MINUS	6：	-Z軸

說明

在垂直6軸型機器人(包含N系列)中，有可能要固定依點資料所定義的工具坐標系位置(原點)，同時只改變姿態，以排列成特定坐標系。Align函數用於進行轉換，透過轉換將指定點資料的姿態資料(U、V、W值)排列在指定本地坐標系的最近坐標軸或指定坐標軸上。

對於非垂直6軸型(包含N系列)機器人，直接傳回指定點。

參照

AlignECP函數、LJM函數

Align函數範例

```
Move Align(P0) ROT

P1 = Align(P0, 1)
Move P1 ROT

P2 = Align(P0, 1, 3)
Move P2 ROT
```

3.5.26 AlignECP函數

用於傳回點資料，該點資料透過轉換將在指定點上的機器人工具坐標系姿態(U、V、W)排列在指定ECP坐標系的最近坐標軸上。

格式

AlignECP (點指定, ECP坐標系編號)

參數

點指定

　　指定作為對象的點數據。

ECP坐標系編號

　　指定要用作姿態排列標準的ECP坐標系編號。

說明

在垂直6軸型機器人(包含N系列)中，有可能要固定依點資料所定義的工具坐標系位置(原點)，同時只改變姿態，以排列成特定坐標系。AlignECP函數用於進行轉換，透過轉換將指定點資料的姿態資料(U、V、W值)排列在指定ECP坐標系的最近坐標軸上。

對於非垂直6軸型(包含N系列)機器人，直接傳回指定點。

參照

Align函數、LJM函數

AlignECP函數範例

```
Move AlignECP(P0) ROT

P1 = AlignECP(P0, 1)
Move P1 ROT
```

3.5.27 And運算子

執行2個數值的And運算(邏輯或位元)。

格式

result = 值1 And 值2

參數

值1、值2

用於在邏輯And運算中指定邏輯值的傳回值。在位元And運算中，用於指定整數運算式。

result

用於在邏輯And運算中傳回邏輯值。在位元And運算中，用於傳回整數。

說明

邏輯And運算用於結合2個以上的值，並導出Bolean型結果。下表所示為表示And運算的模式。

值1	值2	result
True	True	True
True	False	False
False	True	False
False	False	False

位元And運算用於以位元為單位對2個數值進行比較，並依下表將相應的位元導出為result。

值1的位元	值2的位元	result
0	0	0
0	1	0
1	0	0
1	1	1

參照

LShift、Mask、Not、Or、RShift、Xor

And運算子範例

```
Function LogicalAnd(x As Integer, y As Integer)

  If x = 1 And y = 2 Then
    Print "The values are correct"
  EndIf
Fend

Function BitWiseAnd()

  If (Stat(0) And &H800000) = &H800000 Then
    Print "The enable switch is open"
  EndIf
Fend

>print 15 and 7
```

```
7
>
```

3.5.28 AOpen

用新增模式(追加寫入)開啟檔案。

格式

AOpen 檔名As #檔案編號
.
.
Close #檔案編號

參數

檔案名稱
　　指定包含路徑的檔案名稱字串。僅指定檔名時，即指目前目錄中的檔案。詳細內容請參閱ChDisk。
檔案編號
　　以30~63的整數值或運算式來指定。

說明

以指定的檔案編號開啟指定的檔案。此陳述式用於在指定的檔案中新增(追加寫入)。沒有指定的檔案時，則新建檔案。

在檔案處於開啟狀態下，指定的檔案編號用於識別該檔案。因此，在關閉該檔案之前，不可將相同的檔案編號用於其它檔案。在檔案操作命令(Print#, Write, Flush, Close)中使用檔案編號。

以Close陳述式關閉檔案，並釋放檔案編號。

請以FreeFile函數取得檔案編號，以避免將同一編號用在多項工作上。

注意

- 可使用網路路徑。
- 寫入檔案時會被緩衝。
- 可用Flush陳述式寫入被緩衝的資料。以Close陳述式關閉檔案時，也進行寫入。

參照

Close、Print#、BOpen、ROpen、UOpen、WOpen、FreeFile、Flush

AOpen範例

```
Integer fileNum, i
fileNum = FreeFile
WOpen "TEST.DAT " As #fileNum
For i = 0 To 100
   Print #fileNum, i
Next I
Close #fileNum
....
....
....
FileNum = FreeFile
AOpen "TEST.DAT" As #FileNum
For i = 101 to 200
   Print #FileNum, i
Next i
Close #FileNum
```

3.5.29 Arc, Arc3

Arc用於在基礎坐標系的XY平面上，以曲線動作將手臂從目前位置移至指定位置。

Arc3用於依三維方式，以曲線動作將手臂從目前位置移至指定位置。

無論機器人是水平多關節型(包含RS系列)或垂直6軸型(包含N系列)，皆可使用這2個命令。

格式

Arc
(1) Arc 經由坐標, 目標坐標 [ROT] [CP] [Till | Find] [!平行處理!] [SYNC]
(2) Arc 目標坐標, 半徑, 長短, 旋轉方向 [ROT] [CP] [Till | Find] [!平行處理!] [SYNC]
(3) Arc 目標坐標, 旋轉角 [ROT] [CP] [Till | Find] [!平行處理!] [SYNC]
(4) Arc 目標坐標, 中心坐標, 長短 [ROT] [CP] [Till | Find] [!平行處理!] [SYNC]

Arc3
(1) Arc3 經由坐標, 目標坐標 [ROT] [ECP] [CP] [Till | Find] [!平行處理!] [SYNC]
(2) Arc3 目標坐標, 中心坐標, 長短 [ROT] [ECP] [CP] [Till | Find] [!平行處理!] [SYNC]

參數

中間坐標
　以點數據或XY函數來指定。這是手臂在從目前位置移至目標坐標的軌跡上務必通過的點。
目標坐標
　以點數據或XY函數來指定。是手臂以圓弧動作進行移動的到達地點、目標位置。
中心坐標
　以點數據或XY函數來指定。是手臂進行圓弧動作時的中心坐標。Arc命令時，Z坐標與姿勢 (U,V,W) 無法利用。
　Arc3命令時，在通過現在地、目標坐標、中心坐標的X, Y, Z坐標的平面上進行圓弧動作。此時，中心坐標的姿勢 (U,V,W) 無法利用。
半徑
　圓弧動作的半徑以實數或運算式 (單位：mm) 指定。
長短
　指定圓弧從目前位置到目標坐標是通過短路徑或是通過長路徑。通過短路徑時，指定為ARC_SHORT；通過長路徑時指定為ARC_LONG。
旋轉方向
　指定圓弧動作的旋轉方向。相對於基礎坐標系的XY平面，Z軸方向逆時針時指定為ARC_PLUS；順時針時指定為ARC_MINUS。
旋轉角
　圓弧動作的旋轉角度以實數或運算式 (單位：度) 指定。
　指定小於360度、大於-360度的值。相對於基礎坐標系的XY平面，Z軸方向逆時針時指定為PLUS；順時針時指定為MINUS。
ROT
　以工具姿態變化為優先，決定動作速度和加減速度。可省略。
ECP
　用於指定外部控制點的動作。可省略。(唯有使用ECP選項時方可啟用)
CP
　指定路徑運動。可省略。
Till | Find
　用於記述Till或Find運算式。可省略。

```
Till | Find
Till Sw(運算式) = {On | Off}
Find Sw(運算式) = {On | Off}
```

! 平行處理 !
　可在Arc陳述式上使用平行處理陳述式。可省略。(關於詳細內容，請參閱「平行處理」。)
SYNC

用於預約動作命令。在以SyncRobots開始動作之前，機器人不進行動作。可省略

說明

Arc和Arc3用於在將手臂從目前位置移至目標坐標的過程中通過經由坐標並以圓弧內插進行動作的情況。根據被賦予的3點 (目前地、經由坐標、目標坐標)，或者被賦予的2點 (目前地、目標坐標) 與半徑、旋轉角度、中心坐標等參數，自動運算曲線動作軌跡，沿著該軌跡，將手臂移到目標坐標。

而對於6 軸機器人，U、V和W 座標以從當前位置到目標座標的最短旋轉姿勢移動。不會通過在中間坐標中指定的姿勢 (U, V, W)。

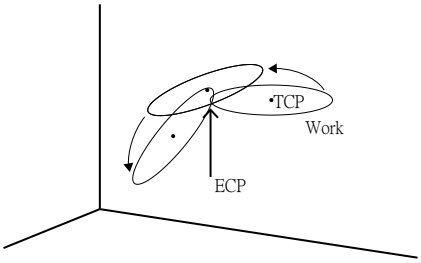
如果使用，請事先檢查實際移動。

Arc和Arc3的速度和加減速度分別使用SpeedS和AccelS的設定值。關於速度和加減速度的關係，請參閱「注意」項目中的「連同CP一起使用Arc、Arc3」。但是，指定ROT修飾參數時的速度和加減速度分別使用SpeedR和AccelR的設定值。此時，SpeedS和AccelS的設定值即呈停用狀態。

機器人的工具中心點位置固定為特定坐標，僅改變工具姿態時，或者相對於工具中心點的移動距離，工具姿態變化較大時，工具姿態變化速度可能會明顯變快。為防止這種情形，工具姿態變化速度大時，自動限制動作速度的功能就會啟用。

想要手動設定CP動作時的工具姿態變化速度的上限值時，請啟用SpeedRLimitation。啟用SpeedRLimitation時，CP動作時的工具姿態變化速度超過設定的SpeedR時，工具姿態變化速度會變為SpeedR來限制動作速度。工具姿態變化速度超過設定的SpeedR時，以設定的SpeedS進行動作。工具姿態變化速度的上限值，請事先在SpeedR設定。

使用ECP時(僅限於Arc3)，工件會在與指定ECP編號相應的外部控制點上，沿著圓弧軌跡進行動作。此時的尖端關節中心不會沿著圓弧軌跡。



Arc動作速度和加速度之設定

分別以SpeedS和AccelS進行相對於Arc和Arc3命令的速度和加減速度之設定。用SpeedS指定速度(單位：mm/sec)；用AccelS指定加減速度(單位：mm/sec^2)。

注意

- 唯有在水平面上，方可啟用Arc命令

  依Arc命令描繪的軌跡為在XY平面上的真圓弧。不通過在中間坐標中指定的Z坐標和姿勢(U, V, W)，對目前的點和目標坐標值進行內插。
  Arc3可用於指定三維空間中的圓弧軌跡。通過在中間坐標中指定的Z坐標，但不通過姿勢(U, V, W)，對目前的點和目標坐標值進行內插。

- 確認相對於Arc命令的範圍

  Arc和Arc3陳述式無法在Arc動作前執行運算以確認軌跡範圍。因此，即便目的位置在動作區域內，當軌跡超出區域時，便有可能停止。此時有可能產生衝擊，或者對手臂造成損害，因此，需事先以低速運行程式，以確認軌跡。

- 設定Arc動作時

由於從目前位置開始進行Arc命令的圓弧動作，因此在執行Arc和Arc3前，有時需要事先利用Go或Jump等其它相關動作命令，將機器人手臂移至適當的位置。

- 連同CP一起使用Arc、Arc3

  若使用CP參數，則在開始減速的同時，移至下一個陳述式控制動作命令。其便利之處在於，使用者可連接數個動作命令，以恆定的速度執行連續動作。在未指定CP的Arc命令、Arc3命令時，手臂務必減速，並停於指定的目標坐標上。

- 以輸送帶追蹤利用Arc, Arc3命令時

  未指定經由坐標的Arc, Arc3格式，無法載入在Cnv_QueGet取得的點數據而加以利用。請使用經由坐標指定的Arc, Arc3格式。

- 圓弧角度在±180度、±360度左右時

  未指定經由坐標的Arc, Arc3格式時，圓弧角度為±180度左右或±360度左右的話，有可能無法形成預想軌跡。此外，在Arc3，以圓弧角度為±180度指定中心坐標、目標坐標時，因圓弧無法特定而出現錯誤。圓弧角度在±180度、±360度左右的話，請使用中間坐標指定的Arc, Arc3格式。

- 半徑指定Arc命令的限制

  半徑請設定為現在地到目標坐標間距離的1/2以上的值。指定的半徑過短時，會由於無法描繪圓弧而發生錯誤。

- 旋轉角指定Arc命令的限制

  旋轉角請指定小於360度、大於-360度的值。0度無法形成圓弧，會發生錯誤。

常見錯誤

- 變更抓手(臂腕)的屬性

  使用Arc命令時，請注意各點的抓手屬性。一旦變更曲線動作之間的抓手方向(例如，右臂腕變左臂腕或進行反向變更等)，會發生錯誤。手臂的屬性值(/L左臂腕、/R右臂腕)務必與實際的目前位置、經由坐標和目標坐標一致。

- 欲將手臂移出活動範圍時

  倘若透過指定的圓弧動作將手臂移出活動範圍，則會發生錯誤。

- 欲以幾近直線動作時

  或者指定的雖是圓弧動作，但幾近直線時，會發生錯誤。請利用Move命令。

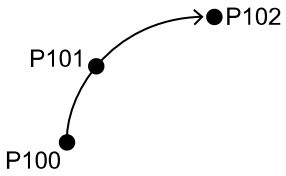
參照

! 平行處理 !、AccelS、Move、SpeedS

Arc, Arc3範例 (指定經由坐標)

如下所述為用於描繪如下軌跡的程式之範例。
從P100開始動作，經由P101沿著到達P102的圓弧動作軌跡。

```
Function ArcTest
   Go P100
   Arc P101, P102
Fend
```



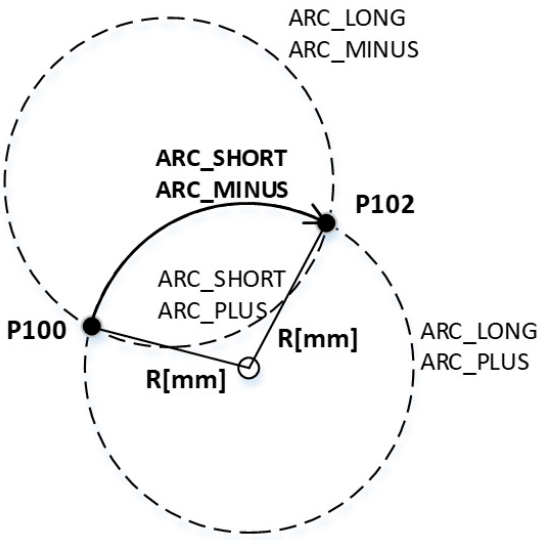
提示

首次使用Arc命令時，建議使用活動範圍中位於機器人側的點，嘗試描繪簡單的圓弧。屆時，請設想實際描繪的圓弧軌跡。請勿教導會導致手臂移至超出平常活動範圍的點。

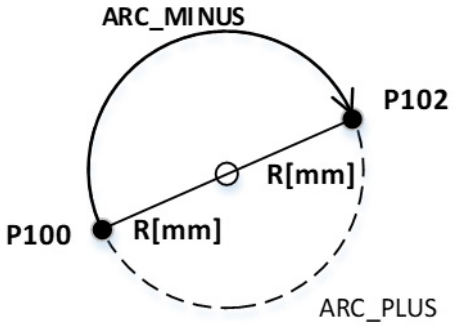
Arc範例 (指定半徑)

如下所述為用於描繪如下軌跡的程式之範例。
從P100開始動作，半徑為 R[mm] (下述範例中為500.0mm)，到達P102的圓弧動作軌跡。

```
Function ArcTest
   Go P100
   Arc P102, 500.0, ARC_SHORT, ARC_MINUS
Fend
```



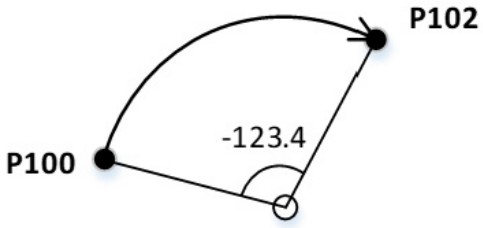
半徑R[mm]等於現在地P100到目標坐標P102間距離的1/2時，會如下動作。半徑R[mm]還更短時，會因為無法描繪圓弧而發生錯誤。



Arc範例 (指定旋轉角)

如下所述為用於描繪如下軌跡的程式之範例。
從P100開始動作，只旋轉指定的旋轉角 (單位：度)，到達P102的圓弧動作軌跡。旋轉角為負值時，如以下範例般，相對於基礎坐標系的XY平面，朝Z軸方向的順時針旋轉。

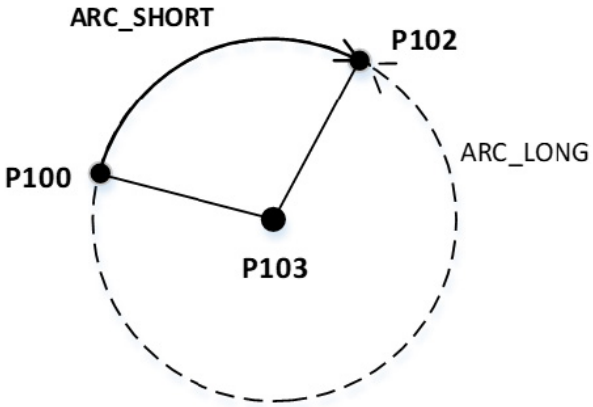
```
Function ArcTest
   Go P100
   Arc P102, -123.4
Fend
```



Arc, Arc3範例 (指定中心坐標)

如下所述為用於描繪如下軌跡的程式之範例。
以P103為中心坐標，從P100開始動作，到達P102的圓弧動作軌跡。

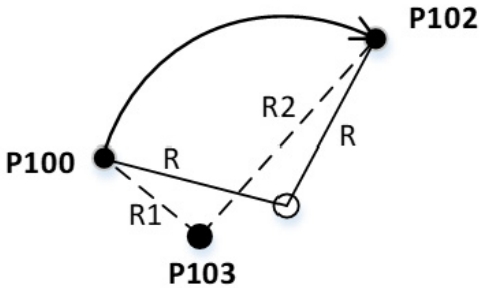
```
Function ArcTest
   Go P100
   Arc P102, P103, ARC_SHORT
Fend
```



提示

- 中心坐標偏離時

  中心坐標P103與現在地P100間的距離R1，與中心坐標P103與目標坐標P102間的距離R2為不同位置，以此指定中心坐標P103時，不會以P103而是以現在地與目標坐標的等距離位置為中心坐標進行圓弧動作。此種情形，可能無法形成預期的軌跡，因此請以現在地與目標坐標的等距離來指定中心坐標。



- 以指定中心坐標的Arc命令指定±180度左右時

  在Arc命令上以圓弧角度近±180度指定中心坐標P103時，指定ARC_LONG的話，相對於基礎坐標系的XY平面，朝Z軸方向的順時針旋轉。指定ARC_SHORT的話，相對於基礎坐標系的XY平面，朝Z軸方向的逆時針旋轉。
  但，圓弧角度如果稍微偏離±180度的話，會沿著ARC_LONG或ARC_SHORT指定的長度路徑旋轉。若要描繪出±180度左右預想的圓弧，請利用經由坐標指定。

3.5.30 Arch

用於設定和顯示Jump、Jump3、Jump3CP命令下的Arch參數。

格式

(1) Arch Arch編號、閃避距離、接近距離

(2) Arch Arch編號

(3) Arch

參數

Arch編號

以0~6的整數指定Arch編號。有效值為0~6的整數。如下頁Arch表格所示，共有7個有效值。

閃避距離

以Jump命令指定水平動作前的閃避距離(從出發點開始的垂直距離)。(單位：mm)以Jump3、Jump3CP命令指定Span動作前的閃避距離。(單位：mm)

接近距離

以Jump命令指定完全結束水平移動階段的接近距離(從目的點開始的垂直距離)。(單位：mm)

以Jump3、Jump3CP命令指定完全結束Span動作階段的接近距離。(單位：mm)

結果

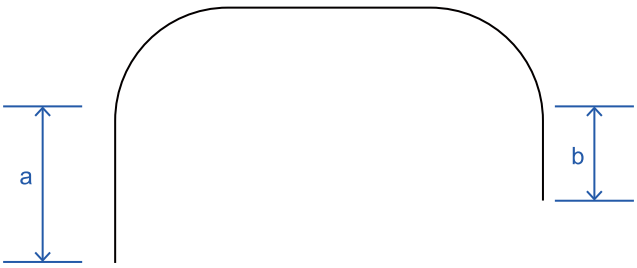
一旦省略所有參數，則顯示Arch表格的所有內容。

若僅指定Arch編號，則顯示指定Arch編號的Arch表格內容。

說明

以Arch命令定義Jump動作命令所需Arch表格的值。使用作為Jump修飾詞的、與Arch編號相應的參數來執行Arch動作。(請先詳閱Jump陳述式，以理解Arch命令。)

若依照Arch設定使用Jump C[Arch編號]，則可將Z方向的角修成圓角。(請參閱範例)在Arch表格中設定：在開始朝水平方向移動前的垂直方向移動距離(閃避距離)以及結束水平方向移動後的到達目標坐標之前的垂直方向移動距離(接近距離)。(請參閱下圖。)



符號	說明
a	閃避距離
b	接近距離

使用者定義的Arch表格值為0~6的整數，共有7個。第8個設定值(Arch 7)為預設值。實際上設定的是門形動作(參照次圖)，而不是拱形動作。若使用預設的Arch值(第8個設定值)執行Jump命令，手臂則會如下圖所示進行動作。

1. 首先，唯有第3關節會移動到以LimZ命令設定的Z坐標值(最大Z值)位置。
2. 接著，手臂水平移動至目標坐標，最後來到X、Y、U位置。
3. 最後，唯有第3關節進行動作，使手臂下降到最後第3關節坐標位置(Z坐標值)，以完成Jump命令動作。

門形動作 (Arch 7的Jump動作)

Arch表格 預設值

Arch編號	閃避距離	接近距離
0	30	30
1	40	40
2	50	50
3	60	60
4	70	70
5	80	80
6	90	90

注意

- 成為門形動作等其它情況時

  若在垂直上升距離、垂直下降距離上設定大於實際垂直移動距離的值，則變成門形動作，而不是拱形動作。

- 儲存Arch值

  只要未變更使用者，就繼續保持Arch表格的數值。

使用Arch時的重要事項

由於透過軌跡控制進行拱形動作的合成，因此不能保證實際軌跡。軌跡會因動作速度和手臂的動作方式而異。請以作業時使用的實際速度和姿態確認實際軌跡。

- 在相同位置上，即便執行帶有相同C[Arch編號]的Jump命令(或Jump3命令)，低速時的軌跡也會比高速動作時低。因此，請注意：即便確認不會因高速而撞擊干擾物，仍有可能在低速動作時發生撞擊。
- 相較於低速動作，高速動作會增加未合成的閃避移動量，呈現出減小未合成的接近移動量之傾向。未出現預期的移動距離時，請降低速度或減速度或將接近距離設得長一些。
- 即便是相同距離的動作，軌跡也會依手臂的動作方式而異。雖然會依手臂的動作方式，而有各種軌跡變化，但如果以一般水平多關節型機器人為例的話，越是大幅移動第1手臂，越會增加垂直上升量，從而呈現出減小垂直下降量的傾向。未出現預期的垂直下降距離時，請降低速度或減速度，或將下降距離設得長一些。

參照

Jump、Jump3、Jump3CP

Arch範例

以下是在命令視窗中執行Arch值設定的範例。

```
> arch 0, 15, 15
> arch 1, 25, 50
> jump p1 c1
> arch
  arch0 =    15.000        15.000
  arch1 =    25.000        50.000
```

```
  arch2 =    50.000        50.000
  arch3 =    60.000        60.000
  arch4 =    70.000        70.000
  arch5 =    80.000        80.000
  arch6 =    90.000        90.000
>
```

3.5.31 Arch函數

用於傳回Arch的設定。

格式

Arch (Arch編號, 參數編號)

參數

Arch編號

指定0~6的整數。

參數編號

- 1：閃避距離
- 2：接近距離

傳回值

用於傳回以參數編號指定的距離。

參照

Arch

Arch函數範例

```
Double archValues(6, 1)
Integer i

'儲存目前Arch的設定
For i = 0 to 6
    archValues(i, 0) = Arch(i, 1)
    archValues(i, 1) = Arch(i, 2)
Next i
```

3.5.32 AreaCorrection函數

本函數用於回傳以補償區域進行補償的點

格式

AreaCorrection(點指定, 區域編號)

參數

點指定

指定作為補償對象的點數據。

區域編號

以運算式或數值指定區域編號(1～8的整數)。

說明

依據事前定義的補償區域，回傳補償結果的點。本座標以與補償前的點相同之本地座標系進行定義。連同動作命令(Go或Jump命令等)一起使用此函數，即可將機器人移動至指定位置。使用本命令將會提升指定點的位置精度。進行點指定時，請輸入在繪圖上的位置。

補償僅適用於位置。附加軸、UVW座標值、姿態旗標均不適用補償。將會直接輸出您輸入的點資料值。

若指定未設定的補償區域，將會發生錯誤。

注意

- 完成教導的點

  請勿對完成教導的點資料適用AreaCorrection函數。將會補償已完成精準調整的教導位置，導致位置偏離。

- 距離補償區域較遠時

  若距離AreaCorrectionSet設定的補償區域較遠，則補償效果將會下降。設定補償區域時，請使基準點包圍動作點。

  校正種類選擇平面時，若為在垂直方向上與選為校正區域的平面存在距離的點，其校正效果將會下降。請將補償區域設定在適當的高度，或在可設置基準點時，將補償種類指定為空間。

- 與設定補償區域的姿態旗標不同時

  若在AreaCorrectionSet設定的基準點與姿態旗標不同，將會發生錯誤。

- 與設定補償區域的姿態(U, V, W)不同時

  若為SCARA機器人(包括RS系列)，將可進行補償。

  若為垂直6軸型機器人(包括N系列)，則補償前的點之工具坐標系Z軸與補償區域基準點的工具坐標系Z軸一致時，將可進行補償。若不一致則為非補償對象，將會發生錯誤。在DiffToolOrientation函數的軸編號指定COORD_Z_PLUS，即可取得工具坐標系Z軸的角度。

參照

AreaCorrectionSet, AreaCorrectionClr, AreaCorrectionDef函數, AreaCorrectionInv, AreaCorrectionOffset函數, DiffToolOrientation函數

AreaCorrection範例

```
Function sample
    ' P(1:4) 基準點
    P1 = XY(-100, 200, -20, 0)
    P2 = XY(100, 200, -20, 0)
```

```
    P3 = XY(-100, 400, -20, 0)
     P4 = XY(100, 400, -20, 0)
    ' P(11:14) 實際上將使用對P(1:4)進行教導後的點
    P11 = XY(-100, 200.5, -20, 0)
    P12 = XY(100.3, 200.1, -20, 0)
    P13 = XY(-100.4, 400.8, -20, 0)
    P14 = XY(100.2, 400.4, -20, 0)
    ' 設定補償區域
    AreaCorrectionSet 1, P(1:4), P(11:14), MODE_PLANE
    P999 = AreaCorrection(P1, 1)  ' P999為完成補償的點
    Print Dist(P11, P999)
    P999 = AreaCorrection(XY(0, 300, -20, 0), 1) ' 補償位於區域內的點
    Print P999
Fend
```

[輸出結果]

```
0
 X:    0.100 Y:  300.450 Z:  -20.000 U:    0.000 /R /0
```

3.5.33 AreaCorrectionClr

用於清除補償區域。

格式

AreaCorrectionClr 區域編號

參數

區域編號

以運算式或數值指定區域編號(1～8的整數)。

結果

將清除與區域編號對應的補償區域。

無法在機器人動作中執行本命令。請在停止狀態下使用。

參照

AreaCorrectionSet, AreaCorrectionDef函數, AreaCorrectionInv, AreaCorrectionOffset函數

AreaCorrectionClr範例

```
AreaCorrectionClr 1
```

3.5.34 AreaCorrectionDef函數

用於回傳是否已設定為指定的補償區域。

格式

AreaCorrectionDef(區域編號)

參數

區域編號

以運算式或數值指定區域編號(1～8的整數)。

傳回值

若已設定補償區域將回傳「True」，其他結果則回傳「False」。

參照

AreaCorrectionSet, AreaCorrectionClr, AreaCorrectionInv, AreaCorrectionOffset函數

AreaCorrectionDef函數範例

```
Function DisplayAreaCorrectionDef(areaNum As Integer)

   If AreaCorrectionDef(areaNum) = False Then
      Print "Area", areaNum, " is not defined"
   Else
      Print "Area Definition:"
      AreaCorrectionSet areaNum
   EndIf
Fend
```

3.5.35 AreaCorrectionInv函數

本函數用於復原完成補償的點

格式

AreaCorrectionInv(點指定, 區域編號)

參數

點指定

　　指定作為補償對象的點數據。

區域編號

　　以運算式或數值指定區域編號(1～8的整數)。

說明

針對完成補償的點，以AreaCorrection函數回傳補償前的點資料。

對實際進行教導而建立的點或完成補償的點適用AreaCorrectionInv，將可取得補償前的點資料。

若指定未設定的補償區域，將會發生錯誤。

參照

AreaCorrectionSet, AreaCorrectionClr, AreaCorrectionDef函數, AreaCorrectionOffset函數

AreaCorrectionInv範例

```
Function AreaCorrectionTest
    '  P(1:4) 基準點
    P1 = XY(-100, 200, -20, 0)
    P2 = XY(100, 200, -20, 0)
    P3 = XY(-100, 400, -20, 0)
    P4 = XY(100, 400, -20, 0)
    '  P(11:14) 實際上將使用對P(1:4)進行教導後的點
    P11 = XY(-100, 200.5, -20, 0)
    P12 = XY(100.3, 200.1, -20, 0)
    P13 = XY(-100.4, 400.8, -20, 0)
    P14 = XY(100.2, 400.4, -20, 0)
    '  設定補償區域
    AreaCorrectionSet 1, P(1:4), P(11:14), MODE_PLANE
    P888 = AreaCorrection(P1, 1)' P888為完成補償的點
    P999 = AreaCorrectionInv(P888, 1)' P999為轉換前的點
    Print Dist(P11, P888)
    Print Dist(P1, P999)
Fend
```

[輸出結果]

```
0
0
```

3.5.36 AreaCorrectionOffset函數

本函數用於回傳從完成補償的點進行相對移動的點

格式

AreaCorrectionOffset(點指定, 相對移動量指定, 區域編號[, 選擇相對關係])

參數

點指定

指定作為相對移動之基準位置的點數據。

相對移動量指定

以點數據指定相對移動量。

區域編號

以運算式或數值指定區域編號(1～8的整數)。

選擇相對關係

表示進行相對移動時，是以哪一個坐標系為基準。省略時，將為本地坐標系基準。

以本地坐標系基準進行動作時，將為以定義點指定的坐標系為基準，進行相對移動後的坐標。以工具坐標系基準進行動作時，將為以點指定位置為基準，進行相對移動後的坐標。

選擇相對關係	常數	值
本地坐標基準	AC_LOCAL	0
工具坐標基準	AC_TOOL	1

說明

針對完成補償的點，以AreaCorrection函數回傳進行相對移動後的點。將為以與指定點相同的本地坐標系進行定義的坐標。連同動作命令(Go或Jump命令等)同時使用此函數，即可將機器人移動至指定位置。

與Here函數併用，即可執行與BGo、TGo相同的動作。若位於補償區域的範圍內，則相對移動量將更為正確。

若指定未設定的補償區域，將會發生錯誤。

注意

- 進行姿態的相對移動後

  進行姿態的相對移動後，相對移動後的姿態為非補償對象，可能會發生錯誤。

參照

AreaCorrectionSet, AreaCorrectionClr, AreaCorrectionDef函數, AreaCorrectionInv, Here

AreaCorrectionOffset範例

```
' 補償區域1已完成定義
' 與BGo XY(50, 0, 0, 0) 相同
Go AreaCorrectionOffset(Here, XY(50, 0, 0, 0), 1)
' 與TGo XY(50, 0, 0, 0) 相同
Go AreaCorrectionOffset(Here, XY(50, 0, 0, 0), 1, AC_TOOL)
```

3.5.37 AreaCorrectionSet

用於設定與顯示補償區域。

格式

(1) AreaCorrectionSet 區域編號, 基準連續點, 教導連續點, 補償種類
(2) AreaCorrectionSet 區域編號
(3) AreaCorrectionSet

參數

區域編號
　以運算式或數值指定區域編號(1～8的整數)。
基準連續點
　以冒號連接欲設為基準點的點數據連號，及起點與終點的2個點編號進行指定(例：P(1:4))。
　為將補償效果發揮至極限，選擇基準點時請使其包圍欲補償點。
教導連續點
　以冒號連接對應於基準連續點的已教導點數據，及起點與終點的2個點編號進行指定(例：P(1:4))。請對應基準連續點設定點資料的排列順序。
補償種類
　為表示補償種類的整數值。
　在平面校正中，可校正以選為參考點的點所構成之平面上的點。若選擇平面校正，請在平面上配置參考一系列點。參考點數量最少須有3點。
　在空間校正中，可校正以選為參考點的點所構成之立體空間上的點。若選擇立體校正，選擇時請使參考一系列點包圍欲校正區域。參考點數量最少須有4點。

補償種類	常數	值
平面	MODE_PLANE	2
空間	MODE_SPACE	3

結果

- 若以(1)的格式進行指定，將會以指定的區域編號設定補償區域。
- 若以(2)的格式進行指定，將會顯示指定區域編號的內容。
- 若以(3)的格式進行指定，將會顯示已定義補償區域的所有內容。

說明

用於設定在區域補償功能中使用的補償區域。設定補償區域並使用AreaCorrection函數、AreaCorrectionInv函數、AreaCorrectionOffset函數，即可提升位於補償區域內的點之位置精度。

在動作中將無法執行本命令。請在停止狀態下使用。

關於基準位置的選擇方法，請參閱下列手冊。
「Epson RC+ 使用者指南 - 區域補償功能」

注意

- 關於補償區域資料

  在關閉控制器電源前，補償區域均為啟用。啟動控制器時，補償區域將為未定義狀態。

- 關於工具

  以AreaCorrection函數進行補償時，請使用與教導補償區域基準點時相同的工具。若使用不同工具，可能會導致補償效果下降。

- 距離補償區域較遠時

  若距離AreaCorrectionSet設定的補償區域較遠，則補償效果將會下降。設定補償區域時，請使基準點包圍動作點。

  校正種類選擇平面時，若為在垂直方向上與選為校正區域的平面存在距離的點，其校正效果將會下降。請將校正區域設定在適當的高度，或在可於高度方向設置參考點時，將校正種類指定為空間。

- 與設定補償區域的姿態旗標不同時

  若在AreaCorrectionSet設定的基準點與姿態旗標不同，將會發生錯誤。請將與執行動作的點相同之姿態旗標設為基準點。

- 與設定補償區域的姿態(U,V,W)不同時

  若為SCARA機器人(包括RS系列)，將可進行補償。

  若為垂直6軸型機器人(包括N系列)，則補償前的點之工具坐標系Z軸與補償區域基準點的工具坐標系Z軸一致時，將可進行補償。若不一致則為非補償對象，將會發生錯誤。在DiffToolOrientation函數的軸編號指定COORD_Z_PLUS，即可取得工具坐標系Z軸的角度。

參照

AreaCorrectionClr, AreaCorrectionDef函數, AreaCorrectionInv, AreaCorrectionOffset函數, DiffToolOrientation函數

AreaCorrectionSet範例

以下為範例。P11～P14請使用已教導的點。

如下所示，若將P1～P4作為基準點在繪圖上的位置，則補償區域將為以P1、P2、P3、P4為頂點，且寬度為200mm的正方形。

```
Function AreaCorrectionTest
  ' P(1:4) 基準點
  P1 = XY(-100, 200, -20, 0)
  P2 = XY(100, 200, -20, 0)
  P3 = XY(-100, 400, -20, 0)
  P4 = XY(100, 400, -20, 0)
  ' P(11:14) 實際上將使用對P(1:4)進行教導後的點
  P11 = XY(-100, 200.5, -20, 0)
  P12 = XY(100.3, 200.1, -20, 0)
  P13 = XY(-100.4, 400.8, -20, 0)
  P14 = XY(100.2, 400.4, -20, 0)
  ' 設定補償區域
  AreaCorrectionSet 1, P(1:4), P(11:14), MODE_PLANE
Fend
```

3.5.38 Arm

用於選擇手臂和顯示目前選擇的手臂編號。

格式

(1) Arm 手臂編號

(2) Arm

參數

手臂編號

以整數值或運算式來指定。有效值範圍是0~15，最多可選擇16個不同手臂。手臂0為標準(預設)機器人手臂。手臂1~15是依ArmSet定義的增設型手臂。可省略。省略時即顯示目前手臂編號。

結果

若在無參數設定的狀態下執行Arm命令，則顯示目前選擇的手臂編號。

說明

指定用於執行機器人命令的手臂。可以Arm在增設型手臂上共享位置資料。沒有增設型手臂設定時，標準手臂(手臂編號0)則會動作。出廠時將手臂編號設為「0」，因此，沒有增設型手臂時，無需變更設定。若要使用增設型手臂，請以ArmSet進行最初的設定。

即便實際的機器人構成不同於標準構成，增設型手臂也支援符合各機器人的適當參數設定。在以下條件下，增設型手臂會正確地進行動作。

- 透過設定讓2個以上的手臂通過1個點資料。
- 使用Pallet。
- 執行CP動作。
- 指定相對位置。
- 使用本地坐標。

若為使用旋轉關節的水平多關節型機器人(包含RS系列)，則以使用ArmSet參數設定的值為基準，對關節角度進行運算。因此，在需設定增設型手臂或抓手的情況下，請使用此命令。

注意

- Arm 0

  不可利用ArmSet設定和變更Arm 0。Arm 0用於設定標準機器人的構成。一旦將Arm值設為「0」，則透過設定使用機器人手臂的標準參數。

- 使用手臂長度補償選配時

  若將ArmCalib設為On，Arm 0便會適用手臂長度補償值，並自動切換為Arm 0。進行適用手臂長度補償值的動作時，請使用Arm 0。即便將ArmCalib設為On，除Arm 0以外，也不會變為適用手臂長度補償值的動作。

  使用機器人手臂的標準參數時，請將ArmCalib設為Off，並使用Arm 0。

- 未設定的手臂編號

  若選擇未以ArmSet定義的增設型手臂編號，則發生錯誤。

參照

ArmClr、ArmSet、ECPSet、TLSet、ArmCalibSet

Arm範例

在以下程式範例中，使用ArmSet和Arm設定增設型手臂。以ArmSet定義增設型手臂，並以Arm設定選擇將哪一個手臂作為目前手臂。(Arm 0屬於預設的機器人手臂，使用者不可進行變更。)

命令視窗中的操作範例

```
> ArmSet 1, 300, -12, -30, 300, 0
> ArmSet
  arm0 250  0  0  300  0
  arm1 300  -12  -30  300  0

> Arm 0
> Jump P1     '透過設定標準手臂，Jump到P1
> Arm 1
> Jump P1     '透過設定增設型手臂1，Jump到P1
```

3.5.39 Arm函數

用於傳回目前選擇的機器人手臂編號。

格式

Arm

傳回值

用於傳回目前手臂編號。

參照

Arm

Arm函數範例

```
Print "The current arm number is: ", Arm
```

3.5.40 ArmCalib

用於啟用或停用目前選擇的機器人的手臂長度補償。

格式

ArmCalib On | Off

參數

On | Off

將手臂長度補償設為啟用狀態時指定On；設為停用狀態時則指定Off。

說明

ArmCalib On命令會啟用手臂長度補償。執行ArmCalib On時，於ArmCalibSet指定的補償值將設定於Arm 0，且Arm將切換至0。

ArmCalib Off命令將會停用手臂長度補償。執行ArmCalib Off時，會將標準參數設定於Arm 0。執行ArmCalib Off時，Arm不會自動切換。

若已購買手臂長度補償選項，則於出廠時已設為啟用狀態。

於手臂長度補償啟用狀態下將Arm設為「0」以外時，將以所設定的Arm編號進行動作。

注意

- 僅限於裝有手臂長度補償選項時，方可使用此命令。
- 請勿將ArmCalib On狀態的備份檔案還原至停用手臂長度補償選項的控制器。

  若將ArmCalib On狀態的備份檔案還原至停用手臂長度補償選項的控制器，將自動變成ArmCalib Off狀態。還原至其他控制器時，敬請注意。

參照

ArmCalibClr, ArmCalibSet, ArmCalibDef

ArmCalib範例

在命令視窗中執行如下範例。

```
> ArmCalib On

> ArmCalib Off
```

3.5.41 ArmCalib函數

用於傳回目前選擇的機器人之手臂長度補償狀態。

格式

ArmCalib

傳回值

- 0 = 手臂長度補償停用
- 1 = 手臂長度補償啟用

參照

ArmCalib

ArmCalib函數範例

```
If ArmCalib = Off then
  Print "Arm length calibration disabled."
Else
  Print "Arm length calibration enabled."
Endif
```

3.5.42 ArmCalibClr

用於清除手臂長度補償設定。

格式

ArmCalibClr

注意

- 若非必要，切勿使用ArmCalibClr。

  ArmCalibClr會清除設定於ArmCalibSet的手臂長度補償參數。出廠時已精密設定ArmCalibSet。(購買手臂長度補償選配時)若誤將設定值清除，則必須再度回廠進行精密測量。若非必要，切勿使用ArmCalibClr。

參照

ArmCalib, ArmCalibSet

ArmCalibClr 範例

```
ArmCalibClr
```

3.5.43 ArmCalibDef

用於傳回手臂長度補償的設定狀態。

格式

ArmCalibDef

傳回值

如手臂長度補償已設定便傳回「True」，未設定則傳回「False」。

參照

ArmCalib, ArmCalibClr, ArmCalibSet

ArmCalibDef範例

```
Function DisplayArmCalibDef
  Integer i

   If ArmCalibDef = False Then
      Print "ArmCalib is not defined"
   Else
      Print "ArmCalib Definition:"
      For i = 1 to 3
        Print ArmCalibSet(i)
      Next i
   EndIf
Fend
```

3.5.44 ArmCalibSet

設定和顯示手臂長度與關節位移。

格式

(1) ArmCalibSet 設定值1, 設定值2, 設定值3

(2) ArmCalibSet

參數

設定值	水平多關節型
1	從第1關節至第2關節的水平距離 (mm)
2	從第2關節至姿態中心的水平距離 (mm)
3	第2關節的位移角度 (°)

結果

若在省略所有參數的狀態下執行ArmCalibSet，則顯示手臂長度補償編號的參數。

說明

ArmCalibSet用於設定手臂長度及關節位移的相關參數。各個體的手臂長度及關節位移已於出廠時進行精密測量，若以本命令進行設定，將可提升距離精度。

注意

- 若非必要，切勿變更ArmCalibSet

  出廠時已精密設定ArmCalibSet。(購買Arm Length Calibration選項時)若誤變更設定值，會導致距離精度及軌跡精度的下降。若非必要，切勿變更ArmCalibSet。

參照

ArmCalib, ArmCalibClr, ArmCalibDef

ArmCalibSet 範例

在命令視窗中執行如下範例。

```
> ArmCalibSet 299.989, 250.001, 0.012
```

3.5.45 ArmCalibSet函數

用於傳回手臂長度補償的1個設定值。

格式

ArmCalibSet(設定值編號)

參數

設定值編號

以運算式或數值指定要參照的設定值編號(0~3的整數)。(請參閱下列資料。)

水平多關節型機器人

設定值編號	傳回值
1	從第1關節至第2關節的水平距離 (mm)
2	從第2關節至姿態中心的水平距離 (mm)
3	第2關節的位移角度 (°)

傳回值

以實數值傳回在上表中指定的任一參數之設定值。

參照

ArmCalibClr, ArmCalibSet

ArmCalibSet函數範例

```
Double L1, L2, Angle

L1 = ArmCalibSet(1)
L2 = ArmCalibSet(2)
Angle = ArmCalibSet(3)
```

3.5.46 ArmClr

用於清除手臂設定。

格式

ArmClr 手臂編號

參數

手臂編號

以整數和運算式指定在15個手臂當中要清除設定的手臂編號。
(手臂0為預設手臂，不可清除。)

參照

Arm、ArmSet、ECPSet、Tool、Local、LocalClr、TLSet

ArmClr範例

```
ArmClr 1
```

3.5.47 ArmDef函數

用於傳回手臂的設定狀態。

格式

ArmDef (手臂編號)

參數

手臂編號

以整數值指定要傳回狀態的手臂編號。

傳回值

若已設定指定的手臂，則傳回「True」；若未設定，則傳回「False」。

參照

Arm、ArmClr、ArmSet、ECPSet、Local、LocalClr、Tool、TLClr、TLSet

ArmDef範例

```
Function DisplayArmDef(armNum As Integer)

   Integer i

   If ArmDef(armNum) = False Then
      Print "Arm ", ArmNum, "is not defined"
   Else
      Print "Arm ", armNum, " Definition:"
      For i = 1 to 5
        Print ArmSet(armNum, i)
      Next i
   EndIf
Fend
```

3.5.48 ArmSet

設定和顯示增設型手臂。

格式

(1) ArmSet 手臂編號, 設定值1, 設定值2, 設定值3 [, 設定值4] [, 設定值5]: 水平多關節型(包括RS系列)
(2) ArmSet 手臂編號, 設定值1, 設定值2, 設定值3, 設定值4, 設定值5, 設定值6, 設定值7, 設定值8, 設定值9, 設定值10, 設定值11, 設定值12: 垂直多關節型(包括N系列)
(3) ArmSet 手臂編號
(4) ArmSet

參數

手臂編號

以運算式或數值指定1~15的整數。最多可設定15個增設型手臂。

設定值編號	水平多關節型(包含RS系列)	垂直多關節型 (包括N系列)
1	從第2關節至姿態中心的水平距離 (mm)	從基點到第2關節的垂直距離(mm)
2	第2關節的位移角度(度)	從第1關節到第2關節的水平距離(mm)
3	作業尖端高度方向的位移量(mm)	從第2關節到第3關節的距離(mm)
4	從第1關節至第2關節的水平距離 (mm)	從第3關節到第5關節的垂直距離(mm)
5	第4關節角度的位移角度(度)	從第3關節到第5關節的水平距離(mm)
6	-	從第5關節到姿態中心的距離(mm)
7	-	第1關節角度的偏移(度)
8	-	第2關節角度的偏移(度)
9	-	第3關節角度的偏移(度)
10	-	第4關節角度的偏移(度)
11	-	第5關節角度的偏移(度)
12	-	第6關節角度的偏移(度)

結果

若在省略所有參數的狀態下執行ArmSet，則顯示已設定的所有增設型手臂編號和參數。

若僅指定手臂編號，即顯示指定的手臂編號和參數。

說明

若在標準設定上增設手臂，則設定該增設型手臂的相關參數。這在機器人上安裝增設型手臂或增設型抓手時十分方便。使用增設型手臂時，則以Arm命令指定作業手臂。

可省略設定值4的第1關節~第2關節的水平距離、設定值5的姿態關節角度的位移角度，一旦省略，即與標準手臂的設定相同。

欲以非標準構成使用機器人時，可透過增設型手臂的設定功能設定相應參數。例如，已在第2機器人上安裝第2旋轉關節時，務必設定新構成的機器人手臂。執行此項設定，可讓增設型手臂進行適當的動作，但務必符合以下條件。

- 數個手臂使用相同資料點
- 使用Pallet

- 執行CP動作
- 指定相對位置
- 使用本地坐標系

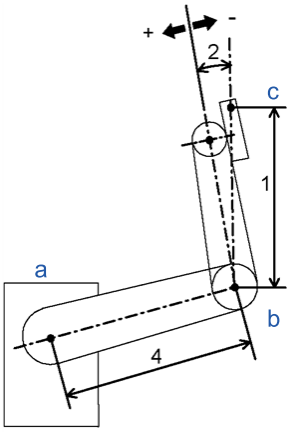
就帶有旋轉關節的水平多關節型(包含RS系列)而言，由於基於ArmSet的參數來運算關節角度，因此如需設定增設型手臂或抓手，則這項設定顯得格外重要。

注意

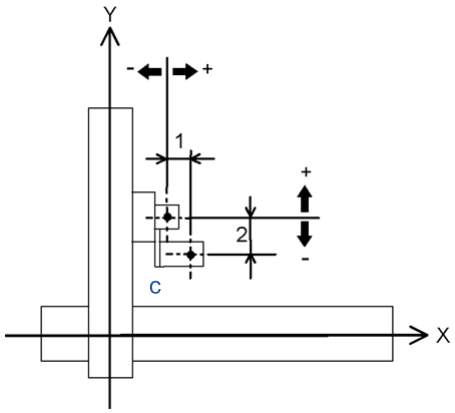
- Arm 0

  手臂編號0用於機器人標準構成時的設定，使用者不可定義或變更。一旦設定手臂編號0，則使用機器人手臂的標準參數。

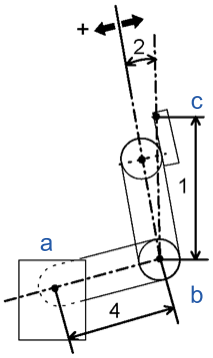
- 水平多關節型機器人



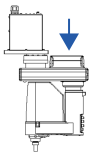
- 直角坐標型機器人



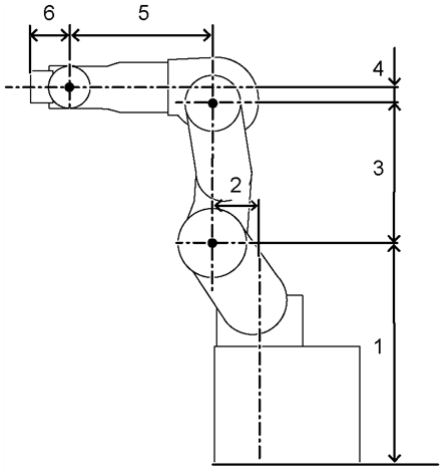
- 水平多關節型機器人(包含RS系列)

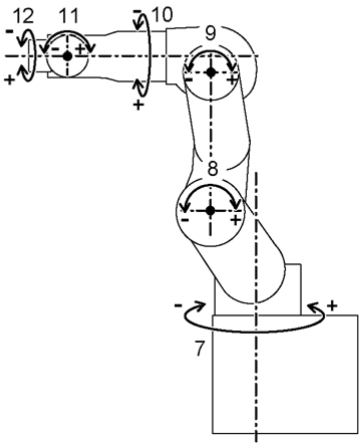


RS系列的插圖是從箭頭方向繪製的。



- 垂直多關節型機器人





符號	說明
a	第1關節
b	第2關節
c	增設型手臂

參照

Arm、ArmClr

ArmSet範例

如下所述為使用ArmSet和Arm的增設型手臂之設定範例。以ArmSet設定增設型手臂，並以Arm設定要選擇的手臂。(手臂編號0為機器人的預設手臂，使用者不可變更。)

命令視窗中的操作範例

```
> ArmSet 1, 300, -12, -30, 300, 0
> ArmSet
  Arm 0: 125.000, 0.000, 0.000, 225.000, 0.000
  Arm 1: 300.000, -12.000, -30.000, 300.000, 0.000

> Arm 0
```

```
> Jump P1     '透過設定標準手臂，Jump到P1
> Arm 1
> Jump P1     '透過設定增設型手臂1，Jump到P1
```

3.5.49 ArmSet函數

用於傳回增設型手臂的1個設定值。

格式

ArmSet (手臂編號, 設定值編號)

參數

手臂編號

以運算式或數值指定要參照的手臂編號。

設定值編號

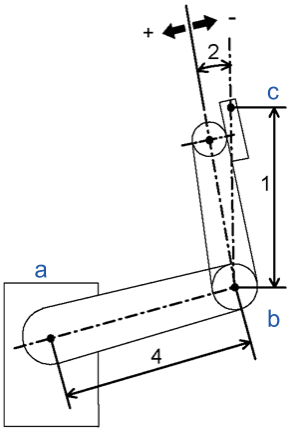
以運算式或數值指定要參照的設定值編號(0~5的整數)。(請參閱如下內容。)

設定值編號	水平多關節型(包含RS系列)	垂直多關節型(包括N系列)
1	從第2關節至姿態中心的水平距離 (mm)	從基點到第2關節的垂直距離(mm)
2	第2關節的位移角度(度)	從第1關節到第2關節的水平距離(mm)
3	作業尖端高度方向的位移量(mm)	從第2關節到第3關節的距離(mm)
4	從第1關節至第2關節的水平距離 (mm)	從第3關節到第5關節的垂直距離(mm)
5	第4關節角度的位移角度(度)	從第3關節到第5關節的水平距離(mm)
6	-	從第5關節到姿態中心的距離(mm)
7	-	第1關節角度的偏移(度)
8	-	第2關節角度的偏移(度)
9	-	第3關節角度的偏移(度)
10	-	第4關節角度的偏移(度)
11	-	第5關節角度的偏移(度)
12	-	第6關節角度的偏移(度)

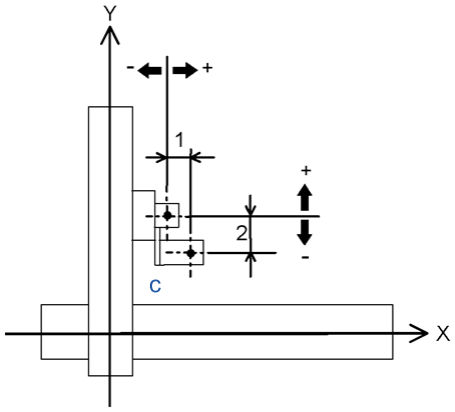
傳回值

以實數值傳回在上表中指定的任一參數之設定值。

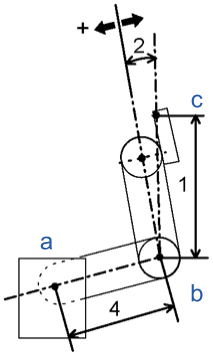
- 水平多關節型機器人



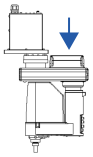
- 直角坐標型機器人



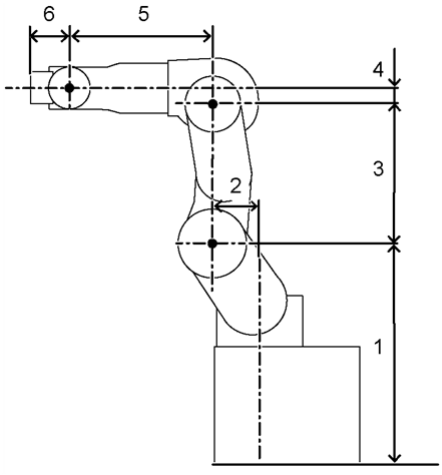
- 水平多關節型機器人(包含RS系列)

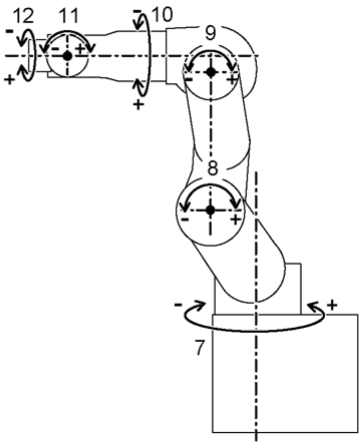


RS系列的插圖是從箭頭方向繪製的。



- 垂直多關節型機器人





符號	說明
a	第1關節
b	第2關節
c	增設型手臂

參照

ArmClr、ArmSet

ArmSet函數範例

```
Real x

x = ArmSet(1, 1)
```

3.5.50 Asc函數

用於對字串開頭的第1個字元傳回ASCII代碼。(以十進位傳回字元碼。)

格式
Asc (取代字串)

參數

取代字串
　　以字串運算式或字串指定1個字元以上的字串。

傳回值
用於對傳送到Asc函數的字串之第1個字元傳回整數值。

說明
Asc函數用於以十進位傳回字串開頭的1個字元之字元碼。傳送到Asc函數的字串，可以是常數，也可以是變數。

注意

- 只對第1個字元傳回ASCII值

  在Asc函數中，雖可使用1個字元以上長度的字串，但實際上只使用其中的第1個字元。Asc用於傳回字串第1個字元的ASCII值。

參照
Chr$、InStr、Left$、Len、Mid$、Right$、Space$、Str$、Val

Asc函數範例
在此範例中，透過程式和命令視窗的操作，使用如下所述的Asc函數。

```
Function asctest
    Integer a, b, c
    a = Asc("a")
    b = Asc("b")
    c = Asc("c")
    Print "The ASCII value of a is ", a
    Print "The ASCII value of b is ", b
    Print "The ASCII value of c is ", c
Fend
```

命令視窗中的操作範例

```
>print asc("a")
97
>print asc("b")
98
>
```

3.5.51 Asin函數

用於傳回指定數值的反正弦。

格式

Asin (數值)

參數

數值

以實數值指定角度的正弦。

傳回值

以實數值(單位：弧度)傳回指定數值的反正弦值。

說明

Asin用於傳回指定數值的反正弦。指定數值範圍為-1~1。傳回值範圍：$-\pi/2 \sim \pi/2$。數值小於-1或大於1時，會發生錯誤。

使用RadToDeg函數，以便將弧度值轉換為度。

參照

Abs、Acos、Atan、Atan2、Cos、DegToRad、RadToDeg、Sgn、Sin、Tan、Val

Asin函數範例

```
Function asintest
  Double x

  x = Sin(DegToRad(45))
  Print "Asin of ", x, " is ", Asin(x)
Fend
```

3.5.52 AtHome函數

用於傳回目前機器人姿態是否位於Home位置。

格式

AtHome

傳回值

目前機器人姿態若在Home位置上，則傳回「True」；若不是Home位置，則傳回「False」。

說明

本函數用於傳回目前機器人姿態是否位於Home位置。要註冊Home位置時，使用HomeSet命令或Robot Manager進行操作。使用Home命令移至Home位置。

參照

Home、HomeClr、HomeDef、HomeSet、Hordr、MCalComplete

3.5.53 Atan函數

用於傳回指定數值的反正切。

格式

Atan (數值)

參數

數值

以實數值指定角度的正切。

傳回值

以實數值(單位：弧度)傳回指定數值的反正切值。

說明

Atan用於傳回指定數值的反正切。可指定任何數值。傳回值的範圍：$-\pi \sim \pi$ 弧度。

使用RadToDeg函數，以便將弧度值轉換為度。

參照

Abs、Acos、Asin、Atan2、Cos、DegToRad、RadToDeg、Sgn、Sin、Tan、Val

Atan函數範例

```
Function atantest
  Real x, y
  x = 0
  y = 1
  Print "Atan of ", x, " is ", Atan(x)
  Print "Atan of ", y, " is ", Atan(y)
Fend
```

3.5.54 Atan2函數

是用於傳回連接坐標原點和指定點(X, Y)的直線角度的函數。單位是弧度。

格式

Atan2 (X坐標值, Y坐標值)

參數

X
　　以實數值指定X坐標值。
Y
　　以實數值指定Y坐標值。

傳回值

用於傳回指定坐標的反正切值。範圍為-π ~ π 。

說明

Atan2 (X, Y)是用於傳回連接坐標原點(0, 0)和([x坐標值],[y坐標值])的直線角度的函數。

此三角函數皆以4個四分之一圓周傳回反正切值(角度)。

參照

Abs、Acos、Asin、Atan、Cos、DegToRad、RadToDeg、Sgn、Sin、Tan、Val

Atan2函數範例

```
Function at2test
  Real x, y
  Print "Please enter a number for the X Coordinate:"
  Input x
  Print "Please enter a number for the Y Coordinate:"
  Input y
  Print "Atan2 of ", x, ", ", y, " is ", Atan2(x, y)
Fend
```

3.5.55 ATCLR

用於清除關節的有效扭矩並進行初始化。

格式

ATCLR [關節指定1 [, 關節指定2 [, 關節指定3 [, 關節指定4 [, 關節指定5 [, 關節指定6 [, 關節指定7 [, 關節指定8 [, 關節指定9]]]]]]]]

參數

關節指定1 -關節指定9
以整數值或運算式指定關節編號。未指定參數時，清除所有關節的有效扭矩值。
附加軸的S軸為8，T軸為9。若指定不存在的關節編號，則發生錯誤。

說明

ATCLR用於清除指定關節的有效扭矩值。

執行ATRQ前，請務必執行ATCLR。

參照

ATRQ、PTRQ

ATCLR範例

[例1] 這是清除所有關節的有效扭矩值後，顯示指定關節編號的扭矩值之命令執行範例。

```
> atclr
> go p1
> atrq 1
     0.028
> atrq
     0.028    0.008
     0.029    0.009
     0.000    0.000
>
```

[例2] 這是對於垂直多關節機器人清除J1、J4、J5的有效扭矩值後，顯示已指定關節編號的扭矩值之命令執行範例。

```
> atclr 4, 1, 5
> go p1
> ptrq 1
     0.227
> ptrq 4
     0.083
```

3.5.56 ATRQ

顯示指定關節的有效扭矩值。

格式
ATRQ [關節編號]

參數

關節編號
以整數值或運算式指定關節編號。可省略。附加軸的S軸為8，T軸為9。

結果
顯示所有關節的目前有效扭矩值。

說明
ATRQ用於顯示指定關節的有效扭矩值。可透過ATRQ命令瞭解馬達的負荷狀態。以0~1的實數值表示結果。最大有效扭矩值為「1」。

執行ATRQ前，請務必執行ATCLR。

ATRQ命令有時間限制。執行ATCLR後，請在60秒內執行ATRQ。若超過60秒，則發生錯誤4030「扭矩執行值飽和狀態」。

參照
ATCLR、ATRQ函數、PTRQ

ATRQ範例

```
> atclr
> go p1
> atrq 1
     0.028
> atrq
     0.028    0.008
     0.029    0.009
     0.000    0.000
>
```

3.5.57 ATRQ函數

傳回指定關節的有效扭矩。

格式
ATRQ (關節編號)

參數

關節編號
以整數值或運算式指定關節編號。附加軸的S軸為8，T軸為9。

傳回值
以0~1的實數值傳回。

說明
ATRQ函數用於傳回指定關節的有效扭矩。可透過ATRQ函數瞭解馬達的負荷狀態。以0~1的實數值表示結果。最大有效扭矩為1。

執行ATRQ函數前，請務必執行ATCLR。

ATRQ函數有時間限制。執行ATCLR後，請在60秒內執行ATRQ函數。若超過60秒，則發生錯誤4030。

參照
ATRQ、PTCLR、PTRQ

ATRQ函數範例
是在程式中使用ATRQ函數的範例。

```
Function CheckAvgTorque
  Integer i

  Go P1
  ATCLR
  Go P2
  Print "Average torques:"
  For i = 1 To 4
    Print "Joint ", i, " = ", ATRQ(i)
  Next i
Fend
```

3.5.58 AutoLJM

設定自動LJM。

格式
AutoLJM { On | Off }

參數

On | Off
- On：啟用自動LJM。
- Off：解除自動LJM。

說明
以下列命令啟用AutoLJM。

Arc, Arc3, Go, Jump3, Jump3CP, Move

將AutoLJM設定為On之後，無論傳遞給各動作命令的位置資料是否適用LJM函數，皆與適用LJM函數時一樣，執行關節移動量最小的動作。

例如，為了對Go LJM (P1)獲得相同的效果，可進行如下設定：

```
AutoLJM On
Go P1
  AutoLJM Off
```

AutoLJM可用於在程式特定區間啟用LJM，所以不需變更各項動作命令。

設定AutoLJM Off之後，唯有在傳遞給各動作命令的位置資料上適用LJM函數，方可啟用LJM函數功能。

在以下情況下，AutoLJM處於透過控制器設定所指定的狀態(工廠出廠時：Off)。
- 啟動控制器時
- 執行Reset時
- 中斷所有工作時
- 執行Motor On時
- 切換Auto/Programming作業模式時

注意

- AutoLJM和LJM函數的雙重使用

  在AutoLJM為On時，若在傳遞給動作命令的點資料上使用LJM函數，則會在執行動作時雙重使用LJM。

  對於動作命令Move LJM (P1, Here)和Move LJM (P1)，無論啟用AutoLJM與否，動作都不會改變。然而，對Move LJM (P1, P0) 動作命令啟用AutoLJM時的Move LJM (LJM (P1, P0), Here) 動作和未啟用AutoLJM時的Move LJM (P1, P0) 動作，兩者的動作完成位置有可能不同。

  編程時，建議避免雙重使用AutoLJM和LJM函數。

- AutoLJM使用注意事項

  可透過控制器的環境設定，在啟動控制器時啟用AutoLJM功能。然而，透過控制器的環境設定或命令始終啟用AutoLJM功能時，對於客戶欲大幅度移動關節的動作命令，會自動變更為減小關節移動量的姿態並進行動作。

編程時，建議利用LJM函數或AutoLJM命令，僅在必要時才使用LJM。

參照

AuoLJM函數、LJM函數

AutoLJM範例

```
AutoLJM On
Go P1
Go P2
AutoLJM Off
```

3.5.59 AutoLJM函數

用於傳回AutoLJM的狀態。

格式

AutoLJM

傳回值

- 0 = 停用自動LJM
- 1 = 啟用自動LJM

參照

AutoLJM

AutoLJM函數範例

```
If AutoLJM = Off Then
    Print "AutoLJM is off"
EndIf
```

3.5.60 AutoOrientationFlag

變更N6-A1000**的姿態旗標。

格式

AutoOrientationFlag { On | Off }

參數

On | Off

- On：啟用AutoOrientationFlag。
- Off：解除AutoOrientationFlag。(預設)

說明

以下列命令啟用AutoOrientationFlag。

Go, BGo, TGo, Jump3, JumpTLZ

變更以下姿態的旗標。

對象機種	參數OFF/ON	姿態旗標			備註
		臂腕	臂肘	腕部	
N6-A1000**	OFF	-	-	-	以使用者指定的姿態旗標進行動作。(預設)
	ON	-	○	○*1	不清楚姿態旗標時，請選擇「ON」。

○：將AutoOrientationFlag設為「ON」之後，變更姿態旗標。

*1：唯有變更臂肘姿態旗標時，才會變更腕部姿態旗標。若變更腕部姿態旗標，則變成J4軸移動量最小的姿態旗標。

與LJM函數併用

併用LJM函數時，腕部姿態Flag、J4Flag、J6Flag則用於形成以LJM函數選擇的姿態。例如，將LJM函數的姿態旗標選擇設為「3」時，則選擇J5軸移動量最小的腕部姿態Flag、J4Flag、J6Flag。此外，未併用LJM函數時，則選擇J4軸移動量最小的腕部姿態Flag、J4Flag、J6Flag。

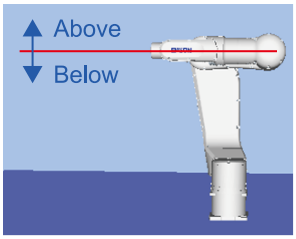
AutoOrientationFlag範例

```
Motor On
Power High
AutoOrientationFlag On

Go P1
Go P2
```

- 將AutoOrientationFlag設為「ON」時

  如下所述為依P點與紅線間的位置關係變更旗標的情況。

- P點位於紅線上方：Above
- P點位於紅線下方：Below


Above
Below

3.5.61 AutoOrientationFlag函數

用於傳回AutoOrientationFlag的狀態。

格式

AutoOrientationFlag

傳回值

- 0 = 停用AutoOrientationFlag
- 1 = 啟用AutoOrientationFlag

參照

AutoOrientationFlag

AutoOrientationFlag函數範例

```
If AutoOrientationFlag = Off Then
    Print " AutoOrientationFlag is off"
EndIf
```

3.5.62 AvgSpeedClear

用於清除關節速度絕對值的平均值並進行初始化。

格式

AvgSpeedClear [關節指定1 [, 關節指定2 [, 關節指定3 [, 關節指定4 [, 關節指定5 [, 關節指定6 [, 關節指定7 [, 關節指定8 [, 關節指定9]]]]]]]]

參數

關節指定1 -關節指定9

以整數值或運算式指定關節編號。未指定參數時，清除所有關節速度絕對值的平均值。
附加軸的S軸為8，T軸為9。若指定不存在的關節編號，則發生錯誤。

說明

AvgSpeedClear用於清除指定關節速度絕對值的平均值。

執行AvgSpeed前，請務必執行AvgSpeedClear。

本命令不支援PG附加軸。

參照

AvgSpeed、PeakSpeed

AvgSpeedClear範例

[例1] 這是清除所有關節的平均速度值後，顯示指定關節編號的平均速度值之命令執行範例。

```
> AvgSpeedClear
> Go P1
> AvgSpeed 1
     0.073
> AvgSpeed
     0.073    0.044
     0.021    0.069
     0.001    0.108
     0.000    0.000
     0.000
>
```

[例2] 這是對於垂直多關節機器人清除第1關節、第4關節、第5關節的平均速度值後，顯示指定關節編號的平均速度值之命令執行範例。

```
> AvgSpeedClear 4, 1, 5
> Go P1
> AvgSpeed 1
     0.226
> AvgSpeed 4
     0.207
```

3.5.63 AvgSpeed

顯示指定關節速度的絕對值之平均值。

格式

AvgSpeed [關節編號]

參數

關節編號

以整數值或運算式指定關節編號。可省略。附加軸的S軸為8，T軸為9。

結果

顯示指定關節的目前速度絕對值之平均值。未指定時，顯示所有關節速度的絕對值之平均值。

說明

AvgSpeed用於顯示指定關節速度的絕對值之平均值。可透過AvgSpeed命令瞭解馬達的負荷狀態。以0~1的實數值表示結果。最大平均速度值為「1」。

平均值為0.001以下的值時，則變為「0」。

執行AvgSpeed前，請務必執行AvgSpeedClear。

AvgSpeed命令有時間限制。執行AvgSpeedClear後，請在60秒內執行AvgSpeed。若超過60秒，則發生錯誤4088。

在虛擬控制器及空運行的情況下，根據命令速度(並非實際速度)計算速度絕對值的平均值。

本命令不支援PG附加軸。

參照

AvgSpeedClear、AvgSpeed函數、PeakSpeed

AvgSpeed範例

```
> AvgSpeedClear
> Go P1
> AvgSpeed 1
     0.226
> AvgSpeed
     0.226    0.133
     0.064    0.207
     0.003    0.314
     0.000    0.000
     0.000
>
```

3.5.64 AvgSpeed函數

用於傳回指定關節速度的絕對值之平均值。

格式

AvgSpeed (關節編號)

參數

關節編號

以整數值或運算式指定關節編號。附加軸的S軸為8，T軸為9。

傳回值

以0~1的實數值傳回。

說明

AvgSpeed函數用於傳回指定關節速度的絕對值之平均值。可透過AvgSpeed函數瞭解馬達的負荷狀態。以0~1的實數值表示結果。最大平均速度為1。

執行AvgSpeed函數前，請務必執行AvgSpeedClear。

AvgSpeed函數有時間限制。執行AvgSpeed後，請在60秒內執行AvgSpeed函數。若超過60秒，則發生錯誤4088。

在虛擬控制器及空運行的情況下，根據命令速度(並非實際速度)計算速度絕對值的平均值。

本函數不支援PG附加軸。

參照

AvgSpeed、AvgSpeedClear、PeakSpeed

AvgSpeed函數範例

是在程式中使用AvgSpeed函數的範例。

```
Function CheckAvgSpeed
  Integer i

  Go P1
  AvgSpeedClear
  Go P2
  Print "Average speeds:"
  For i = 1 To 6
    Print "Joint ", i, " = ", AvgSpeed (i)
  Next i
Fend
```

3.5.65 AvoidSingularity

設定奇點通過功能。

格式

AvoidSingularity { mode }

參數

mode

用於表示使用的特定點通過模式之整數運算式

常數	值	模式
SING_NONE	0	停用奇點通過功能。
SING_THRU	1	啟用奇點通過功能。
SING_THRUROT	2	透過帶ROT修飾詞的CP動作，啟用奇點通過功能。
SING_VSD	3	啟用變速CP動作功能。
SING_AUTO	4	自動選擇奇點通過功能或變速CP動作功能。
SING_AVOID	5	啟用臂肘奇點通過功能。

說明

以下列命令啟用AvoidSingularity的SING_THRU,SING_THRUROT,SING_AVOID。
Move, Tmove, Bmove, Arc, Arc3, Jump3, Jump3CP, JumpTLZ

以下列命令啟用AvoidSingularity的SING_VSD, SING_AUTO。
Move, Tmove, Bmove, Arc, Arc3, Jump3, Jump3CP, JumpTLZ, Cvmove

當垂直6軸機器人(包含N系列)及RS系列機器人在執行CP動作中接近奇點時，可透過奇點通過功能維持速度的同時，通過不同於原本軌跡的軌跡，以迴避加速度錯誤，而在離開奇點之後返回正常軌跡。啟動控制器時，奇點迴避功能為「1：啟用」狀態，通常不需變更。但是，為了確保與未支援奇點通過功能軟體間的相容性，以及避免因奇點迴避動作而偏離軌跡等情況，而不想迴避奇點時，請停用該功能。

當垂直6軸機器人(包含N系列)及RS系列機器人在執行CP動作中接近奇點時，可透過變速CP動作功能維持軌跡的同時，自動抑制速度，以迴避加速度錯誤或過速度錯誤，而在離開奇點後，便可恢復正常的速度命令。第1、第2、第4、第6關節有可能進行幅度較大的動作，以維持軌跡並通過特定點附近。

若有變更AvoidSingularity設定值，則於下一次啟動控制器前即可啟用。
即使已設定奇點通過功能，仍發生加速度錯誤或過速度錯誤時，請降低AccelS、SpeedS的值。
啟動控制器時，AvoidSingularity處於透過控制器設定所指定的狀態(工廠出廠時：1)。此外，變更AvoidSingularity的設定值時，SingularityAngle、SingularitySpeed、SingularityDist的各參數則恢復為預設值。SING_AUTO屬於結合SING_THRU與SING_VSD的模式。依動作或速度選擇SING_THRU或SING_VSD。

即便使用AvoidSingularity仍發生加速度錯誤或過速度錯誤時，請降低AccelS、DecelS、SpeedS的值。

注意

- 設定垂直6軸機器人和N系列機器人的奇點附近的條件

  使用第5關節角度及第4關節角速度，以判斷機器人是否接近腕部奇點附近。第5關節角度被預設為±10度，第4關節角度被預設為最大關節速度的±10%。若要變更這些值，需要使用SingularityAngle及SingularitySpeed命令。此外，使用P點

坐標以判斷是否接近臂腕奇點。P點與機器人第1關節旋轉軸的距離被預設為30 mm。若要變更該值，需要使用SingularityDist命令。

- 設定RS系列機器人的奇點附近的條件

  使用預設的工具0坐標系原點的坐標，以判斷機器人是否接近臂腕奇點。工具0坐標系的原點與機器人第1關節旋轉軸的距離被預設為30 mm。若要變更該值，需要使用SingularityDist命令。

- N系列機器人的注意事項

  N2系列不同於其它機種，啟動控制器時的奇點迴避功能被設為「3：啟用變速CP動作功能」。

  N6系列於其它機種相同，啟動控制器時的奇點迴避功能被設為「1：啟用奇點迴避功能」。

  N系列除了腕部特定點姿態和臂腕特定點姿態之外，還備有臂肘奇點區域。臂肘奇點區域為第3關節為0度時(第3關節和第2關節重疊時)的姿態。

  關於臂肘奇點區域的通過動作相關詳情，請參閱以下手冊。
  「Epson RC+ 使用者指南」

- SING_THRU和SING_AVOID的差異

  SING_THRU雖然會通過腕部奇點和肩部奇點，但不通過臂肘奇點。若要通過臂肘奇點，請選擇SING_AVOID。但是，相較於通過其他奇點，會大幅度改變軌跡，因此請注意使用臂肘奇點通過。此外，非N系列選擇SING_AVOID時，會發生錯誤4002。

參照

AvoidSingularity函數、SingualrityAngle、SingularitySpeed、SingularityDist

AvoidSingularity範例

```
AvoidSingularity SING_NONE '停用奇點迴避的狀態下進行動作
Move P1
Move P2
AvoidSingularity SING_THRU
```

3.5.66 AvoidSingularity函數

用於傳回AvoidSingularity的狀態。

格式

AvoidSingularity

傳回值

- 0 = 停用奇點通過功能
- 1 = 啟用奇點通過功能
- 2 = 以帶ROT修飾詞的CP動作命令啟用奇點通過功能
- 3 = 啟用變速CP動作功能
- 4 = 自動選擇奇點通過功能和變速CP動作功能
- 5 = 啟用臂肘奇點通過功能

參照

AvoidSingularity

AvoidSingularity函數範例

```
If AvoidSingularity = SING_NONE Then
    Print "AvoidSingularity is off"
EndIf
```

3.6 B

3.6.1 Base

定義和顯示基本坐標系。

格式

(1) Base 坐標系資料
(2) Base 原點, X軸指定, Y軸指定 [, { X | Y }]
(3) Base

參數

坐標系資料
　　以點數據直接指定基本坐標系的原點與方向。
原點
　　以P#(整數)或P(運算式)指定要定義基本坐標系原點的機器人坐標系上的位置。
X軸指定
　　以P#(整數)或P(運算式)指定要定義基本坐標系X軸上的點之機器人坐標系上的位置。
Y軸指定
　　以P#(整數)或P(運算式)指定要定義基本坐標系Y軸上的點之機器人坐標系上的位置。
X
　　用於優先讓X軸指定與X軸一致。可省略。(預設)
Y
　　用於優先讓Y軸指定與Y軸一致。可省略。

說明

機器人上有不可變更的標準坐標系，一般將其稱為「機器人坐標系」。相對於前述坐標系，用作一般本地坐標系的基礎並可變更原點坐標的坐標系稱為「基本坐標系」。

透過設定相對於對機器人坐標系的本地坐標系的基點與旋轉角，來定義本地坐標系。

若要將基本坐標系重設為預設值，請執行以下陳述式。執行該陳述式後，基本坐標系便和機器人坐標系一樣。

```
Base XY(0, 0, 0, 0)
```

注意

一旦變更基本坐標系，則會對所有本地定義帶來影響。

變更基本坐標系後，請重新定義所有本地坐標系。

參照

Local

Base範例

將基本坐標系的原點定義為X軸100mm、Y軸100mm的位置。

```
> Base XY(100, 100, 0, 0)
```

3.6.2 BClr函數

用於清除指定數值的1位元，並傳回該值。

格式

BClr (數值, 位元編號)

參數

數值

以運算式或數值指定要清除位元的數值。

位元編號

以運算式或數值指定要清除的位元編號(0~31的整數值)。

傳回值

用於傳回已清除位元的值(整數)。

參照

BClr64、BSet、BSet64、BTst、BTst64

BClr函數範例

```
flags = BClr(flags, 1)
```

3.6.3 BClr64函數

用於清除指定數值的1位元，並傳回該值。

格式

BClr64 (數值, 位元編號)

參數

數值

以運算式或數值指定要清除位元的數值。

位元編號

以運算式或數值指定要清除的位元編號(0~63的整數值)。

傳回值

用於傳回已清除位元的值(整數)。

參照

BClr、BSet、BSet64、BTst、BTst64

BClr64函數範例

```
flags = BClr64(flags, 1)
```

3.6.4 BGo

用於執行所選本地坐標系中的位移PTP動作。

格式

BGo 目標坐標 [CP] [Till | Find] [!平行處理!] [SYNC]

參數

目標坐標
　使用點數據指定動作的目標位置。
CP
　指定路徑運動。可省略。
Till | Find
　用於記述Till或Find運算式。可省略。

```
Till | Find
Till Sw(運算式) = {On | Off}
Find Sw(運算式) = {On | Off}
```

!平行處理!
　可附加平行處理陳述式，以便在動作中執行I/O等命令。可省略。
SYNC
　用於預約動作命令。在以SyncRobots開始動作之前，機器人不進行動作。

說明

執行所選本地坐標系中的位移PTP動作。將以表示目標坐標的點資料指定之坐標系作為基準，進行位移PTP動作。

未指定本地坐標系時，則以本地0(基本坐標系)為基準，進行位移PTP動作。

忽略點資料所賦予的姿態旗標，維持目前姿態旗標。不過，在垂直6軸型機器人(包含N系列)的情況下，將姿態旗標自動變更為縮小關節移動量的狀態。這等同於透過Move命令指定LJM修飾參數時的情況。若要進行180度以上的姿態變化，請分次執行。

可利用Till修飾詞，在Till條件成立時，讓運作中的機器人減速並停止，以完成BGo動作。

動作中當Find條件變為真時，則利用Find修飾詞將點儲存在FindPos中。

可利用!平行處理!，與動作平行進行其它處理。

若已添加CP參數，可在開始動作減速時重疊下一個動作命令的加速。此時，不在目標坐標上進行定位。

參照

Accel、BMove、Find、!平行處理!、P# = 點指定、Speed、Till、TGo、TMove、Tool

BGo範例

```
> BGo XY (100, 0, 0, 0)     '(本地坐標系上)朝X方向移動100mm

Function BGoTest

  Speed 50
  Accel 50, 50
  Power High

  P1 = XY(300, 300, -20, 0)
  P2 = XY(300, 300, -20, 0) /L
  Local 1, XY(0, 0, 0, 45)
```

```
  GoP1
  Print Here
  BGo XY(0, 50, 0, 0)
  Print Here

  Go P2
  Print Here
  BGo XY(0, 50, 0, 0)
  Print Here

  BGo XY(0, 50, 0, 0) /1
  Print Here

Fend
```

[輸出結果]

```
 X:  300.000 Y:  300.000 Z:  -20.000 U:    0.000 V:    0.000 W:    0.000 /R /0
 X:  300.000 Y:  350.000 Z:  -20.000 U:    0.000 V:    0.000 W:    0.000 /R /0
 X:  300.000 Y:  300.000 Z:  -20.000 U:    0.000 V:    0.000 W:    0.000 /L /0
 X:  300.000 Y:  350.000 Z:  -20.000 U:    0.000 V:    0.000 W:    0.000 /L /0
 X:  264.645 Y:  385.355 Z:  -20.000 U:    0.000 V:    0.000 W:    0.000 /L /0
```

3.6.5 BMove

在選擇的本地坐標系執行位移線性動作。

格式

BMove目標坐標 [ROT] [CP] [Till | Find] [!平行處理!] [SYNC]

參數

目標坐標
　使用點數據指定動作的目標位置。
ROT
　以工具姿態變化為優先，指定速度和加減速度。可省略。
CP
　指定路徑運動。可省略。
Till | Find
　用於記述Till或Find運算式。可省略。

```
Till | Find
Till Sw(運算式) = {On | Off}
Find Sw(運算式) = {On | Off}
```

! 平行處理 !
　可附加平行處理陳述式，以便在動作中執行I/O等命令。可省略。
SYNC
　用於預約動作命令。在以SyncRobots開始動作之前，機器人不進行動作。

說明

在選擇的本地坐標系執行位移線性動作。將以表示目標坐標的點資料指定之坐標系作為基準，進行位移線性動作。

未指定本地坐標系時，則以本地0(基本坐標系)為基準，進行位移PTP動作。

忽略點資料所賦予的姿態旗標，維持目前姿態旗標。不過，在垂直6軸型機器人(包含N系列)的情況下，將姿態旗標自動變更為縮小關節移動量的狀態。這等同於透過Move命令指定LJM修飾參數時的情況。若要進行180度以上的姿態變化，請分次執行。

BMove的速度和加減速度分別使用SpeedS和AccelS的設定值。關於速度和加減速度的關係，請參閱「注意」項目中的「連同BMove一起使用CP」。但是，指定ROT修飾參數時的速度和加減速度分別使用SpeedR和AccelR的設定值。此時，SpeedS和AccelS的設定值即呈停用狀態。

機器人的工具中心點位置固定為特定坐標，僅改變工具姿態時，或者相對於工具中心點的移動距離，工具姿態變化較大時，工具姿態變化速度可能會明顯變快。為防止這種情形，工具姿態變化速度大時，自動限制動作速度的功能就會啟用。

想要手動設定CP動作時的工具姿態變化速度的上限值時，請啟用SpeedRLimitation。啟用SpeedRLimitation時，CP動作時的工具姿態變化速度超過設定的SpeedR時，工具姿態變化速度會變為SpeedR來限制動作速度。工具姿態變化速度超過設定的SpeedR時，以設定的SpeedS進行動作。工具姿態變化速度的上限值，請事先在SpeedR設定。

透過使用Till修飾詞使Till條件成立時，可在動作途中使機器人減速並停止，以完成BMove。

動作中當Find條件的值變為真(True)時，則用Find修飾詞將點資料儲存在FindPos中。

可利用!平行處理!，與動作平行進行其它處理。

注意

- 連同CP一起使用BMove

若使用CP參數，則在開始減速的同時，移至下一個陳述式控制動作命令。其便利之處在於，使用者可連接數個動作命令，以恆定的速度執行連續動作。在未指定CP的BMove命令時，手臂一定會減速，並停於指定的目標坐標上。

參照

AccelS、BGo、Find、!平行處理!、P# = 點指定、SpeedS、TGo、Till、TMove、Tool

BMove範例

```
> BMove XY (100, 0, 0, 0)     '(本地坐標系上)朝X方向移動100mm

Function BMoveTest

  Speed 50
  Accel 50, 50
  SpeedS 100
  AccelS 1000, 1000
  Power High

  P1 = XY(300, 300, -20, 0)
  P2 = XY(300, 300, -20, 0) /L
  Local 1, XY(0, 0, 0, 45)

  Go P1
  Print Here
  BMove XY(0, 50, 0, 0)
  Print Here

  Go P2
  Print Here
  BMove XY(0, 50, 0, 0)
  Print Here

  BMove XY(0, 50, 0, 0) /1
  Print Here

Fend
```

[輸出結果]

```
 X:  300.000 Y:  300.000 Z:  -20.000 U:    0.000 V:    0.000 W:    0.000 /R /0
 X:  300.000 Y:  350.000 Z:  -20.000 U:    0.000 V:    0.000 W:    0.000 /R /0
 X:  300.000 Y:  300.000 Z:  -20.000 U:    0.000 V:    0.000 W:    0.000 /L /0
 X:  300.000 Y:  350.000 Z:  -20.000 U:    0.000 V:    0.000 W:    0.000 /L /0
 X:  264.645 Y:  385.355 Z:  -20.000 U:    0.000 V:    0.000 W:    0.000 /L /0
```

3.6.6 Boolean

用於宣告Boolean變數。(大小：2位元組)

格式

Boolean 變數名稱 [(陣列變數的最大元素編號)] [, 變數名稱[(陣列變數的最大元素編號)]...]

參數

變數名稱

指定宣告為Boolean型的變數名稱。

陣列變數的最大元素編號

是陣列變數的最大元素編號，最大可宣告三維。使用如下格式。可省略。

(最大元素編號1, [最大元素編號2], [最大元素編號3])

元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。

在所有元素數不超過以下最大值的範圍內，指定各最大元素編號。

- 本地變數：2,000
- 備份變數(Global Preserve)：4,000
- 全域變數和模組變數：100,000

說明

Boolean用於將變數宣告為Boolean型。Boolean變數為「True」或「False」。在Function的開頭宣告本地變數。在Function之外宣告全域變數和模組變數。

參照

Byte、Double、Global、Int32、Int64、Integer、Long、Real、Short、String、UByte、UInt32、UINT64、UShort

Boolean範例

```
Boolean partOK
Boolean A (10)          'Boolean型的一維陣列
Boolean B (10, 10)      'Boolean型的二維陣列
Boolean C (5, 5, 5)     'Boolean型的三維陣列

partOK = CheckPart()
If Not partOK Then
   Print "Part check failed"
EndIf
```

3.6.7 BOpen

以二進位模式開啟檔案。

格式

BOpen 檔名 As #檔案編號

.

.

Close #檔案編號

參數

檔案名稱

　　指定包含路徑的檔案名稱字串。僅指定檔名時，即指目前目錄中的檔案。詳細內容請參閱ChDisk。

檔案編號

　　以30~63的整數值或運算式來指定。

說明

以指定的檔案編號開啟指定的檔案。此陳述式用於在指定檔案上以二進位模式進行存取。沒有指定的檔案時，則新建檔案。若有，則從既有資料的開頭讀寫資料。若要以二進位模式讀/寫資料，請分別使用ReadBin、WriteBin命令。

注意

可使用網路路徑。

在檔案處於開啟狀態下，指定的檔案編號用於識別該檔案。因此，在關閉該檔案之前，不可將相同的檔案編號用於其它檔案。在檔案操作命令(ReadBin, WriteBin, Seek, Eof, Flush, Close)中使用檔案編號。

可用Seek命令切換檔案的載入/寫入位置(指標)。若要切換載入存取和寫入存取，請用Seek命令重新設定檔案指標位置。

以Close陳述式關閉檔案，並釋放檔案編號。

請以FreeFile函數取得檔案編號，以避免將同一編號用在多項工作上。

參照

Close、AOpen、FreeFile、ReadBin、ROpen、UOpen、WOpen、WriteBin

BOpen範例

```
Integer fileNum, i

fileNum = FreeFile
BOpen "TEST.DAT" As #fileNum
For i = 0 To 100
   WriteBin #fileNum, i
Next i

Flush #fileNum
Seek #fileNum, 10
ReadBin #fileNum, i
Print "data = ", i
Close #fileNum
```

3.6.8 Box

設定和顯示進入檢測區。

格式

(1) Box 區域編號 [, 機器人編號], X軸下限位置, X軸上限位置, Y軸下限位置, Y軸上限位置, Z軸下限位置, Z軸上限位置 [本地編號]
(2) Box 區域編號, 機器人編號, X軸下限位置, X軸上限位置, Y軸下限位置, Y軸上限位置, Z軸下限位置, Z軸上限位置, 遠端輸出邏輯 [本地編號]
(3) Box 區域編號, 機器人編號
(4) Box

參數

區域編號
以1~15的整數值指定要設定的區域編號。
機器人編號
以整數值指定要設定的機器人編號。若在格式(1)中省略此編號，則以目前選擇的機器人為對象。在格式(2)和(3)中不可省略。
X軸下限位置
以數值或運算式指定要設定的區域下限位置之X坐標值(實數)。
X軸上限位置
以數值或運算式指定要設定的區域上限位置之X坐標值(實數)。
Y軸下限位置
以數值或運算式指定要設定的區域下限位置之Y坐標值(實數)。
Y軸上限位置
以數值或運算式指定要設定的區域上限位置之Y坐標值(實數)。
Z軸下限位置
以數值或運算式指定要設定的區域下限位置之Z坐標值(實數)。
Z軸上限位置
以數值或運算式指定要設定的區域上限位置之Z坐標值(實數)。
遠程控制輸出邏輯
On | Off
用於設定遠端輸出的邏輯。進入Box時，若欲開啟I/O輸出，則設定為On；進入Box時，若欲關閉I/O輸出，則設定為Off。省略參數時，則設定為On。
本地編號
指定本地坐標系編號(0~15)。
請務必在編號前加「/LOCAL」。省略參數時，則設定本地坐標系編號0。

結果

- 用格式(3)指定後，則顯示指定區域編號的區域設定。
- 用格式(4)指定後，則顯示目前選擇的機器人中設定的所有區域設定。

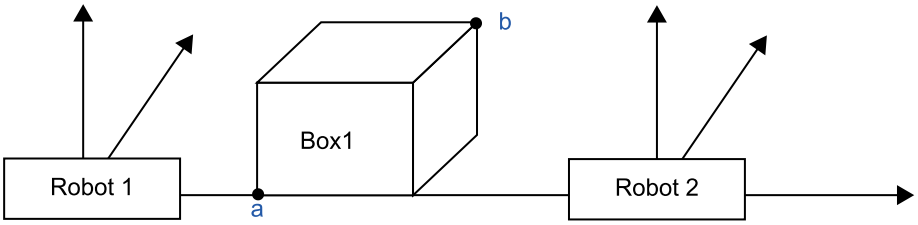
說明

Box用於設定進入檢測區。若設定進入檢測區，可檢測到：基於目前選擇的工具計算出的手臂尖端位置是否進入設定的進入檢測區內。在機器人的基本坐標系或以本地編號指定的本地坐標系上設定進入檢測區。被指定為指定坐標系的X、Y、Z各軸之下限位置和上限位置之間為進入檢測區。

若設定進入檢測區，在啟動控制器時，無論機器人的馬達電源處於何種狀態，都始終執行檢測處理。

可使用GetRobotInsideBox函數、InsideBox函數，隨時取得進入檢測的結果。此外，可利用GetRobotInsideBox函數，作為Wait命令的等待條件運算式。而且，還可進行遠端輸出設定，以便將檢測結果輸出到I/O。

若要讓數個機器人共享1個區域，需定義從每個機器人坐標所見的區域。



符號	說明
a	X、Y、Z軸下限位置
b	X、Y、Z軸上限位置

從機器人1所見Box1的設定

```
Box 1, 1, 100, 200, 0, 100, 0, 100
```

X、Y、Z軸下限位置的坐標為(100, 0, 0)且X、Y、Z軸上限位置的坐標為(200, 100, 100)。

從機器人2所見Box1的設定

```
Box 1, 2, -200, -100, 0, 100, 0, 100
```

X、Y、Z軸下限位置的坐標為(-200, 0, 0)且X、Y、Z軸上限位置的坐標為(-100, 100, 100)。

注意

- 在各坐標軸上，停用進入檢測區的設定

  可在各坐標軸上停用設定。例如，若要只停用Z軸設定，請將Z軸下限位置及Z軸上限位置設定為「0」。例: Box 1, 200, 300, 0, 500, 0, 0

  此時，會判斷是否進入XY的二維平面。

- 進入檢測區的預設值

  Box的預設值是「0, 0, 0, 0, 0, 0」。(進入檢測區的設定會變為停用。)

- 選擇工具

  透過目前選擇的工具執行進入檢測。已變更工具選擇時，有可能在機器人未動作的狀態下從區域內移到區域外，或從區域外移到區域內。

- 附加軸

  在帶有附加軸(包括行走軸在內)S、T的機器人之情況下，設定的進入檢測平面不依賴於附加軸位置。以機器人的基本坐標系為基準進行設定。

提示

- 由Robot Manager設定Box

  Epson RC+可透過[工具]選單-[機器人管理器]的[工作空間]面板設定Box值。

參照

BoxClr、BoxDef、GetRobotInsideBox、InsideBox、Plane

Box範例

[例1] 如下所述為在命令視窗上設定Box值並顯示數值的簡易操作範例。

```
> Box 1, -200, 300, 0, 500, -100, 0

> Box
Box 1: 1, -200.000, 300.000, 0.000, 500.000, -100.000, 0.000, ON /LOCAL0
```

[例2] 如下所述是在本地編號上指定1、2，以設定Box值的簡易程式。

```
Function SetBox

   Integer i

   Box 1, -200, 300, 0, 500, -100, 0 /LOCAL1

   i = 2
   Box 2, 100, 200, 0, 100, -200, 100 /LOCAL(i)

Fend
```

3.6.9 Box函數

用於傳回已設定的進入檢測區。

格式

Box (區域編號[, 機器人編號], 參照資料)

參數

區域編號
　　以運算式或數值指定要確認的區域編號。
機器人編號
　　以整數值指定要設定的機器人編號。省略時，即以目前選擇的機器人為對象。
參照資料
　　以整數值指定作為傳回值來傳回的資料。- 1：下限位置 - 2：上限位置

傳回值

若作為參照資料指定1，則將以Box設定值指定的X軸下限位置視為點資料X，將Y軸下限位置視為點資料Y，將Z軸下限位置視為點資料Z，以進行傳回。

若作為參照資料指定2，則將以Box設定值指定的X軸上限位置視為點資料X，將Y軸上限位置視為點資料Y，將Z軸上限位置視為點資料Z，以進行傳回。

參照

Box、BoxClr、BoxDef、GetRobotInsideBox、InsideBox

Box函數範例

```
P1 = Box(1,1)
P2 = Box(1,2)
```

3.6.10 BoxClr

用於清除設定的Box。

格式

BoxClr 區域編號[, 機器人編號]

參數

區域編號

以運算式或數值指定要清除設定的工作空間編號(1~15的整數)。

機器人編號

以整數值指定要設定的機器人編號。省略時，即以目前選擇的機器人為對象。

參照

Box、BoxDef、GetRobotInsideBox、InsideBox

BoxClr函數範例

以下為使用BoxClr函數的程式。

```
Function ClearBox

   If BoxDef(1) = True Then
      BoxClr 1
   EndIf
Fend
```

3.6.11 BoxDef函數

傳回有無設定進入檢測區。

格式

BoxDef (區域編號 [, 機器人編號])

參數

區域編號

指定要傳回狀態的工作空間編號(1~15的整數)。

機器人編號

以整數值指定要設定的機器人編號。省略時，即以目前選擇的機器人為對象。

傳回值

若有設定以區域編號指定的進入檢測區，則傳回「True」；未設定時，則傳回「False」。

參照

Box、BoxClr、GetRobotInsideBox、InsideBox

BoxDef函數範例

以下為使用BoxDef函數的程式。

```
Function ClearBox

   If BoxDef(1) = True Then
      BoxClr 1
   EndIf
Fend
```

3.6.12 Brake

啟用或停用目前機器人的指定關節之制動器。

格式

Brake 狀態, 關節編號

參數

狀態

- 啟用制動器時：使用On。
- 解除制動器時：使用Off。

關節編號

指定1~6的關節編號。

說明

Brake命令用於對垂直6軸型機器人(包含N系列)的1個關節，啟動或解除制動器。水平多關節機器人(包括RS系列)不會動作。

這是專為維修人員使用而設計的命令。

執行Brake命令後，對機器人控制參數進行初始化。詳細內容請參閱Motor On。

⚠ 警告

解除制動器時，敬請注意。請務必確認是否正確地支撐關節。若未正確支承關節，則會導致其掉落，造成機器人故障或作業員身負重傷。

注意

- 請務必在可操作緊急停止開關的狀態下，執行Brake Off命令。

  若控制器處於緊急停止狀態，馬達制動器則會被鎖定。執行Brake Off命令後，機器人手臂有可能因自重而下降。

參照

Motor、Power、Reset、SFree、SLock

Brake範例

```
> brake on, 1

> brake off, 1
```

3.6.13 Brake函數

用於傳回指定關節的制動器狀態。

格式

Brake (關節編號)

參數

關節編號

以整數值或整數運算式指定關節編號。可指定的值是1~機器人的關節數。

傳回值

0 = 制動器Off、1 = 制動器On

參照

Brake

Brake函數範例

```
If brake (1) = OFF Then
  Print "Joint 1 brake is off"
EndIf
```

3.6.14 BSet函數

用於設定指定數值的1位元，並傳回該結果。

格式

BSet (數值, 位元編號)

參數

數值

以運算式或數值指定要設定位元的數值。

位元編號

以運算式或數值指定要設定的位元編號(0~31的整數值)。

傳回值

用於傳回已設定位元的值(整數)。

參照

BClr、BClr64、BSet64、BTst、BTst64

BSet函數範例

```
flags = BSet(flags, 1)
```

3.6.15 BSet64函數

用於設定指定數值的1位元，並傳回該結果。

格式

BSet64 (數值, 位元編號)

參數

數值

以運算式或數值指定要設定位元的數值。

位元編號

以運算式或數值指定要設定的位元編號(0~63的整數值)。

傳回值

用於傳回已設定位元的值(整數)。

參照

BClr、BClr64、BSet、BTst、BTst64

BSet函數範例

```
flags = BSet64(flags, 1)
```

3.6.16 BTst函數

用於傳回數值的1位元狀態。

格式

BTst (數值, 位元編號)

參數

數值

以運算式或數值指定要進行位元測試的數值。

位元編號

以數值指定要測試的位元編號(0~31的整數值)。

傳回值

以1或0傳回位元測試的值。

參照

BClr、BClr64、BSet、BSet64、BTst64

BTst函數範例

```
If BTst(flags, 1) Then
    Print "Bit 1 is set"
EndIf
```

3.6.17 BTst64函數

用於傳回數值的1位元狀態。

格式

BTst64 (數值, 位元編號)

參數

數值

以運算式或數值指定要進行位元測試的數值。

位元編號

以數值指定要測試的位元編號(0~63的整數值)。

傳回值

以1或0傳回位元測試的值。

參照

BClr、BClr64、BSet、BSet64、BTst

BTst64函數範例

```
If BTst64(flags, 1) Then
    Print "Bit 1 is set"
EndIf
```

3.6.18 Byte

用於宣告Byte變數。(大小：2位元組)

格式

Byte 變數名稱 [(陣列變數的最大元素編號)] [, 變數名稱[(陣列變數的最大元素編號)]...]

參數

變數名稱

指定宣告為Byte型的變數名稱。

陣列變數的最大元素編號

是陣列變數的最大元素編號，最大可宣告三維。使用如下格式。可省略。

(最大元素編號1, [最大元素編號2], [最大元素編號3])

元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。在所有元素數不超過以下最大值的範圍內，指定各最大元素編號。

- 本地變數：2,000
- 備份變數(Global Preserve)：4,000
- 全域變數和模組變數：100,000

說明

Byte用於將變數宣告為Byte型。Byte變數的範圍為-128~+127。在Function的開頭宣告本地變數。在Function之外宣告全域變數和模組變數。

參照

Boolean、Double、Global、Int32、Int64、Integer、Long、Real、Short、String、UByte、UInt32、UInt64、UShort

Byte範例

在以下範例中宣告Byte變數，並對該變數賦予數值。

確認到變數test_ok的最高位元是1或0。將該結果顯示於顯示器上。(在本例中，對變數賦予15的值，因此始終有設定變數test_ok值較高的位元。)

```
Function Test
    Byte A (10)           'Byte型的一維陣列
    Byte B (10, 10)       'Byte型的二維陣列
    Byte C (5, 5, 5)      'Byte型的三維數組陣列
    Byte test_ok
    test_ok = 15
    Print "Initial Value of test_ok = ", test_ok
    test_ok = (test_ok And 8)
    If test_ok <> 8 Then
       Print "test_ok high bit is ON"
    Else
       Print "test_ok high bit is OFF"
    EndIf
Fend
```

3.7 C

3.7.1 Calib

以使用CalPls設定的脈衝值取代目前手臂姿態的脈衝值。

格式

Calib 關節編號1 [, 關節編號2] [, 關節編號3] [, 關節編號4] [, 關節編號5] [, 關節編號6] [, 關節編號7] [, 關節編號8] [, 關節編號9]

參數

關節編號

用於直接以數值指定要執行校正的關節編號(1~9的整數)。通常，針對每一個關節逐一進行校正，但可利用此命令最多同時校正9個關節。附加軸的S軸為8，T軸為9。

說明

自動運算位移(Hofs)值並進行設定。為了與各關節馬達原點相應的機器人機械原點匹配，需要此位移值。

Calib是馬達脈衝值發生變更時所使用的命令。一般用法是，用於更換馬達後的作業。通常，校正位置脈衝值必定會與CalPls脈衝值相同，但在更換馬達等維修作業之後，這2個值不會一致。因此，需要進行校正。

將手臂移至要進行校正的位置後，執行Calib命令。執行Calib後，校正位置的脈衝值會變更為CalPls脈衝值(相對於校正位置的正確脈衝值)。

務必設定Hofs值，以執行正確的校正。若將手臂移至要進行校正的位置並執行Calib，則自動運算Hofs值。控制器基於校正脈衝值和CalPls脈衝值自動運算Hofs值。

注意

- 使用Calib命令時需注意

  Calib僅供維修作業而設計。如非必要，請勿使用。

  執行Calib後，Hofs值會被取代。若在不清楚Hofs值的情況下進行變更，機器人則會進行超出預期的動作，因此，如非必要，嚴禁使用Calib。

  在對應關節精度修正的機型中，執行Calib時，對於指定的軸，JointAccuracy 設定的修正值為“0”。

- 使用搭載Safety板控制器時，請在運行本命令以後啟動安全功能管理員

  使用搭載Safety板的控制器時，控制器Hofs值需要和安全功能的Safety板的Hofs值相匹配。

  運行本命令，僅改變控制器的Hofs值，這樣會因為與Safety板的Hofs值不匹配而發生警報。

  因此，運行版命令之後，請啟動安全功能管理員，更新Safety板設定。

  有關詳細資訊，請參閱以下手冊。
  「機器人控制器 安全功能手冊」

常見錯誤

若未指定關節編號，則發生錯誤

使用Calib命令時，若未指定關節編號，則發生錯誤。

參照

CalPls、JointAccuracy、HofsJointAccuracy、Hofs

Calib範例

命令視窗中的操作

```
> CalPls     '用於顯示目前CalPls值
  65523, 43320, -1550, 21351
> Pulse     '用於顯示目前Pulse值
PULSE: 1: 65526 pls 2: 49358 pls 3: 1542 pls 4: 21299 pls
> Calib 2     '僅對第2關節執行校正
> Pulse     '顯示已變更的Pulse值
PULSE: 1: 65526 pls 2: 43320 pls 3: 1542 pls 4: 21299 pls
>
```

3.7.2 Call

叫用作為副程式的函式。

格式

Call 函式名稱 [(引數清單)]

參數

函數名稱
　　指定要呼叫的函數名稱。
引數清單
　　指定函數宣告所指定的引數清單。請使用以下格式的引數。可省略。[ByRef]變數名稱[( )]、或算式
ByRef
　　參照要呼叫的函數變數時，指定ByRef。此時，可將函式內的引數變更反映於叫用側的變數中。可變更透過參照而傳遞的值。可省略。

說明

使用Call命令將程式控制切換到Function...Fend定義的函式。使用Call命令，將程式的執行從目前函式切換到Call命令所指定的函式。在找到Exit Function或Fend之前，直接以被叫用的函式繼續執行程式。接著，以Call命令的下一個陳述式將控制返回到原有的函式。

亦可省略Call關鍵字或引數的括弧。請參閱以下範例。

```
Call MyFunc(1, 2)
MyFunc 1, 2
```

也可叫用DLL(動態連結程式庫)所定義的外部函式。詳細內容請參閱Declare陳述式。

若要在函式內執行副程式，請使用GoSub...Return。

也可指定變數作為引數。可指定ByRef參數，並將函式內的引數變更反映於叫用側的變數中。

要指定ByRef參數時，需和函式定義(Function陳述式)或DLL函式定義(Declare陳述式)的引數清單一樣，指定ByRef。

要將陣列變數作為引數進行傳遞時，需要ByRef。

參照

Function、GoSub

Call範例

```
[File1: MAIN.PRG]
Function main
    Call InitRobot
Fend

[File2: INIT.PRG]
Function InitRobot

    If Motor = Off Then
        Motor On
    EndIf
    Power High
    Speed 50
    Accel 75, 75
Fend
```

3.7.3 CalPls

設定和顯示用於校正的位置姿態脈衝值。

格式

(1) CalPls 第1關節脈衝值, 第2關節脈衝值, 第3關節脈衝值, 第4關節脈衝值 [, 第5關節脈衝值,第6關節脈衝值] [, 第7關節脈衝值] [, 第8關節脈衝值, 第9關節脈衝值]

(2) CalPls

參數

第1關節脈衝值
　　以運算式或數值指定第1關節的脈衝值(整數)。
第2關節脈衝值
　　以運算式或數值指定第2關節的脈衝值(整數)。
第3關節脈衝值
　　以運算式或數值指定第3關節的脈衝值(整數)。
第4關節脈衝值
　　以運算式或數值指定第4關節的脈衝值(整數)。
第5關節脈衝值
　　以運算式或數值指定第5關節的脈衝值(整數)。可省略。
第6關節脈衝值
　　以運算式或數值指定第6關節的脈衝值(整數)。可省略。
第7關節脈衝值
　　以運算式或數值指定第7關節的脈衝值(整數)。可省略。
第8關節脈衝值
　　以運算式或數值指定第8關節的脈衝值(整數)。可省略。
第9關節脈衝值
　　以運算式或數值指定第9關節的脈衝值(整數)。可省略。

傳回值

若省略脈衝值，則顯示目前設定的脈衝值。

說明

輸入和儲存用於校正執行位置的正確的脈衝值。

此命令用於維護。用於馬達原點偏離機械手臂的機械原點之情況(如更換馬達等時)。一般將使該原點一致的作業稱為校正。

在正常狀態下，校正執行位置的脈衝值和以CalPls設定的脈衝值是一致的。但是，在進行更換馬達等維修之後，這2個值不會一致，因此需進行校正。

作為校正方法之一，有將手臂移至特定位置以執行Calib的方法。透過執行Calib，用於校正的特定位置脈衝值會變更為以CalPls設定的脈衝值(相對於校正位置的正確脈衝值)。

務必設定Hofs值，以執行校正。為了自動運算Hofs值，將手臂移到用於執行校正的位置，以執行Calib。控制器基於校正執行位置的脈衝值自動運算Hofs值。

注意

不得透過關閉電源變更CalPls值

即便關閉控制器電源後重新啟動，也不會對CalPls值進行初始化。變更CalPls值的方法，僅限於執行Calib命令。

參照

Calib、Hofs

CalPls範例

監視器視窗操作

```
> CalPls    ' 用於顯示目前CalPls值
65523, 43320, -1550, 21351
> Pulse
PULSE: 1: 65526 pls  2: 49358 pls  3: -1542 pls  4: 21299 pls
> Calib 4
> Pulse
PULSE: 1: 65526 pls  2: 49358 pls  3: -1542 pls  4: 21351 pls
>
```

3.7.4 CalPls函數

傳回用於校正的脈衝值(以CalPls設定)。

格式

CalPls (關節編號)

參數

關節編號

以運算式或數值指定要參照的關節編號，或傳回有無CalPls設定的0。附加軸的S軸為8，T軸為9。

傳回值

用於傳回指定關節的CalPls設定值(整數值，單位：脈衝)。關節編號為0時，則傳回有無CalPls設定(1或0)。

參照

CalPls

CalPls函數範例

是使用CalPls函數的程式範例。

```
Function DisplayCalPlsValues
   Integer i

   Print "CalPls Values:"
   For i = 1 To 4
      Print "Joint ", i, " CalPls = ", CalPls(i)
   Next i
Fend
```

3.7.5 ChDir

變更和顯示目前目錄。

格式

(1) ChDir 路徑名稱
(2) ChDir

參數

路徑名稱
　　直接以字串或字串運算式指定變更目的地的路徑名稱。詳細內容請參閱ChDisk。

說明

- (1) 若指定參數，則變更為指定的目錄。
- (2) 若省略參數，則顯示目前目錄。用於目前目錄不詳的情況。

磁碟為PC時，可執行。

在程式中執行此命令時，請將路徑名稱首尾加上[""]符號。

已連接的控制器中有最後創建專案的目錄即為目前目錄。控制器中沒有創建資訊時，根目錄即為目前目錄。

以ChDrive變更驅動器時，根目錄即為目前目錄。

參照

ChDrive、ChDisk、CurDir$

ChDir範例

命令視窗操作範例

```
> ChDir \          '將目前目錄變更為根目錄
> ChDir..         '將目前目錄變更為父目錄
```

程式執行範例

```
ChDir "\"         '將目前目錄變更為根目錄
ChDir ".."        '將目前目錄變更為父目錄
```

3.7.6 ChDisk

設定用於操作檔案的對象磁碟。

格式

ChDisk PC|USB|RAM

參數

PC
　PC上的資料夾(硬碟等)
USB
　SPEL+控制部分上的USB記憶體
RAM
　SPEL+控制部分上的記憶體

說明

指定用於操作檔案的對象磁碟。預設為PC。

機器人控制器支援作為檔案操作對象的以下磁碟。

- PC：以PC上的資料夾為對象。

  接通電源時設為PC。一般情況下不需從PC進行變更。

  可存取專案資料夾內的檔案。

- USB：以連接於控制器記憶體連接埠的USB記憶體為對象。用於在不使用PC部分的情況(不與RC+進行協作等)下移動檔案。

  T/VT系列機器人不支援在參數中指定USB。

- RAM：以記憶體中的暫存檔案為對象。

  若關閉控制器電源，則無法儲存檔案。

  用於暫時保管資料。

SPEL+的命令包括以ChDisk的設定改變/不改變檔案操作對象的命令。此外，還包括僅限於PC才能啟用ChDisk設定的命令。

- 不受ChDisk、ChDrive、ChDir的影響
  - Curve、CVMove、LoadPoints、SavePoints、ImportPoints 的檔名
  - 始終以專案資料夾為對象。
  - 只可指定檔名時，若指定路徑，則發生錯誤。
- 不受ChDisk的影響
  - OpenDB的Access、Excel檔名、ImportPoints的來源路徑、VLoadModel、VSaveImage、VSaveModel
  - 始終以Windows資料夾為對象。
  - 僅指定檔名時，則會受目前驅動器和目前資料夾的影響。也可指定完整路徑。
- 唯有在ChDisk為PC時方可執行
  - ChDir, FolderExists, MkDir, RenDir, RmDir
  - 若將ChDisk設定為非PC的情況下執行，則會發生錯誤。
  - 僅指定檔名或目錄名稱時，則會受目前驅動器或目前資料夾的影響。也可指定完整路徑。

    USB和RAM沒有目錄的概念。

- 也可在ChDisk為USB或RAM時執行
  - Copy, Del, FileDataTime, FileExist, FileLen, AOpen, BOpen, ROpen, UOpen, WOpen, Rename
  - ChDisk為PC時：僅指定檔案名稱或目錄名稱時，則會受目前驅動器或目前資料夾的影響。也可指定完整路徑。
  - ChDisk為USB或RAM時：只可指定檔案名稱時，若指定路徑，則發生錯誤
- 特殊
  - Declare
  - 詳細內容請參閱Declare。直接處理指定檔名。不受目前驅動器或目前資料夾的影響。

如下所述為ChDisk為PC時的完整路徑確定方法。

- 僅限於檔名 `"abc.txt"`

  目前驅動器　目前目錄　指定檔名

  `"C:\EpsonRC80\Projects\ProjectName\abc.txt"`

- 未包含驅動器的完整路徑 `"\abc.txt"`

  目前驅動器　指定完整路徑

  `"C:\abc.txt"`

- 包含驅動器的完整路徑 `"d:\abc.txt"`

  直接使用指定完整路徑

  `"d:\abc.txt"`

- 驅動器為網路資料夾 `"k:\abc.txt"`

  直接使用指定完整路徑

  `"k:\abc.txt"`

- 網路路徑 `"\\Epson\data\abc.txt"`

  直接使用指定完整路徑

  `"\\Epson\data\abc.txt"`

控制器上只有1個ChDisk設定。

若要作為系統以數個磁碟為對象，需執行排他控制，以切換ChDisk設定。

參照

ChDir、ChDrive、CurDisk$

ChDisk範例

命令視窗操作範例

```
> ChDisk PC
```

3.7.7 ChDrive

用於變更目前驅動器。

格式
ChDrive 驅動器名稱

參數

驅動器名稱
　　指定有效的驅動器(1個字元)。

說明
磁碟為PC時，可執行。

開啟電源時，若專案處於關閉狀態，C即為目前驅動器。若專案處於開啟狀態，存在該專案的驅動器即為目前驅動器。

詳細內容請參閱ChDisk。

參照
ChDir、ChDisk、CurDrive$

ChDrive範例
命令視窗操作範例

```
> ChDrive d
```

3.7.8 ChkCom函數

用於傳回通訊連接埠接收緩衝區內的字元數。

格式

ChkCom (通訊連接埠編號)

參數

通訊連接埠編號

指定RS-232C連接埠編號的整數值。

- SPEL+控制部分：1～4
- PC部分：1001～1008

傳回值

用於傳回接收字元數(整數值)。

沒有接收資料時，則以下列負值傳回連接埠的狀態。

- -2：其他任務正在使用連接埠
- -3：連接埠未處於開啟狀態

參照

CloseCom、OpenCom、Read、Write

ChkCom函數範例

```
Integer numChars

numChars = ChkCom(1)
```

3.7.9 ChkNet函數

用於傳回網路連接埠接收緩衝區內的字元數。

格式

ChkNet (通訊連接埠編號)

參數

通訊連接埠編號
　　指定TCP/IP的連接埠編號(201~216)。

傳回值

用於傳回接收字元數(整數值)。

沒有接收資料時，則以下列負值傳回連接埠的狀態。

- -1：雖然連接埠處於開啟狀態，但尚未建立通訊
- -2：其他任務正在使用連接埠
- -3：連接埠未處於開啟狀態

參照

CloseNet、OpenNet、Read、Write

ChkNet函數範例

```
Integer numChars

numChars = ChkNet(201)
```

3.7.10 Chr$函數

用於傳回相對於ASCII代碼的字元。

格式

Chr$ (代碼編號)

參數

代碼編號

指定1~255的整數值。

傳回值

用於傳回相對於指定代碼的字元。

說明

Chr$用於傳回相對於ASCII代碼的字元(1個字元)。設定1~255範圍外的參數時，會發生錯誤。

參照

Asc、InStr、Left$、Len、Mid$、Right$、Space$、Str$、Val

Chr$函數範例

在以下範例中，宣告字串變數並設定字串「ABC」。Chr$函數用於將ASCII代碼轉換為「A」、「B」、「C」。&H表示後續數值為十六進位。(&H41為十六進位的41)

```
Function Test
    String temp$
    temp$ = Chr$(&H41) + Chr$(&H42) + Chr$(&H43)
    Print "The value of temp = ", temp$
Fend
```

3.7.11 ClearHistory

刪除所有命令的執行歷程記錄。

格式

ClearHistory

說明

刪除儲存的所有命令的執行歷程記錄。

注意

此命令僅可在命令視窗中執行。

參照

History

ClearHistory範例

```
> Motor On
> Go P1
> History
Motor On
Go P1

> ClearHistory
> History
```

3.7.12 ClearPoints

用於刪除機器人記憶體中的位置資料。

格式

ClearPoints

說明

ClearPoints用於對機器人記憶體中的位置資料區域之資料進行初始化。在教導新位置前，此命令用於刪除記憶體中的位置資料。

參照

Plist、LoadPoints、Savepoints

ClearPoints範例

下列範例是在命令視窗中使用ClearPoints命令的簡易範例。請在執行Clearpoints命令後執行Plist命令，確認不存在教導點。

```
>P1=100,200,-20,0/R
>P2=0,300,0,20/L
>plist
P1=100,200,-20,0/R
P2=0,300,0,20/L
>clearpoints
>Plist
>
```

3.7.13 Close

關閉以AOpen、BOpen、ROpen、UOpen、WOpen命令開啟的檔案。

格式

Close #檔案編號

參數

檔案編號

以30~63的整數值或運算式來指定。

說明

使用檔案編號關閉目前參照的檔案，並開放檔案編號。

參照

AOpen、BOpen、Flush、FreeFile、Input #、Print #、ROpen、UOpen、WOpen

Close範例

以下是開啟檔案以寫入資料，然後再度開啟相同檔案，將資料載入陣列中的範例。

```
Integer fileNumber, i, j

fileNumber = FreeFile
WOpen "TEST.DAT" As #fileNum
For i = 0 To 100
    Print #fileNum, i
Next i
Close #fileNum

FileNum = FreeFile
ROpen "TEST.DAT" As #fileNum
For i = 0 to 100
    Input #fileNum, j
    Print j
Next i
Close #fileNum
```

3.7.14 CloseCom

關閉以OpenCom開啟的RS-232C連接埠。

格式

CloseCom #通訊連接埠編號 | All

參數

通訊連接埠編號

指定要關閉的RS-232C連接埠編號。

- SPEL+控制部分：1～4
- PC部分：1001～1008

若指定All，則關閉已開啟相應工作的所有RS-232C連接埠。

參照

ChkCom、OpenCom

CloseCom範例

```
CloseCom #1
```

3.7.15 CloseDB

用於關閉以OpenDB命令開啟的資料庫，並釋放資料庫編號。

格式

CloseDB #資料庫編號

參數

資料庫編號

用於指定以OpenDB指定的資料庫編號(501~508的整數值)。

說明

關閉資料庫和Excel活頁簿，並開放資料庫編號。

注意

需連接有安裝RC+的PC。

參照

OpenDB、SelectDB、UpdateDB、DeleteDB、Input #、Print #

CloseDB範例

請參閱OpenDB範例。

3.7.16 CloseNet

用於關閉以OpenNet開啟的TCP/IP連接埠。

格式

CloseNet #通訊連接埠編號|All

參數

通訊連接埠編號

指定要關閉的TCP/IP連接埠編號(201~216)。
若指定All，則關閉已開啟相應工作的所有TCP/IP連接埠。

參照

ChkNet、OpenNet

CloseNet範例

```
CloseNet #201
```

3.7.17 Cls

用於清除Epson RC+的Run視窗、操作員視窗、命令視窗的文字區域。

用於清除TP列印面板。

格式
(1) Cls #裝置ID
(2) Cls

參數

設備ID

- 21 RC+
- 20 TP4

省略時，以顯示裝置為對象。

說明
若利用Epson RC+的命令視窗中程式執行Cls，則清除命令視窗的文字區域。

已在程式中執行時，則清除以裝置ID指定的裝置之畫面。

省略裝置ID時，則清除顯示裝置的畫面。

Cls範例
若利用Run視窗或操作員視窗執行此程式範例，則清除文字區域。

```
Function main
    Integer i

    Do
        For i = 1 To 10
            Print i
        Next i
        Wait 3
        Cls
    Loop
Fend
```

3.7.18 Cnv_AbortTrack

用於中斷對輸送帶佇列點的追蹤動作。

格式

Cnv_AbortTrack [停止Z高度]

參數

停止Z高度

以實數值指定停止時的機器人Z位置。可省略。

說明

在執行對輸送帶佇列點(與輸送帶動作同步的點)的動作命令時，Cnv_AbortTrack用於停止動作命令。

若指定參數的停止Z高度，只在指定值高於目前Z位置時閃避到指定值之後，使追蹤動作減速並停止。

一旦省略參數的停止Z高度，機械手就會使追蹤動作減速並停止，並不執行Z方向的閃避動作。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_RobotConveyor

Cnv_AbortTrack範例

```
' 對追蹤的工件被輸送到下游輸送帶上的機器人進行監視的工作

Function WatchDownstream

  Robot 1
  Do
    If g_TrackInCycle And Cnv_QueLen(1, CNV_QUELEN_DOWNSTREAM) > 0 Then
      ' 停止目前機器人的追蹤動作，將z軸移至0
      g_AbortTrackInCycle = TRUE
      Cnv_AbortTrack 0
      g_AbortTrackInCycle = FALSE
    EndIf
    Wait 0.01
  Loop
Fend
```

3.7.19 Cnv_Accel

設定輸送帶追蹤動作前的加減速度。

格式

Cnv_Accel 輸送帶編號 , 加速度・減速度

參數

傳送帶編號
　　以整數值(1~16)指定傳送帶編號。
加速度、減速度
　　用於設定追蹤動作的加速度、減速度。

說明

設定輸送帶追蹤動作的加速度/減速度。

不可分別設定加速度和減速度。

發生加速度命令錯誤或縮短抓取工件時間時，請進行變更。初始值為2000[mm/sec^2]。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Accel函數

Cnv_Accel範例

```
Cnv_Accel 1,2000
```

3.7.20 Cnv_Accel函數

傳回輸送帶追蹤動作前的加減速度。

格式

Cnv_Accel (輸送帶編號)

參數

傳送帶編號

以運算式或數值(1~16)指定傳送帶編號。

傳回值

以實數值(單位：mm)傳回。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Accel

Cnv_Accel函數範例

```
Print Cnv_Accel (1)
```

3.7.21 Cnv_AccelLim

設定輸送帶追蹤後的加速度、減速度設定值。

格式

Cnv_AccelLim 輸送帶編號, 模式編號, 加速度・減速度

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

模式編號

以整數值(0～2)指定傳送帶跟蹤的跟蹤模式。

加速度、減速度

以實數值(單位：mm/sec^2)設定傳送帶跟蹤後的加速度、減速度限制值。設定範圍與AccelS命令相同。

初始值

- 個數優先模式：500 [mm/sec^2]
- 精度優先模式：2000 [mm/sec^2]
- 變速傳送帶支援模式：6000 [mm/sec^2]

說明

輸送帶追蹤時，反覆停止與繼續執行輸送帶運作，會發生機器人對輸送帶速度變化的追蹤延遲。若要提高追蹤性能，可設定加速度、減速度的極限。

無法分別設定加速度與減速度。

過度提高極限時，會因為輸送帶的速度不均與雜訊等使得機器人的動作產生振動。此外，過度調低極限時，即使停止輸送帶，機器人也有可能不會停止追蹤，使得機器人往動作區域外作動。此時請設定追蹤停止線，或使用程式在下游極限停止追蹤。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_AccelLim函數

Cnv_AccelLim範例

```
Cnv_AccelLim 1,2,7000
```

3.7.22 Cnv_AccelLim函數

傳回輸送帶追蹤後的加速度、減速度設定值。

格式

Cnv_AccelLim (輸送帶編號 , 模式編號)

參數

傳送帶編號
　　以整數值(1~16)指定傳送帶編號。
模式編號
　　以整數值(0～2)指定傳送帶跟蹤的跟蹤模式。

傳回值

以實數值(單位：mm/sec^2)回傳。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_AccelLim

Cnv_AccelLim函數範例

```
Print Cnv_AccelLim (1,2)
```

3.7.23 Cnv_Adjust

用於設定是否執行取得輸送帶追蹤的追蹤延遲補償值之動作。

格式

Cnv_Adjust 輸送帶編號, On | Off

參數

傳送帶編號

以運算式或數值(1~16)指定傳送帶編號。

On | Off

設為執行讀取傳送帶跟蹤的跟蹤延遲補償值之動作時，則指定「On」；設為不執行讀取動作時，則指定「Off」。

說明

用於設定是否執行取得輸送帶追蹤的追蹤延遲補償值之動作。

將Cnv_Adjust設為「On」並執行Cnv_QueGet函數，將執行取得補償值之動作。抓取工件時，則將Cnv_Adjust設為「Off」，並執行Cnv_QueGet函數。

若將Cnv_Adjust設為「On」，請務必於取得補償值後設為「Off」。

Cnv_Adjust僅可使用於直線輸送帶。不支援圓形輸送帶。若為圓形輸送帶，即便設為「On」也無法取得補償值。

取得之補償值將於控制器的電源關閉後被清除，控制器的電源開啟時則被設為初始值「0」。因此，取得補償值後請以print Cnv_AdjustGet傳回補償值，並於執行程式的Cnv_QueGet前，以Cnv_AdjustSet設定補償值。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_AdjustGet函數、Cnv_AdjustSet、Cnv_AdjustClear、Cnv_QueGet函數

Cnv_Adjust範例

```
Cnv_Adjust 1, On
Jump Cnv_QueGet(1)
 ·
 ·
Cnv_Adjust 1, Off
```

3.7.24 Cnv_AdjustClear

用於刪去輸送帶追蹤的追蹤延遲補償值。

格式

Cnv_AdjustClear 輸送帶編號

參數

傳送帶編號

以運算式或數值(1~16)指定傳送帶編號。

說明

清除輸送帶追蹤的追蹤延遲補償取得動作的結果、補償量，並設為「0」。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Adjust、Cnv_AdjustSet函數、Cnv_AdjustGet、Cnv_QueGet函數

Cnv_AdjustClear範例

```
Cnv_AdjustClear 1
```

3.7.25 Cnv_AdjustGet函數

用於傳回輸送帶追蹤的追蹤延遲補償值。

格式

Cnv_AdjustGet (輸送帶編號, 模式編號)

參數

傳送帶編號

以運算式或數值(1~16)指定傳送帶編號。

模式編號

- 0：補償讀取動作的結果
- 1：補償量

傳回值

- 模式編號0：傳回實數值0~2。
  - 0：未執行補償值讀取動作
  - 1：補償值讀取完畢
  - 2：補償值讀取失敗
- 模式編號1：傳回實數值(單位：mm)。

說明

若未使用Cnv_Adjust及Cnv_QueGet函數取得輸送帶追蹤的追蹤延遲補償值，則模式編號0~2的傳回值皆傳回「0」。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Adjust、Cnv_AdjustSet、Cnv_AdjustClear、Cnv_QueGet函數

Cnv_AdjustGet函數範例

```
Print Cnv_AdjustGet(1, 1)
```

3.7.26 Cnv_AdjustSet

用於設定輸送帶追蹤之追蹤延遲補償值。

格式

Cnv_AdjustSet 輸送帶編號, 補償量

參數

傳送帶編號
　　以整數值(1~16)指定傳送帶編號。
補償量
　　以實數值(單位：mm)指定補償量。

說明

若不執行Cnv_AdjustSet，補償量將適用上一次的設定值。

若於控制器電源開啟後，一次也沒有執行取得補償值的動作，則補償量將設為初始值的「0」。

Cnv_Mode命令的模式編號設定為1：僅在精度優先模式下，啟用Cnv_AdjustSet命令的設定。

Cnv_AdjustSet不支援圓形輸送帶。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Adjust、Cnv_AdjustGet、Cnv_AdjustClear、Cnv_QueGet函數

Cnv_AdjustSet範例

```
Cnv_AdjustSet 1, 4.5
```

3.7.27 Cnv_Downstream

用於設定指定輸送帶的下游極限。

格式

Cnv_Downstream輸送帶編號 , 下游極限

參數

傳送帶編號

以運算式或數值(1~16)指定傳送帶編號。

下游範圍

用於設定追蹤開始區域下游側的邊界

說明

可變更以校正精靈設定的下游極限。使用傾斜下游極限時，不可以Cnv_Downstream變更下游極限值。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Upstream

Cnv_Downstream範例

```
Cnv_Downstream 1,500
```

3.7.28 Cnv_Downstream函數

用於傳回指定輸送帶的下游極限設定值。

格式

Cnv_Downstream (輸送帶編號)

參數

傳送帶編號

以運算式或數值(1~16)指定傳送帶編號。

傳回值

- 直線輸送帶：以實數值(單位：mm)傳回。
- 圓形輸送帶：以實數值(單位：deg.)傳回。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Upstream

Cnv_Downstream函數範例

```
Print "Downstream limit: ", Cnv_Downstream(1)
```

3.7.29 Cnv_Fine

用於設定和顯示對於指定輸送帶的完成追蹤判斷範圍。

格式

Cnv_Fine 輸送帶編號 [, Fine設定值]

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

Fine設定值

以表示判斷完成追蹤距離的實數值(單位：mm)進行指定。

若將值設為0，則不使用Cnv_Fine。省略Fine設定值後，顯示目前的Cnv_Fine設定。

說明

判斷完成追蹤輸送帶，並指定與工件之間的距離，以接收下一個命令。若設為「0」，則可在不使用Cnv_Fine的情況下，在完成動作命令時接收下一個命令。預設值為「0」，在下一個時序自動設定預設值。

- 定義輸送帶時
- 啟動控制器時

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Fine函數

Cnv_Fine範例

```
Cnv_Fine 1, 5
```

3.7.30 Cnv_Fine函數

用於傳回指定輸送帶的完成追蹤判斷範圍的設定。

格式

Cnv_Fine (輸送帶編號)

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

傳回值

以實數值(單位：mm)傳回Cnv_Fine設定。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Fine

Cnv_Fine函數範例

```
Real f

f = Cnv_Fine(1)
```

3.7.31 Cnv_Flag函數

用於傳回對追蹤停止線的追蹤狀態。

格式

Cnv_Flag (輸送帶編號)

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

傳回值

- 0：追蹤動作已正常執行。(未遭取消或停止)
- 1：因預測工件將超過跟蹤中止線，故取消執行跟蹤動作。
- 2：因工件超過跟蹤中止線，故中止跟蹤動作。Z位置未降至指定高度。
- 3：因工件超過跟蹤中止線，故中止跟蹤動作。Z位置已降至指定高度。
- 4：因工件位於跟蹤開始區域外，故取消執行跟蹤動作。

唯有設定追蹤停止線時，才傳回0以外的值。

關於追蹤停止線的詳細內容，請參閱以下手冊。
「Epson RC+ 使用者指南」

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

Cnv_Flag函數範例

```
Print Cnv_Flag (1)
```

3.7.32 Cnv_LPulse函數

用於傳回以輸送帶觸發閂鎖的脈衝。

格式

Cnv_LPulse (輸送帶編號)

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

說明

傳回以硬體觸發配線或Cnv_Trigger閂鎖的最新輸送帶脈衝。

傳回值

以Long數值傳回指定輸送帶被閂鎖的脈衝。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Trigger、Cnv_Pulse

Cnv_LPulse函數範例

```
Print "Latched conveyor position: ", Cnv_LPulse(1)
```

3.7.33 Cnv_Mode

設定輸送帶的追蹤模式。

格式
Cnv_Mode (輸送帶編號 ,模式編號)

參數

傳送帶編號
　　以整數值(1~16)指定傳送帶編號。
模式編號
- 0：個數優先模式
- 1：精度優先模式
- 2：變速傳送帶支援模式

說明
設定輸送帶的追蹤模式。

僅限於直線輸送帶使用Cnv_Mode。

請在進行追蹤動作前設定追蹤模式。未設定時，則設定個數優先模式。

- 個數優先模式：雖然精度不如精度優先模式，但追蹤所需時間較短，因此，這種模式適用於工件間隔小的傳送帶系統或輸送帶速度較快的傳送帶系統。
- 精度優先模式：追蹤所需時間比個數優先模式長，但精度提升。因此，這種模式適用於使用小型工件的輸送帶系統。
- 變速傳送帶支援模式：此模式適用於正在接觸工件的作業途中，需停止或啟動傳送帶的傳送帶系統。

圓形輸送帶所支援的模式編號僅為「0」。設定「1」和「2」時，機器人會進行與模式編號「0」相同的動作。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。
此外，動作中使用這項命令，無法變更輸送帶追蹤的模式。

參照
Cnv_Mode函數

Cnv_Mode範例

```
Cnv_Mode 1, 1
```

3.7.34 Cnv_Mode函數

用於傳回輸送帶追蹤模式。

格式

Cnv_Mode (輸送帶編號)

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

傳回值

傳回實數值0~2。

- 0：個數優先模式
- 1：精度優先模式
- 2：變速傳送帶支援模式

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Mode

Cnv_Mode函數範例

```
Print Cnv_Mode (1)
```

3.7.35 Cnv_Name$函數

用於傳回指定輸送帶的名稱。

格式

Cnv_Name$ (輸送帶編號)

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

傳回值

以字串傳回輸送帶名稱。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Number

Cnv_Name$函數範例

```
Print "Conveyor 1 Name: ", Cnv_Name$(1)
```

3.7.36 Cnv_Number函數

用於傳回指定輸送帶名稱的編號。

格式

Cnv_Number (輸送帶名稱)

參數

傳送帶名稱
以字串指定傳送帶名稱。

傳回值

以整數值傳回輸送帶編號。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Name$

Cnv_Number函數範例

```
Integer cnvNum

cnvNum = Cnv_Number("Main Conveyor")
```

3.7.37 Cnv_OffsetAngle

設定輸送帶佇列資料的位移值。

格式

Cnv_OffsetAngle 輸送帶編號 [, 位移值]

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

偏移值

以實數值(單位：degree)指定傳送帶佇列資料的偏移值。可省略。若省略，則顯示目前位移值。

說明

設定輸送帶佇列資料的位移值。

僅限於圓形輸送帶使用Cnv_OffsetAngle。

輸送帶追蹤可能會因使用的輸送帶速度而發生追蹤延遲。發生追蹤延遲時，機器人對追蹤延遲量的偏移位置進行處理。Cnv_OffsetAngle用於透過對佇列賦予位移值，以修正該偏移。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_OffsetAngle函數

Cnv_OffsetAngle範例

```
Cnv_OffsetAngle 1, 5
```

3.7.38 Cnv_OffsetAngle函數

用於傳回輸送帶佇列資料的位移值。

格式

Cnv_OffsetAngle (輸送帶編號)

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

傳回值

用於傳回實數值(單位：degree)。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_OffsetAngle

Cnv_OffsetAngle函數範例

```
Real offsetAngle

offsetAngle = Cnv_OffsetAngle (1)
```

3.7.39 Cnv_Point函數

用於將感測器坐標值轉換為輸送帶坐標值並傳回。

格式

Cnv_Point (輸送帶編號, 感測器X坐標值, 感測器Y坐標值 [, 感測器U坐標值])

參數

傳送帶編號
　　以整數值(1~16)指定傳送帶編號。
感測器X坐標值
　　以實數值指定感測器X坐標值。
感測器Y坐標值
　　以實數值指定感測器Y坐標值。
感測器U坐標值
　　以實數值指定感測器U坐標值。可省略。

傳回值

用於將輸送帶坐標值作為點資料後傳回。

說明

Cnv_Point函數用於將作為輸送帶佇列新增的坐標值，作為點資料進行新增。視覺系統輸送帶的感測器X坐標值、感測器Y坐標值是來自相機的視覺坐標值。感測器輸送帶的感測器位置為輸送帶坐標系原點。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Speed

Cnv_Point函數範例

```
Boolean found
Integer i, numFound
Real x, y, u

Cnv_Trigger 1
VRun FindParts
VGet FindParts.Part.NumberFound, numFound
For i = 1 To numFound
  VGet FindParts.Part.CameraXYU(i), found, x, y, u
  Cnv_QueAdd 1, Cnv_Point(1, x, y)
Next i
```

3.7.40 Cnv_PosErr函數

用於傳回目前追蹤位置與目標之間的位置偏差。

格式

Cnv_PosErr (輸送帶編號)

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

傳回值

用於傳回實數值(單位：mm)。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_MakePoint

Cnv_PosErr函數範例

```
Print "Conveyor 1 position error: ", Cnv_PosErr(1)
```

3.7.41 Cnv_PosErrOffset

用於設定目前的追蹤位置與目標之間的位置偏差之補償值。

格式

Cnv_PosErrOffset 輸送帶編號, 補償值

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

補償值

以實數值(0～255，單位：msec)指定預測傳送帶速度的時間。

說明

若在輸送帶追蹤中反復停止及運作輸送帶，將會因機器人對輸送帶速度變化產生的追蹤延遲，導致追蹤位置與目標之間的位置偏差惡化。

藉由預測經過補償值所設時間後的輸送帶速度，再進行輸送帶追蹤，即可改善位置偏差。

僅在變速輸送帶支援模式下，可使用Cnv_PosErrOffset。若為個數優先模式及精度優先模式，則即使設定補償值亦無法改善位置偏差。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Mode、Cnv_PosErr

Cnv_PosErrOffset範例

```
Cnv_Mode 1, 2     ' 變速輸送帶支援模式
Cnv_PosErrOffset 1, 10     ' 補償值10msec
```

3.7.42 Cnv_Pulse函數

用於傳回輸送帶目前位置的脈衝。

格式

Cnv_Pulse (輸送帶編號)

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

傳回值

以Long數值傳回指定輸送帶的脈衝。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Trigger, Cnv_LPulse

Cnv_Pulse函數範例

```
Print "Current conveyor position: ", Cnv_Pulse(1)
```

3.7.43 Cnv_QueAdd

用於在輸送帶佇列資料中新增點資料。

格式

Cnv_QueAdd 輸送帶編號, 點資料 [, 使用者資料]

參數

傳送帶編號
　　以整數值(1~16)指定傳送帶編號。
點數據
　　指定在傳送帶佇列資料中增加的點數據。
用戶資料
　　以實數值指定和點數據一起登錄的用戶資料。可省略。

說明

在已指定輸送帶佇列資料的末尾新增點資料。也一併註冊目前處於閂鎖狀態的輸送帶和脈衝位置。

若有小於以Cnv_QueReject指定距離的佇列資料，則不註冊資料，也不會發生錯誤。

佇列資料上限值為1000。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_RobotConveyor

Cnv_QueAdd範例

```
Boolean found
Integer i, numFound
Real x, y, u

Cnv_Trigger 1
VRun FindParts
VGet FindParts.Part.NumberFound, numFound
For i = 1 To numFound
  VGet FindParts.Part.CameraXYU(i), found, x, y, u
  Cnv_QueAdd 1, Cnv_Point(1, x, y)
Next i
```

3.7.44 Cnv_QueGet函數

用於從指定的輸送帶佇列資料傳回點資料。

格式

Cnv_QueGet (輸送帶編號 [, 指數])

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

指數

以整數值指定要讀取的佇列資料指數。(開頭指數編號為0)可省略。

傳回值

用於從指定的輸送帶佇列傳回點資料。

說明

Cnv_QueGet用於從輸送帶佇列取得點資料。有省略指數時，則傳回佇列資料的開頭資料。有設定指數時，則傳回設定指數的點資料。

Cnv_QueGet不會用於從輸送帶佇列刪除點資料。使用Cnv_QueRemove進行刪除。

想要在運作輸送帶的同時追蹤工件時，需要將Cnv_QueGet加入動作命令。

例：

```
Jump Cnv_QueGet(1) ' 追蹤工件
```

不可將Cnv_QueGet的傳回值指派給點，讓機器人在該點位置進行運作以追蹤工件。

```
P1 = Cnv_QueGet(1)
Jump P1  ' 不追蹤工件
```

若將Cnv_QueGet的傳回值指派給點，該坐標值則成為執行命令時的工件位置。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_QueLen, Cnv_QueRemove

Cnv_QueGet函數範例

```
' Jump到佇列開頭以進行追蹤
Jump Cnv_QueGet(1)
On gripper
Wait .1
Jump place
Off gripper
Wait .1
Cnv_QueRemove 1
```

3.7.45 Cnv_QueLen函數

用於傳回指定輸送帶佇列的資料數。

格式

Cnv_QueLen (輸送帶編號 [, 參數編號])

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

參數編號

以整數值指定要對佇列資料進行計數的區域。可省略。

常數	值	內容
CNV_QUELEN_ALL	0	用於傳回佇列資料的總數。
CNV_QUELEN_UPSTREAM	1	用於從追蹤開始區域傳回上游的佇列資料數。
CNV_QUELEN_PICKUPAREA	2	用於傳回追蹤開始區域內的佇列資料數。
CNV_QUELEN_DOWNSTREAM	3	用於從追蹤開始區域傳回下游的佇列資料數。

傳回值

以整數值傳回資料數。

說明

傳回有效的佇列資料數。這對於取得追蹤開始區域內的資料數十分有效。

還可用作Wait命令的引數。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_QueGet

Cnv_QueLen函數範例

```
Do
  Do While Cnv_QueLen(1, CNV_QUELEN_DOWNSTREAM) > 0
    Cnv_QueRemove 1, 0
  Loop
  If Cnv_QueLen(1, CNV_QUELEN_PICKUPAREA) > 0 Then
    Jump Cnv_QueGet(1, 0) C0
    On gripper
    Wait .1
    Cnv_QueRemove 1, 0
    Jump place
    Off gripper
    Jump idlePos
  EndIf
Loop
```

3.7.46 Cnv_QueList

用於顯示指定輸送帶佇列資料的一覽。

格式

Cnv_QueList 輸送帶編號 [, 顯示數]

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

顯示數

以整數值指定要顯示的資料數量。可省略。若省略，則顯示所有佇列資料。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_QueGet

Cnv_QueList範例

```
Cnv_QueList 1
```

3.7.47 Cnv_QueMove

用於將上游輸送帶的佇列資料移至下游輸送帶佇列。

格式

Cnv_QueMove 輸送帶編號 [, 指數] [, 使用者資料]

參數

傳送帶編號
　　以整數值(1~16)指定傳送帶編號。
指數
　　以整數值指定要移動的佇列資料的指數。(開頭指數編號為0 。)可省略。
用戶資料
　　以實數值指定和資料一起登錄的用戶資料。可省略。

說明

將佇列資料移至下游輸送帶的佇列。已省略指數時，則移動開頭(指數為0)的佇列資料。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_QueGet

Cnv_QueMove範例

```
Cnv_QueMove 1
```

3.7.48 Cnv_QueReject

設定和顯示用於防止佇列重複註冊的最小距離。

格式

Cnv_QueReject 輸送帶編號 [, 防止重複註冊的距離]

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

拒絕距離

以實數值(單位：mm)指定用於防止重複註冊的距離最小值。若指定為負值，則設定0 mm。可省略。若有省略，則顯示用於防止重複註冊的目前距離。

說明

設定用於防止佇列重複註冊的最小距離。即使被視覺系統掃描並檢測多次，也僅註冊1次。Cnv_QueReject是用於防止重複註冊的系統篩選器。預設值為0 mm。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_QueReject函數

Cnv_QueReject範例

```
Cnv_QueReject 1, 20
```

3.7.49 Cnv_QueReject函數

傳回用於防止重複註冊佇列的距離。

格式

Cnv_QueReject (輸送帶編號)

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

傳回值

用於傳回實數值(單位：mm)。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_QueReject

Cnv_QueReject函數範例

```
Real rejectDist

RejectDist = Cnv_QueReject(1)
```

3.7.50 Cnv_QueRemove

用於從輸送帶佇列資料刪除佇列資料。

格式

Cnv_QueRemove 輸送帶編號 [, 指數 | All]

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

指數

以整數值指定要刪除的開頭指數，或以All指定刪除所有指數。可省略。

說明

從輸送帶佇列資料刪除1筆以上的佇列資料。結束使用佇列資料時，則刪除佇列資料。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_QueAdd

Cnv_QueRemove範例

```
Jump Cnv_QueGet(1)
On gripper
Wait .1
Jump place
Off gripper
Wait .1

' 從輸送帶刪除資料
Cnv_QueRemove 1
```

3.7.51 Cnv_QueUserData

用於設定和顯示佇列項目相關使用者資料。

格式

Cnv_QueUserData 輸送帶編號 [, 指數] [, 使用者資料]

參數

傳送帶編號
　　以整數值(1~16)指定傳送帶編號。
指數
　　以整數值指定佇列資料的指數。可省略。
用戶資料
　　以實數值指定用戶資料。可省略。

說明

設定輸送帶佇列的使用者資料。可省略參數的使用者資料，但平時操作時需要用到。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_QueUserData函數

Cnv_QueUserData範例

```
Cnv_QueUserData 1, 1, angle
```

3.7.52 Cnv_QueUserData函數

用於傳回佇列項目相關使用者資料。

格式

Cnv_QueUserData (輸送帶編號 [, 指數])

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

指數

以整數值指定佇列資料的指數。可省略。

傳回值

用於傳回實數值。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_QueUserData

Cnv_QueUserData函數範例

```
' 新增於佇列中
Cnv_QueAdd 1, Cnv_Point(1, x, y), angle

' 從佇列刪除
angle = Cnv_QueUserData(1)  ' 預設指數變成「0」
Jump Cnv_QueGet(1) :U(angle)
Cnv_QueRemove 1
```

3.7.53 Cnv_RobotConveyor函數

用於傳回處於追蹤狀態的輸送帶之編號。

格式

Cnv_RobotConveyor [(機械手編號)]

參數

機械手編號

以整數值指定機械手編號。

傳回值

以整數值傳回輸送帶編號。若沒有要追蹤的輸送帶，則傳回0。

說明

正在使用多台機械手時，用於確認機械手正在追蹤的輸送帶。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_MakePoint

Cnv_RobotConveyor函數範例

```
Integer cnvNum

cnvNum = Cnv_RobotConveyor(1)
```

3.7.54 Cnv_Speed函數

用於傳回輸送帶的動作速度。

格式

Cnv_Speed (輸送帶編號)

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

傳回值

用於傳回實數值(單位：mm/s)。在圓形輸送帶的情況下，則傳回實數值(單位：degree/s)。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Pulse

Cnv_Speed函數範例

```
Print "Conveyor speed: ", Cnv_Speed(1)
```

3.7.55 Cnv_Trigger

用於為下一個Cnv_QueAdd陳述式而對輸送帶的目前位置進行閂鎖。

格式

Cnv_Trigger 輸送帶編號

參數

傳送帶編號

以整數值(1~16)指定傳送帶編號。

說明

在指定輸送帶編碼器的脈衝輸出基板上未進行硬體觸發配線時使用本函數。Cnv_Trigger則變成軟體觸發器。
此外，VRun命令的處理時間可能會發生不平均。執行VRun命令後，請使用Cnv_Trigger命令。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_QueAdd

Cnv_Trigger範例

```
Boolean found
Integer i, numFound
Real x, y, u

VRun FindParts
Cnv_Trigger 1
VGet FindParts.Part.NumberFound, numFound
For i = 1 To numFound
  VGet FindParts.Part.CameraXYU(i), found, x, y, u
  Cnv_QueAdd 1, Cnv_Point(1, x, y)
Next i
```

3.7.56 Cnv_Upstream

用於設定指定輸送帶的上游極限。

格式

Cnv_Upstream 輸送帶編號 , 上游極限

參數

傳送帶編號

以運算式或數值(1~16)指定傳送帶編號。

上游範圍

用於設定追蹤開始區域上游側的邊界。

說明

可變更以校正精靈設定的上游極限。使用傾斜上游極限時，不可以Cnv_Upstream變更上游極限值。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Downstream

Cnv_Upstream範例

```
Cnv_Upstream 1,200
```

3.7.57 Cnv_Upstream函數

用於傳回指定輸送帶的上游極限設定值。

格式

Cnv_Upstream (輸送帶編號)

參數

傳送帶編號
以整數值(1~16)指定傳送帶編號。

傳回值

- 直線輸送帶：以實數值(單位：mm)傳回。
- 圓形輸送帶：以實數值(單位：deg.)傳回。

注意

僅限於裝有輸送帶追蹤選項時，方可使用此命令。

參照

Cnv_Downstream

Cnv_Upstream函數範例

```
Print "Upstream limit: ", Cnv_Upstream(1)
```

3.7.58 CollisionDetect

用於啟用或停用目前機器人的碰撞檢測功能(機器人動作異常檢測功能)。

格式

(1) CollisionDetect 狀態

(2) CollisionDetect 狀態, 關節編號

(3) CollisionDetect

參數

狀態

- On：啟用碰撞檢測(機器人動作異常檢測)。
- Off：停用碰撞檢測(機器人動作異常檢測)。

關節編號

使用SCARA機器人(包括RS系列)時，指定1~4的關節編號；使用垂直六軸機器人(包括N系列)時，則指定1~6的關節編號。

結果

若省略參數，則顯示目前的CollisionDetect狀態。

說明

以運作機器人的期望速度和實際速度之差(速度偏差)來檢測機器人的動作異常。可以本功能檢測的異常分為A和B兩類。

- A：發生機器人手臂或夾具的碰撞或接觸
- B：發生碰撞或接觸以外的機器人動作異常

B類異常又可依功率狀態而進行如下分類。

- 高功率狀態下的異常：因Weight或Inertia的設定過小而導致扭矩飽和
  - 因多關節軸的複合動作或擺動長尺寸物體而導致扭矩飽和
  - 因電源電壓降低而導致扭矩飽和
  - 因硬體異常或軟體誤操作而導致異常動作
- 低功率時的異常：
  - 因硬體異常或軟體誤操作而導致異常動作
  - 因超過規格的抓手重量或保持長尺寸物體而在低功率時導致扭矩飽和

Epson RC+ 7.0 Ver.7.2以後版本支援的通用機器人(垂直6軸型機器人、SCARA機器人)備有碰撞檢測功能。若在連接未支援的機器人(X5系列等)時使用本命令，則會發生錯誤。

執行本命令需要處理時間。被要求有週期時間時，請最小限度地使用本命令。

可以命令開啟/關閉所有軸及各軸。預設為開啟所有軸。(韌體為Ver7.2.1.x 以後版本時，則開啟；Ver7.2.0.x以前版本時，則關閉)

若關閉控制器電源，則返回預設值。除此之外，除非以本命令進行明確設定，否則不會改變狀態。

碰撞檢測時會輸出以下訊息，並停止機器人。

- 錯誤5057「在高功率狀態下檢測到碰撞(檢測到機器人動作異常)」
- 錯誤5058「在低功率狀態下檢測到碰撞(檢測到機器人動作異常)」

若要降低高功率模式時的損害，可併用以LimitTorque命令限制扭矩的功能；若要降低低功率模式時的損害，可併用以LimitTorqueLP命令限制扭矩的功能。使用這些功能是很有效的。

請參照以下的說明。
「Epson RC+ 使用者指南 - 碰撞檢測功能(機器人動作異常檢測功能)」

參照

LimitTorque、LimitTorque函數、LimitTorqueLP、LimitTorqueLP函數

CollisionDetect範例

```
CollisionDetect On          ' 將所有軸碰撞檢測設為On
CollisionDetect Off, 5      ' 只將第5關節設為OFF
CollisionDetect             ' 顯示on, on, on, on, off, on。
```

3.7.59 CollisionDetect函數

用於傳回CollisionDetect命令的設定值。

格式

CollisionDetect (關節編號)

參數

關節編號

以1~6的整數進行指定。

傳回值

以整數值傳回CollsionDetect命令的設定值。

- 1 = ON
- 0 = OFF

參照

CollisionDetect

CollisionDetect函數範例

```
Print CollisionDetect (1)  '顯示第1關節的CollisionDetect值
```

3.7.60 Cont

用於重新開啟處於暫停狀態的控制器，繼續執行所有工作。

本命令為適合高階專業人員使用的命令。請充分理解命令內容後使用。

格式

Cont

說明

若要在程式中執行本命令，需勾選Epson RC+的[設置] - [系統配置] - [控制器] - [選項]的[啟動進階任務命令]的核取方塊。即便完成此設定，也無法在以Trap SGClose啟動的工作中執行Cont命令。

Cont命令用於執行Pause陳述式，或重新開啟以安全門打開輸入處於暫停狀態的控制器，繼續執行所有工作。具有同等於[操作員視窗]的[繼續執行]按鈕或遠端輸入的「Continue」功能。

在WaitRecover狀態(打開安全門後的等待復歸狀態)下執行Cont命令時，在所有機器人進行勵磁和復歸動作之後，重新開始執行程式。

只想進行機器人的勵磁和復歸動作時，請使用Recover命令。

> ⚠ 注意
>
> 若要在程式中執行Cont命令，請理解命令內容，並確認可作為系統繼續執行的條件皆已備齊。若透過迴圈繼續執行命令等進行錯誤的使用，則有可能導致系統安全性下降。請充分注意。

參照

Pause, Recover, ContManualRecover

Cont範例

```
Function main
    Xqt 2, monitor, NoPause
    Do
        Jump P1
        Jump P2
    Loop
Fend

Function monitor
    Do
        If Sw(pswitch) = On then
            Pause
            Wait Sw(pswitch) = Off and Sw(cswitch) = On
            Cont
        EndIf
    Loop
Fend
```

3.7.61 ContManualRecover

安全門關閉後，透過 Recover命令完成復歸動作的狀態下，重新開始全部任務時使用。
本命令為適合高階專業人員使用的命令。請充分理解命令內容後使用。本命令中，重新開始時，必須對安全門開啟時的位置執行復歸動作 (Recover命令)。

格式

ContManualRecover

說明

若要在程式中執行本命令，需勾選Epson RC+的[設置] - [系統配置] - [控制器] - [選項]的[啟動進階任務命令]的核取方塊。
即便完成此設定，也無法在以Trap SGClose啟動的工作中執行ContManualRecover命令。

ContManualRecover命令，是因啟動安全門輸入而變成暫時停止狀態的控制器，透過Recover命令重新啟動，於機器人完成復歸動作到安全門啟動時的位置時使用。
具有等同遠端輸入的「ContinueManualRecover」的功能。

在WaitRecover狀態 (安全門關閉後的等待恢復狀態) 下執行ContManualRecover命令時，有別於Cont命令，會發生錯誤。
請事先透過Recover命令，進行機器人的勵磁及復歸動作後使用。

> ⚠ 注意
>
> 若要在程式中執行ContManualRecover命令，請理解命令內容，並確認可作為系統繼續執行的條件皆已備齊。若透過迴圈繼續執行命令等進行錯誤的使用，則有可能導致系統安全性下降。請充分注意。

參照

Pause, Recover, Cont, Recover函數

ContManualRecover範例

```
Function main
    Xqt 2, monitor, NoPause
    Do
        Jump P1
        Jump P2
    Loop
Fend

Function monitor
    Do
        Wait SafetyOn = True       '等待安全門啟動
        Wait SafetyOn = False      '等待安全門關閉
        If (CtrlInfo(1) and &H2000) = &H2000 Then
            Recover
            ContManualRecover
        EndIf
    Loop
Fend
```

3.7.62 Copy

用於複製檔案。

格式
Copy 複製來源, 目的地

參數

複製來源
　　指定要複製的檔案路徑和檔案名稱。詳細內容請參閱ChDisk。
複製目的地
　　指定要複製原始檔案的目的地路徑和要複製的檔案名稱。詳細內容請參閱ChDisk。

說明
將指定來源檔案複製到指定目的地的檔名上。

複製來源和目的地不可共享相同路徑名稱或檔名。

若已存在目的地，則發生錯誤。

注意

可使用網路路徑。

檔名不可使用「*」,「?」等萬用字元。

在命令視窗中使用時，可省略引號和逗號。

參照
ChDir、MkDir

Copy範例
命令視窗中的操作範例

```
> copy TEST.DAT TEST2.DAT


> Copy TEST.DAT c:     'NG
!!錯誤：7203 存取遭拒
> Copy TEST.DAT c:\      'OK
>
```

3.7.63 Cos函數

是用於傳回指定角度餘弦的函數。

格式
Cos (數值)

參數

數值
　　以數值(弧度)指定角度。

傳回值
用於傳回指定數值的餘弦值(實數值)。

說明
Cos用於傳回指定角度(弧度)的餘弦。請用弧度指定角度(數值)。傳回值範圍為-1~1。

以度賦予角度時，務必用DegToRad函數轉換為弧度。

參照
Abs、Atan、Atan2、Int、Mod、Not、Sgn、Sin、Sqr、Str$、Tan、Val

Cos函數範例
以下為使用Cos的簡易程式範例。

```
Function costest
  Real x
  Print "Please enter a value in radians"
  Input x
  Print "COS of ", x, " is ", Cos(x)
Fend
```

以下是在命令視窗中使用Cos的範例。

```
Display the cosine of 0.55:

>print cos(0.55)
 0.852524522059506
>

Display cosine of 30 degrees:

>print cos(DegToRad(30))
 0.866025403784439
>
```

3.7.64 CP

設定路徑運動。

格式
(1) CP { On | Off }
(2) 動作命令 目標坐標 CP

參數

On | Off

- On：啟用路徑運動。
- Off：解除路徑運動。

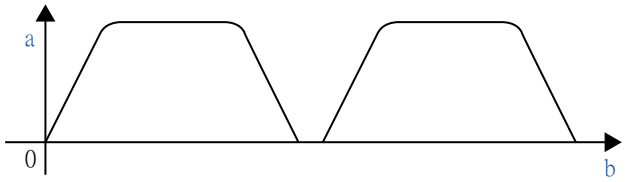
說明
以下列命令使用路徑運動。

Arc, Arc3, Go, Jump, Jump3, Jump3CP, JumpTLZ, Move

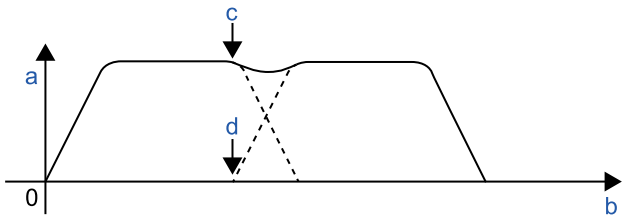
若將CP設定為On，無論是否以各動作命令指定CP參數，動作命令均用於在開始減速的同時執行下一個陳述式。這樣的話，若在減速動作時開始下一個動作，會合成動作軌跡。

若將CP設定為Off，只有在以各動作命令指定CP參數時，方可啟用本功能。

通常動作



路徑運動



符號	說明
a	速度
b	時間
c	開始減速
d	開始加速

透過將CP設定為On，合成CP動作(Arc, Arc3, Jump3, Jump3CP, JumpTLZ, Move)之間或PTP動作(Go, Jump)之間的動作軌跡。
然而，若為CP動作和PTP動作的連續軌跡，則在不合成動作軌跡的情況下先進行減速。
由控制器的環境設定而定，若要在CP On時連接CP動作與PTP動作，則將會合成CP動作與PTP動作的動作軌跡。依據動作加減速度的設定不同，可能會發生過速度錯誤或過加速度錯誤。屆時，請調整或停用加減速度。

此外，垂直6軸型機器人 (包括N系列) 中以手腕奇點為目標值的CP動作，不會合成下一個動作和動作軌跡，而會先進行減速。

若在啟用路徑運動時使用Here，則在指定Here的動作之前將不會合成動作軌跡便進行減速。若採用下列的使用方法，則以P1減速度停止後將開始進行下一個動作。

```
    Go P1 CP
    Go Here+X(100)
```

若不想停止要繼續動作時，請修正程式以便事先計算Here的位置。

```
    Go P1 CP
    Go P1+X(100)
```

以下情況時CP會處於Off狀態。

- 啟動控制器時
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

參照

CP函數、Arc、Arc3、Go、Jump、Jump3、Jump3CP、JumpTLZ、Move

CP範例

```
CP On
Move P1
Move P2
CP Off
```

3.7.65 CP函數

用於傳回路徑運動的狀態。

格式

CP

傳回值

- 0 = 路徑運動OFF
- 1 = 路徑運動ON

參照

CP

CP函數範例

```
If CP = Off Then
    Print "CP is off"
EndIf
```

3.7.66 CP_Offset

CP為On時，用於開始後續動作命令的位移時間。

格式

(1) CP_Offset [On [, OffsetTime]]

(2) CP_Offset Off

參數

On | Off

- On：啟用CP為On時的動作開始偏移功能。省略時，則顯示目前設定。
- Off：停用CP為On時的動作開始偏移功能。

OffsetTime

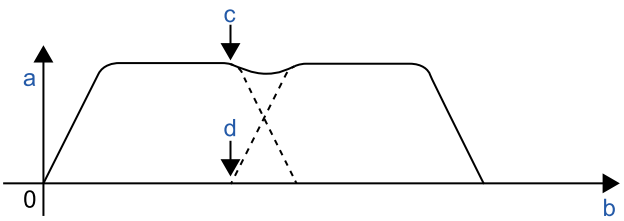
以10~24(單位：ms)範圍的實數值，設定CP為On時的動作開始偏移時間。省略時，被設為預設值10 ms。

說明

CP_Offset以下列動作命令為對象。
Move, Arc, Arc3, CVMove

若為CP On或在動作命令上附加CP參數，則在開始上一個動作減速的同時，執行下一個陳述式。這樣的話，會形成減速區間和下一個動作的加速區間重疊的路徑運動，如下圖所示。此時，由於開始處理陳述式需花費額外時間，因此開始上一個動作減速和開始下一個動作加速的時間並不完全一致。這樣的話，會在路徑運動的連接處附近降低速度而不成為等速軌跡。CP_Offset的功能在於，提前開始後續動作陳述式，以改善這類現象。

路徑運動



符號	說明
a	速度
b	時間
c	開始減速
d	開始加速

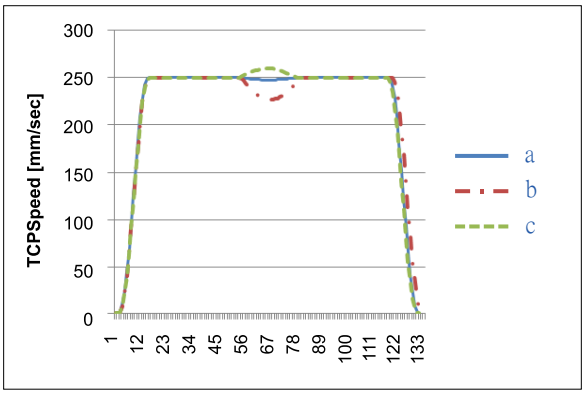
若開啟CP_Offset，則會按設定為OffsetTime參數的時間提前開始後續動作命令處理，使得機器人實際開始減速與開始下一個動作的加速同步，以改善路徑運動的等速性能。雖然在OffsetTime參數中有設定預設值，但請透過應用程式進行微調。特別是後續動作命令包括「!平行處理!」時，會延長開始動作所需額外時間。因此，需將OffsetTime設定為大於預設值(16ms左右)。

為了調整CP_Offset的設定時間(OffsetTime)，在執行欲調整的動作時使用TCPSpeed觀測工具中心點速度。若設定適當的OffsetTime，動作連接處的速度會接近恆定。OffsetTime過長時，TCPSpeed上升；OffsetTime過短時，TCPSpeed下降。請按照實機環境調整CP_Offset。對於實際的控制器和模擬器，由於開始動作的處理時間各不相同，因此無法進行適當的調整。

TCPSpeed計測程式範例

```
Function main
  Motor On
  Power High

  SpeedS 250; AccelS 1500
  Speed 50; Accel 50, 50

  Go XY(300, 500, 500, 90, 0, 180)

  CP_Offset On
  Xqt printTcPSpeed

  Move XY(0, 500, 500, 90, 0, 180) CP
  Move XY(-300, 500, 500, 90, 0, 180)

  Quit printTcPSpeed
  CP_Offset Off
Fend

Function printTcPSpeed
  Do
    Print TCPSpeed
  Loop
Fend
```

OffsetTime調整範例



符號	說明
a	適當的OffsetTime
b	OffsetTime=0
c	OffsetTime過長

本命令不以PTP動作為對象。在PTP動作的情況下，則成為通常的路徑運動。

以下情況時關閉CP_Offset。

- 啟動控制器時
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

參照

CP_Offset函數、CP、Move、Arc、Arc3、CVMove

CP_Offset範例

```
CP_Offset On
Move P1
Move P2
CP_Offset Off
```

3.7.67 CP_Offset函數

CP為On時，用於傳回開始後續動作命令的位移時間。

格式

CP_Offset

傳回值

是表示動作開始位移時間的實數

參照

CP_Offset

CP_Offset函數範例

```
If CP_Offset = O Then
    Print "CP_Offset is off"
EndIf
```

3.7.68 Ctr函數

是用於傳回計數器計數值的函數。

格式

Ctr (輸入位元編號)

參數

輸入位元編號

是計數器中定義的輸入位元編號。最多可同時啟用16個計數器。

傳回值

是計數器的目前計數值(0~65535的整數)

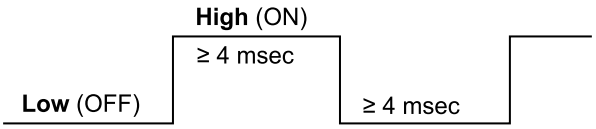
說明

透過設定Ctr，與CTReset陳述式一起使用，以將I/O輸入視為計數之用。

每當被設定為計數器的I/O輸入從關閉轉換為開啟，該輸入便會逐一增加計數器上的計數。

Ctr函數用於獲取指定計數器輸入的目前計數值。無論進行何種I/O輸入，均可在計數器中進行指定。然而，最多可同時啟用16個計數器。

計數器脈衝輸入的時序圖



參照

CTReset

Ctr函數範例

如下所述是獲取I/O輸入的計數值之程式範例。

```
CTReset 3 '將輸入3的計數值重設為0
On 0      '開啟輸出開關
Wait Ctr(3) >= 5
Off 0     '輸入3的輸入週期變為5時，關閉開關(輸出0關閉)
```

3.7.69 CTReset

用於重設指定輸入計數器的值。此外，將輸入設為計數器輸入。

格式

CTReset (輸入位元編號)

參數

輸入位元編號

是作為計數器設定的輸入位元編號。用整數輸入有效的輸入位元編號。最多可使用16個計數器。

說明

CTReset可與Ctr函數一起使用，以便將輸入用作計數器。CTReset用於將指定的輸入位元設定為計數器，並啟動該計數器。倘若指定的輸入已被設定為計數器，則在重設後重新啟動。

注意

- 關閉電源時的計數器

  若關閉電源，則解除所有計數值。

- 使用Ctr函數時

  可以Ctr函數了解目前I/O輸入的計數值。

參照

Ctr

CTReset範例

如下所述是獲取I/O輸入計數值的程式範例。

```
CTReset 3   '將輸入3的計數值重設為0
On 0        '開啟輸出開關
Wait Ctr(3) >= 5
Off 0       '輸入3的輸入週期變為5時，關閉開關(輸出0關閉)
```

3.7.70 CtrlDev函數

用於傳回目前控制裝置的編號。

格式

CtrlDev

傳回值

- 21: PC
- 22：遠端I/O
- 26：遠端乙太網路
- 20: TP4

參照

CtrlInfo函數

CtrlDev函數範例

```
Print "The current control device is: ", CtrlDev
```

3.7.71 CtrlInfo函數

用於傳回控制器資訊。

格式

CtrlInfo (指數)

參數

指數

以整數值指定所要查找的資訊指數。

說明

下表所述為可以CtrlInfo函數取得的資訊。

指數	位元	值	說明
0	N/A		保留以供相容性之需。 若要取得控制器的韌體版本時，請使用指數9。
1	控制器狀態		
	0	&H1	Ready狀態
	1	&H2	Start狀態
	2	&H4	Pause狀態
	3-7		未定義
	8	&H100	緊急停止狀態
	9	&H200	安全門打開狀態
	10	&H400	錯誤狀態
	11	&H800	重大錯誤狀態
	12	&H1000	警告狀態
	13	&H2000	WaitRecover狀態(等待從安全門打開狀態復歸)
	14	&H4000	Recover 狀態(從安全門打開狀態進行復歸動作的狀態)
	15-31		未定義
2	0	&H1	開啟TP的Enable開關
	1-31		未定義
3	0	&H1	TEACH模式回路異常檢測
	1	&H2	安全門回路異常檢測
	2	&H4	緊急停止回路異常檢測
	3-31		未定義
4	N/A		■ 0 - 實際運行模式 ■ 1 - 試運轉模式

指數	位元	值	說明
5	N/A		控制設備： ■ 21 - RC+ ■ 22 - 遠端 ■ 26 - 遠端乙太網路 ■ 29 - 遠端RS232C ■ 20 - TP4
6	N/A		設定的機器人台數
7	N/A		操作模式： ■ 0 - Programing模式 ■ 1 - AUTO模式
8	N/A		未定義
9	N/A		控制器的韌體版本 主版本編號*1000000 + 副版本編號*10000 + 修訂編號*100 + 創建編號 例 1.6.2.4 時：1060204
10	N/A		硬碟的SMART狀態 ■ 0：SMART狀態正常 ■ 1：SMART狀態異常 若SMART狀態異常，可能是硬碟發生故障。因此，請儘速備份資料，並更換為新硬碟。 使用RAID選件時，無法使用SMART狀態。始終傳回正常。
15	N/A		DC電源電壓值 獲得輸入電壓的100倍的值。 例：電壓為48.01V時，獲得4801。 對於不支援DC電壓的機種，將傳回錯誤。
16	N/A		PLC供應商類型 ■ 0: None ■ 1: Allen Bradley ■ 2: CODESYS

傳回值

用於傳回所需Long整數運算式的值。

參照

RobotInfo、TaskInfo

CtrlInfo函數範例

```
Print "The number of robots is: ", CtrlInfo(6)
```

3.7.72 CurDir$函數

以字串傳回目前目錄。

格式

CurDir$

傳回值

用於傳回驅動器名稱和路徑名稱的字串。

參照

ChDir, CurDrive$, CurDisk$

CurDir$函數範例

```
Print "The current directory is: ", CurDir$
```

3.7.73 CurDisk$函數

以字串傳回目前的磁碟名稱。

格式

CurDisk$

傳回值

用於傳回磁碟名稱的字串。

參照

ChDisk, CurDir$, CurDrive$

CurDisk$函數範例

```
Print "The current disk is: ", CurDisk$
```

3.7.74 CurDrive$函數

以字串傳回目前的驅動器名稱。

格式

CurDrive$

傳回值

用於傳回驅動器名稱的字串。

參照

ChDrive, CurDir$, CurDisk$

CurDrive$函數範例

```
Print "The current drive is: ", CurDrive$
```

3.7.75 CurPos函數

用於傳回機器人的目前動作目標位置。

格式

CurPos

傳回值

用於傳回機器人的目前動作目標位置。

參照

InPos、FindPos、RealPos

CurPos函數範例

```
Function main

   Xqt showPosition
   Do
      Jump P0
      Jump P1
   Loop
Fend

Function showPosition

   Do
      P99 = CurPos
      Print CX(P99), CY(P99)
   Loop
Fend
```

3.7.76 Curve

用於建立資料和點，以自由曲線進行CP控制。定義數個點資料，並確保路徑正確。

格式

Curve 檔名, 動作曲線的開閉, 模式指定, 坐標軸數, 連續點資料指定

參數

檔案名稱

以字串指定儲存點數據的檔案名稱。副檔名固定為「.CVT」。若省略副檔名，則自動新增.CVT的副檔名。執行Curve命令，會自動建立檔案。不可指定路徑。此外，不受ChDisk等的影響。詳細內容請參閱ChDisk。

動作曲線的開閉

結束曲線動作時，指定動作曲線的開閉。此參數用於指定以下任一值。

- C -生成的曲線為封閉曲線
- O -生成的曲線為開放曲線

若指定開放曲線，則依Curve命令讓手臂停在連續點數據的最後點位置。若指定封閉曲線，則依Curve命令，通過最終點後仍繼續運作，讓手臂回到連續點資料的起點並停止動作。

模式指定

用於指定是否補償姿態(手臂是否朝U軸正切方向自動進行內插)。可用高階4位元指定ECP編號。

模式指定		姿態補償	ECP編號
十六進位	十進位		
&H00	0	不執行	0
&H10	16		1
&H20	32		2
...	...		...
&HA0	160		10
&HB0	176		11
&HC0	192		12
&HD0	208		13
&HE0	224		14
&HF0	240		15
&H02	2	執行	0
&H12	18		1
&H22	34		2
...	...		...
&HA2	162		10
&HB2	178		11
&HC2	194		12
&HD2	210		13

模式指定		姿態補償	ECP編號
十六進位	十進位		
&HE2	226		14
&HF2	242		15

若指定姿態補償，手臂便僅使用連續點數據起點的U軸坐標。姿態補償用於維持始終讓姿態軸在XY平面上與曲線接觸。諸如刀具那樣，屬於要持續進行正切方向控制的工具時進行指定。朝U軸的正切方向，指定要自動圓弧內插的封閉曲線時，U軸會從起點旋轉360度。因此，在執行CVMove命令前，為避免因U軸旋轉而發生錯誤，請以Range命令指定U軸動作範圍。

要使用ECP時，請在高階4位元上指定ECP編號。

若要生成顧及到包含在點資料中的附加軸位置的自由曲線，請將第9位元指定為「1」。例如，若不使用姿態補償及ECP，生成顧及到附加軸位置的自由曲線，請指定「&H100」。

以附加軸生成自由曲線時，無論機器人坐標系如何，S關節、T關節都將分別獨立地連接連續點資料。

附加軸由PG軸構成時，不會以連續點生成自由曲線，而建立在最終點動作的資料。

座標軸數

以2、3、4、6的整數指定在曲線動作中控制的坐標軸數。

- 在未包含2 - 姿態的XY平面上生成自由曲線。(垂直6軸型(包含N系列)以外)
- 在未包含3 - 姿態的XYZ空間生成自由曲線。(垂直6軸型(包含N系列)以外)
- 在包含4 - 姿態的XYZ空間生成自由曲線。(垂直6軸型(包含N系列)以外)
- 在包含6 - 姿態的XYZ空間生成自由曲線。(僅限於垂直6軸型(包含N系列))

未被選擇為控制對象的軸將維持上一次的編碼器脈衝位置，在Curve動作時不進行動作。

連續點數據指定 { 點數據|點編號(開始:結束) } [,輸出命令] ...

用逗號(,)分隔指定此參數的每個點數據。毫無遺漏地按升序或降序排列點資料時，可用冒號連接2個點編號進行指定(例：P (1:5))。

若要與動作同步，在中途開啟或關閉I/O輸出連接埠，可用逗號(,)分隔記述輸出命令。

如下所述，通常用逗號分隔指定連續點資料。

```
Curve "MyFile", O, 0, 4, P1, P2, P3, P4
```

或使用冒號按如下所述進行指定。

```
Curve "MyFile", O, 0, 4, P(1:4)
```

在上述範例中，使用P1、P2、P3、P4指定曲線。輸出命令可省略，在曲線動作時控制輸出操作的情況下使用。此命令可用於指定開啟和關閉I/O或記憶體I/O。在手臂通過連續點資料的特定點之後執行輸出命令。1個Curve陳述式可包含最多16個輸出命令。在以下範例中，手臂在通過P2之後執行「On 2」命令，然後手臂通過P3至P10的所有點。

```
Curve "MyFile", C, 0, 4, P1, P2, ON 2, P(3:10)
```

說明

此命令用於以指定的點資料建立可讓機器人手臂進行自由曲線CP動作的檔案，然後，將該資料儲存於控制器的檔案中。使用此命令建立的資料用於以CVMove命令執行CP動作之時。

Curve命令用於使用三次樣條函數，獨立計算各指定點的X、Y、Z、U、V、W坐標值，並據此生成軌跡。因此，點之間的間隔或姿態變化較大時，難以預測被生成的軌跡。

不需在Curve命令前指定動作時的速度和加減速度。若是在執行CVMove之前，則可用SpeedS或AccelS等命令進行變更。

若在Curve命令參數的點組中使用本地坐標系中設定的點，則可設定通過該位置的曲線。若讓指定的點資料具有本地屬性，則可變更為Local陳述式 (在Curve命令之後繼續) 的本地坐標系上之點。

注意

- 請盡可能進行姿態補償

  本公司建議盡可能進行姿態補償。尤其是使用CVMove對相同點組連續進行迴圈時，請進行姿態補償。若未進行姿態補償，在高速進行動作時，機器人有可能無法保持正確的位置。

- 開放曲線點組的點數範圍

  為開放曲線時，請指定3~1000點。

- 封閉曲線的點組點數範圍

  使用RC700系列、RC90系列的控制器時，封閉曲線請指定為3～1000點。

  使用T/VT時，封閉曲線請指定為3～300點。(在模擬器指定T/VT時，可動作到1000點，但在T/VT的實機環境下僅到300點為止。)

- 點數多時，處理時間會延長

  以最大點數執行Curve命令時，開放曲線會需要數秒，封閉曲線則會需要數十秒左右的時間。

  特別是封閉曲線的處理時間較長，因此點數多時建議使用開放曲線。

  使用開放曲線製作與封閉曲線相近軌跡的範例如Curve範例2所述。

  但是，若以Curve命令產生自由曲線檔案，並以同樣的檔案進行多次CVMove動作時，只有在執行Curve命令的那一次需要上述時間。

- 檔案相容性

  使用 Ver.7.5.1 或更高版本的韌體創建的檔案在早期版本的韌體中不可用。此外，使用Ver.7.5.1 或更早版本的韌體創建的檔案可用於Ver.7.5.1或更高版本的韌體。

常見錯誤

- 欲將手臂移出活動範圍時

  Curve命令無法用於檢查已設定曲線的動作範圍。這意味著，在機器人手臂動作時，使用者設定的曲線軌跡可能超出活動範圍。此時，會發生「超出動作範圍」的錯誤。

- 點的間隔不平均時

  點間隔不平均時，生成曲線的加速度可能會異常升高。屆時，可能會發生異常加速度錯誤。
  啟用下列設定可能會執行動作而不發生錯誤。
  垂直六軸機器人 (包括N系列), RS系列：AvoidSingularity SING_VSD
  水平多關節型機器人 (不含RS系列)：VSD

參照

AccelS函數、Arc、CVMove、ECP、Move、SpeedS

Curve範例1

在以下範例中，使用被稱為MYCURVE.CVT的自由曲線檔案，追蹤通過P1至P7的曲線。同時，在P2開啟輸出連接埠，並在P7使手臂減速。

設定自由曲線

```
> curve "mycurve", O, 0, 4, P1, P2, On 2, P(3:7)
```

以直線方式將手臂移至P1

```
> jump P1
```

用已定義的自由曲線「mycurve」移動手臂

```
> cvmove "mycurve"
```

Curve範例2

以下列舉(1)開放曲線、(2)封閉曲線、(3)以開放曲線進行與封閉曲線相近動作的範例。

教導點如下列所示。

```
P0 = XY(0, 300, -50, 0)'起點・終點
P1 = XY(300, 200, -50, 0)
P2 = XY(300, 400, -50, 0)
P3 = XY(-300, 400, -50, 0)
P4 = XY(-300, 200, -50, 0)
P10 = XY(10, 299.7, -50, 0)    '起點之後
P11 = XY(-10, 299.7, -50, 0)   '終點之前
```

(1) 開放曲線

設定開放曲線的自由曲線

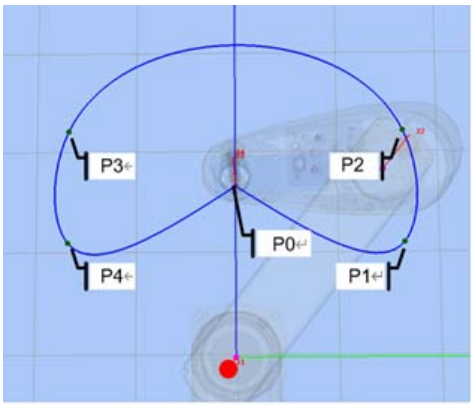
```
> Curve "mycurve_O", O, 0, 2, P(0:4), P0
```

使手臂直線移動至P0

```
> jump P0
```

以所定義之開放曲線的自由曲線「"mycurve_O」移動手臂

```
> CVMove "mycurve_O"
```



由於是開放曲線，起點與終點相同但不會順暢地連接。

(2) 封閉曲線

設定封閉曲線的自由曲線

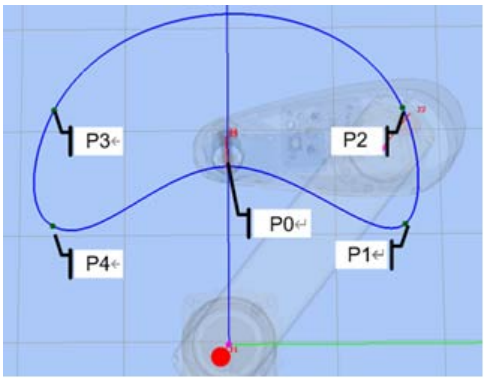
```
> Curve "mycurve_C", O, 0, 2, P(0:4)
```

使手臂直線移動至P0

```
> jump P0
```

以所定義之封閉曲線的自由曲線「mycurve_C」移動手臂

```
> CVMove "mycurve_C"
```



由於是封閉曲線，起點與終點會順暢地連接。

(3) 以開放曲線進行與封閉曲線相近的動作

設定開放曲線的自由曲線。在起點之後與終點之前設定點。

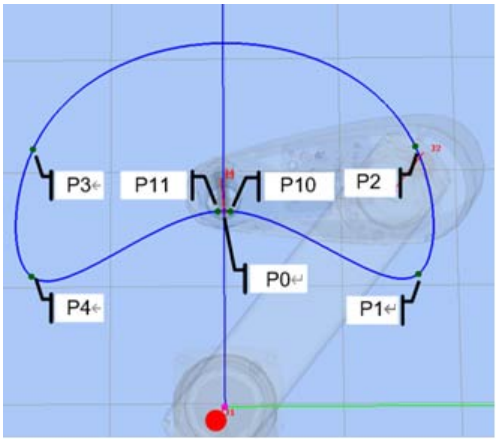
```
> Curve "mycurve_O_mod", O, 0, 2, P0, P10, P(1:4), P11, P0
```

使手臂直線移動至P0

```
> jump P0
```

以所定義之開放曲線的自由曲線「mycurve_O_mod」移動手臂

```
> CVMove "mycurve_O_mod"
```



雖為開放曲線，但由於通過P10、P11，起點與終點會順暢地連接。

3.7.77 CVMove

用於執行以Curve命令定義的自由曲線CP動作。

格式

CVMove 檔名[CP] [Till | Find] [SYNC]

參數

檔案名稱

以字串運算式或字串指定以Curve命令創建且儲存於控制器的檔案名稱。不可指定路徑。此外，不受ChDisk等的影響。詳細內容請參閱ChDisk。

CP

用於指定最後點後面的路徑運動。可省略。

Till | Find

用於記述Till或Find運算式。可省略。

```
Till | Find
Till Sw(運算式) = {On | Off}
Find Sw(運算式) = {On | Off}
```

SYNC

用於預約動作命令。在以SyncRobots開始動作之前，機器人不進行動作。

說明

CVMove用於執行檔案資料所設定的自由曲線CP動作。務必事先以Curve命令建立此檔案。

若檔名無副檔名，則自動附加「.cvt」。

可以SpeedS或AccelS命令變更CVMove的CP動作速度和加減速度。

Curve命令用於事前用Local定義的點執行動作時，可以Local命令變更位置。

執行CVMove時，請充分注意與週邊裝置之間的干擾情況。尤其在垂直6軸型機器人(包含N系列)的情況下，若要在相鄰點之間使姿態突然發生變化，從三次樣條函數的性質上來看，可能從其前後的點姿態就開始變化，導致產生非預期動作。執行CVMove時，請注意與週邊裝置之間的干擾情況，並充分確認軌跡。

請盡可能以等間隔細密地指定點，切勿使相鄰點之間的手臂姿態突然發生變化。
在CVMove執行過程中發生異常加速度錯誤時，啟用下列設定可能會執行動作而不發生錯誤。
垂直六軸機器人 (包括N系列), RS系列：AvoidSingularity SING_VSD
水平多關節型機器人 (不含RS系列)：VSD

若已添加CP參數，可在開始動作減速時重疊下一個動作命令的加速。此時，不在目標坐標上進行定位。

參照

AccelS函數、Arc、Curve、Move、SpeedS、Till、TillOn

CVMove範例

在以下範例中，使用被稱為MYCURVE.CVT的自由曲線檔案，追蹤通過P1至P7的曲線，然後在P2開啟輸出連接埠，最後在P7使手臂減速。

設定自由曲線

```
> curve "mycurve", O, 0, 4, P1, P2, On 2, P(3:7)
```

以直線方式將手臂移至P1

```
> jump P1
```

用已定義的自由曲線「mycurve」移動手臂

```
> cvmove "mycurve"
```

3.7.78 CX、CY、CZ、CU、CV、CW、CR、CS、CT

設定點資料的坐標值。CV、CW只能用於垂直6軸型機器人(包含N系列)。

CR只能用於Joint型機器人。

CS、CT只能用於設定附加軸的機器人。

格式

CX (坐標) = 值
CY (坐標) = 值
CZ (坐標) = 值
CU (坐標) = 值
CV (坐標) = 值
CW (坐標) = 值
CR (坐標) = 值
CS (坐標) = 值
CT (坐標) = 值

參數

坐標
　　指定P編號、P(運算式)，或點標籤。
值
　　以實數值指定要設定的坐標值。

參照

CX、CY、CZ、CU、CV、CW、CR、CS、CT函數

CX、CY、CZ、CU、CV、CW、CR、CS、CT範例

```
CX(pick) = 25.34
```

3.7.79 CX、CY、CZ、CU、CV、CW、CR、CS、CT函數

用於取得點資料的坐標值。

CV、CW函數只能用於垂直6軸型機器人(包含N系列)。

CS、CT函數只能用於設定附加軸的機器人。

格式

CX (坐標)
CY (坐標)
CZ (坐標)
CU (坐標)
CV (坐標)
CW (坐標)
CR (坐標)
CS (坐標)
CT (坐標)

參數

坐標
　　用於指定點數據。

傳回值

用於傳回指定的坐標值。

- CX、CY、CZ各函數的傳回值：實數值(單位：mm)。
- CU、CV、CW各函數的傳回值：實數值(單位：deg)。
- CS、CT各函數的傳回值：實數值(單位：mm)或實數值(單位：deg)。會因附加軸的設定而異。

說明

取得指定點資料的坐標值。

若要取得機器人的目前位置坐標值，請將Here用於參數。

參照

CX、CY、CZ、CU、CV、CW、CR、CS、CT

CX、CY、CZ、CU、CV、CW、CR、CS、CT函數範例

在以下範例中，以點「pick」取出X坐標值，並將其設定為變數x。

```
Function cxtest
  Real x
  x = CX(pick)
  Print "The X Axis Coordinate of point 'pick' is", x
Fend
```

3.8 D

3.8.1 Date

用於顯示日期。

格式
Date

結果
用於顯示目前的日期。

參照
Time, Date$

Date範例
命令視窗中的執行範例

```
> Date
2009/08/01
```

3.8.2 Date$函數

用於傳回目前的日期。

格式

Date$

傳回值

以字串傳回日期。

格式為yyyy/mm/dd [西元年號/月/日]。

參照

Date、Time、Time$

Date$函數範例

```
Print "Today's date: ", Date$
```

3.8.3 Declare

用於叫用DLL(動態連結程式庫)所定義的外部函式。

格式

Declare 函式名稱, "DLL檔案路徑" , "DLL內函式名稱" [, (引數清單)] As函數型態

參數

函數名稱
　指定從程式呼叫時的函數名稱。
DLL檔案路徑
　以使用引號("")圍起的字串或#define定義的巨集指令指定函式庫的路徑和名稱。
　未指定路徑時，RC＋則搜尋目前專案目錄中的檔案。找不到時，則假定檔案位於Windows system32目錄中。可省略副檔名。若已省略，則假定為.DLL。
DLL內函數名稱
　是選件參數。指定DLL中的實際函式名稱或函式指數。名稱被區分為大寫和小寫。以用引號(“”)圍起的字串指定函式名稱。若要使用指數，在指數前附加#。若已省略，則將以「函式名稱」參數指定的函式名稱用作DLL內函式名稱。
引數清單
　是DLL引數的清單。請使用以下格式的引數。可省略。

　[{ByRef | ByVal}] 變數名稱[( )] As變數類型

　ByRef
　　參照要呼叫的函數變數時，指定ByRef。此時，可將函式內的引數變更反映於叫用側的變數中。可變更透過參照而傳遞的值。可省略。
　ByVal
　　是預設值。以值(ByVal)傳遞參數。只傳遞數值，因此，函式返回時，不可更改該變數。在未用叫用的函式來變更變數值時進行指定。可省略。
　變數名稱
　　是必須的參數。是表示引數的變數名稱。依變數命名時的規則進行命名。要將陣列變數用作引數時，請務必指定ByRef。
　變數類型
　　是必須的參數。用於宣告引數型態。

函數類型
　是必須的參數。請宣告函數型態。

說明

用於從目前的程式叫用DLL函式之情況。請在函式之外部分使用Declare。

Declare陳述式用於在編譯時確認存在DLL檔案和函式。

發生錯誤2473「與Epson RC+的連結功能失敗 (忙碌中或未初始化)」時的因應如下。

- 請確認Epson RC+已連接到控制器。DLL函數在PC上執行。
- 請確認使用WindowsStatus函數，可利用DLL叫用。

以ByVal傳遞數值變數

```
SPEL: Declare MyDLLFunc, "mystuff.dll", "MyDLLFunc", (a As Long) As Long
VC++ long _stdcall MyDllFunc(long a);
```

以ByVal傳遞字串變數

```
SPEL: Declare MyDLLFunc, "mystuff.dll", "MyDLLFunc", (a$ As String) As Long
VC++ long _stdcall MyDllFunc(char *a);
```

以ByRef傳遞數值變數

```
SPEL: Declare MyDLLFunc, "mystuff.dll", "MyDLLFunc", (ByRef a As Long) As Long
VC++ long _stdcall MyDllFunc(long *a);
```

以ByRef傳遞字串變數

```
SPEL: Declare MyDLLFunc, "mystuff.dll", "MyDLLFunc", (ByRef a$ As String) As Long
VC++ long _stdcall MyDllFunc(char *a);
```

以ByRef傳遞字串後，可在DLL中變更字串。字串最長可使用255個字元。請注意勿超過最大字元數。SPEL+為字串變數確保內部固定的255個字元區域。

以ByRef傳遞數值陣列

```
SPEL: Declare MyDLLFunc, "mystuff.dll", "MyDLLFunc", (ByRef a() As Long) As Long
VC++ long _stdcall MyDllFunc(long *a);
```

來自DLL函式的傳回值

DLL函式可以返回除字串以外的任何數據類型的傳回值。

如果需要傳回字串，請使用上面的"以ByRef傳遞字串變數"，將字串變量作為參數。

如果傳回值是字串變數，則會發生錯誤 3614: “You cannot specify a String for Declare return data type.”。

變數類型

Epson RC+ 8.0與C/C++變數類型的對應如下表所示。

由於沒有與Epson RC+ 8.0對應的數據，因此不使用C/C++的byte類型或結構。

Epson RC+ 8.0與C/C++變數類型的對應表

Epson RC+ 8.0	C/C++
Boolean	short
Byte	short
Short	short
Integer	short
Long	int
Real	float
Double	double
String	char [256] * 包含Null

程式範例

```
Declare ReturnLong, "mystuff.dll", "ReturnLong", As Long

Function main

    Print "ReturnLong = ", ReturnLong
Fend
```

參照

Function...Fend

Declare範例

```
' 以Declare命令定義外部函式。
' 在DLL檔案路徑中未設定完整路徑時，可將DLL檔案置於目前專案資料夾、
' Windows System32資料夾中。

Declare MyDLLTest, "mystuff.dll", "MyDLLTest" As Long

Function main
    Print MyDLLTest
Fend

' '以Declare命令定義具有2個Integer型引數的外部函式。

 #define MYSTUFF "mystuff.dll"

Declare MyDLLCall, MYSTUFF, "MyTestFunc", (var1 As Integer, var2 As Integer) As
Integer

' 以Declare命令指定外部函式的完整路徑，
' 以指數指定並定義函式。

Declare MyDLLTest, "c:\mydlls\mystuff.dll", "#1" As Long
```

3.8.4 DegToRad函數

用於將角度轉換為弧度。

格式

DegToRad (角度)

參數

角度

指定轉換為弧度的度值(實數值)。

傳回值

用於傳回Double型弧度值。

參照

ATan、ATan2、RadToDeg函數

DegToRad函數範例

```
s = Cos(DegToRad(x))
```

3.8.5 Del

用於刪除檔案。

格式
Del 檔名

參數

檔案名稱
　　指定要刪除的路徑和檔案名稱。為檔名附加副檔名。詳細內容請參閱ChDisk。

說明
刪除指定的檔案。

Del範例
命令視窗中的操作範例

```
> Del TEST.PTS     '用於從目前目錄中刪除點檔案

> Del c:\TEST.PTS     'NG
!! 錯誤： 7213  找不到指定的檔案。
> Del c:\TEST.PTS     'OK
```

3.8.6 DeleteDB

用於從開啟的資料庫內的表格中刪除資料。

格式

DeleteDB #資料庫編號,表格名稱[, 刪除條件]

參數

資料庫編號

用於指定以OpenDB指定的資料庫編號(501~508的整數值)。

表格名稱

指定要刪除資料的表格名稱。

刪除條件

指定刪除條件。可用AND、OR指定複合條件。未指定刪除條件時，則刪除表格內所有資料。

說明

從開啟的資料庫之指定表格中刪除符合刪除條件的資料。

開啟的資料庫為Excel活頁簿時，不可執行此命令。

注意

需連接有安裝RC+的PC。

參照

OpenDB、CloseDB、SelectDB、UpdateDB

3.8.7 DiffPoint函數

用於傳回指定的2點之差異。

格式

DiffPoint (點資料1, 點資料2)

參數

點數據1
　　指定第1個點數據。
點數據2
　　指定第2個點數據。

傳回值

將從點資料1所見點資料2之位置姿態，作為點資料傳回。

說明

將以點資料1為原點之坐標系的點資料2之位置姿態，作為點資料傳回。要傳回的點資料之本地編號及Hand等旗標資訊，以預設值傳回。

當2個點資料中，有任一點資料有未定義值時，將以「0」進行計算。

舉例而言，假設Point1為「XY (10,0,0,0,0,0): ST (10, 10)」、Point2為「XY (10,0,0,0,0,0)」，其S及T值於Point2為未定義，但於Point1為已定義，則傳回值將為將Point2之S及T值以「0」進行計算後的值。

注意

- 支援的控制器型號

  不支援T/VT系列。

DiffPoint函數範例

```
'顯示從點P1所見P2之位置姿態。

Print DiffPoint(P1, P2)

'顯示從目前位置(Here)所見P1之位置姿態。

Print DiffPoint(Here, P1)
```

3.8.8 DiffToolOrientation函數

用於傳回由工具坐標系的各坐標軸形成之角度(單位：度)，以表示在指定的2個點位置上所實現的各工具姿態變化量。

格式

DiffToolOrientation (點資料1, 點資料2, 軸編號)

參數

點數據1
　　指定第1個點數據。
點數據2
　　指定第2個點數據。
軸編號
　　用於指定求出角度變化量的工具坐標系坐標軸。

常數	值
COORD_X_PLUS	1：+X軸
COORD_Y_PLUS	2：+Y軸
COORD_Z_PLUS	3：+Z軸
COORD_ALL	4：任意軸

傳回值

角度(0~180度的實際正數值)

說明

以由指定工具坐標系的坐標軸形成的角度(0~180度的實際正數值)傳回由指定的2個點資料各自實現的工具姿態所形成的姿態變化量。資料1、2的順序不會影響到獲得的結果。此外，2點間的原點位置關係(X、Y、Z的坐標值)也不會影響到獲得的結果。

用於指定COORD_ALL時，傳回繞任意軸旋轉之旋轉量。任意軸意為於有2個姿態(U, V, W)時，繞虛擬軸(1條直線)旋轉，經1次旋轉後可移動的軸。使用於不限定於各軸，求綜合旋轉角度時。

注意

- 支援的控制器型號

  不支援T/VT系列在軸編號中指定COORD_ALL。

DiffToolOrientation函數範例

```
'顯示由點P1和P2的工具坐標Z軸形成的角度。

Print DiffToolOrientation(P1, P2, COORD_Z_PLUS)
```

3.8.9 DispDev

用於設定目前顯示裝置。

格式

DispDev (裝置ID)

參數

設備ID

用於指定顯示設備的設備ID。

- 21 RC+
- 20 TP4

也可使用以下常數。

- DEVID_SELF 21
- DEVID_TP 24
- DEVID_TP3 20

參照

DispDev函數

DispDev範例

```
DispDev DEVID_TP
```

3.8.10 DispDev函數

用於傳回目前顯示裝置。

格式

DispDev

傳回值

用於傳回設有裝置ID的顯示裝置之整數值。

- 21 RC+
- 20 TP4

參照

DispDev

DispDev範例

```
Print "The current display device is ", DispDev
```

3.8.11 Dist函數

用於傳回2個機器人坐標間的距離。

格式

Dist (坐標1, 坐標2)

參數

坐標1、坐標2
　　指定2個機器人坐標。

傳回值

用於以實數值傳回2個坐標間的距離。(單位：mm)

說明

即便使用附加軸，也只傳回機器人的移動距離。例如，即便是將附加軸用作行走軸的情況，也不考量附加軸的移動距離。

對於關節型機器人，本函數傳回值並沒有意義。

參照

CU、CV、CW、CX、CY、CZ

Dist函數範例

```
Real distance

distance = Dist(P1, P2)
```

3.8.12 Do...Loop

在符合條件時或符合條件前(不符合時)，在DO...LOOP之間重複執行。

格式

Do [{ While | Until } 條件運算式]
[陳述式]
[Exit Do]
[陳述式]
Loop

或使用如下格式。
Do
[陳述式]
[Exit Do]
[陳述式]
Loop [{ While | Until } 條件運算式]

Do Loop陳述式分為條件運算式和陳述式兩種。

條件運算式
　　是用於表示True或False的數字或字串運算式。條件運算式為空(Null)時，條件則作為False進行處理。可省略。
陳述式
　　是在符合條件時或符合條件前重複執行之1個以上的陳述式

說明

作為離開Do...Loop的另一個方法，可在Do...Loop中的任何部分插入任何數量的Exit Do陳述式。Exit Do常被用於評估數個條件(If...Then等)之後的情況。按If...Then使用Exit Do陳述式時，則將控制轉移到Loop的下一個陳述式中。

若在巢狀Do...Loop陳述式中使用，Exit Do則用於將控制轉移到發生該情況的迴圈之上1級迴圈。

注意

- 使用時，請勿在Loop陳述式中頻繁重複XQT命令

  使用時，請勿在Do…Loop等Loop陳述式中頻繁重複XQT命令。否則，控制器有可能進入意外停機狀態。若要這樣使用，請新增Wait命令(Wait 0.1)。

- 盡可能將空無限循環和無限迴圈處理與等待命令結合使用

  盡量不要進行空的Do...Loop或類似操作，因為它們可能會影響您的系統。控制器檢測無限循環任務，並確定系統受到影響，則可能會出現錯誤 2556(檢測到過多的迴圈)。執行需要迴圈的演算或等待I/O時，請在迴圈中執行Wait命令(Wait 0.1)等，以避免佔用CPU。

- 如果不使用Exit Do並從嵌套結構退出迴圈

  若重複執行Exit Do以外的命令(Gosub、Goto、Call 命令等)脫離迴圈的程式會導致錯誤2020。如果要在循環中間退出，請使用Exit Do命令退出它。

參照

For...Next、Select...Send

Do...Loop範例

```
Do While Not Lof(1)
    Line Input #1, tLine$
    Print tLine$
Loop
```

3.8.13 Double

宣告Double型變數。(8位元組的雙精度數)

格式

Double 變數名稱 [(陣列變數的最大元素編號)] [, 變數名稱[(陣列變數的最大元素編號)]...]

參數

變數名稱

用於指定宣告Double型的變數名稱。

陣列變數的最大元素編號

是陣列變數的最大元素編號，最大可宣告三維。使用如下格式。可省略。

(最大元素編號1, [最大元素編號2], [最大元素編號3])

元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。在所有元素數不超過以下最大值的範圍內，指定各最大元素編號。

- 本地變數：2,000
- 備份變數(Global Preserve)：4,000
- 全域變數和模組變數：100,000

說明

Double用於將指定的變數宣告為Double型。在Function的開頭宣告本地變數。在Function之外宣告全域變數和模組變數。

Double型的有效位數為14位。

參照

Boolean、Byte、Global、Int32、Int64、Integer、Long、Real、Short、String、UByte、UInt32、UInt64、UShort

Double範例

以下為使用Double宣告數個變數的程式範例。

```
Function doubletest
  Double var1
  Double A (10)          'Double型的一維陣列
  Double B (10, 10)      'Double型的二維陣列
  Double C (5, 5, 5)     'Double型的三維陣列
  Double arrayvar(10)
  Integer i
  Print "Please enter a Number:"
  Input var1
  Print "The variable var1 = ", var1
  For i = 1 To 5
    Print "Please enter a Number:"
    Input arrayvar(i)
    Print "Value Entered was ", arrayvar(i)
  Next i
Fend
```

3.9 E

3.9.1 ECP

用於選擇或顯示目前的ECP(外部控制點)。

格式

(1) ECP ECP編號

(2) ECP

參數

ECP編號

以緊接於後的動作命令從16種ECP設定(整數值0~15)中指定要使用的設定。可省略。ECP 0用於將ECP選擇設為停用。

結果

若省略參數，則顯示目前設定的ECP編號。

說明

ECP命令用於選擇以ECP編號指定的外部控制點。

注意

- 僅限於裝有外部控制點選項時，方可使用此命令。
- 關閉電源對ECP選擇帶來的影響

  關閉主電源，即清除ECP的選擇狀態。

參照

ECPSet

ECP範例

```
>ecpset 1, 100, 200, 0, 0
>ecp 1
```

3.9.2 ECP函數

用於傳回目前指定的ECP(外部控制點)編號。

格式

ECP

傳回值

用於以整數值傳回目前指定的ECP編號。

注意

僅限於裝有外部控制點選項時，方可使用此命令。

參照

ECP

ECP函數範例

```
Integer savECP

savECP = ECP
ECP 2
Call Dispense
ECP savECP
```

3.9.3 ECPClr

清除(未設定)外部控制點的設定。

格式

ECPClr ECP編號

參數

ECP編號

以整數值指定1~15的外部控制點中要清除(未設定)的編號。(ECP的0號為初始設定值，不可清除。)

注意

僅限於裝有外部控制點選項時，方可使用此命令。

參照

Arm、ArmClr、ArmSet、ECPSet、Local、LocalClr、Tool、TLSet

ECPClr範例

```
ECPClr 1
```

3.9.4 ECPDef函數

用於傳回外部控制點的設定狀態。

格式

ECPDef (ECP編號)

參數

ECP編號

以整數值指定要呼叫狀態的ECP編號。

傳回值

若有設定指定ECP編號的外部控制點，則傳回「True」；若未設定，則傳回「False」。

參照

Arm、ArmClr、ArmSet、ECPSet、Local、LocalClr、Tool、TLClr、TLSet

ECPDef範例

```
Function DisplayECPDef(ecpNum As Integer)

   If ECPDef(ecpNum) = False Then
      Print "ECP ", ecpNum, "is not defined"
   Else
      Print "ECP ", ecpNum, ": ",
      Print ECPSet(ecpNum)
   EndIf
Fend
```

3.9.5 ECPSet

用於定義或顯示外部控制點。

格式

(1) ECPSet ECP編號, ECP點指定

(2) ECPSet ECP編號

(3) ECPSet

參數

ECP編號

以運算式或1~15的整數值指定定義為外部控制點的編號。

ECP點指定

以P編號或P(運算式)、點標籤、點數據進行指定。

傳回值

若省略所有參數，則顯示目前ECP設定。

若僅指定ECP編號，則顯示指定的ECP設定。

說明

設定外部控制點。

注意

僅限於裝有外部控制點選項時，方可使用此命令。

ECPSet範例

```
ECPSet 1, P1
ECPSet 2, 100, 200, 0, 0
```

3.9.6 ECPSet函數

用於傳回被分配到指定ECP編號的外部控制點的點資料。

格式

ECPSet (ECP編號)

參數

ECP編號

以整數值指定要呼叫點數據的ECP編號。

傳回值

用於傳回指定ECP編號的外部控制點的點資料。

注意

僅限於裝有外部控制點選項時，方可使用此命令。

參照

ECPSet

ECPSet範例

```
P1 = ECPSet(1)
```

3.9.7 ElapsedTime函數

用於以秒為單位傳回啟動生產節拍時間測量用計時器之後的經過時間。

格式

ElapsedTime

傳回值

用於以實數值(單位：秒)傳回生產節拍時間測量用計時器的經過時間。計時器的範圍為0~約1.7E+31。計時器解析度為0.001秒。

說明

傳回啟動生產節拍時間測量用計時器之後的經過時間。本函數與Tmr函數不同，程式處於暫停狀態的時間並不算作經過時間。

可用ResetElapsedTime重設生產節拍時間測量用計時器。

```
Real overhead

ResetElapsedTime
overHead = ElapsedTime
```

參照

ResetElapsedTime、Tmr函數

ElapsedTime範例

```
ResetElapsedTime        '重設生產節拍時間測量用計時器
For i = 1 To 10         '執行10次
    GoSub Cycle
Next
Print ElapsedTime / 10 '計算並顯示生產節拍時間
```

3.9.8 Elbow

用於設定點的臂肘姿態。

格式

(1) Elbow 點指定 [, 設定值]

(2) Elbow

參數

點指定

指定P編號、P(運算式)或點標籤。

設定值

指定整數或運算式。

- 1 = Above (/A)
- 2 = Below (/B)

傳回值

若2個參數皆省略，則顯示機器人目前位置的臂肘姿態。

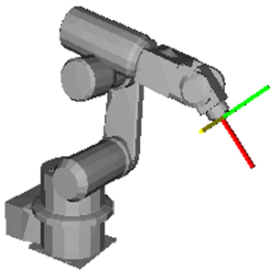
若省略設定值參數，則顯示指定點的臂肘姿態。

參照

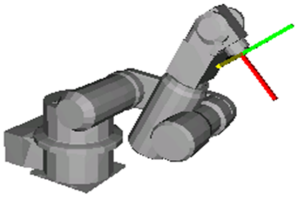
Elbow函數、Hand、J4Flag、J6Flag、Wrist

Elbow範例

```
Elbow P0, Below
Elbow pick, Above
Elbow P(myPoint), myElbow

P1 = 0.000,  490.000,  515.000,   90.000,  -40.000,  180.000
```



```
Elbow P1, Above
Go P1
```



```
Elbow P1, Below
Go P1
```

3.9.9 Elbow函數

用於傳回點的臂肘姿態。

格式

Elbow [(點指定)]

參數

點指定

以點運算式進行指定。可省略。若省略，則傳回機器人目前位置的臂肘姿態。

傳回值

- 1: Above (/A)
- 2: Below (/B)

參照

Elbow、Hand、Wrist、J4Flag、J6Flag

Elbow函數範例

```
Print Elbow(pick)
Print Elbow(P1)
Print Elbow
Print Elbow(P1 + P2)
```

3.9.10 EncTemper

顯示編碼器目前溫度。

格式

EncTemper[Joint number]

參數

關節編號

以1~9整數值或運算式設定關節編號。
附加軸的S軸為8，T軸為9。

說明

顯示編碼器目前溫度 [℃]。
省略關節編號時，顯示所有軸的編碼器溫度。
指定未對應編碼器溫度的機器人或關節編號時，顯示-1000。

提示

- 本命令中溫度讀取使用的感測器，並非已校正儀器。
- 請瀏覽相對的變化，用以掌握溫度變化的趨勢。

參照

EncTemper函數

EncTemper範例

是顯示所有軸溫度的範例

```
> EncTemper
        31.21                   32.82           '第1關節 第2關節
        31.45                   33.28           '第3關節 第4關節
        32.76                   32.87           '第6關節 第5關節
        -1000.00                -1000.00        '第7關節 第8關節
        -1000.00                                '第9關節
```

是顯示第1關節溫度的範例

```
> EncTemper 1
        31.21
```

3.9.11 EncTemper函數

返回編碼器目前溫度。

格式

EncTemper (關節編號)

參數

關節編號

以1~9整數值或運算式設定關節編號。
附加軸的S軸為8，T軸為9。

說明

返回編碼器目前溫度 [℃]。
指定未對應編碼器溫度的機器人或關節編號時，返回-1000。

提示

- 本命令中溫度讀取使用的感測器，並非已校正儀器。
- 請瀏覽相對的變化，用以掌握溫度變化的趨勢。

參照

EncTemper

EncTemper函數範例

是顯示讀取第1關節編碼器溫度的範例。

```
> Print EncTemper(1)
        -1000
```

3.9.12 Eof函數

用於傳回檔案的EOF(已開啟檔案的指標位於終端)。

格式

Eof (檔案編號)

參數

檔案編號

以30~63的整數值或運算式來指定。

傳回值

檔案為EOF時，傳回「True」；除此之外，傳回「False」。

說明

在讀出模式下開啟檔案時發揮其功能。

若用AOpen、WOpen命令開啟檔案，則會發生錯誤。

參照

Lof

Eof函數範例

```
Integer fileNum
String data$

fileNum = FreeFile
UOpen "TEST.DAT" As #fileNum
Do While Not Eof(fileNum)
    Line Input #fileNum, data$
    Print "data = ", data$
Loop
Close #fileNum
```

3.9.13 Era函數

用於傳回發生錯誤的關節之編號。

格式

Era [(工作編號)]

參數

工作編號

以整數指定0~32的任務編號。省略或為「0」時，則為目前工作。

傳回值

以如下0~9的整數值通知發生錯誤的關節編號。

- 0 - 目前錯誤的原因與關節無關。
- 1 - 目前錯誤的原因在於第1關節。
- 2 - 目前錯誤的原因在於第2關節。
- 3 - 目前錯誤的原因在於第3關節。
- 4 - 目前錯誤的原因在於第4關節。
- 5 - 目前錯誤的原因在於第5關節。
- 6 - 目前錯誤的原因在於第6關節。
- 7 - 目前錯誤的原因在於第7關節。
- 8 - 目前錯誤的原因在於第8關節(附加軸S)。
- 9 - 目前錯誤的原因在於第9關節(附加軸T)。

說明

發生錯誤時，Era用於找出發生錯誤的關節，並通知該關節編號。若關節並非造成目前錯誤的原因，則傳回「0」。

對於自動運轉模式(AUTO)的正常工作和NoPause工作，若在「自動運轉時發生錯誤」，則中斷執行並結束工作。在NoEmgAbort工作或背景工作中使用本函數時，若已結束對象工作，即發生「錯誤 2261」。在工作結束前若要取得資訊，請使用OnErr。

參照

Erl、Err、ErrMsg$、Ert、OnErr、Trap

Era函數範例

```
Function main
  OnErr Goto eHandler
  Do
    Call PickPlace
  Loop
  Exit Function
eHandler:
  Print "The Error code is ", Err
  Print "The Error Message is ", ErrMsg$(Err)
  errTask = Ert
  If errTask > 0 Then
    Print "Task number in which error occurred is ", errTask
    Print "The line where the error occurred is Line ", Erl(errTask)
    If Era(errTask) > 0 Then
      Print "Joint which caused the error is ", Era(errTask)
    EndIf
  EndIf
Fend
```

3.9.14 EResume

用於結束錯誤處理常式後重新執行程式。

格式

EResume [{標籤 | Next }]

說明

- EResume

  若在與錯誤處理常式相同的函式內發生錯誤，則利用造成錯誤原因的陳述式重新執行程式。

  若在被叫用的函式內發生錯誤，則利用包括錯誤處理常式在內的函式內Call陳述式，重新執行程式。

- EResume Next

  若在與錯誤處理常式相同的函式內發生錯誤，則利用造成錯誤原因的陳述式之下一個陳述式，重新執行程式。

  若在被叫用的函式內發生錯誤，則利用叫用包括錯誤處理常式在內的函式之最後Call陳述式的下一個陳述式，重新執行程式。

- EResume {標籤}

  若在與錯誤處理常式相同的函式內發生錯誤，則利用包括指定標籤在內的陳述式，重新執行程式。

參照

OnErr

EResume範例

```
Function main
  Integer retry

  OnErr GoTo eHandler
  Do
    RunCycle
  Loop
  Exit Function

eHandler:
  Select Err
    Case MyError
      retry = retry + 1
      If retry < 3 Then
        EResume '重新執行
      Else
        Print "MyError has occurred ", retry, " times
      EndIf
  Send
Fend
```

3.9.15 Erf$函數

用於傳回發生錯誤的函數名稱。

格式
Erf$ [(工作編號)]

參數

工作編號
以0~32的整數值指定任務編號。省略或為「0」時，則為目前工作。

傳回值
用於傳回最後發生錯誤的函數名稱。

說明
Erf$連同OnErr一起使用。Erf$用於傳回發生錯誤的函數名稱。可透過將Erf$與Err、Ert、Erl、Era等組合，針對發生的錯誤，蒐集更詳細的資訊。

對於自動運轉模式(AUTO)的正常工作和NoPause工作，若在「自動運轉時發生錯誤」，則中斷執行並結束工作。在NoEmgAbort工作或背景工作中使用本函數時，若已結束對象工作，即發生「錯誤 2261」。在工作結束前若要取得資訊，請使用OnErr。

參照
Era、Erl、Err、ErrMsg$、Ert、OnErr、Trap

Erf$函數範例
以下是調查下述內容的簡易程式。

- 哪一項工作發生錯誤(Ert函數)
- 哪一個函數發生錯誤(Erf$函數)
- 發生在何處(Erl函數)
- 哪一個關節發生錯誤(Era函數)

```
Function main
  OnErr Goto eHandler
  Do
    Call PickPlace
  Loop
  Exit Function
eHandler:
  Print "The Error code is ", Err
  Print "The Error Message is ", ErrMsg$(Err)
  errTask = Ert
  If errTask > 0 Then
    Print "Task number in which error occurred is ", errTask
    Print "Function at which error occurred is ", Erf$(errTask)
    Print "The line where the error occurred is Line ", Erl(errTask)
    If Era(errTask) > 0 Then
      Print "Joint which caused the error is ", Era(errTask)
    EndIf
  EndIf
Fend
```

3.9.16 Erl函數

用於傳回發生錯誤的行編號。

格式
Erl [(工作編號)]

參數

工作編號
以0~32的整數值指定任務編號。省略或為「0」時，則為目前工作。

傳回值
用於傳回最後發生錯誤的行編號。

說明
Erl連同OnErr一起使用。Erl用於傳回發生錯誤的行編號。可透過將Erl與Err、Ert、Era等組合，針對發生的錯誤，蒐集詳細的資訊。

對於自動運轉模式(AUTO)的正常工作和NoPause工作，若在「自動運轉時發生錯誤」，則中斷執行並結束工作。在NoEmgAbort工作或背景工作中使用本函數時，若已結束對象工作，即發生「錯誤 2261」。在工作結束前若要取得資訊，請使用OnErr。

參照
Era、Erf$、Err、ErrMsg$、Ert、OnErr

Erl函數範例
以下是調查下述內容的簡易程式。

- 哪一項工作發生錯誤(Ert函數)
- 發生在何處(Erl函數)
- 發生何種錯誤(Err函數)
- 哪一個關節發生錯誤(Era函數)

```
Function main
  OnErr Goto eHandler
  Do
    Call PickPlace
  Loop
  Exit Function
eHandler:
  Print "The Error code is ", Err
  Print "The Error Message is ", ErrMsg$(Err)
  errTask = Ert
  If errTask > 0 Then
    Print "Task number in which error occurred is ", errTask
    Print "The line where the error occurred is Line ", Erl(errTask)
    If Era(errTask) > 0 Then
      Print "Joint which caused the error is ", Era(errTask)
    EndIf
  EndIf
Fend
```

3.9.17 Err函數

用於傳回最新的錯誤狀態。

格式

Err [(工作編號)]

參數

工作編號

以0~32的整數值指定任務編號。"「0」用於指定目前的工作。可省略。

傳回值

用於以整數值傳回錯誤代碼。

說明

Err函數用於將目前的錯誤代碼通知給使用者。在發揮SPEL+錯誤應對功能的同時，通知發生何種錯誤，以便進行適當的應對。Err連同OnErr一起使用。

若要取得控制器的錯誤，使用SysErr函數。

對於自動運轉模式(AUTO)的正常工作和NoPause工作，若在「自動運轉時發生錯誤」，則中斷執行並結束工作。在NoEmgAbort工作或背景工作中使用本函數時，若已結束對象工作，即發生「錯誤 2261」。在工作結束前若要取得資訊，請使用OnErr。

參照

Era、Erf$、Erl、ErrMsg$、EResume、Ert、OnErr、Return、SysErr

Err範例

以下範例是確認有無點P0-P399的簡易公用程式。若無點，則在畫面上顯示通知使用者的訊息。使用CX命令測試有無定義各點。如有未定義的點，控制則移往錯誤處理，並在畫面上顯示未定義的點。

```
Function errtest
   Integer i, errnum
   Real x

   OnErr GoTo eHandle
   For i = 0 To 399
     x = CX(P(i))
   Next i
   Exit Function
'
'
'*******************************************
'* Error Handler                           *
'*******************************************
eHandle:
   errnum = Err
   '確認有無使用未定義的點
   If errnum = 78 Then
      Print "Point number P", i, " is undefined!"
   Else
      Print "ERROR: Error number ", errnum, " Occurred."
   EndIf
   EResume Next
Fend
```

3.9.18 Errb函數

用於傳回發生錯誤的機器人之編號。

格式

Errb

傳回值

用於傳回發生錯誤的機器人之編號。

說明

發生錯誤時，Errb函數用於找出發生錯誤的機器人，並通知該機器人的編號。若機器人並非造成目前錯誤的原因，則傳回「0」。

參照

Era、Erl、Err、ErrMsg$、OnErr、Trap

Errb函數範例

在以下程式範例中顯示下述內容。

- 哪一項工作發生錯誤(Ert函數)
- 發生在何處(Erl函數)
- 發生何種錯誤(Err函數)
- 哪一個關節發生錯誤(Era函數)
- 哪一個機器人發生錯誤(Errb函數)

```
Function main
  OnErr Goto eHandler
  Do
    Call PickPlace
  Loop
  Exit Function
eHandler:
  Print "The Error code is ", Err
  Print "The Error Message is ", ErrMsg$(Err)
  errTask = Ert
  If errTask > 0 Then
    Print "Task number in which error occurred is ", errTask
    Print "The line where the error occurred is Line ", Erl(errTask)
    If Era(errTask) > 0 Then
      Print "Joint which caused the error is ", Era(errTask)
    EndIf
  Print "Robot number in which error occurred is ", errb
  EndIf
Fend
```

3.9.19 ErrMsg$函數

用於傳回指定錯誤編號的錯誤訊息。

格式

ErrMsg$ (錯誤編號, 語言編號)

參數

錯誤編號

以整數值指定要傳回訊息的錯誤編號。

語言編號

以下列整數值指定語言。可省略。

- 0 - 英語
- 1 - 日語
- 2 - 德語
- 3 - 法語
- 4 - 中文(簡體)
- 5 - 中文(繁體)
- 6 - 西班牙語

省略時被指定為英語。

傳回值

用於傳回錯誤代碼表的錯誤訊息。

參照

Era、Erl、Err、Ert、OnErr、Trap

ErrMsg$函數範例

以下是調查下述內容的簡易程式。

- 哪一項工作發生錯誤(Ert函數)
- 發生在何處(Erl函數)
- 關節是否發生錯誤(Era函數)

```
Function main
  OnErr Goto eHandler
  Do
    Call PickPlace
  Loop
  Exit Function
eHandler:
  Print "The Error code is ", Err
  Print "The Error Message is ", ErrMsg$(Err)
  errTask = Ert
  If errTask > 0 Then
    Print "Task number in which error occurred is ", errTask
    Print "The line where the error occurred is Line ", Erl(errTask)
    If Era(errTask) > 0 Then
      Print "Joint which caused the error is ", Era(errTask)
    EndIf
  EndIf
Fend
```

3.9.20 Error

發生使用者定義錯誤。

格式

(1) Error 工作編號, 錯誤編號

(2) Error 錯誤編號

參數

工作編號

以0~32的整數值指定任務編號。"「0」用於指定目前的工作。可省略。

錯誤編號

以整數值指定錯誤編號。使用者錯誤編號範圍為8000~8999。

說明

Error陳述式用於發生系統或使用者定義錯誤。可使用Epson RC+開發環境的使用者錯誤編輯器，來定義使用者錯誤的標籤和記述。

參照

Era、Erl、Err、OnErr

Error範例

```
 #define ER_VAC 8000

If Sw(vacuum) = Off Then
    Error ER_VAC
EndIf
```

3.9.21 ErrorOn函數

用於傳回控制器的錯誤狀態。

格式

ErrorOn

傳回值

處於錯誤狀態時，傳回「True」；除此以外，傳回「False」。

說明

本函數僅用於NoEmgAbort工作(在Xqt時，指定NoEmgAbort開始的特殊工作)和背景工作。

參照

ErrorOn、SafetyOn、SysErr、Wait、Xqt

ErrorOn函數範例

以下範例是由控制器監視錯誤狀態，若發生錯誤，則依錯誤編號啟用/停用I/O的程式。

注意

- Forced旗標

  在本程式範例中，在ON/OFF命令中指定Forced旗標。

  在發生錯誤時、緊急停止時或打開安全門時，I/O輸出會發生變化，因此在系統設計上需加以注意。

- 發生錯誤後的處理

  如本範例所示，在發生錯誤時執行必要的處理之後，請迅速結束工作。

```
Function main

Xqt ErrorMonitor, NoEmgAbort
:
:
Fend

Function ErrorMonitor
  Wait ErrorOn
  If 4000 < SysErr Then
    Print "Mortion Error = ", SysErr
    Off 10, Forced
    On 12, Forced
  Else
    Print "Other Error = ", SysErr
    Off 11, Forced
    On 13, Forced
  EndIf

Fend
```

3.9.22 Ert函數

用於傳回發生錯誤的工作編號。

格式
Ert

傳回值
用於傳回發生錯誤的工作編號。

說明
Ert函數用於取得發生錯誤的工作。

傳回未發生錯誤的工作(0)、正常工作(1~32)、背景工作(65~80)、TRAP工作(257~267)的編號。

參照
Era、Erl、Err、ErrMsg$、OnErr、Trap

Ert函數範例
在以下程式範例中顯示下述內容。

- 哪一項工作發生錯誤(Ert函數)
- 發生在何處(Erl函數)
- 哪一個關節發生錯誤(Era函數)

```
Function main
  OnErr Goto eHandler
  Do
    Call PickPlace
  Loop
  Exit Function
eHandler:
  Print "The Error code is ", Err
  Print "The Error Message is ", ErrMsg$(Err)
  errTask = Ert
  If errTask > 0 Then
    Print "Task number in which error occurred is ", errTask
    Print "The line where the error occurred is Line ", Erl(errTask)
    If Era(errTask) > 0 Then
      Print "Joint which caused the error is ", Era(errTask)
    EndIf
  EndIf
Fend
```

3.9.23 EStopOn函數

用於傳回緊急停止狀態。

格式
EstopOn

傳回值
處於緊急停止狀態時，傳回「True」；除此以外，傳回「False」。

說明
本函數僅用於NoEmgAbort工作(在Xqt時，指定NoEmgAbort開始的特殊工作)。

參照
ErrorOn、SafetyOn、Wait、Xqt

EstopOn函數範例
以下範例是由控制器監視緊急停止，若有發生緊急停止，則啟用/停用I/O的程式。

注意

- Forced旗標

  在本程式範例中，在ON/OFF命令中指定Forced旗標。

  在發生錯誤時、緊急停止時或打開安全門時，I/O輸出會發生變化，因此在系統設計上需加以注意。

- 發生錯誤後的處理

  如本範例所示，在發生錯誤時執行必要的處理之後，請迅速結束工作。

- 以緊急停止停用輸出連接埠

  如本範例所示，若工作在緊急停止後仍執行I/O的啟用/停用，建議取消[設定]-[系統設定]-[控制器]-[環境設定]-[以緊急停止停用輸出連接埠]的勾選狀態。若此設定處於勾選狀態，則無法確定優先執行的動作(以控制器停用I/O或以工作啟用I/O)。

```
Function main

  Xqt EStopMonitor, NoEmgAbort
  :
  :
Fend

Function EStopMonitor
  Wait EStopOn
  Print "EStop !!!"
  Off 10, Forced
  On 12, Forced
Fend
```

3.9.24 Eval函數

用於執行命令視窗的陳述式，並傳回錯誤狀態。

格式

Eval (命令 [, 命令的輸出結果])

參數

命令

以字串指定要執行的命令。

命令的輸出結果

指定用於存放命令輸出結果的字串變數。可省略。命令錯誤時，則傳回「!Error: 錯誤代碼」。輸出結果超過255個字元時，捨去超出的部分。

傳回值

用於傳回透過執行命令而返回的錯誤代碼。

即便命令執行出錯，本函數本身也不發生錯誤。此外，在系統記錄檔中也不留下錯誤。

命令正常結束時，則傳回「0」。

說明

若使用Eval函數，則可利用TCP/IP等通信連接埠執行任何命令。
執行的命令為可透過命令視窗執行的命令。
命令字串中可使用雙引號(")。

相較於執行平常的陳述式，執行本函數需要更多的處理時間。

使用命令的輸出結果參數，以獲取來自命令的傳回值。例如，對「Print Sw (1)」命令，傳回命令輸出結果「1」或「0」。

參照

錯誤訊息一覽

Eval函數範例

在以下範例中，表示如何執行用RS-232C讀取的命令。執行命令後，錯誤代碼會被傳回到主機。例如，從主機傳送「motor on」等命令。

```
Integer errCode
String cmd$

OpenCom #1
Do
  Line Input #1, cmd$
  errCode = Eval(cmd$)
  Print #1, errCode
Loop
```

3.9.25 Exit

用於強制終止迴圈或函式。

格式

Exit { Do | For | Function }

說明

如下所述為Exit陳述式的格式。

Exit Do

用於離開Do...Loop陳述式。這只能在Do...Loop陳述式內使用。使用Exit Do將控制轉移到Loop陳述式的下一個陳述式。若在巢狀Do...Loop陳述式中使用，控制則被轉移到Exit Do的某個迴圈的上1級迴圈中。

Exit For

用於離開For迴圈。只可在For...Next迴圈內使用。Exit For用於將控制轉移到Next陳述式的下一個陳述式。若在巢狀For迴圈中使用，控制則被轉移到Exit For的某個迴圈的上1級迴圈中。

Exit Function

用於在任意位置離開函數。在已叫用函式的陳述式之下一個陳述式中繼續執行程式。

參照

Do...Loop、For...Next、Function...Fend

Exit範例

```
For i = 1 To 10
    If Sw(1) = On Then
        Exit For
    EndIf
    Jump P(i)
Next i
```

3.9.26 ExportPoints

用於將點檔案匯出到指定路徑中。

格式

ExportPoints檔名, 儲存目的地

參數

檔案名稱

表示欲導出之特定檔案的字串運算式副檔名為「.pts」。不可指定路徑。此外，不受ChDisk等的影響。詳細內容請參閱ChDisk。

儲存目的地

指定儲存目的地的路徑和檔案名稱。副檔名為「.pts」。詳細內容請參閱ChDisk。

說明

ExportPoints用於將指定的點檔案複製到PC上的資料夾中。在PC上的資料夾中若已存在相同檔案，則進行覆寫。

常見錯誤

- 不存在儲存目的地路徑時

  不存在指定的儲存目的地路徑時，會發生錯誤。

- 找不到指定檔案時

  檔名中含有路徑時會發生錯誤。

參照

Dir、LoadPoints、SavePoints、FileExists、FolderExists

ExportPoints範例

```
Function main
  LoadPoints "robot1.pts"
  :
  SavePoints "robot1.pts"
  If FolderExists("c:\mypoints\") Then
    ExportPoints "robot1.pts", "c:\mypoints\model1.pts"
  EndIf
Fend
```

3.10 F

3.10.1 FbusIO_GetBusStatus函數

用於傳回指定現場匯流排的狀態。

格式

FbusIO_GetBusStatus (匯流排編號)

參數

匯流排編號

是表示現場匯流排系統編號的整數運算式此編號務必為16，並且是連接現場匯流排主站(位於控制器的PC端)的匯流排ID。

傳回值

- 0 - OK
- 1 - 未連接
- 2 - 電源OFF

說明

FbusIO_GetBusStatus函數可用於確認現場匯流排的狀態。

注意

僅限於啟用現場匯流排主選項時，方可使用此命令。

參照

FbusIO_GetDeviceStatus、FbusIO_SendMsg

FbusIO_GetBusStatus函數範例

```
Long sts
sts = FbusIO_GetBusStatus(16)
```

3.10.2 FbusIO_GetDeviceStatus函數

用於傳回指定現場匯流排裝置的狀態。

格式

FbusIO_GetDeviceStatus (匯流排編號, 裝置ID)

參數

匯流排編號

是表示現場匯流排系統編號的整數運算式此編號務必為16，並且是連接現場匯流排主站(位於控制器的PC端)的匯流排ID。

設備ID

是表示設備的現場匯流排ID的整數運算式

傳回值

- 0 - OK
- 1 - 未連接
- 2 - 電源OFF
- 3 - 同步錯誤: 裝置處於初始化狀態，或裝置的傳輸速率不正確。

說明

FbusIO_GetDeviceStatus函數可用於確認現場匯流排裝置的狀態。

注意

僅限於啟用現場匯流排主選項時，方可使用此命令。

參照

FbusIO_GetBusStatus、FbusIO_SendMsg

FbusIO_GetDeviceStatus函數範例

```
Long sts
sts = FbusIO_GetDeviceStatus(16, 10)
```

3.10.3 FbusIO_SendMsg

用於將訊息傳送到現場匯流排I/O裝置並傳回回覆。

格式

FbusIO_SendMsg (匯流排編號, 裝置ID, msgParam, sendData (), recvData ())

參數

匯流排編號
　是表示現場匯流排系統編號的整數運算式此編號務必為16，並且是連接現場匯流排主站(位於控制器的PC端)的匯流排ID。
設備ID
　是表示設備的現場匯流排ID的整數運算式
msgParam
　表示訊息參數的整數運算式不可用於DeviceNet。
sendData
　以Byte型陣列指定傳送到設備的資料。此陣列的維度必須與傳送的位元組數的維度相同。未傳送資料時，指定「0」。
recvData
　以Byte型陣列指定從設備接收的資料。此陣列會自動轉換為與接收的位元組數相應的維數。

說明

FBusIO_SendMsg用作對現場匯流排I/O裝置的查詢。關於支援訊息，請洽設備製造商。

注意

僅限於啟用現場匯流排主選項時，方可使用此命令。

參照

FbusIO_GetBusStatus、FbusIO_GetDeviceStatus

FbusIO_SendMsg範例

```
' 將明確訊息傳送到DeviceNet裝置中
Byte sendData(5)
Byte recvData(0)
Integer i

sendData(0) = &H0E  ' 命令
sendData(1) = 1    ' 等級
sendData(3) = 1     ' 執行個體
sendData(5) = 7     ' 屬性
' DeviceNet的msgParam為「0」
FbusIO_SendMsg 16, 1, 0, sendData(), recvData()
' 顯示回覆
For i = 0 to UBound(recvData)
  Print recvData(i)
Next i

' 將訊息傳送到Profibus裝置中
Byte recvData(0)
Integer i

' msgParam為服務編號
FbusIO_SendMsg 16, 1, 56, 0, recvData()
' 顯示回覆
```

```
For i = 0 to UBound(recvData)
  Print recvData(i)
Next i
```

3.10.4 FileDateTime$函數

用於傳回檔案的日期與時間。

格式

FileDateTime$ (檔名)

參數

檔案名稱

以字串指定要確認的檔案名稱。包括驅動器名稱和路徑名稱。僅指定檔名時，即指目前目錄中的檔案。詳細內容請參閱ChDisk。

注意

可使用網路路徑。

傳回值

用於以下列格式傳回檔案的最後修正日期與時間。

月/日/西元年號 小時:分:秒

參照

FileExists、FileLen

FileDateTime$函數範例

```
String myPath$
myPath$ = "c:\TEST\TEST.DAT"

If FileExists(myPath$) Then
    Print "Last access date and time: ", FileDateTime$(myPath$)
    Print "Size: ", FileLen(myPath$)
EndIf
```

3.10.5 FileExists函數

用於檢查有無檔案。

格式

FileExists (檔名)

參數

檔案名稱

以字串指定要確認的檔案名稱。包括驅動器名稱和路徑名稱。僅指定檔名時，即指目前目錄中的檔案。詳細內容請參閱ChDisk。

注意

可使用網路路徑。

傳回值

- 有檔案時：True
- 沒有檔案時：False

參照

FolderExists、FileLen、FileDateTime$

FileExists函數範例

```
String myPath$
myPath$ = "c:\TEST\TEST.DAT"

If FileExists(myPath$) Then
    Print "Last access date and time: ", FileDateTime$(myPath$)
    Print "Size: ", FileLen(myPath$)
EndIf
```

3.10.6 FileLen函數

用於傳回檔案大小。

格式

FileLen (檔名)

參數

檔案名稱

以字串指定要確認的檔案名稱。包括驅動器名稱和路徑名稱。僅指定檔名時，即指目前目錄中的檔案。詳細內容請參閱ChDisk。

注意

可使用網路路徑。

傳回值

用於傳回檔案的位元組數。

參照

FileDateTime$、FileExists

FileLen函數範例

```
String myPath$
myPath$ = "c:\TEST\TEST.DAT"

If FileExists(myPath$) Then
    Print "Last access date and time: ", FileDateTime$(myPath$)
    Print "Size: ", FileLen(myPath$)
EndIf
```

3.10.7 Find

用於設定和顯示在動作命令中儲存坐標的條件。

格式

Find [條件運算式]

參數

事件條件運算式

用於指定作為觸發器的輸入狀態。

[事件] 比較運算符(=, <>, >=, >, <, <=) [整數運算式]

可將以下函數或變數用於事件。

- 函數：Sw、In、InW、Oport、Out、OutW、MemSw、MemIn、MemInW、Ctr、GetRobotInsideBox、GetRobotInsidePlane、AIO_In、AIO_InW、AIO_Out、AIO_OutW、Hand_On、Hand_Off、SF_GetStatus
- 變數：Byte、Int32、Integer、Long、Short、UByte、UInt32、UShort型的備份變數、全域變數、模組變數

此外，可用以下運算符，對複數事件條件運算式使用遮罩或進行複合組合。

- 運算符：And、Or、Xor

[例]

```
Sense Sw(5) = On
Sense Sw(5) = On And Sw(6) = Off
```

說明

請單獨記述Find陳述式，或作為動作命令陳述式的修飾詞進行記述。

必須在Find條件運算式中包含1個以上的上述函數。

Find條件運算式中含有變數時，則在設定Find條件時運算該值。可能會變成非預期條件，因此建議在條件運算式中不使用變數。還可記述數個Find陳述式。屆時，最後執行的Find條件會處於啟用狀態。

若省略參數，則顯示目前Find設定。

注意

- 電源ON時的Find設定

  電源ON時的Find條件之初始設定為Find Sw (0) = On。設為當輸入位元編號0為ON時儲存坐標。

- 用於檢查Find條件成立的PosFound函數

  在執行使用Find修飾詞的動作命令之後，可用PosFound函數檢查Find條件是否成立。

- 在條件運算式中使用變數時
  - 可使用的變數型態為整數型(Byte, Int32, Integer, Long, Short, UByte, UInt32, UShort)。
  - 不可使用陣列變數。
  - 不可使用本地變數。
  - 變數值未滿足條件的時間超過0.01秒時，系統可能無法檢測到變數變化。
  - 系統內可使用的變數等待數量有限。1個系統內可使用的變數等待數量最多64個(也包括Wait等條件運算式所用的變數等待數量)。若超過最大數量，則在建置專案時將發生錯誤。
  - 若以ByRef傳址要執行變數等待的變數，則發生錯誤。
  - 若條件運算式右邊的整數運算式中含有變數，則在開始動作命令時運算該值。可能會變成非預期條件，因此建議在整數運算式中不使用變數。

參照

FindPos、Go、Jump、PosFound、SF_GetStatus

Find範例

```
Find Sw(5) = On
Go P10 Find
If PosFound Then
    Go FindPos
Else
    Print "Cannot find the sensor signal."
EndIf
```

3.10.8 FindPos函數

用於在執行動作命令中傳回以Find儲存的坐標。

格式

FindPos

傳回值

用於在執行動作命令中傳回以Find儲存的坐標。

參照

Find、Go、Jump、PosFound、CurPos、InPos

FindPos函數範例

```
Find Sw(5) = On
Go P10 Find
If PosFound Then
    Go FindPos
Else
    Print "Cannot find the sensor signal."
EndIf
```

3.10.9 Fine

用於設定和顯示結束目標位置定位的判斷範圍。

格式

(1) Fine 第1關節設定值, 第2關節設定值, 第3關節設定值, 第4關節設定值 [, 第5關節設定值, 第6關節設定值] [, 第7關節設定值] [, 第8關節設定值, 第9關節設定值]

(2) Fine

參數

第1關節設定值
　　以0~65535的整數指定第1關節的定位容許範圍。
第2關節設定值
　　以0~65535的整數指定第2關節的定位容許範圍。
第3關節設定值
　　以0~65535的整數指定第3關節的定位容許範圍。
第4關節設定值
　　以0~65535的整數指定第4關節的定位容許範圍。
第5關節設定值
　　以0~65535的整數指定第5關節的定位容許範圍。可省略。僅用於垂直6軸型機器人(包含N系列)。
第6關節設定值
　　以0~65535的整數指定第6關節的定位容許範圍。可省略。僅用於垂直6軸型機器人(包含N系列)。
第7關節設定值
　　以0~65535的整數指定第7關節的定位容許範圍。可省略。僅用於關節型7軸機器人。
第8關節設定值
　　以0~65535的整數指定第8關節的定位容許範圍。可省略。僅用於附加軸S關節。
第9關節設定值
　　以0~65535的整數指定第9關節的定位容許範圍。可省略。僅用於附加軸T關節。

*若為C8、C12系列，定位容許範圍為0~131070的整數。

結果

未指定參數時，Fine則用於顯示各關節的目前設定值。

說明

Fine屬於在結束動作時確認各關節定位的動作命令，並指定對指定位置判斷為結束定位的容許範圍。

在結束從CPU對伺服系統傳送目標坐標脈衝後，開始結束定位的判斷。機器人因伺服延遲而在該階段未到達目標位置。在各關節的設定容許範圍之內，每隔數毫秒進行一次判斷。若所有關節皆在指定的容許範圍之內，則結束定位。結束定位後，程式控制會轉移到下一個陳述式。但是，伺服系統會繼續將機器人控制到目標位置。

若以Fine命令指定較大範圍，則在較早階段結束定位，並進入下一個陳述式。

Fine的預設值因機器人類型而異。有關詳細資訊，請參閱以下手冊。
「機械手手冊 」

注意

- 週期時間和Fine命令

  Fine值本身並不影響手臂的加速或減速控制，但是，若設定精細的Fine值，伺服則需要多一點的時間(數毫秒)才能達成該容許範圍。系統的週期時間會按該時間延遲。手臂被定位於Fine命令設定的容許範圍內時，CPU將執行下一個命令。

- Fine的初始化(依據Motor On、SLock、SFree等命令)

  若使用以下任一命令，Fine的設定值則被初始化為預設值。
  SLock, SFree, Motor

  請務必在執行這些命令之後重新設定Fine值。

常見錯誤

若2秒內未完成Fine定位，則發生錯誤代碼4024。此錯誤通常是指需調整伺服系統的平衡。

參照

Accel、AccelR、AccelS、Arc、Go、Jump、Move、Speed、SpeedR、SpeedS、Pulse、FineDist、FineStatus

Fine範例

以下程式是以程式函式執行Fine的範例，以及在監視器視窗執行Fine的範例。

```
Function finetest
    Fine 5, 5, 5, 5     '將精度設為+/-5脈衝
    Go P1
    Go P2
Fend


> Fine 10, 10, 10, 10
>
> Fine
10, 10, 10, 10
```

3.10.10 Fine函數

用於傳回指定關節的Fine設定。

格式

Fine (關節編號)

參數

關節編號

以整數值指定讀取Fine設定的關節編號。附加軸的S軸為8，T軸為9。

傳回值

用於傳回實數值。

參照

Accel、AccelS、Arc、Go、Jump、Move、Speed、SpeedS、Pulse

Fine函數範例

以下為使用Fine函數的程式範例。

```
Function finetst
    Integer a
    a = Fine(1)
Fend
```

3.10.11 FineDist

用於設定和顯示結束目標位置定位的判斷範圍。設定值以mm為單位。

格式

(1) FineDist 設定值

(2) FineDist

參數

設定值

定位容許範圍為0.001[mm]~10[mm]。

結果

未指定參數時，FineDist則用於顯示目前設定值。

注意

- 支援的控制器型號

  不支援T/VT系列。

- 關於Fine和FineDist

  Fine和FineDist的差異在於機器人結束動作時的定位判斷單位。

- Fine時，可以pulse設定定位判斷值，以進行各軸的定位判斷。

- FineDist時，可以mm設定定位判斷值，並在工具編號0的坐標系上進行定位判斷。

無法同時使用Fine和FineDist。如下所示，若將Fine和FineDist兩個用於程式，則以FineDist進行定位判斷。

(Fine和FineDist的順序相反時，則以Fine進行定位判斷。)

```
Function test
    Fine 5, 5, 5, 5
    FineDist 0.1

    Go P1
    Go P2
Fend
```

注意

- FineDist的初始化(依據Motor On、SLock、SFree等命令)

  若使用以下任一命令，FineDist的設定值則被初始化為預設值，並以Fine進行定位判斷。
  SLock, SFree, Motor

  請務必在執行這些命令之後重新設定FineDist值。

常見錯誤

若2秒內未完成FineDist定位，則發生錯誤代碼4024。此錯誤通常是指需調整伺服系統的平衡。

參照

Accel、AccelR、AccelS、Arc、Go、Jump、Move、Speed、SpeedR、SpeedS、Pulse、Fine、FineStatus

FineDist範例

以下程式是以程式函式執行FineDist的範例，以及在監視器視窗執行Fine的範例。

```
Function fineDisttest
    Fine 0.1  '將精度設為+/-0.1mm
    Go P1
    Go P2
Fend


> FineDist 0.1
>
> FineDist
0.1
```

3.10.12 FineStatus函數

用於以整數值傳回使用Fine或FineDist。

格式

FineStatus

傳回值

用於以整數值傳回使用Fine或FineDist。

- 0 = 正在使用Fine
- 1 = 正在使用FineDist

參照

Fine、FineDist

FineStatus函數範例

```
Print FineStatus
```

3.10.13 Fix函數

用於從實數值取出整數部分。

格式

Fix (數值)

參數

數值

用於指定實數值。

傳回值

用於傳回以參數設定的實數值之整數值。

參照

Int

Fix函數範例

```
>print Fix(1.123)
 1
>
```

3.10.14 Flush

用於將緩衝區寫入檔案。

格式

Flush #檔案編號

參數

檔案編號

以30~63的整數值或運算式來指定。

說明

將緩衝區寫入目前開啟的檔案。

不可使用以ROpen命令開啟的檔案。

Flush範例

```
Integer fileNum, i

fileNum = FreeFile
UOpen "TEST.DAT" As #fileNum
For i = 0 To 100
   Print #fileNum, i
Next i
Flush #fileNum
Close #fileNum
```

3.10.15 FmtStr

對數值運算式或日期/時間運算式進行格式化。

格式

FmtStr 用於指定格式的運算式, 用於指定格式的字串, 輸出字串變數

參數

格式指定運算式

指定格式指定數值運算式，或日期/時間運算式。
請以「yyyy/mm/dd hh:nn:ss」形式指定日期/時間運算式。

格式指定字串

指定格式指定字串。

輸出字串變數

指定輸出字串變數。

說明

依據格式設定字串傳回設定格式的字串。

數值格式指定子

None

在未格式化的狀態之下直接顯示編號。

(0)

是表示位數位置的位數指標。顯示數值或零。將數值運算式所含之十進位顯示於格式指定字串的「0」位置。在除此以外的情況下，在該位置上顯示「0」。數值位數少於格式指定字串(小數點之前或之後的)「0」的個數時，開頭乃至之後則顯示數個「0」。若數值的小數點右側位數多於格式指定字串小數點右側的「0」的個數，則配合「0」的個數，對數值位數進行四捨五入處理。此外，數值的小數點左側位數多於格式指定字串小數點左側的「0」的個數時，則按原樣直接顯示其餘的位數。

(#)

是表示位數位置的位數指標。用於顯示數值或不顯示任何內容。在格式指定字串的「#」位置上顯示數值運算式的十進位，除此以外情況下，該位置不顯示任何內容。此字元(#)雖具有0位數指標這樣的作用，但數值位數同於或少於格式指定字串小數點左右的「#」的個數時，開頭乃至之後則不顯示「0」。

(.)

用於顯示小數點位置，並設定小數點左右顯示的位數。也有可能依本地設定而用逗號標示小數點。在格式指定字串中，此 (.) 的左側只有數字時，則在開頭處開始和顯示小於數字1的小數點。若要顯示開頭的零，則在小數點左側使用「0」。對於以格式指定輸出的值來表示小數點的字元，則依據使用的Windows所識別的數值格式。

(,)

是用1000分隔數值的千位分隔符號。依本地設定使用句號。對小數點左側有4位數以上的數值，使用每隔一千的分隔方法。標準用法為，格式指定字串包括在左右帶有位數指標(0或#)的千位分隔符號(,)時，指定格式。無論有無指定小數，緊鄰小數點的左側若排列2個千位分隔符號或單獨使用，則表示指定「該數值除以1000，並在必要時進行四捨五入處理」這樣的格式。例如，格式指定字串「##0,,」表示以「100」顯示為1億。小於百萬的數值則顯示為「0」。在緊鄰小數點左側以外的位置上，若並排放置2個千位分隔符號，則只用於分隔千位。在格式指定的輸出值中實際用作千位分隔符號的字元依據使用的Windows所識別的數值格式。

日期/時間格式指定子

(:)

是時間分隔符號。可能依本地設定而使用其它字元。若以格式指定時間值，時間分隔符號則用於對小時、分鐘、秒的值進行分隔。對於以格式指定輸出而實際使用的字元，會依Windows的設定而定。

(/)

是日期分隔符號。可能依本地設定而使用其它字元。若以格式指定日期值，日期分隔符號則用於對日、月、年的值進行分隔。對於以格式指定輸出而實際使用的字元，會依Windows的設定而定。

c

以ddddd顯示日期，以ttttt顯示時間，依此順序顯示。日期序號上若無分數部分，則只顯示日期；時間訊息上若無整數部分，則只顯示時間訊息。

d
以開頭不附加零的方式顯示日期。(1～31)

dd
以開頭附加零的方式顯示日期。(01～31)

ddd
省略星期顯示。(Sun～Sat)

dddd
不省略星期顯示。(Sunday～Saturday)

ddddd
依照Windows的Short資料顯示設定的格式，顯示所有日期、月、年。Windows的Short資料格式預設為m/d/yy。

dddddd
將日期的序列數值作為日期、月、年，並依Windows的Long資料顯示設定格式進行顯示。Windows的Long資料格式預設為mmmm dd, yyyy。

w
以數值顯示星期。(1：星期日~7：星期六)

ww
以數值顯示1年當中的第幾週。(1～54)

m
以開頭不附加零的數值顯示月份。(1～12)
即便在「h」或「hh」之後使用，也不顯示時間的分鐘。若要顯示時間的分鐘，請使用「n」或「nn」。

mm
以開頭附加零的數值顯示月份。(01～12)
即便在「h」或「hh」之後使用，也不顯示時間的分鐘。若要顯示時間的分鐘，請使用「n」或「nn」。

mmm
省略月份名稱顯示。(Jan～Dec)

mmmm
不省略月份名稱顯示。(January～December)

q
以數值顯示季度。(1～4)

y
以數值顯示1年當中的第幾日。(1～366)

yy
以2位數顯示年號。(00～99)

yyyy
以4位數顯示年號。(100～9999)

h
以開頭不附加零的方式顯示24小時制的時間。(0～23)

hh
以開頭附加零的方式顯示24小時制的時間。(00～23)

n
以開頭不附加零的方式顯示時間的分鐘。(0～59)

nn
以開頭附加零的方式顯示時間的分鐘。(00～59)

s
以開頭不附加零的方式顯示時間的秒鐘。(0～59)

ss
以開頭附加零的方式顯示時間的秒鐘。(00～59)

ttttt
依照Windows設定的時間分隔符號格式，顯示時間(小時、分鐘、秒鐘)。"若選擇「開頭的零」選項，則以開頭附加零的時間制顯示10:00am/pm之前的時間。Windows的預設時間格式為h:nn:ss。

AM/PM
以12小時制顯示時間，並以AM/PM(以大寫標記)顯示上午/下午。

am/pm
以12小時制顯示時間，並以am/pm(以小寫標記)顯示上午/下午。

A/P
以12小時制顯示時間，並以A/P(以大寫標記)顯示上午/下午。

a/p
以12小時制顯示時間，並以a/p(以小寫標記)顯示上午/下午。

AMPM
以12小時制顯示時間。依照Windows內的格式設定，上午時間以AM字串顯示，下午時間以PM字串顯示。AM/PM雖不拘大小寫，但Windows的設定和指定字串務必一致。Windows的預設為AM/PM。

注意

- 混合數值格式指定子和日期/時間格式指定子

  若混合指定數值格式指定子和日期/時間格式指定子，會發生錯誤。

參照

Left$、Right$、Str$

FmtStr範例

```
Function SaveData

    String d$, f$, t$

    ' 以月、日、小時、分鐘的形式建立檔名
    d$ = Date$
    t$ = Time$
    d$ = d$ + " " + t$
    FmtStr d$, "mmddhhnn", f$
    f$ = f$ + ".dat"
    WOpen f$ as #30
    Print #30, "data"
    Close #30
Fend
```

3.10.16 FmtStr$函數

對數值運算式或日期/時間運算式進行格式化。

格式

FmtStr$ (格式指定運算式, 格式指定字串)

參數

格式指定運算式

指定格式指定數值運算式或日期/時間運算式。

請以「yyyy/mm/dd hh:nn:ss」形式指定日期/時間運算式。

格式指定字串

指定格式指定字串。

傳回值

用於傳回格式指定的字串。

說明

依據格式設定字串傳回設定格式的字串。

數值格式指定子

None

在未格式化的狀態之下直接顯示編號。

(0)

是表示位數位置的位數指標。顯示數值或零。將數值運算式所含之十進位顯示於格式指定字串的「0」位置。在除此以外的情況下，在該位置上顯示「0」。數值位數少於格式指定字串(小數點之前或之後的)「0」的個數時，開頭乃至之後則顯示數個「0」。若數值的小數點右側位數多於格式指定字串小數點右側的「0」的個數，則配合「0」的個數，對數值位數進行四捨五入處理。此外，數值的小數點左側位數多於格式指定字串小數點左側的「0」的個數時，則按原樣直接顯示其餘的位數。

(#)

是表示位數位置的位數指標。用於顯示數值或不顯示任何內容。在格式指定字串的「#」位置上顯示數值運算式的十進位，除此以外情況下，該位置不顯示任何內容。此字元(#)雖具有0位數指標這樣的作用，但數值位數同於或少於格式指定字串小數點左右的「#」的個數時，開頭乃至之後則不顯示「0」。

(.)

用於顯示小數點位置，並設定小數點左右顯示的位數。也有可能依本地設定而用逗號標示小數點。在格式指定字串中，此 (.) 的左側只有數字時，則在開頭處開始和顯示小於數字1的小數點。若要顯示開頭的零，則在小數點左側使用「0」。對於以格式指定輸出的值來表示小數點的字元，則依據使用的Windows所識別的數值格式。

(,)

是用1000分隔數值的千位分隔符號。依本地設定使用句號。對小數點左側有4位數以上的數值，使用每隔一千的分隔方法。標準用法為，格式指定字串包括在左右帶有位數指標(0或#)的千位分隔符號(,)時，指定格式。無論有無指定小數，緊鄰小數點的左側若排列2個千位分隔符號或單獨使用，則表示指定「該數值除以1000，並在必要時進行四捨五入處理」這樣的格式。例如，格式指定字串「##0,,」表示以「100」顯示為1億。小於百萬的數值則顯示為「0」。在緊鄰小數點左側以外的位置上，若並排放置2個千位分隔符號，則只用於分隔千位。在格式指定的輸出值中實際用作千位分隔符號的字元依據使用的Windows所識別的數值格式。

日期/時間格式指定子

(:)

是時間分隔符號。可能依本地設定而使用其它字元。若以格式指定時間值，時間分隔符號則用於對小時、分鐘、秒的值進行分隔。對於以格式指定輸出而實際使用的字元，會依Windows的設定而定。

(/)

是日期分隔符號。可能依本地設定而使用其它字元。若以格式指定日期值，日期分隔符號則用於對日、月、年的值進行分隔。對於以格式指定輸出而實際使用的字元，會依Windows的設定而定。

c

以ddddd顯示日期，以ttttt顯示時間，依此順序顯示。日期序號上若無分數部分，則只顯示日期；時間訊息上若無整數部分，則只顯示時間訊息。

d
以開頭不附加零的方式顯示日期。(1～31)

dd
以開頭附加零的方式顯示日期。(01～31)

ddd
省略星期顯示。(Sun～Sat)

dddd
不省略星期顯示。(Sunday～Saturday)

ddddd
依照Windows的Short資料顯示設定的格式，顯示所有日期、月、年。Windows的Short資料格式預設為m/d/yy。

dddddd
將日期的序列數值作為日期、月、年，並依Windows的Long資料顯示設定格式進行顯示。Windows的Long資料格式預設為mmmm dd, yyyy。

w
以數值顯示星期。(1：星期日~7：星期六)

ww
以數值顯示1年當中的第幾週。(1～54)

m
以開頭不附加零的數值顯示月份。(1～12)
即便在「h」或「hh」之後使用，也不顯示時間的分鐘。若要顯示時間的分鐘，請使用「n」或「nn」。

mm
以開頭附加零的數值顯示月份。(01～12)
即便在「h」或「hh」之後使用，也不顯示時間的分鐘。若要顯示時間的分鐘，請使用「n」或「nn」。

mmm
省略月份名稱顯示。(Jan～Dec)

mmmm
不省略月份名稱顯示。(January～December)

q
以數值顯示季度。(1～4)

y
以數值顯示1年當中的第幾日。(1～366)

yy
以2位數顯示年號。(00～99)

yyyy
以4位數顯示年號。(100～9999)

h
以開頭不附加零的方式顯示24小時制的時間。(0～23)

hh
以開頭附加零的方式顯示24小時制的時間。(00～23)

n
以開頭不附加零的方式顯示時間的分鐘。(0～59)

nn
以開頭附加零的方式顯示時間的分鐘。(00～59)

s
以開頭不附加零的方式顯示時間的秒鐘。(0～59)

ss
以開頭附加零的方式顯示時間的秒鐘。(00～59)

ttttt
依照Windows設定的時間分隔符號格式，顯示時間(小時、分鐘、秒鐘)。"若選擇「開頭的零」選項，則以開頭附加零的時間制顯示10:00am/pm之前的時間。Windows的預設時間格式為h:nn:ss。

AM/PM
以12小時制顯示時間，並以AM/PM(以大寫標記)顯示上午/下午。

am/pm
以12小時制顯示時間，並以am/pm(以小寫標記)顯示上午/下午。

A/P
以12小時制顯示時間，並以A/P(以大寫標記)顯示上午/下午。

a/p
以12小時制顯示時間，並以a/p(以小寫標記)顯示上午/下午。

AMPM
以12小時制顯示時間。依照Windows內的格式設定，上午時間以AM字串顯示，下午時間以PM字串顯示。AM/PM雖不拘大小寫，但Windows的設定和指定字串務必一致。Windows的預設為AM/PM。

注意

- 混合數值格式指定子和日期/時間格式指定子

  若混合指定數值格式指定子和日期/時間格式指定子，會發生錯誤。

參照

Left$、Right$、Str$

FmtStr$範例

```
Function SendDateCode

    String d$, f$

    f$ = FmtStr$(10, "000.00")
    OpenCom #1
    Print #1, f$
    CloseCom #1
Fend
```

3.10.17 FolderExists函數

用於檢查有無指定的資料夾。

格式

FolderExists (路徑名稱)

參數

路徑名稱

以字串指定要檢查的資料夾路徑名稱。包括驅動器名稱。路徑的詳細內容請參閱ChDisk。

注意

磁碟為PC時，可執行。

傳回值

- 有資料夾時：True
- 沒有資料夾時：False

參照

FileExists、MkDir

FolderExists函數範例

```
If Not FolderExists("c:\TEST") Then
    MkDir "c:\TEST"
EndIf
```

3.10.18 For...Next

用於依指定次數，重複執行For...Next之間的一系列陳述式。

格式

For 變數名稱 = 初始值 To 結束值 [Step增量值] 陳述式 Next [變數名稱]

參數

變數名稱
　　指定用於重複指派資料的變數名稱。通常此變數是整數值，但也可定義為實數變數。
初始值
　　為已指定的變數指定指派給迴圈開頭的數值。
結束值
　　指定表示結束迴圈的值。符合此值時，即結束For...Next迴圈，接下來執行Next命令的下一個陳述式。
增量值
　　指定每當執行For...Next迴圈中的Next陳述式時即進行增量的值。雖然也可將增量值設為負數，但此時必須是初始值＞結束值。未指定增量時，則自動以「1」進行增量。可省略。
陳述式
　　只要是有效的SPEL+陳述式，皆可插入到For...Next迴圈中。

說明

For...Next用於僅以指定的次數重複迴圈中的陳述式。迴圈的開始為For陳述式，最後為Next陳述式。變數則用於計算執行迴圈內陳述式的次數。

初始值的數值為計數器的最初值。若能正確設定結束值變數和增量值，則也可用負值進行設定。

結束值為計數器的最終值。到達此值後迴圈便結束，程式控制會轉移到Next命令的下一個命令。

執行For陳述式的下一個陳述式，直到到達Next命令。計數器變數(變數名稱)會按以增量值參數指定的值進行增量。若未設定增量值，計數器將遞增或遞減「1」。

接下來，計數器變數(變數名稱)會與最終值進行比較。計數器的值少於或等於最終值時，則重新執行For命令的下一個陳述式。計數器變數(變數名稱)大於最終值時，執行則會被分支到For...Next迴圈之外，並繼續執行Next命令的下一個命令。

注意

- 負增量值

  若將負值指定為增量值，計數器變數則會隨著迴圈而減少，因此請設定大於結束值的初始值。

- 可省略Next之後的變數名稱

  可省略Next之後的變數名稱，但若有巢狀等，填寫變數名稱即會比較容易瞭解結構。

- 離開迴圈時的變數值並非是結束值

```
Function forsample
  Integer i
  For i = 0 To 3
  Next
  Print i  '顯示為4
Fend
```

- 如果不使用Exit For 並從嵌套結構退出迴圈

若重複執行Exit For以外的命令(Gosub、Goto、Call命令等)脫離迴圈的程式會導致錯誤2020。如果要在循環中間退出，請使用Exit For命令退出它。

- 盡可能將空無限循環和無限迴圈處理與等待命令結合使用

  盡量不要進行空的For...Next或類似操作，因為它們可能會影響您的系統。控制器檢測無限循環任務，並確定系統受到影響，則可能會出現錯誤 2556(檢測到過多的迴圈)。

  執行需要迴圈的演算或等待I/O時，請在迴圈中執行Wait命令(Wait 0.1)等，以避免佔用CPU。

參照

Do...Loop

For...Next範例

```
Function fornext
    Integer counter
    For counter = 1 to 10
        Go Pctr
    Next counter

    For counter = 10 to 1 Step -1
        Go Pctr
    Next counter
Fend
```

3.10.19 FreeFile函數

用於傳回並預約目前未使用的檔案編號。

格式

FreeFile

傳回值

用於傳回30~63的整數。

參照

AOpen、BOpen、ROpen、UOpen、WOpen、Close

FreeFile函數範例

```
Integer fileNum, i, j

fileNum = FreeFile
WOpen "TEST.DAT" As #fileNum
For i = 0 To 100
    Print #fileNum, i
Next i
Close #fileNum

fileNum = FreeFile
ROpen "TEST.DAT" As #fileNum
For i = 0 to 100
    Input #fileNum, j
    Print "data = ", j
Next i
Close #fileNum
```

3.10.20 Function...Fend

最小單位的可執行程式，1個函式是以Function陳述式開始並以Fend陳述式結束。

格式

Function 函式名稱 [(引數清單)] [As型(函數)] 陳述式 Fend

參數

函數名稱

以64個字元以內的字母，對以Function開始並以Fend命令結束的陳述式群組指定名稱。也可使用下劃線。

引數清單

是呼叫並傳遞給函數的引數變數清單。若有數個變數，則用逗號進行分隔。可省略。

如下所述為引數格式。

[{ByRef | ByVal}] 變數名稱[( )] As型(引數)

ByRef

參照要呼叫的函數變數時，指定ByRef。此時，可將函式內的引數變更反映於叫用側的變數中。可省略。

ByVal

是預設值。不變更以叫用函式參照的變數值時，則指定ByVal。可省略。

變數名稱[()]

為引數的變數名稱，是必須的參數。請依循變數命名規則。若要將陣列變數用作引數，請務必指定ByRef，並在變數之後附加表示陣列的「()」括弧。

As型(引數)

是必須的參數。請宣告引數型態。

As型(函數)

是欲讀取傳回值時附加的參數。請宣告傳回值的型態。

傳回值

是在函式宣告的最後以As指定的資料型態(As型(函數))。

說明

Function陳述式用於表示SPEL+陳述式群組的開始。用Fend陳述式表示1個函式的結束。Function和Fend陳述式之間的所有陳述式，均視為該函式的一部分。

Function和Fend陳述式如同一個容器，包括在兩者之間的所有陳述式，其本身為1個函式。還可將多個函式用於1個程式檔案。

若要使用傳回值，請將值指派與函式名稱同名的變數，然後結束函數。

參照

Call、Fend、Halt、Quit、Return、Xqt

Function...Fend範例

[例1] 以下是1個檔案中包括3個函式的範例。從main叫用task2和task3，並與main同時執行。

```
Function main
  Xqt 2, task2 '同時執行工作2
  Xqt 3, task3 '同時執行工作3
  .
  .
  .
Fend

Function task2
  Do
```

```
    On 1
    On 2
    Off 1
    Off 2
  Loop
Fend

Function task3
  Do
    On 10
    Wait 1
    Off 10
  Loop
Fend
```

[例2] 以下是將週邊裝置的壓力控制序列作為引數賦予並將其傳送到外接裝置時的結果作為傳回值，顯示於畫面時的函式範例。

```
Function main
  Integer iResult
  Real Sequence1(200)
  .
  .
  iResult = PressureControl(ByRef Sequence1())    '引數為陣列
  .
  Print "Result:", iResult
  .
Fend

Function PressureControl(ByRef Array1() As Real) As Integer
  .
  (基於Array1陣列，對週邊裝置進行壓力控制)
  .
  PressureControl = 3     '傳回值
  .
  .
Fend
```

3.11 G

3.11.1 GetRobotInsideBox函數

用於傳回進入到進入檢測區的機器人。

格式
GetRobotInsideBox (區域編號)

參數

區域編號
　　指定要傳回狀態的工作空間編號(1~15的整數)。

傳回值
以位元傳回機器人，該機器人進入到以區域編號指定的進入檢測區。

位元0為機器人1，其後以此類推，位元15為機器人16。

若未設定機器人進入檢測區，則相應位元始終為0。

例如，機器人1和機器人3進入到區域時，位元0和位元2即為On而傳回5。

參照
Box、InsideBox

GetRobotInsideBox函數範例
以下為使用GetRobotInsideBox函數的程式。

用於等待沒有1台機器人進入到進入檢測區的狀態。

```
Function WaitNoBox

  Wait GetRobotInsideBox(1) = 0
```

用於等待只有機器人2進入到進入檢測區的狀態。

```
Function WaitInBoxRobot2

  Wait GetRobotInsideBox(1) = &H2
```

以下程式是在動作命令的平行處理中使用的範例。在執行動作時，若進入到特定進入檢測區，則啟用I/O。是連接於控制器的機器人有1台的情況。

```
Function Main
   Motor On
   Power High
   Speed 30; Accel 30, 30

   Go P1 !D0; Wait GetRobotInsideBox(1) = 1; On 1!

Fend
```

注意

請務必記述D0。

3.11.2 GetRobotInsidePlane函數

用於傳回進入到進入檢測平面的機器人。

格式

GetRobotInsidePlane (平面編號)

參數

平面編號
　　指定傳回狀態的工作平面編號(1~15的整數)。

傳回值

以位元傳回機器人，該機器人進入到以平面編號指定的進入檢測平面。

位元0為機器人1，其後以此類推，位元15為機器人16。

若未設定機器人進入檢測平面，則相應位元始終為0。

例如，機器人1和機器人3進入到平面時，位元0和位元2即為On而傳回5。

參照

InsidePlane、Plane

GetRobotInsidePlane函數範例

以下是使用GetRobotInsidePlane函數的程式。

用於等待沒有1台機器人進入到進入檢測平面的狀態。

```
Function WaitNoPlane

  Wait GetRobotInsidePlane(1) = 0
```

用於等待只有機器人2進入到進入檢測平面的狀態。

```
Function WaitInPlaneRobot2

  Wait GetRobotInsidePlane(1) = &H2
```

以下程式是在動作命令的平行處理中使用的範例。在執行動作時，若進入到特定進入檢測平面，則啟用I/O。是連接於控制器的機器人有1台的情況。

```
Function Main
   Motor On
   Power High
   Speed 30; Accel 30, 30

   Go P1 !D0; Wait GetRobotInsidePlane(1) = 1; On 1!

Fend
```

注意

請務必記述D0。

3.11.3 Global

用於宣告全域變數。可從任何地方存取全域變數。

格式

Global [Preserve] 資料型態 變數名稱 [(陣列變數的最大元素編號)] [, 變數名稱[(陣列變數的最大元素編號)] ,...]

參數

Preserve

若指定Preserve，則維持變數值。若變更專案，則清除此值。可省略。

資料類型

指定Boolean、Byte、Double、Int32、Integer、Long、Real、Short、String、UByte、UInt32、UShort中的任一資料類型。

變數名稱

以32個以內字元指定變數名稱。

陣列變數的最大元素編號

是陣列變數的最大元素編號，最大可宣告三維。使用如下格式。可省略。
(最大元素編號1, [最大元素編號2], [最大元素編號3])
元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。
若為全域變數，所有元素數最大為100,000。但String型時，最大為10,000。
若為備份變數(Global Preserve)，所有元素數最大為4,000。但String型時，最大為400。
在所有元素數不超過最大值的範圍內，指定各最大元素編號。

說明

全域變數是一種只要在相同專案內便可以數個檔案共享的變數。除非以Preserve選項進行宣告，否則每當在Run視窗或操作員視窗啟動函式時便予以清除。

若用Preserve選項進行宣告，即便關閉控制器電源，仍維持該值。

已儲存的全域變數還可用於RC+ API選項。

為了便於在程式中進行區分，請按以下範例，以首碼「g_」開始命名全域變數名稱。

```
Global Long g_PartsCount
```

參照

Boolean、Byte、Double、Int32、Int64、Integer、Long、Real、Short、String、UByte、UInt32、UInt64、UShort

Global範例

以下是不同的2個程式檔案。在第一個檔案中，定義數個全域變數並執行初始化。在接下來的檔案中使用那些全域變數。

```
FILE1 (MAIN.PRG)
Global Integer g_Status
Global Real g_MaxValue

Function Main

    g_Status = 10
    g_MaxValue = 1.1
    .
    .
Fend

Function Test

    Print "status1 =" , g_Status
    Print "MaxValue =" , g_MaxValue
    .
```

```
    .
Fend
```

3.11.4 Go

用於透過PTP動作將手臂從目前位置移到指定位置。

格式

Go目標坐標 [CP] [LJM [選擇姿態旗標]] [Till | Find] [!平行處理!] [SYNC]

參數

目標坐標
　　以點數據指定目標位置。
CP
　　指定路徑運動。可省略。
LJM
　　以LJM函數轉換目標坐標。可省略。
選擇姿態旗標
　　指定賦予LJM函數的姿態旗標選擇參數。可省略。
Till | Find
　　用於記述Till或Find運算式。可省略。

```
Till | Find
Till Sw(運算式) = {On | Off}
Find Sw(運算式) = {On | Off}
```

! 平行處理 !
　　欲在動作中執行I/O等其它命令時，可附加平行處理陳述式。可省略。
SYNC
　　用於預約動作命令。在以SyncRobots開始動作之前，機器人不進行動作。

說明

Go用於透過PTP動作同時運作機器人手臂的所有關節。以下範例用於指定對於Go命令的目標坐標。

- 以特定的點編號進行指定。例: `Go P1`
- 以具體的坐標值指定移動目的地。例: `Go XY(50, 400, 0, 0)`
- 以點編號＋位移值進行指定。例: `Go P1 +X(50)`
- 明確指定點編號＋與其不同的坐標值。例: `Go P1 :X(50)`

由於各關節從目前的點到目標坐標之間進行內插，因此無法預測到其動作路徑的軌跡。請充分注意與週邊裝置之間的干擾情況。

以Speed命令設定Go命令的動作速度。以Accel命令設定加減速度。

若已添加CP參數，可在開始動作減速時重疊下一個動作命令的加速。此時，不在目標坐標上進行定位。

設定LJM參數時，則將目前位置作為參照位置，再將目標位置運作到用LJM函數轉換的位置上。

可將Go LJM(P1, Here,1)簡略記述成Go P1 LJM 1。

此時，並不改變原來的點資料P1。

LJM參數對於垂直6軸型機器人(包含N系列)以及RS系列機器人有效。

若按預設值使用姿態旗標選擇，則可省略。

```
  Go P1 LJM
```

注意

- Go和Move的差異

  Move和Go皆為運作機器人手臂的命令。兩者間的最大差異在於，Go用於PTP動作，而Move則用於以直線軌道移動手臂。認為重視到達目標點時的手臂姿態很重要時，使用Go命令。認為控制動作中的手臂軌道更重要時，使用Move命令。

- Go和Jump的差異

  Jump和Go皆為透過PTP動作移動機器人手臂的命令。然而，Jump具有1個Go沒有的功能。就Jump而言，其作用在於，首先將機器人的抓手抬升到LimZ值位置，水平移動手臂，然後在來到目標坐標上方時，再進行下降移動。此動作的優點在於，可更確實地避開障礙物，而且更重要的是，透過吸附和配置動作，可縮短作業的週期時間。

- 向Go指示適當的速度和加減速度

  以Speed和Accel命令設定Go命令的動作速度和加減速度。最重要的是，Speed和Accel命令與Go命令相同，用於對PTP動作進行設定。以SpeedS和AccelS命令，設定直線及曲線動作的速度與加減速度。

- 使用Till修飾詞的Go用法

  透過使用Till修飾詞，可在到達以Go命令指定的目標坐標之前，在途徑的通過點上設定讓機器人減速並停止的條件。若Till條件未成立，機器人則直接移到目標坐標位置。使用Till的Go用法有以下2種。

  - (1) Go和Till修飾詞

    檢查目前Till條件是否已經成立。若成立，則無須等待完成以Go指定的動作，可透過讓機器人在中途的通過點進行減速和停止完成命令。

  - (2) Go和Till修飾詞、Sw(輸入位元編號)修飾詞、輸入條件

    在此用法中並不使用事先設定的Till既有設定條件，可在與Go命令相同的行上新設定Till條件。設定的條件用於檢查1項輸入，因此需要Sw命令。檢查輸入狀態的ON/OFF，並可依照設定條件停止手臂。此功能幾乎類似於輸入條件成立後便停止動作這樣的「插斷」。若輸入條件在機器人動作時未成立，則手臂會到達指定的目標坐標位置。

- 使用Find修飾詞的Go用法

  使用Find修飾詞，可透過設定條件在用Go命令進行動作時使機器人紀錄1個位置。使用Find的Go用法有以下2種。

  - (1) Go和Find修飾詞

    檢查目前Find條件是否已經成立。若成立，則將目前位置儲存到特別的點、FindPos。

  - (2) Go和Find修飾詞、Sw(輸入位元編號)修飾詞、輸入條件

    在此用法中並不使用事先設定的Find既有設定條件，可在與Go命令相同的行上新設定Find條件。設定條件為檢查1個輸入。因此，需要Sw命令。檢查輸入狀態的ON/OFF，並將目前位置儲存到特別的點、FindPos。

- 使用Go命令時，在停止前必定進行減速

  使用Go命令時，在手臂停於動作的目標坐標之前必定進行減速。

常見錯誤

- 欲將機器人移出活動範圍時

  以直接坐標設定目標坐標時，請務必確認該坐標位置是否在機器人活動範圍之內。若指定機器人活動範圍之外的位置，則會發生錯誤。

參照

! 平行處理 !、Accel、Find、Jump、Move、Pass、P#= 點指定、Pulse、Speed、Sw、Till

Go範例

以下是從P0到P10進行PTP動作的簡易範例。在程式下半段，輸入位元2變為ON之前，手臂朝P2進行PTP動作。若輸入位元2在執行Go的中途變為ON，則手臂減速並停止，不等待到達P2而執行下一個命令。

```
Function sample
  Integer i

  Home
  Go P0
  Go P1
  For i = 1 to 10
    Go P(i)
  Next i
  Go P2 Till Sw(2) = On
  If Sw(2) = On Then
    Print "Input #2 came on during the move and"
    Print "the robot stopped prior to arriving on"
    Print "point P2."
  Else
    Print "The move to P2 completed successfully."
    Print "Input #2 never came on during the move."
  EndIf
Fend
```

這是命令視窗中的操作範例。

```
>Go Here +X(50)                  '從目前位置朝X方向移動50mm
>Go P1                           '朝P1移動
>Go P1 :U(30)                    '目標雖然是P1，但朝U=30的位置移動
>Go P1 /L                        '以左手腕姿態朝P1移動
>Go XY(50, 450, 0, 30)           '朝X=50、Y=450、Z=0、U=30的位置移動
```

[其它範例]

```
Till Sw(1) = Off And Sw(2) = On '將輸入1和2指定為Till條件
Go P1 Till                      '滿足事先定義的Till條件時停止
Go P2 Till Sw(2) = On           '輸入位元2變為ON時停止
Go P3 Till                      '滿足事先定義的Till條件時停止
```

3.11.5 GoSub...Return

GoSub用於將程式控制轉移到副程式。結束執行副程式後，控制便返回到GoSub命令的下一行。

格式

GoSub {標籤}

{標籤::} 陳述式
Return

參數

標籤

以標籤指定轉移目的地。用於將程式執行轉移到有該標籤的行。請指定32個字元以內的標籤名稱。第一個字元不可使用數字。請務必使用字母字元。

說明

GoSub命令用於將程式控制轉移到指定標籤。程式用於執行轉移目的地的陳述式，並執行轉移目的地的行，直到發出Return命令。用GoSub命令執行副程式之後，用Return返回到GoSub命令的下一行。請務必用Return結束副程式。

常見錯誤

- 轉移到不存在的陳述式

  若將不存在GoSub命令的標籤指定為轉移目的地，則發生錯誤3108。

- 沒有GoSub卻使用Return時

  Return命令用於復歸到執行GoSub命令的原程式。若沒有GoSub卻使用Return，則發生錯誤2383。在沒有GoSub的情況下即便使用Return也不知道該復歸到何處，因此沒有任何意義。

參照

GoTo、OnErr、Return

GoSub範例

以下範例表示，使用GoSub命令移到指定標籤，並執行數個I/O命令，然後進行復歸的情形。

```
Function main
    Integer var1, var2

    GoSub checkio  '使用標籤執行GoSub
    On 1
    On 2
    Exit Function

checkio:           '副程式的開始位置
    var1 = In(0)
    var2 = In(1)
    If var1 = 1 And var2 = 1 Then
        On 1
    Else
        Off 1
    EndIf
    Return         '副程式的結束位置
Fend
```

3.11.6 GoTo

GoTo命令用於將程式控制轉移到指定標籤。

格式

GoTo { 標籤 }

參數

標籤

將程式執行轉移到有標籤的行。最多可以32個字元指定標籤，但第一個字符不可使用數字。請務必使用字母。

說明

GoTo命令用於將程式控制轉移到指定標籤。程式用於執行轉移目的地的陳述式行，並執行後續行。

注意

- 過於使用GoTo時

  請注意，若在1個程式中過於使用GoTo命令，則難以理解程式。通常情況下，請盡可能不使用GoTo命令。實際上也有無論如何都要使用GoTo的情形，但是，諸如到處使原始碼轉移這樣的GoTo陳述式用法，會形成錯誤和其它問題的原因。

參照

GoSub、OnErr

GoTo範例

以下是使用GoTo命令，將控制移到行標籤的簡易程式範例。

```
Function main

    If Sw(1) = Off Then
        GoTo mainAbort
    EndIf
    Print "Input 1 was On, continuing cycle"
    .
    .
    Exit Function

mainAbort:
    Print "Input 1 was OFF, cycle aborted!"
Fend
```

3.12 H

3.12.1 Halt

用於暫停正在執行的指定工作。

格式
Halt 工作識別碼

參數

任務識別碼
以整數值或運算式指定任務名稱或任務編號。
工作名稱用於指定Xqt陳述式所用的函式名稱或者在Run視窗或操作員視窗中啟動的函式。
任務編號的指定(整數)

- 一般工作: 1~32
- 背景工作: 65~80
- Trap工作: 257~267

說明
Halt用於暫停以工作名稱或工作編號指定的執行中工作。

停止後若要重新開始，則使用Resume。若要完全結束執行工作，則使用Quit。若要顯示工作狀態，則按一下Epson RC+工具列上的工作管理員圖示以啟動工作管理員。

指定工作即便是NoPause工作、NoEmgAbort工作(執行Xqt時，指定NoPause或NoEmgAbort開始的特殊工作)、Trap工作、背景工作，Halt也用於暫停工作。但是，若要暫停這些工作，需充分進行研討。建議在一般情況下不要對特殊工作執行Halt。

參照
Quit、Resume、Xqt

Halt範例
以下是用Xqt啟動以「flicker」為名的函式，然後用Halt暫停，並用Resume重新開始的範例。

```
Function main
  Xqt flicker         '執行flicker工作

  Do
    Wait 3            '執行flicke工作3秒鐘
    Halt flicker

    Wait 3            '將flicker工作暫停3秒鐘。
    Resume flicker

  Loop
Fend

Function flicker
  Do
    On 1
    Wait 0.2
    Off 1
    Wait 0.2
  Loop
Fend
```

3.12.2 Hand

用於設定點的手臂姿態。

格式

(1) Hand 點指定 [, Lefty | Righty]

(2) Hand

參數

點指定
　　用於指定P編號、P(運算式)、點標籤之一。
若省略Lefty | Righty
　　指定夾具(腕部)姿態。

結果

若2個參數皆省略，則顯示機器人目前位置的手臂姿態。

若省略Lefty | Righty參數，則顯示指定點的手臂姿態。

注意

Hand命令不是控制末端夾具的命令。

- Hand (此命令)：指定機械臂的左手臂或右手臂姿態。|
- Hand_On, Hand_Off, Hand_On函數, Hand_Off函數, Hand_TW函數, Hand_Def函數, Hand_Type函數, Hand_Label$函數, Hand_Number函數：用於控制安裝在機械臂前端的夾具(末端夾具)。

請注意不要混淆。

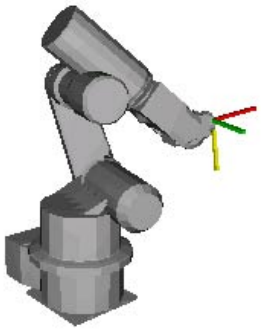
有關末端夾具命令的更多資訊，請參閱以下手冊。
「Hand功能」

參照

Elbow、Hand函數、J4Flag、J6Flag、Wrist、J1Flag、J2Flag

Hand範例

```
Hand P0, Lefty
Hand pick, Righty
Hand P(myPoint), myHand

P1 = -364.474, 120.952, 469.384, 72.414, 1.125, -79.991
```



```
Hand P1, Righty
Go P1
```

```
Hand P1, Lefty
Go P1
```

3.12.3 Hand函數

用於傳回點的手臂姿態。

格式

Hand [(點指定)]

參數

點指定

以點運算式進行指定。可省略。若省略，則傳回機器人目前位置的手臂姿態。

傳回值

- 1: Righty (/R)
- 2: Lefty (/L)

注意

Hand命令不是控制末端夾具的命令。

- Hand函數 (此命令)：指定機械臂的左手臂或右手臂姿態。
- Hand_On, Hand_Off, Hand_On函數, Hand_Off函數, Hand_TW函數, Hand_Def函數, Hand_Type函數, Hand_Label$函數, Hand_Number函數：用於控制安裝在機械臂前端的夾具(末端夾具)。

請注意不要混淆。

有關末端夾具命令的更多資訊，請參閱以下手冊。
「Hand功能」

參照

Elbow、Wrist、J4Flag、J6Flag、J1Flag、J2Flag

Hand函數範例

```
Print Hand(pick)
Print Hand(P1)
Print Hand
Print Hand(P1 + P2)
```

3.12.4 HealthCalcPeriod

設定和顯示對零件消耗管理訊息中的「剩餘月數」進行運算的期間。

格式

(1) HealthCalcPeriod 期間設定值

(2) HealthCalcPeriod

參數

期間設定值

以整數值(1~7)指定要運算的期間。(單位：日) 預設值為「7」。

結果

若省略參數，則顯示目前HealthCalcPeriod設定值。

說明

依照過去的運作狀況定期運算零件消耗管理訊息的剩餘月數。HealthCalcPeriod命令用於設定和顯示運算所用動作期間。

若延長期間，則運算抑制因期間內變動而造成偏差影響的「剩餘月數」。變更動作或速度時，需要花點時間才能正確顯示「剩餘月數」。

HealthCalcPeriod的設定值適用於受執行控制器控制的所有機器人、關節和零件。

注意

- 支援的控制器型號

  不支援T/VT系列。T/VT系列始終為1(天)

- 設定期間

  用HealthCalcPeriod命令設定的期間為啟動控制器的期間。

  請注意不同於實際時間。

- 清除「剩餘月數」以及「消耗率」時的運算

  從Epson RC+的「零件消耗管理訊息」、HealthCtrlReset或HealthRBReset中清除「剩餘月數」以及「消耗率」之後，無論HealthCalcPeriod的設定值如何，在到達最初的設定期間之前，每天均運算剩餘月數。在此期間，剩餘月數的偏差會增大。

參照

HealthCalcPeriod函數、HealthCtrlInfo、HealthRBInfo、HealthCtrlReset、HealthRBReset

HealthCalcPeriod範例

```
> HealthCalcPeriod 3
> HealthCalcPeriod
 3
```

3.12.5 HealthCalcPeriod函數

用於傳回對目前設定的零件消耗管理訊息中的「剩餘月數」進行運算的期間。

格式

HealthCalcPeriod

傳回值

用於以整數值傳回運算期間。(單位：日)

注意

- 支援的控制器型號

  不支援T/VT系列。

參照

HealthCalcPeriod

HealthCalcPeriod函數範例

是顯示運算期間的範例。

```
Print "period is", HealthCalcPeriod
```

3.12.6 HealthCtrlAlarmOn函數

本函數用於傳回與控制器相關的指定類別零件之零件消耗警報狀態。

格式

HealthCtrlAlarmOn (零件類別)

參數

部件類型

以整數值(1)或以下所示之常數指定要傳回警報狀態的部件。

常數	值	模式
HEALTH_CONTROLLER_TYPE_BATTERY	1	指定電池。

傳回值

若指定零件發出零件消耗警報，則傳回True；若未發出，則傳回False。

用HealthRateCtrlInfo取得的零件消耗率超過100%時，則發出零件消耗警報。

參照

HealthCtrlInfo、HealthRateCtrlInfo

HealthCtrlAlarmOn函數範例

這是判斷是否發出控制器電池零件消耗警報的範例。

```
Function PrintAlarm
  If HealthCtrlAlarmOn(HEALTH_CONTROLLER_TYPE_BATTERY) = True Then
    Print "Controller Battery NG"
  Else
    Print "Controller Battery OK"
  EndIf
Fend
```

3.12.7 HealthCtrlInfo

用於顯示與控制器相關的指定類別零件之建議更換時期到期之前的剩餘月數。

格式

HealthCtrlInfo 零件類別

參數

部件類型

以整數值(1)或以下所示之常數指定要傳回建議更換時期到期之前的剩餘月數之部件。

常數	值	模式
HEALTH_CONTROLLER_TYPE_BATTERY	1	指定電池。

說明

用於顯示與控制器相關的指定類別零件之建議更換時期到期之前的剩餘月數。

根據過去使用狀況的零件消耗率、基於每次在HealthCalcPeriod所設定期間運作控制器時取得的消費率變化量，來運算建議更換時期到期之前的月數。

注意

基於每次在HealthCalcPeriod所設定期間運作控制器時取得的消費率變化量，來運算建議更換時期到期之前的剩餘月數，因此遇到以下情況時，則無法正常運算。

- 在運作時間未滿HealthCalcPeriod所設定期間的情況下執行本命令。
- 在結束長期的機器人運作停止期間後執行本命令。
- 在更換零件並重設零件消耗後執行本命令。
- 變更控制器的日期與時間設定。

在這種情況下，若按以HealthCalcPeriod所設定期間的2倍以上時間運作控制器後執行本命令，則顯示正確的值。

參照

HealthCtrlAlarmOn、HealthRateCtrlInfo

HealthCtrlInfo範例

這是顯示控制器電池的建議更換時期到期之前的剩餘月數的範例。

```
> HealthCtrlInfo HEALTH_CONTROLLER_TYPE_BATTERY
BATTERY            240.000
>
```

3.12.8 HealthCtrlInfo函數

本函數用於傳回與控制器相關的指定類別零件之建議更換時期到期之前的剩餘月數。

格式

HealthCtrlInfo (零件類別)

參數

部件類型

以整數值(1)或以下所示之常數指定要傳回建議更換時期到期之前的剩餘月數之部件。

常數	值	模式
HEALTH_CONTROLLER_TYPE_BATTERY	1	指定電池。

傳回值

用於以實數傳回建議更換時期到期之前的剩餘月數(單位：月)。

說明

根據過去使用狀況的零件消耗率、基於每次在HealthCalcPeriod所設定期間運作控制器時取得的消費率變化量，來運算建議更換時期到期之前的月數。

注意

基於每次在HealthCalcPeriod所設定期間運作控制器時取得的消費率變化量，來運算建議更換時期到期之前的剩餘月數，因此遇到以下情況時，則無法正常運算。

- 在運作時間未滿HealthCalcPeriod所設定期間的情況下執行本命令。
- 在結束長期的機器人運作停止期間後執行本命令。
- 在更換零件並重設零件消耗後執行本命令。
- 變更控制器的日期與時間設定。

在這種情況下，若按以HealthCalcPeriod所設定期間的2倍以上時間運作控制器後執行本命令，則顯示正確的值。

參照

HealthCtrlAlarmOn、HealthRateCtrlInfo

HealthCtrlInfo函數範例

這是建議更換時期到期之前的剩餘月數不足1個月時輸出警報的範例。

```
Function AlarmCheck
  Real month

  month = HealthCtrlInfo(HEALTH_CONTROLLER_TYPE_BATTERY)
  If month < 1 Then
    Print "Alarm ON"
  EndIf
Fend
```

3.12.9 HealthCtrlRateOffset

對指定零件類別的消耗率設定位移。

格式

HealthCtrlRateOffset零件類別, 位移值

參數

部件類型

以整數值(1)或以下所示之常數指定控制器相關部件。

常數	值	模式
HEALTH_CONTROLLER_TYPE_BATTERY	1	指定電池。

偏移值

對消耗率指定要偏移的整數值。(單位：%)

說明

對指定零件類別設定消耗率的位移。

參照

HealthRBAlarmOn、HealthRateRBInfo、HealthRBInfo

HealthCtrlRateOffset範例

這是在控制器的電池消耗率上加上10%的範例。

```
> HealthCtrlRateOffset HEALTH_CONTROLLER_TYPE_BATTERY,10
>
```

3.12.10 HealthCtrlReset

用於清除指定零件類別的建議更換時期到期之前的剩餘月數及消耗率。

格式

HealthCtrlReset 零件類別

參數

部件類型

以整數值(1)或以下所示之常數指定控制器相關部件。

常數	值	模式
HEALTH_CONTROLLER_TYPE_BATTERY	1	指定電池。

說明

針對指定零件類別，清除建議更換時期到期之前的剩餘月數及消耗率。同時一併解除警告。

參照

HealthCtrlAlarmOn、HealthRateCtrlInfo、HealthCtrlInfo

HealthCtrlReset範例

```
> HealthCtrlReset HEALTH_CONTROLLER_TYPE_BATTERY
>
```

3.12.11 HealthCtrlWarningEnable

啟用或停用與控制器相關的指定類別零件之零件消耗警報通知。

格式

HealthCtrlWarningEnable 零件類別[, On/Off]

參數

部件類型

以整數值或以下所示之常數指定控制器的部件。

常數	值	模式
HEALTH_CONTROLLER_TYPE_BATTERY	1	指定電池。

On/Off

- On：啟用部件消耗警報通知。- Off：停用部件消耗警報通知。

結果

若省略On/Off參數，則顯示目前On/Off設定值。

說明

設定發出指定類別零件之零件消耗警報時是否進行零件消耗警報通知。

注意

停用指定零件的零件消耗警報通知後，即便已過建議更換時期，也不通知零件消耗警報。執行時，請充分注意用本命令進行的設定。

參照

HealthCtrlAlarmOn

HealthCtrlWarningEnable範例

這是停用控制器的電池零件消耗警報通知之範例。

```
> HealthCtrlWarningEnable HEALTH_CONTROLLER_TYPE_BATTERY, Off
```

這是顯示控制器的電池零件消耗警報通知設定的範例。

```
> HealthCtrlWarningEnable HEALTH_CONTROLLER_TYPE_BATTERY
BATTERY  Off
>
```

3.12.12 HealthCtrlWarningEnable函數

本函數用於傳回與控制器相關的指定類別零件之零件消耗警報通知設定狀態。

格式

HealthCtrlWarningEnable (零件類別)

參數

部件類型

以整數值或以下所示之常數指定控制器的部件。

常數	值	模式
HEALTH_CONTROLLER_TYPE_BATTERY	1	指定電池。

傳回值

用於以整數值傳回零件消耗警報的設定值。

- 1: On
- 0: Off

參照

HealthCtrlAlarmOn

HealthCtrlWarningEnable函數範例

這是顯示控制器的電池零件消耗警報通知設定狀態的範例。

```
Print HealthCtrlWarningEnable(HEALTH_CONTROLLER_TYPE_BATTERY )
```

3.12.13 HealthRateCtrlInfo函數

本函數用於傳回與控制器相關的指定類別零件的消耗率。

格式

HealthRateCtrlInfo (零件類別)

參數

部件類型

以整數值(1)或以下所示之常數指定要傳回建議更換時期到期之前的剩餘月數之部件。

常數	值	模式
HEALTH_CONTROLLER_TYPE_BATTERY	1	指定電池。

傳回值

用於以實數傳回建議更換時期作為100%時的目前零件消耗率(單位：%)。

說明

基於實際運作狀況的資料對零件消耗率進行運算。

注意

建議更換時期是基於統計建議更換零件的時期。在零件消耗率達到100%之前，也有可能需要更換。此外，即便達到100%，也並不代表已經無法再使用。但是，超過100%後會增加故障的可能性，因此建議盡早更換。

參照

HealthCtrlAlarmOn、HealthCtrlInfo,

HealthRateCtrlInfo函數範例

這是控制器電池的零件消耗率超過90%時發出警報的範例。

```
Function AlarmCheck
  Real HealthRate

  HealthRate = HealthRateCtrlInfo (HEALTH_CONTROLLER_TYPE_BATTERY)
  If HealthRate > 90 Then
    Print "Alarm ON"
  EndIf
Fend
```

3.12.14 HealthRateRBInfo函數

本函數用於傳回與機器人相關的指定類別零件之消耗率。

格式

HealthRateRBInfo (機器人編號, 零件類別, 關節編號)

參數

機器人編號

以整數值(1~16)指定要傳回部件消耗率的機器人編號。

部件類型

以整數值(1~6)或以下所示之常數指定要傳回部件消耗率的部件。

常數	值	模式
HEALTH_ROBOT_TYPE_BATTERY	1	指定電池。
HEALTH_ROBOT_TYPE_BELT	2	指定皮帶。
HEALTH_ROBOT_TYPE_GREASE	3	指定潤滑脂。
HEALTH_ROBOT_TYPE_MOTOR	4	指定馬達。
HEALTH_ROBOT_TYPE_GEAR	5	指定減速機。
HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE	6	指定滾珠螺桿花鍵。

關節編號

以整數值(1~9)指定要傳回部件消耗率的關節。本命令不可用於附加軸。

傳回值

用於以實數傳回建議更換時期為100%時的目前零件消耗率(單位：%)。

機器人的關節未使用指定零件時，則傳回-1。

說明

基於實際運作狀況的資料對零件消耗率進行運算。

注意

建議更換時期是基於統計建議更換零件的時期。在零件消耗率達到100%之前，也有可能需要更換。此外，即便達到100%，也並不代表已經無法再使用。但是，超過100%後會增加故障的可能性，因此建議盡早更換。

參照

HealthRBAlarmOn、HealthRBInfo

HealthRateRBInfo函數範例

這是機器人1的第3關節減速機的零件消耗率超過90%時發出警報之範例。

```
Function AlarmCheck
Real HealthRate

HealthRate = HealthRateRBInfo(1, HEALTH_ROBOT_TYPE_GEAR,3)
If HealthRate > 90 Then
Print "Alarm ON"
EndIf
Fend
```

3.12.15 HealthRBAlarmOn函數

本函數用於傳回與機器人相關的指定類別零件之零件消耗警報狀態。

格式

HealthRBAlarmOn (機器人編號, 零件類別, 關節編號)

參數

機器人編號

以整數值(1~16)指定要傳回警報狀態的機器人編號。

部件類型

以整數值(1-6)或以下所示之常數指定要傳回警報狀態的部件。

常數	值	模式
HEALTH_ROBOT_TYPE_BATTERY	1	指定電池。
HEALTH_ROBOT_TYPE_BELT	2	指定皮帶。
HEALTH_ROBOT_TYPE_GREASE	3	指定潤滑脂。
HEALTH_ROBOT_TYPE_MOTOR	4	指定馬達。
HEALTH_ROBOT_TYPE_GEAR	5	指定減速機。
HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE	6	指定滾珠螺桿花鍵。

關節編號

以整數值(1~9)指定要傳回警報狀態的關節。若有按零件類別選擇電池，電池通用於所有關節，無論指定哪一個關節，皆傳回相同值。本命令不可用於附加軸。

傳回值

若指定零件發出零件消耗警報，則傳回True；若未發出，則傳回False。

用HealthRateRBInfo取得的零件消耗率超過100%時，會發出零件消耗警報。

機器人的關節未使用指定零件時，則傳回-1。

參照

HealthRBInfo、HealthRateRBInfo

HealthRBAlarmOn函數範例

這是判斷是否發出機器人1第3關節的潤滑脂零件消耗警報的範例。

```
Function PrintAlarm4
If HealthRBAlarmOn(1, HEALTH_ROBOT_TYPE_GREASE, 3) = True Then
Print "Robot1 Joint3 Grease NG"
Else
Print "Robot1 Joint3 Grease OK"
EndIf
Fend
```

3.12.16 HealthRBAnalysis函數

針對某週期的機器人動作，本函數用於傳回指定零件類別的可使用月數。

格式

HealthRBAnalysis (機器人編號, 零件類別, 關節編號)

參數

機器人編號

以整數值(1~16)指定機器人編號。

部件類型

以整數值(2~6)或以下所示之常數指定機器人相關部件。

常數	值	模式
HEALTH_ROBOT_TYPE_BELT	2	指定皮帶。
HEALTH_ROBOT_TYPE_GREASE	3	指定潤滑脂。
HEALTH_ROBOT_TYPE_MOTOR	4	指定馬達。
HEALTH_ROBOT_TYPE_GEAR	5	指定減速機。
HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE	6	指定滾珠螺桿花鍵。

關節編號

以整數值(1~6)指定關節。本命令不可用於附加軸。

傳回值

用於以實數傳回可使用月數。

若在指定關節不存在指定的零件類別，則傳回-1。

說明

針對某週期的機器人動作，模擬指定零件類別的可使用月數。對零件在新品狀態下連續運作24小時後可使用的月數進行計算。不加入過去的使用情況。

注意

- Auto模式下無法發揮功能。
- 空運行(包括虛擬控制器)時也無法發揮功能。

參照

HealthRBStart、HealthRBStop

HealthRBAnalysis函數範例

```
Function RobotPartAnalysis
  Real month

  Robot 1

  HealthRBStart 1
  Motor On
  Go P0
  Go P1
  Motor Off
  HealthRBStop 1
```

```
  month = HealthRBAnalysis(1, HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE, 3)
  Print "Ball Screw Spline analysis =", Str$(month)
Fend
```

3.12.17 HealthRBDistance

顯示指定關節的馬達驅動量(旋轉量)。

格式

HealthRBDistance [機器人編號] [,關節編號]

參數

機器人編號

以整數值(1~16)指定機器人編號。可省略。若有省略，則使用目前機器人編號。

關節編號

以整數值(1~6)指定關節。未指定關節編號時，則傳回所有關節的值。本命令不可用於附加軸。

說明

針對從HealthRBStart到HealthRBStop為止的機器人動作，計算指定關節的馬達驅動量(旋轉量)並進行顯示。不加入過去的使用量。

注意

- Auto模式下無法發揮功能。
- 空運行(包括虛擬控制器)時也無法發揮功能。

參照

HealthRBStart、HealthRBStop

HealthRBDistance範例

這是顯示SCARA機器人第1關節的驅動量之範例。

```
> HealthRBDistance 1, 1
 1.000
>
```

3.12.18 HealthRBDistance函數

用於傳回指定關節的馬達驅動量(旋轉量)之函數。

格式

HealthRBDistance ([機器人編號,] 關節編號)

參數

機器人編號

以整數值(1~16)指定機器人編號。可省略。若有省略，則使用目前機器人編號。

關節編號

以整數值(1~6)指定關節。本命令不可用於附加軸。

傳回值

用於以實數傳回驅動量。

說明

針對從HealthRBStart到HealthRBStop為止的機器人動作，傳回指定關節的馬達驅動量(旋轉數)。不加入過去的使用量。

注意

- Auto模式下無法發揮功能。
- 空運行(包括虛擬控制器)時也無法發揮功能。

參照

HealthRBStart、HealthRBStop

HealthRBDistance函數範例

```
Function RobotPartAnalysis
  Real healthDistance

  Robot 1

  HealthRBStart 1
  Motor On
  Go P0
  Go P1
  Motor Off
  HealthRBStop 1

  healthDistance = HealthRBDistance(1,1)
  Print "Distance =", Str$(healthDistance)
Fend
```

3.12.19 HealthRBInfo

用於顯示與機器人相關的指定類別零件之建議更換時期到期之前的剩餘月數。

格式

HealthRBInfo 機器人編號, 零件類別[, 關節編號]

參數

機器人編號

以整數值(1~16)指定要傳回建議更換時期到期之前的剩餘月數之機器人編號。

部件類型

以整數值(0~6)或以下所示之常數指定要傳回建議更換時期到期之前的剩餘月數之部件。

常數	值	模式
HEALTH_ROBOT_TYPE_ALL	0	指定所有零件類別。
HEALTH_ROBOT_TYPE_BATTERY	1	指定電池。
HEALTH_ROBOT_TYPE_BELT	2	指定皮帶。
HEALTH_ROBOT_TYPE_GREASE	3	指定潤滑脂。
HEALTH_ROBOT_TYPE_MOTOR	4	指定馬達。
HEALTH_ROBOT_TYPE_GEAR	5	指定減速機。
HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE	6	指定滾珠螺桿花鍵。

關節編號

以整數值(1~9)指定要傳回建議更換時期到期之前的剩餘月數之關節。若有按零件類別選擇電池，電池通用於所有關節，無論指定哪一個關節，皆傳回相同值。未指定關節編號時，則傳回所有關節的值。本命令不可用於附加軸。

說明

用於顯示與機器人相關的指定類別零件之建議更換時期到期之前的剩餘月數。

根據過去使用狀況的零件消耗率、基於每次在HealthCalcPeriod所設定期間運作控制器時取得的消費率變化量，來運算建議更換時期到期之前的月數。

機器人的關節未使用指定零件時，則顯示-1。

注意

基於每次在HealthCalcPeriod所設定期間運作控制器時取得的消費率變化量，來運算建議更換時期到期之前的剩餘月數，因此遇到以下情況時，則無法正常運算。

- 在運作時間未滿HealthCalcPeriod所設定期間的情況下執行本命令。
- 在結束長期的機器人運作停止期間後執行本命令。
- 在更換零件並重設零件消耗後執行本命令。
- 變更控制器的日期與時間設定。

在這種情況下，若按以HealthCalcPeriod所設定期間的2倍以上時間運作控制器後執行本命令，則顯示正確的值。

參照

HealthRBAlarmOn、HealthRateRBInfo

HealthRBInfo範例

這是顯示機器人1所有關節的所有零件之建議更換時期到期之前的剩餘月數的範例。

```
> HealthRBInfo 1, HEALTH_ROBOT_TYPE_ALL
BATTERY            240.000
BELT                -1.000,   -1.000,   38.689,   95.226
GREASE              -1.000,   -1.000,   21.130,   -1.000
MOTOR              240.000,  240.000,  240.000,  240.000
GEAR               240.000,  224.357,   -1.000,   -1.000
BALL_SCREW_SPLINE   -1.000,   -1.000,  240.000,   -1.000
>
```

這是顯示機器人1所有關節減速機建議更換時期到期之前的剩餘月數之範例。

```
> HealthRBInfo 1, HEALTH_ROBOT_TYPE_GEAR
GEAR               240.000,  224.357,   -1.000,   -1.000
>
```

這是顯示機器人1第2關節馬達建議更換時期到期之前的剩餘月數之範例。

```
> HealthRBInfo 1, HEALTH_ROBOT_TYPE_MOTOR, 2
MOTOR              224.357
>
```

3.12.20 HealthRBInfo函數

本函數用於傳回與機器人相關的指定類別零件之建議更換時期到期之前的剩餘月數。

格式

HealthRBInfo (機器人編號, 零件類別, 關節編號)

參數

機器人編號

以整數值(1~16)指定要傳回建議更換時期到期之前的剩餘月數之機器人編號。

部件類型

以整數值(1~6)或以下所示之常數指定要傳回建議更換時期到期之前的剩餘月數之部件。

常數	值	模式
HEALTH_ROBOT_TYPE_BATTERY	1	指定電池。
HEALTH_ROBOT_TYPE_BELT	2	指定皮帶。
HEALTH_ROBOT_TYPE_GREASE	3	指定潤滑脂。
HEALTH_ROBOT_TYPE_MOTOR	4	指定馬達。
HEALTH_ROBOT_TYPE_GEAR	5	指定減速機。
HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE	6	指定滾珠螺桿花鍵。

關節編號

以整數值(1~9)指定要傳回建議更換時期到期之前的剩餘月數之關節。若有按零件類別選擇電池，電池通用於所有關節，無論指定哪一個關節，皆傳回相同值。本命令不可用於附加軸。

傳回值

用於以實數傳回建議更換時期到期之前的剩餘月數(單位：月)。

機器人的關節未使用指定零件時，則傳回-1。

說明

根據過去使用狀況的零件消耗率、基於每次在HealthCalcPeriod所設定期間運作控制器時取得的消費率變化量，來運算建議更換時期到期之前的月數。

注意

基於每次在HealthCalcPeriod所設定期間運作控制器時取得的消費率變化量，來運算建議更換時期到期之前的剩餘月數，因此遇到以下情況時，則無法正常運算。

- 在運作時間未滿HealthCalcPeriod所設定期間的情況下執行本命令。
- 在結束長期的機器人運作停止期間後執行本命令。
- 在更換零件並重設零件消耗後執行本命令。
- 變更控制器的日期與時間設定。

在這種情況下，若按以HealthCalcPeriod所設定期間的2倍以上時間運作控制器後執行本命令，則顯示正確的值。

參照

HealthRBAlarmOn、HealthRateRBInfo

HealthRBInfo函數範例

這是機器人1第3關節滾珠螺桿花鍵的建議更換時期到期之前的剩餘月數不足1個月時輸出警報的範例。

```
Function AlarmCheck
  Real month

  month = HealthRBInfo(1, HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE, 3)
  If month < 1 Then
    Print "Alarm ON"
  EndIf
Fend
```

3.12.21 HealthRBRateOffset

對指定零件類別的消耗率設定位移。

格式

HealthRBRateOffset 機器人編號, 零件類別, 關節編號, 位移值

參數

機器人編號

以整數值(1~16)指定機器人編號。

部件類型

以整數值(1~6)或以下所示之常數指定機器人相關部件。

常數	值	模式
HEALTH_ROBOT_TYPE_BATTERY	1	指定電池。
HEALTH_ROBOT_TYPE_BELT	2	指定皮帶。
HEALTH_ROBOT_TYPE_GREASE	3	指定潤滑脂。
HEALTH_ROBOT_TYPE_MOTOR	4	指定馬達。
HEALTH_ROBOT_TYPE_GEAR	5	指定減速機。
HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE	6	指定滾珠螺桿花鍵。

關節編號

以整數值(1~6)指定關節。若有按零件類別選擇電池，電池通用於所有關節，無論指定哪一個關節，皆設定位移。本命令不可用於附加軸。

偏移值

對消耗率指定要偏移的整數值。(單位：%)

說明

對指定零件類別和關節，設定消耗率的位移。

參照

HealthRBAlarmOn、HealthRateRBInfo、HealthRBInfo

HealthRBRateOffset範例

這是在機器人1第1關節減速機的消耗率上加上10%的範例。

```
> HealthRBRateOffset 1,HEALTH_ROBOT_TYPE_GEAR,1,10
>
```

3.12.22 HealthRBReset

用於清除指定零件類別的建議更換時期到期之前的剩餘月數及消耗率。

格式

HealthRBReset 機器人編號, 零件類別, 關節編號

參數

機器人編號

以整數值(1~16)指定機器人編號。

部件類型

以整數值(1~6)或以下所示之常數指定機器人相關部件。

常數	值	模式
HEALTH_ROBOT_TYPE_BATTERY	1	指定電池。
HEALTH_ROBOT_TYPE_BELT	2	指定皮帶。
HEALTH_ROBOT_TYPE_GREASE	3	指定潤滑脂。
HEALTH_ROBOT_TYPE_MOTOR	4	指定馬達。
HEALTH_ROBOT_TYPE_GEAR	5	指定減速機。
HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE	6	指定滾珠螺桿花鍵。

關節編號

以整數值(1~6)指定要傳回建議更換時期到期之前的剩餘月數之關節。若有按零件類別選擇電池，電池通用於所有關節，無論指定哪一個關節，皆一併清除。本命令不可用於附加軸。

說明

針對指定零件類別和關節，清除建議更換時期到期之前的剩餘月數及消耗率。

同時一併解除警告。

參照

HealthRBAlarmOn、HealthRateRBInfo、HealthRBInfo

HealthRBReset範例

```
> HealthRBReset 1,HEALTH_ROBOT_TYPE_GEAR,1
>
```

3.12.23 HealthRBSpeed

顯示指定關節速度的平均值。

格式

HealthRBSpeed [機器人編號] [, 關節編號]

參數

機器人編號

以整數值(1~16)指定機器人編號。可省略。若有省略，則使用目前機器人編號。

關節編號

以整數值(1~6)指定關節。未指定關節編號時，則傳回所有關節的值。本命令不可用於附加軸。

說明

針對從HealthRBStart到HealthRBStop為止的機器人動作，顯示指定關節速度的平均絕對值。以0~1的實數值表示結果。最大平均速度值為「1」。

平均值為0.001以下的值時，則變為「0」。

注意

- Auto模式下無法發揮功能。
- 空運行(包括虛擬控制器)時也無法發揮功能。

參照

HealthRBStart、HealthRBStop、AveSpeed

HealthRBSpeed範例

這是顯示SCARA機器人第1關節速度的範例。

```
> HealthRBSpeed 1, 1
 0.100
>
```

3.12.24 HealthRBSpeed函數

用於傳回指定關節速度的平均絕對值。

格式

HealthRBSpeed ([機器人編號,] 關節編號)

參數

機器人編號

以整數值(1~16)指定機器人編號。可省略。若有省略，則使用目前機器人編號。

關節編號

以整數值(1~6)指定關節。本命令不可用於附加軸。

傳回值

以0~1的實數值傳回。

說明

針對從HealthRBStart到HealthRBStop為止的機器人動作，傳回指定關節速度的平均絕對值。以0~1的實數值表示結果。最大平均速度為1。

注意

- Auto模式下無法發揮功能。
- 空運行(包括虛擬控制器)時也無法發揮功能。

參照

HealthRBStart、HealthRBStop、AveSpeed

HealthRBSpeed函數範例

```
Function RobotPartAnalysis
  Real healthSpeed

  Robot 1

  HealthRBStart 1
  Motor On
  Go P0
  Go P1
  Motor Off
  HealthRBStop 1

  healthSpeed = HealthRBSpeed(1,1)
  Print "AveSpeed =", Str$(healthSpeed)
Fend
```

3.12.25 HealthRBStart

針對某週期的機器人動作，開始對零件的可使用月數和元素進行運算。

格式

HealthRBStart 機器人編號

參數

機器人編號

以整數值(1~16)指定機器人編號。

說明

針對指定的機器人編號，開始對零件的可使用月數和元素(扭矩、速度、驅動量)進行運算。

若在開始運算的狀態下重新執行，上一次的運算結果則會被初始化。

注意

- Auto模式下無法發揮功能。
- 空運行(包括虛擬控制器)時也無法發揮功能。

參照

HealthRBAnalysis、HealthRBStop、HealthRBTRQ、HealthRBSpeed、HealthRBDistance

HealthRBStart範例

```
Function RobotPartAnalysis
  Real month

  Robot 1

  HealthRBStart 1
  Motor On
  Go P0
  Go P1
  Motor Off
  HealthRBStop 1

  month = HealthRBAnalysis(1, HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE, 3)
  Print "Ball Screw Spline analysis =", Str$(month)
Fend
```

3.12.26 HealthRBAnalysis

針對某週期的機器人動作，模擬指定零件類別的可使用月數並進行顯示。

格式

HealthRBAnalysis 機器人編號, 零件類別[, 關節編號]

參數

機器人編號

以整數值(1~16)指定機器人編號。

部件類型

以整數值或以下所示之常數指定機器人相關部件。

常數	值	模式
HEALTH_ROBOT_TYPE_ALL	0	指定所有零件類別。
HEALTH_ROBOT_TYPE_BELT	2	指定皮帶。
HEALTH_ROBOT_TYPE_GREASE	3	指定潤滑脂。
HEALTH_ROBOT_TYPE_MOTOR	4	指定馬達。
HEALTH_ROBOT_TYPE_GEAR	5	指定減速機。
HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE	6	指定滾珠螺桿花鍵。

關節編號

以整數值(1~6)指定關節。未指定關節編號時，則傳回所有關節的值。本命令不可用於附加軸。

說明

針對某週期的機器人動作，模擬指定零件類別的可使用月數。對零件在新品狀態下連續運作24小時後可使用的月數進行計算並顯示。不加入過去的使用情況。

若在指定關節不存在指定的零件類別，則傳回-1。

注意

- Auto模式下無法發揮功能。
- 空運行(包括虛擬控制器)時也無法發揮功能。

參照

HealthRBStart、HealthRBStop

HealthRBAnalysis範例

這是顯示SCARA機器人的所有關節之所有零件可使用月數的範例。

```
> HealthRBAnalysis 1, HEALTH_ROBOT_TYPE_ALL
BELT                -1.000,   -1.000,   38.689,   95.226
GREASE              -1.000,   -1.000,   21.130,   -1.000
MOTOR              240.000,  240.000,  240.000,  240.000
GEAR               240.000,  224.357,   -1.000,   -1.000
BALL_SCREW_SPLINE   -1.000,   -1.000,  240.000,   -1.000
>
```

這是顯示SCARA機器人的所有關節減速機可使用月數之範例。

```
> HealthRBAnalysis 1, HEALTH_ROBOT_TYPE_GEAR
GEAR                240.000,  224.357,   -1.000,   -1.000
>
```

這是顯示6軸機器人第2關節馬達的可使用月數之範例。

```
> HealthRBAnalysis 1, HEALTH_ROBOT_TYPE_MOTOR, 2
MOTOR               224.357
>
```

3.12.27 HealthRBStop

針對某週期的機器人動作，結束對零件的可使用月數和元素的運算。

格式

HealthRBStop 機器人編號

參數

機器人編號

以整數值(1~16)指定機器人編號。

說明

針對指定的機器人編號，結束對零件的可使用月數和元素(扭矩、速度、驅動量)的運算。

注意

- Auto模式下無法發揮功能。
- 空運行(包括虛擬控制器)時也無法發揮功能。
- 開始運算過1小時後，即自動結束。自動結束之後，一旦執行此命令則發生錯誤。
- 在未執行HealthRBStart命令的情況下，一旦執行此命令，會發生錯誤。
- 執行HealthRBStop後，在未執行HealthRBStart的情況下，重新執行HealthRBStop時，也會發生錯誤。

參照

HealthRBAnalysis、HealthRBStart、HealthRBTRQ、HealthRBSpeed、HealthRBDistance

HealthRBStop範例

```
Function RobotPartAnalysis
  Real month

  Robot 1

  HealthRBStart 1
  Motor On
  Go P0
  Go P1
  Motor Off
  HealthRBStop 1

  month = HealthRBAnalysis(1, HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE, 3)
  Print "Ball Screw Spline analysis =", Str$(month)
Fend
```

3.12.28 HealthRBTRQ

顯示影響到指定關節壽命的扭矩值。

格式

HealthRBTRQ [機器人編號] [, 關節編號]

參數

機器人編號

以整數值(1~16)指定機器人編號。可省略。若有省略，則使用目前機器人編號。

關節編號

以整數值(1~6)指定關節。未指定關節編號時，則傳回所有關節的值。本命令不可用於附加軸。

說明

針對從HealthRBStart到HealthRBStop為止的機器人動作，顯示影響到指定關節壽命的扭矩值。以0~1的實數值表示結果。最大扭矩值為「1」。

注意

- Auto模式下無法發揮功能。
- 空運行(包括虛擬控制器)時也無法發揮功能。

參照

HealthRBStart、HealthRBStop、ATRQ

HealthRBTRQ範例

這是顯示影響到SCARA機器人第1關節壽命的扭矩值之範例。

```
> HealthRBTRQ 1, 1
 0.020
>
```

3.12.29 HealthRBTRQ函數

用於傳回影響到指定關節壽命的扭矩值。

格式

HealthRBTRQ ([機器人編號,] 關節編號)

參數

機器人編號

以整數值(1~16)指定機器人編號。可省略。若有省略，則使用目前機器人編號。

關節編號

以整數值(1~6)指定關節。本命令不可用於附加軸。

傳回值

以0~1的實數值傳回。

說明

針對從HealthRBStart到HealthRBStop為止的機器人動作，傳回影響到指定關節壽命的扭矩值。以0~1的實數值表示結果。最大有效扭矩為1。

注意

- Auto模式下無法發揮功能。
- 空運行(包括虛擬控制器)時也無法發揮功能。

參照

HealthRBStart、HealthRBStop、ATRQ

HealthRBTRQ函數範例

```
Function RobotPartAnalysis
  Real healthTRQ

  Robot 1

  HealthRBStart 1
  Motor On
  Go P0
  Go P1
  Motor Off
  HealthRBStop 1

  healthTRQ = HealthRBTRQ(1,1)
  Print "Torque =", Str$(healthTRQ)
Fend
```

3.12.30 HealthRBWarningEnable

啟用或停用與機器人相關的指定類別零件之零件消耗警報通知。

格式

HealthRBWarningEnable 機器人編號, 零件類別[, On/Off]

參數

機器人編號

以整數值(1~16)指定機器人編號。

部件類型

以整數值(1~6)或以下所示之常數指定機器人相關部件。

常數	值	模式
HEALTH_ROBOT_TYPE_BATTERY	1	指定電池。
HEALTH_ROBOT_TYPE_BELT	2	指定皮帶。
HEALTH_ROBOT_TYPE_GREASE	3	指定潤滑脂。
HEALTH_ROBOT_TYPE_MOTOR	4	指定馬達。
HEALTH_ROBOT_TYPE_GEAR	5	指定減速機。
HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE	6	指定滾珠螺桿花鍵。

On/Off

- On：啟用部件消耗警報通知。
- Off：停用部件消耗警報通知。

結果

若省略On/Off參數，則顯示目前On/Off設定值。

說明

設定發出指定類別零件之零件消耗警報時是否進行零件消耗警報通知。

注意

停用指定零件之零件消耗警報通知後，即便已過建議更換時期，也不通知警告。執行時，請充分注意用本命令進行的設定。

參照

HealthRBAlarmOn

HealthRBWarningEnable範例

這是停用SCARA機器人1的潤滑脂零件之零件消耗警報通知的範例。

```
> HealthRBWarningEnable 1, HEALTH_ROBOT_TYPE_GREASE, Off
```

這是顯示SCARA機器人1的潤滑脂零件之零件消耗警報通知設定的範例。

```
> HealthRBWarningEnable 1, HEALTH_ROBOT_TYPE_GREASE
GREASE  Off
>
```

3.12.31 HealthRBWarningEnable函數

本函數用於傳回與機器人相關的指定類別零件之零件消耗警報通知設定狀態。

格式

HealthRBWarningEnable (機器人編號, 零件類別)

參數

機器人編號

以整數值(1~16)指定要傳回建議更換時期到期之前的剩餘月數之機器人編號。

部件類型

以整數值(1~6)或以下所示之常數指定要傳回建議更換時期到期之前的剩餘月數之部件。

常數	值	模式
HEALTH_ROBOT_TYPE_BATTERY	1	指定電池。
HEALTH_ROBOT_TYPE_BELT	2	指定皮帶。
HEALTH_ROBOT_TYPE_GREASE	3	指定潤滑脂。
HEALTH_ROBOT_TYPE_MOTOR	4	指定馬達。
HEALTH_ROBOT_TYPE_GEAR	5	指定減速機。
HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE	6	指定滾珠螺桿花鍵。

傳回值

用於以整數值傳回零件消耗警報的設定值。

- 1: On
- 0: Off

參照

HealthRBAlarmOn

HealthRBWarningEnable函數範例

這是顯示SCARA機器人1的潤滑脂零件之零件消耗警報通知設定狀態的範例。

```
Print HealthRBWarningEnable(1, HEALTH_ROBOT_TYPE_GREASE )
```

3.12.32 Here

用於將目前位置作為機器人坐標進行教導。

格式

Here 坐標

參數

坐標

用於指定P編號、P(運算式)、點標籤之一。

注意

- Here陳述式和平行處理

  1個動作命令中不可同時存在Here陳述式和平行處理。

  ```
  Go Here :Z(0) ! D10; MemOn 1 !
  ```

  不接受上述用法。

  ```
  P999 = Here
  Go P999 :Z(0) ! D10; MemOn 1 !
  ```

  請變更為如上程式。

- Here陳述式和多工

  在以Xqt等執行的多工函數內執行Here陳述式時，若機器人因主工作內的MoveGo等處於運作狀態，則會因錯誤而停止。

  可以CurPos取得動作中的目前位置。

  例

  ```
  Function Xqt_PrintHere
  Do
  Print CurPOS
  Wait 0.1
  Loop
  Fend
  Function main
  Xqt 10, Xqt_PrintHere
  Go P0
  Fend
  ```

- 與CP併用

  若在啟用路徑運動時使用Here，則在指定Here的動作之前將不會合成動作軌跡便進行減速。若採用以下的使用方法，則以P1減速度停止後將開始進行下一個動作。

  ```
  Go P1 CP
  Go Here+X(100)
  ```

若不想停止要繼續動作時，請修正程式以便事先計算Here的位置。

```
Go P1 CP
Go P1+X(100)
```

參照

Here函數、CurPos

Here範例

```
Here P1
Here pick
```

3.12.33 Here函數

用於傳回機器人目前位置。

格式
Here

傳回值
用於傳回機器人目前位置。

說明
Here函數用於取得使用中機械手的目前位置。

參照
Here函數

Here函數範例

```
P1 = Here
```

3.12.34 Hex$函數

用於將以十六進位表示的數值進行字元化後傳回。

格式

Hex$ (數值)

參數

數值

指定整數值。

傳回值

用於將以十六進位表示的數值作為ASCII值的字串後傳回。

說明

Hex$函數用於將以十六進位表示的數值進行字元化後傳回。以0~9、A~F構成字元。Hex$用於確認Stat函數的結果。

參照

Str$、Stat、Val

Hex$函數範例

```
> print hex$(stat(0))
A00000
> print hex$(255)
FF
```

3.12.35 History

顯示命令的執行歷程記錄。

格式

History

結果

依執行順序顯示命令的執行歷程記錄。

說明

顯示最近在命令視窗中執行的命令的執行歷程記錄(最多100件)。

注意

此命令僅可在命令視窗中執行。

參照

ClearHistory

> History範例

```
> Motor On
> Go P1
> History
Motor On
Go P1
```

3.12.36 Hofs

用於設定和顯示從編碼器原點到軟體原點的補償脈衝。

格式

(1) Hofs 第1關節設定值, 第2關節設定值, 第3關節設定值, 第4關節設定值 [, 第5關節設定值, 第6關節設定值] [, 第7關節設定值] [, 第8關節設定值, 第9關節設定值]

(2) Hofs

參數

第1關節設定值
　以運算式或數值指定第1關節補償脈衝值(整數)。
第2關節設定值
　以運算式或數值指定第2關節補償脈衝值(整數)。
第3關節設定值
　以運算式或數值指定第3關節補償脈衝值(整數)。
第4關節設定值
　以運算式或數值指定第4關節補償脈衝值(整數)。
第5關節設定值
　是用於垂直六軸機器人(包含N系列)的參數。以運算式或數值指定第5關節補償脈衝值(整數)。
第6關節設定值
　是用於垂直六軸機器人(包含N系列)的參數。以運算式或數值指定第6關節補償脈衝值(整數)。
第7關節設定值
　是用於關節型7軸機器人的參數。以運算式或數值指定第7關節補償脈衝值(整數)。
第8關節設定值
　是用於附加軸S關節的參數。以運算式或數值指定第8關節(附加軸S)補償脈衝值(整數)。
第9關節設定值
　是用於附加軸T關節的參數。以運算式或數值指定第9關節(附加軸T)補償脈衝值(整數)。

傳回值

未指定參數時，用於顯示目前的Hofs設定值。

說明

Hofs用於顯示或設定原點補償脈衝值。Hofs用於設定從編碼器原點(Z相)到機械原點為止的位移值。

雖然依據搭載於各關節的編碼器之原點來控制機器人動作，但是編碼器原點不一定與機器人的機械原點一致。因此，用Hofs設定補償脈衝量，以便將與機械原點一致的編碼器位置作為軟體上的原點。

注意

- 若非必要，切勿變更Hofs值

  出廠時已精密設定Hofs值。若在非必要情況下變更此值，會造成定位錯誤和非預期動作，非常危險。若非必要，切勿變更Hofs值。

- 複位JointAccuracy(僅限對應關節精度補償的機型)

  在對應關節精度補償的機型中，通過Hofs設定原點修正脈衝值時，對於有變更的關節，JointAccuracy設定的關節精度補償的補償值將複位為“0”。如果不想複位在JointAccuracy中設置的補償值，請使用HofsJointAccuracy。

- 要自動運算Hofs值時

  為了自動運算Hofs值，將手臂移到想要校正的位置上，以執行Calib。如此一來，控制器基於CalPls脈衝值以及校正位置脈衝值自動運算Hofs值。

- Hofs的儲存和還原

  可用[系統設定]選單-[系統設定]對話方塊-[機器人]-[校正]中的[儲存]及[載入]，對Hofs進行儲存及還原。

- 使用搭載Safety板控制器時，請在運行本命令以後啟動安全功能管理員

  使用搭載Safety板的控制器時，控制器Hofs值需要和安全功能的Safety板的Hofs值相匹配。

  運行本命令，僅改變控制器的Hofs值，這樣會因為與Safety板的Hofs值不匹配而發生警報。

  因此，運行版命令之後，請啟動安全功能管理員，更新Safety板設定。

  有關詳細資訊，請參閱以下手冊。
  「機器人控制器 安全功能手冊」

參照

Calib、CalPls、JointAccuracy、HofsJointAccuracy、Home、Hordr、MCal、SysConfig

Hofs範例

以下是監視器視窗中的簡易範例。將第1關節原點補償值設為-545，將第2關節原點補償值設為514，將第3關節和第4關節的原點補償值設為0，然後會顯示目前的原點補償值。

```
> hofs  -545, 514, 0, 0

> hofs
-545, 514, 0, 0
>
```

3.12.37 Hofs函數

本函數用於傳回各關節的編碼器原點到軟體原點的補償脈衝值。

格式

Hofs (關節編號)

參數

關節編號

以運算式或數值指定求出Hofs值的關節編號(整數)。附加軸的S軸為8，T軸為9。

傳回值

用於傳回指定關節的補償脈衝值(整數值，單位：脈衝)。

參照

Calib、CalPls、Home、Hordr、MCal、SysConfig

Hofs函數範例

以下為使用Hofs函數的程式範例。

```
Function DisplayHofs
   Integer i

   Print "Hofs settings:"
   For i = 1 To 4
      Print "Joint ", i, " = ", Hofs(i)
   Next i
Fend
```

3.12.38 HofsJointAccuracy

用於不變更關節精度補償之補償值，設定與顯示從編碼器原點到軟體原點的補償脈衝。

格式

(1) HofsJointAccuracy 第1關節設定值, 第2關節設定值, 第3關節設定值, 第4關節設定值 [, 第5關節設定值, 第6關節設定值] [, 第7關節設定值] [, 第8關節設定值, 第9關節設定值]

(2) HofsJointAccuracy

參數

第1關節設定值
　　以運算式或數值指定第1關節補償脈衝值(整數)。
第2關節設定值
　　以運算式或數值指定第2關節補償脈衝值(整數)。
第3關節設定值
　　以運算式或數值指定第3關節補償脈衝值(整數)。
第4關節設定值
　　以運算式或數值指定第4關節補償脈衝值(整數)。
第5關節設定值
　　是用於垂直六軸機器人(包含N系列)的參數。以運算式或數值指定第5關節補償脈衝值(整數)。
第6關節設定值
　　是用於垂直六軸機器人(包含N系列)的參數。以運算式或數值指定第6關節補償脈衝值(整數)。
第7關節設定值
　　是用於關節型7軸機器人的參數。以運算式或數值指定第7關節補償脈衝值(整數)。
第8關節設定值
　　是用於附加軸S關節的參數。以運算式或數值指定第8關節(附加軸S)補償脈衝值(整數)。
第9關節設定值
　　是用於附加軸T關節的參數。以運算式或數值指定第9關節(附加軸T)補償脈衝值(整數)。

傳回值

未指定參數時，用於顯示目前的Hofs設定值。

說明

HofsJointAccuracy用於不變更關節精度補償之補償值，顯示或設定原點補償脈衝值。關於可使用關節精度補償之機種，請參閱以下手冊。
「機械手手冊 」

雖然依據搭載於各關節的編碼器之原點來控制機器人動作，但是編碼器原點不一定與機器人的機械原點一致。因此，用Hofs設定補償脈衝量，以便將與機械原點一致的編碼器位置作為軟體上的原點。

對各關節皆有變更時，Hofs雖會將各自以JointAccuracy設定的關節精度補償之補償值重置為「0」，但使用HofsJointAccuracy則不會進行重置。不希望將關節精度補償之補償值重置為「0」時，請使用HofsJointAccuracy。

注意

- 若非必要，切勿變更Hofs值

  出廠時已精密設定Hofs值。若在非必要情況下變更此值，會造成定位錯誤和非預期動作，非常危險。若非必要，切勿變更Hofs值。

參照

JointAccuracy, Hofs

HofsJointAccuracy 範例

以下為命令視窗中的簡易範例。將第1關節原點補償值設為「-545」，將第2關節原點補償值設為「514」，將第3關節和第4關節的原點補償值設為「0」，然後顯示目前的原點補償值。於使用HofsJointAccuracy的前後，關節精度補償的補償值並沒有發生變化。

```
> JointAccuracy 1
1000, 420, 100, 240
> HofsJointAccuracy -545, 514, 0, 0

> HofsJointAccuracy
-545, 514, 0, 0
> JointAccuracy 1
  1000, 420, 100, 240
>
```

3.12.39 Home

用於將機器人手臂移到使用者定義的Home位置。

格式
Home

說明
依循以Hordr設定的關節順序，以低速PTP動作，移動到以HomeSet指定的Home位置(等待姿態)。

就水平多關節型機器人(包含RS系列)而言，通常首先第3關節返回到HomeSet位置之後，第1關節、第2關節、第4關節再同時返回到各自的HomeSet坐標位置。可以Hordr命令變更返回到Home位置的關節之順序。

注意

- Home狀態的輸出

  機器人位於Home位置(等待姿態)時，啟用控制器的Home輸出。

常見錯誤

- 在未設定HomeSet值的狀態下執行Home時

  若在未設定HomeSet值的狀態下執行Home，則發生錯誤2228。

參照
HomeClr、HomeDef、Hordr、HomeSet

Home範例
可在如下程式中使用Home命令。

```
Function InitRobot
   Reset
   If Motor = Off Then
      Motor On
   EndIf
   Home
Fend
```

或在命令視窗中進行如下發行操作。

```
> home
>
```

3.12.40 HomeClr

用於清除Home位置設定。

格式

HomeClr

參照

HomeDef、HomeSet

HomeClr範例

可在如下程式中使用HomeClr函數。

```
Function ClearHome

   If HomeDef = True Then
      HomeClr
   EndIf
Fend
```

3.12.41 HomeDef函數

用於傳回有無設定Home位置。

格式

HomeDef

傳回值

若有設定，則傳回「True」；若無設定，則傳回「False」。

參照

HomeClr、HomeSet

HomeClr範例

可在如下程式中使用HomeDef函數。

```
Function DisplayHomeSet

   Integer i

   If HomeDef = False Then
      Print "Home is not defined"
   Else
      Print "Home values:"
      For i = 1 To 4
         Print "J", i, " = ", HomeSet(i)
      Next i
   EndIf
Fend
```

3.12.42 HomeSet

用於設定和顯示Home位置(等待姿態)的姿態脈衝。

格式

(1) HomeSet 第1關節脈衝值, 第2關節脈衝值, 第3關節脈衝值, 第4關節脈衝值 [, 第5關節脈衝值, 第6關節脈衝值] [, 第7關節脈衝值] [, 第8關節脈衝值, 第9關節脈衝值]

(2) HomeSet

參數

第1關節脈衝值
　　以運算式或數值指定第1關節Home編碼器脈衝值(整數)。
第2關節脈衝值
　　以運算式或數值指定第2關節Home編碼器脈衝值(整數)。
第3關節脈衝值
　　以運算式或數值指定第3關節Home編碼器脈衝值(整數)。
第4關節脈衝值
　　以運算式或數值指定第4關節Home編碼器脈衝值(整數)。
第5關節脈衝值
　　是用於垂直六軸機器人(包含N系列)的參數。以運算式或數值指定第5關節Home編碼器脈衝值(整數)。
第6關節脈衝值
　　是用於垂直六軸機器人(包含N系列)的參數。以運算式或數值指定第6關節Home編碼器脈衝值(整數)。
第7關節脈衝值
　　是用於關節型7軸機器人的參數。以運算式或數值指定第7關節Home編碼器脈衝值(整數)。
第8關節脈衝值
　　是用於附加軸S關節的參數。以運算式或數值指定第8關節(附加軸S)Home編碼器脈衝值(整數)。
第9關節脈衝值
　　是用於附加軸T關節的參數。以運算式或數值指定第9關節(附加軸T)Home編碼器脈衝值(整數)。

結果

若省略參數，則顯示目前Home脈衝值。

說明

使用者可對各關節指定脈衝值，並設定Home (等待) 位置。

常見錯誤

- 在未設定HomeSet值的狀態下執行Home時

  若在未設定HomeSet值的狀態下執行Home，則發生錯誤2228。

- 在未設定HomeSet值的狀態下想要顯示HomeSet值時

  若在未設定HomeSet值的狀態下想要顯示Home位置的脈衝值，則發生錯誤2228。

參照

Home、HomeClr、HomeDef、Hordr、Pls

HomeSet範例

以下是命令視窗中的操作範例。

```
> homeset 0,0,0,0  '將Home位置設為0, 0, 0, 0
> homeset
  0  0
  0  0
```

```
> home                    '機器人移至Home位置
```

使用Pls函數，將手臂的目前位置設為Home位置。

```
> homeset Pls(1), Pls(2), Pls(3), Pls(4)
```

3.12.43 HomeSet函數

本函數用於傳回指定關節的Home位置(等待姿態)之脈衝值。

格式

HomeSet (關節編號)

參數

關節編號

以運算式或數值指定求出HomeSet值的關節編號。附加軸的S軸為8，T軸為9。

傳回值

用於傳回指定關節的Home脈衝值。將「0」指定為關節編號之後，若有設定HomeSet值，則傳回「1」；若未設定，則傳回「0」。

參照

HomeSet

HomeSet函數範例

以下是使用HomeSet函數的程式範例。

```
Function DisplayHomeSet

   Integer i

   If HomeSet(0) = 0 Then
      Print "HomeSet is not defined"
   Else
      Print "HomeSet values:"
      For i = 1 To 4
         Print "J", i, " = ", HomeSet(i)
      Next i
   EndIf
Fend
```

3.12.44 Hordr

用於指定和顯示執行Home時的各關節動作順序。

格式

(1) Hordr 設定值1, 設定值2, 設定值3, 設定值4 [, 設定值5] [, 設定值6] [, 設定值7] [, 設定值8] [, 設定值9]

(2) Hordr

參數

設定值1
　　以位元模式指定在第1步驟復歸的關節。
設定值2
　　以位元模式指定在第2步驟復歸的關節。
設定值3
　　以位元模式指定在第3步驟復歸的關節。
設定值4
　　以位元模式指定在第4步驟復歸的關節。
設定值5
　　以位元模式指定在第5步驟復歸的關節。
設定值6
　　以位元模式指定在第6步驟復歸的關節。
設定值7
　　以位元模式指定在第7步驟復歸的關節。
設定值8
　　以位元模式指定在第8步驟復歸的關節。
設定值9
　　以位元模式指定在第9步驟復歸的關節。

結果

若省略參數，則顯示目前的復歸順序設定。

說明

Hordr用於設定執行Home命令時的關節復歸順序。確定哪一個關節在第幾個順位移到Home位置(等待姿態)。

Hordr命令可供使用者變更執行Home時的各關節復歸順序。依據機器人機種，分4個、6個或9個步驟設定復歸順序。使用者可用Hordr指定在各步驟進行復歸的關節。還可指定在各步驟進行復歸的數個關節。以理論上來說，所有關節皆可同時復歸。然而，若是水平多關節型機器人(包含RS系列的4軸機器人)，一般都建議在最初的步驟1移動第3關節，然後在後續步驟讓其它關節復歸。

執行Hordr命令時，必須對各步驟指定位元模式。規定了對各關節的位元模式。若在某個步驟中位元為「1」，相應關節則移至Home位置(等待姿態)。若將位元清除為「0」，相應關節則不在該步驟中移至Home位置(等待姿態)。對各關節分配的位元模式如下所示。

位元模式表

關節名稱	位元編號	二進位表示
第1關節	bit 0	&B0001
第2關節	bit 1	&B0010
第3關節	bit 2	&B0100
第4關節	bit 3	&B1000
第5關節	bit 4	&B10000

關節名稱	位元編號	二進位表示
第6關節	bit 5	&B100000
第7關節	bit 6	&B1000000
第8關節	bit 7	&B10000000
第9關節	bit 8	&B100000000

參照

Home、HomeSet

Hordr範例

以下是命令視窗中的水平多關節型機器人(包含RS系列的4軸機器人)的操作範例。

在此例中，以二進位按如下所述設定到機器人手臂Home位置的復歸順序。

第3關節在第1步驟復歸到Home位置；第1關節在第2步驟復歸到Home位置；第2關節在第3步驟復歸到Home位置；第4關節在第4步驟復歸到Home位置。

```
>hordr  &B0100, &B0001, &B0010, &B1000
```

在此例中，以十進位按如下所述設定到機器人手臂Home位置的復歸順序。第3關節在第1步驟復歸到Home位置；第1關節、第2關節和第4關節在第2步驟同時復歸到Home位置。

```
>hordr  4, 11, 0, 0
```

在以下範例中，以十進位表示目前的復歸順序。

```
>hordr
4, 11, 0, 0
>
```

3.12.45 Hordr函數

用於傳回執行Home時的各指定關節的動作順序。

格式

Hordr (設定值編號)

參數

設定值編號

指定表示Hordr命令的動作順序之整數值。

傳回值

用於傳回指定動作順序的Horder設定值之數值。

參照

Home、HomeSet

Hordr函數範例

```
Integer a
a = Hordr(1)
```

3.12.46 Hour

用於顯示控制器的累計通電時間。

格式

Hour

說明

打開控制器電源以顯示累計時間。(累計通電時間)累計時間的單位為小時。

參照

Time

Hour範例

在命令視窗中執行如下範例。

```
> hour
2560
>
```

3.12.47 Hour函數

用於傳回控制器的累計通電時間。

格式

Hour

傳回值

用於以實數值傳回控制器的累計通電時間。

參照

Time

Hour函數範例

```
Print "Number of controller operating hours: ", Hour
```

3.13 I

3.13.1 If...Then...Else...EndIf

依照指定的條件分支來執行命令。

格式

(1) If條件運算式 Then
陳述式T1
.
.
[ElseIf 條件運算式 Then]
陳述式T1
.
.
[Else]
陳述式F1
.
.
EndIf

(2) If 條件運算式 Then 陳述式T1 [;陳述式T2...] [Else 陳述式F1 [;陳述式F2...]]

參數

條件運算式
　是返回真假值(True/False值)的有效條件運算式。若為真(True)，則傳回「0」以外的數值；若為假(False)時，則傳回「0」。(請參閱以下條件運算式範例。)
陳述式T1
　條件運算式的值為真(True)時，也就是符合條件時，則予以執行。(可在If...Then...Else形式的區塊中，記述數個陳述式。)
陳述式F1
　條件運算式的值為假(False)時，也就是不符合條件時，則予以執行。(可在If...Then...Else形式的區塊中，記述數個陳述式。)

說明

(1) If...Then...Else用於符合條件時執行陳述式T1以後的部分。不符合條件時，則執行陳述式F1以後的部分。可省略If...Then...Else命令的Else部分。省略Else部分而不符合條件時，則執行EndIf以後的陳述式。無論有無Else，都請用EndIf關閉If...Then...Else陳述式的區塊。

(2) If...Then...Else也可使用區塊以外的形式。這是將所有If...Then...Else的陳述式都記述在相同行的方法。若要按此方法使用If...Then...Else，則不需要EndIf。若符合此行記述的條件運算式(條件運算式的值為真(True)時)，則執行Then和Else之間的陳述式。若不符合條件運算式(條件運算式的值為假(False)時)，則執行Else之後的陳述式。If...Then...Else的Else部分並非必須的。若不符合條件運算式(條件運算式的值為假(False)時)且無Else關鍵字，控制則移至程式中的下一個陳述式。

注意

- 條件運算式範例
  - a = b：a等於b
  - a < b：a小於b
  - a >= b：a大於或等於b
  - a <> b：a不等於b

- a > b：a大於b
- a <= b：a小於或等於b

還可使用邏輯運算子And、Or、Xor等。

- 在條件運算式中使用True時

常數True的值為-1，但由於是Boolean型，因此用於和其它型態變數的比較條件時需加以注意。

```
Function main
  Integer i
  i = 3
  If i = True Then
    Print "i=TRUE"
  EndIf
Fend
```

若執行上述程式，則顯示「i=TRUE」。這是因為用「0」或「非0」執行包括Boolean型在內的條件判定。

「i」的值不是「0」時，則判定為條件成立並進行顯示。

參照

Else、Select...Case、Do...Loop

If/Then/Else範例

[1行的If...Then...Else]

如下所述為檢查輸入以確定特定輸出的啟用/停用的簡易範例。這一類的工作用於始終運行的I/O工作等。

```
Function main
    Do
       If Sw(0) = 1 Then On 1 Else Off 1
    Loop
Fend
```

[區塊形式的If...Then...Else]

如下所述是檢查多個輸入值並輸出該輸入狀態的簡易範例。

```
If Sw(0) = 1 Then Print "Input0 ON" Else Print "Input0 OFF"
'
If Sw(1) = 1 Then
    If Sw(2) = 1 Then
       Print "Input1 On and Input2 ON"
    Else
       Print "Input1 On and Input2 OFF"
    EndIf
Else
    If Sw(2) = 1 Then
       Print "Input1 Off and Input2 ON"
    Else
       Print "Input1 Off and Input2 OFF"
    EndIf
EndIf
```

[其它格式範例]

```
If x = 10 And y = 3 Then GoTo 50
If test <= 10 Then Print "Test Failed"
If Sw(0) = 1 Or Sw(1) = 1 Then Print "Everything OK"
```

3.13.2 ImportPoints

用於將點檔案匯入目前選擇的機器人專案中。

格式

ImportPoints 來源路徑,檔名 [, 機器人編號]

參數

來源路徑

是表示想要在目前專案中匯入的特定資料夾或檔案的字串運算式。副檔名為「.pts」或「.pnt」(Epson RC+ 3.x/4.x形式)。詳細內容請參閱ChDisk。

檔案名稱

表示想要在目前機器人之目前專案中匯入的特定檔案之字串運算式。副檔名為pts。不可指定路徑。此外，不受ChDisk等的影響。詳細內容請參閱ChDisk。

機器人編號

是用於指定哪個機器人與點檔案相關的整數運算式可省略。機器人編號=0時，則將點檔案視為通用點檔案進行匯入。若有省略，則使用目前機器人編號。

說明

ImportPoints用於將點檔案複製到目前選擇的專案中，並將該點檔案新增於目前選擇的機器人檔案中。編譯新增的點檔案後，可用LoadPoints命令載入。目前選擇的機器人中已存在相同檔案時，則進行覆寫，然後會再次編譯。

常見錯誤

- 沒有檔案時

  不存在指定的來源路徑時，會發生錯誤。

- 找不到指定檔案時

  檔名中含有路徑時會發生錯誤。

- 指定的檔案並非目前機器人的檔案時

  若在檔名上指定其它機器人的點檔案，則發生錯誤。

參照

LoadPoints、Robot、SavePoints

ImportPoints範例

```
Function main
  Robot 1
  ImportPoints "c:\mypoints\model1.pts", "robot1.pts"
  LoadPoints "robot1.pts"
Fend
```

3.13.3 In函數

用於以位元組單位傳回指定輸入連接埠的狀態。位元組連接埠由8個輸入位元構成。

格式

In (位元組連接埠編號)

參數

位元組連接埠編號

指定I/O的位元組連接埠。

傳回值

用於傳回0~255的整數。傳回值為8位元。各位元對應1個輸入位元。

說明

可透過In同時查看8個輸入位元的值。In命令的用法為，可將8個I/O位元的狀態作為1個變數值進行儲存。另外，透過同時使用Wait命令，「等待直到2個以上的I/O位元狀態符合特定條件」。

1次可檢查8個輸入位元，因此傳回值為0~255範圍的整數值。關於傳回值的各整數值如何對應各自的輸入位元，請參閱下表。

輸入位元表(位元組連接埠為0時)

傳回值	7	6	5	4	3	2	1	0
1	Off	Off	Off	Off	Off	Off	Off	On
5	Off	Off	Off	Off	Off	On	Off	On
15	Off	Off	Off	Off	On	On	On	On
255	On	On	On	On	On	On	On	On

輸入位元表(位元組連接埠為2時)

傳回值	23	22	21	20	19	18	17	16
3	Off	Off	Off	Off	Off	Off	On	On
7	Off	Off	Off	Off	Off	On	On	On
32	Off	Off	On	Off	Off	Off	Off	Off
255	On	On	On	On	On	On	On	On

參照

InBCD、MemIn、MemOff、MemOn、MemSw、Off、On、OpBCD、Oport、Out、Sw、Wait

In函數範例

以下程式範例基於如下假定進行記述；將輸入位元20、21、22、23連接於感測器裝置，並在各自傳回表示「準備OK」的ON信號後才啟動應用程式。在程式範例中，取得位元組連接埠2的8個輸入位元狀態並確認輸入位元20、21、22、23皆為ON後再進入下一步。若任一輸入位元不是ON(用任一輸入位元傳回「1」)，則顯示錯誤訊息並停止工作。

此時，為了讓輸入位元20、21、22、23皆為ON，In(2)的傳回值必須大於240。為了確認這一點，對變數「var1」與數值239進行比較。(在此，無需意識到輸入位元16、17、18、19。若傳回值為240~255的值，則繼續執行程式。)

```
Function main
    Integer var1
    var1 = In(2)    '取得位元組連接埠2的8個輸入位元之狀態
    If var1 > 239 Then
        Go P1
        Go P2
        '在此處執行其它動作命令
        '.
        '.
    Else
        Print "Error in initialization!"
        Print "Sensory Inputs not ready for cycle start"
        Print "Please check inputs 20,21,22 and 23 for"
        Print "proper state for cycle start and then"
        Print "start program again"
    EndIf
Fend
```

雖然無法在命令視窗中設定輸入，但可以進行確認。在以下範例中，以輸入位元1、5、15是ON為前提。其它輸入皆為OFF。

```
> print In(0)
34
> print In(1)
128
> print In(2)
0
```

3.13.4 InBCD函數

本函數用於將輸入連接埠作為每8位元的群組，並以BCD傳回輸入連接埠狀態的函數。

格式

InBCD (連接埠編號)

參數

連接埠編號

指定I/O的輸入位元組。

傳回值

以BCD(0~9)傳回輸入連接埠(0~99)的狀態。

說明

在InBCD中，以BCD同時讀取8個輸入線路。使用InBCD命令的參數「連接埠編號」指定要讀取狀態的位元群組。例如，連接埠編號 = 0時，讀取輸入位元0~7的狀態；連接埠編號 = 1時，讀取輸入位元8~15的狀態。

用BCD傳回8個輸入位元的狀態。傳回值為0~99的1或2位數值。第1位數(10的位數)與以連接埠編號指定的8個輸出位元當中的高階4位元對應。第2位數(1的位數)與以連接埠編號指定的8個輸出位元當中的低階4位元對應。

各位數的BCD有效值為0~99，因此無法實現所有I/O組合。下表所示為，連接埠編號為0時可實現的I/O狀態之部分組合和傳回值。

輸入設定(輸入編號)

傳回值	7	6	5	4	3	2	1	0
01	Off	Off	Off	Off	Off	Off	Off	On
02	Off	Off	Off	Off	Off	Off	On	Off
03	Off	Off	Off	Off	Off	Off	On	On
08	Off	Off	Off	Off	On	Off	Off	Off
09	Off	Off	Off	Off	On	Off	Off	On
10	Off	Off	Off	On	Off	Off	Off	Off
11	Off	Off	Off	On	Off	Off	Off	On
99	On	Off	Off	On	On	Off	Off	On

BCD只能用於指定十進位。這表示特定連接埠的所有輸入位元同時為ON時，無法以使用BCD的InBCD命令求出有效值。InBCD的最大傳回值為「99」。從上表可知，傳回值為「99」時，所有輸入不會變為ON。當傳回值為「99」時，ON的輸入位元為0、3、4、7，其它位元皆為OFF。

注意

- InBCD和In的差異

  在SPEL+中，InBCD和In命令有幾個重要之處並不相同，以下將說明其差異。

- 在InBCD命令中，以BCD指定8個輸入位元的傳回值。而BCD則不可使用&HA、&HB、&HC、&HD、&HE、&HF等的值，因此無法實現8個輸入位元的所有組合。

- In命令的功能非常類似於InBCD命令，但可傳回8個輸入位元的所有傳回值。(In命令的傳回值為0~255；InBCD的傳回值為0~99)因此，可實現1群組8個輸入位元的所有組合。

參照

In、MemOff、MemOn、MemOut、MemSw、Off、On、OpBCD、Oport、Out、Sw、Wait

InBCD範例

這是命令視窗中的簡易操作範例。

假定輸入位元0、4、10、16、17、18為ON，其它輸入位元為OFF。

```
> Print InBCD(0)
11
> Print InBCD(1)
04
> Print InBCD(2)
07
>
```

3.13.5 Inertia

用於設定機器人的負載慣性和偏心量。

格式

(1) Inertia [負載慣性] [, 偏心量]

(2) Inertia

參數

負載慣性

以數值或運算式指定包括夾具與工件在內的尖端關節中心周圍之慣性(實數，單位：kgm2)。可省略(但不可以僅省略負載偏心量)。

偏心量

以數值或運算式指定距離夾具/工件重心的尖端關節中心之長度(實數，單位：mm)。可省略。

結果

若省略參數，則顯示目前設定的慣性參數。

如果省略 [偏心量] 則會設置成輸入的[負載慣性]，並設置預設值 [偏心量]。

不能只省略[負載慣性]。

請注意，如果慣性值或偏心量的值小於實際值，則設置過大的加速和減速值，可能會損壞機械臂。此外，它可能導致錯誤和衝擊，不僅無法發揮功能，而且可能會縮短部件的使用壽命，或導致皮帶齒跳引起的位置偏移。

說明

使用Inertia陳述式指定尖端關節周圍的轉動慣量。可藉此適當地補償尖端關節的加減速與伺服增益。此外，可使用偏心量參數指定從尖端關節中心到夾具與工件重心的距離。

還可以使用「負載、慣性、偏心/偏移測量實用程式」進行設置。

有關詳細資訊，請參閱以下手冊。
「Epson RC+ 使用者指南 - 負載、慣性、偏心/偏移測量實用程式」

注意

- 即使關閉電源，Inertia的設置也不會更改

  設置后，Inertia的設置將記住在控制器中。即使關閉電源不會更改。

  如果未設置任何內容，則該值將保留為上次設置的值。

參照

Inertia函數

有關末端夾具命令的更多資訊，請參閱以下手冊。
「Hand功能」

Inertia範例

```
Inertia 0.02, 1
```

3.13.6 Inertia函數

用於傳回慣性的參數值。

格式

Inertia (參數編號)

參數

參數編號

用於指定以下整數值。

- 0：機器人支援慣性參數時，傳回「1」; 不支援時，傳回「0」。
- 1：傳回負載慣性。(單位：kgm2)
- 2：傳回偏心量。(單位：mm)

傳回值

用於以實數值傳回指定的設定。

參照

Inertia

有關末端夾具命令的更多資訊，請參閱以下手冊。
「Hand功能」

Inertia函數範例

```
Real loadInertia, eccentricity

loadInertia = Inertia(1)
eccentricity = Inertia(2)
```

3.13.7 InPos函數

用於傳回機器人的定位狀態。

格式

InPos

傳回值

- 完成定位時：True
- 機器人動作中：False

參照

CurPos、FindPos、WaitPos

InPos函數範例

```
Function main

  P0 = XY(0, -100, 0, 0)
  P1 = XY(0, 100, 0, 0)

  Xqt MonitorPosition
  Do
    Jump P0
    Wait .5
    Jump P1
    Wait .5
  Loop

Fend

Function MonitorPosition

  Boolean oldInPos, pos

  Do
    Pos = InPos
    If pos <> oldInPos Then
      Print "InPos = ", pos
    EndIf
    oldInPos = pos
  Loop

Fend
```

3.13.8 Input

接收來自顯示裝置的輸入，並儲存為變數。

格式

Input 變數名稱 [, 變數名稱, 變數名稱,...]

參數

變數名稱

指定變數名稱。若要指定多個變數，則用「,」進行分隔。此時的「,」稱之為分隔符號。

說明

接收來自顯示裝置的資料，並指派給指定的變數。

執行命令時，顯示裝置上則顯示提示「?」。輸入資料後，按下鍵盤上的Return鍵。

注意

- 數值輸入的規則

  輸入數值時，若有分隔符號以外的非數值資料，則無條件捨去該非數值資料和其後的資料。

- 字串輸入的規則

  指派給字串時，則將數字和字母作為字元予以處理。

- 其它Input命令相關規則
  - 若要在代入指派目的地指定多個變數時，則用分隔符號「,」分隔各自要指派的數值資料。
  - 雖可指定數值變數和字串變數，但輸入資料型態必須符合指派目的地的變數型態。

常見錯誤

- 指定變數的數量和輸入資料的數量不符時

  若指定多個變數，則輸入資料的數量必須和指定變數的數量一致。以命令指定的變數之數量和從鍵盤接收的數值資料之數量不符時，則發生錯誤2505。

參照

Input #、Line Input、Line Input #、Print、String

Input範例

以下是簡易Input陳述式範例。

```
Function InputNumbers
    Integer A, B, C

    Print "Please enter 1 number"
    Input A
    Print "Please enter 2 numbers separated by a comma"
    Input B, C
    Print "A = ", A
    Print "B = ", B, "C = ", C
Fend
```

運行上述程式時，則執行下一階段。

```
Please enter 1 number
?-10000
Please enter 2 numbers separated by a comma
?25.1, -10000
A = -10000
B = 25 C = -10000
```

3.13.9 Input

用於從檔案、通訊連接埠、資料庫或裝置輸入字串或數值資料，並儲存為變數。

格式

Input #連接埠編號, 變數名稱 [, 變數名稱, 變數名稱,...]

參數

連接埠編號

是表示檔案、通訊連接埠、資料庫或設備的ID編號。檔案編號是以ROpen、WOpen、AOpen等陳述式指定的編號。通訊連接埠編號是以OpenCom(RS-232C)或OpenNet(TCP/IP)陳述式指定的編號。資料庫編號是以OpenDB陳述式指定的編號。

裝置ID為以下數值。

- 21 RC+
- 20 TP4

變數名稱

指定接收資料的變數名稱。

說明

Input #命令用於從以連接埠編號指定的裝置，接收數值或字串資料，並將該資料輸入到指定的變數。

注意

- 支援的控制器型號

  如果在T/VT系列中指定RS-232C埠時會發生錯誤。

- 數值輸入的規則

  輸入數值時，若有分隔符號以外的非數值資料，則無條件捨去該非數值資料和其後的資料。

- 字串輸入的規則

  指派給字串時，則將數字和字母作為字元予以處理。

- 最大資料長度

  可透過本命令一次處理的最大資料長度為256 Byte。不過，以資料庫為對象時，最大資料長度為4096 Byte。以通訊連接埠(TCP/IP)為對象時，最大資料長度則為1024 Byte。

- 其它Input #命令相關規則

  - 若要在指派目的地指定多個變數時，則用分隔符號「,」或空白「 」分隔各自要指派的數值資料。
  - 若要指定數個字串變數以及數值變數和字串變數兩者，務必用分隔符號「,」或空白「 」分隔數值資料；若為字串資料，則務必用分隔符號「,」進行分隔。
  - 但輸入資料型態必須符合指派目的地的變數型。
  - 在控制器之間使用通訊連接埠交換字串變數、數值變數的範例
  - 傳送側(任一模式皆可)

```
Print #PortNum, "$Status,", InData, OutData
Print #PortNum, "$Status", ",",InData, OutData
```

- 接收側

```
Input #PortNum, Response$, InData, OutData
```

常見錯誤

- 指定變數的數量和輸入資料的數量不符時

  以命令指定的變數之數量和從裝置接收的數值資料之數量不符時，則發生錯誤2505。

參照

Input, Line Input, Line Input #, Print #, Read, ReadBin

Input #範例

以下是使用Input #陳述式的簡易範例。

```
Function GetData
    Integer A
    String B$

    OpenCom #1
    Print #1, "Send"
    Input #1, A   '從連接埠#1取得數值
    Input #1, B$  '從連接埠#1取得字串
    CloseCom #1
Fend
```

3.13.10 InputBox

用於顯示輸入對話方塊。等待使用者輸入字元或按一下按鈕，並傳回輸入的內容。

格式

InputBox 提示, 標題, 預設, 輸入字串

參數

提示
　　顯示於對話方塊的訊息字串
標題
　　顯示於對話方塊標題列的字串
預設
　　在文字方塊預設顯示的字串未設定預設值時，則設定空白。(" ")
輸入字串
　　設定用戶輸入的字串的字串型變數用戶按一下取消時，此變數會被設為「@」。

說明

InputBox用於顯示對話方塊，並等待使用者按一下[OK]按鈕或[取消]按鈕。在參數的輸入字串中設定使用者輸入的字串。

參照

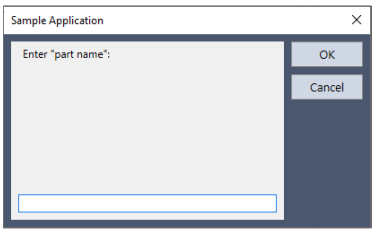
MsgBox

InputBox範例

此程式為InputBox陳述式的範例。

```
Function GetPartName$ As String
    String prompt$, title$, answer$

    prompt$ = "Enter " + Chr$(34) + "part name" + Chr$(34) + ":"
    title$ = "範例"
    InputBox prompt$, title$, "", answer$
    If answer$ <> "@" Then
        GetPartName$ = answer$
    EndIf
Fend
```

下圖所示為前述程式範例的結果。



限制事項

參數的提示、標題以及預設值中包括半形逗號「,」時，無法正確顯示字串。因此請使用不含半形逗號的字串。

3.13.11 InReal函數

用於將2字組(32位元)的輸入資料作為32位元浮點數資料(符合IEEE754)進行載入。

格式

InReal (字組連接埠編號)

參數

字組連接埠編號
　　指定I/O輸入字組連接埠。

傳回值

用於以Real型實數傳回輸入連接埠的狀態。

說明

從以字組連接埠編號指定的輸入字組連接埠取得作為IEEE754之Real型值的2輸入字組值。字組連接埠編號參數可使用輸入字組標籤。

本函數不用作Wait命令以及Till、Find、Sense的條件。

參照

In、InW、InBCD、Out、OutW、OpBCD、OutReal

InW函數範例

```
Real realVal

realVal = InReal(32)
```

3.13.12 InsideBox函數

用於傳回進入檢測區的檢測狀態。

格式

InsideBox (區域編號 [, 機器人編號 | All])

參數

區域編號

指定要傳回狀態的工作空間編號(1~15的整數)。

機器人編號

以整數值指定要檢測的機器人編號。省略機器人編號時，則以目前選擇的機器人為對象。若有指定All，只要有1台機器人進入，就傳回True。

傳回值

若機器人的手臂尖端位置進入指定的進入檢測區，則傳回True；若未進入，則傳回False。

參照

Box、BoxClr、BoxDef、GetRobotInsideBox、InsidePlane

注意

在Epson RC+5.0中，雖可與Wait命令組合等待InsideBox函數結果，但在Epson RC+ 6.0、RC+ 7.0、Epson RC+ 8.0中不可與Wait命令組合。在這種情況下，請使用GetRobotInsideBox函數，以代替InsideBox函數。

對應表

RC+版本	機器人控制器	Wait	Till、Find、Sense、Trap	進行Print等不屬於左側的命令/分支判定處理	GetRobotInsideBox函數的利用
RC+ 8.0	RC700系列	不可	不可	可	皆可
RC+ 8.0	RC90系列	不可	不可	可	皆可
RC+ 7.0	RC700系列	不可	不可	可	皆可
RC+ 7.0	RC90系列	不可	不可	可	皆可
RC+ 6.0	RC620	不可	不可	可	皆可
RC+ 5.0	RC90系列	可	不可	可	不可

- 不可：不可使用的組合
- 可：可使用的組合
- 皆可：可用於Wait、Till、Find、Sense、Trap、Print等顯示/分支判定處理

InsideBox函數範例

以下程式是判斷機器人1是否進入區域No.3的範例。

```
Function PrintInsideBox
If InsideBox(3,1) = True Then
Print "Inside Box3"
Else
Print "Outside Box3"
EndIf
Fend
```

3.13.13 InsidePlane函數

用於傳回進入檢測平面的檢測狀態。

格式

InsidePlane (平面編號[, 機器人編號 | All])

參數

平面編號

指定傳回狀態的工作平面編號(1~15的整數)。

機器人編號

以整數值指定要檢測的機器人編號。省略機器人編號時，則以目前選擇的機器人為對象。若有指定All，只要有1台機器人進入，就傳回True。

傳回值

若機器人的手臂尖端位置進入指定的進入檢測平面，則傳回「True」；若未進入，則傳回「False」。

參照

InsideBox、GetRobotInsidePlane、Plane、PlaneClr、PlaneDef

注意

在Epson RC+5.0中，雖可與Wait命令組合等待InsidePlane函數結果，但在Epson RC+ 6.0、RC+ 7.0、Epson RC+ 8.0中不可與Wait命令組合。在這種情況下，請使用GetRobotInsidePlane函數，以代替InsidePlane函數。

對應表

RC+版本	機器人控制器	Wait	Till、Find、Sense、Trap	進行Print等不屬於左側的命令/分支判定處理	GetRobotInsidePlane函數的利用
RC+ 8.0	RC700系列	不可	不可	可	皆可
RC+ 8.0	RC90系列	不可	不可	可	皆可
RC+ 7.0	RC700系列	不可	不可	可	皆可
RC+ 7.0	RC90系列	不可	不可	可	皆可
RC+ 6.0	RC620	不可	不可	可	皆可
RC+ 5.0	RC90系列	可	不可	可	不可

- 不可：不可使用的組合
- 可：可使用的組合
- 皆可：可用於Wait、Till、Find、Sense、Trap、Print等顯示/分支判定處理

InsidePlane函數範例

以下程式是判斷機器人1是否進入平面No.3的範例。

```
Function PrintInsidePlane
If InsidePlane(3,1) = True Then
Print "Inside Plane3"
Else
Print "Outside Plane3"
EndIf
Fend
```

3.13.14 InStr函數

用於從字串中搜尋字串，並傳回發現的位置。

格式

InStr (字串, 搜尋字串)

參數

字串
　　指定被查找的字串。
查找字串
　　指定要查找的字串。

傳回值

若發現要搜尋的字串，則傳回該位置；若未發現，則傳回-1。

參照

Mid$

Instr函數範例

```
Integer pos

pos = InStr("abc", "b")
```

3.13.15 Int函數

用於將實數值轉換為整數值。傳回無條件捨去小數部分後的數值。

格式

Int (數值)

參數

數值

用於指定實數值。

傳回值

用於將以參數設定的實數值轉換為整數值並進行傳回。

說明

Int (數值)用於傳回無條件捨去小數部分後的數值。

注意

- 數值小於0時(負值時)

  若參數小於0，則朝負值方向傳回無條件捨去後的數值。(數值為-1.35時，則傳回-2。)

參照

Abs、Atan、Atan2、Cos、Mod、Not、Sgn、Sin、Sqr、Str$、Tan、Val

Int函數範例

在命令視窗中執行如下範例。

```
> Print Int(5.1)
5
> Print Int(0.2)
0
> Print Int(-5.1)
-6
>
```

3.13.16 Int32

宣告Int32型變數。(4位元組整數型變數)

格式

Int32 變數名稱 [(陣列變數的最大元素編號)] [, 變數名稱 [(陣列變數的最大元素編號)]...]

參數

變數名稱

指定要進行變數宣告的變數名稱。

陣列變數的最大元素編號

是陣列變數的最大元素編號，最大可宣告三維。使用如下格式。可省略。

(最大元素編號1, [最大元素編號2], [最大元素編號3])

元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。在所有元素數不超過以下最大值的範圍內，指定各最大元素編號。

- 本地變數：2,000
- 備份變數(Global Preserve)：4,000
- 全域變數和模組變數：100,000

說明

Int32用於宣告整數型變數宣告。整數型變數範圍：-2147483648~2147483647。在Function的開頭宣告本地變數。在Function之外宣告全域變數和模組變數。

參照

Boolean、Byte、Double、Global、Int64、Integer、Long、Real、Short、String、UByte、UInt32、UInt64、UShort

Int32範例

以下為使用Int32宣告整數型變數的程式範例。

```
Function int32test
    Int32 A(10)          'Int32型的一維陣列
    Int32 B(10, 10)      'Int32型的二維陣列
    Int32 C(5, 5, 5)     'Int32型的三維陣列
    Int32 var1, arrayvar(10)
    Integer i
    Print "Please enter an Integer Number"
    Input var1
    Print "The Integer variable var1 = ", var1
    For i = 1 To 5
        Print "Please enter an Integer Number"
        Input arrayvar(i)
        Print "Value Entered was ", arrayvar(i)
    Next i
Fend
```

3.13.17 Int64

宣告Int64型變數。(8位元組整數型變數)

格式

Int64 變數名稱 [(陣列變數的最大元素編號)] [, 變數名稱 [(陣列變數的最大元素編號)]...]

參數

變數名稱

指定要進行變數宣告的變數名稱。

陣列變數的最大元素編號

是陣列變數的最大元素編號，最大可宣告三維。使用如下格式。可省略。

(最大元素編號1, [最大元素編號2], [最大元素編號3])

元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。在所有元素數不超過以下最大值的範圍內，指定各最大元素編號。

- 本地變數：2,000
- 備份變數(Global Preserve)：4,000
- 全域變數和模組變數：100,000

說明

Int64用於宣告整數型變數。整數型變數範圍：-9223372036854775808~9223372036854775807。在Function的開頭宣告本地變數。在Function之外宣告全域變數和模組變數。

參照

Boolean、Byte、Double、Global、Int32、Integer、Long、Real、Short、String、UByte、UInt32、UInt64、UShort

Int64範例

以下為使用Int64宣告整數型變數的程式範例。

```
Function int64test
    Int64 A(10)          'Int64型的一維陣列
    Int64 B(10, 10)      'Int64型的二維陣列
    Int64 C(5, 5, 5)     'Int64型的三維陣列
    Int64 var1, arrayvar(10)
    Integer i
    Print "Please enter an Integer Number"
    Input var1
    Print "The Integer variable var1 = ", var1
    For i = 1 To 5
        Print "Please enter an Integer Number"
        Input arrayvar(i)
        Print "Value Entered was ", arrayvar(i)
    Next i
Fend
```

3.13.18 Integer

用與宣告Integer型變數。(2位元組整數型變數)

格式

Integer 變數名稱 [(陣列變數的最大元素編號)] [, 變數名稱 [(陣列變數的最大元素編號)]...]

參數

變數名稱

指定要進行變數宣告的變數名稱。

陣列變數的最大元素編號

是陣列變數的最大元素編號，最大可宣告三維。使用如下格式。可省略。

(最大元素編號1, [最大元素編號2], [最大元素編號3])

元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。在所有元素數不超過以下最大值的範圍內，指定各最大元素編號。

- 本地變數：2,000
- 備份變數(Global Preserve)：4,000
- 全域變數和模組變數：100,000

說明

Integer用於宣告整數型變數。整數型變數範圍：-32768~32767。在Function的開頭宣告本地變數。在Function之外宣告全域變數和模組變數。

參照

Boolean、Byte、Double、Global、Int32、Int64、Long、Real、Short、String、UByte、UInt32、UInt64、UShort

Integer範例

以下為使用Integer宣告整數型變數的程式範例。

```
Function inttest
    Integer A(10)          'Integer型的一維陣列
    Integer B(10, 10)      'Integer型的二維陣列
    Integer C(5, 5, 5)     'Integer型的三維陣列
    Integer var1, arrayvar(10)
    Integer i
    Print "Please enter an Integer Number"
    Input var1
    Print "The Integer variable var1 = ", var1
    For i = 1 To 5
        Print "Please enter an Integer Number"
        Input arrayvar(i)
        Print "Value Entered was ", arrayvar(i)
    Next i
Fend
```

3.13.19 InW函數

用於以字組為單位傳回輸入連接埠的狀態。字組連接埠由16個輸入位元構成。

格式

InW (字組連接埠編號)

參數

字組連接埠編號

指定I/O的字組連接埠。

傳回值

用於傳回輸入連接埠的狀態(0~65535的Long型整數)。

注意

- 包括即時I/O輸入位元在內的字組連接埠相關規則

  字組連接埠=1時，以0~255的整數傳回輸入連接埠的狀態。

  未反映即時I/O的輸入位元。

參照

In、Out、OutW

InW函數範例

```
Long word0

word0 = InW(0)
```

3.13.20 IODef函數

用於傳回有無定義指定的輸入/輸出位元、位元組、字組或I/O標籤。

格式

IODef (IO型, IO寬度, 連接埠編號)

IODef (IO標籤)

參數

IO型

表示I/O類型的整數值

- 0 - 輸入
- 1 - 輸出
- 2 - 記憶體

IO寬

表示連接埠寬度的整數值：1(位元)、8(位元組)或16(字)

連接埠編號

表示用於傳回標籤的位元、位元組或字的整數值

IO標籤

是以字串指定標準I/O或記憶體I/O標籤的字串運算式。

傳回值

若有定義指定的輸入/輸出位元、位元組、字組或I/O標籤，則傳回「True」；除此之外的情況，傳回「False」。

參照

IOLabel$、IONumber

IODef函數範例

```
Integer i

     For i = 0 To 15
       If IODef( 0, 1, i) = TRUE Then
         Print "Port " , i, " is defined"
       Else
         Print "Port " , i, " is undefined"
       EndIf
     Next i
```

3.13.21 IOLabel$函數

用於傳回指定的輸入/輸出位元、位元組、字組的I/O標籤。

格式

IOLabel$ (IO型, IO寬度, 連接埠編號)

參數

IO型

表示I/O類型的整數值

- 0 - 輸入
- 1 - 輸出
- 2 - 記憶體

IO寬

表示連接埠寬度的整數值：1(位元)、8(位元組)、16(字)

連接埠編號

表示用於傳回標籤的位元、位元組或字的整數值

傳回值

用於傳回包括標籤的字串。

參照

PLabel$、IONumber、IODef

IOLabel$函數範例

```
Integer i

For i = 0 To 15
  Print "Input ", i, ": ", IOLabel$(0, 1, i)
Next i
```

3.13.22 IONumber函數

傳回指定I/O標籤的I/O連接埠編號。

格式

IONumber (IO標籤)

參數

IO標籤

是以字串指定標準I/O或記憶體I/O標籤的字串運算式。

傳回值

用於傳回指定I/O標籤的I/O連接埠編號(位元、位元組、字組)。不存在指定的I/O標籤時，會發生錯誤。

參照

IOLabel$、IODef

IONumber函數範例

```
Integer IObit

IObit = IONumber("myIO")

IObit = IONumber("Station" + Str$(station) + "InCycle")
```

3.14 J

3.14.1 J1Angle

設定點的J1Angle屬性。

格式

(1) J1Angle 點指定 [, 設定值]

(2) J1Angle

參數

點指定

用於指定P編號、P(運算式)、點標籤之一。

設定值

以實數值指定設定值。可省略。

結果

僅限於RS系列和N系列機器人方可使用J1Angle屬性。

若省略設定值參數，則顯示對於指定點的J1Angle值。若2個參數皆有省略，則顯示目前機器人位置的J1Angle值。

- RS系列：指定特殊姿態第1關節的角度值，其中點的X坐標值為「0」，Y坐標值也為「0」。對於未形成特殊姿態的點，J1Angle屬性值不具有任何意義。
- N系列：在如下情況下指定第1關節角度值：第1關節、第4關節、第6關節的軸心呈直線狀時；第1關節和第6關節的軸心呈直線狀時；第1關節和第4關節的軸心呈直線狀時。對於未形成特殊姿態的點，J1Angle屬性值不具有任何意義。

參照

Hand、J1Angle函數、J1Flag、J2Flag、J4Angle、J4Angle函數

J1Angle範例

```
J1Angle P0, 10.0
J1Angle P(mypoint), 0.0
```

3.14.2 J1Angle函數

用於傳回點的J1Angle屬性。

格式

J1Angle [(點指定)]

參數

點指定

以點運算式進行指定。可省略。若省略，則傳回機器人目前位置的J1Angle設定。

傳回值

僅限於RS系列和N系列機器人方可使用J1Angle屬性。

- RS系列：以實數值傳回特殊姿態第1關節的角度值，其中點的X坐標值為「0」，Y坐標值也為「0」。
- N系列：在如下情況下，以實數值傳回第1關節角度值：第1關節、第4關節、第6關節的軸心呈直線狀時；第1關節和第6關節的軸心呈直線狀時；第1關節和第4關節的軸心呈直線狀時。

參照

Hand、J1Angle、J1Flag、J2Flag、J4Angle、J4Angle函數

J1Angle函數範例

```
Print J1Angle(pick)
Print J1Angle(P1)
Print J1Angle
```

3.14.3 J1Flag

設定點的J1Flag屬性。

格式

(1) J1Flag 點指定 [, 設定值]

(2) J1Flag

參數

點指定

用於指定P編號、P(運算式)、點標籤之一。

設定值

以整數或運算式指定以下任一項。可省略。

- RS系列時
  - 0 (/J1F0)：J1的範圍：-90 ~ +270(單位：度)
  - 1 (/J1F1)：J1的範圍：-270 ~ -90 或 +270 ~ +450(單位：度)
- C8、C12系列時
  - 0 (/J1F0)：J1的範圍：-180 ~ +180(單位：度)
  - 1 (/J1F1)：J1的範圍：-240 ~ -180 或 +180 ~ +240(單位：度)

結果

J1Flag屬性用於指定對於1個點的第1關節值範圍。若省略設定值參數，則顯示對於指定點的J1Flag值。若2個參數皆有省略，則顯示目前機器人位置的J1Flag值。

參照

Hand、J1Flag函數、J2Flag

J1Flag範例

```
J1Flag P0, 1
J1Flag P(mypoint), 0
```

3.14.4 J1Flag函數

用於傳回點的J1Flag屬性。

格式

J1Flag [(點指定)]

參數

點指定

以點運算式進行指定。可省略。若省略，則傳回機器人目前位置的J1Flag設定。

傳回值

- 0: /J1F0
- 1: /J1F1

參照

Hand、J1Flag、J2Flag

J1Flag函數範例

```
Print J1Flag(pick)
Print J1Flag(P1)
Print J1Flag
Print J1Flag(Pallet(1, 1))
```

3.14.5 J2Flag

設定點的J2Flag屬性。

格式

(1) J2Flag 點指定 [, 設定值]

(2) J2Flag

參數

點指定

用於指定P編號、P(運算式)、點標籤之一。

設定值

以整數或運算式指定以下任一項。可省略。

- 0 (/J2F0)：J2的範圍：-180 ~ +180(單位：度)
- 1 (/J2F1)：J2的範圍：-360 ~ -180 或 +180 ~ +360(單位：度)

結果

J2Flag屬性用於指定對於1個點的第2關節值範圍。若省略設定值參數，則顯示對於指定點的J2Flag值。若2個參數皆有省略，則顯示目前機器人位置的J2Flag值。

參照

Hand、J1Flag、J2Flag函數

J2Flag範例

```
J2Flag P0, 1
J2Flag P(mypoint), 0
```

3.14.6 J2Flag函數

用於傳回點的J2Flag屬性。

格式

J2Flag [(點指定)]

參數

點指定

以點運算式進行指定。可省略。若省略，則傳回機器人目前位置的J2Flag設定。

傳回值

- 0: /J2F0
- 1: /J2F1

參照

Hand、J1Flag、J2Flag

J2Flag函數範例

```
Print J2Flag(pick)
Print J2Flag(P1)
Print J2Flag
Print J2Flag(P1 + P2)
```

3.14.7 J4Angle

設定點的J4Angle屬性。

格式

(1) J4Angle 點指定 [, 設定值]

(2) J4Angle

參數

點指定

用於指定P編號、P(運算式)、點標籤之一。

設定值

以實數值指定設定值。可省略。

結果

僅限於N系列的機器人方可使用J4Angle屬性。

指定第4關節和第6關節的軸心呈直線狀時的第4關節角度值。

對於未形成特殊姿態的點，J4Angle屬性值不具有任何意義。

若省略設定值參數，則顯示對於指定點的J4Angle值。若2個參數皆有省略，則顯示目前機器人位置的J4Angle值。

參照

Hand、J1Angle、J1Angle函數、J4Angle函數

注意

如下所述，使用J4Flag和J4Angle兩者時，以J4Angle為優先。

```
J4Angle P0,0
J4Flag P0,1
```

J4Angle範例

```
J4Angle P0, 10.0
J4Angle P(mypoint), 0.0
```

3.14.8 J4Angle函數

用於傳回點的J4Angle屬性。

格式

J4Angle [(點指定)]

參數

點指定

以點運算式進行指定。可省略。若省略，則傳回機器人目前位置的J4Angle設定。

傳回值

用於以實數值傳回第4關節和第6關節的軸心呈直線狀時的第4關節角度值。

僅限於N系列的機器人方可使用J4Angle屬性。

參照

Hand、J1Angle、J1Angle函數、J4Angle

J4Angle函數範例

```
Print J4Angle(pick)
Print J4Angle(P1)
Print J4Angle
```

3.14.9 J4Flag

設定點的J4Flag屬性。

格式

(1) J4Flag 點指定 [, 設定值]

(2) J4Flag

參數

點指定

用於指定P編號、P(運算式)、點標籤之一。

設定值

以整數或運算式指定以下任一項。可省略。

- 0 (/J4F0)：J4的範圍：-180 ~ +180(單位：度)
- 1 (/J4F1)：J4的範圍：-360 ~ -180 或 +180 ~ +360(單位：度)

結果

J4Flag屬性用於指定對於1個點的第4關節值範圍。若省略設定值參數，則顯示對於指定點的J4Flag值。若2個參數皆有省略，則顯示目前機器人位置的J4Flag值。

參照

Elbow、Hand、J4Flag函數、J6Flag、Wrist

J4Flag範例

```
J4Flag P0, 1
J4Flag P(mypoint), 0
```

3.14.10 J4Flag函數

用於傳回點的J4Flag屬性。

格式

J4Flag [(點指定)]

參數

點指定

以點運算式進行指定。可省略。若省略，則傳回機器人目前位置的J4Flag設定。

傳回值

- 0: /J4F0
- 1: /J4F1

參照

Elbow、Hand、Wrist、J4Flag、J6Flag

J4Flag函數範例

```
Print J4Flag(pick)
Print J4Flag(P1)
Print J4Flag
Print J4Flag(Pallet(1, 1))
```

3.14.11 J6Flag

設定點的J6Flag屬性。

格式

(1) J6Flag 點指定 [, 設定值]

(2) J6Flag

參數

點指定

用於指定P編號、P(運算式)、點標籤之一。

設定值

指定整數或運算式。

- 範圍：對於0到127(/J6F0 ~ /J6F127)
- 指定點的J6範圍如下所述。
  - (-180 * (設定值+1) < J6 <= -180 *設定值)
  - (180 *設定值< J6 <= 180 * (設定值+1))

結果

J6Flag屬性用於指定對於1個點的第6關節值的範圍。若省略設定值參數，則顯示對於指定點的J6Flag值。若2個參數皆有省略，則顯示目前機器人位置的J6Flag值。

參照

Elbow、Hand、J4Flag、J6Flag函數、Wrist

注意

J6Flag範圍因機種而異。
有關範圍的更多資訊，請參閱各機械臂手冊的動作區域。

J6Flag範例

```
J6Flag P0, 1
J6Flag P(mypoint), 0
```

3.14.12 J6Flag函數

用於傳回點的J6Flag屬性。

格式

J6Flag [(點指定)]

參數

點指定

以點運算式進行指定。可省略。若省略，則傳回機器人目前位置的J6Flag設定。

傳回值

0 ～ 127: /J6F0 ～ /J6F127

參照

Elbow、Hand、Wrist、J4Flag、J6Flag

J6Flag函數範例

```
Print J6Flag(pick)
Print J6Flag(P1)
Print J6Flag
Print J6Flag(P1 + P2)
```

3.14.13 JA函數

用於傳回根據關節角度計算的機器人坐標。

格式

JA(第1關節位置,第2關節位置,第3關節位置,第4關節位置[, 第5關節位置, 第6關節位置] [, 第7關節位置] [, 第8關節位置, 第9關節位置])

參數

第1關節位置~第9關節位置以實數值指定關節角度(單位：deg)。若為直動關節，則以mm為單位指定。第5關節位置、第6關節位置為垂直6軸型機器人(包含N系列)以及關節型6軸機器人專用的參數。第7關節位置為關節型7軸機器人專用的參數。第8關節位置、第9關節位置為附加軸S及T關節專用的參數。

補充說明

若有指定動作範圍外的角度，會發生動作範圍外錯誤。

結果

傳回以指定關節位置表示的機器人坐標。

說明

若要以關節角度指定機器人坐標，則使用JA函數。

用JA函數傳回的點形成機器人的特殊姿態時，即便對該點執行動作命令，其角度也不一定與作為JA函數的引數賦予的關節角度一致。若要按以JA函數指定的關節角度運作時，需避開機器人的特殊姿態。

[例]

```
> go ja(0,0,0,90,0,-90)
> where
WORLD:  X:    0.000 mm  Y:  655.000 mm  Z:  675.000 mm  U:    0.000 deg V:  -90.000
deg W:  -90.000 deg
JOINT:  1:    0.000 deg 2:    0.000 deg 3:    0.000 deg 4:    0.000 deg 5:    0.000
deg 6:    0.000 deg
PULSE:  1:        0 pls 2:        0 pls 3:        0 pls 4:        0 pls 5:        0
pls 6:        0 pls
> go ja(0,0,0,90,0.001,-90)
> where
WORLD:  X:   -0.004 mm  Y:  655.000 mm  Z:  675.000 mm  U:    0.000 deg V:  -90.000
deg W:  -89.999 deg
JOINT:  1:    0.000 deg 2:    0.000 deg 3:    0.000 deg 4:   90.000 deg 5:    0.001
deg 6:  -90.000 deg
PULSE:  1:        0 pls 2:        0 pls 3:        0 pls 4:  2621440 pls 5:       29
pls 6: -1638400 pls
```

參照

AglToPls、XY

JA函數範例

```
P10 = JA(60, 30, -50, 45)
GO JA(135, 90, -50, 90)
P3 = JA(0, 0, 0, 0, 0, 0)
```

3.14.14 Joint

以關節坐標系顯示目前機器人位置。

格式

Joint

參照

Pulse、Where

Joint範例

```
>joint
JOINT:  1: -6.905 deg 2: 23.437 deg 3: -1.999 mm  4: -16.529 deg
>
```

3.14.15 JointAccuracy

用於設定和顯示關節精度補償之補償值。

格式

(1) JointAccuracy 關節編號, 第1設定值, 第2設定值, 第3設定值, 第4設定值

(2) JointAccuracy 關節編號

參數

關節編號
　　指定關節編號。
第1設定值
　　以數值指定第1設定值(整數)。範圍為0～2000。
第2設定值
　　以數值指定第2設定值(整數)。範圍為0～999。
第3設定值
　　以數值指定第3設定值(整數)。範圍為0～2000。
第4設定值
　　以數值指定第4設定值(整數)。範圍為0～999。

各機械手可啟用關節精度補償的關節不同，如為不可用的關節，將發生錯誤。關於可啟用關節，請參閱以下手冊。
「機械手手冊」

傳回值

若使用(2)，則顯示對應關節編號的目前關節精度補償之補償值。

說明

JointAccuracy用於設定指定關節的補償值。設定適當的補償值，將提升軌跡精度。

注意

- 若非必要，切勿變更JointAccuracy

  出廠時已精密設定JointAccuracy。若在非必要情況下變更此值，會造成軌跡精度的下降。於執行校正精靈時會自動設定JointAccuracy，故若非必要，切勿變更JointAccuracy。

- 執行Calib與Hofs

  於已設定JointAccuracy的狀態下執行Calib、Hofs命令，則曾有變更的關節的關節精度補償之補償值將為「0」。若希望不變更JointAccuracy補償值並變更Hofs值，請執行HofsJointAccuracy。

參照

HofsJointAccuracy, Calib, Hofs

JointAccuracy範例

以下為命令視窗中的簡易範例。將第1關節的關節精度補償之補償值的第1設定值設為「1000」，第2設定值設為「420」，第3設定值設為「100」，第4設定值設為「240」。其後，顯示目前第1關節的關節精度補償之補償值。

```
> JointAccuracy 1, 1000, 420, 100, 240

> JointAccuracy 1
1000, 420, 100, 240
>
```

3.14.16 JointAccuracy函數

顯示關節精度補償之補償值。

格式

JointAccuracy(関節番号, 設定番号)

參數

關節編號
　　指定關節編號。
設定編號
　　使用以下常數或整數值(1~4)指定顯示的補償值。

常數	值	
JAC_PARAM1	1	第1設定值
JAC_PARAM2	2	第2設定值
JAC_PARAM3	3	第3設定值
JAC_PARAM4	4	第4設定值

傳回值

傳回指定關節設定編號相應的關節精度補償之補償值(整數值)。

參照

JointAccuracy

JointAccuracy函數範例

以下為JointAccuracy函數的使用範例。

```
Function main
    '設置關節#2的精度補償值
    JointAccuracy 2, 32, 873, 6, 437
    '顯示指定關節的精度補償值
    DisplayJointAccuracy(2)
Fend

Function DisplayJointAccuracy(numJoint As Integer)
    Integer i
    Print "Joint =", numJoint, ", JointAccuracy settings:"
    For i = 1 To 4
        Print "  Param ", i, " = ", JointAccuracy(numJoint, i)
    Next i
Fend

[輸出結果]
Joint = 2, JointAccuracy settings:
  Param  1 =  32
  Param  2 =  873
  Param  3 =  6
  Param  4 =  437
```

3.14.17 JRange

以脈衝值設定指定關節的容許動作區域。

格式

JRange 關節編號, 下限脈衝值, 上限脈衝值

參數

關節編號
　　以1~9的整數值設定JRange指定的關節編號。附加軸的S軸為8，T軸為9。
下限脈衝值
　　以運算式或數值對指定關節動作範圍的下限脈衝值進行指定。
上限脈衝值
　　以運算式或數值對指定關節動作範圍的上限脈衝值。

說明

用上限脈衝值和下限脈衝值設定指定關節的動作範圍。Range指令需指定所有關節的動作範圍，但對於JRange指令來說，由於設定每1關節的動作區域，因此可減少參數。使用Range命令，以確認已設定的動作區域。

注意

- 請遵守下限脈衝值≦上限脈衝值

  設定JRange命令的下限脈衝值時，請勿使其大於上限脈衝值。下限脈衝值若大於上限脈衝值，會發生錯誤，從而無法執行動作命令。

- Jrange設定值的變更

  一旦用JRange設定數值，只要未以Range或JRange命令變更，就會繼續保持。關閉電源時不改變用JRange設定的限制值。

- 動作區域的上下限

  動作區域的上限脈衝值因各機械手的機種而異，因此關於區域的上限脈衝值設定，請參閱以下手冊。
  「機械手手冊 」

參照

Range、JRange函數

JRange範例

以下是命令視窗中的操作範例。

```
> JRange 2, -6000, 7000     '定義第2關節的動作範圍

> JRange 1, 0, 7000     '定義第1關節的動作範圍
```

3.14.18 JRange函數

本函數用於傳回指定關節的容許動作區域脈衝值(範圍設定值)。

格式

JRange (關節編號, 參數編號)

參數

關節編號

以運算式或數值指定要參照的關節編號(1~9的整數)。附加軸的S軸為8，T軸為9。

參數編號

以整數值指定以下2個值中的任一值。

- 1：指定下限脈衝值
- 2：指定上限脈衝值

傳回值

用於傳回指定關節的範圍設定值(整數值，單位：脈衝)。

參照

Range、JRange

JRange函數範例

```
Long i, oldRanges(3, 1)

For i = 0 To 3
    oldRanges(i, 0) = JRange(i + 1, 1)
    oldRanges(i, 1) = JRange(i + 1, 2)
Next i
```

3.14.19 JS函數

本函數用於在執行Jump Sense、Jump3 Sense、Jump3CP Sense、JumpTLZ Sense後，傳回Sense輸入是否成立。

格式

JS

傳回值

傳回「True」或「False」。

- True：Sense輸入條件成立並且手臂停止於目標坐標上方時，JS即傳回「True」。
- False：若手臂到達以Jump命令設定的目標位置並完成正常動作，JS則傳回「False」。

說明

JS與Jump或Sense命令結合使用。JS命令的目的在於，傳回Jump命令所指示的動作中，是否成立(用Sense設定的)輸入條件。輸入條件成立後，JS即傳回「True」。若在輸入條件未成立的情況下，手臂到達目標位置，JS則傳回「False」。

JS僅是用於確認狀態的命令，並不具有產生動作或在動作期間檢查輸入的功能。若要指示動作，則使用Jump命令。但有時會用Sense命令，以便在以Jump命令發生的動作中檢查特定輸入。

注意

- JS只和此前使用的Jump命令、Jump3命令、Jump3CP、JumpTLZ命令配合使用

  JS僅用於檢查此前使用之Jump命令的輸入(受到Sense命令指示)。若啟動下一個Jump命令，JS命令則傳回該新Jump命令的狀態。無法傳回對於Jump命令的最初狀態資料。對於需要檢查狀態的Jump命令，務必在此之後進行JS檢查。

參照

JT、Jump、Jump3、Jump3CP、JumpTLZ、Sense

JS函數範例

```
Function SearchSensor As Boolean
    Sense Sw(5) = On

    Jump P0
    Jump P1 Sense
    If JS = TRUE Then
        Print "Sensor was found"
        SearchSensor = TRUE
    EndIf
Fend
```

3.14.20 JT函數

是用於傳回在此之前Jump、Jump3、Jump3CP、JumpTLZ動作結果的函數。

格式
JT

傳回值
用於設定或清除以下形式的Long型數值位元。

- Bit 0：上升動作開始時，或上升動作量0時為1
- Bit 1：水平動作開始時，或水平移動量0時為1
- Bit 2：下降動作開始時，或下降動作量0時為1
- Bit 16：上升動作完成時，或上升動作量0時為1
- Bit 17：水平動作完成時，或水平動作量0時為1
- Bit 18：下降動作完成時，或下降動作量0時為1

說明
可對以下內容進行檢查：在此之前執行的Jump命令是否有成立以Sense、Till、Abort等指定的動作停止條件。

參照
JS、Jump、Jump3、Jump3CP、JumpTLZ、Sense、Till

JT函數範例

```
Function SearchTill As Boolean

    Till Sw(5) = On

    Jump P0
    Jump P1 Till
    If JT And 4 Then
        Print "Motion stopped during descent"
        SearchTill = TRUE
    EndIf
Fend
```

3.14.21 JTran

用於進行無須原點復歸的一個關節的PTP操作。

格式

JTran 關節編號, 移動量

參數

關節編號

以整數值指定要動作的關節編號。附加軸的S軸為8，T軸為9。

移動量

指定實數。旋轉關節以deg為單位指定；直動關節以mm為單位指定。

說明

JTran用於僅使指定關節從目前位置進行指定量的動作。

參照

Go、Jump、Move、PTran

JTran範例

```
JTran 1, 20
```

3.14.22 Jump

透過門形動作(先垂直上升，接著水平移動，最後垂直下降的門形動作)，從目前位置到指定位置，對手臂執行PTP動作。

格式

Jump 目標坐標 [C Arch編號] [LimZ [Z坐標值]] [CP] [{SenseTillFind}] [!平行處理!] [SYNC]

參數

目標坐標
　以點數據指定目標位置。
Arch編號
　Arch編號用於對Jump的Arch動作形狀指定使用哪一個Arch表的設定。請在Arch編號的開頭處附加字母字元「C」。有效值為C0~C7。可省略。
Z坐標值
　請將該值視為用Jump命令可在動作期間移動第3關節的最大值(Z限制值)，或用Jump的Z成分高度頂峰值。只要是有效的第3關節坐標值，任何值都可以。可省略。
CP
　指定路徑運動。可省略。
Sense、Till或Find
　記述Sense、Till、Find中的任一運算式。可省略。

```
Sense | Till | Find
Sense Sw(運算式) = {On | Off}
Till Sw(運算式) = {On | Off}
Find Sw(運算式) = {On | Off}
```

!平行處理!
　可在Jump命令動作時增加平行處理陳述式，以便執行I/O或其它命令。可省略。
SYNC
　用於預約動作命令。在以SyncRobots開始動作之前，機器人不進行動作。

說明

Jump命令用於透過「Arch動作(Arch型動作)」將手臂從目前位置移往目標坐標。換句話說，可視為1次執行3個動作的陳述式。例如，若有定義Arch編號，便執行1次Jump命令進行以下3種動作。

1. 首先，第3關節動作到以Jump命令中Arch編號運算的Z軸高度。
2. 接著，手臂朝Z軸方向上升到以LimZ指定之Z限制位置，並朝目標坐標進行水平移動。然後，第1關節、第2關節、第4關節各自進行動作，並開始朝Z軸方向下降。手臂會動作直到獲取最終X、Y、U坐標位置。
3. 手臂只朝Z軸方向移動直到獲取目標Z坐標位置，在獲取目標坐標後結束Jump命令。

由於無法在Jump命令中指定目標坐標(移動的目的位置)，因此必須在執行Jump命令前進行教導。以Accel執行Jump移動的加速和減速。此外，以Speed執行移動的速度。

不可對垂直6軸型機器人(包含N系列)執行Jump。請使用Jump3。

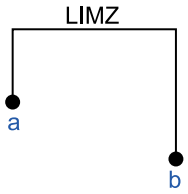
- 關於CP

  若已添加CP參數，可在開始動作減速時重疊下一個動作命令的加速。此時，不在目標坐標上進行定位。

- 關於Arch編號

  可依Jump命令中指定的Arch編號，變更Jump的Arch型態。這樣的話，第1關節、第2關節、第4關節在動作之前，則可確定要朝Z軸方向移動的距離。可在Jump命令中有效使用的Arch編號值為C0~C7的值。使用者可以Arch命令定義對於C0~C6值的Arch表值。然而，C7始終被定義為「門形動作」。「門形動作」是指，機器人在運作第1關節、第2關節、第4關節之前，只先將第3關節移至以LimZ定義的坐標位置之動作。在進行「門形動作」時，一旦移至LimZ定義的Z限制

值位置，便開始進行第1關節、第2關節、第4關節的動作。第1關節、第2關節、第4關節各自移至最終目標坐標位置後，便以以目標坐標定義的最終Z坐標位置為目標，使第3關節下降。下圖表示「門形動作」的運作情形。



符號	說明
a	目前位置
b	目標坐標

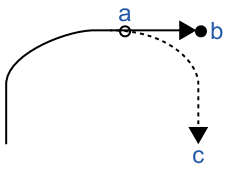
- 關於LimZ

  LimZ Z坐標值用於在目前設定的本地坐標系之水平移動面上指定Z坐標最高值。雖然第1關節、第2關節、第4關節可能會依據指定的Arch設定在到達LimZ值之前開始動作，但LimZ值始終被用於定義這些動作的Z坐標方向上限值。若省略LimZ參數，則以在此之前(最後)以LimZ指定的最高值為基準。

  由LimZ指定的高度方向限制值為本地機器人坐標系上的Z坐標值。也就是說，這並非是Arm或Tool坐標的Z坐標值。因此，若要使用作業高度不同的工具或抓手，請充分注意並採取必要的措施。

- 關於Sense

  Sense是可省略的一個參數。Sense用於要在第3關節進行最後下降動作之前，檢查輸入條件或記憶體I/O條件等情況。若無問題，機器人手臂會停於目標坐標(最後只剩第3關節動作的位置)，並結束執行此命令(Jump)。但請注意，即便檢測到以Sense指定的條件，機器人手臂也不會馬上停下來。



符號	說明
a	檢查條件
b	動作結束
c	目標坐標

  可組合使用JS或Stat等命令以確認以下情況：Sense條件是否成立，機器人手臂是否停在目標坐標位置之前；Sense條件是否未成立，機器人手臂是否停在目標坐標位置上。

- 關於Till

  在執行Jump前，可使用選項Till設定條件，讓機器人減速並停止。可透過包括檢查I/O輸入之一或記憶體I/O之一的條件來進行設定。為此，要使用Sw或MemSw函數。可依照事先設定的條件，檢查輸入為ON或OFF，讓手臂減速並停止。

  可透過使用Stat函數確認以下情況，即，是否成立Till條件而執行此命令，或者機器人是否在Till條件未成立的狀態下停在目標坐標位置上。

注意

- Jump不可用於垂直6軸型機器人(包含N系列)。

  請使用Jump3或Jump3CP。

- 省略Arch編號參數時

  若省略Arch編號參數的設定，執行Jump命令時的預設Arch值則變為C7。如上述所言，Arch值為C7時會以「門形動作」進行移動(請參閱上述內容)。

- Jump和Jump3、Jump3CP的差異

  Jump3和Jump3CP可用於垂直6軸型機器人(包含N系列)，但不可對其使用Jump。水平多關節型機器人(包含RS系列)在朝Z軸方向進行上升/下降時，可用Jump縮短動作時間。也可在Z軸以外的方向執行Jump3的接近動作和閃避動作。

- Jump和Go的差異

  Jump和Go最重要的差異如下：Go時，所有關節動作皆為同步動作，各關節同時動作/停止。而就Jump而言，在開始和結束動作時，第3關節只在垂直方向移動。在進行裝置的吸附和配置等作業時，建議使用此命令。

- Jump命令下的減速和停止

  在Jump命令下，手臂必定在減速的同時停在目標坐標上。

- Jump命令下的適當速度和加速指示

  以Speed和Accel分別設定Jump動作的機器人速度和加減速。請注意，Speed和Accel為僅在進行點到點(如Jump、Go等)的動作時方可設定的命令。例如，若為用於執行直線/曲線動作的命令(諸如Move或Arc)，請使用SpeedS或AccelS命令。

  此外，若為Jump，則可對第3關節的上升移動、包括第4關節旋轉在內的水平動作以及第3關節的下降移動分別設定速度和加減速。

- Jump的Pass功能

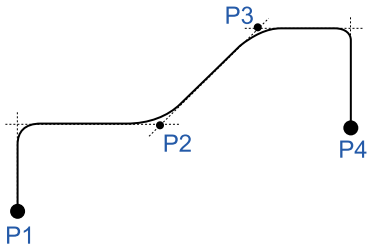
  若在下降動作量為0的Jump上附加CP參數，可在不減速並停止該Jump的水平動作之狀態下，繼續順暢連上PTP動作。

  此外，若在此前的PTP動作命令上附加CP參數，上升動作量為0的Jump可在不未減速並停止該PTP動作之狀態下順暢連上Jump的水平動作。

  要將平常的Jump水平動作(1個PTP動作)替換成順暢連上數個PTP動作時，這個功能非常有用。

  [例]

```
Go P1
Jump P2 :Z(-50) C0 LimZ -50 CP
Go P3 :Z(0) CP
Jump P4 C0 LimZ 0
```



使用Arch時的重要事項

由於透過軌跡控制進行執行第3關節的上升/下降動作與橫向動作的合成，因此不能保證拱形動作的實際軌跡。軌跡會因動作速度和手臂的動作方式而異。請以作業時使用的實際速度和姿態確認實際軌跡。

- 在相同位置上，即便執行帶有相同[C Arch編號]的Jump命令，低速時的軌跡也會比高速動作時低。因此，請注意：即便確認不會因高速而撞擊干擾物，仍有可能在低速動作時發生撞擊。
- 相較於低速動作，高速動作會增加垂直上升量，呈現出垂直下降量減小的傾向。未出現預期的垂直下降距離時，請降低速度或減速度，或將下降距離設得長一些。
- 即便是相同距離的動作，軌跡也會依手臂的動作方式而異。雖然會依手臂的動作方式，而有各種軌跡變化，但如果以一般水平多關節型機器人為例的話，越是大幅移動第1手臂，越會增加垂直上升量，從而呈現出減小垂直下降量的傾向。未出現預期的垂直下降距離時，請降低速度或減速度，或將下降距離設得長一些。

常見錯誤

- LimZ值的設定過低時

  若在第3關節的手臂位置高於LimZ的設定值位置的狀態下執行Jump，則發生錯誤4005。

參照

Accel、Arc、Arch、Go、JS、JT、LimZ、P# = 點指定、Pulse、Sense、Speed、Stat、Till

Jump範例

以下是從點P0到P1進行單純的PTP動作後，以Jump返回P0的範例。在程式下半段，手臂執行Jump。輸入位元4未變為ON時，執行下降動作並移到P1。輸入位元4為ON時，不進行下降動作。

```
Function jumptest
    Home
    Go P0
    Go P1
    Sense Sw(4) = On
    Jump P0 LimZ -10
    Jump P1 LimZ -10 Sense   '檢查輸入4
    If Js(0) = 1 Then
        Print "Input #4 came on during the move and"
        Print "the robot stopped prior to arriving on"
        Print "point P1."
    Else
        Print "The move to P1 completed successfully."
        Print "Input #4 never came on during the move."
    EndIf
Fend


> Jump P10+X50 C0 LimZ-20 Sense !D50;On 0;D80;On 1!
```

以下為命令視窗的操作範例。

```
> Jump P0                              '跳至P0
> Jump P0 C0                           '以Arch編號C0所設定的拱形動作，跳至P0
> Jump P0 LimZ -10                     '以Z極限-10mm為止的設定，跳至P0
> Jump P0  !D0; On 1; D50; Off 1!   '跳至P0。在動作的移動量到達50%前，將輸出的第1個位元設為On；50%以後則將第1個位元設為Off。
```

輸入的第1個位元為On後，停止Jump命令並進行接下來的處理。

```
Function main
    (中間省略)
  Till Sw(1) = On
  Jump P0 C0 CP Till
```

```
    (中間省略)
Fend
```

3.14.23 Jump3, Jump3CP

用於以3維門形動作方式移動手臂。

Jump3為2個CP動作和1個PTP動作的組合。

Jump3CP為3個CP動作的組合。

格式

(1) Jump3 閃避坐標, 接近起始坐標, 目標坐標 [, C Arch編號] [, CP] [, LJM [, 選擇姿態旗標]] [, Sense | Till | Find] [, !平行處理!] [, SYNC]

(2) Jump3CP 閃避坐標, 接近起始坐標, 目標坐標 [, ROT] [, C Arch編號] [, CP] [, LJM [, 選擇姿態旗標]] [, Sense | Till | Find] [, !平行處理!] [, SYNC]

參數

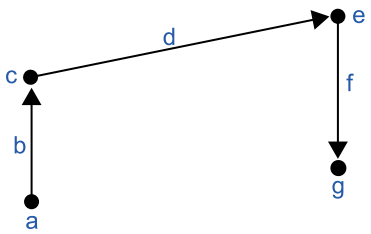
閃避坐標
　以點指定目前位置之上的閃避點。
接近起始坐標
　以點指定目標坐標之上的接近開始點。
目標坐標
　以點指定動作到達的目標坐標。
ROT
　以工具姿態變化為優先，決定動作速度和加減速度。可省略。
Arch編號
　Arch編號用於指定Arch表，以決定Jump3命令的Arch型動作。請務必在Arch編號的開頭處附加大寫「C」。(有效值為C0~C7)可省略Arch編號。
CP
　指定路徑運動。可省略。
LJM
　以LJM函數轉換閃避坐標、接近坐標、目標坐標。可省略。
選擇姿態旗標
　指定賦予LJM函數的姿態旗標選擇參數。可省略。
Sense | Till | Find
　記述Sense、Till、Find中的任一運算式。可省略。

```
Sense | Till | Find
Sense Sw(運算式) = {On | Off}
Till Sw(運算式) = {On | Off}
Find Sw(運算式) = {On | Off}
```

!平行處理!
　可在Jump3、Jump3CP命令中增加平行處理陳述式，以便在動作時執行I/O或其他命令。可省略。
SYNC
　用於預約動作命令。在以SyncRobots開始動作之前，機器人不進行動作。

說明

以3維門形動作方式，將手臂從目前位置移至目標坐標。3維門形動作由閃避動作、Span動作和接近動作構成。從目前位置閃避到閃避坐標的動作是CP動作。對於從閃避坐標到接近起始坐標的Span動作，在Jump3情況下為PTP動作；在Jump3CP情況下為CP動作。從接近起始坐標到目標坐標的接近動作是CP動作。

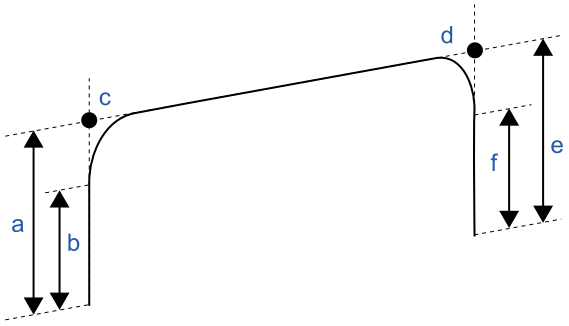


符號	說明
a	目前位置
b	閃避動作 CP
c	閃避坐標
d	Span動作 PTP/CP
e	接近起始坐標
f	接近動作 CP
g	目標坐標

依照Arch編號設定執行Arch動作。

下圖所示為Jump3以及Jump3CP的Arch動作。

請設定大於Arch上升距離的閃避距離以及大於Arch下降距離的接近距離。



符號	說明
a	閃避距離
b	Arch上升距離
c	閃避坐標
d	接近起始坐標
e	接近距離
f	Arch下降距離

Jump3CP的速度和加減速度分別使用SpeedS和AccelS的設定值。關於速度和加減速度的關係，請參閱「注意」項目中的「連同CP一起使用Jump3、Jump3CP」。但是，指定ROT修飾參數時的速度和加減速度分別使用SpeedR和AccelR的設定值。此時，SpeedS和AccelS的設定值即呈停用狀態。

機器人的工具中心點位置固定為特定坐標，僅改變工具姿態時，或者相對於工具中心點的移動距離，工具姿態變化較大時，工具姿態變化速度可能會明顯變快。為防止這種情形，工具姿態變化速度大時，自動限制動作速度的功能就會啟用。

想要手動設定CP動作時的工具姿態變化速度的上限值時，請啟用SpeedRLimitation。啟用SpeedRLimitation時，CP動作時的工具姿態變化速度超過設定的SpeedR時，工具姿態變化速度會變為SpeedR來限制動作速度。工具姿態變化速度超過設定的SpeedR時，以設定的SpeedS進行動作。工具姿態變化速度的上限值，請事先在SpeedR設定。

注意

- LimZ不會影響到Jump3及Jump3CP。

  Span動作不一定垂直於坐標系的Z軸，因此LimZ不會影響到Jump3及Jump3CP。

- Jump3的Span動作為PTP動作。

  由於PTP動作難以預測到其軌跡，因此請充分注意機器人本體和週邊裝置之間的干擾情況。

- 連同CP一起使用Jump3、Jump3CP

  若使用CP參數，則在開始減速的同時，移至下一個陳述式控制動作命令。其便利之處在於，使用者可連接數個動作命令，以恆定的速度執行連續動作。在未指定CP的Jump3命令、Jump3CP命令時，手臂務必減速，並停於指定的目標坐標上。

- Jump3的Pass功能

  若在接近動作量為0的Jump3上附加CP參數，可在不減速並停止該Jump3的Span動作之狀態下繼續順暢連上PTP動作。

  此外，若在此前的PTP動作命令上附加CP參數，閃避動作量為0的Jump3可在不減速並停止該PTP動作之狀態下順暢連上Jump3的Span動作。

  要將平常的Jump3的Span動作(1個PTP動作)替換成順暢連上數個PTP動作時，這個功能非常有用。

- Jump3CP的Pass功能

  若在接近動作量為0的Jump3CP上附加CP參數，可在不減速並停止該Jump3CP的Span動作之狀態下繼續順暢連上CP動作。

  此外，若在此前的CP動作命令上附加CP參數，閃避動作量為0的Jump3CP可在不減速並停止該CP動作之狀態下順暢連上Jump3CP的Span動作。

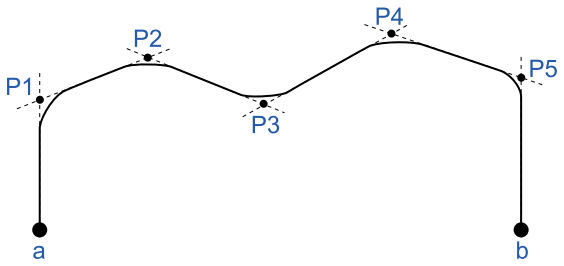
  要將平常的Jump3CP的Span動作(1個CP動作)替換成順暢連上數個CP動作時，這個功能非常有用。

  (例1)

```
Jump3 P1,P2,P2 CP
Go P3,P4 CP
Jump3 P4,P5,P5+tlz(50)
```

  (例2)

```
Jump3CP P1,P2,P2 CP
Move P3,P4 CP
Jump3CP P4,P5,P5+tlz(50)
```



- 連同LJM一起使用Jump3、Jump3CP

  若使用LJM參數，則可簡化使用LJM函數的程式。

  例如：

```
P11 = LJM(P1, Here, 2)
P12 = LJM(P2, P11, 2)
P13 = LJM(P3, P12, 2)
Jump3 P11, P12, P13
```

  可將上述4行的程式取代為

```
Jump3 P1, P2, P3 LJM 2
```

  的1行程式。

  LJM參數對於垂直6軸型機器人(包含N系列)以及RS系列機器人有效。

  Jump3CP的Span動作為直線(CP)動作，因此不可在中途切換腕部姿態。請勿使用有可能切換腕部姿態之LJM函數的選擇姿態旗標(LJM 1)。

使用Arch時的重要事項

由於透過軌跡控制進行拱形動作的合成，因此不能保證實際軌跡。軌跡會因動作速度和手臂的動作方式而異。請以作業時使用的實際速度和姿態確認實際軌跡。

- 在相同位置上，即便執行帶有相同[C Arch編號]的Jump3命令，低速時的軌跡也會比高速動作時低。因此，請注意：即便確認不會因高速而撞擊干擾物，仍有可能在低速動作時發生撞擊。
- 相較於低速動作，高速動作會增加未合成的閃避移動量，呈現出減小未合成的接近移動量之傾向。未出現預期的移動距離時，請降低速度或減速度或將接近距離設得長一些。
- 即便是相同距離的動作，軌跡也會依手臂的動作方式而異。

常見錯誤

- 在閃避動作(接近動作)和Span動作中，主要動作的關節相同時

  以Jump3、Jump3CP命令執行Arch動作時，有可能發生異常加速度錯誤。這在閃避動作(接近動作)和Span動作中，主要動作的關節相同時最為明顯。在這種情況下，請以下列命令降低Span動作的加減速度，避免出現這種狀況：若為Jump3，則用Accel命令；若為Jump3CP，則用AccelS命令。此外，對於部分動作姿態，以AccelS命令降低閃避動作(接近動作)的加減速度，也可能有效。

參照

Accel、Arc、Arch、Go、JS、JT、P# = 點指定、Pulse、Sense、Speed、Stat、Till

Jump3範例

```
'如同SCARA機器人的Jump動作之垂直6軸型機器人(包含N系列)動作
Jump3  Here :Z(100), P3 :Z(100), P3

'使用Z工具坐標的閃避動作和接近動作
Jump3  Here -TLZ(100), P3 -TLZ(100), P3

'使用Z基本坐標的閃避動作和使用Z工具坐標的接近動作
Jump3  Here +Z(100), P3 -TLZ(100), P3
```

以Tool1進行閃避動作，以Tool2進行接近動作的範例

```
Arch 0,20,20
Tool 1
Go P1

P2 = P1 -TLZ(100)
Tool 2
Jump3 P2, P3-TLZ(100), P3 C0
```

3.14.24 JumpTLZ

用於以3維門形動作方式移動手臂。
N系列專用的命令。
JumpTLZ為2個CP動作和1個PTP動作的組合。

格式

JumpTLZ
目標坐標,TLZ方向移動量[, C Arch編號] [, CP] [, LJM [, 選擇姿態旗標]] [, Sense | Till | Find] [, !平行處理!] [, SYNC]

參數

目標坐標
　以點指定動作到達的目標坐標。
TLZ方向移動量
　指定朝Tool坐標的Z方向移動的量。單位是mm。使用目前設定的Tool編號之Tool坐標。
Arch編號
　Arch編號用於指定Arch表，以決定JumpTLZ命令的Arch型動作。請務必在Arch編號的開頭處附加大寫「C」。(有效值為C0~C7 。)可省略Arch編號。
CP
　指定路徑運動。可省略。
LJM
　以LJM函數轉換目標坐標。可省略。
選擇姿態旗標
　指定賦予LJM函數的姿態旗標選擇參數。可省略。
Sense | Till | Find
　記述Sense、Till、Find中的任一運算式。可省略。

```
Sense | Till | Find
Sense Sw(運算式) = {On | Off}
Till Sw(運算式) = {On | Off}
Find Sw(運算式) = {On | Off}
```

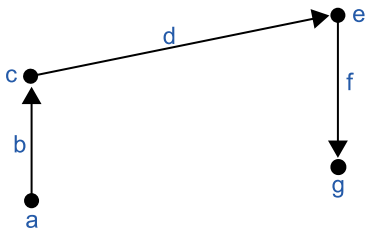
!平行處理!
　可在Jump3、Jump3CP命令中增加平行處理陳述式，以便在動作時執行I/O或其他命令。可省略。
SYNC
　用於預約動作命令。在以SyncRobots開始動作之前，機器人不進行動作。

說明

以3維門形動作方式，將手臂從目前位置移至目標坐標。3維門形動作由閃避動作、Span動作和接近動作構成。從目前位置閃避到閃避坐標的動作是CP動作。從閃避坐標到接近起始坐標的Span動作是PTP動作。

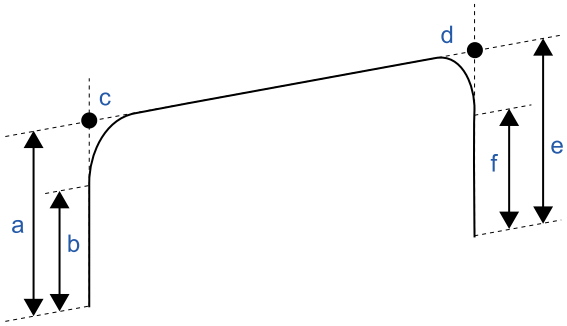
閃避坐標為按TLZ方向移動量從目前位置朝Tool坐標的Z方向移動的位置。閃避坐標的機器人姿態與目前位置相同。(唯有通過特定點或特定點附近時，機器人姿態才有可能不同。)

接近起始坐標是按到目標坐標為止的移動量從閃避坐標朝Tool坐標的X、Y方向移動的位置。閃避坐標的U、V、W坐標和機器人的姿態與目標位置相同。(唯有通過特定點或特定點附近時，機器人姿態才有可能不同。)



符號	說明
a	目前位置
b	閃避動作 CP
c	閃避坐標
d	Span動作 PTP
e	接近起始坐標
f	接近動作 CP
g	目標坐標

依照Arch編號設定執行Arch動作。請設定大於Arch上升距離的閃避距離以及大於Arch下降距離的接近距離。



符號	說明
a	閃避距離
b	Arch上升距離
c	閃避坐標
d	接近起始坐標
e	接近距離
f	Arch下降距離

注意

- LimZ不會影響到JumpTLZ。

  Span動作不一定垂直於坐標系的Z軸，因此LimZ不會影響到JumpTLZ。

- JumpTLZ的Span動作為PTP動作。

  由於PTP動作難以預測到其軌跡，因此請充分注意機器人本體和週邊裝置之間的干擾情況。

- JumpTLZ和Jump3的差異

  如下所述為JumpTLZ和Jump3的差異點。

  - JumpTLZ:
    - 閃避坐標不可設為從目前位置朝Tool坐標的Z方向移動的位置之外的地方。
    - 接近坐標不可從目標坐標移至Tool坐標Z方向以外的地方。
    - 此外，無法指定接近距離。

- 無法在閃避坐標、接近坐標、目標位置選擇不同的Tool坐標。(不可以Tool1進行閃避動作，或者以Tool2進行接近動作)

- Jump3:
  - 可任意指定閃避坐標的位置。
  - 可任意指定接近坐標的位置。
  - 可在閃避坐標、接近坐標、目標位置選擇不同的Tool坐標。(可以Tool1進行閃避動作，以Tool2進行接近動作)

- 可使用JumpTLZ的機種

  只可使用N系列。

使用Arch時的重要事項

由於透過軌跡控制進行拱形動作的合成，因此不能保證實際軌跡。軌跡會因動作速度和手臂的動作方式而異。請以作業時使用的實際速度和姿態確認實際軌跡。

- 在相同位置上，即便執行帶有相同[C Arch編號]的JumpTLZ命令，低速時的軌跡也會比高速動作時低。因此，請注意：即便確認不會因高速而撞擊干擾物，仍有可能在低速動作時發生撞擊。
- 相較於低速動作，高速動作會增加未合成的閃避移動量，呈現出減小未合成的接近移動量之傾向。未出現預期的移動距離時，請降低速度或減速度或將接近距離設得長一些。
- 即便是相同距離的動作，軌跡也會依手臂的動作方式而異。

常見錯誤

- 在閃避動作(接近動作)和Span動作中，主要動作的關節相同時

  以JumpTLZ命令執行Arch動作時，有可能發生異常加速度錯誤。這在閃避動作(接近動作)和Span動作中，主要動作的關節相同時最為明顯。在這種情況下，請以Accel命令降低Span動作的加減速度，避免出現這種狀況。此外，對於部分動作姿態，以AccelS命令降低閃避動作(接近動作)的加減速度，也可能有效。

參照

Accel、Arc、Arch、Go、JS、JT、P# = 點指定、Pulse、Sense、Speed、Stat、Till

JumpTLZ範例

從目前位置朝Tool坐標的Z方向上升100 mm，然後移至目的地(P0)時：

```
JumpTLZ  P0, -100
```

3.15 L

3.15.1 LatchEnable

用於啟用/停用以R-I/O輸入對機器人位置進行閂鎖的功能。

格式

LatchEnable { On | Off }

參數

On | Off

On：啟用機器人位置的閂鎖功能。Off：停用機器人位置的閂鎖功能。

結果

若有省略參數，則顯示目前閂鎖功能的啟用/停用。

說明

啟用/停用透過連接於R-I/O的觸發輸入信號對機器人位置進行閂鎖的功能。啟用閂鎖功能後，透過最初的觸發輸入，機器人的位置會被閂鎖。若要重複對機器人位置進行閂鎖，首先透過LatchEnable Off解除閂鎖功能，然後重新執行LatchEnable On。要重複使用時，若考量各命令的處理時間，則需要有約60 ms以上的時間間隔。可忽略LatchEnable本身的執行時間。

注意

在啟用閂鎖功能前，請用SetLatch設定觸發輸入連接埠和觸發信號邏輯。

參照

LatchPos函數、LatchState函數、SetLatch

LatchEnable範例

```
Function main
    SetLatch 24, SETLATCH_TRIGGERMODE_LEADINGEDGE
    LatchEnable On    '啟用閂鎖功能
    Go P1
    Wait LatchState = True     '等待觸發
    Print LatchPos     '顯示閂鎖位置
    LatchEnable Off     '停用閂鎖功能
Fend
```

3.15.2 LatchState函數

本函數用於傳回以R-I/O對機器人位置進行閂鎖的狀態。

格式

LatchState

傳回值

完成機器人位置的閂鎖時，則傳回「True」；未完成時，則傳回「False」。

確認完成閂鎖後，以LatchPos函數取得閂鎖位置資訊。

如果在SetLatch中指定了連續閂鎖計數，則在完成指定次數的所有閂鎖後，將返回「True」。

參照

LatchEnable、LatchPos函數、SetLatch、Wait

LatchState函數範例

```
Function main
    SetLatch 24, SETLATCH_TRIGGERMODE_LEADINGEDGE
    LatchEnable On    '啟用閂鎖功能
    Go P1
    Wait LatchState = True      '等待觸發
    Print LatchPos      '顯示閂鎖位置
    LatchEnable Off      '停用閂鎖功能
Fend
```

3.15.3 LatchPos函數

本函數用於傳回以R-I/O輸入信號進行閂鎖的機器人位置。

格式

LatchPos ([WithToolArm | WithoutToolArm], 閂鎖編號)

參數

WithToolArm | WithoutToolArm

指定呼叫函數時，傳回值為基於Tool、Arm設定的位置，或為Tool 0、Arm 0的位置。參數雖可省略，但指定閂鎖編號時，則請勿省略。若省略，則設定WithToolArm。

- WithToolArm: 0
- WithoutToolArm: 1

WithToolArm

值0的常數。用於傳回基於叫用函數時的Tool、Arm設定之位置。

WithoutToolArm

值1的常數。不管Tool、Arm設定如何，均傳回Tool 0、Arm 0的位置。

閂鎖編號

指定於LatchEnable On後，需傳回以第幾次的R-I/O輸入信號進行閂鎖的點數據。可指定為1、2、3、4。若以SetLatch指定連續閂鎖的次數，於LatchEnable On後，最多可執行4次以R-I/O輸入信號對點資料進行閂鎖。可省略參數。省略時則傳回以第1次的R-I/O輸入信號進行閂鎖的點資料。

傳回值

用於以點資料傳回使用R-I/O輸入信號進行閂鎖的機器人位置。

執行LatchPos函數大約需花15 msec的處理時間。

參數為WithToolArm時，傳回基於叫用函數時的Tool、Arm設定之位置。

參數為WithoutToolArm時，不管Tool、Arm設定如何，均傳回Tool 0、Arm 0的位置。

參照

LatchEnable、LatchState函數、SetLatch

LatchPos函數範例

```
Function main
    SetLatch 24, SETLATCH_TRIGGERMODE_LEADINGEDGE, 4
    LatchEnable On    '啟用閂鎖功能
    Go P1
    Wait LatchState = True     '等待觸發
    Print LatchPos(WithoutToolArm, 1)     '顯示閂鎖位置1
    Print LatchPos(WithoutToolArm, 2)     '顯示閂鎖位置2
    Print LatchPos(WithoutToolArm, 3)     '顯示閂鎖位置3
    Print LatchPos(WithoutToolArm, 4)     '顯示閂鎖位置4
    LatchEnable Off     '停用閂鎖功能
Fend
```

省略參數時的範例

```
Function main
    SetLatch 24, SETLATCH_TRIGGERMODE_LEADINGEDGE
    LatchEnable On    '啟用閂鎖功能
    Go P1
    Wait LatchState = True     '等待觸發
    Print LatchPos     '顯示閂鎖位置1
```

```
    LatchEnable Off      '停用閂鎖功能
Fend
```

將LatchPos的傳回值帶入點數據中的範例

```
P2 = LatchPos     '代入閂鎖位置1。省略參數

P2 = LatchPos(WithoutToolArm, 1)     '代入閂鎖位置1
P3 = LatchPos(WithoutToolArm, 2)      '代入閂鎖位置2
P4 = LatchPos(WithoutToolArm, 3)      '代入閂鎖位置3
P5 = LatchPos(WithoutToolArm, 4)      '代入閂鎖位置4
```

3.15.4 LCase$函數

用於以小寫格式傳回字串。

格式

LCase$ (字串)

參數

字串

指定要小寫的字串。

傳回值

用於傳回轉換為小寫的字串。

參照

LTrim$、Trim$、RTrim$、UCase$

LCase$函數範例

```
str$ = "Data"
str$ = LCase$(str$)  ' str$ = "data"
```

3.15.5 Left$函數

本函數用於從字串的左側取出指定字串。

格式

Left$ (字串, 字元數)

參數

字串
　　從左側複製指定字串的原字串。
字元數
　　直接以數值指定要從字串左側複製的字元數。

傳回值

用於從指定字串的左側取得指定字元數並進行傳回。

說明

Left$用於從指定字串的左側，按字元數取出字元並進行傳回。Left$可用於按指定字串內的字元數傳回字元。

參照

Asc、Chr$、InStr、Len、Mid$、Right$、Space$、Str$、Val

Left$函數範例

以下是用於分析字串的程式範例。

```
Function ParsePartData(DataIn$ As String, ByRef PartNum$ As String, ByRef PartName$
As String, ByRef PartCount As Integer)

    Integer pos
    String temp$

    pos = Instr(DataIn$, ",")
    PartNum$ = Left$(DataIn$, pos - 1)

    DataIn$ = Right$(datain$, Len(DataIn$) - pos)
    pos = Instr(DataIn$, ",")

    PartName$ = Left$(DataIn$, pos - 1)

    PartCount = Val(Right$(datain$, Len(DataIn$) - pos))

Fend
```

這是在命令視窗中使用Left$命令的範例。

```
> Print Left$("ABCDEFG", 2)
 AB

> Print Left$("ABC", 3)
 ABC
```

3.15.6 Len函數

用於傳回字串的字元數。

格式
Len (字串)

參數

字串
　　指定字串運算式。

傳回值
用於以整數值傳回作為Len命令的引數而賦予的字串字元數。

說明
Len用於以整數值(0~255)傳回指定字串的字元數。(字串的字數限制範圍為0~255。)

參照
Asc、Chr$、InStr、Left$、Mid$、Right$、Space$、Str$、Val

Len函數範例
以下是用於分析字串的程式範例。

```
Function ParsePartData(DataIn$ As String, ByRef PartNum$ As String, ByRef PartName$
As String, ByRef PartCount As Integer)

    Integer pos
    String temp$

    pos = Instr(DataIn$, ",")
    PartNum$ = Left$(DataIn$, pos - 1)

    DataIn$ = Right$(datain$, Len(DataIn$) - pos)
    pos = Instr(DataIn$, ",")

    PartName$ = Left$(DataIn$, pos - 1)

    PartCount = Val(Right$(datain$, Len(DataIn$) - pos))

Fend
```

這是其它在命令視窗中使用Len命令的範例。

```
> ? len("ABCDEFG")
7

> ? len("ABC")
3

> ? len("")
0
>
```

3.15.7 LimitTorque

設定並顯示高功率模式時的扭矩上限值。

格式

(1) LimitTorque 所有關節高功率扭矩上限值

(2) LimitTorque 第1關節高功率扭矩上限值、第2關節高功率扭矩上限值、第3關節高功率扭矩上限值、第4關節高功率扭矩上限值

(3) LimitTorque 第1關節高功率扭矩上限值、第2關節高功率扭矩上限值、第3關節高功率扭矩上限值、第4關節高功率扭矩上限值、第5關節高功率扭矩上限值、第6關節高功率扭矩上限值

(4) LimitTorque

參數

所有關節高功率扭矩上限值
　　以表示對於各關節瞬時最大扭矩的比例之整數值指定所有關節的高功率扭矩上限值。
第n關節高功率扭矩上限值
　　以表示對於第n關節瞬時最大扭矩的比例之整數值指定第n關節的高功率扭矩上限值。

結果

若省略參數，則顯示目前LimitTorque值。

說明

限制高功率模式時的扭矩上限值。通常被設為最大扭矩，不需變更本設定值。但是，為了減少因週邊設備間的干擾造成的機器人或裝置的損壞，而欲限制發生超過特定動作所需扭矩以上的扭矩時，這還是有效的。上限值是以PTRQ測量特定動作的峰值扭矩，並在該值上加上考慮到波動部分(約10%)的寬裕值。

無法以本命令設定低於低功率模式時的扭矩上限值之高功率扭矩上限值。可對各機種、關節設定的最小值並不相同。請於設定後顯示設定值，並確認實際設定的上限值。

在以下情況下，LimitTorque會恢復為預設值。

- 啟動控制器時
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

注意

- 過低的LimitTorque設定

  無論以設定的加減速度進行特定動作所需的扭矩大小如何，LimitTorque皆可將設定的扭矩限制值作為上限值，對扭矩施加限制。這樣的話，特定動作如需大於設定上限值的扭矩，機器人可能無法進行適當的動作，這會造成動作振動，發出異音，或發生位置偏差錯誤及超出。請務必測量PTRQ，然後使用扭矩限制功能。萬一發生上述狀態，表示扭矩不足。此時，請增大扭矩上限值，調整為可正常動作的值。

參照

LimitTorque函數、Power、PTrq、RealTorque

LimitTorque範例

這是將第1關節的最大扭矩限制為80%進行動作的範例。

```
Function main
    Motor On
    Power high
    Speed 100; Accel 100,100
    LimitTorque 80,100,100,100    '將第1關節的最大扭矩限制為80%
    Jump P1                       '執行Jump動作
Fend
```

3.15.8 LimitTorque函數

用於傳回LimitTorque命令的設定值。

格式

LimitTorque (關節編號)

參數

關節編號

以1~9的整數進行指定。附加軸的S軸為8，T軸為9。

傳回值

用於以整數值傳回LimitTorque命令的設定值。

參照

LimitTorque

LimitTorque函數範例

```
Print LimitTorque(1) '顯示第1關節的LimitTorque值
```

3.15.9 LimitTorqueLP

設定並顯示低功率模式時的扭矩上限值。

格式

(1) LimitTorqueLP 所有關節低功率扭矩上限值

(2) LimitTorqueLP 第1關節低功率扭矩上限值, 第2關節低功率扭矩上限值, 第3關節低功率扭矩上限值, 第4關節低功率扭矩上限值

(3) LimitTorqueLP 第1關節低功率扭矩上限值, 第2關節低功率扭矩上限值, 第3關節低功率扭矩上限值, 第4關節低功率扭矩上限值, 第5關節低功率扭矩上限值, 第6關節低功率扭矩上限值

(4) LimitTorqueLP

參數

所有關節低功率扭矩上限值
　　以表示對於各關節瞬時最大扭矩的比例之整數值指定所有關節的低功率扭矩上限值。
第n關節低功率扭矩上限值
　　以表示對於第n關節瞬時最大扭矩的比例之整數值指定第n關節的低功率扭矩上限值。

結果

若省略參數，則顯示目前LimitTorqueLP值。

未以本命令變更值時，則顯示預設值。

說明

限制低功率模式時的扭矩上限值。預設情況下，已將低功率動作所需的扭矩作為上限值進行了設定(數值因機種或軸而異。15~60%左右)。雖不需變更本設定值，但是，在為了減少諸如因與週邊設備的衝撞造成的機器人或裝置的損壞，而欲限制發生遠遠超過無衝撞的正常動作所需之扭矩的扭矩時，這還是有效的。上限值是以PTRQ測量正常動作時的峰值扭矩，並在該值上加上考慮到波動部分(建議為40%)的邊界值。若要將相同值適用於不同機器人，請再加上10~20%的邊界。

將低功率時的預設最大扭矩作為1.0來顯示PTRQ的值。例如，變更前的預設值為27%，PTRQ的測量值為0.43時，則變為27% × 0.43 × 1.4 = 16.25，因此，取整數的話設為17。

無法在本命令中設定低於5%的值或高於預設值的值。此時，低於5的設定值皆歸為5，超過預設值的設定值則歸為預設值。例如，若設為「LimitTorqueLP 100」，預設值必定小於100，因此，所有軸均恢復為預設值。請於設定後顯示設定值，並確認實際設定的上限值。

在重新啟動控制器之前，LimitTorqueLP的設定值呈啟用狀態。

注意

- 過低的LimitTorqueLP設定

  無論以設定的加減速度進行特定動作所需的扭矩大小如何，LimitTorqueLP皆可將設定的扭矩限制值作為上限值，對扭矩施加限制。這樣的話，特定動作如需大於設定上限值的扭矩，機器人可能無法進行適當的動作，這會造成位置偏差錯誤。請務必在低功率狀態下測量PTRQ，然後使用扭矩限制功能。萬一發生上述狀態，表示扭矩不足。此時，請增大扭矩上限值，調整為可正常動作的值。

參照

LimitTorqueLP函數、PTRQ

LimitTorqueLP範例

這是將第1關節的最大扭矩限制為10%進行動作的範例。

```
Function main
Motor On
Power low
LimitTorqueLP 10,27,31,42     '將第1關節的最大扭矩限制為10%
'其它軸則設定預設值
Go P1     '執行Go動作
Fend
```

3.15.10 LimitTorqueLP函數

用於傳回LimitTorqueLP命令的設定值。

格式

LimitTorqueLP (關節編號)

參數

關節編號

以1~9的整數進行指定。附加軸的S軸為8，T軸為9。

傳回值

用於以整數值傳回LimitTorqueLP命令的設定值。

參照

LimitTorqueLP

LimitTorqueLP函數範例

```
Print LimitTorqueLP(1) '顯示第1關節的LimitTorqueLP值
```

3.15.11 LimitTorqueStop

若在高功率模式下達到扭矩上限，則啟用或停用停止機器人的功能。

格式

(1) LimitTorqueStop 狀態

(2) LimitTorqueStop 狀態, 關節編號

(3) LimitTorqueStop

參數

狀態

- On：啟用扭矩上限的停止功能。
- Off：停用扭矩上限的停止功能。

關節編號

指定1~6的關節編號。(若為SCARA機器人，關節編號為1~4)

結果

若省略參數，則顯示目前LimitTorqueStop狀態。

說明

啟用高功率動作時的扭矩上限停止功能。達到扭矩上限(預設值為100%)時，馬上停止機器人。可透過與LimitTorque的扭矩限制功能併用，獲得降低因高功率模式時的衝撞或接觸導致機器人或裝置損壞的效果。

可設定所有軸的ON/OFF以及各軸的ON/OFF。預設值為所有軸OFF。

雖然啟動控制器時會返回預設值，但除此之外，除非以本命令進行明確設定，否則不會改變狀態。

到達扭矩上限時，會輸出錯誤5040「馬達扭矩輸出在高功率狀態下出現異常」的訊息，停止機器人動作。

參照

LimitTorque、LimitTorque函數

LimitTorqueStop範例

這是將第1關節的最大扭矩限制為30%並馬上停止的範例。

```
Function main
Motor On
Power high
Speed 20
Accel 20,20
LimitTorque 30,100,100,100     '將第1關節的最大扭矩限制為30%
LimitTorqueStop On, 1     '第1關節在最大扭矩時馬上停止
Go P1     '執行Go動作
Fend
```

3.15.12 LimitTorqueStop函數

用於傳回LimitTorqueStop命令的設定值。

格式

LimitTorqueStop (關節編號)

參數

關節編號
以1~6的整數進行指定。

傳回值

用於以整數值傳回LimitTorqueStop命令的設定值。

- 0 = OFF
- 1 = ON

參照

LimitTorqueStop

LimitTorqueStop函數範例

```
Print LimitTorqueStop(1) '顯示第1關節的LimitTorqueStop值
```

3.15.13 LimitTorqueStopLP

若在低功率模式下達到扭矩上限，則啟用或停用停止機器人的功能。

格式

(1) LimitTorqueStopLP 狀態

(2) LimitTorqueStopLP 狀態, 關節編號

(3) LimitTorqueStopLP

參數

狀態

- On：啟用扭矩上限的停止功能。
- Off：停用扭矩上限的停止功能。

關節編號

指定1~6的關節編號。(若為SCARA機器人，關節編號為1~4)

結果

若省略參數，則顯示目前LimitTorqueStopLP狀態。

說明

啟用低功率動作時的扭矩上限停止功能。達到扭矩上限時，馬上停止機器人。可透過與LimitTorqueLP的扭矩限制功能併用，獲得降低因低功率模式時的衝撞或接觸導致機器人或裝置損壞的效果。

可設定所有軸的ON/OFF以及各軸的ON/OFF。預設值為所有軸OFF。

雖然啟動控制器時會返回預設值，但除此之外，除非以本命令進行明確設定，否則不會改變狀態。

到達扭矩上限時，會輸出錯誤5041「馬達扭矩輸出在低功率狀態下出現異常」的訊息，停止機器人動作。

參照

LimitTorqueLP、LimitTorqueLP函數

LimitTorqueStopLP範例

這是將第3關節的最大扭矩限制為15%並馬上停止的範例。

```
Function main
Motor On
Power low
LimitTorqueLP 20,27,15,42     '將第3關節的最大扭矩限制為15% '其它軸則設定預設值
LimitTorqueStopLP On, 3     '第3關節在最大扭矩時馬上停止
Go P1     '執行Go動作
Fend
```

3.15.14 LimitTorqueStopLP函數

用於傳回LimitTorqueStopLP命令的設定值。

格式

LimitTorqueStopLP (關節編號)

參數

關節編號
以1~6的整數進行指定。

傳回值

用於以整數值傳回LimitTorqueStopLP命令的設定值。

- 0 = OFF
- 1 = ON

參照

LimitTorqueStopLP

LimitTorqueStopLP函數範例

```
Print LimitTorqueStopLP(3)  '顯示第3關節的LimitTorqueStopLP值
```

3.15.15 LimZ

設定Jump命令時的第3關節高度(Z坐標值)初始值。

格式

(1) LimZ Z坐標值

(2) LimZ

參數

Z坐標值

指定第3關節動作範圍內的坐標值。

結果

若省略參數，則顯示目前LimZ值。

說明

執行Jump命令時，機器人手臂會朝第3關節(Z軸)方向上升，接著在X-Y平面上移動，最後朝第3關節(Z軸)方向下降。此時，LimZ用於設定手臂朝第3關節(Z軸)方向進行動作的高度上限。LimZ用於設定Jump命令時的第3關節動作範圍之最高坐標預設值。執行Jump命令時，若未設定特定的LimZ值，則使用最後設定的LimZ值。

注意

- 將LimZ值重設為0

  LimZ值會因控制器的重新啟動、SFree/SLock/Motor On等命令而被初始化為0。

- LimZ值不可用於Arm、Tool或Local坐標

  LimZ的高度限制值為機器人坐標的Z坐標值。並非Arm、Tool或Local坐標的Z坐標值。因此，使用高度不同的抓手或手臂尖端工具時，請予以注意。

- LimZ不會影響到Jump3及Jump3CP。

  Span動作不一定垂直於坐標系的Z軸，因此LimZ不會影響到Jump3及Jump3CP。

參照

Jump

LimZ範例

這是操作Jump時使用LimZ的範例。

```
Function main
    LimZ -10             '設定LimZ的預設值
    Jump P1              '執行Jump時，以-10進行水平移動
    Jump P2 LimZ -20     '執行Jump時，以-20進行水平移動
    Jump P3              '執行Jump時，以-10進行水平移動
Fend
```

3.15.16 LimZ函數

用於傳回LimZ命令的設定值。

格式

LimZ

傳回值

用於以實數值傳回LimZ命令的設定值。

參照

LimZ

LimZ函數範例

```
Real savLimz

savLimz = LimZ
LimZ -25
Go pick
LimZ savLimZ
```

3.15.17 LimZMargin

用於設定在高於LimZ設定值的位置開始動作時的錯誤檢測邊界值，並傳回設定值。

格式

(1) LimZMargin LimZ邊界

(2) LimZMargin

參數

LimZ邊界

指定LimZ錯誤檢測的邊界值。

結果

若省略參數，則顯示目前LimZMargin值。

說明

執行Jump命令時，第3關節(Z軸)會上升到以LimZ設定的高度，但開始執行Jump動作時，若在高於LimZ位置的位置上有第3關節(Z軸)，則發生錯誤。LimZMargin用於對此錯誤檢測設定邊界值。預設值為0.02 mm。

注意

- 將LimZMargin值重設為預設值

  LimZMargin值會因控制器的重新啟動、SFree/SLock/Motor On等命令而被初始化為預設值。

參照

LimZMargin函數、LimZ

LimZMargin範例

這是操作Jump時使用LimZMargin的範例。

```
Function main
    LimZ -10              '設定LimZ的預設值
    LimZMargin 0.03       '將LimZ的錯誤檢測邊界設為0.03 mm
    Jump P1               '執行Jump時，以-10進行水平移動
    Jump P2 LimZ -20      '執行Jump時，以-20進行水平移動
    Jump P3               '執行Jump時，以-10進行水平移動
Fend
```

3.15.18 LimZMargin函數

用於傳回LimZMargin命令的設定值。

格式

LimZMargin

傳回值

用於以實數值傳回LimZMargin命令的設定值。

參照

LimZMargin、LimZ

LimZMargin函數範例

```
Real savLimzMargin

savLimzZMargin = LimZMargin
LimZMargin 0.03
Jump pick
LimZ savLimZMargin
```

3.15.19 Line Input

載入1行輸入資料，並將該資料指派給字串變數。

格式

Line Input 字串變數名稱$

參數

字串變數名稱$

指定字串變數名稱。(請在字串變數最後附加$。)

說明

Line Input用於載入來自顯示裝置的1行輸入資料，並指派給Line Input命令的字串變數。Line Input命令時，若處於可從使用者接收資料的狀態，顯示裝置上則顯示提示「?」。此提示之後輸入的資料行作為字串的值被指派。輸入資料後，請按一下[ENTER]鍵。

參照

Input、Input #、Line Input#、ParseStr

Line Input範例

這是Line Input的範例。

```
Function Main
    String A$
    Line Input A$  '載入1行輸入資料並指派給A$
    Print A$
Fend
```

若執行上述程式，則執行以下階段。

```
?A, B, C
A, B, C
```

3.15.20 Line Input

從檔案、通訊連接埠、資料庫、裝置載入1行資料。

格式

Line Input #連接埠編號, 字串變數名稱$

參數

連接埠編號

表示檔案、通訊連接埠、資料庫、設備的ID編號。檔案編號是以ROpen、WOpen、AOpen等陳述式指定的編號。通訊連接埠編號是以OpenCom(RS-232C)或OpenNet(TCP/IP)陳述式指定的編號。資料庫編號是以OpenDB陳述式指定的編號。

裝置ID為以下數值。

- 21 RC+
- 20 TP4

字串變數名稱$

指定字串變數名稱。(請在字串變數最後附加$。)

說明

Line Input #用於從以連接埠編號指定的裝置載入1行資料，並指派給以字串變數名稱$指定的變數。

注意

- 支援的控制器型號

  如果在T/VT系列中指定RS-232C埠時會發生錯誤。

參照

Input、Input #、Line Input

Line Input #範例

在以下範例中，從通訊連接埠1接收字串資料，並指派給字串變數A$。

```
Function lintest
    String a$
    Print #1, "Please input string to be sent to robot"
    Line Input #1, a$
    Print "Value entered = ", a$
Fend
```

3.15.21 LJM函數

用於傳回對姿態旗標進行轉換的點資料，以確保最小的關節移動量(相對於指定點，從參照點來看的情況下)。

格式

LJM (點指定 [, 參照點指定 [, 選擇姿態旗標]])

參數

點指定

指定作為對象的點數據。

參照點指定

指定作為基準的點數據。若省略參照點指定，則將目前位置(Here)作為參照點。

選擇姿態旗標

垂直6軸型

- 1：以腕部姿態(Wrist旗標)、J4Flag、J6Flag以及J1Flag進行轉換，以便使J4軸形成最短移動量。是省略選擇姿態旗標時的預設值。
- 2：以J4Flag以及J6Flag進行轉換。
- 3：以腕部姿態(Wrist旗標)、J4Flag、J6Flag以及J1Flag進行轉換，以便使J5軸形成最短移動量。
- 4：以腕部姿態(Wrist旗標)、J4Flag、J6Flag以及J1Flag進行轉換，以便使J6軸形成最短移動量。

「選擇姿態旗標」	手腕姿態	臂肘姿態	腕部姿態	J1Flag	J4Flag	J6Flag	移動量最短的軸之優先順序
1	-	-	○	○	○	○	J4
2	-	-	-	○	○	○	-
3	-	-	○	○	○	○	J5
4	-	-	○	○	○	○	J6

Note：「-」的姿態與以「參照點指定」指定的姿態相同。

RS系列

- 1：以手腕姿態(Hand旗標)、J1Flag以及J2Flag進行轉換。是省略「選擇姿態旗標」時的預設值。
- 2：以手腕姿態(Hand旗標)、J1Flag以及J2Flag進行轉換。用於防止在轉換「選擇姿態旗標」時發生U軸不在動作範圍內的錯誤。

N2系列

- 1：依照J1、J5軸的優先順序，轉換為減小關節移動量的姿態。作為轉換對象的姿態為手腕姿態(Hand旗標)、臂肘姿態(Elbow旗標)、腕部姿態(Wrist旗標)、J4Flag以及J6Flag。臂肘姿態(Elbow旗標)務必為上肘姿態。是省略「選擇姿態旗標」時的預設值。
- 2：依照J1、J4軸的優先順序，轉換為減小關節移動量的姿態。作為轉換對象的姿態為手腕姿態(Hand旗標)、臂肘姿態(Elbow旗標)、腕部姿態(Wrist旗標)、J4Flag以及J6Flag。臂肘姿態(Elbow旗標)務必為上肘姿態。
- 3：以腕部姿態(Wrist旗標)、J4Flag以及J6Flag進行轉換，以便使J4軸形成最短移動量。
- 4：以J4Flag以及J6Flag進行轉換。
- 5：變更為與以「參照點指定」指定之姿態不同的手腕姿態(Hand旗標)，並以腕部姿態(Wrist旗標)、J4Flag以及J6Flag進行轉換，以便使J5軸形成最短移動量。作為轉換對象的姿態為手腕姿態(Hand旗標)、臂肘姿態(Elbow旗標)、腕部姿態(Wrist旗標)、J4Flag以及J6Flag。此外，臂肘姿態(Elbow旗標)務必為上肘姿態。
- 6：變更為與以「參照點指定」指定之姿態不同的手腕姿態(Hand旗標)，並以腕部姿態(Wrist旗標)、J4Flag以及J6Flag進行轉換，以便使J4軸形成最短移動量。作為轉換對象的姿態為手腕姿態(Hand旗標)、

臂肘姿態(Elbow旗標)、腕部姿態(Wrist旗標)、J4Flag以及J6Flag。此外，臂肘姿態(Elbow旗標)務必為上肘姿態。

- 7：將臂肘姿態(Elbow旗標)設為下肘姿態，並以腕部姿態(Wrist旗標)、J4Flag以及J6Flag進行轉換，以便依照J1、J5軸的優先順序形成最短移動量。作為轉換對象的姿態為手腕姿態(Hand旗標)、臂肘姿態(Elbow旗標)、腕部姿態(Wrist旗標)、J4Flag以及J6Flag。
- 8：將臂肘姿態(Elbow旗標)設為下肘姿態，並以腕部姿態(Wrist旗標)、J4Flag以及J6Flag進行轉換，以便依照J1、J4軸的優先順序形成最短移動量。作為轉換對象的姿態為手腕姿態(Hand旗標)、臂肘姿態(Elbow旗標)、腕部姿態(Wrist旗標)、J4Flag以及J6Flag。

「選擇姿態旗標」	手腕姿態	臂肘姿態	腕部姿態	J4Flag	J6Flag	移動量最短的軸之優先順序
1	○	*1	○	○	○	J1>J5
2	○	*1	○	○	○	J1>J4
3	-	-	○	○	○	J4
4	-	-	-	○	○	-
5	*2	*1	○	○	○	J5
6	*2	*1	○	○	○	J4
7	○	*3	○	○	○	J1>J5
8	○	*3	○	○	○	J1>J4

Note：「-」的姿態與以「參照點指定」指定的姿態相同。

- *1：上肘姿態
- *2：以「參照點指定」指定的姿態和手腕姿態並不相同。
- *3：下肘姿態

N6系列

- 1：以腕部姿態(Wrist旗標)、J4Flag以及J6Flag進行轉換，以便使J4軸形成最短移動量。"是省略「選擇姿態旗標」時的預設值。
- 2：以J4Flag以及J6Flag進行轉換。
- 3：以腕部姿態(Wrist旗標)、J4Flag以及J6Flag進行轉換，以便使J5軸形成最短移動量。
- 4：以腕部姿態(Wrist旗標)、J4Flag以及J6Flag進行轉換，以便使J6軸形成最短移動量。

「選擇姿態旗標」	手腕姿態	臂肘姿態	腕部姿態	J1Flag	J4Flag	J6Flag	移動量最短的軸之優先順序
1	-	-	○	○	○	○	J4
2	-	-	-	○	○	○	-
3	-	-	○	○	○	○	J5
4	-	-	○	○	○	○	J6

Note：「-」的姿態與以「點指定」指定的姿態相同。

說明

垂直6軸型機器人或N系列在向透過棧板或相對位移等點運算獲取的點進行動作時，腕部有可能朝非預期的方向轉動。其原因在於，上述點運算屬於不依賴於機器人機種的命令，因此在未進行所需姿態旗標的轉換之狀態下進行動作。

為了防止發生這種腕部的非預期轉動，LJM函數用於適當轉換點資料的姿態旗標。

此外，N系列還可透過變更手腕姿態旗標或臂肘姿態旗標，縮短生產節拍時間或省去教導垂直6軸機器人所需的迴避點。

對於RS系列，在向透過棧板或相對位移等點運算獲取的點進行動作時，第1手臂有可能朝非預期的方向轉動。為了防止發生這種第1手臂的非預期轉動，LJM函數用於適當轉換點資料的姿態旗標。

此外，對於RS系列，若轉換姿態旗標，可能會發生U軸向動作範圍外進行動作的錯誤。為了防止發生這種U軸的動作範圍外錯誤，LJM函數用於將U軸的目標角度補償為動作範圍內的目標角度。將選擇姿態旗標設為2，則可使用此功能。

對於垂直6軸型、N系列、RS系列以外的機器人，直接傳回指定點。

注意

- 參照點的省略和平行處理

  1個動作命令中不可同時存在參照點的省略和平行處理。

  ```
  Go LJM(P10)  ! D10; MemOn 1 !
  ```

  不接受上述用法。

  ```
  P999 = Here
  Go LJM(P10,P999) ! D10; MemOn 1 !
  ```

  請變更為如上程式。

- 關於N2系列的選擇姿態旗標

  - 選擇姿態旗標1、2：

    欲縮短機器人的生產節拍時間時，請選擇姿態旗標1或2。

    由於採取第1關節移動量最小的姿態，因此幾乎所有動作都可以說是生產節拍時間最短的動作。若要執行減小第5關節移動量的動作，請選擇姿態選擇旗標1；若要執行減小第4關節移動量的動作，請選擇姿態選擇旗標2。

  - 選擇姿態旗標3、4：

    若要和垂直6軸型一樣使用，請進行選擇。

    姿態旗標3同於垂直6軸型的姿態旗標1。

    姿態旗標4同於垂直6軸型的姿態旗標2。

  - 選擇姿態旗標5、6：

    在機器人動作時，若抓手會接觸機器人週邊的牆壁等，請選擇姿態旗標5或6。抓手會通過機器人的原點附近，因此機器人可在不易接觸到周圍障礙物的狀態下進行動作。若要執行減小第5關節移動量的動作，請選擇姿態選擇旗標5；若要執行減小第4關節移動量的動作，請選擇姿態選擇旗標6。

  - 選擇姿態旗標7、8：

    若要採取下肘姿態，請選擇姿態旗標7或8。有時機器人會進行通過原點附近的動作(如姿態選擇旗標5、6)，因此當機器人周圍有障礙物時，可進行不易接觸到這些物品的動作。若要執行減小第5關節移動量的動作，請選擇姿態選擇旗標7；若要執行減小第4關節移動量的動作，請選擇姿態選擇旗標8。

- 本地編號

  以LJM函數傳回的點的本地編號與「點指定」的本地編號相同。

參照

Pallet

LJM函數範例

```
Function main
  Integer i, j

  P0 = XY(300, 300, 300, 90, 0, 180)
  P1 = XY(200, 280, 150, 90, 0, 180)
  P2 = XY(200, 330, 150, 90, 0, 180)
  P3 = XY(-200, 280, 150, 90, 0, 180)

  Pallet 1, P1, P2, P3, 10, 10

  Motor On
  Power High
  Speed 50; Accel 50, 50
  SpeedS 1000; AccelS 5000

  Go P0
  P11 = P0 -TLZ(50)

  For i = 1 To 10
      For j = 1 To 10
        '點指定
        P10 = P11    '閃避點
      P12 = Pallet(1, i, j)    '目標點
      P11 = P12 -TLZ(50)    '接近起始點
      '各點的LJM轉換
      P10 = LJM(P10)
      P11 = LJM(P11, P10)
      P12 = LJM(P12, P11)
      '執行動作
        Jump3 P10, P11, P12 C0
    Next
  Next
Fend

Function main2
  P0 = XY(300, 300, 300, 90, 0, 180)
  P1 = XY(400, 0, 150, 90, 0, 180)
  P2 = XY(400, 500, 150, 90, 0, 180)
  P3 = XY(-400, 0, 150, 90, 0, 180)
  Pallet 1, P1, P2, P3, 10, 10

  Motor On
  Power High
  Speed 50; Accel 50, 50
  SpeedS 1000; AccelS 5000

  Go P0

  Do
    '點指定
    P10 = Here -TLZ(50)    '閃避點
    P12 = Pallet(1, Int(Rnd(9)) + 1, Int(Rnd(9)) + 1)    '目標點
    P11 = P12 -TLZ(50)    '接近起始點

    If TargetOK(P11) And TargetOK(P12) Then    '點的檢查
    '各點的LJM轉換
    P10 = LJM(P10)
    P11 = LJM(P11, P10)
    P12 = LJM(P12, P11)
    '執行動作
      Jump3 P10, P11, P12 C0
```

```
    EndIf
   Loop
Fend
```

3.15.22 LoadPoints

將點檔案載入到機器人的點記憶區中。

格式

LoadPoints 檔名 [, Merge]

參數

檔案名稱

指定載入機器人點記憶體區的檔案之字串，副檔名固定為「.pts」。若省略副檔名，則新增「.pts」。指定檔案限定於專案內檔案。不可指定路徑。此外，不受ChDisk等的影響。詳細內容請參閱ChDisk。

Merge

不希望在載入新點之前清除目前點時，需事先設定此值。完成設定後，則在設定的點上新增點。新增的點已在檔案中時，則進行覆蓋。可省略。

說明

LoadPoints用於將點檔案載入到控制器的主記憶體中。

若要整合點檔案，則設定Merge。例如，假設有在0~100範圍內包括通用點的一個主要點檔案。若要在不清除這些點的情況下對目前動作的各零件載入新的點檔案，則設定Merge。此時的設定範圍為101~999。

常見錯誤

- 不可指定路徑

  檔案名稱含有路徑時會發生錯誤。

- 找不到指定檔案時(沒有檔案)

  找不到檔案時會發生錯誤。

- 其它機器人的點檔案

  若在檔名上指定其它機器人的點檔案，則發生錯誤。此時，使用專案編輯器新增點檔案，或者執行SavePoints或ImportPoints。

參照

ImportPoints、Robot、SavePoints

LoadPoints範例

```
Function main

  '將通用點載入目前機器人
  LoadPoints "R1Common.pts"

  '整合零件模型1的點
  LoadPoints "R1Model1.pts", Merge

  機器人 2
  ' 載入機器人2的點檔案
  LoadPoints "R2Model1.pts"

Fend
```

3.15.23 Local

定義和顯示本地坐標系。

格式

(1) Local 本地坐標系編號, (點編號1 : 點編號2), (點編號3 : 點編號4) [, { L | R }] [, BaseU]

(2) Local 本地坐標系編號, 坐標系資料

(3) Local 本地坐標系編號, 原點, [X軸指定], [Y軸指定], [{ X | Y }]

(4) Local 本地坐標系編號

參數

本地坐標系編號
　　指定本地坐標系的編號。可用1~15的整數(合計15個)定義本地坐標系。
點編號1、點編號3
　　以點變數指定本地坐標系的點數據。
點編號2、點編號4
　　以點變數指定基本坐標系的點數據。
L | R
　　將本地原點對齊左側(1號)或右側(2號)。可省略。
BaseU
　　若指定，U軸坐標則使用基本坐標系。可省略。若省略，U軸坐標則使用本地坐標系。
坐標系資料
　　以點數據直接指定本地坐標系的原點和方向。水平多關節機器人(包括RS系列)，請指定V坐標和W坐標為「0」。
原點
　　以P#(整數)或P(運算式)指定要定義本地坐標系原點的機器人坐標系上的位置。
X軸指定
　　以P#(整數)或P(運算式)指定要定義本地坐標系X軸上的點之機器人坐標系上的位置。可省略。
Y軸指定
　　以P#(整數)或P(運算式)指定要定義本地坐標系Y軸上的點之機器人坐標系上的位置。可省略。
X
　　將連結原點和X軸指定的直線定義為本地坐標系的X軸。由X軸和3個點創建的平面算出本地坐標系，因此Y軸指定不一定是Y軸上的點。可省略。(預設)
Y
　　將連結原點和Y軸指定的直線定義為本地坐標系的Y軸。由Y軸和3個點創建的平面算出本地坐標系，因此X軸指定不一定是X軸上的點。可省略。

說明

(1) Local用於指定基本坐標系的2點位置資料、與點編號2和4一致的本地坐標系的2點、點編號1以及點編號3，並定義本地坐標系。

例：

```
Local 1, (P1:P11), (P2:P12)
```

P1和P2是本地坐標系的點。P11和P12是基本坐標系的點。

若以本地坐標系指定的2點間距離不同於基本坐標系的2點間距離，則在本地坐標系的2點之間中點和基本坐標系的2點之間中點一致的位置上，定義本地坐標系。

同樣，使用2個坐標系的中點定義本地坐標系的Z軸。

(2) 以原點和對於基本坐標系的角度，來定義本地坐標系。

例：

```
Local 1, XY(x, y, z, u)
Local 1, XY(x, y, z, u, v, w)
Local 1, P1
```

(3) 指定原點、X軸上點、Y軸上點，並定義3維本地坐標系。只使用各點的X、Y、Z坐標，忽略U、V、W坐標。若指定參數X，X軸指定便位於本地坐標系的X軸上，僅使用Y軸指定的Z坐標。若指定參數Y，Y軸指定便位於本地坐標系的Y軸上，僅使用X軸指定的Z坐標。

例：

```
Local 1, P1, P2, P3
Local 1, P1, P2, P3, X
Local 1, P1, P2, P3, Y
```

(4) 顯示指定本地的設定。

- 使用L/R參數：如上所述，主要以中點定義本地坐標系，不過也可使用選項的L或R，來指定坐標系的左右側。
  - 左本地：左本地用於在本地坐標系的點編號1與基本坐標系的點編號2一致的位置上定義本地坐標。(亦包括Z軸方向。)
  - 右本地：右本地用於在本地坐標系的點編號3與基本坐標系的點編號4一致的位置上定義本地坐標。(亦包括Z軸方向。)
- 使用BaseU參數：若省略BaseU參數，會自動補償本地坐標系的U軸，以符合所設4點XY坐標值。也有可能基本坐標系的2點從一開始便包含U軸坐標值。

  相較於自動補償，建議透過以教導補償旋轉軸等手段，基於基本坐標系的2點U軸值來對本地坐標系的U軸進行補償。此時，需設定BaseU參數。

注意

- 水平多關節機器人(包括RS系列)，請不要指定V坐標和W坐標。

  使用水平多關節機器人時，請不要設定本地坐標系中的V,W坐標的數值，或者設定為「0」。如果設定了數值，可能會導致J4範圍外或發生錯誤等。

參照

ArmSet、Base、ECPSet、LocalClr、TLSet、Where

Local範例

這是命令視窗中的操作範例。

在左本地定義本地坐標系原點的範例：

```
> p1 = 0, 0, 0, 0/1
> p2 = 100, 0, 0, 0/1
> p11 = 150, 150, 0, 0
> p12 = 300, 150, 0, 0
> local 1, (P1:P11), (P2:P12), L

> p21 = 50, 0, 0, 0/1
> go p21
```

將原點定義為本地坐標系原點的範例：

```
> local 1, 100, 200, -20, 0
```

將讓X軸旋轉45度的原點定義為本地坐標系原點的範例：

```
> local 2, 50, 200, 0, 0, 45, 0
```

將讓P2對齊本地坐標系X軸的位置定義為3維本地座標系原點的範例：

```
> local 3, p1, p2, p3, x
```

將讓P3對齊本地坐標系Y軸的位置定義為3維本地坐標系原點的範例：

```
> local 4, p1, p2, p3, y
```

3.15.24 Local函數

本函數用於傳回已設定的本地坐標系資料。

格式

Local (本地坐標系編號)

參數

本地坐標系編號

以運算式或數值指定本地坐標系編號(1~15的整數)。

傳回值

用於將設定的本地坐標系資料作為點資料傳回。

參照

Local

Local函數範例

```
P1 = Local(1)
```

3.15.25 LocalClr

用於清除本地坐標系(未定義)。

格式

LocalClr 本地坐標系編號

參數

本地坐標系編號

以運算式或數值指定要清除(未定義)設定的本地坐標系編號(1~15的整數)。

參照

Arm、ArmSet、ECPSet、Local、Tool、TLClr、TLSet

LocalClr範例

```
LocalClr 1
```

3.15.26 LocalDef函數

用於傳回本地的設定狀態。

格式

LocalDef (本地坐標系編號)

參數

本地坐標系編號
　　指定要傳回狀態的本地坐標系編號(1~15的整數)。

傳回值

若有設定指定的本地，則傳回「True」；若未設定，則傳回「False」。

參照

Arm、ArmClr、ArmSet、ECPSet、Local、LocalClr、Tool、TLClr、TLSet

LocalDef函數範例

```
Function DisplayLocalDef(localNum As Integer)

   If LocalDef(localNum) = False Then
      Print "Local ", localNum, "is not defined"
   Else
      Print "Local 1: ",
      Print Local(localNum)
   EndIf
Fend
```

3.15.27 Lof函數

用於傳回指定的RS-232C連接埠或TCP/IP連接埠緩衝區所接收資料的行數。

格式

Lof (通訊連接埠編號)

參數

通訊連接埠編號

指定以OpenCom(RS-232C)或OpenNet(TCP/IP)陳述式指定的編號。

傳回值

用於傳回接收緩衝區的資料行數。沒有接收資料時，則傳回「0」。

說明

Lof函數用於確認指定通訊連接埠有無接收資料。以Input#命令產生接收的資料。可用Wait命令等待Lof函數的傳回值。

注意

- 支援的控制器型號

  如果在T/VT系列中指定RS-232C埠時會發生錯誤。

  正在使用PC COM連接埠(1001~1008)時，則無法以Wait命令使用Lof函數。

參照

ChkCom、ChkNet、Input#、Wait

Lof函數範例

在命令視窗中執行如下範例。顯示通訊連接埠1所接收資料的行數。

```
>print lof(1)
 5
>
```

3.15.28 LogIn

以其它使用者身份登入Epson RC+。

格式

LogIn 登入ID, 密碼

參數

登入ID
　　用戶登入ID的字串運算式
密碼
　　用戶密碼的字串運算式

說明

可透過應用程式操作Epson RC+的安全性。例如，可顯示讓其它使用者登入系統的選單。使用者擁有獨自的安全權限。關於安全的詳細內容，請參閱以下手冊。
「Epson RC+ 使用者指南」

若在開發環境中執行程式，程式停止後，使用者會恢復為啟動程式前的狀態。

在Auto模式下執行操作員視窗時，只要啟用Auto LogIn，才可以作為來賓使用者登入應用程式。此時，若有透過Epson RC+系統進行設定，則可以作為目前的Windows使用者登入應用程式。

注意

僅限於啟用安全選項時，方可使用此命令。

LogIn範例

```
Integer errCode
errCode = LogIn("operator", "oprpass")
```

3.15.29 Long

宣告Long型變數。(4位元組整數型變數)

格式

Long 變數名稱 [(陣列變數的最大元素編號)] [, 變數名稱 [(陣列變數的最大元素編號)]...]

參數

變數名稱

指定宣告為Long型的變數名稱。陣列變數的最大元素編號是陣列變數的最大元素編號，最大可宣告三維。使用如下格式。可省略。

(最大元素編號1, [最大元素編號2], [最大元素編號3])

元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。在所有元素數不超過以下最大值的範圍內，指定各最大元素編號。

- 本地變數：2,000
- 備份變數(Global Preserve)：4,000
- 全域變數和模組變數：100,000

說明

Long用於宣告整數型變數。Long型變數範圍：-2147483648~2147483647。在Function的開頭宣告本地變數。在Function之外宣告全域變數和模組變數。

參照

Boolean、Byte、Double、Global、Int32、Int64、Integer、Real、Short、String、UByte、UInt32、UInt64、UShort

Long範例

以下範例是使用Long宣告Long型變數的程式。

```
Function longtest

    Long A(10)          'Long型的一維陣列
    Long B(10, 10)      'Long型的二維陣列
    Long C(5, 5, 5)     'Long型的三維陣列
    Long var1, arrayVar(10)
    Long i
    Print "Please enter a Long Number"
    Input var1
    Print "The Integer variable var1 = ", var1
    For i = 1 To 5
        Print "Please enter a Long Number"
        Input arrayVar(i)
        Print "Value Entered was ", arrayVar(i)
    Next I
Fend
```

3.15.30 LSet$函數

在指定字串的末尾 加入半型空格，並傳回指定長度的字串。

格式

LSet$ (字串,字串的長度)

參數

字串
　　指定字串運算式。
字串的長度
　　指定整數或運算式，以表示要傳回的字串長度。

傳回值

用於傳回在指定字串的末尾加入半型空格的字串。

參照

RSet$、Space$

LSet$函數範例

```
temp$ = "123"
temp$ = LSet$(temp$, 10)  ' temp$ = "123       "
```

3.15.31 LShift函數

依照指定位元數使數值資料左移。

格式

LShift (數值, 移位位元數)

參數

數值
　　指定要移位的整數值。
移位位元數
　　指定要左移位的位元數(0~31的整數值)。

傳回值

用於傳回依照指定位元數使指定數值左移的結果。

說明

依照指定位元數使指定數值向左(位較大一側)移位。移位部分的低階位元始終被設為0。

Lshift與數值 * 2移位位元數(只將數值乘以2移位位元數的次數)一致。

注意

- 數值資料型態

  數值型有多個種類。Lshift可用於Byte型、Double型、Int32型、Integer型、Long型、Real型、Short型、UByte型、UInt32型、UShort型數值。

參照

And、LShift64、Not、Or、RShift、RShift64、Xor

LShift函數範例

```
Function lshiftst
    Integer i
    Integer num, snum
    num = 1
    For i = 1 to 10
         Print "i =", i
         snum = LShift(num, i)
         Print "The shifted num is ", snum
    Next i
Fend
```

這是在命令視窗中傳回Lshift函數結果的其它範例。

```
> Print LShift(2,2)
8
> Print LShift(5,1)
10
> Print LShift(3,2)
12
>
```

3.15.32 LShift64函數

依照指定位元數使數值資料左移。

格式

LShift64 (數值, 移位位元數)

參數

數值
　　指定要移位的整數值。
移位位元數
　　指定要左移位的位元數(0~63的整數值)。

傳回值

用於傳回依照指定位元數使指定數值左移的結果。

說明

依照指定位元數使指定數值向左(位較大一側)移位。移位部分的低階位元始終被設為0。

Lshift64與數值 * 2移位位元數(只將數值乘以2移位位元數的次數)一致。

注意

- 數值資料型態

  數值型有多個種類。Lshift64可用於Int64型、UInt64型數值。

參照

And、LShift、Not、Or、RShift、RShift64、Xor

LShift64函數範例

```
Function lshiftst
    Int64 i
    Int64 num, snum
    num = 1
    For i = 1 to 10
         Print "i =", i
         snum = LShift64(num, i)
         Print "The shifted num is ", snum
    Next i
Fend
```

這是在命令視窗中傳回Lshift64函數結果的其它範例。

```
> Print LShift64(2,2)
8
> Print LShift64(5,1)
10
> Print LShift64(3,2)
12
>
```

3.15.33 LTrim$函數

用於傳回刪除左側半型空格後的字串。

格式

LTrim$ (字串)

參數

字串

指定字串運算式。

傳回值

用於傳回刪除左側半型空格的字串。

參照

RTrim$、Trim$

LTrim$函數範例

```
str$ = "  data  "
str$ = LTrim$(str$)  ' str$ = "data  "
```

3.16 M

3.16.1 Mask運算子

以位元為單位遮罩表示Wait命令條件的值。

格式

Wait 值1 Mask 值2

參數

值1

指定表示Wait輸入條件的值。

值2

指定以result傳回的數值。

說明

Mask運算子用於對表示Wait輸入條件的值進行位元And運算。

參照

Wait

Mask運算子範例

```
'等待輸入連接埠0的低階3位元變為1
Wait In(0) Mask 7 = 1
```

3.16.2 MCal

進行增量型編碼器機器人的原點復歸操作(機械原點檢測)。

格式

MCal

說明

增量型編碼器的機器人必須進行原點復歸操作(機械原點檢測)。請在接通電源後執行此原點復歸操作。若在執行原點復歸前執行動作命令或需要目前位置資料的其它命令，則發生錯誤。

依照以MCordr命令指定的關節順序執行原點復歸。出廠時的MCordr初始值會因機械手的機種而異。有關詳細資訊，請參閱以下手冊。
「機械手手冊 」

常見錯誤

- 欲在執行MCal前執行動作命令時

  若在執行原點復歸前執行動作命令或需要目前位置資料的其它命令(例如，Plist*命令等)，則發生錯誤。

- 搭載絕對式編碼器的機器人

  搭載絕對式編碼器的機器人不需要MCal。

機器人設置注意事項

- 用於第3關節原點復歸的空間

  第3關節原點復歸時，首先是上升移動，然後下降並停在原點。因此，事先確保可讓第3關節進行原點復歸的空間，顯得格外重要。建議在Z上限值的上方確保6mm以上的空間。(請勿在機器人上方6mm內的空間設置工具或冶具。)

參照

Hofs、Home、Hordr、Mcorg、MCordr

MCal範例

這是監視器視窗中的操作範例。

```
> Motor On
> Mcal
>
```

3.16.3 MCalComplete函數

用於傳回MCal的狀態。

格式

MCalComplete

傳回值

MCal正常運作時，傳回True；除此之外，傳回False。

參照

MCal

MCalComplete函數範例

```
If Not MCalComplete Then
    MCal
EndIf
```

3.16.4 MCordr

用於指定和顯示以MCal進行原點復歸時的關節動作順序。搭載增量型編碼器的機器人才需要執行。

格式

(1) MCordr 設定值1, 設定值2, 設定值3, 設定值4[, 設定值5][, 設定值6][, 設定值7][, 設定值8] [, 設定值9]

(2) MCordr

參數

設定值1
　以位元模式(二進位的值)指定在MCal流程的第1步驟進行原點復歸的關節(0~9)。(請參閱以下位元模式表。)
設定值2
　以位元模式(二進位的值)指定在MCal流程的第2步驟進行原點復歸的關節(0~9)。(請參閱以下位元模式表。)
設定值3
　以位元模式(二進位的值)指定在MCal流程的第3步驟進行原點復歸的關節(0~9)。(請參閱以下位元模式表。)
設定值4
　以位元模式(二進位的值)指定在MCal流程的第4步驟進行原點復歸的關節(0~9)。(請參閱以下位元模式表。)
設定值5
　以位元模式(二進位的值)指定在MCal流程的第5步驟進行原點復歸的關節(0~9)。(請參閱以下位元模式表。)
設定值6
　以位元模式(二進位的值)指定在MCal流程的第6步驟進行原點復歸的關節(0~9)。(請參閱以下位元模式表。)
設定值7
　以位元模式(二進位的值)指定在MCal流程的第7步驟進行原點復歸的關節(0~9)。(請參閱以下位元模式表。)
設定值8
　以位元模式(二進位的值)指定在MCal流程的第8步驟進行原點復歸的關節(0~9)。(請參閱以下位元模式表。)
設定值9
　以位元模式(二進位的值)指定在MCal流程的第9步驟進行原點復歸的關節(0~9)。(請參閱以下位元模式表。)

傳回值

若省略參數，則顯示目前機械原點復歸順序。

說明

開啟電源時，請務必在運作手臂前執行MCal命令。若執行MCal命令，各關節則移至各自的原點復歸位置。

指定執行MCal命令時的關節動作順序。以設定值1指定的關節進行動作並結束原點復歸之後，以設定值2指定的關節開始動作。如此，依照設定值3的關節、設定值4的關節之順序進行原點復歸。

MCordr命令的意義在於，讓使用者得以變更原點復歸時各關節的復歸順序。分9個步驟設定復歸順序。使用者可以MCordr指定在各步驟進行復歸的關節。還可指定在各步驟進行復歸的數個關節。然而，一般都建議在最初的步驟1移動第3關節，然後在後續步驟讓其它關節復歸。(請參閱注意。)

執行MCordr命令時，必須對9個步驟指定位元模式。規定了對各關節的位元模式。若在某步驟位元為「1」，相應關節則會進行原點復歸。若位元為「0」，相應關節則不會在該步驟進行原點復歸。對各關節分配的位元模式如下所示。

位元模式表

關節名稱	位元編號	二進位表示
第1關節	bit 0	&B000001
第2關節	bit 1	&B000010
第3關節	bit 2	&B000100
第4關節	bit 3	&B001000

關節名稱	位元編號	二進位表示
第5關節	bit 4	&B010000
第6關節	bit 5	&B100000
第7關節	bit 6	&B1000000
第8關節	bit 7	&B10000000
第9關節	bit 8	&B100000000

注意

- MCordr和Hordr的差異

  Hordr和MCordr命令在重要方面有所不同。MCordr連同MCal一起使用，用於指定機器人原點復歸時的關節復歸順序。而Hordr連同Home一起使用，用於指定到Home位置的關節復歸順序。

- 到原點復歸位置的預設復歸順序

  出廠設定如下所示。

- 第3關節(Z)在第1步驟進行復歸。
- 第1關節(X)、第2關節(Y)、第4關節(U)在第2步驟同時復歸到原點復歸位置。
- 未使用第3、第4步驟。預設值如下所示。

```
MCordr &B0100, &B1011, 0, 0
```

- 通常，先讓第3關節(Z)進行原點復歸

  先單獨讓第3關節(Z)復歸的原因在於，在進行水平移動前，從工件表面移開工具。這是為了預防在進行原點復歸時，工具干擾動作區域內的物品。

- 會保持MCordr值

  只要未變更使用者或未重新設定機器人，就繼續保持MCordr值。

參照

MCal

MCordr範例

以下是監視器視窗中操作4軸機器人的範例。

在此例中，以二進位按如下所述設定原點復歸順序。

第3關節在第1步驟進行原點復歸；而第1關節在第2步驟進行原點復歸；第2關節在第3步驟進行原點復歸；第4關節在第4步驟進行原點復歸。

```
> MCordr  &B0100, &B0001, &B0010, &B1000
```

在此例中，以十進位按如下所述設定原點復歸順序。

第3關節在第1步驟進行原點復歸；第1關節、第2關節和第4關節在第2步驟同時進行原點復歸。

```
> MCordr  4, 11, 0, 0
```

在以下範例中，以十進位顯示目前的原點復歸順序。

```
>mcordr
4, 11, 0, 0
>
```

3.16.5 MCordr函數

用於傳回MCordr參數設定。

格式

MCordr (設定值編號)

參數

設定值編號

以運算式或數值指定要參照的設定值編號(1~9的整數)。

傳回值

用於以二進位的值(整數)傳回要進行原點復歸的指定關節的設定值。

說明

傳回以MCal進行原點復歸時的關節動作順序。

參照

MCal

MCordr函數範例

這是使用MCordr函數的程式範例。

```
Integer a
a = MCordr(1)
```

3.16.6 MemIn函數

用於傳回指定記憶體I/O連接埠的狀態。各連接埠有8個記憶體位元。

格式

MemIn (連接埠編號)

參數

連接埠編號
　　指定記憶體I/O的位元組。

傳回值

用於傳回0~255的整數。傳回值為8位元。各位元對應1個記憶體I/O位元。

說明

可透過MemIn一次查看8個記憶體I/O位元的值。MemIn命令的作用在於，可將8個記憶體I/O位元的狀態作為1個變數進行儲存。另外，透過同時使用Wait等待程式，直到與2個以上I/O位元的相關特定條件一致。

1次可獲取8位元的值，因此傳回值的範圍為0~255。請在下表參照各傳回值和各記憶體I/O位元狀態的對應關係。

記憶體I/O位元表(使用連接埠0時)

傳回值	7	6	5	4	3	2	1	0
1	Off	Off	Off	Off	Off	Off	Off	On
5	Off	Off	Off	Off	Off	On	Off	On
15	Off	Off	Off	Off	On	On	On	On
255	On	On	On	On	On	On	On	On

記憶體I/O位元表(使用連接埠31時)

傳回值	255	254	253	252	251	250	249	248
3	Off	Off	Off	Off	Off	Off	On	On
7	Off	Off	Off	Off	Off	On	On	On
32	Off	Off	On	Off	Off	Off	Off	Off
255	On	On	On	On	On	On	On	On

注意

- MemIn和MemSw的差異

  可以MemSw命令讀取記憶體I/O位元1的值。MemSw的傳回值為0或1，表示記憶體I/O位元為On或Off。MemSw用於個別檢查記憶體I/O位元。對於檢查記憶體I/O位元狀態這一點，MemIn命令類似於MemSw命令。但是，MemSw命令針對每1位元檢查記憶體I/O狀態，而MemIn命令可同時檢查8位元。MemIn傳回值為0~255的值。可根據該值瞭解8位元當中哪一個是On/Off。

參照

In、InBCD、Off、MemOff、On、MemOn、OpBCD、Oport、Out、MemOut、Sw、MemSw、Wait

MemIn函數範例

以下是求出前8個記憶體I/O的目前值，然後確認所有8個值皆為「0」，並進入下一步的程式範例。若傳回值不是「0」，則會顯示錯誤訊息，並停止工作。

```
Function main
    Integer var1

    var1 = MemIn(0)  '取得前8個記憶體I/O位元的值
    If var1 = 0 Then
        Go P1
        Go P2
    Else
        Print "Error in initialization!"
        Print "First 8 memory I/O bits were not all set to 0"
    EndIf
Fend
```

以下是命令視窗中的簡易操作範例。

```
> memout 0, 1
> print MemIn(0)
1
> memon 1
> print MemIn(0)
3
> memout 31,3
> print MemIn(31)
3
> memoff 249
> print MemIn(31)
1
>
```

3.16.7 MemInW函數

用於以字組為單位傳回記憶體I/O連接埠的狀態。字組連接埠由16個記憶體I/O位元構成。

格式

MemInW (字組連接埠編號)

參數

字組連接埠編號
　　指定I/O的字組連接埠。

傳回值

用於傳回記憶體I/O連接埠的狀態(0~65535的Long型整數)。

參照

MemIn、MemOut、MemOutW

MemInW函數範例

```
Long word0

word0 = MemInW(0)
```

3.16.8 MemOff

將記憶體I/O的指定位元設為OFF。

格式

MemOff { 位元編號 | 記憶體I/O標籤 }

參數

位元編號
　　以整數指定記憶體I/O的位元。
記憶體I/O標籤
　　指定記憶體I/O標籤。

說明

MemOff用於將指定記憶體I/O的位元設為OFF(0)。以MemSw命令檢查指定記憶體位元的狀態。Wait命令也用於記憶體位元，讓系統等待，直到變為指定的記憶體I/O狀態。

注意

- 記憶體I/O的OFF

  重新啟動控制器時，則將記憶體I/O置為OFF。緊急停止、打開安全門、程式結束、Reset、重新啟動Epson RC+時不置為OFF。

參照

In、MemIn、InBCD、Off、On、MemOn、OpBCD、Oport、Out、MemOut、Sw、MemSw、Wait

MemOff範例

以下是各自啟動動作命令的2項工作之使用範例。在這2項工作中啟用聯鎖功能，以便在另一方結束機器人動作命令時，依序對控制進行增益處理。因此，2項工作可依序按照各自的指示執行動作陳述式。MemSw連同Wait命令一起使用。在重新開始動作之前，會等待記憶體I/O位元1變為適當值，以確保安全。MemOn和MemOff用於啟用或停用記憶體I/O，以正確地進行同步。

```
Function main
    Integer I
    MemOff 1
    Xqt 2, task2
    For i = 1 to 100
        Wait MemSw(1) = Off
        Go P(i)
        MemOn 1
    Next I
Fend


Function task2
    Integer I
    For i = 101 to 200
        Wait MemSw(1) = On
        Go P(i)
        MemOff 1
    Next I
Fend
```

以下是命令視窗中的簡易操作範例。

```
> MemOn 1          '啟用記憶體I/O位元1
> Print MemSw(1)
1
> MemOff 1         '將記憶體I/O位元1設為OFF
> Print MemSw(1)
0
```

3.16.9 MemOn

啟用記憶體I/O的指定位元。

格式

MemOn {位元編號 | 記憶體I/O標籤}

參數

位元編號
　　以整數指定記憶體I/O的位元。
記憶體IO標籤
　　指定記憶體I/O標籤。

說明

MemOn用於啟用(1)指定位元。以MemSw命令檢查指定記憶體位元的狀態。Wait命令也用於記憶體位元，讓系統等待，直到變為指定S/W狀態。

注意

- 記憶體I/O的OFF

  重新啟動控制器時，則將記憶體I/O置為OFF。緊急停止、打開安全門、程式結束、Reset、重新啟動Epson RC+時不置為OFF。

參照

In、MemIn、InBCD、Off、MemOff、On、OpBCD、Oport、Out、MemOut、Sw、MemSw、Wait

MemOn範例

以下是各自啟動動作命令的2項工作之使用範例。在這2項工作中啟用聯鎖功能，以便在另一方結束機器人動作命令時，依序對控制進行增益處理。因此，2項工作可依序按照各自的指示執行動作陳述式。MemSw連同Wait命令一起使用。在重新開始動作之前，會等待記憶體I/O位元1變為適當值，以確保安全。MemOn和MemOff用於啟用或停用記憶體I/O，以正確地進行同步。

也可以將Signal和Waitsig命令用於工作同步。

```
Function main
    Integer I
    MemOff 1
    Xqt 2, task2
    For i = 1 to 100
        Wait MemSw(1) = Off
        Go P(i)
        MemOn 1
    Next I
Fend

Function task2
    Integer I
    For i = 101 to 200
        Wait MemSw(1) = On
        Go P(i)
        MemOff 1
    Next I
Fend
```

以下是命令視窗中的簡易操作範例。

```
> memon 1
> print memsw(1)
1
> memoff 1
> print memsw(1)
0
```

3.16.10 MemOut

用於同時設定8個記憶體I/O位元。

格式

MemOut 連接埠編號, 輸出資料

參數

連接埠編號

指定記憶體I/O的位元組連接埠編號。如下所述，連接埠編號與位元對應。

連接埠編號	輸出資料
0	0-7
1	8-15
...	...

輸出資料

以0~255的整數值指定以連接埠編號指定的輸出群組之輸出模式。若以十六進位顯示，則以&H0~&HFF為其範圍。低階位表示低階位元(或前4個輸出位元)；高階位表示高階位元(或後4個輸出位元)。

說明

MemOut的作用在於，透過將指示設定的輸出位元的連接埠編號和輸出資料進行組合，同時設定8個記憶體I/O位元。以連接埠編號參數指定使用的群組(8個輸出位元)。例如，連接埠編號 = 0 時，指定輸出位元0~7。連接埠編號 = 1時，指定輸出位元8~15。

首先，以連接埠編號指定8個輸出位元之後，再利用輸出資料參數來定義特定輸出模式。輸出資料參數可獲得的值為0~255，以十六進位或十進位的整數進行指定。(&H0~&HFF或0~255)

下表所示為，分為連接埠編號為「0」以及「1」兩種情況，在顯示與其相應的輸出資料值的同時，顯示部分I/O組合範例的情況。

連接埠編號=0時的輸出設定(輸出位元編號)

輸出資料值	7	6	5	4	3	2	1	0
01	Off	Off	Off	Off	Off	Off	Off	On
02	Off	Off	Off	Off	Off	Off	On	Off
03	Off	Off	Off	Off	Off	Off	On	On
08	Off	Off	Off	Off	On	Off	Off	Off
09	Off	Off	Off	Off	On	Off	Off	On
10	Off	Off	Off	Off	On	Off	On	Off
11	Off	Off	Off	Off	On	Off	On	On
99	Off	On	On	Off	Off	Off	On	On
255	On	On	On	On	On	On	On	On

連接埠編號=1時的輸出設定(輸出位元編號)

輸出資料值	15	14	13	12	11	10	9	8
01	Off	Off	Off	Off	Off	Off	Off	On
02	Off	Off	Off	Off	Off	Off	On	Off
03	Off	Off	Off	Off	Off	Off	On	On
08	Off	Off	Off	Off	On	Off	Off	Off
09	Off	Off	Off	Off	On	Off	Off	On
10	Off	Off	Off	Off	On	Off	On	Off
11	Off	Off	Off	Off	On	Off	On	On
99	Off	On	On	Off	Off	Off	On	On
255	On	On	On	On	On	On	On	On

參照

In、MemIn、InBCD、Off、MemOff、On、MemOn、OpBCD、Oport、Out、Sw、MemSw、Wait

MemOut範例

以下是用於啟動以「iotask」為名的主工作之程式。「iotask」是用於啟用/停用S/W記憶體I/O位元的0~3的簡易工作。若使用MemOut命令，則可用1個命令進行操作，不必分別啟用/停用S/W記憶體I/O位元。

```
Function main
  Xqt 2, iotask
  Go P1
  .
  .
Fend

Function iotask

  Do
    MemOut 0, &H

    Wait 1
    MemOut 0, &H0
    Wait 1
  Loop
Fend
```

以下是命令視窗中的簡易操作範例。

```
> MemOut 1,6     '啟用記憶體I/O位元9和10
> MemOut 2,1     '啟用記憶體I/O位元8
> MemOut 3,91     '啟用記憶體I/O位元24、25、27、28、30
```

3.16.11 MemOutW

用於以字組為單位，同時以16位元設定記憶體I/O連接埠的狀態。

格式

MemOutW 字組連接埠編號, 輸出資料

參數

字組連接埠編號
　　指定記憶體I/O字組。
輸出資料
　　以運算式或數值指定記憶體I/O資料(0~65535的整數)。

說明

將以參數字組連接埠編號指定的記憶體I/O連接埠群組的狀態，變更為指定的輸出資料。

參照

MemIn、MemInW、MemOut

MemOutW範例

```
MemOutW 0, 25
```

3.16.12 MemSw函數

用於傳回指定記憶體I/O位元的狀態。

格式

MemSw (位元編號)

參數

位元編號

以表示記憶體I/O位元編號的數值指定。

傳回值

指定的位元為ON時，傳回「1」；OFF時，傳回「0」。

說明

MemSw用於傳回1記憶體I/O位元的狀態。通常，MemOn和MemOff命令一起用於MemSw。MemOn用於啟用指定位元，MemOff則用於停用指定位元。

參照

In、MemIn、InBCD、Off、MemOff、On、MemOn、OpBCD、Oport、Out、MemOut、Sw、Wait

MemSw函數範例

在以下範例中表示的是，2項工作各自具有啟動動作命令的功能，並有施加聯鎖，以確保一方未控制機器人時，由另一方控制機器人動作。藉此，各工作可依照預期的順序確實地執行被賦予的動作陳述式。

通過將MemSw連同Wait命令一起使用，在記憶體I/O位元1成為可安全執行下一動作的適當值之前進行等待，然後重新開始動作。

```
Function main
    Integer I
    MemOff 1
    Xqt 2, task2
    For i = 1 to 100
        Wait MemSw(1) = Off
        Go P(i)
        MemOn 1
    Next I
Fend

Function task2
    Integer I
    For i = 101 to 200
        Wait MemSw(1) = On
        Go P(i)
        MemOff 1
    Next I
Fend
```

以下是命令視窗中的簡易操作範例。

```
> memon 1
> print memsw(1)
1
> memoff 1
> print memsw(1)
0
```

3.16.13 MHour函數

用於傳回機器人馬達累計勵磁時間。

格式

MHour ([機器人編號])

參數

機器人編號

以整數值指定要確認勵磁時間的機器人編號。省略時，即以目前選擇的機器人為對象。

傳回值

用於以實數值傳回馬達的累計勵磁時間。

參照

Time、Hour

MHour函數範例

```
Robot 2
Print MHour
Print MHour(1)
```

3.16.14 Mid$函數

用於從以字串指定的開始位置取出指定數量的字元。

格式

Mid$ (字串, 開始位置 [, 字元數])

參數

字串
　　指定原始字串。
開始位置
　　指定要取出字串的開始位置。
字元數
　　指定要從字串取出的字元數。可省略。若省略，則傳回從開始位置到字串結尾的字元。

傳回值

用於傳回從原始字串取出的字串。

說明

Mid$用於從以原始字串指定的開始位置，依照字元數取出字串。

參照

Asc、Chr$、InStr、Left$、Len、Right$、Space$、Str$、Val

Mid$函數範例

以下範例表示，從字串「ABCDEFGHIJ」的正當中取出2個字元，並取出開始位置5的剩餘字串的情況。

```
Function midtest
    String basestr$, m1$, m2$
    basestr$ = "ABCDEFGHIJ"
    m1$ = Mid$(basestr$, (Len(basestr$) / 2), 2)
    Print "The middle 2 characters are: ", m1$
    m2$ = Mid$(basestr$, 5)
    Print "The string starting at 5 is: ", m2$
Fend
```

3.16.15 MkDir

用於建立子目錄。

格式
MkDir 目錄名稱

參數

目錄名稱
　　指定要創建的子目錄路徑和名稱的字串。路徑的詳細內容請參閱ChDisk。

說明
對指定的路徑建立新目錄。若省略路徑，則在目前目錄中建立子目錄。

注意

磁碟為PC時，可執行。

參照
ChDir、ChDrive、RenDir、RmDir

MkDir範例
命令視窗中的操作範例

```
> MkDir \Data

> MkDir \Data\PTS

> MkDir \TEST1 \TEST2
```

3.16.16 Mod運算子

用於傳回將運算式除以其它數值運算式的剩餘值。

格式
被除數 Mod 除數

參數

被除數
　　指定被除的數。
除數
　　指定除數。

結果
用於傳回將被除數除以除數的剩餘值。

說明
Mod用於獲取除2個數值後的剩餘值(整數)。若用於調查是偶數還是奇數，Mod命令十分方便。

如下所述是Mod命令的程序。

：將被除數除以除數。此除算的剩餘值作為Mod命令的傳回值被傳回。

參照
Abs、Atan、Atan2、Cos、Int、Not、Sgn、Sin、Sqr、Str$、Tan、Val

Mod運算子範例
在以下範例中，調查某數值(var1)是偶數還是奇數的情況。該數值若為偶數，Mod命令則傳回結果「0」。若是奇數，則傳回結果「1」。

```
Function modtest
Integer var1, result

Print "Enter an integer number:"
Input var1
result = var1 Mod 2
Print "Result = ", result
If result = 0 Then
Print "The number is EVEN"
Else
Print "The number is ODD"
EndIf
Fend
```

作為其它範例，還有命令視窗中的操作範例。

```
> Print 36 Mod 6
> 0

> Print 25 Mod 10
> 5
>
```

3.16.17 Motor

開啟或關閉機器人所有軸的馬達。

格式

Motor {On | Off}

參數

On | Off

將馬達電源設為ON狀態時，則指定On；設為OFF狀態時，則指定Off。

說明

Motor On命令用於將馬達電源設為ON，並解除所有軸的制動。Motor Off命令用於將馬達電源設為OFF，並設定所有軸的制動。

若要運作機器人，務必將馬達電源設為ON。

在緊急停止後或發生需要Reset命令的錯誤後，執行Reset命令，然後進行Motor On操作。

執行Motor On命令後，對機器人控制參數進行如下設定。

機器人控制參數

- Speed和SpeedR、SpeedS的設定值：(初始化為初始值。)
- Accel和AccelR、AccelS的設定值：(初始化為初始值。)
- QPDecelR、QPDecelS的設定值：(初始化為初始值。)
- LimZ參數的設定值：(初始化為0。)
- CP參數的設定值：(初始化為Off。)
- SoftCP參數的設定值：(初始化為Off。)
- Fine的設定：(初始化為初始值。)
- Power Low設定：(變為低功率模式。)
- PTPBoost的設定值：(初始化為初始值。)
- TCLim、TCSpeed的設定值：(初始化為初始值。)
- PgLSpeed的設定值：(初始化為初始值。)
- PerformMode的設定值：(初始化為標準模式。)

參照

Brake、Power、Reset、SFree、SLock

Motor範例

在命令視窗中執行如下範例。

```
> Motor On

> Motor Off
```

3.16.18 Motor函數

用於傳回指定機器人的馬達電源狀態。

格式

Motor [(機器人編號)]

參數

機器人編號

以整數值指定要確認狀態的機器人編號。省略時，即以目前選擇的機器人為對象。

傳回值

- 0 = 馬達OFF
- 1 = 馬達ON

參照

Motor

Motor函數範例

```
If Motor = Off Then
    Motor On
EndIf
```

3.16.19 Move

將手臂從目前位置到指定位置之間進行直線內插動作。

格式

Move 目標坐標 [ROT] [ECP] [CP] [Till | Find] [!平行處理!] [SYNC]

參數

目標坐標
　以點數據指定目標位置。
ROT
　以工具姿態變化為優先，決定動作速度和加減速度。可省略。
ECP
　用於指定外部控制點的動作。可省略。(唯有使用ECP選項時方可啟用)
CP
　指定路徑運動。可省略。
Till | Find
　用於記述Till或Find運算式。可省略。

```
Till | Find
Till Sw(運算式) = {On | Off}
Find Sw(運算式) = {On | Off}
```

! 平行處理 !
　可增加平行處理陳述式，以便在動作中可執行I/O等命令。可省略。
SYNC
　用於預約動作命令。在以SyncRobots開始動作之前，機器人不進行動作。

說明

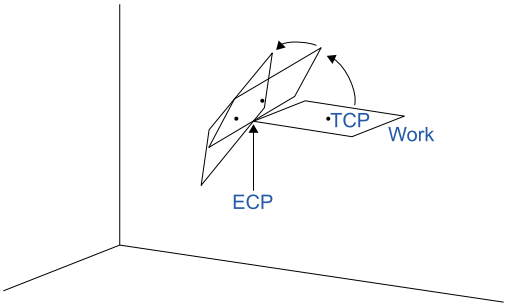
Move用於將手臂從目前位置直線移動至目標坐標。Move可讓所有關節同時開始動作、同時停止。執行Move命令前，應事先對目標坐標進行教導。以AccelS命令控制Move的加減速。以SpeedS命令控制Move的速度。若用SpeedS設定的值超過任何一個各關節容許速度，則關閉馬達勵磁，並停止機器人。

Move的速度和加減速度分別使用SpeedS和AccelS的設定值。關於速度和加減速度的關係，請參閱「注意」項目中的「連同CP使用Move」。但是，指定ROT修飾參數時的速度和加減速度分別使用SpeedR和AccelR的設定值。此時，SpeedS和AccelS的設定值即呈停用狀態。附加ROT修飾參數時，若姿態無變化且移動距離不是「0」，會發生錯誤。

機器人的工具中心點位置固定為特定坐標，僅改變工具姿態時，或者相對於工具中心點的移動距離，工具姿態變化較大時，工具姿態變化速度可能會明顯變快。為防止這種情形，工具姿態變化速度大時，自動限制動作速度的功能就會啟用。

想要手動設定CP動作時的工具姿態變化速度的上限值時，請啟用SpeedRLimitation。啟用SpeedRLimitation時，CP動作時的工具姿態變化速度超過設定的SpeedR時，工具姿態變化速度會變為SpeedR來限制動作速度。工具姿態變化速度超過設定的SpeedR時，以設定的SpeedS進行動作。工具姿態變化速度的上限值，請事先在SpeedR設定。

使用ECP時，工件會在與指定ECP編號因應的外部控制點上直線移動。此時的尖端關節中心不會沿著直線。



若要讓機器人在尚未完成Move動作前減速並停止，使用者可使用Till修飾詞指定該條件。在此指定的條件為，使用Sw命令檢查其中一項輸入位元。使用者檢查輸入為ON/OFF，並按照指定的條件停止手臂。此功能類似於輸入條件成立後便停止Move這樣的插斷。如果在Move動作期間輸入條件未成立，手臂就會到達以目標坐標指定的位置。

可省略Till修飾詞。關於Till修飾詞的詳細內容，請參閱Till命令。

注意

- 無法用Move進行的動作

  無法在執行動作之前確認動作範圍。因此，即便目標坐標位置在容許動作範圍之內，當來到此處的軌道通過容許動作範圍之外時，手臂就會突然停止，對伺服施加衝擊，而有發生障礙的危險。為防止發生這種情況，在高速執行Move前，請先以低速確認動作範圍。也就是，即便目標坐標在手臂動作範圍之內，以Move動作來到此處的軌道位於手臂容許動作範圍之外時，便無法運作手臂。

- 連同CP一起使用Move

  若使用CP參數，則在開始減速的同時，移至下一個陳述式控制動作命令。其便利之處在於，使用者可連接數個動作命令，以恆定的速度執行連續動作。在未指定CP的Move命令時，手臂一定會減速，並停於指定的目標坐標上。

- 向Move指示適當的速度和加減速度

  請注意，SpeedS和AccelS命令雖用於指定Move動作中的機械手速度和加減速度，但SpeedS和AccelS屬於對直線以及曲線動作的命令。Speed和Accel命令適用於PTP動作。

常見錯誤

- 欲執行直線移動距離為0的動作時

  Move若執行僅變更4自由度機器人(水平多關節型(包含RS系列)等)的U坐標值或6自由度機器人(垂直6軸型(包含N系列))的U、V、W坐標值的動作，則會發生錯誤。此時，請使用ROT參數。

- 超過關節限制速度的錯誤

  在指示動作中，只要有1個關節超過容許速度，就會發生超速錯誤。發生馬達超速錯誤時手臂即停止動作，並關閉馬達勵磁。

- RS系列執行了通過原點附近的動作時

  RS系列若用Move執行通過原點附近的動作，可能會發生超速錯誤。針對通過原點附近這樣的動作，請執行以下對策。
  - 請降低SpeedS的設定速度。
  - 請變更為不通過原點附近的路徑。
  - 請使用Go等PTP動作，以代替Move。

參照

AccelS, Arc, CP, Go, Jump, Jump3, Jump3CP, SpeedS, Sw, Till, SpeedR, SpeedRLimitation

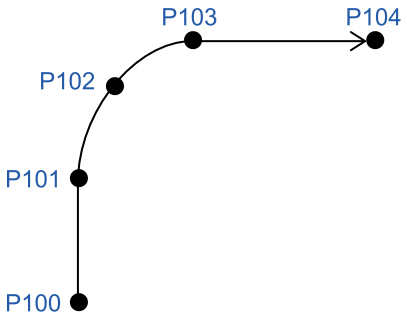
Move範例

以下範例是讓手臂執行P0~P1的PTP動作，然後再以直線返回到P0的簡易範例。在程式下半段輸入位元2變為ON之前，手臂朝P2直線移動。若在動作中輸入位元2變成ON，即便手臂尚未到達P2，也會減速並停止，然後執行以下程式命令。

```
Function movetest
  Home
  Go P0
  Go P1
  Move P0
  Move P2 Till Sw(2) = On
```

```
  If Sw(2) = On Then
    Print "Input #2 came on during the move and"
    Print "the robot stopped prior to arriving on"
    Print "point P2."
  Else
    Print "The move to P2 completed successfully."
    Print "Input #2 never came on during the move."
  EndIf
Fend
```

以下是連同CP一起使用Move的範例。下圖所示為從P100開始的圓弧軌道。P100~P101為直線移動，從P101開始進行圓弧移動，通過P102向P103繼續繪製圓弧。P103~P104呈直線動作，在此期間減速並停止。請注意，在P104停止之前，不會在中途點減速。在以下函式中編寫這種動作的程式。




```
Function CornerArc
   Go P100
   Move P101 CP        '不在P101停止
   Arc P102, P103 CP   '不在P103停止
   Move P104           '在P104減速並停止
Fend
```

3.16.20 MsgBox

顯示對話方塊的訊息，並等待使用者選擇按鈕。

格式

MsgBox 訊息$ [, 按鈕的種類] [, 標題$] [, Integer]

參數

訊息$

顯示的訊息

按鈕的種類

指定用於表示指定值(用於指定顯示的按鈕數和類型、圖示的樣式、按鈕的標題等)合計的數值或運算式。Epson RC+有事先為此參數確定的常數，並使用下表中的值。可省略。

常數	值	意義
MB_OK	0	只顯示[OK]按鈕
MB_OKCANCEL	1	顯示[OK]和[取消]按鈕
MB_ABORTRETRYIGNORE	2	顯示[停止][重試][忽略]按鈕
MB_YESNOCANCEL	3	顯示[是][否][取消]按鈕
MB_YESNO	4	顯示[是][否]按鈕
MB_RETRYCANCEL	5	顯示[重試]和[取消]按鈕
MB_ICONSTOP	16	Stop信號
MB_ICONQUESTION	32	「?」標記
MB_ICONEXCLAMATION	64	「!」標記
MB_DEFBUTTON1	0	第1個按鈕為預設按鈕
MB_DEFBUTTON2	256	第2個按鈕為預設按鈕

標題$

指定顯示於對話方塊標題列的內容。可省略。

接收結果的整數

指定用於接收表示用戶選擇值(整數)的變數。Epson RC+有事先為此參數確定的常數。下表所述為用此參數傳回的值。可省略。

常數	值	意義
IDOK	1	已選擇[OK]按鈕。
IDCANCEL	2	已選擇[取消]按鈕。
IDABORT	3	已選擇[停止]按鈕。
IDRETRY	4	已選擇[重試]按鈕。
IDYES	6	已選擇[是]按鈕。
IDNO	7	已選擇[否]按鈕。

說明

MsgBox用於自動對訊息進行格式化。欲設為空白狀態時，則對訊息使用CRLF。請參閱以下範例。

參照

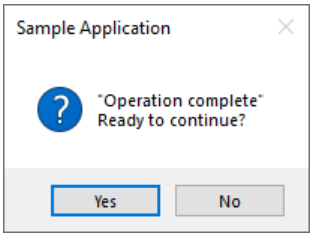
InputBox

MsgBox範例

以下示例是對使用者顯示是否繼續作業的確認訊息方塊。在訊息方塊內顯示[是]和[否]2個按鈕。也顯示「?」圖示。若使用者選擇按鈕而傳回MsgBox，則可考量如何回覆。若回覆[否]，便用Quit命令結束所有工作。

```
Function msgtest
  String msg$, title$
  Integer mFlags, answer

  msg$ = Chr$(34) + "完成作業" + Chr$(34) + CRLF
  msg$ = msg$ + "是否繼續？"
  title$ = "範例"
  mFlags = MB_YESNO + MB_ICONQUESTION
  MsgBox msg$, mFlags, title$, answer
  If answer = IDNO then
    Quit All
  EndIf
Fend
```

以下畫面所示為用上述代碼建立的訊息方塊。



限制事項

參數的訊息$以及標題$中包括半形逗號「,」時，無法正確顯示字串。因此請使用不含半形逗號的字串。

3.16.21 MyTask函數

用於傳回目前程式的工作編號。

格式

MyTask

傳回值

用於傳回目前工作的工作編號。工作編號的範圍為以下整數值。

- 一般工作: 1~32
- 背景工作: 65~80
- Trap工作: 257~267

說明

MyTask用於以數值傳回目前程式的工作編號。在程式上記述並執行MyTask函數時，則傳回執行程式的工作編號。

參照

Xqt

MyTask函數範例

下例所述為啟用或停用1~8的I/O連接埠的情形。

```
Function main
  Xqt 2, task      '執行工作2
  Xqt 3, task      '執行工作3
  Xqt 4, task      '執行工作4
  Xqt 5, task      '執行工作5
  Xqt 6, task      '執行工作6
  Xqt 7, task      '執行工作7
  Xqt 8, task      '執行工作8
  Call task
Fend

Function task
  Do
    On MyTask      '啟用與目前工作編號相同編號的I/O連接埠
    Off MyTask     '將與目前工作編號相同編號的I/O連接埠設為OFF
  Loop
Fend
```

3.17 N

3.17.1 Next

運用For和Next命令的組合建立迴圈。依照使用者指定次數，重複執行For和Next間的一系列命令。

格式

For 變數名稱1 = 初始值To結束值 [Step增量值] 陳述式 Next變數名稱1

參數

變數名稱1
　　用For/Next迴圈指定用於指派重複資料的計數器變數名稱。通常是定義為整數變數，但也可指定實數。
初始值
　　為已指定變數，以運算式或數值指定指派給迴圈開頭的數值。
結束值
　　以運算式或數值指定表示結束迴圈的數值。此值成立時，即結束For/Next迴圈，並執行Next命令的下一個陳述式。
增量值
　　以運算式或數值指定每1次For/Next迴圈的變數增量值。此值無論是正值或負值皆可指定。但指定負值時，請設為初始值＞結束值。若省略此參數，便自動將「1」視為增量值而進行增減。可省略。
陳述式
　　記述用於插入For/Next迴圈的SPEL+陳述式。

說明

For...Next用於僅以指定的次數重複迴圈中的一系列陳述式。迴圈以For陳述式開始，以Next陳述式結束。用變數計算在迴圈內重複執行陳述式的次數。

初始值為計數器的最初值。結束值和增量值的設定只要正確，就不限定為正值或負值。

結束值為計數器的最終值。到達此值後便結束For/Next迴圈，程式控制轉移到Next命令的下一個命令。

程式執行For陳述式的下一個陳述式，直到達成Next命令。僅以增量值所設定的值，增減計數器變數。未設定增量值時，則以「1」進行增減。

接著，將計數器變數值與結束值進行比較。若計數器變數值少於或同於結束值，則返回執行For陳述式的下一個命令。計數器變數值若大於結束值，便執行For/Next迴圈的下一個命令，並繼續執行Next命令的下一個命令。

注意

- 負值的增量值

  增量值為負值時，每執行一次迴圈就減少計數器的變數值。此時，請務必讓初始值大於結束值。

參照

For

For/Next範例

```
Function fornext
    Integer ctr
    For ctr = 1 to 10
        Go Pctr
    Next ctr
    '
    For ctr = 10 to 1 Step -1
        Go Pctr
```

```
    Next ctr
Fend
```

3.17.2 Not運算子

以位元為單位獲取數值的補數。

格式

Not 運算元

參數

運算元

指定整數值。

結果

傳回運算元1的補數。

說明

Not函數用於以位元為單位獲取運算元的補數。結果的位元用於反轉相應運算元位元。將0的位元轉換為1，將1的位元轉換為0。

參照

Abs、And、Atan、Atan2、Cos、Int、LShift、Mod、Or、RShift、Sgn、Sin、Sqr、Str$、Tan、Val、Xor

Not範例

這是在命令視窗中使用Not的範例。

```
>print not(1)
 -2
>
```

3.18 O

3.18.1 Off

停用以位元編號指定的輸出位元，或在按指定時間停用後再啟用。

格式

Off { 輸出位元編號輸出標籤 } [,時間] [,非同步指定] [,Forced]

參數

輸出位元編號
　以整數值指定要設為OFF的I/O輸出位元。
輸出標籤
　指定輸出標籤。
時間
　以秒指定輸出位元為OFF的時間。已過指定時間後，輸出位元再次變為ON。請以0.01秒~10秒的範圍指定OFF時間。可省略。
非同步指定
　一旦設定計時器，則用非同步指定，指定執行下一個命令的時序。可省略。
- 0 - 在停用輸出位元的同時執行下一個命令。
- 1 - 預設值。按指定時間停用後再啟用，並執行下一個命令。

Forced
　可省略。通常會省略。

說明

Off用於停用指定輸出位元(或返回0)。

一旦指定時間參數，按指定時間停用後再啟用以輸出位元編號指定的輸出位元。若在停用前輸出位元已為停用狀態，則經過指定時間後再啟用。

對指定時間進行如下設定時，啟用非同步指定設定。

- 1：停用輸出位元，經過指定時間後重新啟用，然後執行下一個命令。(這是非同步指定的預設值。若省略此參數，設定即變成「1」。)
- 0：在停用輸出位元的同時，執行下一個命令。

注意

- 被分配為遠端控制的輸出位元

  一旦指定為了系統輸出而設定的輸出位元，即發生錯誤。依照系統狀態自動啟用或停用遠端輸出位元。

- 發生緊急停止時

  在Epson RC+中，一旦發生緊急停止，便停用所有輸出位元。即便是緊急停止仍想維持設定時，可從[設定]選單顯示[控制器]畫面，再以其中的[設定]面板重新進行設定。

- Forced旗標

  在緊急停止時或打開安全門而將I/O輸出設為OFF時，從NoPause工作、NoEmgAbort工作(執行Xqt時，指定NoPause或NoEmgAbort開始的特別工作)指定此旗標。

  在緊急停止時或打開安全門時，I/O輸出會發生變化，因此在系統設計上需加以注意。

參照

In、InBCD、MemOn、MemOff、MemOut、MemSw、OpBCD、Oport、Out、Wait

Off範例

在以下範例中，啟動以「iotask」為名的主工作。「iotask」是用於分別啟用或停用輸出位元1和2，並於10秒後重新執行的簡易工作。

```
Function main
  Xqt 2, iotask
  Go P1
  .
  .
  .
Fend


Function iotask
  Do
    On 1
    On 2
    Off 1
    Off 2
    Wait 10
  Loop
Fend
```

以下是命令視窗中的簡易操作範例。

```
> on 1
> off 1, 10     '停用輸出1，經過10秒後再啟用
> on 2
> off 2
```

3.18.2 OLAccel

用於設定與過載率相應的加減速度自動調整。

格式

OLAccel {On | Off}

參數

On | Off

- On：啟用與過載率相應的加減速度自動調整。
- Off：解除與過載率相應的加減速度自動調整。

說明

OLAccel用於設定是否啟用與機器人過載率(OLRate)相應的加減速度自動調整功能。若開啟OLAccel，在執行PTP動作命令時，則對與當時的機器人過載率相應的加減速度進行自動調整。也就是說，其作用在於：執行PTP動作時，若機器人的過載率超過一定值，會自動降低動作時的加減速度，以防發生過載錯誤。以往對於發生過載錯誤的高Duty動作，需由客戶以程式設計讓機器人停止，或調節速度或加速度，但可透過使用OLAccel，來減少其必要性。不過，並非所有週期都不會發生過載錯誤，對於以非常高的Duty進行的大負載週期，無法完全迴避過載錯誤。此時，必須由客戶停止機器人或調節速度和加減速度。此外，有可能因使用環境而在不發生過載錯誤之狀態下持續運作機器人，從而造成馬達升溫並發生過熱錯誤。

在適當的負載狀態中，不會用到此命令。

測試週期時，可使用OLRate檢查機器人是否處於容易發生過載錯誤的狀態。

在以下任一情況下，OLAccel值會被初始化。

- 啟動控制器時
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

注意

對於不支援加減速度自動調整功能的機器人，若執行OLAccel On，則發生錯誤。

參照

OLAccel函數、OLRate

OLAccel範例

```
>olaccel on
>olaccel
OLACCEL is ON

Function main
Motor On
Power High
Speed 100
Accel 100, 100
OLAccel On
Xqt 2, MonitorOLRate
Do
Jump P0
```

```
Jump P1
Loop
Fend

Function MonitorOLRate
Do
'顯示OLRate
OLRate
Wait 1
Loop
Fend
```

3.18.3 OLAccel函數

用於傳回與過載率相應的加減速度之自動調整狀態。

格式

OLAccel

傳回值

- Off = 解除與過載率相應的加減速度之自動調整
- On = 將與過載率相應的加減速度之自動調整設為啟用

參照

OLAccel、OLRate

OLAccel函數範例

```
If OLAccel = Off Then
  Print "OLAccel is off"
EndIf
```

3.18.4 OLRate

用於顯示指定關節的過載率。

格式

OLRate [關節編號]

參數

關節編號

以1~9的整數進行指定。附加軸的S軸為8，T軸為9。

說明

OLRate是用於顯示指定關節過載率的函數。OLRate用於檢查週期是否對關節造成過度負擔。若以過載的週期進行使用，溫度或電流有可能成為伺服錯誤的原因。使用OLRate檢查機器人是否處於容易發生伺服錯誤的狀態。

在週期運轉中，執行其它工作以監視OLRate。如有OLRate超過1.0的關節，將發生伺服錯誤。

過載時，最容易發生伺服錯誤。可在測試週期時使用OLRate確認速度和加減速度設定，採取預防措施，以免在實際使用時發生伺服錯誤。

請在機器人動作時執行OLRate，以獲取有效值。

在適當的負載狀態中，不會用到此命令。

參照

OLRate函數

OLRate範例

```
>olrate
0.10000  0.20000
0.30000  0.40000
0.50000  0.60000

Function main
  Power High
  Speed 50
  Accel 50, 50
  Xqt 2, MonitorOLRate
  Do
    Jump P0
    Jump P1
  Loop
Fend

Function MonitorOLRate
  Do
    OLRate    '顯示OLRate
    Wait 1
  Loop
Fend
```

3.18.5 OLRate函數

用於傳回指定關節的過載率。

格式

OLRate (關節編號)

參數

關節編號

以1~9的整數進行指定。附加軸的S軸為8，T軸為9。

傳回值

用於傳回指定關節的過載率。傳回值為0.0~2.0。

說明

OLRate用於檢查週期是否對關節造成過度負擔。若以過載的週期進行使用，溫度或電流有可能成為伺服錯誤的原因。使用OLRate檢查機器人是否處於容易發生伺服錯誤的狀態。

在週期運轉中，執行其它工作以監視OLRate。如有OLRate超過1.0的關節，將發生伺服錯誤。

過載時，最容易發生伺服錯誤。可在測試週期時使用OLRate確認速度和加減速度設定，採取預防措施，以免在實際使用時發生伺服錯誤。

請在機器人動作時執行OLRate，以獲取有效值。

在適當的負載狀態中，不會用到此命令。

參照

OLRate

OLRate函數範例

```
Function main
  Power High
  Speed 50
  Accel 50, 50
  Xqt 2, MonitorOLRate
  Do
    Jump P0
    Jump P1
  Loop
Fend

Function MonitorOLRate
  Integer i
  Real olRates(4)
  Do
    For i = 1 to 4
      olRates(i) = OLRate(i)
      If olRate(i) > .5 Then
        Print "Warning: OLRate(", i, ") is over .5"
      EndIf
    Next i
  Loop
Fend
```

3.18.6 On

啟用指定的輸出位元，經過指定時間後停用。

格式

On { 輸出位元編號|輸出標籤 } [,時間] [,非同步指定] [,Forced]

參數

輸出位元編號
　以整數指定要啟用的I/O輸出位元。
輸出標籤
　指定輸出標籤。
時間
　以秒指定已指定的輸出位元為啟用狀態的時間。經過此時間後，將輸出位元設為OFF。(請將經過時間指定為0.01秒以上。)可省略。
非同步指定
　若執行時間設定，則可透過非同步指定，指定執行下一個命令的時序。可省略。
- 0 -在啟用輸出位元的同時，執行下一個命令。此時，時間最長可設為10秒。
- 1 -預設值。按指定時間啟用後再停用，並執行下一個命令。

Forced
　可省略。通常會省略。

說明

On用於啟用指定的輸出位元(設為1)。若有設定時間參數，則啟用指定的輸出位元，並在經過指定時間後停用。

設定時間時，使用下述的非同步指定參數的設定。

- 1：啟用輸出位元，經過指定時間後停用，然後執行下一個命令。(這是非同步指定的預設值。若省略此參數，設定即變成「1」。)
- 0：在啟用輸出位元的同時，執行下一個命令。

注意

- 設為遠端的輸出位元

  一旦指定設為遠端的輸出位元，即發生錯誤。依照系統狀態自動啟用或停用遠端輸出位元。關於遠端的詳細內容，請參閱以下手冊。
  「Epson RC+ 使用者指南」

  若要將遠端連接器的各位元設為輸出或I/O，則在Epson RC+的[設定]選單-[系統設定]的[遠端]對話方塊中進行設定。

- 發生緊急停止時

  在機器人控制器中，一旦發生緊急停止，便停用所有輸出位元。即便是緊急停止仍想維持設定時，可從[設定]選單顯示[控制器]畫面，再以其中的[環境]面板重新進行設定。

- Forced旗標

  在緊急停止時或打開安全門時，若要在NoPause工作、NoEmgAbort工作(執行Xqt時，指定NoPause或NoEmgAbort開始的特別工作)以及背景工作上啟用I/O輸出，則指定此旗標。

  在緊急停止時或打開安全門時，I/O輸出會發生變化，因此在系統設計上需加以注意。

參照

In、InBCD、MemOff、MemOn、Off、OpBCD、Oport、Out、Wait

On範例

在以下範例中，啟動以「iotask」為名的主工作。「iotask」是用於分別啟用或停用輸出位元1和2，並於10秒後重新執行的簡易工作。

```
Function main
    Xqt iotask
    Go P1
    .
    .
    .
Fend


Function iotask
    Do
        On 1
        On 2
        Off 1
        Off 2
        Wait 10
    Loop
Fend
```

以下是命令視窗中的簡易操作範例。

```
> on 1
> off 1, 10     '停用輸出1，10秒後再啟用
> on 2
> off 2
```

3.18.7 OnErr

設定在發生錯誤時使錯誤處理副程式對控制進行分支的插斷。使用者能進行錯誤處理。

格式

OnErr GoTo {標籤|0}

參數

標籤

指定發生錯誤時的轉移目的地之陳述式標籤。

0

指定清除OnErr設定的參數。

說明

使用者可以OnErr處理錯誤。若未使用OnErr，發生錯誤時則終止工作，並顯示錯誤。然而，一旦使用OnErr，便可將控制移至錯誤處理副程式，以自動從錯誤中恢復。錯誤恢復後，控制則移至以EResume命令指定的返回目的地。這樣的話，即便發生錯誤，也可在不中斷執行工作之狀態下自動處理錯誤。此外，即便是容易變得複雜的問題，也始終以相同的方法進行自動處理，因此，可更順暢地運作工作單元。

若在OnErr命令中指定參數「0」進行設定，則清除目前的OnErr設定。(執行OnErr 0後，若發生錯誤，則停止執行程式。)

參照

Err、EResume

OnErr範例

以下是簡易公用程式的範例，用於檢查是否存在點資料P0-P399。沒有點資料時，畫面上會顯示「此點不存在」訊息。在此程式中，利用CX命令測試有無定義點。若未定義，控制則移至錯誤處理副程式，並在畫面上顯示未定義的點。

```
Function errDemo
    Integer i, errNum

    OnErr GoTo errHandler

    For i = 0 To 399
        temp = CX(P(i))
    Next i
    Exit Function
    '
    '
    '*******************************************
    '* Error Handler                           *
    '*******************************************
errHandler:
    errNum = Err
    '確認有無使用未定義的點
    If errNum = 7007 Then
       Print "Point number P", i, " is undefined!"
    Else
       Print "ERROR: Error number ", errNum, " occurred while"
       Print "       trying to process point P", i, " !"
    EndIf
    EResume Next
Fend
```

3.18.8 OpBCD

以二進碼十進數(BCD)同時設定8位元的輸出。

格式

OpBCD 連接埠編號, 輸出資料 [, Forced]

參數

連接埠編號

指定I/O的輸出位元組。如下所述，各指定數值與輸出位元對應。

連接埠編號	輸出位元
0	0-7
1	8-15
2	16-23
3	24-31
...	...

輸出資料

以0~99的整數值指定以連接埠編號指定的輸出群組的輸出模式。第2個位(1的位數)表示已選擇群組當中的輸出低階4位元；第1個位(10的位數)表示輸出高階4位元。

Forced

可省略。通常會省略。

說明

OpBCD用於以BCD同時設定8個輸出位元。標準以及擴充輸出位元是以各8個為1組建立群組的。以OpBCD命令的連接埠編號參數指定使用的群組(具體的8個輸出位元)。例如，連接埠編號 = 0時，指定輸出位元0~7；連接埠編號 = 1時，指定輸出位元8~15。

選擇連接埠編號(8個輸出位元)之後，以輸出資料參數來定義特定輸出模式。輸出資料參數為1或2位數的值，有效範圍為0~99。第1個位(10的位數)表示以連接埠編號選擇的8個輸出位元當中的輸出高階4位元；第2個位(1的位數)表示以連接埠編號選擇的8個輸出位元當中的輸出低階4位元。

各位數的BCD有效值為0~9，因此無法實現所有I/O組合。下表所示為，連接埠編號=0時的I/O的組合範例和與此對應的輸出編號值。

輸出設定(輸出編號)

輸出編號	7	6	5	4	3	2	1	0
01	Off	Off	Off	Off	Off	Off	Off	On
02	Off	Off	Off	Off	Off	Off	On	Off
03	Off	Off	Off	Off	Off	Off	On	On
08	Off	Off	Off	Off	On	Off	Off	Off
09	Off	Off	Off	Off	On	Off	Off	On
10	Off	Off	Off	On	Off	Off	Off	Off
11	Off	Off	Off	On	Off	Off	Off	On
99	On	Off	Off	On	On	Off	Off	On

BCD只能用於指定十進位值。因此，對於使用BCD的OpBCD命令，無法啟用所有輸出位元。各位數的輸出編號最大值皆為「9」，而OpBCD可使用的最大值為「99」。從上表可看出，輸出編號值為「99」時，無法啟用所有輸出。在輸出編號值為「99」地情況下，0、3、4、7的輸出狀態為ON，其它皆為OFF。

注意

- OpBCD和Out的差異

  如下所述為SPEL+的Out和OpBCD的最大差異之處。

  - 在OpBCD命令中，以二進位編碼十進位(BCD)格式指定用於啟用或停用的8個輸出位元。二進碼十進數格式不可使用&HA、&HB、&HC、&HD、&HE、&HF等值，因此無法符合8個輸出位元的所有組合。
  - Out命令將0~255的值用作啟用或停用輸出的8位元值。(OpBCD：0~99)這樣的話，便符合所有8位元輸出群組的組合，可配合使用者的規格進行指定。

- 設為遠端的輸出位元

  對於遠端設定的輸出位元，若指定啟用OpBCD，則發生錯誤。依照系統狀態自動啟用或停用遠端輸出位元。關於遠端的詳細內容，請參閱以下手冊。
  「Epson RC+ 使用者指南」

  可從[設定]選單中顯示[系統設定]對話方塊，然後在[遠端]索引標籤上設定遠端連接器的各位元設定(設為遠端或I/O等)。

- 緊急停止時的輸出

  機器人控制器具有緊急停止時停用所有輸出的功能。若要啟用或停用此功能，則從[設定]選單中顯示[控制器]對話方塊，再透過[環境]面板設定選項按鈕。

- Forced旗標

  在緊急停止時或打開安全門時，若要在NoPause工作、NoEmgAbort工作(執行Xqt時，指定NoPause或NoEmgAbort開始的特別工作)以及背景工作上啟用I/O輸出，指定此旗標。

  在緊急停止時或打開安全門時，I/O輸出會發生變化，因此在系統設計上需加以注意。

參照

In、InBCD、MemOff、MemOn、MemSw、Off、On、Oport、Out、Sw、Wait

OpBCD範例

以下是用於啟動以「iotask」為名的主工作之程式。「iotask」是啟用輸出位元1和2之後，再啟用輸出位元0和3的簡易工作。一旦啟用輸出位元1和2，便停用輸出位元0和3。此外，若停用前者，則啟用後者。

```
Function main
    Xqt 2, iotask
    Go P1
    .
    .
    .
Fend

Function iotask
    Do
        OpBCD 0, 6
        OpBCD 0, 9
        Wait 10
    Loop
Fend
```

以下是命令視窗中的簡易操作範例。

```
> OpBCD 1,6     '啟用輸出1和2
> OpBCD 2,1     '啟用輸出8
> OpBCD 3, 91   '啟用輸出24、28、31
```

3.18.9 OpenDB

開啟資料庫、Excel活頁簿。

格式

OpenDB #資料庫編號, 資料庫類別 [, SQL伺服器名稱],資料庫名稱

參數

資料庫編號

用於指定501~508的整數。

資料庫類型

從[SQL]、[Access]、[Excel]中選擇要開啟的資料庫類型。

SQL伺服器名稱

若將[SQL]指定為資料庫類型，則指定SQL伺服器名稱。若省略(LOCAL)則表示已指定伺服器。不可指定網路上的SQL伺服器。

若將[Access]或[Excel]指定為資料庫類別，則不指定SQL伺服器名稱。

資料庫名稱

- 若將[SQL]指定為資料庫類別，則指定SQL伺服器上的資料庫名稱。
- 若有指定[Access]，則指定Access檔名。若省略Access檔名路徑，則搜尋目前資料夾。詳細內容請參閱ChDisk。
- 若有指定[Excel]，則指定Excel檔名。Excel檔案可指定的格式為Excel 2007活頁簿、Excel 97-2003活頁簿檔案。若省略Excel檔名路徑，則搜尋目前資料夾。詳細內容請參閱ChDisk。

說明

以指定的編號開啟指定的資料庫。

指定的資料庫必須存在於裝有RC+的PC磁碟上。不存在時則發生錯誤。在資料庫處於開啟狀態時，指定的檔案編號用於識別該資料庫。在以CloseDB命令關閉資料庫之前，不可用於瀏覽其它資料庫。在資料庫操作命令(SelectDB, Print#, Input#, CloseDB)中使用檔案編號。

不可使用Microsoft office 2010 64位元版本的Access和Excel檔案。

注意

需連接有安裝RC+的PC。

參照

SelectDB、CloseDB、UpdateDB、DeleteDB、Input #、Print #

OpenDB範例

使用SQL資料庫的範例

以下所示為在使用SQL伺服器2000範例資料庫Northwind的情況下從表中載入資料的簡易範例。

```
Integer count, i, eid
String Lastname$, Firstname$, Title$

OpenDB #501, SQL, "(LOCAL)", "Northwind"
count = SelectDB(#501, "Employees")
For i = 0 To count - 1
  Input #501, eid, Lastname$, Firstname$, Title$
  Print eid, ",", Lastname$, ",", Firstname$, ",", Title$
Next
CloseDB #501
```

使用Access資料庫的範例 以下所示為在使用Microsoft Access 2007範例資料庫學生名冊的情況下從表中載入資料的簡易範例。

```
Integer count, i, eid
String Lastname$, Firstname$, dummy$

OpenDB #502, Access, "c:\MyDataBase\學生名冊.accdb"
count = SelectDB(#502, "學生")
For i = 0 To count - 1
  Input #502, eid, dummy$, dummy$, Lastname$, dummy$, Firstname$
  Print eid, ",", Lastname$, ",", Firstname$
Next
CloseDB #502
```

使用Excel活頁簿的範例 以下所示為在使用Microsoft Excel活頁簿學生名冊的情況下從工作表中載入資料的簡易範例。

```
Integer count, i, eid
String Lastname$, Firstname$

OpenDB #503, Excel, "c:\MyDataBase\學生名冊.xls"
count = SelectDB(#503, "[學生$]")
For i = 0 To count - 1
  Input #503, eid, Lastname$, Firstname$
  Print eid, ",", Lastname$, ",", Firstname$
Next
CloseDB #503
```

3.18.10 OpenCom

用於開啟RS-232C連接埠。

格式

OpenCom #連接埠編號

參數

連接埠編號
　　以整數值指定要開啟的RS-232C連接埠編號。
SPEL+控制部分
　　1~4
PC部分
　　1001~1008

說明

可在工作上啟用RS-232C連接埠。

若要使用SPEL+控制部分的連接埠，控制器上需安裝I/O板(選項)。

若要使用PC部分的連接埠，需設定RC+。有關詳細資訊，請參閱以下手冊的RS-232C相關說明。
「Epson RC+ 使用者指南 - [設置]功能表」

參照

ChkCom、CloseCom、SetCom

OpenCom範例

```
Integer PortNo

PortNo = 1001
OpenCom #PortNo
Print #PortNo, "Data from COM1"
CloseCom #PortNo
```

3.18.11 OpenCom函數

用於取得實施OpenCom的工作編號。

格式

OpenCom (連接埠編號)

參數

連接埠編號
　　以整數值指定RS-232C的連接埠編號。
SPEL+控制部分
　　1~4
PC部分
　　1001~1008

說明

用於取得實施OpenCom的工作編號。

參照

ChkCom、CloseCom、OpenCom、SetCom

OpenCom數範例

```
Print OpenCom(PortNo)
```

3.18.12 OpenNet

開啟TCP/IP網路連接埠。

格式

OpenNet #連接埠編號 As { Client | Server }

參數

連接埠編號

指定要開啟的TCP/IP連接埠編號的整數值。連接埠編號的範圍為201~216。

說明

OpenNet用於開啟TCP/IP連接埠，以便與網路上的其它電腦進行通訊。

將1個系統作為伺服器、將其它系統作為用戶端而開啟。並未限定先啟動哪一個系統。

參照

ChkNet、CloseNet、SetNet

OpenNet範例

以下為2個控制器的TCP/IP設定範例。

Controller #1:
Port: #201
Host Name: 192.168.0.2
TCP/IP Port: 1000

```
Function tcpip
  OpenNet #201 As Server
  WaitNet #201
  Print #201, "Data from host 1"
Fend
```

Controller #2:
Port: #201
Host Name: 192.168.0.1
TCP/IP Port: 1000

```
Function tcpip
  String data$
  OpenNet #201 As Client
  WaitNet #201
  Input #201, data$
  Print "received '", data$, "' from host 1"
Fend
```

3.18.13 OpenNet函數

用於取得實施OpenNet的工作編號。

格式

OpenNet (連接埠編號)

參數

連接埠編號

指定TCP/IP連接埠編號的整數值。連接埠編號的範圍為201~216。

說明

用於取得實施OpenNet的工作編號。

參照

ChkNet、CloseNet、OpenNet、SetNet

OpenNet函數範例

```
Print OpenNet(PortNo)
```

3.18.14 Oport函數

本函數用於傳回指定輸出位元狀態。

格式

Oport (輸出位元編號)

參數

輸出位元編號
　　指定I/O的輸出位元。

傳回值

以0或1的整數傳回I/O的指定輸出位元狀態。

- 0：OFF
- 1：ON

說明

Oport用於檢查輸出位元的狀態。其作用與對輸入位元發揮功能的Sw命令類似。Oport最常用於供料器、輸送帶及夾持機構等。此外，也用於透過I/O，檢查連接於發揮功能的其它裝置主機的輸出位元狀態。以1或0的值傳回透過Oport函數檢查的狀態。這些值用於表示指定的輸出位元是ON或OFF。

注意

- Oport和Sw的差異

  Oport和Sw命令之間存在重要差異之處。兩者皆用於檢查I/O狀態，但檢查I/O的類型不同。Sw命令用於檢查輸入位元。Oport命令用於檢查標準以及擴充硬體的輸出位元。這些硬體連接埠是與控制器外部裝置進行通訊的獨立輸出位元。

參照

In、InBCD、MemIn、MemOff、MemOn、MemOut、MemSw、Off、On、OpBCD、Out、Sw、Wait

Oport函數範例

以下是啟用輸出位元5，在檢查是否啟用後繼續執行程式的範例。

```
Function main
    TMOut 10
    OnErr errchk
    Integer errnum
    On 5    '啟用輸出5
    Wait Oport(5)
    Call mkpart1
    Exit Function

errchk:
    errnum = Err(0)
    If errnum = 94 Then
       Print "TIME Out Error Occurred during period"
       Print "waiting for Oport to come on. Check"
       Print "Output #5 for proper operation. Then"
       Print "restart this program."
    Else
       Print "ERROR number ", errnum, "Occurred"
       Print "Program stopped due to errors!"
    EndIf
    Exit Function
Fend
```

這是命令視窗中的簡易操作範例。

```
> On 1
> Print Oport(1)
1
> Off 1
> Print Oport(1)
0
>
```

3.18.15 Or運算子

以位元為單位或以邏輯對2個值進行Or運算。

格式

值1 Or 值2

參數

值1
　　指定整數值或Boolean值。
值2
　　指定整數值或Boolean值。

結果

若以整數值指定，則傳回以位元為單位傳回對值進行Or運算的結果。若以Boolean值指定，則傳回邏輯Or運算的結果。

說明

若以整數值指定，Or運算子則用於以位元為單位對運算元進行Or運算。運算結果是對2個運算元的各位元進行Or運算的值。

以Boolean值指定時，若任一值為真(True)，運算結果則為真(True)。

參照

And、LShift、Mod、Not、RShift、Xor

Or運算子範例

以下是以位元為單位的Or運算子的範例。

```
>print 1 Or 2
 3
>
```

以下是邏輯Or運算子的範例。

```
If a = 1 Or b = 2 Then
c = 3
EndIf
```

3.18.16 Out

用於同時設定(輸出)8個輸出位元。

格式

Out 連接埠編號, 輸出資料 [, Forced]

參數

連接埠編號

指定I/O輸出位元組。如下所述，指定數值與輸出位元對應。

連接埠編號	輸出位元
0	0-7
1	8-15
...	...

輸出資料

以0~255的整數值指定以連接埠編號指定的輸出群組之輸出模式。若以十六進位顯示，則以&H0~&HFF為其範圍。低階位表示低階位元(或前4個輸出位元)；高階位表示高階位元(或後4個輸出位元)。

Forced

可省略。通常會省略。

說明

Out的作用在於，透過將連接埠編號和輸出資料進行組合，同時設定8個輸出位元。以連接埠編號參數指定使用的群組(8個輸出位元)。例如，連接埠編號 = 0 時，指定輸出位元0~7；連接埠編號 = 1時，指定輸出位元8~15。

首先，以連接埠編號指定8個輸出位元之後，再利用輸出資料參數來指定特定輸出模式。輸出資料參數可獲得的值為0~255，以十六進位或十進位的整數進行指定。(&H0~&HFF或0~255)

下表所示為，分為連接埠編號為「0」以及「1」兩種情況，在顯示與其相應的輸出資料值的同時，顯示部分I/O組合範例的情況。

連接埠編號=0時的輸出設定(輸出位元編號)

輸出資料值	7	6	5	4	3	2	1	0
01	Off	Off	Off	Off	Off	Off	Off	On
02	Off	Off	Off	Off	Off	Off	On	Off
03	Off	Off	Off	Off	Off	Off	On	On
08	Off	Off	Off	Off	On	Off	Off	Off
09	Off	Off	Off	Off	On	Off	Off	On
10	Off	Off	Off	Off	On	Off	On	Off
11	Off	Off	Off	Off	On	Off	On	On
99	Off	On	On	Off	Off	Off	On	On
255	On	On	On	On	On	On	On	On

連接埠編號=1時的輸出設定(輸出位元編號)

輸出資料值	15	14	13	12	11	10	9	8
01	Off	Off	Off	Off	Off	Off	Off	On
02	Off	Off	Off	Off	Off	Off	On	Off
03	Off	Off	Off	Off	Off	Off	On	On
08	Off	Off	Off	Off	On	Off	Off	Off
09	Off	Off	Off	Off	On	Off	Off	On
10	Off	Off	Off	Off	On	Off	On	Off
11	Off	Off	Off	Off	On	Off	On	On
99	Off	On	On	Off	Off	Off	On	On
255	On	On	On	On	On	On	On	On

注意

- OpBCD和Out的差異

  如下所述為SPEL+的Out和OpBCD的最大差異之處。

  - 在OpBCD命令中，以二進位編碼十進位(BCD)格式指定用於啟用或停用的8個輸出位元。二進碼十進數格式不可使用&HA、&HB、&HC、&HD、&HE、&HF等值，因此無法符合8個輸出位元的所有組合。
  - Out命令將0~255的值用作啟用或停用輸出的8位元值(OpBCD時為0~99)。這樣的話，便符合所有8位元輸出群組的組合，可配合使用者的規格進行指定。

- Forced旗標

  在緊急停止時或打開安全門時，若要在NoPause工作、NoEmgAbort工作(執行Xqt時，指定NoPause或NoEmgAbort開始的特別工作)以及背景工作上啟用I/O輸出，指定此旗標。

  在緊急停止時或打開安全門時，I/O輸出會發生變化，因此在系統設計上需加以注意。

參照

In、InBCD、MemOff、MemOn、MemOut、MemSw、Off、On、Oport、Sw、Wait

Out範例

以下是用於啟動以「iotask」為名的主工作之程式。「iotask」是啟用或停用輸出位元0~3的簡易工作。若使用Out命令，則可以1個命令執行工作，不必分別啟用或停用輸出位元。

```
Function main

  Xqt iotask
  Do
    Go P1
    Go P2
  Loop
Fend

Function iotask

  Do
    Out 0, &H0F
    Out 0, &H00
    Wait 10
```

```
  Loop
Fend
```

這是命令視窗中的簡易操作範例。

```
> Out 1,6     '啟用輸出9和10
> Out 2,1     '啟用輸出8
> Out 3,91    '啟用輸出24、25、27、28、30
```

3.18.17 Out函數

用於以位元組為單位傳回輸出連接埠的狀態。

格式

Out (連接埠編號)

參數

連接埠編號

指定I/O輸出位元組。如下所述，指定數值與輸出位元對應。

連接埠編號	輸出位元
0	0-7
1	8-15
...	...

傳回值

用於以位元組為單位傳回指定輸出連接埠的狀態。

參照

Out

Out函數範例

```
Print Out(0)
```

3.18.18 OutReal

將實數值的輸出資料作為32位元浮點數資料(符合IEEE754)，設定輸出連接埠2字組(32位元)的狀態。

格式

OutReal 字組連接埠編號, 輸出資料 [,Forced]

參數

字組連接埠編號
　　指定I/O輸出字組。
輸出資料
　　以運算式或數值指定輸出資料(實體型實數)。
Forced
　　可省略。通常會省略。

說明

對以字組連接埠編號指定的輸出字組連接埠和下一個輸出字組連接埠輸出指定的IEEE754的Real值。字組連接埠編號參數可使用輸出字組標籤。

注意

- Forced旗標

  在緊急停止時或打開安全門而啟用I/O輸出時，從NoPause工作、NoEmgAbort工作(執行Xqt時，指定NoPause或NoEmgAbort開始的特別工作)指定此旗標。

  在緊急停止時或打開安全門時，I/O輸出會發生變化，因此在系統設計上需加以注意。

參照

In、InW、InBCD、InReal、Out、OutW、OpBCD、OutReal函數

OutReal範例

```
OutReal 32, 2.543
```

3.18.19 OutReal函數

用於作為32位元浮點數資料(符合IEEE754)取得輸出連接埠的狀態。

格式

OutReal (字組連接埠編號)

參數

字組連接埠編號
　　指定I/O輸出字組。

傳回值

用於以32位元浮點數資料(符合IEEE754)傳回指定輸出連接埠的狀態。

參照

In、InW、InBCD、InReal、Out、OutW、OpBCD、OutReal

OutReal函數範例

```
Real rdata01

rdata01 = OutReal(0)
```

3.18.20 OutW

以字組為單位，同時以16位元設定輸出連接埠的狀態。

格式

OutW 字組連接埠編號, 輸出資料 [, Forced]

參數

字組連接埠編號
　　指定I/O輸出字組。
輸出資料
　　以運算式或數值指定輸出資料(0~65535的整數)。
Forced
　　可省略。通常會省略。

說明

將以參數字組連接埠編號指定的使用者I/O的輸出連接埠群組的狀態變更為指定的輸出資料。

注意

- Forced旗標

  在緊急停止時或打開安全門時，若要在NoPause工作、NoEmgAbort工作(執行Xqt時，指定NoPause或NoEmgAbort開始的特別工作)以及背景工作上啟用I/O輸出，指定此旗標。

  在緊急停止時或打開安全門時，I/O輸出會發生變化，因此在系統設計上需加以注意。

參照

In、InW、Out

OutW範例

```
OutW 0, 25
```

3.18.21 OutW函數

用於以字組單位(2位元組)傳回輸出連接埠的狀態。

格式

OutW (字組連接埠編號)

參數

字組連接埠編號
　　指定I/O輸出字組。

傳回值

用於以16位元傳回指定輸出連接埠的狀態。

參照

OutW

OutW函數範例

```
OutW 0, &H1010
```

3.19 P

3.19.1 P# (1. 定義點)

用於定義點資料。

格式

P點編號 = 點指定

點標籤 = 點指定

參數

點編號

以數值或帶括號的運算式進行指定。

- P編號
- P(運算式)

點標籤

指定點標籤。

點指定

以下列點數據進行指定。
P點編號、點標籤、Here、Pallet、點資料函數(Here函數、XY函數、JA函數、Pulse函數等)
詳細內容請參閱「P# (2. 點指定)」。

說明

透過指派其它點或點運算式，對點資料進行定義。

參照

Local、Pallet、PDef、PDel、Plist

點的指派範例

以下是命令視窗中的操作範例。

在P1中設定坐標資料。

```
> P1 = XY(300,200,-50,100)
```

指定左手臂的姿態：

```
> P2 = XY(-400,200,-80,100)/L
```

將在P2的X坐標資料上加上20之後的值定義為P3。

```
> P3 = P2 +X(20)
> plist 3
P3 = XY(-380,200,-80,100)/L
```

從P2的Y坐標資料減去50，將Z坐標資料作為-30，然後將該值定義為P4，作為右手臂的姿態。

```
>P4=P2 -Y(50) :Z(-30) /R
> plist 4
P4 = XY(-450,200,-30,100)/R
```

將在Pallet (3, 5) 的U coord(U坐標值)上加上90的值定義為P6。

```
> P5 = Here
> P6 = pallet(3,5) +U(90)
```

3.19.2 P# (2. 點指定)

指定用於點指定和動作命令的點資料。

格式

point [{ +|- } 點指定] [本地編號] [手臂姿態] [臂肘姿態] [腕部姿態] [j4flag值] [j6flag值] [j1flag值] [j2flag值] [相對位移值] [絕對坐標]

參數

點指定

以如下任一項指定點數據。

- P編號
- P(運算式)
- 點標籤
- Pallet(棧板編號, 分割編號)
- 目前位置指定
- Here
- 點資料函數
- XY函數、JA函數、Pulse函數等

本地編號

指定本地編號(1~15)。請務必在編號前附加正斜線「/」或「@」標記。「/」表示坐標變成本地之意。「@」標記表示坐標轉換為本地坐標之意。可省略。

夾具(腕部)姿態

是水平多關節型(包含RS系列)機器人和垂直六軸(包含N系列)機器人的夾具(腕部)姿態。指定/L(左臂腕姿態)或/R(右臂腕姿態)。可省略。

臂肘姿態

是垂直六軸機器人(包含N系列)所需參數。/指定/A (ABOVE) 或/B (BELOW)。

腕部姿態

是垂直六軸機器人(包含N系列)所需參數。/指定/F (FLIP) 或/NF (NOFLIP)。

j4flag值

是垂直六軸機器人(包含N系列)所需參數。/指定/J4F0或/J4F1。

j6flag值

是垂直六軸機器人(包含N系列)所需參數。/指定/J6F0~/J6F127的值。

j1flag值

是RS系列和垂直六軸機器人(不包含N系列)所需參數。/指定/J1F0或/J1F1。

j2flag值

是RS系列所需參數。/指定/J2F0或/J2F1。

j1angle值

是RS系列和N系列所需參數。/指定/J1A(實數值)。

j4angle值

是N系列所需參數。/指定/J4A(實數值)。

相對偏移值

用於調整1個以上的相對坐標。可省略。

{+|-} {X|Y|Z|U|V|W|RZ|RY|RX|R|S|T|ST } (運算式)

TL偏移是目前工具坐標系的相對偏移。

{+|-} {TLX|TLY|TLZ|TLU|TLV|TLW} (運算式)

絕對坐標

指定1個以上的絕對坐標。請務必在坐標前附加冒號「:」。可省略本絕對坐標。

:{X|Y|Z|U|V|WR|S|T|ST } (運算式)

說明

在點的指派或動作命令中使用點運算式。(請參閱以下範例)

```
Go P1 + P2
P1 = P2 + XY(100, 100, 0, 0)
```

相對位移值的使用

可針對基點進行1個以上坐標的位移。以下範例是將機器人從目前位置朝X軸正方向移動20 mm的情形。

```
Go Here +X(20)
```

若再次執行此操作，因為相對移動，機器人會朝X軸方向再移動20 mm。

在垂直6軸型機器人(包含N系列)中，若要環繞坐標軸進行相對旋轉，則指定以下內容。以下範例是，以目前姿態為基準將機器人的工具姿態朝X軸正方向旋轉20°的情形。

```
Go Here +RX(20)
```

可執行相對工具位移。

```
Go Here +TLX(20) -TLY(5.5)
```

垂直6軸型機器人(包含N系列)在向透過相對位移獲得的點進行動作時，腕部有可能朝非預期的方向轉動。由於相對位移運算屬於不依賴於機器人機種的命令，因此在未進行所需姿態旗標的轉換之狀態下進行動作。LJM函數可用於防止這種腕部的非預期轉動。

```
Go LJM(Here +X(20))
```

絕對坐標的使用

可使用絕對坐標變更1個以上基點的坐標。以下範例是將機器人移至X軸20 mm位置的情形。

```
Go Here :X(20)
```

因為機器人的手臂已透過先前的移動而到達指定的絕對坐標位置。因此，即便再次執行，也不會移動。

相對位移值和絕對坐標可用於暫時修正點資料，十分方便。以下範例說明，透過指定Z的相對位移10 mm，移動抓取位置上方10 mm的部分，然後再慢慢移至抓取位置的情形。

```
Speed fast
Jump pick +Z(10)
Speed slow
Go pick
```

以下範例說明，透過指定第3關節的絕對值「0」，從目前位置進行垂直移動的情形。

```
LimZ 0
Jump Here :Z(0)
```

本地的使用

可使用「/」或「@」標記指定本地編號。「/」和「@」標記的功能各有不同。

若要以本地坐標系指定坐標，則使用「/」。在以下陳述式範例中，透過附加「/1」指定「P1位置，即本地1的0,0,0,0」。

```
P1 = XY(0, 0, 0, 0) /1
```

若要將坐標轉換為本地坐標，則使用「@」標記。在以下陳述式範例中，將表示目前位置的本地坐標值註冊於P1上。

```
P1 = Here @1
```

參照

Go、LJM、Local、Pallet、Pdel、Plist、Hand、Elbow、Wrist、J4Flag、J6Flag、J1Flag、J2Flag

點指定範例

以下範例是以指派陳述式或動作命令指定點的情形。

```
P1 = XY(300,200,-50,100)
P2 = P1 /R
P3 = pick /1
P4 = P5 + P6
P(i) = XY(100, 200, CZ(P100), 0)
Go P1 -X(20) :Z(-20) /R
Go Pallet(1, 1) -Y(25.5)
Move pick /R
Jump Here :Z(0)
Go Here :Z(-25.5)
Go JA(25, 0, -20, 180)
pick = XY(100, 100, -50, 0)

P1 = XY(300,200,-50,100, -90, 0)
P2 = P1 /F /B
P2 = P1 +TLV(25)
```

3.19.3 PAgl函數

根據指定的坐標值計算關節位置並進行傳回。

格式

PAgl (坐標值, 關節編號)

參數

坐標值

指定用於計算關節位置的坐標值。

關節編號

以運算式或數值指定關節編號(1~9的整數值)。附加軸的S軸為8，T軸為9。

傳回值

以實數值傳回計算的關節位置。旋轉關節(單位：deg)；直動關節(單位：mm)

參照

Agl、CX、CY、CZ、CU、CV、CW、CR、CS、CT、PPls

PAgl函數範例

```
Real joint1

joint1 = PAgl(P10, 1)
```

3.19.4 Pallet

用於定義棧板並顯示定義棧板。

格式

(1) Pallet [Outside,] [棧板編號, 點編號1, 點編號2, 點編號3 [, 點編號4], 分割數1, 分割數2]

(2) Pallet [Outside,] 棧板編號, 坐標系資料1, 坐標系資料2, 坐標系資料3 [, 坐標系資料4], 分割數1, 分割數2

(3) Pallet

參數

Outside
　　創建可對指定行及列的範圍之外進行存取的棧板。可省略。
棧板編號
　　以運算式或數值指定棧板編號(0~15的整數)。
點編號1~3
　　指定棧板定義(標準3點定義)所使用的點變數。
點編號4
　　要進行4點定義時，與點編號1~3一起使用。可省略。
分割數1
　　以整數指定棧板的點編號1(坐標系資料1)和點編號2(坐標系資料2)的分割數。範圍：1~32767。(分割數1×分割數2<32767)
分割數2
　　以整數指定棧板的點編號1(坐標系資料1)和點編號3(坐標系資料3)的分割數。範圍：1~32767。(分割數1×分割數2<32767)
坐標系資料1~3
　　直接以點數據指定棧板定義(標準3點定義)所使用的坐標系。
坐標系資料4
　　要進行4點定義時，與坐標系資料1~3一起使用。可省略。

結果

(3) 若省略參數，則顯示已定義的全部棧板。

說明

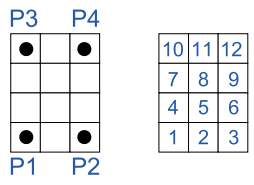
對機器人進行點編號1(坐標系資料1)、點編號2(坐標系資料2)、點編號3(坐標系資料3)這3點的最小限度之教導，並指定點編號1(坐標系資料1)和點編號2(坐標系資料2)的分割數以及點編號1(坐標系資料1)和點編號3(坐標系資料3)的分割數，以定義棧板。

若棧板呈高精度的方形，在邊角4點之中僅指定3點位置即可。但是，在大多情況下，建議指定所有邊角(4點)位置以定義棧板。

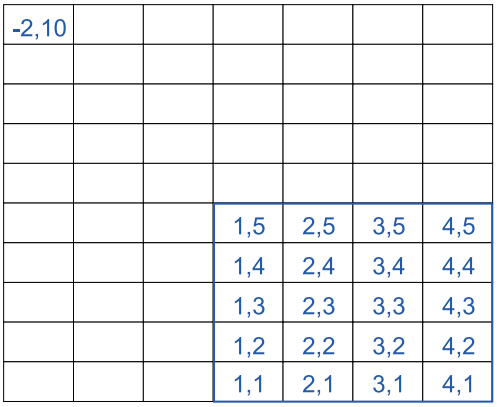
在定義棧板時，首先對邊角的3或4點進行教導，然後進行以下操作。

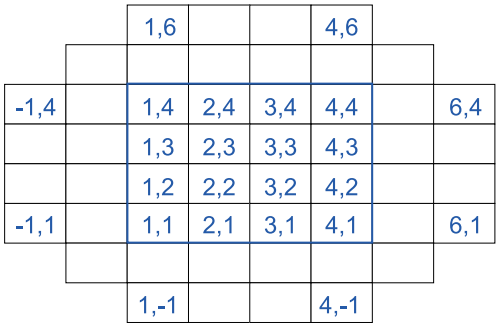
4點定義時：如下顯示P1、P2、P3以及P4。P1-P2之間有3點，P1-P3之間有4點。總共使用12點，以下列格式進行定義。

```
Pallet 1, P1, P2, P3, P4, 3, 4
```



10	11	12
7	8	9
4	5	6
1	2	3





會對表示棧板分割的各點自動分配割分割編號。在上圖情形時，則從P1開始。該分割編號也需要用於Pallet函數。

可以Outside指定建立可對行及列的範圍之外進行存取的棧板。

例：

```
Pallet Outside 1, P1, P2, P3, 4, 5
Jump Pallet(1, -2, 10)
```

注意

- 棧板尺寸的上限

  用於棧板定義的點數必須小於32767。(分割數1×分割數2＜32767)

- 不正確的棧板形狀定義

  若點的順序或點之間的分割數錯誤，則會定義不正確的棧板形狀。

- 棧板表面的定義

  以邊角3點的Z坐標值定義棧板平面的高度。因此，也可定義豎直擺放的棧板。

- 1列棧板的定義

  還可以3點指定的Pallet命令定義1列棧板。若為1列，則對兩端的2點進行教導，並按如下所述進行輸入和執行。將相同編號方向的分割數作為1。

  ```
  > Pallet 2, P20, P21, P20, 5, 1     '定義5&time;1棧板
  ```

- UVW坐標值

  以Pallet命令指定的3點(4點)UVW坐標值不同時，則使用點編號1以及坐標系資料1的UVW坐標值。

忽略點編號2~4以及坐標系資料2~4的UVW坐標值。

- 附加軸坐標值

  以Pallet命令指定的3點(4點)坐標值保持附加軸坐標值(ST軸值)時，即便是附加軸坐標值，也會被均等分割。總之，將附加軸作為行走軸使用時，在棧板定義之際，也會在考量行走軸的動作前提下進行計算。因此，可定義考量行走軸位置的、超過機器人動作範圍的大棧板。反之，即便定義不受棧板定義影響的附加軸，在定義棧板時，也需注意附加軸的位置。

參照

Pallet函數

Pallet範例

以下是在命令視窗中設定以P1、P2、P3定義的棧板之範例。在棧板表面上均等配置有15點，棧板點編號1~3排列在P1-P2之間。

```
> pallet 1, P1, P2, P3, 3, 5
> jump pallet(1, 2)          'Jump至棧板的指定位置
```

如下所示為透過該設定所建立的棧板。

P3

13	14	15
10	11	12
7	8	9
4	5	6
1	2	3

P1 P2

3.19.5 Pallet函數

本函數是用於參照棧板上位置的點資料函數。

格式

(1) Pallet (棧板編號, 棧板位置編號)

(2) Pallet (棧板編號, 分割橫坐標, 分割縱坐標)

參數

棧板編號
　　以數值指定棧板編號(0~15的整數)。
棧板位置編號
　　以運算式或數值(1~32767)指定分割點的指定編號(整數)。
分割橫坐標
　　以數值(-32768~32767)指定以棧板定義指定的橫坐標。
分割縱坐標
　　以數值(-32768~32767)指定以棧板定義指定的縱坐標。

說明

Pallet用於傳回事先以Pallet陳述式定義的棧板上之1點位置。可透過將此函數與動作命令(Go或Jump命令等)一起使用，將手臂移至棧板上的指定位置。

可以運算式或整數值定義棧板位置編號。

注意

- 垂直6軸型機器人(包含N系列)的棧板動作

  垂直6軸型機器人(包含N系列)在向透過棧板或相對位移等點運算獲取的點進行動作時，腕部有可能朝非預期的方向轉動。其原因在於，上述點運算屬於不依賴於機器人機種的命令，因此在未進行所需姿態旗標的轉換之狀態下進行動作。

  LJM函數可用於防止這種腕部的非預期轉動。

- RS系列的棧板動作

  同樣，RS系列在向透過棧板或相對位移等點運算獲取的點進行動作時，第1手臂有可能朝非預期的方向轉動。為了防止發生這種第1手臂的非預期轉動，LJM函數用於適當轉換點資料的姿態旗標。

  此外，對於RS系列，若轉換姿態旗標，可能會發生U軸向動作範圍外進行動作的錯誤。為了防止發生這種U軸的動作範圍外錯誤，LJM函數用於將U軸的目標角度補償為動作範圍內的目標角度。將選擇姿態旗標設為2，則可使用此功能。

- UVW坐標值

  以Pallet命令指定的3點(4點)UVW坐標值不同時，則使用點編號1以及坐標系資料1的UVW坐標值。

  忽略點編號2~4以及坐標系資料2~4的UVW坐標值。

- 附加軸坐標值

  以Pallet命令指定的3點(4點)坐標值保持附加軸坐標值(ST軸值)時，即便是附加軸坐標值，也會被均等分割。總之，將附加軸作為行走軸使用時，在棧板定義之際，也會在考量行走軸的動作前提下進行計算。因此，可定義考量行走軸

位置的、超過機器人動作範圍的大棧板。反之，即便定義不受棧板定義影響的附加軸，在定義棧板時，也需注意附加軸的位置。

參照

LJM、Pallet

Pallet函數範例

以下是將工件從棧板1移至棧板2的程式範例。

```
Function main
  Integer index
  Pallet 1, P1, P2, P3, 3, 5      '定義棧板1
  Pallet 2, P12, P13, P11, 5, 3   '定義棧板2
  For index = 1 To 15
    Jump Pallet(1, index)         '移至棧板1的點指數
    On 1                          '抓取工件
    Wait 0.5
    Jump Pallet(2, index)         '移至棧板2的點指數
    Off 1                         '釋放工件
    Wait 0.5
  Next I
Fend

Function main
  Integer i, j

  P0 = XY(300, 300, 300, 90, 0, 180)
  P1 = XY(200, 280, 150, 90, 0, 180)
  P2 = XY(200, 330, 150, 90, 0, 180)
  P3 = XY(-200, 280, 150, 90, 0, 180)

  Pallet 1, P1, P2, P3, 10, 10

  Motor On
  Power High
  Speed 50; Accel 50, 50
  SpeedS 1000; AccelS 5000

  Go P0
  P11 = P0 -TLZ(50)

  For i = 1 To 10
      For j = 1 To 10
        '點指定
        P10 = P11     '閃避點
      P12 = Pallet(1, i, j)     '目標點
      P11 = P12 -TLZ(50)     '接近起始點
      '各點的LJM轉換
      P10 = LJM(P10)
      P11 = LJM(P11, P10)
      P12 = LJM(P12, P11)
      '執行動作
        Jump3 P10, P11, P12 C0
    Next
  Next
Fend

Function main2
  P0 = XY(300, 300, 300, 90, 0, 180)
  P1 = XY(400, 0, 150, 90, 0, 180)
  P2 = XY(400, 500, 150, 90, 0, 180)
  P3 = XY(-400, 0, 150, 90, 0, 180)
```

```
  Pallet 1, P1, P2, P3, 10, 10

  Motor On
  Power High
  Speed 50; Accel 50, 50
  SpeedS 1000; AccelS 5000

  Go P0

  Do
    '點指定
    P10 = Here -TLZ(50)     '閃避點
    P12 = Pallet(1, Int(Rnd(9)) + 1, Int(Rnd(9)) + 1)     '目標點
    P11 = P12 -TLZ(50)     '接近起始點

    If TargetOK(P11) And TargetOK(P12) Then     '點的檢查
    '各點的LJM轉換
    P10 = LJM(P10)
    P11 = LJM(P11, P10)
    P12 = LJM(P12, P11)
    '執行動作
      Jump3 P10, P11, P12 C0
    EndIf
   Loop
Fend
```

3.19.6 PalletClr

清除設定的Pallet。

格式

PalletClr 棧板編號

參數

棧板編號

以數值指定要清除設定的棧板編號(0~15的整數)。

參照

Pallet

PalletClr範例

```
PalletClr 1
```

3.19.7 ParseStr/ParseStr函數

用於分析字串並傳回Token陣列。

格式

ParseStr 字串$, Token $(), 分隔符$

Token數 = ParseStr(輸入字串$, Token $(), 分隔符$)

參數

字串$
　　指定要分析的字串。
Token $()
　　指定要存放Token的陣列。不可指定已進行ByRef宣告的陣列。
分隔字元
　　指定1個字元以上的分隔字元。

傳回值

用作函數時，用於傳回已分析之Token數量。

參照

Redim、String

ParseStr/ParseStr函數範例

```
String toks$(0)
Integer i

ParseStr "1 2 3 4", toks$(), " "

For i = 0 To UBound(toks$)
   Print "token ", i, " = ", toks$(i)
Next i
```

3.19.8 Pass

用於通過(不停止)指定點附近的PTP動作。

格式

Pass 點指定 [, {On | Off | MemOn | MemOff} 位元編號 [, 點指定 ...]] [LJM [選擇姿態旗標]]

參數

點指定

指定P編號、P(運算式)和點標籤。

毫無遺漏地按升序或降序排列點資料時，可用冒號連接2個點編號進行指定(例：P (1:5))。

位元編號

指定要啟用/停用的I/O輸出位元或記憶體I/O位元。以整數或輸出標籤進行指定。

LJM

以LJM函數轉換閃避坐標、接近坐標、目標坐標。可省略。

選擇姿態旗標

指定賦予LJM函數的姿態旗標選擇參數。可省略。

說明

Pass用於通過指定點資料附近，並運作機器人手臂。不通過指定點資料本身。

若要指定點資料，則使用 (P0，P1，...)。請用逗號分隔各點。

若要在執行動作時啟用/停用輸出位元，請用逗號分隔各點之間，然後插入On/Off命令。在機器人手臂到達剛插入On/Off前的點之前，執行On/Off。

若在Pass之後緊接著另1個Pass陳述式，控制則不會停止在最初Pass的最終指定點附近，而會移至下一個Pass。

若在Pass之後緊接著Pass以外的動作命令，雖然機器人會停止在Pass陳述式的最終指定點，但不會進行Fine定位。

若在Pass之後緊接著動作命令以外的命令、陳述式、函式等，則在機器人手臂到達以Pass指定的最終點之前執行這些命令、陳述式、函式等。

若要求以Fine正確定位到目標位置，則如下範例所示，請指定目標位置，並在Pass之後插入Go。

```
Pass P5; Go P5; On 1; Move P10
```

加速值或減速值越大，手臂會越靠近指定點附近。可利用Pass命令避開障礙物。

若使用LJM參數，則可簡化使用LJM函數的程式。

例如：

```
P11 = LJM(P1, Here, 1)
P12 = LJM(P2, P11, 1)
P13 = LJM(P3, P12, 1)
Pass P11, P12, P13
```

可將上述4行的程式取代為

```
  Pass P1, P2, P3 LJM 1
```

的1行程式。

LJM參數對於垂直6軸型機器人(包含N系列)以及RS系列機器人有效。

若按預設值使用姿態旗標選擇，則可省略。

```
  Pass P1, P2, P3 LJM
```

參照

Accel、Go、Jump、Speed

Pass範例

以下是使用Pass命令操作機器人手臂的範例。

```
Function main
  Jump P1
  Pass P2    '使手臂靠近P2，在到達P2之前執行下一個命令
  On 2
  Pass P3
  Pass P4
  Off 0
  Pass P5
Fend
```

3.19.9 Pause

用於暫停所有可暫停的工作。

格式

Pause

說明

執行Pause，會暫停所有可暫停的工作(執行Xqt命令時，未指定NoPause或NoEmgAbort的工作)。暫停所有正在執行動作命令的工作。

Pause無法暫停背景工作。

注意

- QP對Pause的影響

  QP命令用於設定在執行Pause時立即停止機器人動作，還是在完成動作後暫停程式。詳細內容請參閱說明的QP命令項目。

Pause範例

以下範例是以Pause暫停工作的情形。使用Pause行，會暫停程式。按下操作員視窗的[繼續執行]按鈕，重新開始工作。

```
Function main

  Xqt monitor
  Go P1
  On 1
  Jump P2
  Off 1
  Pause             '暫停程式
  Go P40
  Jump P50
Fend
```

3.19.10 PauseOn函數

用於傳回暫停狀態(Pause狀態)。

格式

PauseOn

傳回值

處於暫停狀態時，傳回「True」；除此之外，傳回「False」。

說明

本函數僅用於NoPause工作、NoEmgAbort工作(在Xqt時，指定NoPause或NoEmgAbort開始的特殊工作)和背景工作。

參照

ErrorOn、EstopOn、SafetyOn、Wait、Xqt

PauseOn函數範例

以下範例是由控制器監視暫停，若有發生暫停，則啟用/停用I/O的程式。但是，因安全門打開而處於暫停狀態時，則不啟用/停用I/O。

```
Function main

  Xqt PauseMonitor, NoPause
  :
  :
Fend

Function PauseMonitor
  Boolean IsPause
  IsPause = False
  Do
    Wait 0.1
    If SafetyOn = True Then
      If IsPause = False Then
        Print "Safety On"
        IsPause = True
      EndIf
    ElseIf PauseOn = True Then
      If IsPause = False Then
        Print "InPause"
        If SafetyOn = False Then
          Off 10
          On 12
        EndIf
      IsPause = True
      EndIf
    Else
      If IsPause = True Then
        Print "OutPause"
        On 10
        Off 12
        IsPause = False
      EndIf
    EndIf
  Loop
Fend
```

3.19.11 PDef函數

用於傳回是否定義指定的點資料。

格式

PDef (點資料)

參數

點數據

指定整數值、P編號、P(運算式)、點標籤。相容性相關注意事項目前無法在點資料引數上指定變數。若要使用變數，請記述為PDef(P(varName))。

傳回值

定義點資料時，傳回「True」；除此之外，傳回「False」。

參照

Here、Pdel

PDef函數範例

```
If Not PDef(1) Then
    Here P1
EndIf
Integer i
For i = 0 to 10
    If PDef (P(i)) Then
        Print "P(";i;") is defined"
    EndIf
Next
```

3.19.12 PDel

用於刪除指定的點資料。

格式

PDel 起始點編號 [, 結束點編號]

參數

開始點編號

以整數值或運算式指定要刪除的點數據範圍的開始編號。

結束點編號

以整數值或運算式指定要刪除的點數據範圍的結束編號。

說明

刪除從機器人記憶體上指定的點資料。刪除參數的起始點編號至結束點編號的點資料。請將起始點編號設為小於結束點編號。

PDel範例

```
> p1=10,300,-10,0/L
> p2=0,300,-40,0
> p10=-50,350,0,0
> pdel 1,2        '刪除點1和2
> plist
P10  =   -50.000,  350.000,    0.000,    0.000 /R /0
> pdel 50         '刪除點50
> pdel 100,200    '刪除點100~200
>
```

3.19.13 PDescription

設定指定點資料的註解。

格式

PDescription 點資料,新註解

參數

點數據

以整數值、P編號、P(運算式)、點標籤進行指定。無法在點資料引數上指定變數。若要使用變數，請記述為 PDescription(P(varName)), “新註解”。

新描述

指定已指定的點數據描述之字串運算式。

說明

PDescription用於將註解儲存於控制器記憶體中指定的點資料中。

若建立專案或執行程式，則從記憶體中清除儲存於控制器記憶體中的註解。需儲存時，請執行「SavePoints」，然後儲存於點檔案中。

參照

PDef函數、PDescription$函數、PLabel、PLabel$函數

PDescription範例

```
PDescription 1, "Comment"
```

3.19.14 PDescription$函數

用於傳回在指定點編號中定義的點之註解。

格式

PDescription$ (點資料)

參數

點數據

以整數值、P編號、P(運算式)、點標籤進行指定。無法在點資料引數上指定變數。若要使用變數，請記述為PDescription$(P(varName))。

傳回值

以字串傳回指定點編號的註解。

參照

PDef函數、PDescription、PLabel、PLabel$函數

PDescription$函數範例

```
Print PDescription$(1)
Print PDescription$(P(i))
```

3.19.15 PeakSpeedClear

清除關節的峰值速度並進行初始化。

格式

PeakSpeedClear [關節指定1 [, 關節指定2 [, 關節指定3 [, 關節指定4 [, 關節指定5 [, 關節指定6 [, 關節指定7 [, 關節指定8 [, 關節指定9]]]]]]]]

參數

關節指定1 - 關節指定9

以整數值或運算式指定關節編號。未指定參數時，清除所有關節的峰值速度。附加軸的S軸為8，T軸為9。若指定不存在的關節編號，則發生錯誤。

說明

PeakSpeedClear用於清除指定關節的峰值速度。

執行PeakSpeed前，請務必執行PeakSpeedClear。

本命令不支援PG附加軸。

參照

AvgSpeed、PeakSpeed

PeakSpeedClear範例

[例1] 這是清除所有關節的峰值速度後，顯示指定關節編號的速度值之命令執行範例。

```
> PeakSpeedClear
> Go P1
> PeakSpeed 1
    -0.273
> PeakSpeed
    -0.273    -0.164
    -0.080     0.258
    -0.005     0.401
     0.000     0.000
     0.000
>
```

[例2] 這是對於垂直多關節機器人清除第1關節、第4關節、第5關節的峰值速度後，顯示指定關節編號的峰值速度之命令執行範例。

```
> PeakSpeedClear 4, 1, 5
> Go P1
> PeakSpeed 1
     -0.273
> PeakSpeed 4
     0.258
```

3.19.16 PeakSpeed

顯示指定關節的峰值速度。

格式

PeakSpeed [關節編號]

參數

關節編號

以整數值或運算式指定關節編號。可省略。附加軸的S軸為8，T軸為9。

結果

顯示所有關節的目前峰值速度。

說明

PeakSpeed用於顯示在關節速度的絕對值最大的速度上標上符號的值。最大速度為1。以-1~1的實數值表示峰值速度。

請於執行PeakSpeedClear後執行PeakSpeed，並顯示關節的峰值速度。

在虛擬控制器及空運行的情況下，根據命令速度(並非實際速度)計算峰值速度。

本命令不支援PG附加軸。

參照

AvgSpeed、PeakSpeedClear、PeakSpeed函數

PeakSpeed範例

```
> PeakSpeedClear
> Go P1
> PeakSpeed 1
     -0.273
> PeakSpeed
    -0.273     0.163
    -0.080     0.258
    -0.005    -0.401
     0.000     0.000
     0.000
>
```

3.19.17 PeakSpeed函數

用於傳回指定關節的峰值速度。

格式

PeakSpeed (關節編號)

參數

關節編號

以整數值或運算式指定關節編號。附加軸的S軸為8，T軸為9。

傳回值

以-1~1的實數值傳回。

說明

PeakSpeed函數用於傳回在關節速度的絕對值最大的速度上標上符號的值。最大速度為1。以-1~1的實數值表示峰值速度。

請於執行PeakSpeedClear後執行PeakSpeed，並顯示關節的峰值速度。

在虛擬控制器及空運行的情況下，根據命令速度(並非實際速度)計算峰值速度。

本函數不支援PG附加軸。

參照

AvgSpeed、PeakSpeedClear、PeakSpeed

PeakSpeed函數範例

是在程式中使用PeakSpeed函數的範例。

```
Function DisplayPeakSpeed
   Integer i

   PeakSpeedClear
   Go P1
   Print "Peak Speeds:"
   For i = 1 To 6
      Print "Joint ", i, " = ", PeakSpeed (i)
   Next i
Fend
```

3.19.18 PerformMode

可選擇機器人的模式。

格式

(1) PerformMode [模式編號] [, 機器人編號]

(2) PerformMode

參數

模式編號

以整數值(0~2)或如下所示的常數指定要設定的模式編號。唯有在命令視窗中執行時方可省略。

常數	值	內容
MODE_STANDARD	0	設定標準模式。
MODE_BOOST	1	設定大功率模式。
MODE_LOW_VIBRATRION	2	設定低振動模式。

機器人編號

以整數值指定要設定的機器人編號。省略時，即以目前選擇的機器人為對象。

結果

- 若用 (1) 的格式指定，則以指定的模式編號設定動作模式。
- 若用 (2) 的格式指定，則顯示目前選擇的機器人中設定的模式編號。

說明

PerformMode是一項按用途改變機器人的優先性能(模式)的功能。支援的模式，請參閱以下手冊。
「機械手手冊 」

- 標準模式：是以經過時間、動作Duty、動作停止時的振動平衡為優先的模式。可用於各種應用程式。
- 大功率模式：是以縮短單一動作的動作時間為優先的模式。與標準模式相比，雖然動作Duty和停止動作時的振動有所惡化，但可縮短單一動作時間。用於高Duty用途時，標準模式較能縮短生產節拍時間。

  建議應用程式：搬運等
- 低振動模式：是以減低動作停止時的振動為優先的模式。動作時間比標準模式長，但降低了停止時的振動。

  建議應用程式：精密部件的搬運、組裝等

各模式性能比較表

模式	比較項目		
	單一動作時間 (*1)	停止時振動	動作Duty (*2)
標準	○	○	○
大功率	◎	△	△
低振動	△	◎	○

表中符號表示性能的程度。

○：標準 ◎：提升 △：略微下降

- (*1) 單一動作時間是指，機器人從目前位置移往目標坐標所花費的移動時間。
- (*2) 動作Duty是指，在最大加減速度下，無過過載錯誤下，可動作的動作時間之比。

注意

- 如果是不對應的機型，當修改成大功率模式和低震動模式時，會發生錯誤。
- 對應動作命令：PTP動作(Go、BGo、TGo、Jump、JTran、PTran、Pulse)
- 不改變CP動作的以下性能。
  - 軌跡精度
  - AccelS、AccelR、SpeedS、SpeedR的上限值
  - 加速度命令錯誤、速度命令錯誤的發生機率
- 關於模式自動變更為初始模式(標準模式)的條件

  下表所示為自動變更模式的條件。

	模式的變化
控制器電源ON	變更為初始模式
重新啟動控制器	變更為初始模式
馬達ON	變更為初始模式
組建/重建	不變更
執行Reset時	不變更
執行SFree時	變更為初始模式

參照

Bo、Go、Jump、JTran、PerformMode函數、TGo

PerformMode範例

```
PerformMode MODE_STANDARD
Go P1
PerformMode 2
Go P2
```

3.19.19 PerformMode函數

用於傳回機器人動作模式的狀態。

格式

PerformMode ([機器人編號])

參數

機器人編號

以整數值指定要確認狀態的機器人編號。省略時，即以目前選擇的機器人為對象。

傳回值

用於以整數值傳回目前設定動作模式的值。

- 0 = 標準模式
- 1 = 大功率模式
- 2 = 低振動模式

參照

PerformMode

PerformMode函數範例

```
Print PerformMode(1)
```

3.19.20 PG_FastStop

緊急停止連續旋轉的脈衝輸出軸。

格式

PG_FastStop

說明

若執行PG_FastStop，則緊急停止目前選擇的PG機器人之連續旋轉動作。一般停止時，請使用減速停止(PG_SlowStop)。

參照

PG_Scan、PG_SlowStop

PG_FastStop範例

以下範例是使脈衝輸出軸連續進行旋轉動作10秒後緊急停止的情形。

```
Function main
  Motor On
  PG_Scan 0
  Wait 10
  PG_FastStop                    '緊急停止連續旋轉動作
Fend
```

3.19.21 PG_LSpeed

用於設定脈衝輸出軸開始加速時的脈衝速度以及結束減速時的脈衝速度。

格式

PG_LSpeed 速度設定值 [, 結束減速速度],

參數

速度設定值

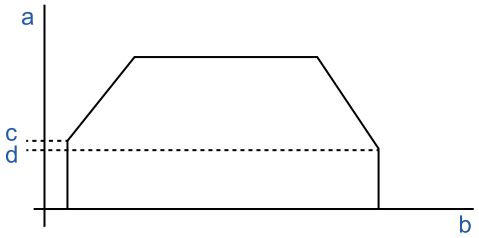
以運算式或數值指定脈衝速度(1~32767的整數，單位：pulse/sec)。

結束減速速度

以運算式或數值指定結束減速時的脈衝速度(1~32767的整數，單位：pulse/sec)。

說明

PG_LSpeed用於指定脈衝輸出軸開始加速時的脈衝速度以及結束減速時的脈衝速度。用於在最大自起動頻率的範圍內將步進馬達的初始速度以及結束速度設為較高的值，以發揮高性能馬達的性能，或降低設定值，以防步進馬達失步。預設值被設為300 pulse/sec，平常請依此使用。



符號	說明
a	速度
b	時間
c	開始加速速度
d	結束減速速度

一旦省略結束減速速度，速度設定值就會變成結束減速速度的設定值。

在以下任一情況下，PG_LSpeed值會被初始化。

- 啟動控制器時
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

參照

PG_LSpeed函數

PG_LSpeed範例

PG_LSpeed可在命令視窗或程式中使用。以下為兩種情況時的範例。

```
Function pglspeedtst
        Motor On
        Power High
Speed 30;Accel 30,30
```

```
PG_LSpeed 1000
Go P0
Fend
```

以下是在命令視窗中設定PG_LSpeed值的範例。

```
> PG_LSpeed 1000,1100
>
```

3.19.22 PG_LSpeed函數

用於傳回目前設定的脈衝輸出軸開始加速時的脈衝速度以及結束減速時的脈衝速度。

格式

PG_LSpeed [(參數編號)]

參數

參數編號

由如下所示的編號，指定1個設定值編號。省略時被視為指定1。

- 1：開始加速時的脈衝速度
- 2：結束減速時的脈衝速度

傳回值

用於傳回1~32767的整數值(單位：pulse/sec)。

參照

PG_LSpeed

PG_LSpeed函數範例

```
Integer savPGLSpeed

savPGLSpeed = PG_LSpeed(1)
```

3.19.23 PG_Scan

用於開始脈衝輸出軸的連續旋轉動作。

格式

PG_Scan 旋轉方向

參數

旋轉方向

指定連續旋轉的旋轉方向。

- 0：+ (CW)方向
- 1：- (CCW)方向

說明

執行PG_Scan，會開始目前選擇的PG機器人之連續旋轉動作。

若要執行連續旋轉動作，必須透過機器人設定將PG參數的連續旋轉設為啟用。

完成程式執行工作時，會停止連續旋轉。

參照

PG_FastStop

PG_Scan範例

以下範例是使脈衝輸出軸連續進行旋轉動作10秒後緊急停止的情形。

```
Function main
  Motor On
  Power High
  Speed 10; Accel 10,10
  PG_Scan 0
  Wait 10
  PG_SlowStop
Fend
```

3.19.24 PG_SlowStop

用於減速並停止連續旋轉的脈衝輸出軸。

格式

PG_SlowStop

說明

執行PG_SlowStop，會減速並停止目前選擇的PG機器人之連續旋轉動作。

參照

PG_Scan、PG_FastStop

PG_SlowStop範例

以下範例是使脈衝輸出軸連續進行旋轉動作10秒後緊急停止的情形。

```
Function main
  Motor On
  PG_Scan 0
  Wait 10
  PG_SlowStop                    '緊急停止連續旋轉動作
Fend
```

3.19.25 PLabel

用於設定指定點資料的標籤。

格式

PLabel 點編號,新標籤

參數

點編號

以運算式或數值指定點編號。

新標籤

指定已指定的點數據所使用標籤之字串運算式。

參照

PDef函數、PDescription、PDescription$函數、PLabel$函數、PNumber函數

PLabel範例

```
PLabel 1, "pick"
```

3.19.26 PLabel$函數

用於傳回在指定點編號中定義的點標籤。

格式

PLabel$ (點資料)

參數

點數據

以整數值、P編號、P(運算式)、點標籤進行指定。相容性相關注意事項目前無法在點資料引數上指定變數。若要使用變數，請記述為PLabel$(P(varName))。

參照

PDef函數、PDescription、PDescription$函數、PLabel、PNumber函數

PLabel$函數範例

```
Print PLabel$(1)
Print PLabel$(P(i))
```

3.19.27 Plane

用於設定和顯示進入檢測平面。

格式

(1) Plane 平面編號[, 機器人編號], 坐標系資料 Plane 平面編號[, 機器人編號], 原點, X軸指定, Y軸指定

(3) Plane 平面編號[, 機器人編號]

(4) Plane

參數

平面編號
　　指定工作平面的編號。可用1~15的整數(合計15個)定義進入檢測平面。
機器人編號
　　以整數值指定要設定的機器人編號。省略機器人編號時，則以目前選擇的機器人為對象。
坐標系資料
　　以點數據直接指定工作平面的坐標系。
原點
　　以P#(整數)或P(運算式)指定要定義工作平面原點的機器人坐標系上的位置。
X軸指定
　　以P#(整數)或P(運算式)指定要定義工作平面X軸上點的機器人坐標系上的位置。
Y軸指定
　　以P#(整數)或P(運算式)指定要定義工作平面Y軸上點的機器人坐標系上的位置。

結果

- 用格式 (3) 指定後，則顯示指定平面編號的進入檢測平面設定。
- 用格式 (4) 指定後，則顯示目前選擇的機器人所設定的所有進入檢測平面設定。

說明

Plane用於設定進入檢測平面。進入檢測平面用於檢測：基於目前選擇的工具計算出的手臂尖端位置是否進入到以設定的進入檢測平面分隔出來的一方空間中。以設置於機器人基本坐標系上的任意坐標系的XY平面設定進入檢測平面。接著，當手臂尖端位置進入以該平面分隔出來的空間之中包括該坐標系的Z軸+方向的空間內時，即視為進入檢測平面。

若設定進入檢測平面，在啟動控制器時，無論機器人的馬達電源處於何種狀態，都始終執行檢測處理。

下面將針對各格式進行說明。

(1) 使用以基本坐標系為基準的平移量和表示旋轉量的點資料指定作為進入檢測平面之本的坐標系，以設定進入檢測平面。

例：

```
Plane 1, XY(x, y, z, u, v, w)
Plane 1, P1
```

(2) 透過指定原點、X軸上點、Y軸上點這3點，定義進入檢測平面(XY平面)。只使用各點的X、Y、Z坐標，忽略U、V、W坐標。同時計算Z軸，以形成右手系統，並設定進入檢測方向。

例：

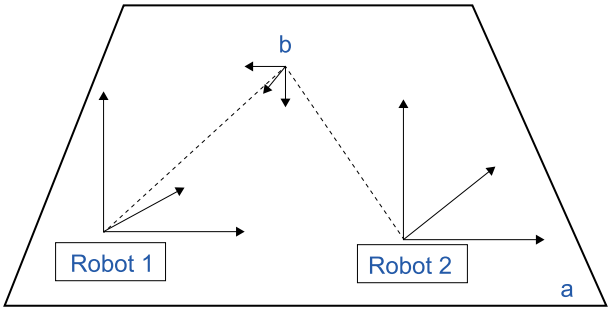
```
Plane 1, P1, P2, P3
```

(3) 顯示指定進入檢測平面的設定。

(4) 顯示已設定的所有進入檢測平面的設定。

可隨時以GetRobotInsidePlane函數、InsidePlane函數取得對進入檢測平面的進入檢測結果。此外，還可利用GetRobotInsidePlane函數，作為Wait命令的等待條件運算式。而且，還可進行遠端輸出設定，以便將檢測結果輸出到I/O。

若要讓數個機器人共享1個平面，需定義從每個機器人坐標所見的平面。



符號	說明
a	進入檢測平面
b	進入檢測平面坐標系

注意

- 選擇工具

  透過目前選擇的工具執行進入檢測。已變更工具選擇時，有可能在機器人未動作的狀態下從平面內移到平面外，或從平面外移到平面內。

- 附加軸

  在帶有附加軸(包括行走軸在內)S、T的機器人之情況下，設定的進入檢測平面不依賴於附加軸位置。以機器人的基本坐標系為基準進行設定。

參照

Box、GetRobotInsidePlane、InsidePlane、PlaneClr、PlaneDef

提示

- 由Robot Manager設定Plane

  Epson RC+可透過[專案]選單-[Robot Manager]的[進入檢測平面]面板設定Plane值。

Plane範例

這是命令視窗中的操作範例。

以機器人坐標系為基準，在朝Z軸方向的−-20mm水平面上，將朝下方向定義為檢測方向的範例：

```
> plane 1, xy(100, 200, -20, 90, 0, 180)
```

以機器人坐標系為基準，朝X方向移動50mm，朝Y方向移動200mm後，將以環繞Y軸旋轉45度所獲得的坐標系形成的XY平面定義為進入檢測平面的範例：

```
> plane 2, xy(50, 200, 0, 0, 45, 0)
```

使用機器人的工具坐標系直接設定進入檢測平面。(垂直6軸型機器人)

```
> plane 3, here
```

3.19.28 Plane函數

本函數用於傳回已設定的進入檢測平面。

格式

Plane (平面編號[, 機器人編號])

參數

平面編號

以運算式或數值指定要確認的工作平面編號(1~15的整數)。

機器人編號

以整數值指定要設定的機器人編號。省略機器人編號時，則以目前選擇的機器人為對象。

傳回值

用於將設定的進入檢測平面之本的坐標系資料作為點資料傳回。

參照

GetRobotInsidePlane、InsidePlane、Plane、PlaneClr、PlaneDef

Plane函數範例

```
P1 = Plane(1)
```

3.19.29 PlaneClr

清除進入檢測平面(未定義)。

格式

PlaneClr 平面編號[, 機器人編號]

參數

平面編號

以運算式或數值指定要清除設定(未定義)的工作平面編號(1~15的整數)。

機器人編號

以整數值指定要設定的機器人編號。省略機器人編號時，則以目前選擇的機器人為對象。

參照

GetRobotInsidePlane、InsidePlane、Plane、PlaneDef

PlaneClr範例

```
PlaneClr 1
```

3.19.30 PlaneDef函數

用於傳回進入檢測平面的設定狀態。

格式

PlaneDef (平面編號[, 機器人編號])

參數

平面編號
　　指定傳回狀態的工作平面編號(1~15的整數)。
機器人編號
　　以整數值指定要設定的機器人編號。省略機器人編號時，則以目前選擇的機器人為對象。

傳回值

若有設定以平面編號指定的進入檢測平面，則傳回「True」；未設定時，則傳回「False」。

參照

GetRobotInsidePlane、Box、InsidePlane、Plane、PlaneClr

PlaneDef函數範例

```
Function DisplayPlaneDef(planeNum As Integer)

   If PlaneDef(planeNum) = False Then
      Print "Plane ", planeNum, "is not defined"
   Else
      Print "Plane 1: ",
      Print Plane(PlaneNum)
   EndIf
Fend
```

3.19.31 PList

用於顯示目前機器人記憶體中的點資料。

格式

(1) Plist

(2) Plist 點編號

(3) Plist 起始點編號,

(4) Plist 起始點編號, 結束點編號

參數

點編號
　　編號範圍為0~999。
開始點編號
　　以0~999的編號指定要顯示的開頭點編號。
結束點編號
　　以0~999的編號指定要顯示的最後點編號。

結果

傳回點資料。

說明

Plist用於顯示目前機器人記憶體中的點資料。

不存在指定範圍的點資料時，不會顯示資料。

指定的起始點編號大於結束點編號時，會發生錯誤。

- (1) PList：顯示所有位置資料。
- (2) PList 點編號：顯示指定的1個點編號的位置資料。
- (3) PList 起始點編號,：顯示從起始點編號到位置資料的結束為止。
- (4) PList 開始點編號、結束點編號：顯示從指定的開始點編號到結束點編號為止的位置資料。

PList範例

顯示的形式因機器人的類型或有無附加軸的情況而異。

以下範例表示的是SCARA機器人無附加軸時的情形。

在以下範例中顯示指定的點資料。

```
> plist 1
P1   = XY(   200.000,    0.000,  -20.000,    0.000 )  /R /0
>
```

在以下範例中顯示10~20的點資料。此範例中的指定範圍內存在3個點資料。

```
> plist 10, 20
P10  =   XY( 290.000,    0.000,  -20.000,    0.000 ) /R /0
P12  =   XY( 300.000,    0.000,    0.000,    0.000 ) /R /0
P20  =   XY( 285.000,   10.000,  -30.000,   45.000 ) /R /0
>
```

在以下範例中顯示從10到結束為止的點資料。

```
> plist 10,
P10  =   XY( 290.000,    0.000,  -20.000,    0.000 ) /R /0
P12  =   XY( 300.000,    0.000,    0.000,    0.000 ) /R /0
P20  =   XY( 285.000,   10.000,  -30.000,   45.000 ) /R /0
P30  =   XY( 310.000,   20.000,  -50.000,   90.000 ) /R /0
```

3.19.32 PLocal

用於設定指定點資料的本地屬性。

格式

PLocal (點資料) = 本地編號

參數

點數據

指定整數值、P編號、P(運算式)、點標籤。相容性相關注意事項目前無法在點資料引數上指定變數。若要使用變數，請記述為PLocal(P(varName))。

本地編號

以0~15的整數值或運算式指定要設定的本地屬性編號。

參照

PLocal函數

PLocal範例

```
PLocal(pick) = 1
```

3.19.33 PLocal函數

用於傳回在指定點編號中設定的本地編號。

格式

PLocal (點資料)

參數

點數據

指定整數值、P編號、P(運算式)、點標籤。相容性相關注意事項目前無法在點資料引數上指定變數。若要使用變數，請記述為PLocal(P(varName))。

傳回值

用於傳回指定點編號的本地編號。

參照

PLocal

PLocal函數範例

```
Integer localNum

localNum = PLocal(pick)
```

3.19.34 Pls函數

用於傳回目前位置的各關節脈衝值。

格式

Pls (關節編號)

參數

關節編號

指定要求出目前脈衝值的關節。附加軸的S軸為8，T軸為9。

傳回值

用於以數值傳回關節(以關節編號指定)的目前位置脈衝值。

說明

Pls用於讀取各關節的目前脈衝值。可儲存求出的值，之後連同Pulse命令一起使用。

參照

CX、CY、CZ、CU、CV、CW、Pulse

Pls函數範例

以下是求出各關節脈衝值並輸出這些值的簡易範例。

```
Function plstest
  Real t1, t2, z, u
  t1 = pls(1)
  t2 = pls(2)
  z = pls(3)
  u = pls(4)
  Print "T1 joint current Pulse Value: ", t1
  Print "T2 joint current Pulse Value: ", t2
  Print "Z joint current Pulse Value: ", z
  Print "U joint current Pulse Value: ", u
Fend
```

3.19.35 PNumber函數

用於傳回與點標籤對應的點編號。

格式

PNumber (點標籤)

參數

點標籤

指定目前點檔案中的點標籤或表示點標籤的字串。

參照

PDef函數、PLabel$函數

PNumber函數範例

```
Integer pNum
String pointName$

pNum = PNumber(pick)

pNum = PNumber("pick")

pointName$ = "place"
pNum = PNumber(pointName$)
```

3.19.36 PosFound函數

用於傳回執行Find命令時的狀態。

格式

PosFound

傳回值

在手臂移動中發現坐標位置時，傳回「True」；除此之外，傳回「False」。

參照

Find

PosFound函數範例

```
Find Sw(5) = ON
Go P10 Find
If PosFound Then
    Go FindPos
Else
    Print "Error: Cannot find the sensor signal."
EndIf
```

3.19.37 Power

將功率模式設定為High或Low。

格式

(1) Power { High | Low } [, Forced]

參數

High | Low
　　設定High或Low。預設為Low。
Forced
　　可省略。通常會省略。

結果

若省略參數，則顯示目前的功率模式。

說明

將功率模式設為High或Low。此外，顯示目前的功率模式。

- Low：將功率模式設為Low時，低功率模式為ON。這表示機器人處於緩慢(250mm/sec以下的速度)動作狀態。此外，用於抑制馬達功率輸出。
- High：將功率模式設為High時，低功率模式為OFF。這表示機器人按以Speed、Accel、SpeedS、AccelS指定的速度/加減速度進行動作。

以下表示切換為低功率模式的操作。此時，速度和加減速度均被限制為各機器人的預設值。預設值記載於各機器人的規格表中。請一併參照以下手冊。
「Epson RC+ 使用者指南 - 關於安全」

成為低功率模式的條件

- 控制器電源 開啟
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

被限制為預設值的設定

- Speed
- Accel
- SpeedS
- AccelS

注意

- 低功率模式(Power Low)和最大速度的關係

  在低功率模式下會限制馬達輸出，使實際的動作速度在初始值範圍內。設為低功率模式時，即使在命令視窗中或透過程式下達高速指示，仍以初始值速度進行動作。如需更高速的動作，就必須設為Power High。

  當機器人在高功率模式下動作，並切換到低功率模式時，可能會導致超速錯誤和低功率扭矩錯誤。

- 高功率模式(Power High)和最大速度的關係

  在高功率模式下可使用高於初始值的速度。

- Forced旗標

  可在機器人動作時(包括暫停在內)變更功率模式。

  若在機器人處於低功率動作的狀態下切換為高功率模式，則從下一個動作開始，可透過設定進行高速動作。

  若在機器人處於高功率動作的狀態下切換為低功率模式，則有可能發生超速錯誤或低功率扭矩錯誤。

  請暫停機器人，並在指定Forced旗標之後切換為低功率模式。

參照

Accel、AccelS、Speed、SpeedS

Power範例

以下表示使用Power的命令視窗中的操作範例。

```
> Speed 50          '在Low Power模式下設定高速

> Accel 100、100    '設定高加速度

> Jump P1           '以低速、低加速度移動

> Speed             '顯示目前速度
Low Power Mode
   50
   50     50

> Accel             '顯示目前的加速度
Low Power Mode
   100    100
   100    100
   100    100

> Power High        '設為High Power模式

> Jump P2           '以高速運作機器人
```

3.19.38 Power函數

用於傳回功率模式。

格式

Power [(機器人編號)]

參數

機器人編號

以整數值指定要確認狀態的機器人編號。省略時，即以目前選擇的機器人為對象。

傳回值

- 0 = 低功率模式
- 1 = 高功率模式

參照

Power

Power函數範例

```
If Power = 0 Then
    Print "Low Power Mode"
EndIf
```

3.19.39 PPls函數

用於根據指定的坐標值計算關節脈衝並進行傳回。

格式

PPls (坐標值, 關節編號)

參數

坐標值

用於指定點數據。

關節編號

以運算式或數值指定關節編號(1~9的整數)。附加軸的S軸為8，T軸為9。

傳回值

以Long型數值傳回計算的關節脈衝。

參照

Agl、CX、CY、CZ、CU、CV、CW、PAgl

PPls函數範例

```
Long pulses1

pulses1 = PPls(P10, 1)
```

3.19.40 Print

用於在Run視窗、操作員視窗、命令視窗、巨集視窗等顯示器上顯示資料。

格式

Print 顯示資料 [,顯示資料...] [,]

Print

參數

顯示資料

以數值或字串運算式進行指定。可省略。

, (逗號)

陳述式的結尾如有逗號，則不換行。可省略。

結果

傳回變數資料或指定的字串。

說明

在顯示裝置上顯示變數資料或字串。

除非行尾無逗號，否則會自動換行。

注意

可透過本命令一次處理的最大資料長度為256 Byte。

- 請勿在Loop陳述式中只使用Print命令

  如果在Loop陳述式中只使用Print命令，控制器有可能进入意外停機狀態。

  請與Wait命令或動作命令一起使用。

  不良範例

```
Do
    Print "1234"
Loop
```

  正常範例

```
Do
    Print "1234"
    Wait 0.1
Loop
```

參照

Print #

Print範例

在以下範例中，將從P100的坐標值減去U軸坐標值並將該值加入變數uvar後的值顯示在目前的顯示器上。

```
Function test
  Real uvar
  uvar = CU(P100)
```

```
  Print "The U Axis Coordinate of " + Chr$(34) + "P100" + Chr$(34) + " is ", uvar
Fend
```

3.19.41 Print

用於將資料輸出到指定的檔案、通訊連接埠、資料庫或裝置。

格式

Print #連接埠編號, 輸出資料 [,輸出資料...][,]

參數

連接埠編號

表示檔案、通訊連接埠、資料庫或設備的ID編號。檔案編號是以ROpen、WOpen、AOpen等陳述式指定的編號。通訊連接埠編號是以OpenCom(RS-232C)或OpenNet(TCP/IP)陳述式指定的編號。資料庫編號是以OpenDB陳述式指定的編號。

裝置ID為以下數值。

- 21 RC+
- 20 TP4

輸出資料...

指定數值或字串。

, (逗號)

陳述式的結尾如有逗號，則不換行。可省略。

說明

Print #用於將變數資料、數值或字串輸出到以連接埠编号指定的通訊連接埠或裝置。

注意

- 支援的控制器型號

  如果在T/VT系列中指定RS-232C埠時會發生錯誤。

- 最大資料長度
  - 最大資料長度

  可透過本命令一次處理的最大資料長度為256 Byte。不過，以資料庫為對象時，最大資料長度為4096 Byte。以通訊連接埠(TCP/IP)為對象時，最大資料長度則為1024 Byte。

- 與其它控制器交換變數時

  若要指定数個字串變數以及數值變數和字串變數兩者，則需要在字串資料中明確新增分隔符號(「, 」)。

  [在控制器之間使用通訊連接埠交換字串變數、數值變數的範例]

  傳送側(任一模式皆可。)

  ```
  Print #PortNum, "$Status,", InData, OutData
  Print #PortNum, "$Status", ",",InData, OutData
  ```

  接收側

  ```
  Input #PortNum, Response$, InData, OutData
  ```

- 寫入檔案時會被緩衝

  可用Flush陳述式寫入被緩衝的資料。以Close陳述式關閉檔案時，也進行寫入。

- 請勿同时使用Print #命令、Wait命令和操作命令
  請勿在Loop陳述式中只使用Print #命令

  如果在Loop陳述式中只使用Print #命令，控制器有可能进入意外停機狀態。

  根據控制器的負載狀態，即使使用Wait命令或操作命令，咨詢可能也無法正確顯示。Wait時間請設置0.1 (秒) 以上。

  不良範例

```
Do
    Print #24，"1234"
Loop
```

  正常範例

```
Do
    Print #24,"1234"
    Wait 1
Loop
```

參照

Input#、Print、Write、WriteBin

Print #範例

以下是使用Print #的簡易範例。

```
Function printex
  String temp$
  Print #1, "5"     ' 將「5」輸出到連接埠1 temp$ = "hello”
  Print #1, temp$
  Print #2, temp$
  Print #1 " Next message for " + Chr$(34) + "port 1" + Chr$(34)
  Print #2 " Next message for " + Chr$(34) + "port 2" + Chr$(34)
Fend
```

3.19.42 ProjectName$函數

用於傳回目前載入的專案名稱。

格式

ProjectName$

傳回值

用於以字串傳回專案名稱。

說明

用於傳回控制器目前載入的專案名稱。

3.19.43 PTCLR

清除關節的峰值扭矩並進行初始化。

格式

PTCLR [關節指定1 [, 關節指定2 [, 關節指定3 [, 關節指定4 [, 關節指定5 [, 關節指定6 [, 關節指定7 [, 關節指定8 [, 關節指定9]]]]]]]]

參數

關節指定1 - 關節指定9

以整數值或運算式指定關節編號。未指定參數時，清除所有關節的峰值扭矩。附加軸的S軸為8，T軸為9。若指定不存在的關節編號，則發生錯誤。

說明

PTCLR用於清除指定關節的峰值扭矩。

執行PTRQ前，請務必執行PTCLR。

參照

ATRQ、PTRQ

PTCLR範例

[例1] 以下是清除所有關節的峰值扭矩後，顯示指定關節編號的扭矩值之命令執行範例。

```
> ptclr
> go p1
> ptrq 1
    0.227
> ptrq
    0.227    0.118
    0.249    0.083
    0.000    0.000
>
```

[例2] 以下是垂直多關節機器人清除J1、J4、J5的峰值扭矩後，顯示已指定關節編號的扭矩值之命令執行範例。

```
> ptclr 4, 1, 5
> go p1
> ptrq 1
    0.227
> ptrq 4
    0.083
```

3.19.44 PTPBoost

用於設定和顯示移動微小距離進行PTP(point to point)動作時的加減速演算法調整參數。

格式

(1) PTPBoost 調整設定值 [, Jump閃避調整] [, Jump接近調整]

(2) PTPBoost

參數

調整設定值
　　以運算式或數值指定調整設定值(0~100的整數)。
Jump閃避調整
　　以運算式或數值指定Jump動作中的Z坐標閃避調整設定值(0~100的整數)。可省略。
Jump接近調整
　　用於以運算式或數值指定Jump動作中的Z坐標接近調整設定值(0~100的整數)。可省略。

結果

若省略參數，則顯示目前PTPBoost設定值。

說明

設定移動微小距離進行PTP動作時的加減速演算法。只在移動距離微小時，本設定值方為有效。以PTPBoostOK函數確認至目標坐標的移動距離是否微小。

PTPBoost在正常使用時不需變更設定值。請僅用於兩種情況：移動微小距離進行動作時想儘量縮短週期時間；即便延長週期時間也想減輕殘留振動。

若增大設定值，雖然會縮短週期時間，但容易發生停止時的振動。另外，若減小設定值，雖然會延長週期時間，但停止時不易發生振動。請注意，若在停止時的振動大的狀態下使用機器人，會產生錯誤和衝擊。這不僅無法充分發揮機器人的功能，還會縮短機械手的使用壽命。

在以下任一情況下，PTPBoost值會被初始化。

- 啟動控制器時
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

參照

PTPBoost函數、PTPBoostOK

PTPBoost範例

```
PTPBoost 50, 30, 30
```

3.19.45 PTPBoost函數

用於傳回指定的PTPBoost值。

格式

PTPBoost (參數編號)

參數

設定編號

以整數設定如下所示的設定值。

- 1：調整設定值
- 2：Jump閃避調整設定值
- 3：Jump接近調整設定值

傳回值

用於傳回調整設定值(0~100的整數值)。

參照

PTPBoost、PTPBoostOK

PTPBoost函數範例

```
Print PTPBoost(1)
```

3.19.46 PTPBoostOK函數

用於傳回從目前位置至目標坐標進行的PTP(point to point)動作是否為微小距離的移動。

格式

PTPBoostOK (目標坐標)

參數

目標坐標

以點數據指定目標坐標。

傳回值

若是從目前位置到目標坐標的PTP動作為微小距離的移動，則傳回「True」；除此之外則傳回「False」。

說明

確認從目前位置到目標坐標的PTP動作是否為可透過PTPBoost命令進行加減速演算法調整的微小距離移動。

參照

PTPBoost

PTPBoostOK函數範例

```
If PTPBoostOK(P1) Then
  PTPBoost 50
EndIf
Go P1
```

3.19.47 PTPTime函數

用於傳回PTP動作命令的估計所需時間。不會進行PTP動作。

格式

(1) PTPTime (目標坐標, 目標手臂, 目標工具)

(2) PTPTime (起始坐標, 開始手臂, 開始工具, 目標坐標, 目標手臂, 目標工具)

參數

開始坐標
　　以點運算式指定開始位置。
目標坐標
　　以點運算式指定目標位置。
目標手臂
　　以整數值或運算式指定目標位置的手臂編號。
目標工具
　　以整數值或運算式指定目標位置的工具編號。
開始手臂
　　以整數值或運算式指定開始位置的手臂編號。
開始工具
　　以整數值或運算式指定開始位置的工具編號。

傳回值

以秒為單位傳回實數值。

說明

可以PTPTime函數估算PTP動作命令(Go)的所需時間。若要估算從目前位置到目標位置的所需時間，請使用格式 (1)；若要估算從開始位置到目標位置的所需時間，請使用格式 (2)。

即便執行此函數，也不進行實際的PTP動作。不會變更目前位置、手臂和工具的設定。

若設定無法到達的位置，或者錯誤設定手臂或工具，傳回值則為0。

對於帶附加軸的機器人，若附加軸由伺服軸構成，也會加入附加軸的動作時間。若附加軸為脈衝輸出軸，則傳回機器人的單獨動作時間。

參照

ATRQ、Go、PTRQ

PTPTime函數範例

```
Real secs

secs = PTPTime(P1, 0, 0, P2, 0, 1)
Print "Time to go from P1 to P2 is:", secs

Go P1
secs = PTPTime(P2, 0, 1)
Print "Time to go from P1 to P2 is:", secs
```

3.19.48 PTran

用於以根據目前位置指定的脈衝量進行一個關節的PTP動作。

格式

PTran 關節編號, 脈衝

參數

關節編號

指定移動關節編號的整數值。附加軸的S軸為8，T軸為9。

脈衝

指定移動的脈衝量。

說明

Ptran用於使根據目前位置指定的脈衝量移動一個關節。

參照

Go、JTran、Jump、Move

Ptran範例

```
PTran 1, 2000
```

3.19.49 PTRQ

用於顯示指定關節的峰值扭矩。

格式
PTRQ [關節編號]

參數

關節編號
以整數值或運算式指定關節編號。可省略。附加軸的S軸為8，T軸為9。

結果
顯示所有關節的目前峰值扭矩。

說明
請於執行PTCLR後執行PTRQ，以顯示關節的峰值扭矩。

以0~1的實數值表示峰值扭矩。

參照
ATRQ、PTCLR、PTRQ函數

PTRQ範例

```
> ptclr
> go p1
> ptrq 1
    0.227
> ptrq
    0.227    0.118
    0.249    0.083
    0.000    0.000
>
```

3.19.50 PTRQ函數

用於傳回指定關節的峰值扭矩。

格式

PTRQ (關節編號)

參數

關節編號

以整數值或運算式指定關節編號。附加軸的S軸為8，T軸為9。

傳回值

以0~1的實數值傳回。

參照

ATRQ、PTCLR、PTRQ

PTRQ函數範例

是在程式中使用PTRQ函數的範例。

```
Function DisplayPeakTorque
   Integer i

   Print "Peak torques:"
   For i = 1 To 4
      Print "Joint ", i, " = ", PTRQ(i)
   Next i
Fend
```

3.19.51 Pulse

透過PTP動作，將機器人手臂移至以各關節脈衝值指定的點。

格式

(1) Pulse 第1關節脈衝值, 第2關節脈衝值, 第3關節脈衝值, 第4關節脈衝值[, 第5關節脈衝值, 第6關節脈衝值] [, 第7關節脈衝值] [, 第8關節脈衝值, 第9關節脈衝值]

(2) Pulse

參數

第1~第4關節脈衝值
　　首先指定4個關節的脈衝值。以整數或Long運算式指定Range命令指定範圍內的值。
第5關節脈衝值、第6關節脈衝值
　　用於垂直六軸機器人(包含N系列)以及關節型6軸機器人。可省略。
第7關節脈衝值
　　用於關節型7軸機器人。可省略。
第8關節脈衝值、第9關節脈衝值
　　用於附加軸。可省略。

結果

在省略參數時，顯示表示目前機器人位置的脈衝值。

說明

Pulse不使用直交坐標系的坐標，而使用從0脈衝位置開始的各關節脈衝值，以表示機器人手臂的位置。Pulse命令用於透過PTP動作運作機器人手臂。

以Range命令設定Pulse命令可使用的上限值和下限值。

注意

- 請在使用Pulse之前確認軌道上沒有障礙物

  Pulse不同於Jump，用於同時運作所有關節(包括第3關節上升以及下降到目標坐標)。因此，使用Pulse時，請充分注意不要讓抓手碰撞到障礙物。

常見錯誤

- 超過限制值的脈衝值

  若以Pulse命令指示的脈衝值超過以Range設定的限制值範圍，則發生錯誤。

參照

Go、Accel、Range、Speed、Pls、Pulse函數

Pulse範例

以下是命令視窗中的操作範例。

將機器人手臂移至以各關節脈衝值指定的位置。

```
> pulse 16000, 10000, -100, 10
```

顯示目前機器人手臂位置的第1至第4關節的脈衝值。

```
> pulse
PULSE:  1:  27306 pls 2:  11378 pls 3:  -3072 pls 4:   1297 pls
>
```

3.19.52 Pulse函數

用於將以脈衝指定的機器人坐標值傳回至各關節。

格式

Pulse (第1關節脈衝值, 第2關節脈衝值, 第3關節脈衝值, 第4關節脈衝值 [, 第5關節脈衝值, 第6關節脈衝值] [, 第7關節脈衝值] [, 第8關節脈衝值, 第9關節脈衝值])

參數

第1~第4關節脈衝值
　　指定第1關節至第4關節的各關節脈衝值。以整數或Long運算式指定Range命令指定範圍內的值。
第5關節脈衝值、第6關節脈衝值
　　用於垂直六軸機器人(包含N系列)以及關節型6軸機器人。可省略。
第7關節脈衝值
　　用於關節型7軸機器人。可省略。
第8關節脈衝值、第9關節脈衝值
　　用於附加軸。可省略。

傳回值

用於傳回以指定脈衝使用的機器人坐標值。

參照

Go、JA、Jump、Move、Pulse、XY

Pulse函數範例

```
Jump Pulse(1000, 2000, 0, 0)
```

3.20 Q

3.20.1 QP

用於設定、解除快速暫停功能並顯示目前的設定。

格式
(1) QP { On | Off }

(2) QP

參數

On | Off
　　指定快速暫停的On(設定)或Off(解除)。

結果
若省略參數，則顯示目前QP設定。

說明
可利用快速暫停功能設定：在執行動作命令時，若按下Pause開關或在控制器上輸入暫停，則立即停止機器人動作，或者等到完成執行動作命令後再停止機器人動作。

將立即減速並停止的操作稱之為「快速暫停」。

若指定參數「On」，QP則用於設定快速暫停。若指定參數「Off」，QP則用於解除快速暫停。

QP用於顯示目前的設定狀態(即機器人對暫停的輸入作出反應而立即停止，或者在完成動作後再停止)。QP是僅用於顯示快速暫停功能的設定或解除狀態之命令。

注意

- 開啟電源時，快速暫停功能為預設狀態

  執行Reset命令後，仍會維持以QP命令設定的快速暫停功能。但是，若在關閉控制器或驅動裝置電源後重新開啟電源，快速暫停功能的設定則會變為「ON(設定)」狀態。

- QP和安全門輸入

  即便解除快速暫停功能，若安全門為「開」，則立即停止機器人動作。

參照
Pause

QP範例
以下命令視窗中的操作範例顯示是否能透過暫停輸入立即停止機器人動作(有無設定快速暫停功能)。

```
> qp
QP ON

> qp on   '設為快速暫停模式
>
```

3.20.2 QPDecelR

針對CP動作時的工具姿態變化，設定快速暫停減速度。

格式

(1) QPDecelR QP減速度設定值

(2) QPDecelR

參數

QP減速度設定值

以實数值指定CP動作的快速暫停時的減速度。(單位：deg/sec^2)

結果

若省略參數，則顯示目前QPDecelR設定值。

說明

在Move、Arc、Arc3、BMove、TMove、Jump3CP上使用ROT修飾參數時，QPDecelR會處於啟用狀態。

在上述動作中執行快速暫停時，有可能會發生關節過加速度錯誤。其原因在於，透過平常的快速暫停動作自動設定的快速暫停減速度超過關節容許減速度。尤其在CP動作的AecelR設定值較大或通過機器人的奇點附近時，最容易發生。發生這種錯誤時，請以QPDecelR將快速暫停減速度設為低值。若QPDecelR的設定值過小，會增加快速暫停所需的移動量，因此請盡可能設定較大值。平常不需設定QPDecelR。

QPDecelR不能用於將減速度設為小於以AccelR設定的CP動作之姿態變化減速度。此時，會發生參數範圍外錯誤。

此外，設定QPDecelR之後，若以AccelR設定大於QP減速度設定值的減速度，QPDecelR則會自動設定與以AccelR設定的減速度相同的QP減速度。

在以下任一情況下，QPDecelR值會被初始化為預設的最大減速度。

- 啟動控制器時
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

參照

QPDecelR函數、QPDecelS、AccelR

QPDecelR範例

以下為用於設定Move命令的QPDecelR之簡易動作程式範例。

```
Function QPDecelTest
  AccelR 3000
QPDecelR 4000
  SpeedR 100
  Move P1 ROT
  .
  .
  .
Fend
```

3.20.3 QPDecelR函數

針對CP動作的工具姿態變化，本函數用於傳回快速暫停減速度的設定值。

格式

QPDecelR

傳回值

針對CP動作的工具變化，傳回快速暫停減速度設定值(實數值，單位：deg/s^2)。

參照

QPDecelR、QPDecelS函數

QPDecelR函數範例

```
Real savQPDecelR

savQPDecelR = QPDecelR
```

3.20.4 QPDecelS

用於設定CP動作時的快速暫停減速度。

格式

(1) QPDecelS QP減速度設定值 [, Jump3閃避QP減速度設定值, Jump3接近QP減速度設定值]

(2) QPDecelS

參數

QP減速度設定值

以實数值指定CP動作的快速暫停時的減速度。(單位：mm/sec^2)

Jump3閃避QP減速度設定值

以實數值指定Jump3閃避動作的快速暫停時的減速度。(單位：mm/sec^2)

Jump3接近QP減速度設定值

以實數值指定Jump3接近動作的快速暫停時的減速度。(單位：mm/sec^2)

結果

若省略參數，則顯示目前QPDecelS設定值。

說明

在CP動作中執行快速暫停時，有可能會發生關節過加速度錯誤。其原因在於，透過平常的快速暫停動作自動設定的快速暫停減速度超過關節容許減速度。尤其在CP動作的AecelS設定值較大或通過機器人的奇點附近時，最容易發生。發生這種錯誤時，請以QPDecelS將快速暫停減速度設為低值。若QPDecelS的設定值過小，會增加快速暫停所需的移動量，因此請盡可能設定較大值。平常不需設定QPDecelS。

QPDecelS不能用於將減速度設為小於以AccelS設定的CP動作減速度。此時，會發生參數範圍外錯誤。

此外，設定QPDecelS之後，若以AccelS設定大於QP減速度設定值的減速度，QPDecelS則會自動設定與以AccelS設定的減速度相同的QP減速度。

在以下任一情況下，QPDecelS值會被初始化為預設的最大減速度。

- 啟動控制器時
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

參照

QPDecelS函數、QPDecelR、AccelS

QPDecelS範例

以下為用於設定Move命令的QPDecelS之簡易動作程式範例。

```
Function QPDecelTest
  AccelS 3000
QPDecelS 4000
  SpeedS 100
  Move P1
  .
  .
  .
Fend
```

3.20.5 QPDecelS函數

本函數用於傳回CP動作的快速暫停減速度設定值。

格式

QPDecelS (設定值編號)

參數

設定值編號

以整數值或運算式指定以下任一值。

- 1：CP動作時的快速暫停減速度設定值
- 2：Jump3、Jump3CP時的閃避動作快速暫停減速度設定值
- 3：Jump3、Jump3CP時的接近動作快速暫停減速度設定值

傳回值

傳回快速暫停減速度設定值(實數值，單位：mm/s^2)。

參照

QPDecelS、QPDecelR函數

QPDecelS函數範例

```
Real savQPDecelS

savQPDecelS = QPDecelS(1)
```

3.20.6 Quit

用於完成執行指定的工作或所有工作。

格式

Quit { 工作識別碼 | All }

參數

任務識別碼

以整數值或運算式指定任務名稱或任務編號。工作名稱用於指定Xqt陳述式所用的函式名稱或者在Run視窗或操作員視窗中啟動的函式。

任務編號的指定(整數)

- 一般工作: 1~32
- 背景工作: 65~80
- Trap工作: 257~267
- All：要完成背景任務以外的所有任務時進行指定。

說明

Quit用於完成目前執行的工作以及以Halt暫停的工作。

指定工作即便是NoPause工作、NoEmgAbort工作(執行Xqt時，指定NoPause或NoEmgAbort開始的特殊工作)、背景工作，Quit也用於完成執行工作。此外，Quit All用於完成執行包括這些工作在內的所有工作(背景工作除外)。

執行Quit All後，對機器人控制參數進行如下設定。

機器人控制參數

- Speed和SpeedR、SpeedS的設定值：(初始化為初始值。)
- Accel和AccelR、AccelS的設定值：(初始化為初始值。)
- QPDecelR、QPDecelS的設定值：(初始化為初始值。)
- LimZ參數的設定值：(初始化為0。)
- CP參數的設定值：(初始化為Off。)
- SoftCP參數的設定值：(初始化為Off。)
- Fine的設定：(初始化為初始值。)
- Power Low設定：(變為低功率模式。)
- PTPBoost的設定值：(初始化為初始值。)
- TCLim、TCSpeed的設定值：(初始化為初始值。)
- PgLSpeed的設定值：(初始化為初始值。)

參照

Exit、Halt、Resume、Xqt

Quit範例

以下是10秒後完成2項工作的範例。

```
Function main
  Xqt winc1   '開始工作winc1
  Xqt winc2   '開始工作winc2
  Wait 10
  Quit winc1  '完成工作winc1
  Quit winc2  '完成工作winc2
Fend
```

```
Function winc1
  Do
     On 1; Wait 0.2
     Off 1; Wait 0.2
  Loop
Fend

Function winc2
  Do
     On 2; Wait 0.5
     Off 2; Wait 0.5
  Loop
Fend
```

3.21 R

3.21.1 RadToDeg函數

用於將弧度轉換為角度。

格式
RadToDeg (弧度)

參數

弧度
以實數值指定要轉換為角度的弧度。

傳回值
用於傳回表示角度的Double型值。

參照
ATan、ATan2、DegToRad函數

RadToDeg函數範例

```
s = Cos(RadToDeg(x))
```

3.21.2 Randomize

用於執行亂數系列的初始化。

格式

(1) Randomize Seed值

(2) Randomize

參數

Seed值

以0以上的實數值指定用於求出亂數的基值。

參照

Rnd函數

Randomize範例

```
Function main
    Real r
    Randomize
    Integer randNum

    randNum = Int(Rnd(10)) + 1
    Print "Random number is:", randNum
Fend
```

3.21.3 Range

用於設定和顯示各伺服關節的容許動作區域。

格式

(1) Range 設定值1, 設定值2, 設定值3, 設定值4, 設定值5, 設定值6, 設定值7, 設定值8 [, 設定值9, 設定值10, 設定值11, 設定值12] [, 設定值13, 設定值14] [, 設定值15, 設定值16, 設定值17, 設定值18]

(2) Range

參數

設定值1
　　第1關節的下限脈衝值(單位：脈衝)
設定值2
　　第1關節的上限脈衝值(單位：脈衝)
設定值3
　　第2關節的下限脈衝值(單位：脈衝)
設定值4
　　第2關節的上限脈衝值(單位：脈衝)
設定值5
　　第3關節的下限脈衝值(單位：脈衝)
設定值6
　　第3關節的上限脈衝值(單位：脈衝)
設定值7
　　第4關節的下限脈衝值(單位：脈衝)
設定值8
　　第4關節的上限脈衝值(單位：脈衝)
設定值9
　　第5關節的下限脈衝值(單位：脈衝)專用於垂直六軸機器人(包含N系列)以及關節型6軸機器人的參數。
設定值10
　　第5關節的上限脈衝值(單位：脈衝)專用於垂直六軸機器人(包含N系列)以及關節型6軸機器人的參數。
設定值11
　　第6關節的下限脈衝值(單位：脈衝)專用於垂直六軸機器人(包含N系列)以及關節型6軸機器人的參數。
設定值12
　　第6關節的上限脈衝值(單位：脈衝)專用於垂直六軸機器人(包含N系列)以及關節型6軸機器人的參數。
設定值13
　　第7關節的下限脈衝值(單位：脈衝)專用於關節型7軸機器人的參數。
設定值14
　　第7關節的上限脈衝值(單位：脈衝) 專用於關節型7軸機器人的參數。
設定值15
　　第8關節的下限脈衝值(單位：脈衝) 專用於附加軸S關節的參數。
設定值16
　　第8關節的上限脈衝值(單位：脈衝) 專用於附加軸S關節的參數。
設定值17
　　第9關節的下限脈衝值(單位：脈衝) 專用於附加軸T關節的參數。
設定值18
　　第9關節的上限脈衝值(單位：脈衝) 專用於附加軸T關節的參數。

結果

若省略參數，則顯示目前Range值。

說明

Range用於設定各馬達關節的下限/上限脈衝值。單位是脈衝。藉此，使用者可定義各關節的最大及最小容許動作範圍。同樣，以XYLim命令定義XY坐標方向的限制值。

Range的初始值因機器人的機種而異。即便關閉電源，仍儲存以此命令設定的值。

在省略參數時，顯示目前Range值。

常見錯誤

- 欲移出容許範圍時

  即便是1個關節，若欲將機器人手臂移出容許動作範圍，則發生錯誤。

- 關節動不了

  下限脈衝值大於等於上限脈衝值時，關節不會進行動作。

注意

第6關節的上限脈衝值和下限脈衝值的設定範圍因機種而異。
有關設定範圍的更多資訊，請參閱各機械臂手冊的動作區域。

參照

JRange、SysConfig、XYLim

Range範例

以下是命令視窗中顯示和變更目前Range設定值的簡易範例。

```
> range
-18205, 182045, -82489, 82489, -36864, 0, -46695, 46695
>
> range 0, 32000, 0, 32224, -10000, 0, -40000, 40000
>
```

3.21.4 Read

從檔案或通訊連接埠載入指定的字元數。

格式

Read #連接埠編號, 字串變數$, 字元數

參數

連接埠編號

表示檔案或通訊連接埠的ID編號。檔案編號是以ROpen、WOpen、AOpen等陳述式指定的編號。通訊連接埠編號是以OpenCom(RS-232C)或OpenNet(TCP/IP)陳述式指定的編號。

字串變數$,

指定接收字串的字串變數名稱。

字元數

指定載入的位元組數。

注意

- 支援的控制器型號

  如果在T/VT系列中指定RS-232C埠時會發生錯誤。

參照

ChkCom、ChkNet、OpenCom、OpenNet、Write、ReadBin

Read範例

```
Integer numOfChars
String data$

numOfChars = ChkCom(1)

If numOfChars > 0 Then
    Read #1, data$, numOfChars
EndIf
```

3.21.5 ReadBin

從檔案或通訊連接埠載入二進位資料。

格式

ReadBin #連接埠編號, 變數名稱

ReadBin #連接埠編號, 陣列變數名稱(), 位元組數

參數

連接埠編號
　　表示檔案或通訊連接埠的ID編號。檔案編號是以BOpen等陳述式指定的編號。通訊連接埠編號是以OpenCom(RS-232C)或OpenNet(TCP/IP)陳述式指定的編號。
變數名稱
　　指定接收資料位元組的Byte型變數、整數變數或Long型變數之名稱。
陣列變數名稱()
　　指定接收資料位元組的Byte型變數、整數變數或Long型變數之名稱。可指定的是一維的陣列變數。
位元組數
　　指定載入的位元組數。必須小於陣列的最大元素編號且小於256 Byte。若以通訊連接埠(TCP/IP)為適用對象時，必須小於陣列的最大元素編號且小於1024 Byte。

參照

Write、WriteBin、Read

ReadBin範例

```
Integer data
Integer dataArray(10)

numOfChars = ChkCom(1)

If numOfChars > 0 Then
    ReadBin #1, data
EndIf

numOfChars = ChkCom(1)

If numOfChars > 10 Then
    ReadBin #1, dataArray(), 10
EndIf
```

3.21.6 Real

用於宣告Real型變數。(4位元組的實數值)

格式

Real 變數名稱 [(陣列變數的最大元素編號)] [, 變數名稱 [(陣列變數的最大元素編號)]...]

參數

變數名稱

指定宣告為實體型的變數名稱。

陣列變數的最大元素編號

是陣列變數的最大元素編號，最大可宣告三維。使用如下格式。可省略。

(最大元素編號1, [最大元素編號2], [最大元素編號3])

元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。在所有元素數不超過以下最大值的範圍內，指定各最大元素編號。- 本地變數：2,000 - 備份變數(Global Preserve)：4,000 - 全域變數和模組變數：100,000

說明

Real用於宣告Real型變數。在Function的開頭宣告本地變數。在Function之外宣告全域變數和模組變數。

Real型的有效位數為6位。

參照

Boolean、Byte、Double、Global、Int32、Int64、Integer、Long、Short、String、UByte、UInt32、UInt64、UShort

Real範例

以下範例是使用Real宣告Real型變數的程式。

```
Function realtest
  Real var1
  Real A (10)          'Real型的一維陣列
  Real B (10, 10)      'Real型的二維陣列
  Real C (5, 5, 5)     'Real型的三維陣列
  Real arrayVar(10)
  Integer i
  Print "Please enter a Real Number:"
  Input var1
  Print "The Real variable var1 = ", var1
  For i = 1 To 5
    Print "Please enter a Real Number:"
    Input arrayVar(i)
    Print "Value Entered was ", arrayVar(i)
  Next i
Fend
```

3.21.7 RealAccel函數

本函數用於傳回OLAccel調整後的加減速度設定值。

格式

RealAccel (設定值編號)

參數

設定值編號

以整數值指定以下各值。

- 1：加速設定值
- 2：減速設定值
- 3：Jump動作時的閃避加速設定值
- 4：Jump動作時的閃避減速設定值
- 5：Jump動作時的接近加速設定值
- 6：Jump動作時的接近減速設定值

傳回值

用於傳回1以上的整數(%)。

用途

透過使用RealAccel能瞭解機器人可進行連續動作的最大加減速度。

程序如下所述。

1. 在啟用OLAccel命令的狀態下運作機器人。
2. 執行OLRate命令，確認過載率是否上升。
3. 過載率上升到0.5以上時，將開始自動調整加減速度。
4. 請於經過一定時間後執行OLRate命令，確認過載率不上升。
5. 確認過載率不上升後，執行RealAccel函數。
6. RealAccel函數的傳回值是機器人可透過 (1) 的動作連續進行動作的最大加減速度。
   - 若當過載率正在上升時使用RealAccel函數，則無法瞭解機器人可連續動作的最大加減速度。
   - 發生過熱錯誤時，即便執行上述程序，也無法瞭解機器人可連續動作的最大加減速度。

參照

Accel、OLAccel、OLRate

RealAccel函數範例

是在程式中使用RealAccel函數的範例。

```
Integer RealAccel1, RealDecel1

Accel 100, 100
OLAccel on

'取得目前加減速度
RealAccel1 = RealAccel (1)
RealDecel1 = RealAccel (2)

顯示目前加速度
Print RealAccel1
顯示目前減速度
Print RealDecel1
```

3.21.8 RealPls函數

用於傳回指定關節的脈衝值。

格式

RealPls (關節編號)

參數

關節編號

指定要傳回目前脈衝值的關節。附加軸的S軸為8，T軸為9。

傳回值

用於以整數值傳回以關節編號指定的關節之目前編碼器脈衝值。

說明

以RealPls函數傳回各關節的編碼器目前位置或脈衝值。可儲存並以Pulse命令使用該值。

參照

CX、CY、CZ、CU、CV、CW、Pulse

RealPls函數範例

```
Function DisplayPulses

  Long joint1Pulses

  joint1Pulses = RealPls(1)
  Print "Joint 1 Current Pulse Value: ", joint1Pulses
Fend
```

3.21.9 RealPos函數

用於傳回指定機器人的目前位置。

格式

RealPos

傳回值

用於傳回指定機器人目前位置的點。

說明

RealPos函數用於傳回機器人的目前位置。

參照

CurPos、CX、CY、CZ、CU、CV、CW、RealPls

RealPos函數範例

```
Function ShowRealPos

  Print RealPos
Fend

P1 = RealPos
```

3.21.10 RealTorque函數

用於傳回指定關節的目前扭矩命令值。

格式

RealTorque (關節編號)

參數

關節編號

以運算式或數值指定欲讀取扭矩命令值的關節編號。附加軸的S軸為8，T軸為9。

傳回值

以-1~1的實數值傳回目前功率模式之對於最大扭矩的比例。

正值表示關節角的正方向；負值表示關節角的負方向。

參照

TC、TCSpeed、TCLim

RealTorque函數範例

```
Print "目前的Z軸扭矩命令值(SCARA機器人)：", RealTorque(3)
```

3.21.11 Recover

用於執行朝開啟安全門時的位置之復歸動作，並傳回狀態。

本命令為適合高階專業人員使用的命令。請充分理解命令內容後使用。

格式

(1) Recover 機器人編號 | All

(2) Recover 機器人編號 | All , WithMove | WithoutMove

參數

機器人編號
　　指定執行復歸動作的機器人編號。
All
　　執行所有機器人的復歸動作。省略時則變成All。
WithMove
　　值0的常數。
　　恢復勵磁並移至打開安全門時的位置。省略時則變成WithMove。
WithoutMove
　　值1的常數。
　　只用於恢復勵磁。平常不使用。與AbortMotion搭配使用以實現特殊復歸。

說明

若要在程式中執行本命令，需勾選Epson RC+的[設置] - [系統配置] - [控制器] - [選項]的[啟動進階任務命令]的核取方塊。

Recover用於在關閉安全門後重新開啟馬達，並以低功率的PTP動作，讓機器人回到打開安全門時的位置。完成復歸動作後，可用Cont或ContManualRecover繼續週期。

若控制器設有多台機器人並指定All，則依機器人編號從小到大的順序執行復歸動作。

參照

AbortMotion, Cont, Recover函數, RecoverPos, ContManualRecover

Recover範例

> ⚠ 注意
>
> 若要在程式中執行Recover命令，請理解命令內容，並確認可作為系統進行復歸動作的條件皆已備齊。若透過迴圈繼續執行命令等進行錯誤的使用，則有可能導致系統安全性下降。請充分注意。

```
Function main
    Xqt 2, monitor, NoPause
    Do
        Jump P1
        Jump P2
    Loop
Fend

Function monitor
    Do
        If Sw(SGOpenSwitch) = On then
            Wait Sw(SGOpenSwitch) = Off and Sw(RecoverSwitch) = On
            Recover All
        EndIf
```

```
    Loop
Fend
```

3.21.12 Recover函數

用於執行朝開啟安全門時的位置之復歸動作，並傳回狀態。

本命令為適合高階專業人員使用的命令。請充分理解命令內容後使用。

格式

(1) Recover

(2) Recover (機器人編號 | All)

(3) Recover (機器人編號 | All , WithMove | WithoutMove)

參數

機器人編號
　　指定執行復歸動作的機器人編號。若省略機器人編號，則所有機器人皆執行復歸動作。
All
　　執行所有機器人的復歸動作。省略時則變成All。
WithMove
　　值0的常數。
　　恢復勵磁並移至打開安全門時的位置。省略時則變成WithMove。
WithoutMove
　　值1的常數。
　　只用於恢復勵磁。平常不使用。與AbortMotion搭配使用以實現特殊復歸。

傳回值

用於傳回Boolean型的值。若完成復歸動作，即傳回「True」。若未完成，則傳回「False」。

說明

若要在程式中執行本命令，需勾選Epson RC+的[設置] - [系統配置] - [控制器] - [選項]的[啟動進階任務命令]的核取方塊。

Recover用於在關閉安全門後重新開啟馬達，並以低功率的PTP動作，讓機器人回到打開安全門時的位置。完成復歸動作後，可用Cont或ContManualRecover繼續週期。若毫無問題地完成復歸動作，則傳回「True」。在復歸動作中若執行暫停、中斷或打開安全門，Recover則傳回「False」。

若控制器設有多台機器人並指定All，則依機器人編號從小到大的順序執行復歸動作。

> ⚠ 注意
>
> 若要在程式中執行Recover命令，請理解命令內容，並確認可作為系統進行復歸動作的條件皆已備齊。若透過迴圈繼續執行命令等進行錯誤的使用，則有可能導致系統安全性下降。請充分注意。

參照

AbortMotion, Cont, Recover, RecoverPos, ContManualRecover

Recover函數範例

```
Boolean sts
Integer answer

sts = Recover
If sts = True Then
MsgBox "Ready to continue", MB_ICONQUESTION + MB_YESNO, "MyProject", answer
If answer = IDYES Then
  Cont
```

```
EndIf
EndIf
```

3.21.13 RecoverPos函數

用於傳回開啟安全門時的位置。

本命令為適合高階專業人員使用的命令。請充分理解命令內容後使用。

格式

RecoverPos ([機器人編號])

參數

機器人編號

以整數值指定機器人編號。省略機器人編號時，則以目前選擇的機器人為對象。

結果

用於傳回開啟安全門時的位置。

未打開安全門或機器人完成復歸動作時，傳回X到W的值中有0的點資料。

說明

傳回以Cont或Recover執行復歸動作時的機器人復歸位置。

參照

AbortMotion、Cont、Recover、Recover函數、RealPos

RecoverPos函數範例

復歸動作的直線距離未滿10mm時，則執行復歸動作；高於該值時，則終止程式。

```
If Dist(RecoverPos, RealPos) < 10 Then
Recover All
Else
Quit All
EndIf
```

3.21.14 Redim

用於在執行階段變更陣列的最大元素編號。

格式

Redim [Preserve] 陣列名稱(陣列變數的最大元素編號)

參數

Preserve

儲存陣列以前的值。可省略。省略時即清除陣列。

陣列名稱

指定陣列變數的名稱。依據平常的變數名稱慣例進行指定。務必事先宣告陣列。

陣列變數的最大元素編號

指定陣列變數的新最大元素編號。請賦予與宣告變數時相同數量的最大元素編號。使用如下格式。
(最大元素編號1, [最大元素編號2], [最大元素編號3])
元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。
在所有元素數不超過以下最大值的範圍內，指定各最大元素編號。

	String型以外	String型
本地變數	2,000	200
備份變數(Global Preserve)	4,000	400
全域變數和模組變數	100,000	10,000

說明

Redim用於在執行階段變更陣列變數的最大元素編號。若要儲存前一個值，則指定Preserve。

以Byref指定的陣列變數不可用於Redim。

頻繁執行Redim會降低程式的執行速度。尤其，建議對備份變數執行最小限度的Redim時。

參照

UBound

Redim範例

```
Integer i, numParts, a(0)

Print "Enter number of parts "
Input numParts

Redim a(numParts)

For i = 0 to UBound(a)
    a(i) = i
Next

'新增20個最大元素編號
Redim Preserve a(numParts + 20)

'維持最初的元素編號
For i = 0 to UBound(a)
    Print a(i)
Next
```

3.21.15 Rename

用於變更檔名。

格式

Rename 變更來源檔名, 變更目的地檔名

參數

變更來源檔案名稱

指定要變更名稱的檔案名稱和路徑的字串運算式。詳細內容請參閱ChDisk。

變更目的地檔案名稱

在以變更來源檔案名稱指定的檔案中指定新的名稱。詳細內容請參閱ChDisk。

說明

將以變更來源檔名指定的檔名變更為以變更目的地檔名指定的名稱。若省略路徑，則此陳述式在目前目錄尋找變更來源檔名。

唯有在相同驅動器內指定變更來源檔名和變更目的地檔名時方可執行。

不可變更為與存在於相同路徑的其它檔案相同的名稱。

作為萬用字元之用的字元不可用於變更來源檔名、變更目的地檔名的參數。

參照

Copy

Rename範例

命令視窗中的執行範例

```
> Rename A.PRG  B.PRG
```

3.21.16 RenDir

變更目錄名稱。

格式

Rendir 變更來源目錄名稱, 變更目的地目錄名稱

參數

變更來源目錄名稱

以字串運算式指定要變更的目錄之路徑和目錄名稱。

變更目的地目錄名稱

以字串運算式指定變更後的目錄路徑和目錄名稱。路徑的詳細內容請參閱ChDisk。

說明

變更目的地目錄的路徑必須在變更來源目錄的路徑中。

若省略變更來源目錄名稱、變更目的地目錄名稱的路徑，並僅指定目錄名稱，則指定目前目錄中的目錄。

作為萬用字元之用的字元不可用於變更來源目錄名稱、變更目的地目錄名稱的參數。

注意

磁碟為PC時，可執行。

參照

MkDir

RenDir範例

```
RenDir "c:\mydata", "c:\mydata1"
```

3.21.17 Reset

用於將控制器重設為初始狀態。

格式

(1) Reset

(2) Reset Error

說明

- Reset用於進行以下重設。
- Reset Error用於完成所有一般工作，只對錯誤狀態和機器人控制參數進行重設。
- 若要在程式中執行Reset Error命令，需勾選Epson RC+的[設定] - [系統設定] - [控制器] - [環境設定]的[啟用進階工作控制命令]的核取方塊。
  - 緊急停止狀態(僅限Reset時)
  - 錯誤狀態
  - 輸出位元(僅限Reset時)：除了被分配為遠端輸出的I/O和分配為末端夾具的之外，停用所有輸出位元。可透過Epson RC+解除此功能。
  - 機器人控制參數
    - Speed和SpeedR、SpeedS的設定值：(初始化為初始值。)
    - Accel和AccelR、AccelS的設定值：(初始化為初始值。)
    - QPDecelR、QPDecelS的設定值：(初始化為初始值。)
    - LimZ參數的設定值：(初始化為0。)
    - CP參數的設定值：(初始化為Off。)
    - SoftCP參數的設定值：(初始化為Off。)
    - Fine的設定：(初始化為初始值。)
    - Power Low設定：(變為低功率模式。)
    - PTPBoost的設定值：(初始化為初始值。)
    - TCLim、TCSpeed的設定值：(初始化為初始值。)
    - PgLSpeed的設定值：(初始化為初始值。)

在發生伺服相關錯誤、緊急停止狀態以及處於其它需重設的狀態下，無法接收Reset以外的命令。此時，請先執行Reset，然後執行其它所需處理。

舉例來說，在緊急停止後首先要確認周圍狀況和操作安全，然後執行Reset。其後，請執行Motor On。

無法透過Reset解除重大錯誤狀態。

發生重大錯誤時，請關閉控制器電源並排除錯誤原因。

無法在背景工作以及以Trap Emergency、Trap Error啟動的工作上執行Reset命令。不可透過程式解除緊急停止狀態。

注意

- [透過Reset將輸出連接埠設爲Off]核取方塊

  勾選Epson RC+的 - [設定] - [系統設定] - [環境設定]的[透過Reset停用輸出連接埠]核取方塊時，若發行Reset命令，則停用所有除了設定成末端夾具以外的輸出。藉此設定停用輸出時，為避免發生工具掉落或類似情況，務必考量配線。

有關末端夾具的更多資訊，請參閱以下手冊。
「Hand功能」

參照

Accel、AccelS、Fine、LimZ、Motor、Off、On、PTPBoost、SFree、SLock、Speed、SpeedS

Reset範例

以下是在命令視窗執行Reset命令的範例。

```
>reset
>
```

3.21.18 ResetElapsedTime

用於重設ElapsedTime函數所使用的生產節拍時間測量用計時器。

格式

ResetElapsedTime

說明

重設並啟動生產節拍時間測量用計時器。

參照

ElapsedTime函數

ResetElapsedTime範例

```
ResetElapsedTime        '重設生產節拍時間測量用計時器
For i = 1 To 10         '執行10次
    GoSub Cycle
Next
Print ElapsedTime / 10 '計算並顯示生產節拍時間
```

3.21.19 Restart

重新執行目前的主程式。

本命令為適合高階專業人員使用的命令。請充分理解命令內容後使用。

格式
Restart

說明
Restart用於中斷所有執行中的工作，並重新執行最後執行的主程式。

在不中斷背景工作之狀態下繼續執行。

由於解除所有Trap設定，因此即便以本命令中斷工作，也不執行Trap Abort。

透過執行本命令解除Pause狀態。

若在錯誤狀態下執行本命令，則會發生錯誤。首先，請以Reset Error命令等解除錯誤。

若在緊急停止狀態下執行本命令，則會發生錯誤。不可透過程式解除緊急停止狀態。

> ⚠ 注意
> 若要在程式中執行Restart命令，請理解命令內容，並確認可作為系統繼續執行的條件皆已備齊。若透過迴圈繼續執行命令等進行錯誤的使用，則有可能導致系統安全性下降。請充分注意。

注意

使用遠程I/O控制時，請不要同時執行SPEL+程式的Restart命令和遠端輸入的start訊號。會使程式雙重運行並可能發生2503錯誤。

參照
Quit、Reset、Trap、Xqt

Restart範例

```
Function main
    Trap Error Xqt eTrap
    Motor On
    Call PickPlac
Fend

Function eTrap

Wait Sw(ERresetSwitch)
Reset Error
Wait Sw(RestartSwitch)
Restart
Fend
```

3.21.20 Resume

透過Halt命令繼續執行暫停的工作。

格式

Resume { 工作識別碼 | All }

參數

任務識別碼

以整數值或運算式指定任務名稱或任務編號。工作名稱用於指定Xqt陳述式所用的函式名稱或者在Run視窗或操作員視窗中啟動的函式。

任務編號的指定(整數)

- 一般任務：1~32
- 背景任務：65~80
- Trap任務：257~267

All

要繼續執行所有任務時進行指定。

說明

Resume用於以Halt命令繼續執行暫停的工作。

參照

Halt、Quit、Xqt

Resume範例

以下是在Halt命令之後使用Resume命令的範例。

```
Function main
   Xqt 2, flicker    '在工作2執行flicker

   Do
      Wait 3         '執行flicker3秒鐘
      Halt flicker   '停止flicker工作
      Wait 3
      Resume flicker '暫停flicker工作
   Loop
Fend

Function flicker
   Do
      On 1
      Wait 0.2
      Off 1
      Wait 0.2
   Loop
Fend
```

3.21.21 Return

Return陳述式與GoSub陳述式搭配使用。GoSub用於將程式控制移至副程式。完成副程式後，以Return繼續在開始執行副程式的GoSub命令的下一行執行程式。

格式

Return

說明

Return陳述式與GoSub陳述式搭配使用。Return陳述式的主要目的在於，讓程序控制返回至將控制轉移到副程式GoSub命令之後的命令。

GoSub命令用於將程式控制分支到使用者指定的陳述式行或標籤。程式用於執行該轉移目的地的行列和後續行，直到發出Return命令。Return命令用於讓程式控制返回到指示移往副程式的GoSub的下一行列。(總之，GoSub命令用於執行副程式，以Return返回到GoSub命令的下一陳述式。)

常見錯誤

- 沒有GoSub却使用Return時

  Return命令用於從副程式返回到發行GoSub的原程式。若沒有GoSub卻使用Return命令，則發生錯誤2383。由於系統無法判斷返回位置，因此單獨使用Return命令沒有意義。

參照

OnErr、GoSub、GoTo

Return範例

以下是以GoSub命令分支為checkio標籤，然後檢查前16個使用者輸入的簡易範例。之後，從副程式返回到主程式。

```
Function main
     Integer var1, var2
     GoSub checkio
     On 1
     On 2
     Exit Function

checkio:      '副程式的開始位置
     var1 = In(0)
     var2 = In(1)
     If var1 <> 0 Or var2 <> 0 Then
          Print "Message to Operator here"
     EndIf
finished:
     Return  '副程式的結束位置 返回到第40行
Fend
```

3.21.22 Right$函數

本函數用於從字串的末尾取出指定數的字串。

格式

Right$ (字串, 字元數)

參數

字串

從末尾取出指定數的字串。指定最長255個字元的字串運算式或字串。

字元數

以運算式或數值指定從字串末尾複製的字元數(正整數)。

傳回值

用於從指定字串的末尾取出並傳回指定數字串。

說明

Right$用於從指定字串的末尾傳回指定數的字串。Right$可用於直接傳回存在於字串的字元數。

參照

Asc、Chr$、InStr、Left$、Len、Mid$、Space$、Str$、Val

Right$範例

以下是輸入工件的資料字串即可傳回工件的零件編號、名稱、零件計數等資訊之程式範例。

```
Function SplitPartData(DataIn$ As String, ByRef PartNum$ As String, ByRef PartName$
As String, ByRef PartCount As Integer)

    PartNum$ = Left$(DataIn$, 10)

    DataIn$ = Right$(datain$, Len(DataIn$) - pos)
    pos = Instr(DataIn$, ",")

    PartName$ = Mid$(DataIn$, 11, 10)

    PartCount = Val(Right$(dataIn$, 5))

Fend
```

此外，以下是命令視窗中的操作範例。

```
> Print Right$("ABCDEFG", 2)
 FG

> Print Right$("ABC", 3)
 ABC
```

3.21.23 RmDir

刪除子目錄。

格式

Rmdir 目錄名稱

參數

目錄名稱

以字串運算式指定要刪除的目錄之路徑和名稱。若省略路徑，而僅指定目錄名稱，則指定目前目錄中的子目錄。路徑的詳細內容請參閱ChDisk。

說明

刪除指定的子目錄。執行此陳述式前，務必刪除子目錄內的檔案。

不可刪除目前目錄及父目錄。

在命令視窗中執行時，可省略引號。

注意

磁碟為PC時，可執行。

Rmdir範例

命令視窗中的執行範例

```
> RmDir \mydata
```

3.21.24 Rnd函數

用於傳回亂數。

格式
Rnd (最大值)

參數

最大值
以實數值設定最大傳回值。

傳回值
用於傳回0~以參數指定的最大值的實數值亂數。

說明
用於生成亂數。

參照
Int、Randomize

Rnd函數範例
是生成1~10的亂數之範例。

```
Function main
    Real r
    Integer randNum

    Randomize
    randNum = Int(Rnd(9)) + 1
    Print "Random number is:", randNum
Fend
```

3.21.25 Robot

選擇機器人。

格式

Robot 機器人編號

參數

機器人編號
　　指定機器人編號。範圍是1~設置的機器人數。

說明

Robot陳述式用於選擇執行動作命令的下一個機器人。

若為1台機器人，不需使用Robot陳述式。

參照

Accel、AccelS、Arm、ArmSet、Go、Hofs、Home、HOrdr、Local、Move、Pulse、Robot函數、Speed、SpeedS

Robot範例

```
Function main
    Integer I
    For I = 1 to 100
        Robot 1
        Go P(i)
        Robot 2
        Go P(i)
    Next I
Fend
```

3.21.26 Robot函數

用於傳回目前的機器人編號。

格式

Robot

傳回值

用於傳回目前機器人編號的整數值。

參照

Robot

Robot函數範例

```
Print "The current robot is: ", Robot
```

3.21.27 RobotInfo函數

用於傳回機器人的狀態資訊。

格式

RobotInfo (指數)

參數

指數

以整數值指定所要查找的資訊指數。

傳回值

用於傳回指定資訊的整數值。

說明

下表所示為傳回值的位元資訊。

指數	位元	值	說明
0	0	&H1	未定義
	1	&H2	發生可重設的錯誤
	2	&H4	發生不可重設的錯誤
	3	&H8	馬達ON
	4	&H10	High功率
	5	&H20	未定義
	6	&H40	未定義
	7	&H80	未定義
	8	&H100	機器人呈Halt狀態
	9	&H200	機器人未呈Halt狀態(動作中或快速暫停中)
	10	&H400	因暫停或安全門打開而停止機器人
	11		未定義
	12		未定義
	13		未定義
	14	&H4000	發出動作命令後符合TILL條件
	15	&H8000	發出動作命令後符合SENSE條件
	16-31		未定義
1	0	&H1	追蹤動作中(輸送帶追蹤中)
	1	&H2	等待復歸動作(WaitRecover狀態)
	2	&H4	復歸動作執行中
	3-31		未定義

指數	位元	值	說明
2	0	&H1	機器人位於Home位置
	1-31		未定義
3	0	&H1	第1關節伺服勵磁中
	1	&H2	第2關節伺服勵磁中
	2	&H4	第3關節伺服勵磁中
	3	&H8	第4關節伺服勵磁中
	4	&H10	第5關節伺服勵磁中
	5	&H20	第6關節伺服勵磁中
	6	&H40	第7關節伺服勵磁中
	7	&H80	S關節伺服勵磁中
	8	&H100	T關節伺服勵磁中
	9-31		未定義
4	N/A	0 - 32 -1	正在執行機器人命令的任務編號 ▪ 0 = 從命令視窗或巨集指令執行命令 ▪ -1 =未使用任務或機械臂
5	0	&H1	第1關節制動器ON
	1	&H2	第2關節制動器ON
	2	&H4	第3關節制動器ON
	3	&H8	第4關節制動器ON
	4	&H10	第5關節制動器ON
	5	&H20	第6關節制動器ON
	6	&H40	第7關節制動器ON
	7	&H80	S關節制動器ON
	8	&H100	T關節制動器ON
	9-31		未定義

參照

CtrlInfo、RobotInfo$、TaskInfo

RobotInfo函數範例

```
If (RobotInfo(3) And &H1) = &H1 Then
  Print "Joint 1 is locked"
Else
  Print "Joint 1 is free"
EndIf
```

3.21.28 RobotInfo$函數

用於傳回機器人的文字資訊。

格式

RobotInfo$ (指數)

參數

指數

以整數值指定所要查找的資訊指數。

傳回值

用於傳回指定資訊的字串。

說明

- 0：機器人名稱
- 1：型號
- 2：預設點檔案名稱
- 3：未定義
- 4：機器人序號

參照

CtrlInfo、RobotInfo、TaskInfo

RobotInfo$函數範例

```
Print "Robot Name: ", RobotInfo$(0)
```

3.21.29 RobotModel$函數

用於傳回機器人的型號。

格式

RobotModel$

傳回值

用於傳回型號字串。這是記載於機器人後方面板上的機器人名稱。

參照

RobotType

RobotModel$函數範例

```
Print "The robot model is ", RobotModel$
```

3.21.30 RobotName$函數

用於傳回機器人名稱。

格式

RobotName$

傳回值

用於以字串傳回機器人名稱。

參照

RobotInfo、RobotModel$

RobotName$函數範例

```
Print "The robot name is ", RobotName$
```

3.21.31 RobotSerial$函數

用於傳回機器人序號。

格式

RobotSerial$

傳回值

用於以字串傳回機器人序號。

參照

RobotInfo、RobotName$、RobotModel$

RobotSerial$函數範例

```
Print "The robot serial number is ", RobotSerial$
```

3.21.32 RobotType函數

用於傳回機器人類型。

格式

RobotType

傳回值

- 1：關節型
- 2：直角坐標型
- 3：水平多關節型
- 5：垂直六軸
- 6：RS系列
- 7：N系列

參照

RobotModel$

RobotType函數範例

```
If RobotType = 3 Then
  Print "Robot type is SCARA"
EndIf
```

3.21.33 ROpen

用於以唯讀模式開啟檔案。

格式

ROpen 檔名As #檔案編號 . . Close #檔案編號

參數

檔案名稱

指定包含路徑的檔案名稱字串。僅指定檔名時，即指目前目錄中的檔案。詳細內容請參閱ChDisk。

檔案編號

以30~63的整數值或運算式來指定。

說明

以指定檔案編號開啟指定的檔案。此陳述式用於從指定檔案讀出資料。

注意

- 僅限於PC磁碟。
- 可使用網路路徑。

在檔案處於開啟狀態下，指定的檔案編號用於識別該檔案。因此，在關閉該檔案之前，不可將相同的檔案編號用於其它檔案。在檔案操作命令 (Input#, Read, Seek, Eof, Close)中使用檔案編號。

以Close陳述式關閉檔案，並釋放檔案編號。

請以FreeFile函數取得檔案編號，以避免將同一編號用在多項工作上。

參照

Close、Input #、AOpen、BOpen、UOpen、WOpen、FreeFile

ROpen範例

```
Integer fileNum, i, j

fileNum = FreeFile
WOpen "TEST.DAT" As #fileNum
For i = 0 To 100
    Print #fileNum, i
Next i
Close #fileNum

fileNum = FreeFile
ROpen "TEST.DAT" As #fileNum
For i = 0 to 100
    Input #fileNum, j
    Print "data = ", j
Next i
Close #fileNum
```

3.21.34 ROTOK函數

用於傳回至目標坐標的動作命令時，可否附加ROT修飾參數。

格式

ROTOK (目標坐標)

ROTOK (標準坐標, 目標坐標)

參數

目標坐標

以點數據指定用於調查可否附加ROT修飾參數的目標坐標。

基準坐標

以點數據指定用於調查可否附加ROT修飾參數的基準坐標。

傳回值

如可附加ROT修飾參數，則傳回「True」，除此之外則傳回「False」。

注意

- 支援的控制器型號

  不支援T/VT系列。

說明

在實際運作(機器人)前確認可否附加ROT修飾參數。

ROT修飾參數為可附加於Move等直線內插動作命令，並指定以工具姿態變化的加減速為優先的參數。

ROTOK函數用於從標準坐標Move至目標坐標等，以直線內插動作命令移動時，於動作前判定附加ROT是否會發生錯誤。

若省略標準坐標，則將以目前位置(Here)為標準，傳回判定結果。

若姿態變化角度為「0」或為微小變化，則判定若附加ROT修飾參數會發生的錯誤。但是，無法判定於動作中機器人的關節速度及關節加速度，是否超過機械手的限度，如錯誤4242等。屆時，請執行調低SpeedS、SpeedR、AccelS、AccelR的各值等調整。

參照

Move

ROTOK函數範例

```
If ROTOK(P1) = True Then
  Move P1 ROT
Else
  Move P1
EndIf
```

3.21.35 RSet$函數

用於在字串的開頭加入半型空格，並傳回指定長度的字串。

格式

RSet$ (字串, 字串的長度)

參數

字串
　　指定字串運算式。
字串的長度
　　指定整數或運算式，以表示要傳回的字串長度。

傳回值

用於傳回在開頭新增半型空格的指定字串。

參照

LSet$、Space$

RSet$函數範例

```
temp$ = "123"
temp$ = RSet$(temp$, 10)   ' temp$ = "       123"
```

3.21.36 RShift函數

用於數值資料的邏輯右移

格式
RShift (數值資料, 移位位元數)

參數

數值資料
　　以運算式或數值指定要進行邏輯移位的數值。
移位位元數
　　指定要邏輯右移的位元數值(0~31的整數值)。

傳回值
用於傳回使指定數值資料邏輯右移後的值。

說明
RShift用於依照指定位元數使指定數值資料向右(低階方向)移位。已移位的高階位元始終被設為0。

最簡單的說明則是，Rshift用於傳回將數值資料除以2的移位位元乘方得到的數值。

注意

- 數值資料型態

  數值資料可以是任何有效的數值資料型態。

  RShift支援以下資料型態。
  Byte型、Double型、Int32型、Integer型、Long型、Real型、Short型、UByte型、UInt32型、UShort型

參照
And、LShift、LShift64、Not、Or、RShift64、Xor

RShift範例
以下是針對從「0」開始的Integer型數值資料且表示所有Rshift值的程式範例。

```
Function rshiftst
  Integer num, snum, i
  num = 32767
  For i = 1 to 16
     Print "i =", i
     snum = RShift(num, i)
     Print "RShift(32767, ", i, ") = ", snum
  Next i
Fend
```

以下是在命令視窗中操作Rshift命令的範例。

```
> Print RShift(10,1)
5
> Print RShift(8,3)
1
> Print RShift(16,2)
4
```

3.21.37 RShift64函數

用於數值資料的邏輯右移

格式

RShift64 (數值資料, 移位位元數)

參數

數值資料

以運算式或數值指定要進行邏輯移位的數值。

移位位元數

指定要邏輯右移的位元數值(0~63的整數值)。

傳回值

用於傳回使指定數值資料邏輯右移後的值。

說明

RShift64用於依照指定位元數使將指定數值資料向右(低階方向)移位。已移位的高階位元始終被設為0。

最簡單的說明則是，Rshift64用於傳回將數值資料除以2的移位位元乘方得到的數值。

注意

- 數值資料型態

  數值型有多個種類。RShift64可用於Int64型、UInt64型數值。

參照

And、LShift、LShift64、Not、Or、RShift、Xor

RShift 64範例

以下是針對從「0」開始的UInt64型數值資料且表示所有Rshift64值的程式範例。

```
Function rshif64tst
  UInt64 num, snum, i
  num = 18446744073709551615
  For i = 1 to 63
    Print "i =", i
    snum = RShift64(num, i)
    Print "RShift64(18446744073709551615, ", i, ") = ", snum
  Next i
Fend
```

以下是在命令視窗中操作Rshift64命令的範例。

```
> Print RShift64(10,1)
5
> Print RShift64(8,3)
1
> Print RShift64(16,2)
4
```

3.21.38 RTrim$函數

用於傳回刪除右側半型空格後的字串。

格式

RTrim$ (字串)

參數

字串

以字串運算式或字串進行指定。

傳回值

刪除右側半型空格後的字串

參照

LTrim$、Trim$

RTrim$函數範例

```
str$ = "  data  "
str$ = RTrim$(str$)  ' str$ = "  data"
```

3.21.39 RunDialog

用於以SPEL+程式啟動Epson RC+畫面。

格式

(1) RunDialog 對話方塊ID

(2) RunDialog DLG_ROBOTMNG [, 機器人選擇位元]

參數

對話方塊ID

指定包含有效對話方塊ID的整數值。事先用以下常數定義這些值。

- DLG_ROBOTMNG: 100: 啟動機器人管理器對話方塊
- DLG_IOMON: 102: 啟動I/O監視器
- DLG_VGUIDE: 110: 啟動Vision Guide對話方塊

機器人選擇位元

唯有將DLG_ROBOTMNG指定為對話方塊ID時方為有效。以位元值指定可透過Robot Manager選擇的機器人。

設定範例	設定值	bit15	bit14	...	bit2	bit1	bit0
機器人1	&H0001	Off	Off		Off	Off	On
機器人2	&H0002	Off	Off		Off	On	Off
機器人1和2	&H0003	Off	Off		Off	On	On
:							
機器人16	&H1000	On	Off		Off	Off	Off

說明

在SPEL+的工作上啟動/顯示Epson RC+畫面時，使用RunDialog。在關閉畫面之前，工作處於暫停狀態。

若要在Epson RC+畫面上執行機器人命令，請確認在顯示該畫面時並無控制其它機器人的工作。若有啟動控制機器人的其它工作，則會發生錯誤。

參照

InputBox、MsgBox

RunDialog範例

```
If Motor = Off Then
    RunDialog DLG_ROBOTMNG
    If Motor = Off Then
        Print "Motors are off, aborting program"
        Quit All
    EndIf
EndIf
```

3.22 S

3.22.1 SafetyOn函數

用於傳回安全門的狀態。

格式
SafetyOn

傳回值
處於開啟安全門狀態時，傳回「True」；除此之外，傳回「False」。

說明
本函數僅用於NoPause工作、NoEmgAbort工作(在Xqt時，指定NoPause或NoEmgAbort開始的特殊工作)和背景工作。

參照
ErrorOn、EstopOn、PauseOn、Wait、Xqt

SafetyOn函數範例
以下範例是由控制器監視安全門開啟，若有安全門開啟，則啟用/停用I/O的程式。

注意

- Forced旗標

  在本程式範例中，在ON/OFF命令中指定Forced旗標。

  在發生錯誤時、緊急停止時或打開安全門時，I/O輸出會發生變化，因此在系統設計上需加以注意。

```
Function main

  Xqt SafetyOnOffMonitor, NoPause
  :
  :
Fend

Function SafetyOnOffMonitor
  Do
    Wait SafetyOn = True
    Print "Saftey Open"
    Off 10, Forced
    On 12, Forced

    Wait SafetyOn = False
    Print "Saftey Close"
    On 10, Forced
    Off 12, Forced
  Loop
Fend
```

3.22.2 SavePoints

用於將主記憶體中的點資料儲存到目前機器人的磁碟檔案中。

格式

SavePoints 檔名

參數

檔案名稱

以字串運算式指定點數據儲存目的地的檔案名稱。副檔名固定為「.pts」。不可指定路徑。此外，不受ChDisk等的影響。詳細內容請參閱ChDisk。

說明

SavePoints用於將點儲存於指定檔案中。副檔名固定為「.pts」。若省略副檔名，則新增「.pts」。

此外，若沒有點檔案，SavePoints命令則用於將其新增於目前機器人的專案中。

常見錯誤

- 超過儲存空間時

  若沒有剩餘的儲存空間，則會發生錯誤。

- 指定的檔案並非目前機器人的檔案時

  若在檔名上指定其它機器人的點檔案，則發生錯誤。

- 找不到指定檔案時

  檔名中含有路徑時會發生錯誤。僅可指定目前專案的檔名。

- 檔名錯誤

  若檔名含有空格或無效字元，則會發生錯誤。

參照

ImportPoints、LoadPoints

SavePoints範例

```
ClearPoints
For i = 1 To 10
   P(i) = XY( i, 100, 0, 0 )
Next i
SavePoints "TEST.PTS"
```

3.22.3 Seek

用於變更指定檔案的指標位置。

格式

Seek #檔案編號, 指標

參數

檔案編號

以30~63的整數值或運算式來指定。

指針

以整數值或運算式指定0~檔案大小的檔案指針。

參照

BOpen、Read、ROpen、UOpen、Write、WOpen

Seek範例

```
Integer fileNum
String data$

fileNumber = FreeFile
UOpen "TEST.DAT" As #fileNum
Seek #fileNum, 20
Read #fileNum, data$, 2
Close #fileNum
```

3.22.4 Select...Send

依循運算式的值，將控制移至數個陳述式當中的任一個陳述式。

格式

Select運算式 Case項目 陳述式 [Case項目 陳述式] [Default 陳述式] Send

參數

式
　　指定數值運算式或字串運算式。
項目
　　指定與運算式一致類型的數值或字串運算式。
陳述式
　　指定1個或多個有效SPEL+陳述式或多個陳述式。

說明

在Case陳述式項目之中，若有與Select陳述式的運算式結果一致的項目，則執行最初一致的Case陳述式之後的陳述式群組。執行後，程式控制則移至緊接於Send陳述式之後的陳述式。

在Case陳述式項目之中，若有與Select陳述式的運算式結果一致的項目，則執行Default陳述式，並且程式控制移至緊接於Send陳述式之後的陳述式。

在Case陳述式項目之中，若無與Select陳述式的運算式結果一致的項目，當省略Default時，則不執行任何操作，程式控制會移至Send陳述式的下一個陳述式。

在Select陳述式的運算式上，可指定常數、變數和And、Or、Xor等運算子。Case陳述式的項目也可指定常數、變數和And、Or、Xor等運算子。此時，會將Case陳述式項目的運算結果與Select陳述式的運算式進行比較。此外，請勿在Case陳述式的項目中指定變數，否則動作會變得複雜。

參照

If…Then…Else

Select...Send範例

以下表示的是簡易的Select...Send範例。

```
Function Main
  Integer I
  For i = 0 To 10
    Select I
      Case 0
           Off 1;On 2;Jump P1
      Case 3
           On 1;Off 2
           Jump P2;Move P3;On 3
      Case 7
           On 4
      Default
           On 7
    Send
  Next
Fend
```

3.22.5 SelectDB

本函數用於從開啟的資料庫內的表格中搜尋資料。

格式

SelectDB (#資料庫編號,表格名稱,Select條件,Sort方法)

參數

資料庫編號

用於指定以OpenDB指定的資料庫編號(501~508的整數值)。

表格名稱

指定要執行資料查找的表格名稱。若以檔案編號指定的資料庫類別為Excel活頁簿，則指定Excel工作表或有命名的表格。若要指定Excel工作表，請在工作表名稱末尾附加$，並用[]括起來。若要指定以Excel工作表內的名稱指定的區域時，請用[]括起名稱。

Select條件

指定查找條件。可用AND、OR指定複合條件。未指定搜尋條件時，則搜尋表格內的所有資料。

Sort方法

指定提取查找資料的順序。指定Sort鍵和Sort順序(按升序[ASC]/降序[DESC])。若省略Sort順序，則表示已指定Sort鍵的升序。若省略Sort方法，則依開啟的資料庫確定順序。

傳回值

用於傳回搜尋行的總數。

說明

依照Sort條件從開啟的資料庫之指定表格中對符合Select條件的資料進行Sort操作。

務必在透過Input#、Print#載入/寫入資料前執行。

若已開啟的資料庫為Excel活頁簿，請在以工作表及名稱定義之區域的第1行記述用於搜尋的列名。

此外，若為Excel 2007活頁簿，請指定工作表名稱。無法存取以名稱定義的區域。

注意

需連接有安裝RC+的PC。

參照

OpenDB、CloseDB、UpdateDB、DeleteDB、Input #、Print #

SelectDB函數範例

使用SQL資料庫的範例

以下是以TitleOfCourtesy按EmployeeID的降序從SQL服務器2000範例資料庫Northwind的表格Employees載入Ms.資料的簡易範例。

```
Integer count, i, eid
String Lastname$, Firstname$, Title$

OpenDB #501, SQL, "(LOCAL)", "Northwind"
count = SelectDB(#501, "Employees", "TitleOfCourtesy = 'Ms.'", "EmployeeID DESC")
For i = 0 To count - 1
  Input #501, eid, Lastname$, Firstname$, Title$
  Print eid, ",", Lastname$, ",", Firstname$, ",", Title$
Next
CloseDB #501
```

使用Access資料庫的範例

以下是按ID的升序從Microsoft Access 2007範例資料庫學生名冊的表格中載入職務為組長的資料之簡易範例。

```
Integer count, i, eid
String Lastname$, Firstname$, dummy$

OpenDB #502, Access, "c:\MyDataBase\學生名冊.accdb"
count = SelectDB(#502, "學生", "職務 = '組長'", "ID")
For i = 0 To count - 1
  Input #502, eid, dummy$, dummy$, Lastname$, dummy$, Firstname$
  Print eid, ",", Lastname$, ",", Firstname$
Next
CloseDB #502
```

使用Excel活頁簿的範例

以下所示為按ID的升序從Microsoft Excel活頁簿學生名冊的學生工作表載入Age未滿25歲的資料之簡易範例。

```
Integer count, i, eid
String Lastname$, Firstname$

OpenDB #503, Excel, "c:\MyDataBase\學生名冊.xls"
count = SelectDB(#503, "[學生$]", "Age < 25", "ID ASC")
For i = 0 To count - 1
  Input #503, eid, Lastname$, Firstname$
  Print eid, ",", Lastname$, ",", Firstname$
Next
CloseDB #503
```

3.22.6 Sense

以Jump、Jump3、Jump3CP指定Sense時，設定並顯示停在目標坐標上方的條件。

格式

Sense [條件運算式]

參數

事件條件運算式

用於指定作為觸發器的輸入狀態。

[事件] 比較運算符(=, <>, >=, >, <, <=) [整數運算式]

可將以下函數或變數用於事件。

- 函數：Sw、In、InW、Oport、Out、OutW、MemSw、MemIn、MemInW、Ctr、GetRobotInsideBox、GetRobotInsidePlane、AIO_In、AIO_InW、AIO_Out、AIO_OutW、Hand_On、Hand_Off、SF_GetStatus
- 變數：Byte、Int32、Integer、Long、Short、UByte、UInt32、UShort型的備份變數、全域變數、模組變數

此外，可用以下運算符，對複數事件條件運算式使用遮罩或進行複合組合。

- 運算符：And、Or、Xor

[例]

```
Sense Sw(5) = On
Sense Sw(5) = On And Sw(6) = Off
```

說明

在執行Jump命令時且第3關節開始下降前，Sense用於檢查輸入條件。

此外，在執行Jump3、Jump3CP命令時且開始進行接近動作前，Sense用於檢查輸入條件。

必須在Sense條件運算式中包含1個以上的上述函數。

若Sense條件運算式中含有變數，則在設定Sense條件時運算該值。可能會變成非預期條件，因此建議在條件運算式中不使用變數。也可使用多個Sense陳述式。最後執行的輸入條件在移至下一個Sense陳述式之前有效。

- Jump和Sense修飾詞

  檢查目前Sense條件是否成立。若成立，則在機器人停在目標坐標上方的狀態下完成Jump命令。總之，Sense條件為「True」時，機器人會在第3關節開始下降前停在目標坐標上方。Sense條件為「False」時，機器人在目標坐標上完成Jump命令的動作。

- Jump3、Jump3CP和Sense修飾詞

  檢查目前Sense條件是否成立。若成立，則在機器人停在接近開始位置的狀態下完成Jump3、Jump3CP命令。

  若省略參數，則顯示目前Sense設定。

注意

- 電源ON時的Sense設定

  電源ON時的Sense條件之初始設定為Sense Sw (0) = On。有設為當輸入位元編號0為ON時機器人不執行下降動作。

- 檢查Sense條件成立的JS函數和State函數

  在執行使用Sense修飾詞的動作命令之後，可用JS函數或State函數檢查是否成立Sense條件。

- 在條件運算式中使用變數時

- 可使用的變數型態為整數型(Byte, Int32, Integer, Long, Short, UByte, UInt32, UShort)。
- 不可使用陣列變數。
- 不可使用本地變數。
- 變數值未滿足條件的時間超過0.01秒時，系統可能無法檢測到變數變化。
- 系統內可使用的變數等待數量有限。1個系統內可使用的變數等待數量最多64個(也包括Wait等條件運算式所用的變數等待數量)。若超過最大數量，則在建置專案時將發生錯誤。
- 若以ByRef傳址要執行變數等待的變數，則發生錯誤。
- 若條件運算式右邊的整數運算式中含有變數，則在開始動作命令時運算該值。可能會變成非預期條件，因此建議在整數運算式中不使用變數。

參照

In、JS、Jump、Jump3、Jump3CP、MemIn、MemSw、Stat、Sw、SF_GetStatus

Sense範例

是Sense命令的簡易範例。

```
Function test
  .
  .
TrySense:
    Sense Sw (1) = Off    '設為在停用輸入位元1時停在目標坐標上方
    Jump P1 C2 Sense
    If JS = True Then
         GoSub ERRPRC    '手臂停在目標坐標上方時
         GoTo TrySense   '執行ERRPRC並移動到TrySense
    EndIf
    On 1; Wait 0.2; Off 1
  .
  .
Fend
```

[其它格式範例]

```
> Sense Sw(1)=1 And MemSw(1)=1

> Sense Sw(0) Or (Sw(1) And MemSw(1))
```

3.22.7 SetCom

設定或顯示RS-232C連接埠的參數。

格式

SetCom #通訊連接埠編號 [, 通訊速度] [, 資料位元長度] [, 停止位元長度] [, 同位] [, 傳送接收行尾] [, H/W流量控制] [, S/W流量控制] [, 超時時間]

參數

通訊連接埠編號

以整數值指定RS-232C的連接埠編號。

- SPEL+控制部分：1～4
- PC部分：1001～1008

通訊速度

指定傳輸速率。有效值如下所示。可省略。(預設值：9600)

- 110
- 300
- 600
- 1200
- 2400
- 4800
- 9600
- 14400
- 19200
- 38400
- 57600
- 115200

使用PC部分的連接埠時，若通訊速度為19200以上，則有可能遇到丟失資料的情況。

資料位元長度

以7或8的數值指定每1個字元的資料位元長度。可省略。

停止位元長度

以1或2的數值指定每1個字元的停止位元長度。可省略。

同位

指定同位。奇數時指定O，偶數時指定E，沒有時指定N。可省略。

傳送接收行尾

指定CR、LF、CRLF中任一傳送接收行尾。可省略。

H/W流量控制

啟用硬體控制時指定RTS；停用時指定NONE。可省略。

S/W流量控制

啟用軟體控制時指定XON；停用時指定NONE。可省略。

超時時間

以運算式或數值指定超時時間(正實數值，單位：秒)。若指定0，則無限超時。可省略。

說明

若省略所有參數，則顯示通訊連接埠的設定。

若將多個連接埠的通訊速度設為19200以上並執行通訊，則有可能發生錯誤2929或2922。此時，請選擇較慢的傳送速度或不同時進行通訊。

使用PC部分的連接埠時，若通訊速度為19200以上，則有可能遇到丟失資料的情況。

資料丟失時，請選擇更慢的傳送速度，或使用SPEL+控制部分的連接埠。

參照

OpenCom、CloseCom、SetNet

SetCom範例

```
SetCom #1, 9600, 8, 1, N, CRLF, NONE, NONE, 0

SetCom #2, 4800
```

3.22.8 SetLatch

設定R-I/O輸入對機器人位置進行閂鎖的功能。

格式

SetLatch { 連接埠編號, 輸入邏輯 , 連續閂鎖次數}

參數

連接埠編號

指定用於連接觸發器輸入信號的R-I/O輸入連接埠編號。

可指定的連接埠編號如下所示。指定對象機器人所連接裝置的連接埠編號。

		點	連接埠編號
控制裝置	輸入	4點	24,25, 26, 27
	輸出	-	-

將以下常數定義為連接埠編號。

常數	連接埠編號
SETLATCH_PORT_CU_0	24
SETLATCH_PORT_CU_1	25
SETLATCH_PORT_CU_2	26
SETLATCH_PORT_CU_3	27

輸入邏輯

指定R-I/O連接之觸發器輸入信號的邏輯。可用以下常數指定邏輯。

常數	值	意義
SETLATCH_TRIGGERMODE_TRAILINGEDGE	0	負邏輯
SETLATCH_TRIGGERMODE_LEADINGEDGE	1	正邏輯

負邏輯時，將機器人位置閂鎖在輸入信號從High切換到Low的邊緣。正邏輯時，將機器人位置閂鎖在輸入信號從Low切換為High的邊緣。

連續閂鎖次數

指定基於R-I/O 輸入信號的機器人位置閂鎖的連續次數。1，2，3，4 是可用的。從LatchEnable On 開始，可對指定的連續閂鎖次數的點數據進行閂鎖。最多可鎖定四次。參數是可省略的。如果省略，則連續閂鎖次數為1次。

說明

設定以R-I/O輸入信號對機器人位置進行閂鎖的條件。1台機器人不可同時等待多個連接埠的觸發信號。執行SetLatch大約需花40 msec的處理時間。

注意

若指定與選擇的機器人無關的其它裝置之連接埠編號，則發生超出參數範圍外的錯誤。

參照

LatchEnable、LatchState函數、LatchPos函數

SetLatch範例

```
Function main
  SetLatch 24, SETLATCH_TRIGGERMODE_LEADINGEDGE, 4
```

```
'正邏輯　設定連續閂鎖4次
  LatchEnable On    '啟用閂鎖功能
  Go P1
  Wait LatchState = True     '等待觸發
  Print LatchPos(WithoutToolArm, 1)     '顯示閂鎖位置1
  Print LatchPos(WithoutToolArm, 2)     '顯示閂鎖位置2
  Print LatchPos(WithoutToolArm, 3)     '顯示閂鎖位置3
  Print LatchPos(WithoutToolArm, 4)     '顯示閂鎖位置4
  LatchEnable Off     '停用閂鎖功能
Fend
```

省略參數時的範例 :

```
Function main
  SetLatch 24, SETLATCH_TRIGGERMODE_LEADINGEDGE    '正邏輯
  LatchEnable On    '啟用閂鎖功能
  Go P1
  Wait LatchState = True     '等待觸發
  Print LatchPos     '顯示閂鎖位置
  LatchEnable Off     '停用閂鎖功能
Fend
```

3.22.9 SetIn

設定虛擬I/O的輸入連接埠(8位元)。

格式

SetIn 連接埠編號, 設定值

參數

連接埠編號
　　指定I/O輸入位元組。
設定值
　　以範圍為0~255的整數指定連接埠編號。

說明

若啟用虛擬I/O，則同時設定8個輸入位元。

若停用虛擬I/O，本命令則會發生錯誤。

參照

SetInReal, SetSW, SetInW

SetIn範例

```
> setin 0, 1  '啟用連接埠0的最初位元
```

3.22.10 SetInReal

設定虛擬I/O的輸入連接埠(32位元)。

格式

SetInReal 連接埠編號, 設定值

參數

連接埠編號
　　指定I/O輸入字組。
設定值
　　指定實體型的值。

說明

若啟用虛擬I/O，則同時設定32個輸入位元。
若停用虛擬I/O，本命令則會發生錯誤。

參照

SetSw, SetIn, S0etInW

SetIn範例

```
> setinReal 32, 2.543  '設定為輸入連接埠32
```

3.22.11 SetInW

設定虛擬I/O的輸入連接埠(16位元)。

格式

SetInW 連接埠編號, 設定值

參數

連接埠編號
　　指定I/O輸入字組。
設定值
　　以範圍為0~65535的整數指定字組。

注意

- 包括即時I/O輸入位元在內的字組連接埠相關規則

  無法反映即時I/O的輸入位元。

  包含即時I/O輸入位元的字組連接埠為1時，請以0~255的整數指定設定值範圍。若數值大於255，則發生錯誤。

說明

若啟用虛擬I/O，則同時設定16個輸入位元。

若停用虛擬I/O，本命令則會發生錯誤。

參照

SetInReal, SetSw, SetIn

SetInW範例

```
> setinw 0, 1  '啟用字組0的最初位元
```

3.22.12 SetNet

用於設定TCP/IP連接埠的參數。

格式

(1) SetNet #通訊連接埠編號, 主機位址 [, TCP/IP連接埠編號 [, 結束字元 [, 流量控制 [, 超時時間 [, 通訊協定 [, CloseNet超時時間]]]]]]

(2) SetNet

參數

通訊連接埠編號
　　指定要設定參數的TCP/IP連接埠編號。範圍：201~216。
主機地址
　　指定主機的IP地址。
TCP/IP連接埠編號
　　指定TCP/IP連接埠編號。
結束字元
　　指定CR、LF、CRLF中任一行尾字元。
流量控制
　　軟體控制流量。用於指定NONE。
超時時間
　　以秒為單位指定傳送/接收的最長時間。若為0，則無限超時。
通訊協定
　　指定通訊用的通訊協定。
- TCP：TCP通訊
- UDP：UDP通訊
- UDP_SEND：UDP傳送
- UDP_RECV：UDP接收

CloseNet超時時間
　　以秒為單位指定CloseNet關閉套接字所需的時間。(整數：0 - 5)
　　如果指定0，則關閉套接字，而無需等待對關閉請求的回應。

說明

與其他IP位址或埠號重複時，會發生錯誤。

參照

OpenNet, CloseNet, SetCom

SetNet範例

```
SetNet #201, "192.168.0.1", 2001, CRLF, NONE, 0
```

3.22.13 SetSw

設定虛擬I/O的輸入。

格式

SetSw 位元編號, 設定值

參數

位元編號
　　指定I/O輸入位元。
設定值
　　指定0(停用)或1(啟用)的整數。

說明

若啟用虛擬I/O，則設定輸入位元。

若停用虛擬I/O，本命令則會發生錯誤。

參照

SetInReal, SetIn, SetInW

SetSw範例

```
> setsw 2, on  '啟用第2個輸入位元
```

3.22.14 SF_GetParam函數

回傳安全功能參數資訊。

格式

SF_GetParam (指數)

參數

指數

以整數值或常數指定查找資訊的指數。當指數尾端指定「_EN」的索引時，啟用時會送回1，停用時會送回0。

傳回值

回傳所指定之資訊的整數值。

說明

回傳所指定之安全功能的參數值。

指數	常數	說明
1	DRYRUNOFF	試運轉停用狀態
2	SLS_1_HAND_EN	SLS_1的末端夾具速度監視狀態
3	SLS_1_SPEED	SLS_1的監視速度設定值
4	SLS_1_ELBOW_EN	SLS_1的臂肘 (水平多關節型：J2, 垂直6軸型： J3) 速度監視狀態 *1
5	SLS_1_JOINT_EN	SLS_1的關節速度監視狀態
6	SLS_1_JOINTSPEED	SLS_1的監視關節速度設定值
7	SLS_1_WRIST_EN	SLS_1的腕部(垂直六軸：J5)速度監視狀態 *1
8	SLS_1_SHOULDER_EN	SLS_1的肩部(垂直六軸: J2)速度監視狀態 *1
9	SLS_2_HAND_EN	SLS_2的末端夾具速度監視狀態
10	SLS_2_SPEED	SLS_2的監視速度設定值
11	SLS_2_ELBOW_EN	SLS_2的臂肘 (水平多關節型：J2, 垂直6軸型：J3) 速度監視狀態 *1
12	SLS_2_JOINT_EN	SLS_2的關節速度監視狀態
13	SLS_2_JOINTSPEED	SLS_2的監視關節速度設定值
14	SLS_2_WRIST_EN	SLS_2的腕部(垂直六軸：J5)速度監視狀態 *1
15	SLS_2_SHOULDER_EN	SLS_2的肩部(垂直六軸: J2)速度監視狀態 *1
16	SLS_3_HAND_EN	SLS_3的末端夾具速度監視狀態
17	SLS_3_SPEED	SLS_3的監視速度設定值
18	SLS_3_ELBOW_EN	SLS_3的臂肘 (水平多關節型：J2, 垂直6軸型：J3) 速度監視狀態 *1
19	SLS_3_JOINT_EN	SLS_3的關節速度監視狀態

指數	常數	說明
20	SLS_3_JOINTSPEED	SLS_3的監視關節速度設定值
21	SLS_3_WRIST_EN	SLS_3的腕部(垂直六軸：J5)速度監視狀態 *1
22	SLS_3_SHOULDER_EN	SLS_3的肩部(垂直六軸: J2)速度監視狀態 *1
23	SLS_T2_HAND_EN	SLS_T2的末端夾具速度監視狀態
24	SLS_T2_SPEED	SLS_T2的監視速度設定值
25	SLS_T2_ELBOW_EN	SLS_T2的臂肘 (水平多關節型：J2, 垂直6軸型：J3) 速度監視狀態 *1
26	SLS_T2_JOINT_EN	SLS_T2的關節速度監視狀態
27	SLS_T2_JOINTSPEED	SLS_T2的監視關節速度設定值
28	SLS_T2_WRIST_EN	SLS_T2的腕部(垂直六軸：J5)速度監視狀態 *1
29	SLS_T2_SHOULDER_EN	SLS_T2的肩部(垂直六軸：J2)速度監視狀態 *1
30	SLS_T_SPEED	SLS_T的監視速度設定值
31	SLS_T_JOINT_EN	SLS_T的關節速度監視狀態
32	SLS_T_JOINTSPEED	SLS_T的監視關節速度設定值
33	SLS_HAND_OFS_X	SLS的X軸方向 TCP偏移位置
34	SLS_HAND_OFS_Y	SLS的Y軸方向 TCP偏移位置
35	SLS_HAND_OFS_Z	SLS的Z軸方向 TCP偏移位置
36	SLS_1_DELAY	SLS_1的延遲時間設定值
37	SLS_2_DELAY	SLS_2的延遲時間設定值
38	SLS_3_DELAY	SLS_3的延遲時間設定值
39	SLS_JOINT_POS_EN	關節角度極限的監視狀態
40	SLS_JOINT_POS_ANGLE	監視關節角度設定值
41	SLP_A_XU_EN	SLP_A的XU (牆壁：X2，限制區域：X1) 位置監視狀態 *2
42	SLP_A_XU_POS	SLP_A的XU (牆壁：X2，限制區域：X1) 監視位置設定值 *2
43	SLP_A_XL_EN	SLP_A的XL (牆壁：X1，限制區域：X2) 位置監視狀態 *2
44	SLP_A_XL_POS	SLP_A的XL (牆壁：X1，限制區域：X2) 監視位置設定值 *2
45	SLP_A_YU_EN	SLP_A的YU (牆壁：Y2，限制區域：Y1) 位置監視狀態 *2
46	SLP_A_YU_POS	SLP_A的YU (牆壁：Y2，限制區域：Y1) 監視位置設定值 *2
47	SLP_A_YL_EN	SLP_A的YL (牆壁：Y1，限制區域：Y2) 位置監視狀態 *2
48	SLP_A_YL_POS	SLP_A的YL (牆壁：Y1，限制區域：Y2) 監視位置設定值 *2
49	SLP_A_ZU_EN	SLP_A的Z2 位置監視狀態 *2
50	SLP_A_ZU_POS	SLP_A的Z2 監視位置設定值 *2

指數	常數	說明
51	SLP_A_ZL_EN	SLP_A的Z1 位置監視狀態 *2
52	SLP_A_ZL_POS	SLP_A的Z1 監視位置設定值 *2
53	SLP_B_XU_EN	SLP_B的XU (牆壁：X2，限制區域：X1) 位置監視狀態 *2
54	SLP_B_XU_POS	SLP_B的XU (牆壁：X2，限制區域：X1) 監視位置設定值 *2
55	SLP_B_XL_EN	SLP_B的XL (牆壁：X1，限制區域：X2) 位置監視狀態 *2
56	SLP_B_XL_POS	SLP_B的XL (牆壁：X1，限制區域：X2) 監視位置設定值 *2
57	SLP_B_YU_EN	SLP_B的YU (牆壁：Y2，限制區域：Y1) 位置監視狀態 *2
58	SLP_B_YU_POS	SLP_B的YU (牆壁：Y2，限制區域：Y1) 監視位置設定值 *2
59	SLP_B_YL_EN	SLP_B的YL (牆壁：Y1，限制區域：Y2) 位置監視狀態 *2
60	SLP_B_YL_POS	SLP_B的YL (牆壁：Y1，限制區域：Y2) 監視位置設定值 *2
61	SLP_B_ZU_EN	SLP_B的Z2 位置監視狀態 *2
62	SLP_B_ZU_POS	SLP_B的Z2 監視位置設定值 *2
63	SLP_B_ZL_EN	SLP_B的Z1 位置監視狀態 *2
64	SLP_B_ZL_POS	SLP_B的Z1 監視位置設定值 *2
65	SLP_C_XU_EN	SLP_C的XU (牆壁：X2，限制區域：X1) 位置監視狀態 *2
66	SLP_C_XU_POS	SLP_C的XU (牆壁：X2，限制區域：X1) 監視位置設定值 *2
67	SLP_C_XL_EN	SLP_C的XL (牆壁：X1，限制區域：X2) 位置監視狀態 *2
68	SLP_C_XL_POS	SLP_C的XL (牆壁：X1，限制區域：X2) 監視位置設定值 *2
69	SLP_C_YU_EN	SLP_C的YU (牆壁：Y2，限制區域：Y1) 位置監視狀態 *2
70	SLP_C_YU_POS	SLP_C的YU (牆壁：Y2，限制區域：Y1) 監視位置設定值 *2
71	SLP_C_YL_EN	SLP_C的YL (牆壁：Y1，限制區域：Y2) 位置監視狀態 *2
72	SLP_C_YL_POS	SLP_C的YL (牆壁：Y1，限制區域：Y2) 監視位置設定值 *2
73	SLP_C_ZU_EN	SLP_C的Z2 位置監視狀態 *2
74	SLP_C_ZU_POS	SLP_C的Z2 監視位置設定值 *2
75	SLP_C_ZL_EN	SLP_C的Z1 位置監視狀態 *2
76	SLP_C_ZL_POS	SLP_C的Z1 監視位置設定值 *2
77	SLP_J2_MON_RAD	SLP的J2軸監視範圍半徑設定值
78	SLP_J3_MON_RAD	SLP的J3軸監視範圍半徑設定值
79	SLP_J5_MON_RAD	SLP的J5軸監視範圍半徑設定值
80	SLP_J6_MON_RAD	SLP的J6軸監視範圍半徑設定值
81	SLP_J1_RANGE_MAX	軟軸極限的J1軸極限範圍最大設定值

指數	常數	說明
82	SLP_J1_RANGE_MIN	軟軸極限的J1軸極限範圍最小設定值
83	SLP_J2_RANGE_MAX	軟軸極限的J2軸極限範圍最大設定值
84	SLP_J2_RANGE_MIN	軟軸極限的J2軸極限範圍最小設定值
85	SLP_J3_RANGE_MAX	軟軸極限的J3軸極限範圍最大設定值
86	SLP_J3_RANGE_MIN	軟軸極限的J3軸極限範圍最小設定值
87	SLP_J4_RANGE_MAX	軟軸極限的J4軸極限範圍最大設定值
88	SLP_J4_RANGE_MIN	軟軸極限的J4軸極限範圍最小設定值
89	SLP_J5_RANGE_MAX	軟軸極限的J5軸極限範圍最大設定值
90	SLP_J5_RANGE_MIN	軟軸極限的J5軸極限範圍最小設定值
91	SLP_J6_RANGE_MAX	軟軸極限的J6軸極限範圍最大設定值
92	SLP_J6_RANGE_MIN	軟軸極限的J6軸極限範圍最小設定值
93	SIN_1_SLS_1_EN	對SAFETY_IN1的SLS_1功能分配狀態
94	SIN_1_SLS_2_EN	對SAFETY_IN1的SLS_2功能分配狀態
95	SIN_1_SLS_3_EN	對SAFETY_IN1的SLS_3功能分配狀態
96	SIN_1_SLP_A_EN	對SAFETY_IN1的SLP_A功能分配狀態
97	SIN_1_SLP_B_EN	對SAFETY_IN1的SLP_B功能分配狀態
98	SIN_1_SLP_C_EN	對SAFETY_IN1的SLP_C功能分配狀態
99	SIN_1_SG_EN	對SAFETY_IN1的保護停止功能分配狀態
100	SIN_1_ESTOP_EN	對SAFETY_IN1的緊急停止功能分配狀態
101	SIN_2_SLS_1_EN	對SAFETY_IN2的SLS_1功能分配狀態
102	SIN_2_SLS_2_EN	對SAFETY_IN2的SLS_2功能分配狀態
103	SIN_2_SLS_3_EN	對SAFETY_IN2的SLS_3功能分配狀態
104	SIN_2_SLP_A_EN	對SAFETY_IN2的SLP_A功能分配狀態
105	SIN_2_SLP_B_EN	對SAFETY_IN2的SLP_B功能分配狀態
106	SIN_2_SLP_C_EN	對SAFETY_IN2的SLP_C功能分配狀態
107	SIN_2_SG_EN	對SAFETY_IN2的保護停止功能分配狀態
108	SIN_2_ESTOP_EN	對SAFETY_IN2的緊急停止功能分配狀態
109	SIN_3_SLS_1_EN	對SAFETY_IN3的SLS_1功能分配狀態
110	SIN_3_SLS_2_EN	對SAFETY_IN3的SLS_2功能分配狀態
111	SIN_3_SLS_3_EN	對SAFETY_IN3的SLS_3功能分配狀態
112	SIN_3_SLP_A_EN	對SAFETY_IN3的SLP_A功能分配狀態

指數	常數	說明
113	SIN_3_SLP_B_EN	對SAFETY_IN3的SLP_B功能分配狀態
114	SIN_3_SLP_C_EN	對SAFETY_IN3的SLP_C功能分配狀態
115	SIN_3_SG_EN	對SAFETY_IN3的保護停止功能分配狀態
116	SIN_3_ESTOP_EN	對SAFETY_IN3的緊急停止功能分配狀態
117	SIN_4_SLS_1_EN	對SAFETY_IN4的SLS_1功能分配狀態
118	SIN_4_SLS_2_EN	對SAFETY_IN4的SLS_2功能分配狀態
119	SIN_4_SLS_3_EN	對SAFETY_IN4的SLS_3功能分配狀態
120	SIN_4_SLP_A_EN	對SAFETY_IN4的SLP_A功能分配狀態
121	SIN_4_SLP_B_EN	對SAFETY_IN4的SLP_B功能分配狀態
122	SIN_4_SLP_C_EN	對SAFETY_IN4的SLP_C功能分配狀態
123	SIN_4_SG_EN	對SAFETY_IN4的保護停止功能分配狀態
124	SIN_4_ESTOP_EN	對SAFETY_IN4的緊急停止功能分配狀態
125	SIN_5_SLS_1_EN	對SAFETY_IN5的SLS_1功能分配狀態
126	SIN_5_SLS_2_EN	對SAFETY_IN5的SLS_2功能分配狀態
127	SIN_5_SLS_3_EN	對SAFETY_IN5的SLS_3功能分配狀態
128	SIN_5_SLP_A_EN	對SAFETY_IN5的SLP_A功能分配狀態
129	SIN_5_SLP_B_EN	對SAFETY_IN5的SLP_B功能分配狀態
130	SIN_5_SLP_C_EN	對SAFETY_IN5的SLP_C功能分配狀態
131	SIN_5_SG_EN	對SAFETY_IN5的保護停止功能分配狀態
132	SIN_5_ESTOP_EN	對SAFETY_IN5的緊急停止功能分配狀態
133	SOUT_1_STO	對SAFETY_OUT1的STO功能分配狀態
134	SOUT_1_SLS_1	對SAFETY_OUT1的SLS_1功能分配狀態
135	SOUT_1_SLS_2	對SAFETY_OUT1的SLS_2功能分配狀態
136	SOUT_1_SLS_3	對SAFETY_OUT1的SLS_3功能分配狀態
137	SOUT_1_SLS_T2	對SAFETY_OUT1的SLS_T2功能分配狀態
138	SOUT_1_SLS_T	對SAFETY_OUT1的SLS_T功能分配狀態
139	SOUT_1_SLP_A	對SAFETY_OUT1的SLP_A功能分配狀態
140	SOUT_1_SLP_B	對SAFETY_OUT1的SLP_B功能分配狀態
141	SOUT_1_SLP_C	對SAFETY_OUT1的SLP_C功能分配狀態
142	SOUT_1_EP_RC	對SAFETY_OUT1的緊急停止(控制器)功能分配狀態
143	SOUT_1_EP_TP	對SAFETY_OUT1的緊急停止(示教墜飾)功能分配狀態

指數	常數	說明
144	SOUT_1_EN_SW	對SAFETY_OUT1的啟用開關功能分配狀態
145	SOUT_2_STO	對SAFETY_OUT2的STO功能分配狀態
146	SOUT_2_SLS_1	對SAFETY_OUT2的SLS_1功能分配狀態
147	SOUT_2_SLS_2	對SAFETY_OUT2的SLS_2功能分配狀態
148	SOUT_2_SLS_3	對SAFETY_OUT2的SLS_3功能分配狀態
149	SOUT_2_SLS_T2	對SAFETY_OUT2的SLS_T2功能分配狀態
150	SOUT_2_SLS_T	對SAFETY_OUT2的SLS_T功能分配狀態
151	SOUT_2_SLP_A	對SAFETY_OUT2的SLP_A功能分配狀態
152	SOUT_2_SLP_B	對SAFETY_OUT2的SLP_B功能分配狀態
153	SOUT_2_SLP_C	對SAFETY_OUT2的SLP_C功能分配狀態
154	SOUT_2_EP_RC	對SAFETY_OUT2的緊急停止(控制器)功能分配狀態
155	SOUT_2_EP_TP	對SAFETY_OUT2的緊急停止(示教墜飾)功能分配狀態
156	SOUT_2_EN_SW	對SAFETY_OUT2的啟用開關功能分配狀態
157	SOUT_3_STO	對SAFETY_OUT3的STO功能分配狀態
158	SOUT_3_SLS_1	對SAFETY_OUT3的SLS_1功能分配狀態
159	SOUT_3_SLS_2	對SAFETY_OUT3的SLS_2功能分配狀態
160	SOUT_3_SLS_3	對SAFETY_OUT3的SLS_3功能分配狀態
161	SOUT_3_SLS_T2	對SAFETY_OUT3的SLS_T2功能分配狀態
162	SOUT_3_SLS_T	對SAFETY_OUT3的SLS_T功能分配狀態
163	SOUT_3_SLP_A	對SAFETY_OUT3的SLP_A功能分配狀態
164	SOUT_3_SLP_B	對SAFETY_OUT3的SLP_B功能分配狀態
165	SOUT_3_SLP_C	對SAFETY_OUT3的SLP_C功能分配狀態
166	SOUT_3_EP_RC	對SAFETY_OUT3的緊急停止(控制器)功能分配狀態
167	SOUT_3_EP_TP	對SAFETY_OUT3的緊急停止(示教墜飾)功能分配狀態
168	SOUT_3_EN_SW	對SAFETY_OUT3的啟用開關功能分配狀態
169	POS_ROT_U	設定平面旋轉 U_ROT 設定值
170	POS_ROT_V	設定平面旋轉 V_ROT 設定值
171	POS_ROT_W	設定平面旋轉 W_ROT 設定值
172	POS_OFS_X	安裝位置偏移 X_OFS 設定值
173	POS_OFS_Y	安裝位置偏移 Y_OFS 設定值
174	POS_OFS_Z	安裝位置偏移 Z_OFS 設定值

*1 安全功能管理員中監視安全速度的監視部位J2、J3、J5與本手冊所記載之超速部位末端夾具、腕部、臂肘、肩部的對應關係如下列所示。

- 水平多關節型
  - J2：臂肘
  - J3：無對應部位
- 垂直6軸型
  - J2：肩部
  - J3：臂肘
  - J5：腕部
  - Hand：夾具

*2 安全功能管理員中監視安全位置的監視位置X1、X2、Y1、Y2、Z1、Z2與本手冊所記載之監視位置XL、XU、YL、YU、ZL、ZU的對應關係如下列所示。

- 監視位置選擇「牆壁」時
  - X1 = XL, X2 = XU
  - Y1 = YL, Y2 = YU
  - Z1 = ZL, Z2 = ZU (僅限垂直6軸型)
- 監視位置選擇「限制區域」時
  - X1 = XU, X2 = XL
  - Y1 = YU, Y2 = YL
  - Z1 = ZL, Z2 = ZU (僅限垂直6軸型)

本命令可使用於搭載有Safety板的控制器。

SF_GetParam函數範例

```
If SF_GetParam (SLS_1_HAND_EN) = 1 Then
Print "SLS_1 hand speed monitoring is enabled."
EndIf
```

3.22.15 SF_GetParam$函數

回傳安全功能參數的文字資訊。

格式

SF_GetParam$ (指數)

參數

指數

以整數值或常數指定查找資訊的指數。

傳回值

回傳所指定之資訊的字串值。

說明

回傳所指定之安全功能的參數值。

指數	常數	說明
1	SF_TOOLVERSION	設定工具版本
2	SF_CHECKSUM	安全功能參數檢查碼
3	SF_LAST_MODIFIED	安全功能參數更新日期
4	SF_ROBOT_MODEL_NAME	機器人型號名稱
5	SF_ROBOT_CHECKSUM	機器人參數檢查碼
6	SF_HOFS	對機器人之物理性原點的編碼器偏移值(Hofs)
7	SF_HOFS_LAST_MODIFIED	Hofs更新日期

本命令可使用於搭載有Safety板的控制器。

SF_GetParam$函數範例

```
String Checksum$
Checksum$ = SF_GetParam$(SF_CHECKSUM)
Print "Safety function parameter Checksum is ", Checksum$

> Print SF_GetParam$(SF_LAST_MODIFIED)
2022/01/01 00:00:00
```

3.22.16 SF_GetStatus函數

回傳安全功能的狀態位元。

格式

SF_GetStatus (指數)

參數

指數

以整數值指定所要查找的資訊指數。

傳回值

以整數值回傳所指定的指數資訊。

說明

傳回值的位元資訊如下表所示。

指數	位元	值	說明
0	0-6	-	已預約
	7	&H80	Safety板的故障偵測
1	0	&H1	啟用SLS_1功能
	1	&H2	啟用SLS_2功能
	2	&H4	啟用SLS_3功能
	3-7	-	已預約
2	0	&H1	啟用SLP_A功能
	1	&H2	啟用SLP_B功能
	2	&H4	啟用SLP_C功能
	3-6	-	已預約
	7	&H80	啟用軟軸極限功能(始終啟用)
3	0	&H1	SAFETY_IN1信號High (關閉功能) *1
	1	&H2	SAFETY_IN2信號High (關閉功能) *1
	2	&H4	SAFETY_IN3信號High (關閉功能) *1
	3	&H8	SAFETY_IN4信號High (關閉功能) *1
	4	&H10	SAFETY_IN5信號High (關閉功能) *1
	5-7	-	已預約
4	0	&H1	SAFETY_OUT1信號High (關閉功能) *2
	1	&H2	SAFETY_OUT2信號High (關閉功能) *2
	2	&H4	SAFETY_OUT3信號High (關閉功能) *2
	3-7	-	已預約

指數	位元	值	說明
5	0	&H1	SLP_A *3
	1	&H2	SLP_B *3
	2	&H4	SLP_C *3
	3-6	-	已預約
	7	&H80	SLP_J *3
6～11	0-7	-	已預約
12	0-7	-	STF_ID
13	0-7	-	STF_DET_L
14	0-7	-	STF_DET_U
15	0-7	-	已預約

*1. 安全輸入信號為負邏輯 (Active Low)。
安全輸入中若輸入為信號等級High，本函數將回傳1，且分配予安全輸入的功能將不會動作。
安全輸入中若輸入為信號等級Low，本函數將回傳0，且分配予安全輸入的功能將會動作。未將機器連接於安全輸入時，將會呈現此狀態。
*2 安全輸出信號為負邏輯 (Active Low)。
分配予安全輸出的任一功能處於動作的狀態時，將啟用安全輸出，安全輸出的信號等級將變為Low，且本函數將回傳0。
分配予安全輸出的任何功能均不處於動作的狀態時，將停用安全輸出，安全輸出的信號等級將變為High，且本函數將回傳1。
任何功能均未分配予安全輸出時，安全輸出的信號等級將變為Low，且本函數將回傳0。
*3 無論安全輸入有無分配到SLP功能，也無論輸入信號有效或無效，進入監視位置時為1。

使用STF_ID以及STF_DET_L、STF_DET_H，可確認錯誤的發生原因。使用SPEL+的確認方法詳見後述。STF_ID以及STF_DET_L、STF_DET_H的資訊如下列所示。

STF_ID	錯誤名稱	STF_DET_L	STF_DET_U
100	停止通知 安全輸入	安全輸入埠 SAFETY_IN1 0x01 SAFETY_IN2 0x02 SAFETY_IN3 0x04 SAFETY_IN4 0x08 SAFETY_IN5 0x10	不使用
101	停止通知 SLS_1 超速 關節	關節編號 J1 0x01 / J2 0x02 J3 0x04 / J4 0x08 J5 0x10 / J6 0x20	不使用
102	停止通知 SLS_1 超速 部位	部位 末端夾具 0x01 / 臂肘 0x02 *1 臂肘 0x04 / 肩部 0x08 *1	不使用
103	停止通知 SLS_2 超速 關節	關節編號 J1 0x01 / J2 0x02 J3 0x04 / J4 0x08 J5 0x10 / J6 0x20	不使用

STF_ID	錯誤名稱	STF_DET_L	STF_DET_U
104	停止通知 SLS_2 超速 部位	部位 末端夾具 0x01 / 臂肘 0x02 *1 臂肘 0x04 / 肩部 0x08 *1	不使用
105	停止通知 SLS_3 超速 關節	關節編號 J1 0x01 / J2 0x02 J3 0x04 / J4 0x08 J5 0x10 / J6 0x20	不使用
106	停止通知 SLS_3 超速 部位	部位 末端夾具 0x01 / 臂肘 0x02 *1 臂肘 0x04 / 肩部 0x08 *1	不使用
107	停止通知 SLS_T 超速 關節	關節編號 J1 0x01 / J2 0x02 J3 0x04 / J4 0x08 J5 0x10 / J6 0x20	不使用
108	停止通知 SLS_T 超速 部位	部位 末端夾具 0x01 / 臂肘 0x02 *1 臂肘 0x04 / 肩部 0x08 *1	不使用
109	停止通知 SLS_T2 超速 關節	關節編號 J1 0x01 / J2 0x02 J3 0x04 / J4 0x08 J5 0x10 / J6 0x20	不使用
110	停止通知 SLS_T2 超速 部位	部位 末端夾具 0x01 / 臂肘 0x02 *1 臂肘 0x04 / 肩部 0x08 *1	不使用
115	停止通知 SLP_A違反位置 監視位置	監視位置 YL 0x01 / YU 0x02 XL 0x04 / XU 0x08 ZL 0x10 / ZU 0x20 *2	關節編號 J6 0x08 J5 0x04 J3 0x02 J2 0x01
116	停止通知 SLP_B違反位置 監視位置	監視位置 YL 0x01 / YU 0x02 XL 0x04 / XU 0x08 ZL 0x10 / ZU 0x20 *2	關節編號 J6 0x08 J5 0x04 J3 0x02 J2 0x01
117	停止通知 SLP_C違反位置 監視位置	監視位置 YL 0x01 / YU 0x02 XL 0x04 / XU 0x08 ZL 0x10 / ZU 0x20 *2	關節編號 J6 0x08 J5 0x04 J3 0x02 J2 0x01
118	停止通知 軟軸極限	關節編號 J1 0x01 / J2 0x02 J3 0x04 / J4 0x08 J5 0x10 / J6 0x20	不使用

STF_ID	錯誤名稱	STF_DET_L	STF_DET_U
121	停止通知 輸入開關	開關編號 啟用開關 0x01 緊急停止開關(控制器連接) 0x02 緊急停止開關(示教墜飾) 0x04	不使用
122	停止通知 模式控制	模式 允許參數通訊 0x08 停用安全功能 (Safety板) 0x04 切換操作模式 0x02 設定參數 已認證 0x01	不使用
123	停止通知 監視減速	偵測異常 減速異常0x08, 0x04 減速完成0x02 監控時間經過 0x01	不使用
124	停止通知 關節角度極限	關節編號 J1 0x01 / J2 0x02 J3 0x04 / J4 0x08 J5 0x10 / J6 0x20	不使用
131	故障停止通知 編碼器通訊異常	關節編號 J1 0x01 / J2 0x02 J3 0x04 / J4 0x08 J5 0x10 / J6 0x20	不使用
132	故障停止通知 位置異常	關節編號 J1 0x01 / J2 0x02 J3 0x04 / J4 0x08 J5 0x10 / J6 0x20	不使用
133	故障停止通知 輸入重複異常	偵測異常部位 安全輸入埠 SAFETY_IN 1 0x01 SAFETY_IN 2 0x02 SAFETY_IN 3 0x04 SAFETY_IN 4 0x08 SAFETY_IN 5 0x10 啟用開關 0x20 緊急停止開關(控制器連接) 0x40 緊急停止開關(示教墜飾) 0x80	不使用
134	故障停止通知 輸出重複異常	偵測異常部位 安全輸出埠 SAFETY_OUT 1 0x01 SAFETY_OUT 2 0x02 SAFETY_OUT 3 0x04 STO 0x80	不使用

STF_ID	錯誤名稱	STF_DET_L	STF_DET_U
135	故障停止通知 Safety板異常	偵測異常部位 通訊匯流排 0x20 電源(3.3V) 0x08 電源(5V) 0x04 看門狗計時器檢測到異常 0x02 繼電器熔接 0x01	不使用
136	故障停止通知 Safety板 MCU異常	偵測異常部位 序列監視器 0x10 CPU 0x08 RAM 0x04 程式ROM 0x02 資料ROM 0x01	DET_L = 0x01時 (資料ROM) 0x00 - 0xFE 資料故障處 0xFF 參數故障
137	故障停止通知 Safety板 重複內部異常	偵測異常部位 TCP位置不一致 0x02 狀態不一致 0x01	不使用
138	故障停止通知 編碼器 內部異常	關節編號 J1 0x01 / J2 0x02 J3 0x04 / J4 0x08 J5 0x10 / J6 0x20	不使用
139	故障停止通知 控制器 內部異常	偵測異常部位 操作模式收訊錯誤 0x01	不使用

*1 安全功能管理員中監視安全速度的監視部位J2、J3、J5與本手冊所記載之超速部位末端夾具、腕部、臂肘、肩部的對應關係如下列所示。

- 水平多關節型
  - J2：臂肘
  - J3：無對應部位
  - J5：無對應部位
  - Hand：夾具
- 垂直6軸型
  - J2：肩部
  - J3：臂肘
  - J5：腕部
  - Hand：夾具

*2 安全功能管理員中監視安全位置的監視位置X1、X2、Y1、Y2、Z1、Z2與本手冊所記載之監視位置XL、XU、YL、YU、ZL、ZU的對應關係如下列所示。

- 監視位置選擇「牆壁」時
  - X1 = XL, X2 = XU
  - Y1 = YL, Y2 = YU
  - Z1 = ZL, Z2 = ZU (僅限垂直6軸型)
- 監視位置選擇「限制區域」時
  - X1 = XU, X2 = XL
  - Y1 = YU, Y2 = YL

- Z1 = ZL, Z2 = ZU (僅限垂直6軸型)

如何使用STF_ID以及STF_DET_L、STF_DET_H確認錯誤發生的原因

在命令視窗等依照下列順序輸入命令，並加以確認。

```
> SF_GetStatus (12)
115     '表示「停止通知 SLP_A 違反位置」造成的錯誤。
> SF_GetStatus (13)
1     '表示超過了「YL」方向的監視位置。(確認STF_ID：115的內容。)
> SF_GetStatus (14)
1     '表示「J2軸」已超過。(確認STF_ID：115的內容。)
```

整合以上資訊後，可了解到錯誤發生的原因為「J2軸超過了SLP_A的YL監視位置」。

錯誤資訊亦會記錄於Epson RC+的系統歷史記錄中。錯誤發生原因會記錄於「新增資訊」。請參閱狀態碼與錯誤碼的錯誤訊息。

本命令可使用於搭載有Safety板的控制器。

SF_GetStatus函數範例

```
If (SF_GetStatus(3) And &H1) = &H1 Then
Print "SAFETY_IN1 is High"
Else
Print "SAFETY_IN1 is Low"
EndIf
```

3.22.17 SF_LimitSpeedS

啟用SLS時，針對調整透過Tool指令所設定位置的速度的功能，設定、顯示速度調整值。

格式

SF_LimitSpeedS [SLS編號 [, 速度調整值]]

參數

SLS編號

以整數值或下列所示的常數指定SLS編號。可省略。

常數	值	內容
SLS_1	1	SLS_1啟用時的速度調整功能
SLS_2	2	SLS_2啟用時的速度調整功能
SLS_3	3	SLS_3啟用時的速度調整功能
SLS_T	9	SLS_T啟用時的速度調整功能 *1
SLS_T2	10	SLS_T2啟用時的速度調整功能 *1

*1 SLS_T與SLS_T2無法透過這個命令設定速度調整值。而是以安全功能管理器中設定的監視速度進行速度調整。有關詳細資訊，請參閱以下手冊。

「機器人控制器 安全功能手冊」

速度調整值

以整數值(0～10000，單位：mm/sec)指定速度。可省略。指定為0時，會自動設定速度調整值。初始值為0。

說明

啟用SLS時，設定、顯示針對調整透過Tool指令所設定位置的速度的功能。以本功能進行速度調整的部位，是現在透過Tool指令選擇的工具位置。請注意並非為透過安全功能參數設定TCP位置。

設定SLS啟用時速度調整功能所指定之SLS編號的速度調整值[mm/sec]。省略第二參數時，將顯示所指定之SLS編號的速度調整值。

省略所有參數時，將顯示所有SLS編號的速度調整值。

本命令可使用於搭載有Safety板的控制器。

僅限安全功能選項啟用時方可使用。

注意

- 使用SF_LimitSpeedS進行速度調整的位置，是透過Tool命令選擇的工具位置

  使用SF_LimitSpeedS設定的速度調整值，適用於透過Tool命令設定的工具位置的速度。安全功能管理器的TCP位置並非由Tool指令自動設定。請利用Tool指令設定適當的TCP位置。

  此外，當速度自動調整功能無法正確運作時，請使用SF_PeakSpeedS、SF_RealSpeedS測量機器人手臂的動作速度，並使用Speed, SpeedFactor、SpeedS等控制機器人手臂的動作速度，使其不會超過SLS監視速度(安全功能參數)。

SF_LimitSpeedS範例

如何將SLS_1的速度調整值設定為1500mm/sec

```
SF_LimitSpeedS SLS_1, 1500
```

如何顯示SLS_2的速度調整值(使用命令視窗)

```
> SF_LimitSpeedS SLS_2
SLS_2: 400
```

如何顯示所有SLS編號的速度調整值(使用命令視窗)

```
> SF_LimitSpeedS
SLS_1: 1500
SLS_2: 400
SLS_3: 750
SLS_T: 250
SLS_T2: 3000
```

3.22.18 SF_LimitSpeedS函數

啟用SLS時，會送回對透過Tool指令設定位置的速度進行調整的功能的速度調整值。

格式

SF_LimitSpeedS (SLS編號)

參數

SLS編號

以整數值或下列所示的常數指定SLS編號。

常數	值	內容
SLS_1	1	SLS_1啟用時的速度調整功能
SLS_2	2	SLS_2啟用時的速度調整功能
SLS_3	3	SLS_3啟用時的速度調整功能
SLS_T	9	SLS_T啟用時的速度調整功能
SLS_T2	10	SLS_T2啟用時的速度調整功能

傳回值

回傳所指定之SLS編號的速度調整值[mm/sec]。

說明

回傳SLS啟用時速度調整功能所指定之SLS編號的速度調整值[mm/sec]。

本命令可使用於搭載有Safety板的控制器。

SF_LimitSpeedS函數範例

```
Integer i
i = SF_LimitSpeedS(SLS_1)
Print "SLS_1 limit speed is ", i
```

3.22.19 SF_LimitSpeedSEnable

啟用SLS時，針對調整透過Tool指令設定位置的速度的功能，設定On/Off，並顯示設定狀態。

格式

SF_LimitSpeedSEnable [SLS編號 [, {On | Off}]]

參數

SLS編號

以整數值或下列所示的常數指定SLS編號。可省略。

常數	值	內容
SLS_1	1	SLS_1啟用時的速度調整功能
SLS_2	2	SLS_2啟用時的速度調整功能
SLS_3	3	SLS_3啟用時的速度調整功能
SLS_T	9	SLS_T啟用時的速度調整功能 *1
SLS_T2	10	SLS_T2啟用時的速度調整功能 *1

*1 SLS_T與SLS_T2無法透過這個命令設定速度調整功能。SLS_T在操作模式為TEACH和TEST T1時，速度調整功能為On。SLS_T2在操作模式為TEST T2時，速度調整功能為On。有關詳細資訊，請參閱以下手冊。
「機器人控制器 安全功能手冊」

On | Off

指定速度調整功能的On/Off。可省略。預設之設定為On。

傳回值

無

說明

當指定的SLS編號處於監視狀態時，針對調整透過Tool指令設定位置的速度的功能，設定、顯示On/Off。當設為On時，於啟用SLS時調整工具設定位置的速度。當設為Off時，即使啟用SLS也不會調整速度。不論設為On/Off，於停用SLS時，不會調整工具位置的速度。

省略第二參數時，將顯示所指定之SLS編號的速度調整狀態(On/Off)。

省略所有參數時，將顯示所有SLS編號的速度調整狀態(On/Off)。

本命令可使用於搭載有Safety板的控制器。

僅限安全功能選項啟用時方可使用。

SF_LimitSpeedSEnable範例

如何將SLS_1的速度調整功能設定為啟用

```
SF_LimitSpeedSEnable SLS_1, On
```

如何顯示SLS_2的速度調整功能的狀態(使用命令視窗)

```
> SF_LimitSpeedSEnable SLS_2
SLS_2: Off
```

如何顯示所有SLS編號的速度調整功能的狀態(使用命令視窗)

```
> SF_LimitSpeedSEnable
SLS_1: On
```

```
SLS_2: Off
SLS_3: Off
SLS_T: On
SLS_T2: On
```

注意

- SF_LimitSpeedSEnable無法與奇點迴避、輸送帶追蹤、力控制同時使用

  SF_LimitSpeedSEnable的預設設定為On。力控制、輸送帶追蹤、奇點迴避於動作中時，以本功能進行速度調整會發生動作錯誤(分別為4093、4403、5830)。此外，SF_LimitSpeedSEnable設為On的情形下，輸送帶動作開始時會發生動作錯誤 (即4500)。

  若要在這些動作使用SLS，請在動作開始前將SF_LimitSpeedSEnable設為Off。

  此外，請使用SF_PeakSpeedS、SF_RealSpeedS測量機器人手臂的動作速度，並使用Speed、SpeedFactor、SpeedS等控制機器人手臂的動作速度，使其不超過SLS監視速度(安全功能參數)。

- 使用SF_LimitSpeedS進行速度調整的位置，是透過Tool指令設定的工具位置

  於SF_LimitSpeedSEnable設為On時適用的速度調整值(使用SF_LimitSpeedS設定)，適用於透過Tool指令設定工具位置的速度。安全功能管理器的TCP位置並非由Tool指令自動設定。請利用Tool指令設定適當的TCP位置。

  此外，當速度自動調整功能無法正確運作時，請使用SF_PeakSpeedS、SF_RealSpeedS測量機器人手臂的動作速度，並使用Speed, SpeedFactor、SpeedS等控制機器人手臂的動作速度，使其不會超過SLS監視速度(安全功能參數)。

3.22.20 SF_LimitSpeedSEnable函數

啟用SLS時，會送回調整透過Tool指令設定位置的速度的功能狀態。

格式

SF_LimitSpeedSEnable (SLS編號)

參數

SLS編號

以整數值或下列所示的常數指定SLS編號。

常數	值	內容
SLS_1	1	SLS_1啟用時的速度調整功能
SLS_2	2	SLS_2啟用時的速度調整功能
SLS_3	3	SLS_3啟用時的速度調整功能
SLS_T	9	SLS_T啟用時的速度調整功能
SLS_T2	10	SLS_T2啟用時的速度調整功能

傳回值

- 所指定之SLS編號的速度調整為On時，回傳1。
- 所指定之SLS編號的速度調整為Off時，回傳0。

說明

回傳所指定之SLS編號處於監視狀態時TCP動作速度調整功能的狀態。

本命令可使用於搭載有Safety板的控制器。

SF_LimitSpeedSEnable函數範例

```
If SF_LimitSpeedSEnable(SLS_1) = 0 Then
Print "SLS_1 linked speed adjustment function is disabled."
Endif
```

3.22.21 SF_PeakSpeedS

顯示速度監視點的峰值速度。

格式
SF_PeakSpeedS [速度監視點編號]

參數

速度監視點編號
以整數值(1~4)指定速度監視點編號。可省略。
- 1：夾具
- 2：臂肘
- 3：腕部
- 4：肩部

傳回值
無

說明
顯示所指定之速度監視點的峰值速度。

省略參數時，將顯示所有速度監視點的峰值速度。

末端夾具的速度是透過安全功能管理員設定在TCP偏移位置的速度。

本命令可使用於搭載有Safety板的控制器。

SF_PeakSpeedS範例
如何顯示末端夾具的峰值速度(使用命令視窗)

```
> SF_PeakSpeedS 1
250
```

如何顯示所有速度監視點的峰值速度(使用命令視窗)

```
> SF_PeakSpeedS
250      150
240      100
```

顯示順序如下列所示。
夾具臂肘
腕部 肩部

3.22.22 SF_PeakSpeedS函數

回傳速度監視點的峰值速度。

格式

SF_PeakSpeedS (速度監視點編號)

參數

速度監視點編號

以整數值(1~4)指定速度監視點編號。可省略。

- 1：夾具
- 2：臂肘
- 3：腕部
- 4：肩部

傳回值

回傳峰值速度[mm/sec]。

說明

回傳所指定之速度監視點的峰值速度。

末端夾具的速度是透過安全功能管理員設定在TCP偏移位置的速度。

本命令可使用於搭載有Safety板的控制器。

SF_PeakSpeedS函數範例

```
Print "Hand peak speed is ", SF_PeakSpeedS (1)
```

3.22.23 SF_PeakSpeedSClear

清除速度監視點的峰值速度，並進行初始化。

格式

SF_PeakSpeedSClear [速度監視點編號1 [, 速度監視點編號2]]

參數

速度監視點編號1

以整數值(1~4)指定第1個速度監視點編號。可省略。

速度監視點編號2

以整數值(1~4)指定第2個速度監視點編號。可省略。

- 1：夾具
- 2：臂肘
- 3：腕部
- 4：肩部

傳回值

無

說明

清除(初始化)所指定之速度監視點的峰值速度。

未指定參數時，會清除所有速度監視點的峰值速度。

本命令可使用於搭載有Safety板的控制器。

SF_PeakSpeedSClear範例

如何清除末端夾具的峰值速度

```
SF_PeakSpeedSClear 1
```

如何清除所有速度監視點的峰值速度

```
SF_PeakSpeedSClear
```

3.22.24 SF_RealSpeedS

顯示速度監視點的目前速度。

格式

SF_RealSpeedS [速度監視點編號]

參數

速度監視點編號

以整數值(1~4)指定速度監視點編號。可省略。

- 1：夾具
- 2：臂肘
- 3：腕部
- 4：肩部

傳回值

無

說明

顯示所指定之速度監視點的目前速度[mm/sec]。

末端夾具的速度是透過安全功能管理員設定在TCP偏移位置的速度。

省略參數時，將顯示所有速度監視點的目前速度。

本命令可使用於搭載有Safety板的控制器。

SF_RealSpeedS範例

如何顯示末端夾具的目前速度(使用命令視窗)

```
> SF_RealSpeedS 1
250
```

如何顯示所有速度監視點的目前速度(使用命令視窗)

```
> SF_RealSpeedS
250      150
240      100
```

顯示順序如下列所示。

夾具臂肘

腕部 肩部

3.22.25 SF_RealSpeedS函數

回傳速度監視點的目前速度。

格式

SF_RealSpeedS (速度監視點編號)

參數

速度監視點編號

以整數值(1~4)指定速度監視點編號。可省略。

- 1：夾具
- 2：臂肘
- 3：腕部
- 4：肩部

傳回值

回傳目前速度[mm/sec]。

說明

回傳所指定之速度監視點的目前速度[mm/sec]。

末端夾具的速度是透過安全功能管理員設定在TCP偏移位置的速度。

本命令可使用於搭載有Safety板的控制器。

SF_RealSpeedS函數範例

```
Print "Hand real speed is ", SF_RealSpeedS (1)
```

3.22.26 SFree

用於設定指定關節的SFree狀態。

格式

SFree 關節編號[, 關節編號,...]

參數

關節編號

以運算式或數值指定關節編號(1~9的整數)。附加軸的S軸為8，T軸為9。

說明

SFree用於切斷指定關節馬達的電源。此時的狀態稱為SFree狀態。此命令用於直接教導或關閉特定關節的SFree狀態並進行組合。若要解除該關節SFree狀態，則執行SLock命令、Motor On或Motor Off。

執行SFree命令，會對機器人控制參數進行初始化。

詳細內容請參閱Motor On。

注意

- 部分系統設定透過SFree被初始化

  SFree用於對機器人的動作速度或加減速度相關參數(Speed, SpeedS, Accel, AccelS等)以及LimZ參數進行初始化，以確保安全。詳細內容請參閱Motor On。

  韌體版本在7.5.1.0之前，僅Motor ON狀態下可以SFree。

  SFree的功能因韌體版本而異。

韌體	可否SFree
7.5.1.0以前	僅啟動馬達時
7.5.1.0或以後	馬達啟動和關閉時

重要事項

- 將SFree用於水平多關節型機器人(包含RS系列)的第3關節以及第4關節時

  在水平多關節型機器人(包含RS系列)的第3關節上會啟動電磁制動器，因此即便有設定SFree，第3關節也不會立即動作。若要手動運作第3關節，需持續按下機器人手臂上方的制動器解除開關。

  此外，也有第4關節設有制動器的機種。若為在第4關節設有制動器的機器人，則在第4關節上會啟動電磁制動器，因此即便有設定SFree，第4關節也不會立即動作。若要手動運作第4關節，需持續按下機器人手臂上方制動器解除開關。

- 不可將SFree用於垂直6軸型機器人(包含N系列)

  若在垂直6軸型機器人(包含N系列)中使用SFree，則會發生錯誤。

  如果要手動移動手臂，請在Motor Off后在Brake Off中釋放電子制動器。

- 在關節處於SFree狀態時執行動作命令

  若在關節處於SFree狀態時執行動作命令，控制器則會在預設狀態下發生錯誤。即便有1個關節處於SFree狀態也要執行動作命令時，則勾選[設定]選單[控制器] - [環境設定]面板的[允許關節在非勵磁狀態的移動命令]核取方塊。

- 請勿在輸送帶追蹤中使用SFree

  若在輸送帶追蹤中使用SFree，則發生錯誤5057、5058等。請以Cnv_AbortTrack命令等完成輸送帶追蹤之後使用SFree。

參照

Brake、LimZ、Motor、SFree函數、SLock

SFree範例

是SFree命令的簡易範例。若要在本範例中動作，必須勾選[設定]選單的[控制器] - [環境設定]面板的[允許關節在非勵磁狀態的移動命令]核取方塊。

```
Function GoPick
   Speed pickSpeed
   SFree 1, 2    '將J1和J2設為SFree狀態
移動Z和U關節以安裝料件
   Go pick
   SLock 1, 2    '對J1和J2解除SFree狀態
Fend
```

3.22.27 SFree函數

用於傳回指定關節的SFree狀態。

格式

SFree (關節編號)

參數

關節編號

以數值或運算式指定用於檢查SFree狀態的關節編號(整數)。附加軸的S軸為8，T軸為9。

傳回值

- 1：SFree狀態
- 0：非SFree狀態

參照

SFree

SetFree範例

```
If SFree(1) Then
    Print "Joint 1 is free"
EndIf
```

3.22.28 Sgn函數

用於傳回數值的符號。

格式

Sgn (運算數)

參數

運算元

指定數值。

傳回值

- 1：運算數為正值
- 0：運算數為0
- -1：運算數為負值

說明

Sgn函數用於傳回數值的符號。

參照

Abs、And、Atan、Atan2、Cos、Int、Mod、Or、Not、Sin、Sqr、Str$、Tan、Val、Xor

Sgn函數範例

以下範例是在命令視窗中使用Sgn的範例。

```
>print sgn(123)
 1
>print sgn(-123)
 -1
>
```

3.22.29 Short

宣告Short型變數。(2位元組整數型變數)

格式

Short 變數名稱 [(陣列變數的最大元素編號)] [, 變數名稱 [(陣列變數的最大元素編號)]...]

參數

變數名稱

指定要進行變數宣告的變數名稱。

陣列變數的最大元素編號

是陣列變數的最大元素編號，最大可宣告三維。使用如下格式。可省略。

(最大元素編號1, [最大元素編號2], [最大元素編號3])

元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。在所有元素數不超過以下最大值的範圍內，指定各最大元素編號。

- 本地變數：2,000
- 備份變數(Global Preserve)：4,000
- 全域變數和模組變數：100,000

說明

Short用於宣告整數型變數。整數型變數範圍：-32768~32767。在Function的開頭宣告本地變數。在Function之外宣告全域變數和模組變數。

參照

Boolean、Byte、Double、Global、Int32、Int64、Integer、Long、Real、String、UByte、UInt32、UInt64、UShort

Short範例

以下為使用Short宣告整數型變數的程式範例。

```
Function shorttest
  Short A(10)          'Short型的一維陣列
  Short B(10, 10)      'Short型的二維陣列
  Short C(5, 5, 5)     'Short型的三維陣列
  Short var1, arrayvar(10)
  Integer i
  Print "Please enter an Integer Number"
  Input var1
  Print "The Integer variable var1 = ", var1
  For i = 1 To 5
    Print "Please enter an Integer Number"
    Input arrayvar(i)
    Print "Value Entered was ", arrayvar(i)
  Next i
Fend
```

3.22.30 Signal

用於向正在執行WaitSig命令的工作傳送信號。

格式

Signal 信號編號

參數

信號編號

以0~63指定要傳送的信號編號。

說明

Signal用於對多工進行同步。忽略執行WaitSig前的Signal。

參照

WaitSig

Signal範例

```
Function Main
   Xqt 2, SubTask
   Call InitSys
   Signal 1

Fend

Function SubTask
   WaitSig 1

Fend
```

3.22.31 SimGet

用於取得模擬器各種物件的屬性設定值。

格式

SimGet Object.Property, Var

SimGet Robot.Hand.Propoerty, Var

參數

Object
　表示用於讀取屬性值的物件名稱之字串變數
Robot
　表示有安裝夾具(以「Hand」指定)的機器人名稱之字串變數
Hand
　表示用於讀取屬性值的夾具名稱之字串變數
Property
　用於讀取數值的屬性名稱。屬性詳如後述。
Var
　表示傳回值的變數

說明

此命令用於取得模擬器各種物件的屬性設定值。

可透過指定如下所示的屬性，取得物件的設定值。

屬性	說明	單位	資料類型	傳回值
PositionX	取得X坐標位置	公釐(mm)	Double	
PositionY	取得Y坐標位置	公釐(mm)	Double	
PositionZ	取得Z坐標位置	公釐(mm)	Double	
RotationX	取得X軸旋轉角度	度(degree)	Double	
RotationY	取得Y軸旋轉角度	度(degree)	Double	
RotationZ	取得Z軸旋轉角度	度(degree)	Double	
CollisionCheck	取得碰撞檢測的啟用/停用	-	Boolean	True或False
CollisionCheckSelf	取得機器人自我碰撞檢測的啟用/停用	-	Boolean	True或False
Visible	取得顯示/隱藏的狀態	-	Boolean	True或False
Type	用於取得物件類型	-	Integer	Layout: 0 Part: 1 Mounted Device: 3
HalfSizeX	取得BOX物件的X方向的長度	公釐(mm)	Double	

屬性	說明	單位	資料類型	傳回值
HalfSizeY	取得BOX物件的Y方向的長度	公釐 (mm)	Double	
HalfSizeZ	取得BOX物件的Z方向的長度	公釐 (mm)	Double	
HalfSizeHeight	取得Plane物件的長度	公釐 (mm)	Double	
HalfSizeWidth	取得Plane物件的寬度	公釐 (mm)	Double	
PlaneType	取得Plane物件的類型	-	Integer	Horizontal: 0 Vertical: 1
Radius	取得Sphere物件或Cylinder物件的半徑	公釐 (mm)	Double	
Height	取得Cylinder物件的高度	公釐 (mm)	Double	
Name	取得物件名稱		String	
Color	取得物件展現的顏色		String	顏色名稱或者16進位顏色代碼 (ARGB)

可透過下表所示的組合取得屬性。

屬性	物件							
	Robot	Hand	Box	Sphere	Cylinder	Plane	CAD	Camera
PositionX	○	○	○	○	○	○	○	○
PositionY	○	○	○	○	○	○	○	○
PositionZ	○	○	○	○	○	○	○	○
RotationX	○	○	○	○	○	○	○	○
RotationY	○	○	○	○	○	○	○	○
RotationZ	○	○	○	○	○	○	○	○
CollisionCheck	○	○	○	○	○	○	○	○
CollisionCheckSelf	○	-	-	-	-	-	-	-
Visible	-	○	○	○	○	○	○	○
Type	-	-	○	○	○	○	○	-
HalfSizeX	-	-	○	-	-	-	-	-
HalfSizeY	-	-	○	-	-	-	-	-
HalfSizeZ	-	-	○	-	-	-	-	-
HalfSizeHeight	-	-	-	-	-	○	-	-

屬性	物件							
	Robot	Hand	Box	Sphere	Cylinder	Plane	CAD	Camera
HalfSizeWidth	-	-	-	-	-	○	-	-
PlaneType	-	-	-	-	-	○	-	-
Radius	-	-	-	○	○	-	-	-
Height	-	-	-	-	○	-	-	-
Name	○	○	○	○	○	○	○	○
Color	-	-	○	○	○	○	-	-

參照

SimSet

SimGet範例

```
'取得SBox_1物件的X坐標值
Double boxPosX
SimGet SBox_1.PositionX, boxPosX

'取得SBox_1物件的顯示/隱藏狀態
Boolean boxVisible
SimGet SBox_1.Visible, boxVisible

'取得SBox_1物件的類型
Integer boxType
SimGet SBox_1.Type, boxType
```

3.22.32 SimSet

用於設定模擬器各種物件的屬性設定。此外，用於進行機器人的動作、物件操作和設定模擬器相關的操作。

格式

(1) 物件的屬性設定

```
SimSet Object.Property, Value
SimSet Robot.Hand.Property, Value
```

(2) 機器人的動作設定(Pick及Place)

```
SimSet Robot.Pick, Object [,Tool]
SimSet Robot.Place, Object
```

(3) 物件的操作設定(指定父物件)

```
SimSet Object.SetParent [, ParentObject]
```

(4) 模擬器的設定(重設碰撞檢測)

```
SimSet ResetCollision
```

參數

(1) 物件的屬性設定

Object
　　表示用於設定屬性值的物件名稱之字串變數
Robot
　　表示有安裝夾具(以「Hand」指定)的機器人名稱之字串變數
Hand
　　表示用於設定屬性值的夾具名稱之字串變數
Property
　　用於設定新值的屬性名稱。屬性詳如後述。
Value
　　新值的運算式。資料類型因屬性而異。

(2) 機器人動作設定(Pick及Place)

Robot
　　表示用於進行Pick或Place的機器人名稱之字串變數
Object
　　表示被Pick或Place的物件名稱之字串變數
Tool
　　表示用於Pick的Tool編號之運算式

(3) 物件的操作設定(指定父物件)

Object
　　表示設定父物件的物件名稱之字串變數
ParentObject
　　表示父物件名稱的字串變數

說明

此命令用於進行模擬器各種物件的屬性設定/操作、機器人動作、模擬器設定的變更。

(1) 物件的屬性設定

可透過指定如下所示屬性，設定物件。

屬性	說明	單位	資料類型	設定值
PositionX	用於設定X坐標位置	公釐 (mm)	Double	最大值：100000 最小值：-100000
PositionY	用於設定Y坐標位置	公釐 (mm)	Double	最大值：100000 最小值：-100000
PositionZ	用於設定Z坐標位置	公釐 (mm)	Double	最大值：100000 最小值：-100000
RotationX	用於設定X軸旋轉角度	度 (degree)	Double	最大值：360 最小值：-360
RotationY	用於設定Y軸旋轉角度	度 (degree)	Double	最大值：360 最小值：-360
RotationZ	用於設定Z軸旋轉角度	度 (degree)	Double	最大值：360 最小值：-360
CollisionCheck	用於設定碰撞檢測的啟用/停用	-	Boolean	True或False
CollisionCheckSelf	用於設定機器人自我碰撞檢測的啟用/停用	-	Boolean	True或False
Visible	用於設定顯示/隱藏	-	Boolean	True或False
HalfSizeX	取得BOX物件的X方向的長度	公釐 (mm)	Double	最大值：100000 最小值：0.001
HalfSizeY	取得BOX物件的Y方向的長度	公釐 (mm)	Double	最大值：100000 最小值：0.001
HalfSizeZ	取得BOX物件的Z方向的長度	公釐 (mm)	Double	最大值：100000 最小值：0.001
HalfSizeHeight	取得Plane物件的長度	公釐 (mm)	Double	最大值：100000 最小值：0.001
HalfSizeWidth	取得Plane物件的寬度	公釐 (mm)	Double	最大值：100000 最小值：0.001
PlaneType	取得Plane物件的類型	-	Integer	Horizontal: 0 Vertical: 1
Radius	取得Sphere物件或Cylinder物件的半徑	公釐 (mm)	Double	最大值：100000 最小值：0.001
Height	取得Cylinder物件的高度	公釐 (mm)	Double	最大值：100000 最小值：0.001
Name	取得物件名稱		String	
Color	取得物件展現的顏色		String	顏色名稱或者16進位顏色代碼 (ARGB)

可透過下表所示組合設定屬性。

屬性	物件							
	Robot	Hand	Box	Sphere	Cylinder	Plane	CAD	Camera
PositionX	○	○	○	○	○	○	○	○
PositionY	○	○	○	○	○	○	○	○
PositionZ	○	○	○	○	○	○	○	○
RotationX	○	○	○	○	○	○	○	○
RotationY	○	○	○	○	○	○	○	○
RotationZ	○	○	○	○	○	○	○	○
CollisionCheck	○	○	○	○	○	○	○	○
CollisionCheckSelf	○	-	-	-	-	-	-	-
Visible	-	○	○	○	○	○	○	○
HalfSizeX	-	-	○	-	-	-	-	-
HalfSizeY	-	-	○	-	-	-	-	-
HalfSizeZ	-	-	○	-	-	-	-	-
HalfSizeHeight	-	-	-	-	-	○	-	-
HalfSizeWidth	-	-	-	-	-	○	-	-
PlaneType	-	-	-	-	-	○	-	-
Radius	-	-	-	○	○	-	-	-
Height	-	-	-	-	○	-	-	-
Name	○	○	○	○	○	○	○	○
Color	-	-	○	○	○	○	-	-

(2) 機器人的動作設定(Pick及Place)

可設定如下所示機器人的動作。

- Pick

  以「Robot」指定的機器人進行握持以「Object」指定的物件之動作。

  被握持的物件將被註冊為該機器人的零件。此外，透過在「Tool」上指定任意工具編號，利用指定工具進行握持動作。若省略指定「Tool」，則利用Tool0進行握持動作。

  無法握持已註冊為零件的物件或設為手臂安裝設備的物件。此外，也無法握持相機。

- Place

  以「Robot」指定的機器人進行配置以「Object」指定的物件之動作。配置的物件將被解除該機器人零件的註冊。

  不可配置已被解除零件註冊的物件。

可透過下表所示組合握持或配置物件。

動作	物件							
	Robot	Hand	Box	Sphere	Cylinder	Plane	CAD	Camera
Pick	-	-	○	○	○	○	○	-
Place	-	-	○	○	○	○	○	-

(3) 物件的操作設定(指定父物件)

可設定如下所示物件的操作。

- SetParent

  對於以「Object」指定的物件，將以「ParentObject」指定的物件視為父物件進行設定。可省略「ParentObject」。此時，以「Object」指定的物件即為父物件。舉例來說，若以「Object」指定的物件為某個物件的子物件，則解除子物件的設定。

  此外，若將以「Object」指定的物件設為零件或手臂安裝機器，不可指定父物件。

  下表所示為可指定SetParent的物件。在相機物件方面，唯有作為固定相機而設定的物件方可利用SetParent。

操作	物件							
	Robot	Hand	Box	Sphere	Cylinder	Plane	CAD	Camera
SetParent	-	-	○	○	○	○	○	○

  此外，SetParent可用於下表所示的搭配。

		ParentObject(父物件)							
		Robot	Hand	Box	Sphere	Cylinder	Plane	CAD	Camera
Object (子物件)	Robot	-	-	-	-	-	-	-	-
	Hand	-	-	-	-	-	-	-	-
	Box	-	-	○	○	○	○	○	○
	Sphere	-	-	○	○	○	○	○	○
	Cylinder	-	-	○	○	○	○	○	○
	Plane	-	-	○	○	○	○	○	○
	CAD	-	-	○	○	○	○	○	○
	Camera	-	-	○	○	○	○	○	○

(4)模擬器的設定(重設碰撞檢測)
可變更如下模擬器的設定。

- ResetCollision

  用於重設碰撞檢測。執行ResetCollision後，若機器人和物件之間未發生碰撞，則解除碰撞狀態，並更新模擬器的3D顯示。機器人和物件之間發生碰撞時，不解除碰撞狀態，也不更新模擬器的3D顯示。

參照

SimGet

SimSet範例

```
'將SBox_1物件的X坐標值設為100.0mm
SimSet SBox_1.PositionX, 100.0

'透過Robot1以Tool1握持SBox_1
SimSet Robot1.Pick, SBox_1, 1

'配置透過Robot1握持的SBox_1
SimSet Robot1.Place, SBox_1

'在SBox_1的父物件上設定CAD_1
SimSet SBox_1.SetParent, CAD_1

'將SBox_1作為父物件
SimSet SBox_1.SetParent

'重設碰撞檢測
SimSet ResetCollision
```

3.22.33 Sin函數

用於傳回指定數值的正弦值。

格式

Sin (角度)

參數

角度

以實數值指定角度。

傳回值

用於以數值傳回表示指定角度的正弦值。

說明

傳回指定角度(弧度)的正弦值。參數單位必須是弧度。

傳回值的範圍為-1~1。

若要將弧度轉換為角度，則使用RadToDeg函數。

參照

Abs、Atan、Atan2、Cos、Int、Mod、Not、Sgn、Sqr、Str$、Tan、Val

Sin函數範例

以下是使用Sin函數的範例。

```
Function sintest
    Real x
    Print "Please enter a value in radians:"
    Input x
    Print "Sin of ", x, " is ", Sin(x)
Fend
```

3.22.34 SingularityAngle

用於設定奇點通過功能所需的特定點附近角度。

格式

SingularityAngle {角度設定值}

參數

角度設定值

以運算式或數值指定用於判斷垂直六軸機器人(包含N系列)腕部奇點附近的第5關節角度(0.1以上的實數，單位是deg)。

結果

若省略參數，則顯示目前SingularityAngle設定值。

說明

唯有使用奇點通過功能時，此命令方為有效。

預設值為10 deg。用於調整接近奇點時進行迴避動作的開始位置。若設定小於預設值的值，則在更加接近奇點之後開始進行迴避動作。平常不需變更本參數，但若有變更，則在奇點通過時可抑制錯誤發生。

若有變更SingularityAngle設定值，則於下一次啟動控制器前即可啟用。

參照

AvoidSingularity、SingualrityAngle函數、SingularitySpeed

SingularityAngle範例

```
SingularityAngle 7.0 '將奇點附近角度設為7度
```

3.22.35 SingularityAngle函數

用於傳回SingularityAngle的設定值。

格式

SingularityAngle

傳回值

用於傳回有設定的奇點附近角度(單位：deg)。

參照

AvoidSingularity、SingularityAngle、SingularitySpeed、SingularitySpeed函數

SingularityAngle函數範例

```
Real currSingularityAngle
currSingularityAngle = SingularityAngle
```

3.22.36 SingularityDist

設定奇點通過功能所需特定點附近距離。

格式

SingularityDist {距離設定值}

參數

距離設定值

以運算式或數值指定用於判斷垂直六軸機器人(包含N系列)以及RS系列肩部奇點附近的P點和第1關節旋轉軸距離(0以上的實數，單位是mm)。

結果

若省略參數，則顯示目前SingularityDist設定值。

說明

唯有使用奇點通過功能時，此命令方為有效。

預設值為30 mm。用於調整接近肩部奇點時進行迴避動作的開始位置。若設定小於預設值的值，則在更加接近奇點之後開始進行迴避動作。平常不需變更本參數，但若有變更，則在奇點通過時可抑制錯誤發生。

若有變更SingularityDist設定值，則於下一次啟動控制器前即可啟用。

參照

AvoidSingularity、SingularityAngle、SingualrityAngle函數、SingularityDist函數、SingularitySpeed、SingularitySpeed函數

SingularityDist範例

```
SingularityDist 10.0 '將奇點附近距離設為10 mm
```

3.22.37 SingularityDist函數

用於傳回SingularityDist的設定值。

格式

SingularityDist

傳回值

用於傳回有設定的奇點附近距離(單位：mm)。

參照

SingularityDist、AvoidSingularity、SingularityAngle、SingularityAngle函數、SingularitySpeed、SingularitySpeed函數

SingularityDist函數範例

```
Real currSingularityDist
currSingularityDist = SingularityDist
```

3.22.38 SingularitySpeed

用於設定奇點通過功能所需特定點附近角速度。

格式

SingularitySpeed {角速度設定值}

參數

角速度設定值

以運算式或數值指定用於判斷垂直六軸機器人 (包含N系列) 的腕部奇點附近之第4關節角速度對於最大角速度的比例 (1以上100以下的整數，單位是%)。

結果

若省略參數，則顯示目前SingularitySpeed設定值。

說明

唯有使用奇點通過功能時，此命令方為有效。

預設值為10 %。用於調整接近奇點時進行迴避動作的開始位置。若設定大於預設值的值，則在更加接近奇點之後開始進行迴避動作。平常不需調整或變更本參數，但若有設定較小的值，則在奇點通過時可抑制錯誤發生。

若有變更SingularitySpeed設定值，則於下一次啟動控制器前即可啟用。

參照

AvoidSingularity函數、SingualrityAngle、SingularitySpeed

SingularitySpeed範例

```
SingularitySpeed 30 '將奇點附近角速度的比例設為30 %
```

3.22.39 SingularitySpeed函數

用於傳回SingularitySpeed的設定值。

格式

SingularitySpeed

傳回值

用於傳回對於設定的奇點附近角速度的最大角速度之比例(單位：%)。

參照

SingularitySpeed、SingularityAngle、AvoidSingularity

SingularitySpeed函數範例

```
Real currSingularitySpeed
currSingularitySpeed = SingularitySpeed
```

3.22.40 SLock

解除指定關節的SFree狀態。

格式

SLock 關節編號 [, 關節編號,...]

參數

關節編號

以運算式或數值指定關節編號(1~9的整數)。附加軸的S軸為8，T軸為9。

說明

SLock用於爲了直接教導或裝配工件等，而透過SFree命令設定成SFree狀態的關節，解除關節的SFree狀態。

若省略關節編號，則重新啟動所有關節馬達的SFree狀態。

若啟動第3關節的SLock，則解除電磁制動器。

執行SLock命令後，對機器人控制參數進行初始化。詳細內容請參閱Motor On。

垂直6軸型機器人(包含N系列)無法透過SFree命令進入SFree狀態。若執行SLock，則會發生錯誤。

參照

Brake、LimZ、Reset、SFree

SLock範例

以下是使用SLock命令的範例。若要在本範例中動作，必須啟用[設定]選單的[系統設定] - [一般]面板的[允許關節在非勵磁狀態的移動命令]選項按鈕。

```
Function test
   .
   .
   .
   SFree 1, 2      '將J1和J2設為SFree狀態，移動Z和U型關節，以安裝零件
   Go P1
   SLock 1, 2      '解除J1和J2的SFree狀態
   .
   .
   .
Fend
```

3.22.41 SoftCP

設定SoftCP動作模式。

格式

SoftCP { On | Off }

參數

On | Off

- On：啟用SoftCP動作模式。
- Off：解除SoftCP動作模式。

說明

SoftCP動作模式用於抑制以高加減速度執行CP動作時的振動性動作。

平常的CP動作被設為重視軌跡追蹤性能及等速動作性能。因此，以高加減速度進行運作時，有可能形成振動。以往將SpeedS或AccelS這樣的加減速度限制在較低設定值，以抑制此振動。但是，比起高軌跡追蹤性或等速性，部分應用程式更要求以高加減速度執行振動更少的CP動作。在SoftCP動作模式下，透過將CP動作時的軌跡追蹤性及等速性抑制到比平時較小的值，會更進一步減低設定高加減速度之CP動作所帶來的振動。

SoftCP動作模式對以下CP動作命令有效。
Move, BMove, TMove, Arc, Arc3, CVMove, Jump3CP

請勿在平常的CP動作並無振動問題時或在重視軌跡追蹤性或等速性的應用程式中使用SoftCP動作模式。

重要事項

- 在CP On時連接CP動作和PTP動作

  若要按如下所述連接CP動作和PTP動作，請啟用SoftCP。若未啟用，則有可能機器人因為某個動作而發出異音。連接完成後，請停用SoftCP。

  ```
  SoftCP On
  Go P1 CP
  Move P2
  SoftCP Off
  ```

參照

SoftCP函數

SoftCP範例

```
SoftCP On
Move P1
Move P2
SoftCP Off
```

3.22.42 SoftCP函數

用於傳回SoftCP動作模式的狀態。

格式
SoftCP

傳回值

- 0 = SoftCP動作模式OFF
- 1 = SoftCP動作模式ON

參照
SoftCP

SoftCP函數範例

```
If SoftCP = Off Then
    Print "SoftCP is off"
EndIf
```

3.22.43 Space$函數

用於傳回指定數量的半形空格字串。

格式

Space$ (指定數量)

參數

指定數

指定作為傳回值而傳回的空格數。

傳回值

指定數量的半形空格字串

說明

Space$用於傳回指定數量的半形空格字串。Space$用於傳回最多255個字元的半形空格(字串變數的容許範圍)。

Space$用於在其它字串之前、後和中間添加半形空格。

參照

Asc、Chr$、InStr、Left$、Len、LSet$、Mid$、Right$、RSet$、Str$、Val

Space$函數範例

```
> Print "XYZ" + Space$(1) + "ABC"
XYZ ABC

> Print Space$(3) + "ABC"
   ABC
>
```

3.22.44 Speed

以Go、Jump、Pulse命令等設定和顯示PTP動作的速度。

格式

(1) Speed 速度設定值 [, 閃避速度, 接近速度]

(2) Speed

參數

速度設定值
　　以運算式或數值指定對於最大動作速度(PTP動作)的比例(1~100的整數，單位：%)。
閃避速度
　　以運算式或數值指定Jump命令時的閃避動作速度(1~100的整數，單位：%)。可省略。唯有Jump命令時方可設定。
接近速度
　　以運算式或數值指定Jump命令時的接近動作速度(1~100的整數，單位：%)。可省略。唯有Jump命令時方可設定。

結果

若省略參數，則顯示目前Speed設定值。

說明

Speed用於指定所有PTP動作命令的速度。其中包括Go、Jump、Pulse等動作命令相關速度設定。設定速度時，以1~100的整數指定對最大速度的比例 (%)。若指定「100」，則以最大速度進行動作。

閃避速度及接近速度僅適用於Jump命令。一旦省略，速度設定值就會變成閃避速度和接近速度的設定值。

在以下任一情況下，Speed值會被初始化。

- 啟動控制器時
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

低功率模式時，Speed設定值會低於預設值。即便透過命令視窗或程式輸入更高的值，仍被設為預設值。高功率模式時，動作速度則為以Speed設定的值。

若需要更快的動作速度，則以Power High設定高功率模式，並關閉安全門。若安全門開著，Speed值則會被變更為預設值。

若在低功率模式時執行Speed，則顯示訊息。在以下範例中，機器人處於低功率模式。因此，即便Speed設定值為「80」，機器人仍以預設值「5」的速度進行運作。

```
> speed 80
> speed
Low Power Mode
    80
    80     80
>
```

參照

Accel、Go、Jump、Power、Pass、Pulse、SpeedS

Speed範例

可在命令視窗或程式中使用Speed。以下為兩種情況時的範例。

```
Function speedtst
    Integer slow, fast, i
    slow = 10
    fast = 100
    For i = 1 To 10
        Speed slow
        Go P0
        Go P1
        Speed fast
        Go P0
        Go P1
    Next i
Fend
```

以下是在命令視窗中設定Speed值的範例。

```
> Speed 100,100,50     '將Z關節的下降速度設為50%
> Speed 50
> Speed
    Low Power State: Speed is limited to 5
       50
       50         50
>
```

3.22.45 Speed函數

用於傳回目前設定的PTP動作速度。

格式

Speed [(參數編號)]

參數

參數編號

由如下所示的編號，指定1個設定值編號。省略時被視為指定1。

- 1：PTP動作速度
- 2：JUMP閃避速度
- 3：JUMP接近速度

傳回值

用於傳回1~100的整數值。

參照

Speed

Speed函數範例

```
Function SpeedEx
    Integer savSpeed

    savSpeed = Speed(1)
    Speed 50
    Go pick
    Speed savSpeed
Fend
```

3.22.46 SpeedFactor

設定機器人動作的速度係數並傳回設定值。

格式

(1) SpeedFactor 速度比率

(2) SpeedFactor

參數

速度比率

以運算式或數值指定機器人動作的速度比率(1~100的整數，單位：%)。

結果

若省略參數，則顯示目前SpeedFactor值。

說明

SpeedFactor是用於在控制器上設定的所有機器人以及所有動作之速度係數。通常，SpeedFactor被設為100%，並以Speed、SpeedS設定各機器人和各動作命令的速度。SpeedFactor有助於在欲以固定的速度比率對所有機器人的所有動作變更速度。舉例來說，若將Speed 80%的動作按速度比率50%進行設定時，動作速度則變成40%。

SpeedFactor也用於以相同比例變更加速度，以考量到機器人動作的加減速度之平衡。

SpeedFactor的設定值等於操作員視窗的速度比率設定值，並做連動變化。

啟動控制器時，SpeedFactor會被初始化為預設值100%。

參照

SpeedFactor函數

SpeedFactor範例

```
Function main
    Motor On
    Power High
    SpeedFactor 80

    Speed 100; Accel 100,100
    Go P1            '按Speed 80; Accel 80,80進行動作

    Speed 50; Accel 50,50
    Go P2            '按Speed 40; Accel 40,40進行動作
Fend
```

3.22.47 SpeedFactor函數

用於傳回SpeedFactor命令的設定值。

格式

SpeedFactor

傳回值

用於以整數值傳回SpeedFactor命令的設定值。

參照

SpeedFactor

SpeedFactor函數範例

```
Real savSpeedFactor

savSpeedFactor = SpeedFactor
SpeedFactor 80
Go P1
Go P2
SpeedFactor savSpeedFactor
```

3.22.48 SpeedR

用於設定和顯示使用ROT時的CP動作之工具姿態變化速度。

格式

(1) SpeedR 速度設定值

(2) SpeedR

參數

速度設定值

以運算式或數值指定CP動作時的工具姿態變化速度(0.1以上的實數，單位：deg/sec)。
參數的有效設定值：0.1～1000

結果

若省略參數，則顯示目前SpeedR設定值。

說明

在以Move、Arc、Arc3、BMove、TMove、Jump3CP的動作命令指定ROT修飾參數時，此命令才會啟用。此外，在SpeedRLimitation啟用時，做為工具姿態變化速度上限值的功能。

在以下任一情況下，SpeedR值會被初始化為預設值(低速)。

- 啟動控制器時
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

參照

AccelR, Arc, Arc3, BMove, Jump3CP, Power, SpeedR函數, TMove, SpeedRLimitation

SpeedR範例

```
SpeedR 200
```

3.22.49 SpeedR函數

用於傳回工具姿態變化速度。

格式

SpeedR

傳回值

以實數值(單位：deg/sec)傳回設定的速度。

參照

AccelR、SpeedR

SpeedR函數範例

```
Real currSpeedR

currSpeedR = SpeedR
```

3.22.50 SpeedRLimitation

是用於設定和顯示工具姿態變化速度的動作速度限制的函數。

格式

(1) SpeedRLimitation {On | Off}

(2) SpeedRLimitation

參數

On | Off

On：用於啟動工具姿態變化的動作速度限制。
Off：用於關閉工具姿態變化的動作速度限制。

結果

若省略參數，則顯示目前SpeedR設定值。

說明

以下列命令啟用SpeedRLimitation。
Move, TMove, BMove, Arc, Arc3, Jump3CP

SpeedRLimitation啟用時，未避免CP動作時的工具姿態變化速度超過設定的SpeedR，用來限制動作速度。工具姿態變化速度未超過SpeedR時，動作速度以SpeedS中設定的速度動作。
即使SpeedRLimitation停用，工具姿態變化速度時，自動作動限制動作速度的功能。想要調整工具姿態變化速度時，請啟用SpeedRLimitation功能，調整適當的SpeedR。

以下情況時，SpeedRLimitation處於控制器設定所指定的狀態(工廠出廠時：Off)。

- 啟動控制器時
- 執行Motor On
- 執行Reset
- 執行Reset、Reset Error
- 因停止按鈕，執行Quit All等而結束工作

注意

- SpeedRLimitation使用須知
  根據控制器的環境設定，控制器啟動時可開啟透過工具姿態變化速度的動作速度限制。
- SpeedR的設定
  SpeedR的預設值為低速設定，未適當設定SpeedR時，隨著方向變化的CP動作會跟著變慢。在SpeedRLimitation啟用時，請視SpeedS適當設定做為工具姿態變化速度上限的SpeedR。

參照

SpeedR, SpeedS, SpeedRLimitation函數, Move, TMove, BMove, Arc, Arc3, Jump3CP

SpeedRLimitation範例

```
SpeedR 50 'deg/s
SpeedRLimitation On
Move P1
```

3.22.51 SpeedRLimitation函數

用於傳回工具姿態變化的動作速度限制啟動或停用。

格式

SpeedRLimitation

傳回值

0 = 停用
1 = 啟動

SpeedRLimitation函數範例

```
If SpeedRLimitation = On Then
        Print "SpeedRLimitation is On"
EndIf
```

3.22.52 SpeedS

用於設定和顯示Move、Arc、Arc3、Jump3、Jump3CP等CP動作時的手臂速度。

格式

(1) SpeedS 速度設定值 [, 閃避速度, 接近速度]

(2) SpeedS

參數

速度設定值
　　以運算式或數值(整數，單位：mm/sec)指定速度。
閃避速度
　　指定用於表示Jump3閃避速度的實數或運算式(單位：mm/sec)。可省略。
接近速度
　　指定用於表示Jump3接近速度的實數或運算式(單位：mm/sec)。可省略。

參數的有效設定值:

- 非N系列：0.1～2000
- N系列：0.1～1120

結果

若省略參數，則顯示目前SpeedS值。

說明

SpeedS用於指定CP動作(Move以及Arc)命令時的速度。

SpeedS值為指定機器人速度的值。指定值以mm/sec為單位。預設值會因機器人的機種而異。關於SpeedS的預設值，請參閱以下手冊。
「機械手手冊 」

開啟控制器電源時，始終會自動設定該值。

在以下任一情況下，SpeedS值會被初始化。

- 啟動控制器時
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

在低功率模式下，預設值或設定值中較低的一方之SpeedS值有效。若透過命令視窗或程式指定更高的速度設定，速度則會被設為預設值。高功率模式時，動作速度則為SpeedS設定值。

若需要更快的動作速度，則以Power High設定高功率模式，並關閉安全門。若安全門開著，SpeedS值則會變更為預設值。

參照

AccelS、Arc、Jump3、Move、Speed

SpeedS範例

可在命令視窗或程式中使用SpeedS。以下為兩種情況時的範例。

```
Function speedtst
  Integer slow, fast, i
  slow = 50
```

```
  fast = 500
  For i = 1 To 10
    SpeedS slow
    Move P0
    Move P1
    SpeedS fast
    Move P0
    Move P1
  Next i
Fend
```

還可在命令視窗中設定SpeedS。

```
> speeds 1000
> speeds 500
> speed 30      '設定PTP動作速度
> go p0         '按PTP動作進行移動
> speeds 100    '以mm/sec設定直線速度
> move P1       '進行直線移動
```

3.22.53 SpeedS函數

用於傳回SpeedS設定。

格式

SpeedS [(參數編號)]

參數

參數編號

從下述速度當中，以整數或運算式指定要傳回的SpeedS值。可省略。

- 1：CP速度
- 2：Jump3閃避速度
- 3：Jump3接近速度

傳回值

用於傳回實數值。單位：mm/sec。

參照

SpeedS

SpeedS函數範例

```
Real savSpeeds

savSpeeds = SpeedS

Print "Jump3 depart speed = ", SpeedS(2)
```

3.22.54 Sqr函數

用於傳回平方根。

格式
Sqr (數值)

參數

數值
　　以實數值或運算式指定。

傳回值
用於傳回平方根。

說明
用於傳回平方根。

常見錯誤

- 負數運算子

  數值為負數時會發生錯誤。

參照
Abs、And、Atan、Atan2、Cos、Int、Mod、Not、Or、Sgn、Sin、Str$、Tan、Val、Xor

Sqr函數範例
以下是在命令視窗中使用Sqr函數的範例。

```
>print sqr(2)
 1.414214
>
```

以下是使用Sqr的簡易程式範例。

```
Function sqrtest
    Real x
    Print "Please enter a numeric value:"
    Input x
    Print "The Square Root of ", x, " is ", Sqr(x)
Fend
```

3.22.55 ST函數

用於將指定附加軸的坐標值作為點資料傳回。

格式

ST (s坐標元素, t坐標元素)

參數

s坐標元素
　　以實數值指定s軸的坐標。
t坐標元素
　　以實數值指定t軸的坐標。

傳回值

用於將指定附加軸的坐標值作為點資料傳回。

說明

唯有使用附加軸(S軸、T軸)時才使用本函數。

採用Go ST(10,20)那樣的用法時，附加軸會移至指定的坐標值位置，但機器人不會動作。若要同時運作機器人，請進行如Go XY(60,30,-50,45)：ST(10,20)這樣的指定。

有關詳細資訊，請參閱以下手冊。
「Epson RC+ 使用者指南 - 附加軸」

參照

XY函數

ST函數範例

```
P10 = ST(10, 20)
```

3.22.56 StartMain

從背景工作執行Main函式。

本命令為適合高階專業人員使用的命令。請充分理解命令內容後使用。

格式

StartMain Main函式名稱

參數

主要函數名稱

指定要執行主要函數名稱(main~main63)。

說明

若要在程式中執行本命令，需勾選Epson RC+的[設定] - [系統設定] - [控制器] - [環境設定]的[啟用進階工作控制命令]的核取方塊。

若在背景工作以Xqt命令執行工作，執行的工作也會變成背景工作。若使用本命令，則可在背景工作將Main函式作為一般工作予以執行。

若已執行Main函式或在一般工作執行本命令，則會發生錯誤。

> ⚠ 注意
>
> 若要在程式中執行StartMain命令，請理解命令內容，並確認可作為系統執行Main函式的條件皆已備齊。若透過迴圈繼續執行命令等進行錯誤的使用，則有可能導致系統安全性下降。請充分注意。

參照

Xqt

StartMain範例

```
Function bgmain

    :
  If Sw(StartMainSwitch) = On And Sw(ErrSwitch) = Off Then
    StartMain main
  EndIf
    :

Fend
```

3.22.57 Stat函數

用於傳回控制器的狀態。

格式

Stat (位址)

參數

位址

指定表示控制器狀態的地址(0~2的整數)。

傳回值

用於傳回表示控制器狀態的4位元組的值。(請參閱下表 。)

說明

Stat命令用於傳回下表所示的資料。請參閱各位元內容。

位址	Bit		位元為ON時所示的控制器狀態
0	0-15	&H1- &H8000	工作1~16為執行中(Xqt)或處於Halt狀態
	16	&H10000	工作執行中
	17	&H20000	暫停狀態
	18	&H40000	錯誤狀態
	19	&H80000	TEACH模式
	20	&H100000	緊急停止狀態
	21	&H200000	低功率模式(Power Low)
	22	&H400000	安全門輸入為「開」
	23	&H800000	Enable開關為「開」
	24	&H1000000	未定義
	25	&H2000000	未定義
	26	&H4000000	測試模式
	27	&H8000000	T2模式狀態
	28-31		未定義

位址	Bit		位元為ON時所示的控制器狀態
1	0	&H1	因Jump...Sense陳述式的條件成立而停於目標坐標上方的歷程記錄。(若執行下一個Jump陳述式，則刪除此歷程記錄。)
	1	&H2	因Go/Jump/Move...Till陳述式的條件成立而在動作途中停止的歷程記錄。(若執行下一個Go/Jump/Move...Till陳述式，則刪除該歷程記錄。)
	2	&H4	未定義
	3	&H8	因Trap陳述式的條件成立而在動作途中停止的歷程記錄
	4	&H10	Motor On狀態
	5	&H20	目前位於Home位置
	6	&H40	低功率狀態
	7	&H80	未定義
	8	&H100	第4關節勵磁中
	9	&H200	第3關節勵磁中
	10	&H400	第2關節勵磁中
	11	&H800	第1關節勵磁中
	12	&H1000	第6關節勵磁中
	13	&H2000	第5關節勵磁中
	14	&H4000	第T關節勵磁中
	15	&H8000	第S關節勵磁中
	16	&H10000	第7關節勵磁中
	17-31		未定義
2	0-15	&H1-&H8000	工作17~32為執行中(Xqt)或處於Halt狀態

參照

EStopOn函數、TillOn函數、PauseOn函數、SafetyOn函數

Stat函數範例

```
Function StatDemo

    Integer rbt1_sts
    rbt1_sts = RShift((Stat(0) And &H070000), 16)
    Select TRUE
        Case (rbt1_sts And &H01) = 1
            Print "Tasks are running"
        Case (rbt1_sts And &H02) = 2
            Print "Pause Output is ON"
        Case (rbt1_sts And &H04) = 4
            Print "Error Output is ON"
    Send
Fend
```

3.22.58 Str$函數

本函數用於將數值轉換為字串後傳回。

格式
Str$ (數值)

參數

數值
　　以運算式或數值指定要轉換為字串的數值。

傳回值
用於將數值轉換為字串後傳回。

說明
Str$用於將數值(正負皆可)轉換為字串。

參照
Abs、Asc、Chr$、InStr、Int、Left$、Len、Mid$、Mod、Right$、Sgn、Space$、Val

Str$函數範例
以下範例表示的是，將幾個不同的數值轉換為字串並顯示於螢幕的情形。

```
Function strtest
    Integer intvar
    Real realvar
    '
    intvar = -32767
    Print "intvar = ", Str$(intvar)
    '
    realvar = 567.9987
    Print "realvar = ", Str$(realvar)
    '
Fend
```

以下是在命令視窗中執行Str$命令的結果。

```
> Print Str$(99999999999999)
 1.000000E+014

> Print Str$(25.999)
 25.999
```

3.22.59 String

將變數宣告為字串變數。

格式

String 變數名稱$[(陣列的最大元素編號)] [,變數名稱$[(陣列變數的最大元素編號)]...]

參數

變數名稱$

指定宣告為字串型的變數名稱。陣列變數的最大元素編號
是陣列變數的最大元素編號，最大可宣告三維。使用如下格式。可省略。
(最大元素編號1, [最大元素編號2], [最大元素編號3])
元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。在所有元素數不超過以下最大值的範圍內，指定各最大元素編號。

- 本地變數：200
- 備份變數(Global Preserve)：400
- 全域變數和模組變數：10,000

說明

String用於宣告變數的字串型。字串型變數最大可使用255個字元。在Function的開頭宣告本地變數。在Function之外宣告全域變數和模組變數。

- String運算子

  可對字串變數使用以下運算子。

- +：合併多個字串變數。可用於對字串變數進行分配的陳述式或Print命令。

```
Example: name$ = fname$ + " " + lname$
```

- =：對多個字串變數進行比較。包含Case在內的2個字串完全一致時，傳回True。

```
Example: If temp1$ = "A" Then GoSub test
```

- <>：對多個字串變數進行比較。2個字串當中有1個以上的字元不同時，傳回True。

```
Example: If temp1$ <> "A" Then GoSub test
```

注意

- 請在變數名稱最後附加「$」

  請在String型變數名稱最後附加「$」。

參照

Boolean、Byte、Double、Global、Int32、Int64、Integer、Long、Real、Short、UByte、UInt32、UInt64、UShort

String範例

```
String password$
String A$(10)          'String型的一維陣列
String B$(10, 10)      'String型的二維陣列
String C$(5, 5, 5)     'String型的三維陣列
```

```
Print "Enter password:"
Input password$
If UCase$(password$) = "Epson" Then
    Call RunMaintenance
Else
    Print "Password invalid!"
EndIf
```

3.22.60 Sw函數

本函數用於傳回指定輸入位元狀態。

格式
Sw (位元編號)

參數

位元編號
以整數或運算式指定I/O的輸入位元。

傳回值
指定的輸入為ON時，傳回「1」; OFF時，傳回「0」。

說明
Sw用於檢查I/O輸入的狀態。Sw最常用於檢查連接到由I/O驅動的載入器、輸送帶、夾持機構等週邊裝置的感測器。以Sw命令檢查的輸入狀態包括「1」或「0」兩種。分別表示裝置為ON(1)或OFF(0)。

參照
In、InBCD、MemOn、MemOff、MemSw、Off、On OpBCD、Oport、Out、Wait

Sw函數範例
以下是檢查輸入位元5，並根據該結果對程式進行分支的範例。為了更為明確，使用「On」以替代「1」。

```
Function main
    Integer i, feed5Ready
    feed5Ready = Sw(5)
    '確認是否已準備Feeder
    If feed5Ready = On Then
        Call mkpart1
    Else
        Print "Feeder #5 is not ready. Please reset and"
        Print "then restart program"
    EndIf
Fend
```

以下是命令視窗中的簡易操作範例。

```
> print sw(5)
1
>
```

3.22.61 SyncLock

使用互斥鎖，對多項工作進行同步。

格式

SyncLock 信號編號 [, 超時]

參數

信號編號

以運算式或數值指定要接收的信號編號(0~63的整數)。

超時

以運算式或數值(實數)指定鎖定前等待的最長等待時間。可省略。

說明

SyncLock用於同步鎖定公共資源，以便1次只能用於1項工作。當某項工作用完公共資源時，則叫用SyncUnlock進行解鎖，讓其它工作得以使用該資源。

只有事先自行鎖定的syncID，方可將1個工作解鎖。

必須執行SyncUnlock才能解鎖工作。結束工作時，則解除該工作所進行的鎖定。

若連續2次對相同信號編號執行SyncLock，則會發生錯誤。

若有指定超時參數，則請使用Tw函數檢查是否成功鎖定。

注意

就Epson RC+5.0而言，即便結束工作也不自動解除鎖定，而Epson RC+ 6.0、RC+ 7.0、Epson RC+ 8.0則會自動解除。

參照

Signal、SyncLock、Tw、Wait、WaitPos

SyncLock範例

在以下範例中，使用SyncLock和SyncUnlock進行設定，以便每次只有1項工作可將訊息寫入記錄檔中。

```
Function Main

    Xqt Func1
    Xqt Func2
Fend

Function Func1
  Long count
  Do
    Wait .5
    count = count + 1
    LogMsg "Msg from Func1, " + Str$(count)
  Loop
Fend

Function Func2
  Long count
  Do
    Wait .5
    count = count + 1
    LogMsg "Msg from Func2, " + Str$(count)
  Loop
Fend
```

```
Function LogMsg(msg$ As String)
  SyncLock 1
  OpenCom #1
  Print #1, msg$
  CloseCom #1
  SyncUnlock 1
Fend
```

以下是使用超時(選項)的SyncLock範例。使用Tw檢查是否成功鎖定。在使用超時等待鎖定資源時，可執行其它程式。

```
Function MySyncLock(syncID As Integer)
  Do
    SyncLock syncID, .5
    If Tw = 0 Then
      Exit Function
    EndIf
    If Sw(1) = On Then
      Off 1
    EndIf
  Loop
Fend
```

3.22.62 SyncUnlock

用於解除事先用SyncLock鎖定的信號編號鎖定狀態。

格式

SyncUnlock 信號編號

參數

信號編號

以運算式或數值指定要接收的信號編號(1~63的整數)。

說明

SyncUnlock用於解除事先用SyncLock鎖定的信號編號鎖定狀態。

若不是事先鎖定的信號編號，則不可解鎖。

參照

Signal、SyncLock、Wait、WaitPos

SyncUnlock範例

```
Function Main

    Xqt task
    Xqt task
    Xqt task
    Xqt task
Fend

Function task
  Do
    SyncLock 1
    Print "resource 1 is locked by task", MyTask
    Wait .5
    SyncUnlock 1
  Loop
Fend
```

3.22.63 SyncRobots

用於開始動作預約中的機器人動作。

格式

SyncRobots 機器人編號 [, 機器人編號] [,...]

SyncRobots All

參數

機器人編號
　　以運算式或數值(整數)指定要開始運作的機器人編號。
All
　　指定有動作預約的所有機器人。

說明

SyncRobots用於利用各動作命令所支援的SYNC參數，啟動已進行動作預約的機器人。以SyncRobots指定的機器人按相同時序開始動作。與透過一般的多工程式等待I/O信號事件進行同步相比，無切換工作的影響，因此可更正確地進行機器人啟動的同步。

若指定未預約動作的機器人編號，則會發生錯誤。

參照

SyncRobots函數

SyncRobots範例

以下範例表示，使用動作命令的SYNC參數和SyncRobots，同時開始2台機器人的動作。

```
Function Main
    Xqt Func1
    Xqt Func2
    Do
       Wait 0.1
       If (SyncRobots And &H03) = &H03 Then
        Exit Do
      EndIf
    Loop
SyncRobots 1,2
Fend

Function Func1
  Robot 1
  Motor On
  Go P1 SYNC
Fend

Function Func2
  Robot 2
  Motor On
  Go P1 SYNC
Fend
```

3.22.64 SyncRobots函數

本函數用於傳回動作預約中的機器人狀態。

格式

SyncRobots

傳回值

機器人處於動作預約狀態時，傳回將與該機器人編號對應的位元作為「1」的整數值；處於非預約狀態時，則傳回作為「0」的整數值。

- 位元0：機器人編號1
- 位元1：機器人編號2
- ：
- 位元15：機器人編號16

說明

SyncRobots函數用於使用機器人動作命令的SYNC參數檢查機器人動作預約的狀態。以與機器人編號對應的位元狀態表示以SyncRobots檢查的狀態。各位元分別顯示：已預約機器人動作 (1)；未預約機器人動作 (0)。可以SyncRobots命令開始運作已預約的機器人。

參照

SyncRobots

SyncRobots函數範例

以下範例表示，使用動作命令的SYNC參數和SyncRobots，同時開始2台機器人的動作。

```
Function Main
    Xqt Func1
    Xqt Func2
    Do
       Wait 0.1
       If (SyncRobots And &H03) = &H03 Then
        Exit Do
      EndIf
    Loop
SyncRobots 1,2
Fend

Function Func1
  Robot 1
  Motor On
  Go P1 SYNC
Fend

Function Func2
  Robot 2
  Motor On
  Go P1 SYNC
Fend
```

3.22.65 SysConfig

用於顯示系統設定參數。

格式

SysConfig

結果

用於顯示系統設定參數。

說明

顯示目前設定的值，以作為系統控制資料。交付機器人時或變更資料時，請儲存該資料。透過Epson RC+的[工具] – [維修]選單 – [控制器設定備份]進行儲存。

會顯示如下資料。(以下數值僅為範例，實際數值因控制器機種而異。)

```
' Version:
'   Firmware 1, 0, 0, 0

' Options:
'   External Control Point
'   RC+ API

' HOUR: 414.634

' Controller:
'   Serial #: 0001

' ROBOT 1:
' Name: Mnp01
' Model: PS3-AS10
' Serial #: 0001
' Motor On Time: 32.738
'   Motor 1: Enabled, Power = 400
'   Motor 2: Enabled, Power = 400
'   Motor 3: Enabled, Power = 200
'   Motor 4: Enabled, Power = 50
'   Motor 5: Enabled, Power = 50
'   Motor 6: Enabled, Power = 50

  ARCH 0, 30, 30
  ARCH 1, 40, 40
  ARCH 2, 50, 50
  ARCH 3, 60, 60
  ARCH 4, 70, 70
  ARCH 5, 80, 80
  ARCH 6, 90, 90
  ARMSET 0, 0, 0, 0, 0, 0
  HOFS 0, 0, 0, 0, 0, 0
  HORDR 63, 0, 0, 0, 0, 0
  RANGE -7427414, 7427414, -8738134, 2621440, -3145728, 8301227, -5534152, 5534152,
-3640889, 3640889, -6553600, 6553600
  BASE 0, 0, 0, 0, 0, 0
  WEIGHT 2, 0
  INERTIA  0.1, 0
  XYLIM 0, 0, 0, 0, 0, 0


' Extended I/O Boards:
'   1: Installed
'   2: Installed
'   3: None installed
```

```
'   4: None installed

' Fieldbus I/O Slave Board:
'   Installed
'   Type: PROFIBUS

' Fieldbus I/O Master Board:
'   None installed

' RS232C Boards:
'   1: Installed
'   2: None installed

' PG Boards:
'   1: None installed
'   2: None installed
'   3: None installed
'   4: None installed
```

SysConfig範例

```
> SysConfig
```

3.22.66 SysErr函數

用於傳回最新錯誤狀態或警告狀態。

格式

SysErr [(資訊編號)]

參數

資訊編號

以整數值指定要讀取錯誤代碼還是警告代碼。(可省略)0：錯誤代碼 (省略時)1：警告代碼

傳回值

用於以整數值傳回控制器的錯誤代碼或警告代碼。

說明

本函數僅用於NoEmgAbort工作(在Xqt時，指定NoEmgAbort開始的特殊工作)和背景工作。

控制器的錯誤代碼和警告代碼是指顯示於LCD的錯誤代碼和警告代碼。

未發生錯誤或警告時，則傳回「0」。

參照

ErrMsg$、ErrorOn、Trap、Xqt

SysErr函數範例

以下範例是由控制器監視錯誤狀態，若發生錯誤，則依錯誤編號啟用/停用I/O的程式。

注意

- Forced旗標

  在本程式範例中，在ON/OFF命令中指定Forced旗標。

  在發生錯誤時、緊急停止時或打開安全門時，I/O輸出會發生變化，因此在系統設計上需加以注意。

- 發生錯誤後的處理

  如本範例所示，在發生錯誤時執行必要的處理之後，請迅速結束工作。

```
Function main

Xqt ErrorMonitor, NoEmgAbort
:
:
Fend

Function ErrorMonitor
  Wait ErrorOn
  If 4000 < SysErr Then
    Print "Mortion Error = ", SysErr
    Off 10, Forced
    On 12, Forced
  Else
    Print "Other Error = ", SysErr
    Off 11, Forced
    On 13, Forced
  EndIf
```

```
Fend
```

3.23 T

3.23.1 Tab$函數

用於傳回指定數量的半型標籤字串。

格式

Tab$ (指定數量)

參數

指定數

以整數值指定半型標籤數。

傳回值

用於傳回含有半型標籤字元的字串。

說明

用於傳回指定數量的半型標籤字串。

參照

Left$、Mid$、Right$、Space$

Tab$函數範例

```
Print "X", Tab$(1), "Y"
Print
For i = 1 To 10
    Print x(i), Tab$(1), y(i)
Next i
```

3.23.2 Tan函數

用於傳回指定數值的正切值。

格式

Tan (角度)

參數

角度

指定表示角度的實數值。

傳回值

用於以數值傳回表示指定角度數值的正切值。

說明

Tan用於傳回正切值。參數單位必須是弧度。

若要將弧度轉換為角度，則使用RadToDeg函數。

參照

Abs、Atan、Atan2、Cos、Int、Mod、Not、Sgn、Sin、Sqr、Str$、Val

Tan函數範例

```
Function tantest
    Real num
    Print "Enter number in radians to calculate tangent for:"
    Input num
    Print "The tangent of ", num, "is ", Tan(num)
Fend
```

以下是在命令視窗中使用Tan函數的範例。

```
> print tan(0)
0.00
> print tan(45)
1.6197751905439
>
```

3.23.3 TargetOK函數

用於傳回可否從目前位置至目標坐標進行PTP (point to point)動作。

格式

TargetOK (目標坐標)

參數

目標坐標

以點數據指定用於調查可否進行動作的目標坐標。

傳回值

若能進行從目前位置到目標坐標為止的PTP動作，則傳回「True」；除此之外則傳回「False」。

說明

在實際運作(機器人)前確認可否到達目標坐標或形成目標姿態。但不考慮到達目標坐標的軌道。

參照

CurPos、FindPos、InPos、WaitPos

TargetOK函數範例

```
If TargetOK(P1) Then
  Go P1
EndIf

If TargetOK(P10 /L /F) Then
  Go P10 /L /F
EndIf
```

3.23.4 TaskDone函數

用於確認結束工作的函數。

格式

TaskDone (工作識別碼)

參數

任務識別碼

以整數值或運算式指定任務名稱或任務編號。工作名稱用於指定Xqt陳述式所用的函式名稱或者在Run視窗或操作員視窗中啟動的函式。

任務編號的指定(整數)

- 一般任務：1~32
- 背景任務：65~80
- Trap任務：257~267

傳回值

工作結束時，傳回「True」；除此之外，傳回「False」。

說明

TaskDone用於確認工作是否結束。

參照

TaskState、TaskWait

TaskDone函數範例

```
Xqt 2, conveyor
Do
    .
    .
Loop Until TaskDone(conveyor)
```

3.23.5 TaskInfo函數

用於傳回工作的狀態資訊。

格式

TaskInfo (工作識別碼, 指數)

參數

任務識別碼

以整數值指定任務名稱或任務編號。工作名稱用於指定Xqt陳述式所用的函式名稱或者在Run視窗或操作員視窗中啟動的函式。

任務編號的指定(整數)

- 一般任務：1~32
- 背景任務：65~80
- Trap任務：257~267

指數

以整數值指定所要查找的資訊指數。

傳回值

用於以整數值傳回指定資訊。

說明

指數	說明
0	工作編號
1	0 - 背景工作 1 - 一般工作、NoPause工作或NoEmgAbort工作
2	工作類型 0 - 一般工作 以Xqt不指定任何工作或指定Normal開始的工作 1 - NoPause工作 以Xqt指定NoPause並開始的工作 2 - NoEmgAbort工作 以Xqt指定NoEmgAbort並開始的工作 3 - Trap工作 4 - 背景工作
3	-1 - 未執行指定工作 1 - 指定工作執行中 2 - 指定工作等待事件 3 - 指定工作處於暫停或Halt狀態 4 - 指定工作處於快速暫停狀態 5 - 指定工作處於錯誤狀態
4	等待事件時發生超時(與TW相同)
5	事件等待時間(msec)
6	工作選擇的機器人編號
7	工作正在使用的機器人編號

參照

CtrlInfo、RobotInfo、TaskInfo$函數

TaskInfo函數範例

```
If (TaskInfo(1, 3) <> 0 Then
  Print "Task 1 is runnning"
Else
  Print "Task 1 is not running"
EndIf
```

3.23.6 TaskInfo$函數

用於傳回工作的文字資訊。

格式

TaskInfo$ (工作識別碼, 指數)

參數

任務識別碼

以整數值指定任務名稱或任務編號。工作名稱用於指定Xqt陳述式所用的函式名稱或者在Run視窗或操作員視窗中啟動的函式。

任務編號的指定(整數)

- 一般任務：1~32
- 背景任務：65~80
- Trap任務：257~267

指數

以整數值指定所要查找的資訊指數。

傳回值

用於以字串傳回指定資訊。

說明

下表所述為可以TaskInfo$搜尋的資訊。

指數	說明
0	工作名稱
1	開始日期與時間
2	目前執行的函式之名稱
3	含有函式的程式之行編號

參照

CtrlInfo、RobotInfo、TaskInfo

TaskInfo$函數範例

```
Print "Task 1 started: "TaskInfo$(1, 1)
```

3.23.7 TaskState函數

本函數用於取得工作目前狀態。

格式

TaskState (工作識別碼)

參數

任務識別碼

以整數值指定任務名稱或任務編號。工作名稱用於指定Xqt陳述式所用的函式名稱或者在Run視窗或操作員視窗中啟動的函式。

任務編號的指定(整數)

- 一般任務：1~32
- 背景任務：65~80
- Trap任務：257~267

傳回值

- 0：未執行指定任務
- 1：指定任務執行中
- 2：指定任務等待事件
- 3：指定任務暫停中
- 4：指定任務快速暫停中
- 5：指定任務錯誤狀態

說明

TaskState用於取得工作編號或以工作名稱指定的工作之目前狀態。

參照

TaskDone、TaskWait

TaskState函數範例

```
If TaskState(conveyor) = 0 Then
    Xqt 2, conveyor
EndIf
```

3.23.8 TaskWait

用於等待指定工作結束。

格式

TaskWait (工作識別碼)

參數

任務識別碼

以整數值或運算式指定任務名稱或任務編號。工作名稱用於指定Xqt陳述式所用的函式名稱或者在Run視窗或操作員視窗中啟動的函式。

任務編號的指定(整數)

- 一般任務：1~32
- 背景任務：65~80
- Trap任務：257~267

參照

TaskDone、TaskState

TaskWait範例

```
Xqt 2, conveyor
TaskWait conveyor
```

3.23.9 TC

用於設定扭矩控制模式及顯示目前模式。

格式
(1) TC { On | Off }

(2) TC

參數

On | Off
- On：啟用扭矩控制模式。
- Off：停用扭矩控制模式。

傳回值
在省略參數時，顯示目前扭矩控制模式。

說明
以TC On/Off啟用或停用扭矩控制模式。

扭矩控制模式是透過設定馬達輸出限制值產生固定力的一種模式，用於如下情形：用固定的力使抓手按壓標的物；讓抓手保持接觸標的物的狀態，同時配合標的物的動作進行模仿動作等。

啟用扭矩控制前，請以TCLim設定扭矩限制值。

機器人會進行動作，以便在扭矩控制模式下執行動作命令時對目標位置進行定位。當機器人接觸標的物，馬達輸出到達扭矩限制值時，機器人會停止並維持固定扭矩。

在以下任一情況下，停用扭矩控制模式。
- 啟動控制器時
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

參照
TCLim 、TCSpeed

TC範例1

```
Speed 5
Go ApproachPoint

'將z軸扭矩限制值設為20。
TCLim -1, -1, 20, -1

TC On
Go TargetPoint
Wait 3
Go ApproachPoint
TC Off
```

注意

- 搭載Safety板的控制器偵測到位置異常時

請將程式修正為不使「機器人的目前位置」與「機器人目前的動作目標位置」相距過遠。

相距過遠時，Safety板會判斷為故障，發生「No.9801錯誤 Safety板偵測到位置錯誤」的錯誤。(搭載有Safety板的控制器)

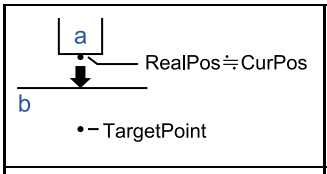
機器人的目前位置可用RealPos函數取得。

機器人目前的動作目標位置可用CurPos函數取得。

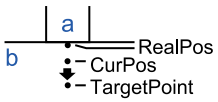
若為SCARA型機器人，RealPos的差與CurPos的差建議使用約在J1：10度、J2：10度、J3：40mm、J4：90度以下的數值。

發生錯誤的數值依各機種而不同。在不會發生錯誤的範圍內，可依照各機種分別設定較大數值。

使用TCSpeed限制速度會導致在接觸物件之前，RealPos和CurPos之間發生差異。請不要使用TCSpeed，或者，使用TCSpeed時將其設置為與Speed相同的值。



1. 未接觸目標對象時



2. 接觸目標對象時

符號	說明
a	夾具
b	目標對象

1. 機器人的目前位置(RealPos)與機器人目前的動作目標位置(CurPos)為相近的位置。
2. 機器人的目前位置(RealPos)維持在與標的物接觸的位置。
   機器人目前的動作目標位置(CurPos)朝向TagetPoint行進。
   若拉開一定以上的距離，將會發生No.9801錯誤。

不使Safety板發生位置異常的使用方法，如下列程式範本所示。

使用Till命令，在「機器人的目前位置」與「機器人目前的動作目標位置」具有一定以上的差距下，進行下一步處理，即可避免錯誤。

將差距盡量設定在較小的數值，可較快進行下一步處理，進而改善生產節拍時間。

TC範例2

```
Speed 5
Go ApproachPoint

'將Z軸扭矩極限設定為20%。
TCLim -1, -1, 20, -1

Xqt posDiffChk(10.0) '啟動位置差的檢查工作。J3軸若產生10.0mm以上的差，將設置旗標

TC On
Go TargetPoint Till MemSw(0)
Wait 3
Go ApproachPoint
TC Off
```

```
Function posDiffChk(Zth As Double)
    Do
        If (Abs(CZ(RealPos) - CZ(CurPos)) > Zth) Then
'「機器人的目前位置」與「機器人目前的目標位置」之差是否超過了閾值？
            MemOn (0) ' 位置偏差過大旗標 ON
        Else
            MemOff (0) ' 位置偏差過大旗標 清除
        EndIf
        Wait 0.01
    Loop
Fend
```

3.23.10 TCLim

用於設定為扭矩控制模式的各關節扭矩限制值。

格式

TCLim [第1關節扭矩限制值, 第2關節扭矩限制值, 第3關節扭矩限制值, 第4關節扭矩限制值 [, 第5關節扭矩限制值] [, 第6關節扭矩限制值] [, 第7關節扭矩限制值] [, 第8關節扭矩限制值] [, 第9關節扭矩限制值]]

參數

第1關節扭矩限制值
　以運算式或數值指定對於瞬間最大扭矩的比例(1~100的整數，單位： %)。設為-1時，停用扭矩限制值，變為一般的位置控制模式。
第2關節扭矩限制值
　以運算式或數值指定對於瞬間最大扭矩的比例(1~100的整數，單位： %)。設為-1時，停用扭矩限制值，變為一般的位置控制模式。
第3關節扭矩限制值
　以運算式或數值指定對於瞬間最大扭矩的比例(1~100的整數，單位： %)。設為-1時，停用扭矩限制值，變為一般的位置控制模式。
第4關節扭矩限制值
　以運算式或數值指定對於瞬間最大扭矩的比例(1~100的整數，單位： %)。設為-1時，停用扭矩限制值，變為一般的位置控制模式。
第5關節扭矩限制值
　可省略。以運算式或數值指定對於瞬間最大扭矩的比例(1~100的整數，單位： %)。設為-1時，停用扭矩限制值，變為一般的位置控制模式。
第6關節扭矩限制值
　可省略。以運算式或數值指定對於瞬間最大扭矩的比例(1~100的整數，單位： %)。設為-1時，停用扭矩限制值，變為一般的位置控制模式。
第7關節扭矩限制值
　可省略。以運算式或數值指定對於瞬間最大扭矩的比例(1~100的整數，單位： %)。設為-1時，停用扭矩限制值，變為一般的位置控制模式。
第8關節扭矩限制值
　可省略。以運算式或數值指定對於S軸瞬間最大扭矩的比例(1~100的整數，單位： %)。設為-1時，停用扭矩限制值，變為一般的位置控制模式。
第9關節扭矩限制值
　可省略。以運算式或數值指定對於T軸瞬間最大扭矩的比例(1~100的整數，單位： %)。設為-1時，停用扭矩限制值，變為一般的位置控制模式。

傳回值

在省略參數時，顯示目前扭矩限制值。

說明

請在TC Off時設定扭矩限制值。設為TC On後，設定的值即為有效。

若設定值過低，機器人則不會動作，將在尚未到達目標位置之前結束動作命令。

在以下任一情況下，TCLim設定值會被初始化。

- 啟動控制器時
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

參照

TC、TCLim函數、TCSpeed

TCLim範例

```
Speed 5
Go ApproachPoint

' 將Z軸扭矩極限設定為20%。
TCLim -1, -1, 20, -1

TC On
Go TargetPoint
Wait 3
Go ApproachPoint
TC Off
```

注意

- 搭載Safety板的控制器偵測到位置異常時

  請將程式修正為不使「機器人的目前位置」與「機器人目前的動作目標位置」相距過遠。

  相距過遠時，Safety板會判斷為故障，發生「No.9801錯誤 Safety板偵測到位置錯誤」的錯誤。(搭載有Safety板的控制器)

  如需詳細資訊，請參閱TC命令之說明。

3.23.11 TCLim函數

用於傳回指定關節的扭矩限制值。

格式

TCLim (關節編號)

參數

關節編號

以運算式或數值指定欲讀取扭矩限制值的關節編號。附加軸的S軸為8，T軸為9。

傳回值

用於傳回表示目前扭矩限制值的1~100的整數。停用扭矩限制值時，則傳回-1。

參照

TC、TCLim、TCSpeed

TCLim函數範例

```
Print "目前的Z軸扭矩限制值：", TCLim(3)
```

3.23.12 TCPSpeed函數

用於傳回計算獲得的目前工具中心點(TCP)速度。

格式
TCPSpeed

傳回值
用於以實數傳回計算獲得的目前工具中心點速度。(單位：mm/秒)

說明
以mm/秒為單位傳回執行CP動作命令時計算獲得的工具中心點速度。

CP動作命令是指Move、TMove、Arc、Arc3、CVMove以及Jump3CP。此速度不同於實際速度。此速度是在叫用函式時由系統建立動作計畫的工具中心點速度。

不考慮馬達的追蹤延遲。機器人正在執行PTP動作命令時，則傳回「0」。

即便使用附加軸，也只傳回機器人的移動速度。例如，即便是將附加軸用作行走軸的情況，也不考量附加軸的移動速度。

參照
AccelS、CurPos、InPos、SpeedS

TCPSpeed函數範例

```
Function MoveTest
AccelS 4000, 4000
SpeedS 200
Xqt ShowTCPSpeed
Do
  Move P1
  Move P2
Loop
Fend

Function ShowTCPSpeed
  Do
    Print "Current TCP speed is: ", TCPSpeed
    Wait .1
  Loop
Fend
```

3.23.13 TCSpeed

用於設定扭矩控制中的速度限制值。

格式

TCSpeed [速度]

參數

速度

以運算式或數值指定對於最大速度的比例(1~100的整數，單位：%)。

說明

在扭矩控制時，無論Speed命令等速度設定如何，皆受限於以TCSpeed設定的速度。

若在扭矩控制時檢測到限制值以上的速度，則發生錯誤。

在以下任一情況下，TCSpeed設定值會被初始化為100%。

- 啟動控制器時
- 執行Motor On
- 執行SFree、SLock、Brake
- 執行Reset、Reset Error
- 因停止按鈕、執行Quit All等而結束工作

參照

TC、TCLim、TCSpeed函數

TCSpeed範例

```
Speed 5
Go ApproachPoint

' 將z軸扭矩限制值設為20%。
TCLim -1, -1, 20, -1
' 將扭矩控制中的速度設為5 (與Speed設置相同)。
TcSpeed 5

TC On
Go TargetPoint
Wait 3
Go ApproachPoint
TC Off
```

注意

- 搭載Safety板的控制器偵測到位置異常時

  請將程式修正為不使「機器人的目前位置」與「機器人目前的動作目標位置」相距過遠。

  使用TCSpeed限制速度會導致「機器人的目前位置」與「機器人目前的動作目標位置」之間發生差異。請不要使用TCSpeed，或者，使用 TCSpeed時將其設置為與Speed相同的值。

  相距過遠時，Safety板會判斷為故障，發生「No.9801錯誤 Safety板偵測到位置錯誤」的錯誤。(搭載有Safety板的控制器)

  如需詳細資訊，請參閱TC命令之說明。

3.23.14 TCSpeed函數

用於傳回扭矩控制中的速度限制值。

格式

TCSpeed

傳回值

用於傳回表示目前速度限制值的1~100的整數。

參照

TC、TCSpeed、TCLim

TCSpeed範例

```
Integer var
var = TCSpeed
```

3.23.15 TeachOn函數

用於傳回教導模式的狀態。

格式

TeachOn

傳回值

處於教導模式時，傳回「True」；除此之外，傳回「False」。

說明

本函數僅用於背景工作。

參照

ErrorOn、EstopOn、SafetyOn、Xqt

TeachOn函數範例

以下範例是由控制器監視進入教導模式，若轉移到教導模式，則啟用/停用I/O的程式。

```
Function BGMain
  Do
    Wait 0.1
    If TeachOn = True Then
      On teachBit
    Else
      Off teachBit
    EndIf
    If SafetyOn = True Then
      On safetyBit
    Else
      Off safetyBit
    EndIf
    If PauseOn = True Then
      On PauseBit
    Else
      Off PauseBit
    EndIf
  Loop

Fend
```

3.23.16 TGo

用於執行目前工具坐標系中的位移PTP動作。

格式

TGo 目標坐標 [CP] [Till | Find] [!平行處理!] [SYNC]

參數

目標坐標
　　使用點數據指定動作的目標位置。
CP
　　指定路徑運動。可省略。
Till | Find
　　用於記述Till或Find運算式。可省略。

```
Till | Find
Till Sw(運算式) = {On | Off}
Find Sw(運算式) = {On | Off}
```

! 平行處理!
　　可附加平行處理陳述式，以便在動作中執行I/O等命令。可省略。
SYNC
　　用於預約動作命令。在以SyncRobots開始動作之前，機器人不進行動作。

說明

用於執行目前工具坐標系中的位移PTP動作。

忽略點資料所賦予的姿態旗標，維持目前姿態旗標。不過，在垂直6軸型機器人(包含N系列)的情況下，將姿態旗標自動變更為縮小關節移動量的狀態。

透過使用Till修飾詞使Till條件成立時，可在動作中途使機器人減速並停止，以完成TGo動作。

動作中當Find條件變為真時，則利用Find修飾詞將點儲存在FindPos中。

可利用!平行處理!，與動作平行進行其它處理。

若已添加CP參數，可在開始動作減速時重疊下一個動作命令的加速。此時，不在目標坐標上進行定位。

參照

Accel、CP、Find、!平行處理!、P# = 點指定、Speed、Till、TMove、Tool

TGo範例

```
> TGo XY (100, 0, 0, 0)    '(工具坐標系上)朝X方向移動100 mm
Function TGoTest

  Speed 50
  Accel 50, 50
  Power High

  Tool 0
  P1 = XY(300, 300, -20, 0)
  P2 = XY(300, 300, -20, 0) /L

  Go P1
  Print Here
  TGo XY(0, 0, -30, 0)
  Print Here
```

```
  Go P2
  Print Here
  TGo XY(0, 0, -30, 0)
  Print Here

Fend
```

[輸出結果]

```
 X:  300.000 Y:  300.000 Z:  -20.000 U:    0.000 V:    0.000 W:    0.000 /R /0
 X:  300.000 Y:  300.000 Z:  -50.000 U:    0.000 V:    0.000 W:    0.000 /R /0
 X:  300.000 Y:  300.000 Z:  -20.000 U:    0.000 V:    0.000 W:    0.000 /L /0
 X:  300.000 Y:  300.000 Z:  -50.000 U:    0.000 V:    0.000 W:    0.000 /L /0
```

3.23.17 Till

用於以Jump、Go、Move或其它動作命令指定Till時，在動作途中停止，並設定和顯示結束處理的條件。

格式

Till [條件運算式]

參數

事件條件運算式

用於指定作為觸發器的輸入狀態。

[事件] 比較運算符 (=, <>, >=, >, <, <=) [整數運算式]

可將以下函數或變數用於事件。

- 函數：Sw、In、InW、Oport、Out、OutW、MemSw、MemIn、MemInW、Ctr、GetRobotInsideBox、GetRobotInsidePlane、Force、AIO_In、AIO_InW、AIO_Out、AIO_OutW、Hand_On、Hand_Off、SF_GetStatus
- 變數：Byte、Int32、Integer、Long、Short、UByte、UInt32、UShort型的備份變數、全域變數、模組變數

此外，可用以下運算符，對複數事件條件運算式使用遮罩或進行複合組合。

- 運算符：And、Or、Xor

[例]

```
Till Sw(5) = On
Till Sw(5) = On And Till(6) = Off
```

說明

請單獨記述Till陳述式，或作為動作命令陳述式的修飾詞進行記述。

必須在Till條件運算式中包含1個以上的上述函數。

Till條件運算式中含有變數時，則在設定Till條件時運算該值。可能會變成非預期條件，因此建議在條件運算式中不使用變數。還可記述數個Till陳述式。屆時，最後執行的Till條件會處於啟用狀態。

若省略參數，則顯示目前Till設定。

注意

- 電源ON時的Till設定

  電源ON時的Till條件初始設定為Till Sw(0) = On。設為當輸入位元編號0為ON時減速並停止。

- 檢查Till條件成立的Stat函數和TillOn函數

  在執行使用Till修飾詞的動作命令之後，可用Stat函數或TillOn函數檢查是否成立Till條件。

- 在條件運算式中使用變數時
  - 可使用的變數型態為整數型(Byte, Int32, Integer, Long, Short, UByte, UInt32, UShort)。
  - 不可使用陣列變數。
  - 不可使用本地變數。
  - 變數值未滿足條件的時間超過0.01秒時，系統可能無法檢測到變數變化。
  - 系統內可使用的變數等待數量有限。1個系統內可使用的變數等待數量最多64個(也包括Wait等條件運算式所用的變數等待數量)。若超過最大數量，則在建置專案時將發生錯誤。
  - 若以ByRef傳址要執行變數等待的變數，則發生錯誤。
  - 若條件運算式右邊的整數運算式中含有變數，則在開始動作命令時運算該值。可能會變成非預期條件，因此建議在整數運算式中不使用變數。

參照

Find、Go、In、InW、Jump、MemIn、MemSw、Move、Stat、Sw、TillOn、SF_GetStatus

Till範例

以下是在程式中使用Till命令的範例。

```
Till Sw(1) = Off                    '設定Till條件(停用輸入位元1)
Go P1 Till                          '滿足前一行的條件時停止
Till Sw(1) = On And Sw($1) = On     '設定新的Till條件
Move P2 Till                        '滿足前一行的條件時停止
Move P5 Till Sw(10) = On            '滿足該行條件時停止
```

3.23.18 TillOn函數

用於傳回Till的狀態。

格式

TillOn

傳回值

以使用Till前的動作命令成立Till條件後傳回True。

說明

Till條件成立後，傳回True。

TillOn與以下相同。

```
((Stat(1) And 2) <> 0)
```

參照

EStopOn、SafetyOn、Sense、Stat、Till

TillOn函數範例

```
Go P0 Till Sw(1) = On
If TillOn Then
    Print "Till condition occurred during move to P0"
EndIf
```

3.23.19 Time

用於顯示時間。

格式
Time

說明
以24小時制顯示目前時間。

參照
Date、Time$

Time範例
命令視窗中的執行範例

```
> Time
10:15:32
```

3.23.20 Time函數

用於傳回控制器的累計通電時間。

格式

Time (單位指定)

參數

單位指定

指定0~2的整數值。以0~2表示控制器累計通電時間的單位。

- 0：時
- 1：分
- 2：秒

傳回值

用於以實數值傳回控制器的累計通電時間。

參照

Hour

Time函數範例

```
Function main
    Integer t_h, t_m, t_s

    t_h = Time(0)    '以小時傳回通電時間
    t_m = Time(1)    '以分鐘傳回通電時間
    t_s = Time(2)    '以秒傳回通電時間
    Print "This controller has been used:"
    Print t_h, "hours, ",
    Print t_m, "minutes, ",
    Print t_s, "seconds"
Fend
```

3.23.21 Time$函數

用於傳回目前系統時間。

格式
Time$

傳回值
以24小時制字串傳回時間。

格式為hh:mm:ss [小時:分鐘:秒]。

參照
Date、Date$、Time

Time$函數範例

```
Print "The current time is: ", Time$
```

3.23.22 TLClr

清除工具坐標系的設定。

格式

TLClr 工具編號

參數

工具編號

以整數或運算式指定要清除的工具。(工具0為預設值，不可清除。)

參照

Arm、ArmClr、ArmSet、ECPSet、Local、LocalClr、Tool、TLSet

TLClr範例

```
TLClr 1
```

3.23.23 TLDef函數

用於傳回工具的設定狀態。

格式

TLDef (工具編號)

參數

工具編號
以整數或運算式指定要傳回狀態的工具。

傳回值

若設定指定的工具，則傳回「True」；若未設定，則傳回「False」。

參照

Arm、ArmClr、ArmSet、ECPSet、Local、LocalClr、Tool、TLClr、TLSet

TLDef函數範例

```
Function DisplayToolDef(toolNum As Integer)

   If TlDef(toolNum) = False Then
      Print "Tool ", toolNum, "is not defined"
   Else
      Print "Tool ", toolNum, ": ",
      Print TlSet(toolNum)
   EndIf
Fend
```

3.23.24 TLSet

設定並顯示工具坐標系。

格式

(1) TLSet 工具坐標系編號, 工具設定資料

(2) TLSet 工具坐標系編號

(3) TLSet

參數

工具坐標系編號
　　以1~15的整數值指定要設定的工具。(Tool 0為預設工具，不可變更。)
工具設定資料
　　以P編號、P(運算式)、點標籤、點運算式之一指定要設定的工具坐標系原點及方向。

結果

若省略所有參數，則顯示所有TLSet設定。

若僅指定工具編號，則顯示指定的TLSet設定。

說明

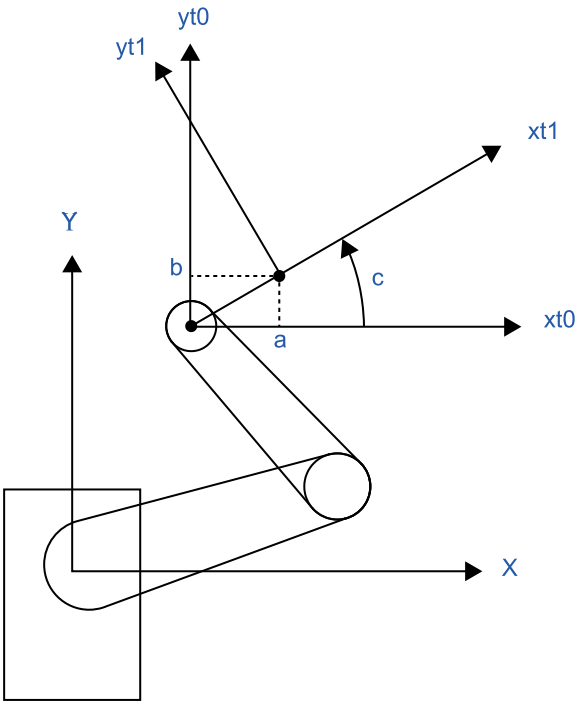
指定相對於Tool 0坐標系(抓手坐標系)的相對原點位置和相對旋轉角度，定義工具坐標系Tool 1、Tool 2、Tool 3。

```
TLSet 1, XY(50,100,-20,30)
TLSet 2, P10 +X(20)
```

在上述情況下，參照坐標值P10，在X值上加上20。會忽略手臂屬性和本地坐標系編號。

```
TLSET 1, XY(100,60,-20,30)
      a      b   c   d   e
```

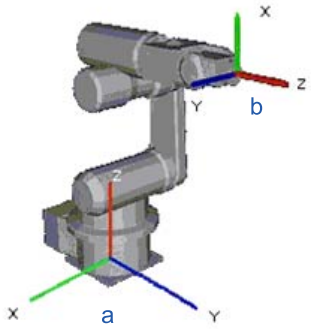
符號	說明
a	工具坐標系編號
b	工具坐標系的原點X軸方向位置(下圖a)
c	工具坐標系的原點Y軸方向位置(下圖b)
d	工具坐標系的原點Z軸方向位置
e	工具坐標系的旋轉角度(下圖c)



符號	說明
X, Y	機器人坐標系
xt0, yt0	工具0坐標系
xt1, yt1	工具1坐標系

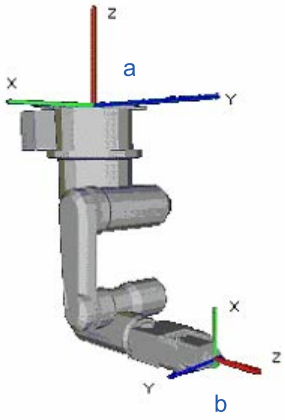
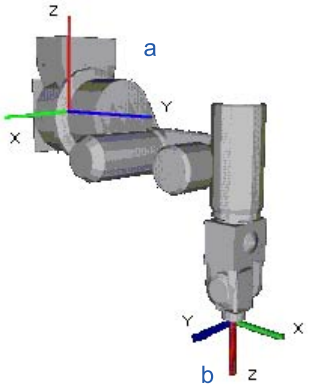
垂直6軸型機器人使用TLSet

在垂直6軸型機器人中，若將所有關節設為0度位置，則以第6關節的法蘭面中心為原點，垂直向上方向上有X軸，機器人坐標系X軸方向上有Y軸，在相對於第6關節法蘭面的垂直方向上有Z軸的坐標系為工具0坐標系。(參照下圖)



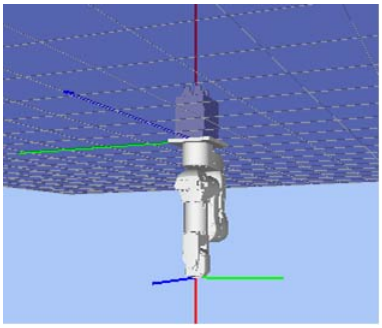
符號	說明
a	機器人坐標系
b	工具0坐標系

工具0坐標系的定義因垂直6軸型機器人的設置方法而異。

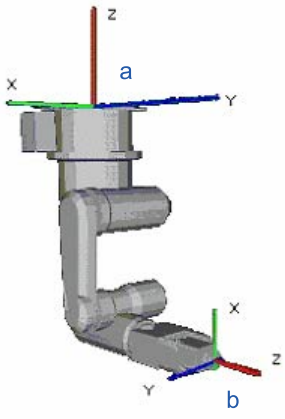
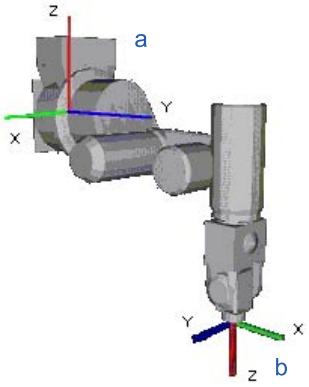
懸掛設置	壁掛設置
	
a：機器人坐標系 b：工具0坐標系	

N系列使用TLSet

在N系列中，若將所有關節設為0度位置，機器人坐標系-X軸方向上有X軸，機器人坐標系Y軸方向上有Y軸，機器人坐標系-Z軸方向上有Z軸的坐標系為工具0坐標系。(參照下圖)



工具0坐標系的定義因N系列的設置方法而異。

懸掛設置	壁掛設置
	
a：機器人坐標系 b：工具0坐標系	

注意

- 維持TLSet值

  維持TLSet值。若要清除工具定義，請使用TLClr。

參照

Tool、Arm、ArmSet、TLSet函數、TLClr

TLSet範例

以下是在命令視窗中執行操作的範例，用於理解進行工具定義時和未進行工具定義時的動作差異。

```
> TLSet 1, XY (100, 0, 0, 0) 'Tool 1(從抓手的坐標系朝X方向設定100 mm)
                             '在工具坐標系定義
> Tool 1                     '選擇以TLSet定義的Tool 1
> TGo P1                     '將Tool 1的目標坐標設為P1
> Tool 0                     '設為往後的動作不使用工具
> Go P1                      '將U關節的中央設為P1
```

3.23.25 TLSet函數

用於傳回設定的工具坐標系資料。

格式

TLSet (工具坐標系編號)

參數

工具坐標系編號
以整數值指定工具編號。

傳回值

用於傳回工具坐標系資料。

參照

TLSet

TLSet函數範例

```
P1 = TLSet(1)
```

3.23.26 TMOut

設定執行Wait命令時發生超時錯誤(錯誤2280)之前的時間。

格式

TMOut 秒

參數

秒

以整數值指定超時時間。範圍：0~2147483。(單位：秒)

說明

TMOut用於設定執行Wait命令時發生超時錯誤(錯誤2280)之前的時間。若將超時時間設為0秒，則超時設定不起作用，在Wait命令所指定的條件尚未成立之前，無限期地等待。

TMOut的預設值為「0」。

參照

In、MemSw、OnErr、Sw、TW、Wait

TMOut範例

```
TMOut 5
Wait MemSw(0) = On
```

3.23.27 TMove

用於執行目前工具坐標系中的位移直線內插動作。

格式

TMove 目標坐標 [ROT] [CP] [Till | Find] [!平行處理!] [SYNC]

參數

目標坐標
　使用點數據指定動作的目標位置。
ROT
　以工具姿態變化為優先，決定動作速度和加減速度。可省略。
CP
　指定路徑運動。可省略。
Till | Find
　用於記述Till或Find運算式。可省略。

```
Till | Find
Till Sw(運算式) = {On | Off}
Find Sw(運算式) = {On | Off}
```

! 平行處理 !
　可附加平行處理陳述式，以便在動作中執行I/O等命令。可省略。
SYNC
　用於預約動作命令。在以SyncRobots開始動作之前，機器人不進行動作。

說明

用於執行目前工具坐標系中的位移直線內插動作。

忽略點資料所賦予的姿態旗標，維持目前姿態旗標。不過，在垂直6軸型機器人(包含N系列)的情況下，將姿態旗標自動變更為縮小關節移動量的狀態。這等同於透過Move命令指定LJM修飾參數時的情況。若要進行180度以上的姿態變化，請分次執行。

TMove的速度和加減速度分別使用SpeedS和AccelS的設定值。關於速度和加減速度的關係，請參閱「注意」項目中的「連同CP使用TMove」。但是，使用ROT修飾參數時的速度和加減速度分別使用SpeedR和AccelR的設定值。此時，SpeedS和AccelS的設定值即呈停用狀態。

機器人的工具中心點位置固定為特定坐標，僅改變工具姿態時，或者相對於工具中心點的移動距離，工具姿態變化較大時，工具姿態變化速度可能會明顯變快。為防止這種情形，工具姿態變化速度大時，自動限制動作速度的功能就會啟用。

想要手動設定CP動作時的工具姿態變化速度的上限值時，請啟用SpeedRLimitation。啟用SpeedRLimitation時，CP動作時的工具姿態變化速度超過設定的SpeedR時，工具姿態變化速度會變為SpeedR來限制動作速度。工具姿態變化速度超過設定的SpeedR時，以設定的SpeedS進行動作。工具姿態變化速度的上限值，請事先在SpeedR設定。

透過使用Till修飾詞使Till條件成立時，可在動作中途使機器人減速並停止，以完成TMove。

動作中當Find條件的值變為真(True)時，則用Find修飾詞將點資料儲存在FindPos中。

可利用!平行處理!，與動作平行進行其它處理。

注意

- 連同CP使用TMove

若使用CP參數，則在開始減速的同時，移至下一個陳述式控制動作命令。其便利之處在於，使用者可連接數個動作命令，以恆定的速度執行連續動作。在未指定CP的TMove命令時，手臂一定會減速，並停於指定的目標坐標上。

參照

AccelS、CP、Find、! 平行處理 !、P#= 點指定、SpeedS、TGo、Till、Tool

TMove範例

```
> TMove XY (100, 0, 0, 0)  '(工具坐標系上)朝X方向移動100 mm
Function TMoveTest

  Speed 50
  Accel 50, 50
  SpeedS 100
  AccelS 1000, 1000
  Power High

  Tool 0
  P1 = XY(300, 300, -20, 0)
  P2 = XY(300, 300, -20, 0) /L

  Go P1
  Print Here
  TMove XY(0, 0, -30, 0)
  Print Here

  Go P2
  Print Here
  TMove XY(0, 0, -30, 0)
  Print Here

Fend
```

[輸出結果]

```
 X:  300.000 Y:  300.000 Z:  -20.000 U:    0.000 V:    0.000 W:    0.000 /R /0
 X:  300.000 Y:  300.000 Z:  -50.000 U:    0.000 V:    0.000 W:    0.000 /R /0
 X:  300.000 Y:  300.000 Z:  -20.000 U:    0.000 V:    0.000 W:    0.000 /L /0
 X:  300.000 Y:  300.000 Z:  -50.000 U:    0.000 V:    0.000 W:    0.000 /L /0
```

3.23.28 Tmr函數

Tmr函數用於以秒為單位傳回計時器啟動後的經過時間。

格式

Tmr (計時器編號)

參數

計時器編號

以整數(0~63)或數值指定64個計時器之中要檢查的計時器。

傳回值

用於以實數值(單位：秒)傳回指定計時器的經過時間。計時器的範圍為0~約1.7E+31。計時器解像度為0.001秒。

說明

傳回指定計時器啟動之後的經過時間。與ElapsedTime函數不同，程式處於暫停狀態的時間也算作經過時間。

可以TmReset重設計時器。

```
Real overhead

TmReset 0
overHead = Tmr(0)
```

參照

ElapsedTime函數、TmReset

Tmr函數範例

```
TmReset 0           '重設計時器0
For i = 1 To 10     '執行10次
    GoSub Cycle
Next
Print Tmr(0) / 10   '計算並顯示週期時間
```

3.23.29 TmReset

用於重設Tmr函數所使用的計時器。

格式

TmReset 計時器編號

參數

計時器編號

以整數(0~63)指定64個計時器之中要進行重置的計時器編號。

說明

重設並啟動以計時器編號指定的計時器。

Tmr函數用於取得指定計時器的經過時間。

參照

Tmr

TmReset範例

```
TmReset 0           '重設計時器0
For i = 1 To 10     '執行10次
    GoSub CYL
Next
Print Tmr(0)/10     '計算並顯示週期時間
```

3.23.30 Toff

關閉LCD上的執行行顯示。

格式
Toff

說明
若執行Toff，LCD上則不顯示工作的執行行。

注意

- 支援的控制器型號

  不支援RC90/T/VT系列。

參照
Ton

Toff範例
以下是命令視窗中的測試範例。有助於理解工具設定時和未設定時的動作差異。

```
Function main
  Ton MyTask
  ...
  Toff
Fend
```

3.23.31 Ton

用於指定在LCD上顯示執行行的工作。

格式

Ton 工作識別碼

Ton

參數

任務識別碼

以整數值或運算式指定任務名稱或任務編號。工作名稱用於指定Xqt陳述式所用的函式名稱或者在Run視窗或操作員視窗中啟動的函式。

任務編號的指定(整數)

- 一般任務：1~32

注意

- 支援的控制器型號

  不支援RC90/T/VT系列。

說明

在初始狀態下顯示工作編號1的執行行。

若使用Ton，則可在LCD上顯示指定工作的執行行。

若省略工作識別碼，則在LCD上顯示已執行Ton的工作之執行行。

參照

Toff

Ton範例

```
Function main
  Ton MyTask
  ...
  Toff
Fend
```

3.23.32 Tool

用於選擇工具或顯示選擇的工具編號。

格式

(1) Tool 工具編號

(2) Tool

參數

工具編號

以緊接於後的動作命令從16個工具(整數值0~15)中指定要使用的工具。可省略。

結果

若省略參數，則顯示目前設定的工具編號。

說明

Tool用於選擇以工具編號指定的工具。工具編號為「0」時，沒有選擇工具，對尖端關節(旋轉關節)中心執行所有動作。但是，選擇工具編號「1」、「2」、「3」等時，則對已設定的工具之尖端執行動作。

注意

- 關閉電源對工具選擇帶來的影響

  即使關閉電源，也不更改選擇的工具坐標系。

參照

TGo、TLSet、TMove

Tool範例

以下是命令視窗中的測試範例。有助於理解工具設定時和未設定時的動作差異。

```
>tlset 1, 100, 0, 0, 0    '將Tool 1(從抓手的坐標系朝X方向設定100mm)
                          '在工具坐標系定義
>tool 1                   '選擇以TLSet定義的Tool 1
>tgo p1                   '將Tool 1的目標坐標設為P1
>tool 0                   '設為往後的動作不使用工具
>go p1                    '將U關節的中央設為P1
```

3.23.33 Tool函數

用於傳回目前設定的工具編號。

格式

Tool

傳回值

用於以整數值傳回工具編號。

參照

Tool

Tool函數範例

```
Integer savTool

savTool = Tool
Tool 2
Go P1
Tool savTool
```

3.23.34 Trap(使用者定義觸發)

用於定義插斷以及發生插斷時的處理。

若使用Trap命令，則可因發生事件而跳到標籤或叫用函式。Trap命令有2種類型。

- 將使用者所定義的輸入狀態作為觸發的4個Trap
- 將系統狀態作為觸發的7個Trap。

在本項目中，對使用者定義觸發的Trap進行說明。

格式

Trap Trap編號, 條件運算式 GoTo 標籤

Trap Trap編號, 條件運算式 Call 函式名稱

Trap Trap編號, 條件運算式 Xqt 函式名稱

Trap Trap編號

參數

Trap編號

以運算式或數值指定Trap編號(1~4的整數)。(SPEL+最多支援4個同時啟用的Trap。)

事件條件運算式

用於指定作為觸發器的輸入狀態。

[事件] 比較運算符(=, <>, >=, >, <, <=) [整數運算式]

可將以下函數或變數用於事件。

- 函數：Sw、In、InW、Oport、Out、OutW、MemSw、MemIn、MemInW、Ctr、GetRobotInsideBox、GetRobotInsidePlane、AIO_In、AIO_InW、AIO_Out、AIO_OutW、Hand_On、Hand_Off、SF_GetStatus
- 變數：Byte、Int32、Integer、Long、Short、UByte、UInt32、UShort型的備份變數、全域變數、模組變數

此外，可用以下運算符，對複數事件條件運算式使用遮罩或進行複合組合。

- 運算符：And、Or、Xor

[例]

```
Trap 1, Sw(5) = On Call, TrapFunc
Trap 1, Sw(5) = On And Till(6) = Off, Call TrapFunc
```

標籤

在Trap條件成立時，讓程式執行轉移的目的地標籤。

函數名稱

Trap條件成立時執行Call或Xqt的函數。不可指定有引數的函式。

說明

Trap用於在條件成立時執行以GoTo、Call、Xqt等指定的插斷處理。必須在Trap條件運算式中包含1個以上的上述函數。

Trap條件運算式中含有變數時，則在設定Trap條件時運算該值。可能會變成非預期條件，因此建議在條件運算式中不使用變數。

一旦執行插入處理，則清除該Trap設定。若要執行相同的插斷處理，必須再度執行Trap命令。

若要取消Trap設定，則僅指定Trap編號參數執行Trap命令。

[例]「Trap 3」用於取消Trap #3。

此外，若從宣告的函式退出，則自動取消Trap Goto。

若完成宣告的工作，則取消Trap Call。

在尚未完成所有工作之前，不會取消Trap Xqt。

指定GoTo時

在設定Trap的工作中，會對執行中的命令進行如下處理，並將控制轉移到指定目的地標籤。

- 立即暫停手臂動作(快速暫停)。
- 中斷因Wait或Input命令發生的等待狀態。
- 其它命令的執行皆於轉移控制前完成。

指定Call時

執行與GoTo相同的處理後，將控制轉移至指定目的地的函式。

函式執行完成後，程式會返回到發生插斷時的下一個陳述式。

Call不可用於Trap處理的函式。

此外，若在Trap處理的函式中發生錯誤，透過OnErr進行的錯誤處理會變為停用狀態，而且必定發生錯誤。

指定Xqt時

透過程式控制生成指定函式，將其做為插斷處理專用工作。此時，會繼續執行Trap命令的工作。

無法在插斷處理的工作中以Xqt執行其它工作。

注意

- 面向Epson RC+4.x使用者

  在Epson RC+ 8.0中，已以Trap Xqt取代Epson RC+4.x之前版本的Trap Call功能。

  在Epson RC+ 8.0中，已刪除Epson RC+4.x之前版本的Trap GoSub功能。請以Trap Call代替。

- 在條件運算式中使用變數時
  - 可使用的變數型態為整數型(Byte, Int32, Integer, Long, Short, UByte, UInt32, UShort)。
  - 不可使用陣列變數。
  - 不可使用本地變數。
  - 變數值未滿足條件的時間超過0.01秒時，系統可能無法檢測到變數變化。
  - 系統內可使用的變數等待數量有限。1個系統內可使用的變數等待數量最多64個(也包括Wait等條件運算式所用的變數等待數量)。若超過最大數量，則在建置專案時將發生錯誤。
  - 若以ByRef傳址要執行變數等待的變數，則發生錯誤。
  - 若條件運算式右邊的整數運算式中含有變數，則在設定Trap條件時運算該值。可能會變成非預期條件，因此建議在整數運算式中不使用變數。

參照

Call、GoTo、Xqt、SF_GetStatus

Trap範例

[例1] 使用者定義的錯誤處理

Sw(0) Input用於輸入使用者定義的錯誤。

```
Function Main
    Trap 1, Sw(0)= On GoTo EHandle  '定義Trap
    .
    .
    .
EHandle:
    On 31                           '信號塔燈亮燈
    OpenCom #1
    Print #1, "Error is issued"
    CloseCom #1
Fend
```

[例2] 多工的使用

```
Function Main
    Trap 2, MemSw(0) = On Or MemSw(1) = On Call Feeder
    .
    .
    .
Fend
.


Function Feeder
  Select TRUE
    Case MemSw(0) = On
      MemOff 0
      On 2
    Case MemSw(1) = On
      MemOff 1
      On 3
  Send

  '重新設定下一個週期的Trap
  Trap 2, MemSw(0) = On Or MemSw(1) = On Call Feeder
Fend
```

[例3] 將全域變數用於條件

```
Global Integer gi

Function main
  Trap 1, gi = 5 GoTo THandle
  Xqt sub
  Wait 100
  Exit Function

THandle:
  Print "IN Trap ", gi

Fend

Function sub
  For gi = 0 To 10
    Print gi
    Wait 0.5
  Next
Fend
```

3.23.35 Trap(系統狀態觸發)

用於定義插斷以及發生插斷時的處理。

若使用Trap命令，則可因發生事件而跳到標籤或叫用函式。Trap命令有2種類型。

- 將使用者所定義的輸入狀態作為觸發的4個Trap
- 將系統狀態作為觸發的7個Trap

在本項目中，對系統狀態觸發的Trap進行說明。

格式

Trap {Emergency | Error | Pause | SGOpen | SGClose | Abort | Finish } Xqt 函數名稱

Trap {Emergency | Error | Pause | SGOpen | SGClose | Abort | Finish }

參數

Emergency
　　發生緊急停止時，執行指定函數。
Error
　　發生錯誤時，執行指定函數。
Pause
　　進入暫停狀態時，執行指定函數。
SGOpen
　　開啟安全門回路時，執行指定函數。
SGClose
　　關閉安全門回路時，執行指定函數。
Abort
　　由用戶或系統而停止所有工作(背景工作除外)時(已執行相當於Abort All的命令或按下中斷按鈕時)，執行指定函式。
Finish
　　完成所有任務(背景任務除外)時，執行指定函數。不會按Trap Abort的執行條件執行函式。
函數名稱
　　在系統狀態成立時用於進行Xqt的插斷處理任務之函數。不可指定有引數的函式。不過，若在參數中指定「Error」，可指定3個引數。

注意

在Epson RC+ 8.0中，已以Trap *** Xqt取代Epson RC+4.x之前版本的Trap *** Call功能。

說明

系統狀態成立時，執行指定的插斷處理工作。

即便執行插斷處理工作，也不清除Trap設定。

若要清除Trap設定，則省略函式名稱並執行Trap命令。

[例]「Trap Emergency」用於清除「Trap Emergency」。

若結束所有一般工作，控制器處於Ready狀態，則清除所有Trap設定。

無法在插斷處理的工作中以Xqt執行其它工作。

⚠ 注意

- Forced旗標

  即便處於緊急停止狀態、安全門開啟狀態、教導模式或發生錯誤的狀態，也可透過在On、Off等的I/O輸出命令上指定Forced旗標，開啟/關閉I/O輸出。

  若要進行指定Forced旗標的I/O輸出，切勿連接伴隨有機械動作的外部設備(如傳動器)。否則，外部設備可能會在緊急停止狀態、安全門開啟狀態、教導模式或發生錯誤狀態下運作，這會帶來極大的危險。

  指定Forced旗標的I/O輸出用於設想連接不會伴隨有機械動作的外部設備(如狀態顯示LED)。

- 指定Emergency時

  發生緊急停止時，按照NoEmgAbort工作屬性執行指定函式。

可在插斷處理工作中執行的命令為可執行NoEmgAbort工作的命令。

請於完成緊急停止的插斷處理後迅速完成工作。若未完成工作，控制器則不會變為Ready狀態。無法在插斷處理工作中執行Reset命令並自動解除緊急停止。

若要在插斷處理工作中進行I/O的啟用/停用工作，請取消[控制器設定] - [環境設定] - [以緊急停止停用輸出連接埠]核取方塊的勾選狀態。若此設定處於勾選狀態，則無法確定優先執行的動作(以控制器停用I/O或以工作啟用I/O)。

- 指定Error時

  發生錯誤時，按照NoEmgAbort工作屬性執行指定的函式。

可在插斷處理工作中執行的命令為可執行NoEmgAbort工作的命令。

請於完成錯誤的插斷處理後迅速完成工作。若未完成工作，控制器則不會變為Ready狀態。

可在使用者函式中指定可省略的3個參數(錯誤編號、機器人編號、關節編號)。若要使用這些參數，請在Trap函式中附加3個byval整數參數。

若發生動作系統的錯誤，則設定錯誤編號、機器人編號、關節編號。

若發生動作系統以外的錯誤，則將機器人編號、關節編號設為「0」。

- 指定Pause時

  進入暫停狀態時，按照NoPause工作屬性執行指定函式。

- 指定SGOpen時

  開啟安全門回路時，則按照NoPause工作屬性執行指定函式。

- 指定SGClose時

  安全門回路關閉且被閂鎖時，則按照NoPause工作屬性執行指定函式。

若在插斷處理工作中執行Cont命令，則會發生錯誤。

- 指定Abort時

  由使用者或系統而停止所有工作(背景工作除外)時(執行相當於Abort All的命令或按下中斷按鈕時)，則按照NoPause工作屬性執行指定函式。

請於完成中斷的插斷處理後迅速完成工作。若未完成工作，控制器則不會變為Ready狀態。

即便在以Trap Abort執行的工作中發生錯誤，也不執行Trap Error的處理工作。

若以Shutdown命令或Restart命令中斷工作，則不執行Trap Abort和Trap Finish的處理工作。

- 指定Finish時

  完成所有工作(背景工作除外)時，則按照NoPause工作屬性執行指定函式。不會按執行Trap Abort處理工作的條件執行。

請於完成終止的插斷處理後迅速完成工作。若未完成工作，控制器則不會變為Ready狀態。

參照

Era, Erl, Err, Ert, ErrMsg$, OnErr, Reset, Restart, SysErr, Xqt

Trap範例

```
Function main
    :
  Trap Error Xqt suberr
    :
Fend

Function suberr
  Print "Error =", Err
  On ErrorSwitch
Fend

Function main

  Trap Error Xqt trapError

FEnd


Function trapError(errNum As Integer, robotNum As Integer, jointNum As Integer)
Print "error number = ", errNum
Print "robot number = ", robotNum
Print "joint number = ", jointNum
If Ert = 0 Then
  Print "system error"
Else
  Print "task error"
  Print "function = ", Erf$(Ert)
  Print "line number = ", Erl(Ert)
EndIf
FEnd
```

3.23.36 Trim$函數

用於在前後不含半型空格的狀態下，傳回與指定字串相同的字串。

格式

Trim$ (字串)

參數

字串

指定字串運算式。

傳回值

指定的字串前後含有半型空格時，則刪除半型空格。

參照

LTrim$、RTrim$

Trim$函數範例

```
str$ = "  data  "
str$ = Trim$(str$)  ' str$ = "data"
```

3.23.37 TW函數

用於傳回Wait命令、WaitNet命令、WaitSig命令的狀態。

格式

TW

傳回值

在時間內成立Wait狀態時，則傳回「False」。

若發生超時，則傳回「True」。

說明

先前的Wait命令條件成立時，則傳回「False」；發生超時，傳回「True」。

參照

TMOut、Wait、Hand_TW

TW函數範例

```
Wait Sw(0) = On, 5     '在啟用輸入位元0之前等待5秒鐘
If TW = True Then
    Print "Time Up"    '超過5秒鐘則顯示「Time Up」
EndIf
```

3.24 U

3.24.1 UBound函數

用於傳回可在指定陣列中設定的元素編號之最大值。

格式

UBound (陣列變數名稱 [, 維度])

參數

陣列變數名稱

依照平常的變數名稱命名慣例進行命名。

維度

以如下整數值設定要傳回元素編號最大值的維度。可省略。省略時則假定為「1」。

- 1：第1維度
- 2：第2維度
- 3：第3維度

參照

Redim

UBound函數範例

```
Integer i, a(10)

For i = 0 to UBound(a)
    a(i) = i
Next
```

3.24.2 UByte

宣告UByte型變數。(無符號整數型，大小：2位元組)

格式

UByte 變數名稱 [(陣列變數的最大元素編號)] [, 變數名稱[(陣列變數的最大元素編號)]…]

參數

變數名稱

指定宣告為UByte型的變數名稱。

陣列變數的最大元素編號

是陣列變數的最大元素編號，最大可宣告三維。使用如下格式。可省略。

(最大元素編號1, [最大元素編號2], [最大元素編號3])

元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。在所有元素數不超過以下最大值的範圍內，指定各最大元素編號。

- 本地變數：2,000
- 備份變數(Global Preserve)：4,000
- 全域變數和模組變數：100,000

說明

UByte用於將變數宣告為UByte型。UByte變數的範圍為0~255。在Function的開頭宣告本地變數。在Function之外宣告全域變數和模組變數。

參照

Boolean、Byte、Double、Global、Int32、Int64、Integer、Long、Real、Short、String、UInt32、UInt64、UShort

UByte範例

在以下範例中宣告UByte型變數，並對該變數賦予數值。

確認到變數test_ok的最高位元是1或0。將該結果顯示於顯示器上。(在本例中，對變數賦予15的值，因此始終有設定變數test_ok值較高的位元。)

```
Function Test
    UByte A(10)          'UByte型的一維陣列
    UByte B(10, 10)      'UByte型的二維陣列
    UByte C(5, 5, 5)     'UByte型的三維陣列
    UByte test_ok
    test_ok = 15
    Print "Initial Value of test_ok = ", test_ok
    test_ok = (test_ok And 8)
    If test_ok <> 8 Then
       Print "test_ok high bit is ON"
    Else
       Print "test_ok high bit is OFF"
    EndIf
Fend
```

3.24.3 UCase$函數

用於傳回從小寫格式轉換為大寫格式的字串。

格式

UCase$ (字串)

參數

字串

指定要轉換為大寫格式的字串。

傳回值

用於傳回轉換為大寫格式的字串。

參照

LCase$、LTrim$、Trim$、RTrim$

UCase$函數範例

```
str$ = "Data"
str$ = UCase$(str$)  ' str$ = "DATA"
```

3.24.4 UInt32

宣告UInt32型變數。(無符號4位元組整數型變數)

格式

UInt32 變數名稱 [(陣列變數的最大元素編號)] [, 變數名稱 [(陣列變數的最大元素編號)]...]

參數

變數名稱

指定要進行變數宣告的變數名稱。

陣列變數的最大元素編號

是陣列變數的最大元素編號，最大可宣告三維。使用如下格式。可省略。

(最大元素編號1, [最大元素編號2], [最大元素編號3])

元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。在所有元素數不超過以下最大值的範圍內，指定各最大元素編號。

- 本地變數：2,000
- 備份變數(Global Preserve)：4,000
- 全域變數和模組變數：100,000

說明

UInt32用於宣告整數型變數。整數型變數範圍：0~4294967295。在Function的開頭宣告本地變數。在Function之外宣告全域變數和模組變數。

參照

Boolean、Byte、Double、Global、Int32、Int64、Integer、Long、Real、Short、String、UByte、UInt64、UShort

UInt32範例

以下為使用UInt32宣告整數型變數的程式範例。

```
Function uint32test
    UInt32 A(10)          'UInt32型的一維陣列
    UInt32 B(10, 10)      'UInt32型的二維陣列
    UInt32 C(5, 5, 5)     'UInt32型的三維陣列
    UInt32 var1, arrayvar(10)
    Integer i
    Print "Please enter an Integer Number"
    Input var1
    Print "The Integer variable var1 = ", var1
    For i = 1 To 5
        Print "Please enter an Integer Number"
        Input arrayvar(i)
        Print "Value Entered was ", arrayvar(i)
    Next i
Fend
```

3.24.5 Uint64

宣告Uint64型變數。(無符號8位元組整數型變數)

格式

Uint64 變數名稱 [(陣列變數的最大元素編號)] [, 變數名稱 [(陣列變數的最大元素編號)]...]

參數

變數名稱

指定要進行變數宣告的變數名稱。

陣列變數的最大元素編號

是陣列變數的最大元素編號，最大可宣告三維。使用如下格式。可省略。

(最大元素編號1, [最大元素編號2], [最大元素編號3])

元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。在所有元素數不超過以下最大值的範圍內，指定各最大元素編號。

- 本地變數：2,000
- 備份變數(Global Preserve)：4,000
- 全域變數和模組變數：100,000

說明

UInt64用於宣告整數型變數。整數型變數範圍：0~18446744073709551615。在Function的開頭宣告本地變數。在Function之外宣告全域變數和模組變數。

參照

Boolean、Byte、Double、Global、Int32、Int64、Integer、Long、Real、Short、String、UByte、UInt32、UShort

UInt64範例

以下為使用UInt64宣告整數型變數的程式範例。

```
Function uint64test
    UInt64 A(10)          'UInt64型的一維陣列
    UInt64 B(10, 10)      'UInt64型的二維陣列
    UInt64 C(5, 5, 5)     'UInt64型的三維陣列
    UInt64 var1, arrayvar(10)
    Integer i
    Print "Please enter an Integer Number"
    Input var1
    Print "The Integer variable var1 = ", var1
    For i = 1 To 5
        Print "Please enter an Integer Number"
        Input arrayvar(i)
        Print "Value Entered was ", arrayvar(i)
    Next i
Fend
```

3.24.6 UOpen

在讀出和寫入兩種模式下開啟檔案。

格式

UOpen 檔名As #檔案編號 . . Close #檔案編號

參數

檔案名稱

指定包含路徑的檔案名稱字串。僅指定檔名時，即指目前目錄中的檔案。詳細內容請參閱ChDisk。

檔案編號

以30~63的整數值或運算式來指定。

說明

以指定的檔案編號開啟指定的檔案。此陳述式用於寫入指定檔案或讀出資料。

注意

可使用網路路徑。

若指定不存在的檔案，則建立該檔案並寫入資料。若指定存在的檔案，則從既有資料的開頭讀寫資料。

可用Seek命令切換檔案的載入/寫入位置(指標)。若要切換載入存取和寫入存取，請用Seek命令重新設定檔案指標。

在檔案處於開啟狀態下，指定的檔案編號用於識別該檔案。因此，在關閉該檔案之前，不可將相同的檔案編號用於其它檔案。在檔案操作命令(Print#, Input#, Read, Write, Seek, Eof, Flush, Close)中使用檔案編號。

以Close陳述式關閉檔案，並釋放檔案編號。

請以FreeFile函數取得檔案編號，以避免將同一編號用在多項工作上。

參照

Close、Print #、Input#、AOpen、BOpen、ROpen、WOpen、FreeFile、Seek

UOpen範例

```
Integer fileNum, i, j

fileNum = FreeFile
UOpen "TEST.DAT" As #fileNum
For i = 0 To 100
   Print #fileNum, i
Next i
Close #fileNum

fileNum = FreeFile
UOpen "TEST.DAT" As #fileNum
Seek #fileNum, 10
Input #fileNum, j
Print "data = ", j
Close #fileNum
```

3.24.7 UpdateDB

用於更新開啟的資料庫內的已搜尋表格中之資料。

格式

UpdateDB #資料庫編號, 項目,值

參數

資料庫編號
　　用於指定以OpenDB指定的資料庫編號(501~508的整數值)。
項目
　　指定要更新的表格項目名稱。
值
　　指定要更新的值。

說明

按照指定的項目值更新處於開啟狀態的資料庫內之已選表格中資料。

務必在更新資料前執行SelectDB，並選擇要更新的記錄。

注意

需連接有安裝RC+的PC。

參照

OpenDB、CloseDB、SelectDB、DeleteDB

UpdateDB範例

使用SQL資料庫的範例 以下所示為在SQL服務器2000範例資料庫Northwind的表格Employees中註冊資料，並更新已註冊資料項目的簡易範例。

```
Integer count, i, eid
String Lastname$, Firstname$, Title$

OpenDB #501, SQL, "(LOCAL)", "Northwind"
count = SelectDB(#501, "Employees", "TitleOfCourtesy = 'Mr.'")
Print #501, "Epson", "Taro", "Engineer", "Mr."
count = SelectDB(#501, "Employees", "LastName = 'Epson' and FirstName = 'Taro'")
Input #501, eid, Lastname$, Firstname$, Title$
Print eid, ",", Lastname$, ",", Firstname$, ",", Title$
UpdateDB #501, "Title", "Chief Engineer"
count = SelectDB(#501, "Employees", "LastName = 'Epson' and FirstName = 'Taro'")
Input #501, eid, Lastname$, Firstname$, Title$
Print eid, ",", Lastname$, ",", Firstname$, ",", Title$
CloseDB #501
```

3.24.8 UShort

宣告UShort型變數。(無符號2位元組整數型變數)

格式

UShort 變數名稱 [(陣列變數的最大元素編號)] [, 變數名稱 [(陣列變數的最大元素編號)]...]

參數

變數名稱

指定要進行變數宣告的變數名稱。

陣列變數的最大元素編號

是陣列變數的最大元素編號，最大可宣告三維。使用如下格式。可省略。

(最大元素編號1, [最大元素編號2], [最大元素編號3])

元素編號是從0開始的，因此元素數是最大元素編號上加上1的數。在所有元素數不超過以下最大值的範圍內，指定各最大元素編號。

- 本地變數：2,000
- 備份變數(Global Preserve)：4,000
- 全域變數和模組變數：100,000

說明

UShort用於宣告整數型變數。整數型變數範圍：0~65535。在Function的開頭宣告本地變數。在Function之外宣告全域變數和模組變數。

參照

Boolean、Byte、Double、Global、Int32、Int64、Integer、Long、Real、Short、String、UByte、UInt32、UInt64

UShort範例

以下為使用UShort宣告整數型變數的程式範例。

```
Function ushorttest
    UShort A(10)          'UShort型的一維陣列
    UShort B(10, 10)      'UShort型的二維陣列
    UShort C(5, 5, 5)     'UShort型的三維陣列
    UShort var1, arrayvar(10)
    Integer i
    Print "Please enter an Integer Number"
    Input var1
    Print "The Integer variable var1 = ", var1
    For i = 1 To 5
        Print "Please enter an Integer Number"
        Input arrayvar(i)
        Print "Value Entered was ", arrayvar(i)
    Next i
Fend
```

3.25 V

3.25.1 Val函數

用於將由數字構成的指定字串轉換為數值，並傳回該值。

格式
Val (字串)

參數

字串
　　指定僅由數字構成的字串運算式。字串中也含有前置詞。
　　&H (16進位)、&O (8進位)、&B (2進位)

傳回值
以整數或輸入的字串運算式傳回浮點數。含有小數點的輸入字串運算式會被轉換為浮點數。除此以外，以整數值傳回值。

說明
Val用於將由數字構成的字串運算式轉換為數值。以整數或浮點數顯示結果。若對Val命令賦予含有小數點的輸入字串運算式，則傳回浮點數。除此之外，傳回整數值。

參照
Abs、Asc、Chr$、Int、Left$、Len、Mid$、Mod、Right$、Sgn、Space$、Str$

Val函數範例
以下是將幾個不同字串運算式轉換為數值，然後在畫面上顯示結果的程式範例。

```
Function ValDemo
    String realstr$, intstr$
    Real realsqr, realvar
    Integer intsqr, intvar

    realstr$ = "2.5"
    realvar = Val(realstr$)
    realsqr = realvar * realvar
    Print "The value of ", realstr$, " squared is: ", realsqr

    intstr$ = "25"
    intvar = Val(intstr$)
    intsqr = intvar * intvar
    Print "The value of ", intstr$, " squared is: ", intsqr
Fend
```

以下是命令視窗中的操作範例。

```
> Print Val("25.999")
25.999
>
```

3.25.2 VSD

用於設定SCARA機器人的變速CP動作功能。

格式
VSD { ON | Off }

參數

On | Off

- On：啟用SCARA機器人的變速CP動作功能。
- Off：停用SCARA機器人的變速CP動作功能。

說明
以下列命令啟用VSD。
Move, Arc, Arc3

本命令僅可用於SCARA機器人。對於SCARA機器人以外的機種，請使用AvoidSinularity SING_VSD。

變速CP動作功能用於防止在SCARA機器人執行CP動作時發生加速度錯誤或過速度錯誤。該功能作用在於，在維持動作軌跡的狀態下自動限制關節速度進行動作。關節速度受限時，雖然不維持以SpeedS設定的工具中心點速度，但關節速度低於限制值時，則恢復為原有的工具中心點速度。若要以等速度為優先，請降低AccelS、DecelS、SpeedS的值，以防發生錯誤。

即便使用VSD仍發生加速度錯誤或過速度錯誤時，請降低AccelS、DecelS、SpeedS的值。

若有變更VSD設定值，則於下一次啟動控制器前即可啟用。

控制器啟動時，VSD處於OFF狀態。

參照
VSD函數

VSD範例

```
VSD On '啟用變速CP動作以進行運作
Move P1
Move P2
VSD Off
```

3.25.3 VSD函數

用於傳回SCARA機器人的變速CP動作功能之設定。

格式
VSD

傳回值

- On = 啟用變速CP動作功能
- Off = 停用變速CP動作功能

參照
VSD

VSD函數範例

```
If VSD = Off Then
    Print "Variable Speed Drive is off"
EndIf
```

3.25.4 VxCalib

請以Vision Guide以外的、客戶自備的視覺系統使用此命令。

本命令用於建立客戶自備之視覺系統的校正資料。

格式

(1) VxCalib CalNo

(2) VxCalib CalNo, CamOrient, P(pixel_st : pixel_ed), P(robot_st : robot_ed) [, TwoRefPoints]

(3) VxCalib CalNo, CamOrient, P(pixel_st : pixel_ed), P(robot_st : robot_ed),P(ref0) [, P(ref180)]

參數

CalNo

以整數值指定校正資料的編號。可用0~15的整數(合計16個)定義。

CamOrient

以如下整數值指定相機安裝方向。1~3只能以格式 (2) 指定。4~7只能以格式 (3) 指定。

- 1：固定相機
- 2：朝下固定相機
- 3：朝上固定相機
- 4：可移動相機 搭載第2軸
- 5：可移動相機 搭載第4軸
- 6：可移動相機 搭載第5軸
- 7：可移動相機 搭載第6軸

P(pixel_st : pixel_ed)

以連續點數據指定圖元坐標(僅限X、Y)。

P(robot_st : robot_ed)

以連續點數據指定機器人坐標。指定的點資料因以CamOrient指定的相機安裝方向而異。在CamOrient = 1~3的情況下，請以目前的TOOL和ARM設定機器人坐標。在CamOrient = 4~7的情況下，請以TOOL: 0、ARM: 0設定機器人坐標。

TwoRefPoints

格式 (1) 時可指定。使用2個測量點時，指定「True」; 使用1個測量點時，指定「False」。可透過指定2個測量點，實施更正確的校正。預設為「False」。可省略。

P(ref0)

格式 (3) 時可指定。以點資料指定基準點的機器人坐標。

P(ref180)

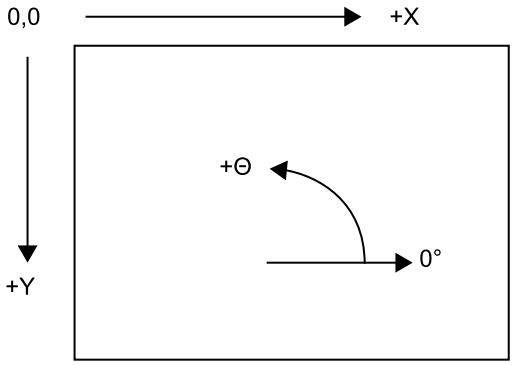
格式 (3) 時可指定。以點資料指定第2個基準點的機器人坐標。可透過指定2個基準點，實施更正確的校正。可省略。

說明

根據使用指定相機姿態的指定校正編號、以引數傳遞的像素坐標、機器人坐標、基準點(僅限可移動相機)，計算視覺校正資料。

若僅指定CalNo，則顯示定義時的點資料等(僅限命令視窗)。

下表所示為像素坐標的坐標系。單位是像素。



在設定像素坐標、機器人坐標時，請將畫面上的左上位置作為點1，並依下表順序將右下位置作為點9。以CamOrient、TwoRefPoints引數分為4種。

1) CamOrient = 1~3(固定、朝下固定、朝上固定)、TwoRefPoints = False時

資料順序	畫面位置	像素坐標	機器人坐標
1	左上	檢測坐標1	測量點坐標1
2	中上	檢測坐標2	測量點坐標2
3	右上	檢測坐標3	測量點坐標3
4	右中	檢測坐標4	測量點坐標4
5	中中	檢測坐標5	測量點坐標5
6	左中	檢測坐標6	測量點坐標6
7	左下	檢測坐標7	測量點坐標7
8	中下	檢測坐標8	測量點坐標8
9	右下	檢測坐標9	測量點坐標9

2) CamOrient = 2(朝下固定)、TwoRefPoints = True時

若正確定義工具，則不需要TwoRefPoints。請設為「False」。

若將TwoRefPoints設為「True」，則可使用2個測量點。因此，可實施更正確的校正。

僅限於機器人坐標需要18個點。

設定1~9的測量點坐標後，請將U軸旋轉180度並將抓手(連桿等)對準校正目標位置以設定測量點坐標10~18。

資料順序	畫面位置	像素坐標	機器人坐標
1	左上	檢測坐標1	測量點坐標1
2	中上	檢測坐標2	測量點坐標2
3	右上	檢測坐標3	測量點坐標3
4	右中	檢測坐標4	測量點坐標4
5	中中	檢測坐標5	測量點坐標5
6	左中	檢測坐標6	測量點坐標6
7	左下	檢測坐標7	測量點坐標7

資料順序	畫面位置	像素坐標	機器人坐標
8	中下	檢測坐標8	測量點坐標8
9	右下	檢測坐標9	測量點坐標9
10	左上	－－－	測量點坐標10
11	中上	－－－	測量點坐標11
12	右上	－－－	測量點坐標12
13	右中	－－－	測量點坐標13
14	中中	－－－	測量點坐標14
15	左中	－－－	測量點坐標15
16	左下	－－－	測量點坐標16
17	中下	－－－	測量點坐標17
18	右下	－－－	測量點坐標18

3) CamOrient = 3(朝上固定)、TwoRefPoints = True時

若正確定義工具，則不需要TwoRefPoints。請設為「False」。

若將TwoRefPoints設為「True」，則使用2個檢測點。因此，可實施更正確的校正。

僅限於像素坐標需要18個點。

在各測量點坐標設定1~9的檢測坐標後，請設定將U軸旋轉180度位置的檢測坐標10~18。

資料順序	畫面位置	像素坐標	機器人坐標
1	左上	檢測坐標1	測量點坐標1
2	中上	檢測坐標2	測量點坐標2
3	右上	檢測坐標3	測量點坐標3
4	右中	檢測坐標4	測量點坐標4
5	中中	檢測坐標5	測量點坐標5
6	左中	檢測坐標6	測量點坐標6
7	左下	檢測坐標7	測量點坐標7
8	中下	檢測坐標8	測量點坐標8
9	右下	檢測坐標9	測量點坐標9
10	左上	檢測坐標10	－－－
11	中上	檢測坐標11	－－－
12	右上	檢測坐標12	－－－
13	右中	檢測坐標13	－－－
14	中中	檢測坐標14	－－－

資料順序	畫面位置	像素坐標	機器人坐標
15	左中	檢測坐標15	－－－
16	左下	檢測坐標16	－－－
17	中下	檢測坐標17	－－－
18	右下	檢測坐標18	－－－

4) CamOrient = 4~7時

資料順序	畫面位置	像素坐標	機器人坐標
1	左上	檢測坐標1	測量點坐標1
2	中上	檢測坐標2	測量點坐標2
3	右上	檢測坐標3	測量點坐標3
4	右中	檢測坐標4	測量點坐標4
5	中中	檢測坐標5	測量點坐標5
6	左中	檢測坐標6	測量點坐標6
7	左下	檢測坐標7	測量點坐標7
8	中下	檢測坐標8	測量點坐標8
9	右下	檢測坐標9	測量點坐標9

注意

- 除了上表之外，請指定基準點的機器人坐標。

  可透過使用2個基準點，實施更正確的校正。此時，需要2個U 軸相差180度的點。

  設定第1個基準點坐標後，請將U軸旋轉180度並將抓手(連桿等)對準校正目標位置以設定第2個基準點坐標。若正確定義工具，則不需使用2個基準點。

參照

VxTrans函數、VxCalInfo函數、VxCalDelete、VxCalSave、VxCalLoad

VxCalib範例

```
Function MobileJ2

  Integer i
  Double d(8)

  Robot 1
  LoadPoints "MobileJ2.pts"

  VxCalib 0, 4, P(21:29), P(1:9), P(0)

  If (VxCalInfo(0, 1) = True) Then
    For i = 0 To 7
      d(i) = VxCalInfo(0, i + 2)
    Next i
    Print "Calibration result:"
```

```
    Print d(0), d(1), d(2), d(3), d(4), d(5), d(6), d(7)

    P52 = VxTrans(0, P51, P50)
    Print "Coordinates conversion result:"
    Print P52
    SavePoints "MobileJ2.pts"
    VxCalSave "MobileJ2.caa"
  Else
    Print "Calibration failed"
  EndIf

Fend
```

3.25.5 VxCalDelete

請以Vision Guide以外的、客戶自備的視覺系統使用此命令。

刪除客戶自備之視覺系統的校正資料。

格式

VxCalDelete CalNo

參數

CalNo

以整數值指定校正資料的編號。可用0~15的整數(合計16個)定義。

說明

刪除以指定校正編號定義的校正資料。

參照

VxCalib、VxTrans函數、VxCalInfo函數、VxCalSave、VxCalLoad

VxCalDelete範例

```
VxCalDelete "MobileJ2.caa"
```

3.25.6 VxCalLoad

請以Vision Guide以外的、客戶自備的視覺系統使用此命令。

從檔案載入客戶自備之視覺系統的校正資料。

格式

VxCalLoad FileName

參數

FileName

以字串運算式指定要載入校正資料的檔案名稱。副檔名固定為「.caa」。若省略，則新增「.caa」。若不是「.caa」，則轉換為「.caa」。不可指定路徑。

說明

從目前專案內的指定檔案載入校正資料。

參照

VxCalib、VxTrans函數、VxCalInfo函數、VxCalDelete、VxCalSave

VxCalLoad範例

```
VxCalLoad "MobileJ2.caa"
```

3.25.7 VxCalInfo函數

請以Vision Guide以外的、客戶自備的視覺系統使用此命令。

傳回客戶自備之視覺系統的校正完成狀態和校正資料。

格式

VxCalInfo (CalNo,CalData)

參數

CalNo

以整數值指定校正資料的編號。可用0~15的整數(合計16個)定義。

CalData

以下表所示的整數值指定欲讀取的校正資料種類。

CalData	校正資料種類
1	校正的完成狀態
2	X方向的平均偏差 [mm]
3	X方向的最大偏差 [mm]
4	每1像素的X方向長度 [mm]
5	X方向的傾斜度
6	Y方向的平均偏差 [mm]
7	Y方向的最大偏差 [mm]
8	每1像素的Y方向長度 [mm]
9	Y方向的傾斜度

傳回值

若為CalData = 1，則以Bool型值傳回指定的校正資料；若為CalData = 2~9，則以Double型值傳回指定的校正資料。

說明

可確認以哪一個校正編號來定義校正資料。

還可取得校正資料值。

參照

VxCalib、VxTrans函數、VxCalDelete、VxCalSave、VxCalLoad

VxCalInfo函數範例

```
Print VxCalInfo(0, 1)
```

3.25.8 VxCalSave

請以Vision Guide以外的、客戶自備的視覺系統使用此命令。

將客戶自備之視覺系統的校正資料儲存到檔案中。

格式

VxCalSave FileName

參數

FileName

以字串運算式指定要儲存校正資料的檔案名稱。副檔名固定為「.caa」。若省略，則自動新增「.caa」。若不是「.caa」，則自動轉換為「.caa」。不可指定路徑。

說明

使用指定檔名儲存校正資料。檔案會被儲存於目前專案中。存在同名檔案時，會覆蓋校正資料。

參照

VxCalib、VxTrans函數、VxCalInfo函數、VxCalDelete、VxCalLoad

VxCalSave範例

```
VxCalSave "MobileJ2.caa"
```

3.25.9 VxTrans函數

請以Vision Guide以外的、客戶自備的視覺系統使用此命令。

將像素坐標轉換為機器人坐標，並傳回已轉換的點資料。

格式

VxTrans (CalNo, P(pixel)[, P(camRobot)]) As Pose

參數

CalNo
　以整數值指定校正資料的編號。可用0~15的整數(合計16個)定義。
P(pixel)
　以點數據指定視覺圖元坐標(僅限X、Y、U)。
P(camRobot)
　可省略。若為可移動相機，則指定拍攝時的機器人位置。若省略，則使用機器人的目前位置。請將點資料設定為TOOL: 0、ARM: 0。

傳回值

以點資料傳回計算獲得的機器人坐標。

說明

使用指定校正編號的校正資料，並將像素坐標轉換為機器人坐標。

若用可移動相機指定非目前值，則指定拍攝時的機器人坐標P(camRobot)。請將P(camRobot)設定為TOOL: 0、ARM: 0。用於計算已設定機器人坐標的第4關節角度、第6關節角度。

參照

VxCalib、VxCalInfo函數、VxCalDelete、VxCalSave、VxCalLoad

VxTrans函數範例

```
P52 = VxTrans(0, P51, P50)
```

3.26 W

3.26.1 Wait

使用MemSw或Sw，在指定條件成立之前或在指定時間等待程式。(也可用Oport替代Sw，以檢查I/O輸出。)

也可等待全域變數的值發生變化。

格式

(1) Wait 時間

(2) Wait 條件運算式

(3) Wait 條件運算式, 時間

參數

時間

以0~2147483的實數(單位：秒)指定等待時間。最小有效位數為0.01秒。

事件條件運算式

以如下格式指定事件條件。

[事件] 比較運算符(=, <>, >=, >, <, <=) [整數運算式]

可將以下函數或變數用於事件。

- 函數：Sw、In、InW、Oport、Out、OutW、MemSw、MemIn、MemInW、Ctr,GetRobotInsideBox、GetRobotInsidePlane、MCalComplete,Motor、LOF、ErrorOn、SafetyOn、EstopOn、TeachOn 、Cnv_QueLen、WindowsStatus、AtHome、LatchState、WorkQue_Len、PauseOn、AIO_In、AIO_InW、AIO_Out、AIO_OutW、Hand_On、Hand_Off、SF_GetStatus
- 變數：Byte、Int32、Integer、Long、Short、UByte、UInt32、UShort型的備份變數、全域變數、模組變數

此外，可用以下運算符，對複數事件條件運算式使用遮罩或進行複合組合。

- 運算符：And、Or、Xor、Mask

說明

(1) 僅指定時間時

使用作為計時器的Wait命令時，只按指定時間暫停程式，其後繼續執行程式。

(2) 僅指定條件運算式時

若作為條件聯鎖功能使用Wait命令，在指定條件成立之前等待程式。以TMout命令指定超時時，即便經過該指定時間也未成立條件運算式時，會發生錯誤。可以And、Mask、Or或Xor命令等，對1個Wait命令檢查多個條件。請參閱範例。

(3) 指定時間和條件運算式時

指定條件運算式和時間時，則在條件成立或經過指定時間兩者任何一項成立之時，執行下一個命令。可使用Tw函數，確認條件運算式是否成立或是否經過指定時間。

注意

- 併用Wait和超時的情況

  未在Wait命令中指定等待時間時，一旦指定超時後，便可設定等待指定狀態的限制時間。以TMOut命令指定超時。詳細內容請參閱TMOut命令項目。(TMOut命令的預設值為「0」，表示無時間限制。)

- 以Wait等待變數時

- 可等待變數的變數型態為整數型(Byte, Int32, Integer, Long, Short, UByte, UInt32, UShort)。
- 不可使用陣列變數。
- 不可使用本地變數。
- 變數值未滿足條件的時間超過0.01秒時，系統可能無法檢測到變數變化。
- 系統內可使用的變數等待數量有限。1個系統內可使用的變數等待數量最多64個(也包括Till等條件運算式所用的變數等待數量)。若超過最大數量，則在建置專案時將發生錯誤。
- 若以ByRef傳址要執行變數等待的變數，則發生錯誤。
- 若條件運算式右邊的整數運算式中含有變數或函數，則在設定Wait條件時運算該值。可能會變成非預期條件，因此建議在整數運算式中不使用變數或函數。

- 使用PC COM連接埠(1001~1008)時

  不可以Wait命令使用Lof函數。

- 在Wait命令執行期間程式暫停時

  在Wait命令執行期間程式暫停(Pause狀態)時，Wait命令也不會停止。當滿足條件運算式或超過指定的時間，Wait命令將終止。

  如果在Wait命令中指定了時間，則在指定時間到時之前，如果程式在連續運行中回復，以前經過的時間將被重置，并且程式將在指定的時間內等待。

參照

AtHome、Cnv_QueLen、Ctr、ErrorOn、EstopOn、GetRobotInsideBox、GetRobotInsidePlane、In、InW、LatchState、LOF、Mask、MCalComplete、MemIn、MemInW、MemSw Motor, Oport, Out, OutW, PauseOn, SafetyOn, Sw, TeachOn, TMOut, WindowsStatus, Tw, WorkQue_Len, SF_GetStatus

Wait範例

以下範例表示，對於可啟動各動作命令的2項工作，在一方未控制機器人時才能啟用可控制機器人的聯鎖功能。藉此，可依序執行各項工作所指定的動作。併用MemSw的Wait命令用於在記憶體I/O位元1變為適當值之前等待程式，並在確保安全之後重新開始動作。

```
Function main
    Integer I
    MemOff 1
    Xqt !2, task2
    For i = 1 to 100
       Wait MemSw(1) = Off
       Go P(i)
       MemOn 1
    Next I
Fend

Function task2
    Integer i
    For i = 101 to 200
       Wait MemSw(1) = On
       Go P(i)
       MemOff 1
    Next i
Fend

'等待到啟用輸入0為止
Wait Sw(0) = On

'等待60.5秒後繼續執行
Wait 60.5
```

```
'等待到停用輸入0、啟用輸入1為止
Wait Sw(0) = Off And Sw(1) = On

'等待到啟用記憶體位元0或記憶體位元1為止
Wait MemSw(0) = On Or MemSw(1) = On

'等待1秒後啟用輸出1
Wait 1; On 1

'等待輸入連接埠0的低階3位元變為1
Wait In(0) Mask 7 = 1

'等待到全域Integer型變數giCounter的值超過10為止
Wait giCounter > 10

'在全域Long型變數glCheck的值變為30000之前等待10秒鐘
Wait glCheck = 30000, 10
```

3.26.2 WaitNet

等待到確立連接TCP/IP連接埠為止。

格式

WaitNet # 連接埠編號 [, 超時時間]

參數

連接埠編號

以201~216的整數值指定等待連接TCP/IP的連接埠編號。

超時時間

指定最長連接等待時間。可省略。

參照

OpenNet、CloseNet

WaitNet範例

以下是2個控制器的TCP/IP設定範例。

```
Controller #1:
Port: #201
Host Name: 192.168.0.2
TCP/IP Port: 1000

Function tcpip
  OpenNet #201 As Server
  WaitNet #201
  Print #201, "Data from host 1"
Fend

Controller #2:
Port: #201
Host Name: 192.168.0.1
TCP/IP Port: 1000

Function tcpip
  String data$
  OpenNet #201 As Client
  WaitNet #201
  Input #201, data$
  Print "received '", data$, "' from host 1"
Fend
```

3.26.3 WaitPos

即便路徑運動處於啟用狀態，在執行下一陳述式前仍等待機器人減速並停止。

格式

WaitPos

說明

通常，路徑運動為啟用狀態(指定CP On或CP參數)時，在開始減速的同時，動作命令會將控制移至下一個陳述式。

但是，若緊接著插入WaitPos命令，則在完成減速動作後移至下一個動作。

參照

Wait、WaitSig、CP

WaitPos範例

```
Off 1
CP On
Move P1
Move P2
WaitPos  '等待到機器人減速為止
On 1
CP Off
```

3.26.4 WaitSig

用於等待來自其它工作的Signal命令之同步信號。

格式

WaitSig 信號編號 [, 超時時間]

參數

信號編號
　　以整數值(0~63)指定要接收的信號編號。
超時時間
　　以實數值指定最長等待時間。可省略。

說明

等待來自其它工作的信號。執行WaitSig後即變為等待信號狀態，而忽略之前的信號。

參照

Wait、WaitPos、Signal

WaitSig範例

```
Function Main
  Xqt SubTask
  Wait 1
  Signal 1
  .
  .
Fend

Function SubTask
  WaitSig 1
  Print "signal received"
  .
Fend
```

3.26.5 Weight

設定和顯示用於補償PTP動作時的速度、加減速度的參數。

格式

Weight [抓手重量 [, {手臂長度 | S | T }]]

Weight

參數

抓手重量
　　指定對手臂施加的夾具重量。(單位：kg 小數點後2位)
　　可以省略 (但是不能只省略抓手重量)。
手臂長度
　　唯有水平多關節型機器人(包含RS系列)方為有效。指定從第2關節中心到第3關節中心的距離。(單位：mm) 可省略。
S
　　指定對附加軸S關節的負載重量。(單位：kg 小數點後2位)
T
　　指定對附加軸T關節的負載重量。(單位：kg 小數點後2位)

結果

若省略參數，則顯示目前Weight設定值。

如果省略 [手臂長度] 則會設置成輸入的[抓手重量]，並設置預設值 [手臂長度]。

不能只省略 [抓手重量]。

說明

指定用於運算PTP動作最大加減速度的參數。Weight命令用於設定抓手和工件的重量。唯有水平多關節型機器人(包含RS系列)才需指定手臂長度。這是從第2關節中心到第3關節中心的距離。非水平多關節型機器人(包含RS系列)則無效。若機器人設有附加軸，則需使用S、T參數個別設定帶有附加軸的負載重量。

若根據設定值計算的等效搬運重量大於最大可搬運重量，則發生錯誤。

還可以使用「負載、慣性、偏心/偏移測量實用程式」進行設置。

有關詳細資訊，請參閱以下手冊。
「Epson RC+ 使用者指南 - 負載、慣性、偏心/偏移測量實用程式」

常見錯誤

- 超過最大容許負載重量時

  若根據設定值計算的等效負載重量大於最大容許負載重量，則發生錯誤。

- 對機械手手臂的損傷

  請注意，若以遠遠小於實際重量的值設定Weight的抓手重量，則會過度設定加速值及減速值，從而對機械手造成損傷。

注意

- 即使關閉電源，Weight的設置也不會更改

  設置后，Weight的設置將記住在控制器中。即使關閉電源不會更改。

如果未設置任何內容，則該值將保留為上次設置的值。

參照

Accel、Inertia

有關末端夾具的更多資訊，請參閱以下手冊。
「Hand功能」

Weight範例

以下是在命令視窗中使用Weight命令顯示目前設定值的範例。

```
> weight
2.000, 200.000
>
```

以下是以Weight命令設定抓手重量(3 kg)的範例。

```
Weight 3.0
```

以下是以Weight命令設定附加軸S的負載重量(30 kg)的範例。

```
Weight 30.0, S
```

3.26.6 Weight函數

用於傳回以Weight命令設定的抓手重量和手臂長度。

格式

Weight (參數編號)

參數

參數編號

以如下所示的整數值設定參數編號。

- 1：抓手重量
- 2：手臂長度
- 3：附加軸S關節負載重量
- 4：附加軸T關節負載重量

傳回值

用於以實數值傳回參數。

參照

Inertia、Weight

有關末端夾具的更多資訊，請參閱以下手冊。
「Hand功能」

Weight函數範例

```
Print "The current Weight parameters are: ", Weight(1)
```

3.26.7 Where

用於顯示機器人目前位置資料。

格式

Where [本地編號]

參數

本地編號

指定本地坐標系編號。預設為「Local 0」。可省略。

參照

Joint、PList、Pulse

Where範例

顯示的形式因機器人的類型或有無附加軸的情況而異。

以下範例表示的是SCARA機器人無附加軸時的情形。

```
> where
WORLD: X: 350.000 mm Y: 0.000 mm  Z: 0.000 mm  U: 0.000 deg V: 0.000 deg W: 0.000
deg
JOINT: 1:  0.000 deg 2: 0.000 deg 3: 0.000 mm  4: 0.000 deg
PULSE: 1:     0 pls  2:     0 pls 3:    0 pls  4:     0 pls

> local 1, 100,100,0,0

> where 1
WORLD: X: 250.000 mm Y:-100.000 mm  Z: 0.000 mm U: 0.000 deg V: 0.000 deg W: 0.000
deg
JOINT: 1:  0.000 deg 2:   0.000 deg 3: 0.000 mm 4: 0.000 deg
PULSE: 1:     0 pls  2:       0 pls 3:    0 pls 4:     0 pls
```

3.26.8 WindowsStatus函數

用於傳回Windows的啟動狀態。

格式

WindowsStatus

傳回值

用於以整數值傳回目前的Windows啟動狀態。以位元影像傳回Windows啟動狀態，並表示以下狀態。

功能名稱	系統預約	RC+可使用功能	PC可使用功能
位元編號	15 - 2	1	0
可使用功能詳情		Vision Guide(影像擷取卡型) RC+ API 現場匯流排主板	PC檔案 PC RS-232C 資料庫存取 DLL叫用

注意

- 支援的控制器型號

  不支援T/VT系列。

說明

本函數用於將控制器設為「獨立模式」時確認控制器的啟動狀態。若控制器被設為「協作模式」，則在尚未得以使用RC+功能、PC功能之前，無法開始執行程式。

WindowsStatus函數範例

```
Print "The current PC Booting up Status is: ", WindowsStatus
```

3.26.9 WOpen

以寫入模式開啟檔案。

格式

WOpen 檔名 As #檔案編號

.

.

Close #檔案編號

參數

檔案名稱

指定包含路徑的檔案名稱字串。僅指定檔名時，即指目前目錄中的檔案。詳細內容請參閱ChDisk。

檔案編號

以30~63的整數值或運算式來指定。

說明

以指定檔案編號開啟指定的檔案。此陳述式用於開啟指定的檔案並寫入資料。(在已有的檔案中覆寫資料時，請參閱AOpen相關說明。)

沒有指定的檔案時，則新建並寫入檔案。若有指定的檔案，則刪除所有既有資料，並執行新的寫入操作。

在檔案處於開啟狀態下，指定的檔案編號用於識別該檔案。因此，在關閉該檔案之前，不可將相同的檔案編號用於其它檔案。在檔案操作命令(Print#, Write, Seek, Flush, Close)中使用檔案編號。

以Close陳述式關閉檔案，並釋放檔案編號。

請以FreeFile函數取得檔案編號，以避免將同一編號用在多項工作上。

注意

- 可使用網路路徑。
- 寫入檔案時會被緩衝。

  可用Flush陳述式寫入被緩衝的資料。以Close陳述式關閉檔案時，也進行寫入。

參照

AOpen、BOpen、Close、Print#、ROpen、UOpen、FreeFile

WOpen範例

```
Integer fileNum, i, j

fileNum = FreeFile
WOpen "TEST.DAT" As #fileNum
For i = 0 To 100
    Print #fileNum, i
Next i
Close #fileNum

fileNum = FreeFile
ROpen "TEST.DAT" As #fileNum
For i = 0 to 100
    Input #fileNum, j
    Print "data = ", j
```

```
Next i
Close #fileNum
```

3.26.10 WorkQue_Add

在指定工作佇列上新增工作佇列資料(點資料和使用者資料)。

格式

WorkQue_Add 工作佇列編號, 點資料 [, 使用者資料]

參數

工作佇列編號
　　以整數值(1~16)指定工作佇列的編號。
點數據
　　指定工作佇列上增加的點數據。
用戶資料
　　以實數值指定和點數據一起登錄的用戶資料。可省略。若省略，則將0(實數)註冊為使用者資料。

說明

在工作佇列的末尾新增點資料和使用者資料。

但是，若以WorkQue_Sort設定Sort方法，則依該Sort方法進行註冊。

若以WorkQue_Reject設定防止重複註冊的距離，則計算與已完成註冊的點資料之間的距離。若與點資料之間小於該距離，則不將點資料和使用者資料新增於工作佇列中。此時，不會發生錯誤。

工作佇列資料上限值為「1000」。結束使用工作佇列資料時，則以WorkQue_Remove刪除工作佇列資料。

參照

WorkQue_AutoRemove、WorkQue_Len、WorkQue_Reject、WorkQue_Remove、WorkQue_Sort

WorkQueAdd範例

```
Integer x, y
Real u

P0 = XY(300, 300, 300, 90, 0, 180)
P1 = XY(200, 280, 150, 90, 0, 180)
P2 = XY(200, 330, 150, 90, 0, 180)
P3 = XY(-200, 280, 150, 90, 0, 180)

Pallet 1, P1, P2, P3, 10, 10
x = 1
y = 1
u = 5.3
WorkQue_Add 1, Pallet(1, x, y), u
```

3.26.11 WorkQue_AutoRemove

用於在指定工作佇列上設定自動刪除功能。

格式

WorkQue_AutoRemove 工作佇列編號 ,{True | False}

參數

工作佇列編號

以整數值(1~16)指定工作佇列的編號。

True | False

True：啟用自動刪除功能。False：停用自動刪除功能。

說明

在工作佇列上設定自動刪除功能。若啟用自動刪除功能，則在以WorkQue_Get從工作佇列取得點資料時，從工作佇列自動刪除點資料和使用者資料。

若停用自動刪除功能，則不刪除點資料和使用者資料。若要刪除，則使用WorkQue_Remove。

若以WorkQue_UserData取得使用者資料，則不進行自動刪除。

可在每個工作佇列上設定啟用/停用自動刪除功能。

參照

WorkQue_AutoRemove函數、WorkQue_Get

WorkQue_AutoRemove範例

```
WorkQue_AutoRemove 1, True
```

3.26.12 WorkQue_AutoRemove函數

用於傳回在工作佇列上設定的自動刪除功能的狀態。

格式

WorkQue_AutoRemove (工作佇列編號)

參數

工作佇列編號

以整數值(1~16)指定工作佇列的編號。

傳回值

啟用指定工作佇列的自動刪除功能時，傳回「True」；停用時，傳回「False」。

參照

WorkQue_AutoRemove、WorkQue_Get

WorkQue_AutoRemove函數範例

```
Boolean autoremove

autoremove = WorkQue_AutoRemove (1)
```

3.26.13 WorkQue_Get函數

用於從指定的工作佇列傳回點資料。

格式

WorkQue_Get (工作佇列編號 [, 指數])

參數

工作佇列編號

以整數值(1~16)指定工作佇列的編號。

指數

以整數值指定要讀取的佇列資料指數。(開頭指數編號為0)可省略。

傳回值

用於從指定的工作佇列傳回點資料。

說明

WorkQue_Get用於從工作佇列取得點資料。有省略指數時，則傳回佇列資料的開頭資料。有設定指數時，則傳回設定指數的點資料。

若以WorkQue_AutoRemove啟用佇列資料的自動刪除功能，則以WorkQue_Get刪除點資料和使用者資料。

停用時，不刪除點資料和使用者資料。若要刪除，則使用WorkQue_Remove。

參照

WorkQue_AutoRemove、WorkQue_Len、WorkQue_Reject、WorkQue_Remove、WorkQue_Sort

WorkQue_Get函數範例

```
' Jump到佇列的開頭工件以進行追蹤
Jump WorkQue_Get(1)
On gripper
Wait .1
Jump place
Off gripper
Wait .1
WorkQueRemove 1
```

3.26.14 WorkQue_Len函數

用於傳回在指定工作佇列上註冊的啟用工作佇列資料數量。

格式
WorkQue_Len (工作佇列編號)

參數

工作佇列編號
以整數值(1~16)指定工作佇列的編號。

傳回值
以整數值傳回有效的工作佇列資料的註冊數量。

說明
傳回有效的工作佇列資料數的註冊數量。

還可用作Wait命令的引數。

參照
WorkQue_Add、WorkQue_Get、WorkQue_Remove

WorkQue_Len函數範例

```
Do
  Do While WorkQue_Len(1) > 0
    WorkQue_Remove 1, 0
  Loop
  If WorkQue_Len(1) > 0 Then
    Jump WorkQue_Get(1, 0) C0
    On gripper
    Wait .1
    WorkQue_Remove 1, 0
    Jump place
    Off gripper
    Jump idlePos
  EndIf
Loop
```

3.26.15 WorkQue_List

用於顯示指定工作佇列的工作佇列資料一覽(點資料和使用者資料)。

格式

WorkQue_List 工作佇列編號 [, 顯示數]

參數

工作佇列編號

以整數值(1~16)指定工作佇列的編號。

顯示數

以整數值指定要顯示的資料數量。可省略。若省略，則顯示所有佇列資料。

注意

此命令僅可在命令視窗中執行。

參照

WorkQue_Add、WorkQue_Get、WorkQue_Remove

WorkQue_List範例

命令視窗中的操作範例

```
> WorkQue_List 1
Queue 0    = XY(     1.000,    1.000,    0.000,    0.000 )  /R /0 (   0.000)
Queue 1    = XY(     3.000,    1.000,    0.000,    0.000 )  /R /0 (   2.000)
Queue 2    = XY(     4.000,    1.000,    0.000,    0.000 )  /R /0 (   3.000)
Queue 3    = XY(     5.000,    1.000,    0.000,    0.000 )  /R /0 (   4.000)
Queue 4    = XY(     6.000,    1.000,    0.000,    0.000 )  /R /0 (   5.000)
```

3.26.16 WorkQue_Reject

用於在指定工作佇列上設定和顯示防止重複註冊點資料的最小距離。

格式

WorkQue_Reject 工作佇列編號 [, 防止重複註冊的距離]

參數

工作佇列編號

以整數值(1~16)指定工作佇列的編號。

防止重複註冊的距離

以實數值(單位：mm)指定用於防止重複註冊的距離最小值。若指定為負值，則設定0 mm。若在命令視窗中執行，則可省略。若有省略，則顯示用於防止重複註冊的目前距離。

說明

設定用於防止重複註冊點資料的最小距離。即便以WorkQue_Add註冊最小距離以下的點資料，也不註冊於工作佇列上。WorkQue_Reject是用於防止重複註冊的系統篩選器。預設值為0 mm。

在以WorkQue_Add新增工作佇列資料(點資料和使用者資料)之前，需執行WorkQue_Reject。

可對每個工作佇列設定防止重複註冊的距離。

參照

WorkQue_Add、WorkQue_Reject函數

WorkQue_Reject範例

```
WorkQue_Reject 1, 2.5
```

3.26.17 WorkQue_Reject函數

用於傳回在指定工作佇列上設定的防止重複註冊之距離。

格式

WorkQue_Reject (工作佇列編號)

參數

工作佇列編號

以整數值(1~16)指定工作佇列的編號。

傳回值

用於傳回實數值(單位：mm)。

參照

WorkQue_Add、WorkQue_Reject

WorkQue_Reject函數範例

```
Real rejectDist

RejectDist = WorkQue_Reject(1)
```

3.26.18 WorkQue_Remove

從指定工作佇列刪除工作佇列資料(點資料和使用者資料)。

格式

WorkQue_Remove 工作佇列編號 [, 指數 | All]

參數

工作佇列編號

以整數值(1~16)指定傳送帶編號。

指數

以整數值指定要刪除的工作佇列資料之指數。(開頭指數編號為0 。)可省略。若要從工作佇列刪除所有工作佇列資料，則以All進行指定。

說明

從工作佇列資料刪除1筆以上的工作佇列資料(點資料和使用者資料)。結束使用工作佇列資料時，則刪除佇列資料。

參照

WorkQue_Add

WorkQue_Remove範例

```
Jump WorkQue_Get(1)
On gripper
Wait .1
Jump place
Off gripper
Wait .1

' 從工作佇列刪除資料
WorkQue_Remove 1
```

3.26.19 WorkQue_Sort

用於在指定工作佇列上設定和顯示Sort方法。

格式

WorkQue_Sort 工作佇列編號 [, Sort方法]

參數

工作佇列編號

以整數值(1~16)指定工作佇列的編號。

Sort方法

以整數值(0~6)或如下所示的常數指定Sort方法。若在命令視窗中執行，則可省略。若省略，則顯示目前Sort方法。

常數	值	內容
QUE_SORT_NONE	0	無排序(按註冊到工作佇列的順序)
QUE_SORT_POS_X	1	X坐標升序
QUE_SORT_INV_X	2	X坐標降序
QUE_SORT_POS_Y	3	Y坐標升序
QUE_SORT_INV_Y	4	Y坐標降序
QUE_SORT_POS_USER	5	使用者資料(實數)升序
QUE_SORT_INV_USER	6	使用者資料(實數)降序

說明

在工作佇列上設定Sort方法。以WorkQue_Add新增點資料和使用者資料時，則依設定的Sort方法在工作佇列上進行註冊。

以WorkQue_UserData重新設定使用者資料時，則依設定的Sort方法改變工作佇列的排列順序。

在以WorkQue_Add新增工作佇列資料(點資料和使用者資料)之前，需執行WorkQue_Sort。

在以WorkQue_UserData重新設定使用者資料之前，需執行WorkQue_Sort。

可對每個工作佇列設定Sort方法。

參照

WorkQue_Add, WorkQue_UserData

WorkQue_Sort範例

```
WorkQue_Sort 1, QUE_SORT_POS_X
```

3.26.20 WorkQue_Sort函數

用於傳回在指定的工作佇列上設定的Sort方法。

格式

WorkQue_Sort (工作佇列編號)

參數

工作佇列編號
以整數值(1~16)指定工作佇列的編號。

傳回值

用於以整數值傳回在工作佇列上設定的Sort方法。

- 0 = 無排序(按註冊到工作佇列的順序)
- 1 = X坐標升序
- 2 = X坐標降序
- 3 = Y坐標升序
- 4 = Y坐標降序
- 5 = 使用者資料(實數)升序
- 6 = 使用者資料(實數)降序

參照

WorkQue_Add、WorkQue_Sort、WorkQue_UserData

WorkQue_Sort函數範例

```
Integer quesort

quesort = WorkQue_Sort(1)
```

3.26.21 WorkQue_UserData

用於重新設定和顯示在指定工作佇列上註冊的使用者資料(實數)。

格式

WorkQue_UserData 工作佇列編號 [, 指數] [, 使用者資料]

參數

工作佇列編號
　　以整數值(1~16)指定工作佇列的編號。
指數
　　工件以整數值指定佇列資料的指數。(開頭指數編號為0 。)若在命令視窗中執行，則可省略。
用戶資料
　　以實數值指定要重新設定的用戶資料。若在命令視窗中執行，則可省略。若省略，則顯示目前使用者資料(實數)。

說明

重新設定和顯示在工作佇列上註冊的使用者資料。

若以WorkQue_Sort設定以下Sort方法，則依該設定的Sort方法改變工作佇列資料的排列順序。

- QUE_SORT_POS_USER：使用者資料(實數)升序
- QUE_SORT_INV_USER：使用者資料(實數)降序

參照

WorkQue_UserData函數

WorkQue_UserData範例

```
WorkQue_UserData 1, 1, angle
```

3.26.22 WorkQue_UserData函數

用於傳回在指定工作佇列上註冊的使用者資料(實數)。

格式

WorkQue_UserData (工作佇列編號 [, 指數])

參數

工作佇列編號
　　以整數值(1~16)指定工作佇列的編號。
指數
　　工件以整數值指定佇列資料的指數。(開頭指數編號為0 。)可省略。

傳回值

用於傳回實數值。

參照

WorkQue_UserData

WorkQue_UserData函數範例

```
' 從佇列刪除
angle = WorkQue_UserData(1)  '預設指數變成「0」
Jump WorkQue_Get(1) :U(angle)
WorkQue_Remove 1
```

3.26.23 Wrist

設定點的腕部姿態。

格式

(1) Wrist 點指定 [, Flip | NoFlip]

(2) Wrist

參數

點指定

　　用於指定P編號、P(運算式)、點標籤之一。

若省略Flip | NoFlip

　　指定腕部姿態。

結果

若2個參數皆省略，則顯示機器人目前位置的腕部姿態。

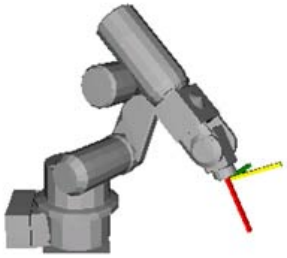
若省略Flip | NoFlip，則顯示指定點的腕部姿態。

參照

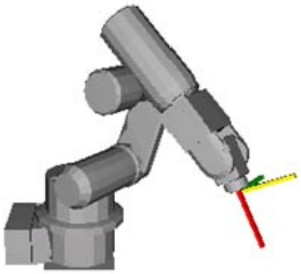
Elbow、Hand、J4Flag、J6Flag、Wrist函數

Wrist範例

```
Wrist P0, Flip
Wrist P(mypoint), NoFlip

P1 = 320.000, 400.000, 350.000, 140.000, 0.000, 150.000
```



```
Wrist P1, NoFlip
Go P1
```



```
Wrist P1, Flip
Go P1
```

3.26.24 Wrist函數

用於傳回點的腕部姿態。

格式

Wrist [(點指定)]

參數

點指定

指定P編號、P(運算式)、點標籤、點運算式之一。可省略。若省略點指定，則傳回目前機器人位置的腕部姿態。

傳回值

- 1: NoFlip (/NF)
- 2: Flip (/F)

參照

Elbow、Hand、J4Flag、J6Flag、Wrist

Wrist函數範例

```
Print Wrist(pick)
Print Wrist(P1)
Print Wrist
Print Wrist(P1 + P2)
```

3.26.25 Write

將字串寫入檔案或通訊連接埠中。不附加行尾結束字元。

格式

Write # 連接埠編號, 字串

參數

連接埠編號

表示檔案或通訊連接埠的ID編號。檔案編號是以ROpen、WOpen、AOpen等陳述式指定的編號。通訊連接埠編號是以OpenCom(RS-232C)或OpenNet(TCP/IP)陳述式指定的編號。

字串

指定寫入的字串。

說明

Write命令不同於Print命令，不附加行尾結束字元。

注意

- 支援的控制器型號

  如果在T/VT系列中指定RS-232C埠時會發生錯誤。

- 寫入檔案時會被緩衝

  可用Flush陳述式寫入被緩衝的資料。以Close陳述式關閉檔案時，也進行寫入。

參照

Print、Read、WriteBin

Write範例

```
OpenCom #1
For i = 1 to 10
    Write #1, data$(i)
Next i
CloseCom #1
```

3.26.26 WriteBin

將二進位資料寫出到檔案或通訊連接埠中。

格式

WriteBin #接埠編號, 寫入資料

WriteBin #連接埠編號, 陣列變數名稱 (), 位元組數

參數

連接埠編號

表示檔案或通訊連接埠的ID編號。檔案編號是以BOpen等陳述式指定的編號。通訊連接埠編號是以OpenCom(RS-232C)或OpenNet(TCP/IP)陳述式指定的編號。

寫入資料

以整數或運算式指定寫入的資料。

陣列變數名稱()

指定存放寫出的資料位元組之Byte型變數、整數變數或Long型變數的名稱。可指定的是一維的陣列變數。

位元組數

指定要寫出的位元組數。必須小於陣列的最大元素編號且小於256 Byte。若以通訊連接埠(TCP/IP)為適用對象時，必須小於陣列的最大元素編號且小於1024 Byte。

注意

- 支援的控制器型號

  如果在T/VT系列中指定RS-232C埠時會發生錯誤。

參照

ReadBin、Write

WriteBin範例

```
Integer i, data(100)

OpenCom #1
For i = 0 To 100
  WriteBin #1, i
Next i
WriteBin #1, data(), 100
CloseCom #1
```

3.27 X

3.27.1 Xor運算子

以位元為單位對2個值進行Xor運算(互斥OR運算)。

格式

result = 值1 Xor 值2

參數

值1、值2
　　指定數值或變數名稱。
result
　　傳回整數。

結果

以位元為單位傳回已進行Xor運算的result。

說明

Xor運算子用於以位元單位進行Xor運算。result位元是以位元為單位對2個值進行Xor的結果。

值1的位元	值2的位元	result
0	0	0
0	1	1
1	0	1
1	1	0

參照

And、LShift、Not、Or、Rshift

Xor運算子範例

```
>print 2 Xor 6
 4
>
```

3.27.2 Xqt

用於執行以函式名稱指定的程式並生成工作。

格式

Xqt [工作編號,] 函式名稱 [(引數清單)] [, Normal | NoPause | NoEmgAbort]

參數

工作編號
　以1~32的整數指定要執行的任務編號。可省略。若為背景工作，則指定65~80的整數。
函數名稱
　指定所要執行函數名稱。
引數清單
　指定呼叫時傳遞給函數的引數清單。若有數個引數，則請用逗號分隔。可省略。
工作類型
　可省略。通常會省略。若為背景工作，指定工作類型沒有意義。
Normal
　生成一般任務。
NoPause
　發生Pause陳述式或Pause輸入信號時、以及在啟動安全門的狀態下生成不暫停的任務時，指定本命令。
NoEmgAbort
　若要生成在緊急停止時以及發生錯誤時繼續處理的任務，則指定本命令。

說明

Xqt用於啟動指定函式並立即返回。

平常不需要工作編號參數。若省略工作編號，SPEL＋則會自動在函式上附加工作編號，因此使用者無需管理工作編號。

注意

- 工作類型

  可透過在工作類型中指定NoPause或NoEmgAbort，生成用於監視整體控制器的工作。

  但是，強烈建議在充分理解SPEL+的工作運作和特殊工作的限制事項之前提下使用這些特殊工作。

  有關特殊工作的詳細資訊，請參閱以下手冊。
  「Epson RC+ 使用者指南 - 特殊工作」

- 背景工作

  若在背景工作執行Xqt，生成的工作也會變成背景工作。

  若要在背景工作執行Main函式，請使用StartMain命令。

  有關背景工作的詳細資訊，請參閱以下手冊。
  「Epson RC+ 使用者指南 - 特殊工作」

無法在NoEmgAbort工作和背景工作中執行的命令

不可在NoEmgAbort工作和背景工作中執行以下命令。

A	Accel
	AccelR
	AccelS

	AIO_TrackingStart
	AIO_TrackingEnd
	Arc
	Arc3
	Arch
	Arm
	ArmCalib
	ArmCalibCLR
	ArmCalibSET
	ArmClr
	ArmSet
	AutoLJM
	AutoOrientationFlag
	AvoidSingularity
B	Base
	BGo
	BMove
	Box
	BoxClr
	Brake
C	Calib
	Cnv_AbortTrack
	Cnv_Accel
	Cnv_AccelLim
	Cnv_Adjust
	Cnv_AdjustClear
	Cnv_AdjustGet
	Cnv_AdjustSet
	Cnv_DownStream
	Cnv_Fine
	Cnv_Mode
	Cnv_OffsetAngle

	Cnv_QueAdd
	Cnv_QueMove
	Cnv_QueReject
	Cnv_QueRemove
	Cnv_QueUserData
	Cnv_Trigger
	Cnv_UpStream
	CollisionDetect
	CP
	CP_Offset
	Curve
	CVMove
E	ECP
	ECPClr
	ECPSet
F	Find
	Fine
	FineDist
	Force_Calibrate
	Force_ClearTrigger
	Force_Sensor
	Force_SetTrigger
G	Go
H	Hand_On
	Hand_Off
	Home
	HomeClr
	HomeSet
	Hordr
I	Inertia
J	JTran
	Jump

	Jump3
	Jump3CP
	JRange
L	LatchEnable
	LimitTorque
	LimZ
	LimZMargin
	Local
	LocalClr
M	MCal
	MCordr
	Motor
	Move
O	OLAccel
P	Pass
	PerformMode
	Pg_LSpeed
	Pg_Scan
	Plane
	PlaneClr
	Power
	PTPBoost
	Pulse
Q	QP
	QPDecelR
	QPDecelS
R	Range
	Reset *1
	Restart *2
S	Sense
	SetLatch
	SFree

	SingularityAngle
	SingularityDist
	SingularitySpeed
	SLock
	SoftCP
	Speed
	SpeedFactor
	SpeedR
	SpeedS
	SyncRobots
T	TC
	TGo
	Till
	TLSet
	TLClr
	TMove
	Tool
	Trap
V	VCal
	VcalPoints
	VCls
	VCreateCalibration
	VCreateObject
	VCreateSequence
	VDefArm
	VDefGetMotionRange
	VDefLocal
	VDefSetMotionRange
	VDefTool
	VDeleteCalibration
	VDeleteObject
	VDeleteSeuence

	VEditWindow
	VGet
	VGoCenter
	VLoad
	VLoadModel
	VRun
	VSave
	VSaveImage
	VSaveModel
	VSet
	VShowModel
	VStatsShow
	VStatsReset
	VStatsResetAll
	VStatsSave
	VSD
	VTeach
	VTrain
W	WaitPos
	Weight
	WorkQue_Add
	WorkQue_Reject
	WorkQue_Remove
	WorkQue_Sort
	WorkQue_UserData
X	Xqt *3
	XYLim

- *1 可執行Reset Error
- *2 可在Trap Error的處理工作中執行
- *3 可在背景工作中執行

使用時，請勿在Loop陳述式中頻繁重複XQT命令 使用時，請勿在Do…Loop等Loop陳述式中頻繁重複XQT命令。否則，控制器有可能進入意外停機狀態。若要這樣使用，請輸入Wait命令(Wait 0.1)。

參照

Function/Fend、Halt、Resume、Quit、Startmain、Trap

Xqt範例

```
Function main
    Xqt flash           '啟動工作flash
    Xqt Cycle(5)        '啟動工作Cycle

    Do
        Wait 3          '執行工作flash3秒鐘
        Halt flash      '暫停工作

        Wait 3
        Resume flash    '重新開始工作
    Loop
Fend

Function Cycle(count As Integer)
    Integer i

    For i = 1 To count
        Jump pick
        On vac
        Wait .2
        Jump place
        Off vac
        Wait .2
    Next i
Fend

Function flash
    Do
        On 1
        Wait 0.2
        Off 1
        Wait 0.2
    Loop
Fend
```

3.27.3 XY函數

用於將指定的坐標值作為點資料傳回。

格式

XY (x坐標元素, y坐標元素, z坐標元素, u坐標元素 [, v坐標元素, w坐標元素])

參數

x坐標元素
　　以實數值指定x坐標。
y坐標元素
　　以實數值指定y坐標。
z坐標元素
　　以實數值指定z坐標。
u坐標元素
　　以實數值指定u坐標。
v坐標元素
　　以實數值指定v坐標。是用於垂直6軸型機器人(包含N系列)的參數。可省略。
w坐標元素
　　以實數值指定w坐標。是用於垂直6軸型機器人(包含N系列)的參數。可省略。

傳回值

用於將指定的坐標值作為點資料傳回。

說明

未使用附加軸(S軸、T軸)時，不需特別注意。

可透過Go XY (60,30,-50,45) 等使用方法，向指定的坐標值運作機器人。

使用附加軸(S軸、T軸)時，需加以注意。

XY函數只傳回除附加軸之外的機器人的點資料。

若採用Go XY (60,30,-50,45) 等使用方法，機器人則會朝指定的坐標值進行運作，但附加軸不動作。若要同時運作附加軸的情況，請進行如Go XY (60,30,-50,45) : ST (10,20)這樣的指定。

有關詳細資訊，請參閱以下手冊。「Epson RC+ 使用者指南 - 附加軸」

參照

JA、P# = 點指定、ST函數

XY函數範例

```
P10 = XY(60, 30, -50, 45) + P20
```

3.27.4 XYLim

用於設定和顯示容許動作區域。

格式

XYLim X軸下限位置, X軸上限位置, Y軸下限位置, Y軸上限位置 [, Z軸下限位置] [, Z軸上限位置]

XYLim

參數

X軸下限位置
　以數值或運算式指定可運作機械手的下限位置X坐標值(實數)。
X軸上限位置
　以數值或運算式指定可運作機械手的上限位置X坐標值(實數)。
Y軸下限位置
　以數值或運算式指定可運作機械手的下限位置Y坐標值(實數)。
Y軸上限位置
　以數值或運算式指定可運作機械手的上限位置Y坐標值(實數)。
Z軸下限位置
　以數值或運算式指定可運作機械手的下限位置Z坐標值(實數)。可省略。
Z軸上限位置
　以數值或運算式指定可運作機械手的上限位置Z坐標值(實數)。可省略。

結果

若省略參數，則顯示目前XYLim值。

說明

XYLim用於設定容許動作區域。不少機器人系統可由使用者來設定關節的動作容許範圍，但在SPEL+語言方面，除了關節的動作範圍，還可設定動作容許範圍。這樣的話，可依機器人的應用程式限制運作範圍。

使用XYLim值設置的動作範圍，會應用在以XYLimMode命令設置的監視方法。有關監視方法的詳細資訊，請參閱XYLimMode語句。

注意

- 停用容許動作區域的設定

  大部分的用途無須設定容許動作區域。因此，可輕鬆地停用此設定。若要停用此設定，請將各參數值(X軸下限位置、X軸上限位置、Y軸下限位置、Y軸上限位置)設為「0」。例: XYLim 0, 0, 0, 0.

- 容許動作限制值的預設值

  XYLim的預設值是「0, 0, 0, 0」。(會停用容許動作區域的設定。)

提示

- 透過點&點擊設定XYLim

  Epson RC+可透過[工具]選單－[機器人管理器]的[XYL限定]面板設定XYLim值。

參照

Range, XYLimMode

XYLim範例

如下所述為在命令視窗上設定XYLim值並顯示目前值的簡易操作範例。

```
> XYlim -200, 300, 0, 500

> XYLim
-200.000, 300.000, 0.000, 500.000
```

3.27.5 XYLim函數

用於傳回已設定的XY平面容許動作區域。

格式

XYLim (參照資料)

參數

參照資料

以整數值指定作為傳回值而傳回的容許區域。

- 1：下限值
- 2：上限值

傳回值

若作為參照資料指定1，則將以XYLim設定值指定的X軸下限位置視為點資料X，將Y軸下限位置視為點資料Y，將Z軸下限位置視為點資料Z，以進行傳回。

若作為參照資料指定2，則將以XYLim設定值指定的X軸上限位置視為點資料X，將Y軸上限位置視為點資料Y，將Z軸上限位置視為點資料Z，以進行傳回。

參照

XYLim

XYLim函數範例

```
P1 = XYLim(1)
P2 = XYLim(2)
```

3.27.6 XYLimClr

清除設定的XYLim。

格式

XYLimClr

參照

XYLim、XYLimDef

XYLimClr函數範例

以下是使用XYLimClr函數的程式。

```
Function ClearXYLim

   If XYLimDef = True Then
      XYLimClr
   EndIf
Fend
```

3.27.7 XYLimDef函數

用於傳回是否設定XYLim。

格式

XYLimDef

傳回值

若設定XYLim，則傳回「True」；若未設定，則傳回「False」。

參照

XYLim、XYLimClr

XYLimDef函數範例

以下是使用XYLimDef函數的程式。

```
Function ClearXYLim

   If XYLimDef = True Then
      XYLimClr
   EndIf
Fend
```

3.27.8 XYLimMode

設定並顯示XYLim的監視方法。

格式

(1) XYLimMode 監視方法

(2) XYLimMode

參數

監視方法

表示XYLim中使用的監視方法的整數運算式

常數	值	內容
XYLIM_STANDARD	0	將XYLim應用於操作命令的目標座標。(不會影響Pulse 。)
XYLIM_STRICT	1	除了XYLIM_STANDARD的監視方法外，XYLim還適用於運動軌跡和脈衝動作。

結果

如果省略參數，則顯示當前設置的XYLim的監視方法。

說明

XYLimMode設定指定機器人的XYLim的監視方法。

指定XYLIM_STANDARD時，XYLim中設定的動作範圍只對動作命令的目標坐標有效。不適用於從起點位置到目標坐標的動作路徑。因此，在動作過程中機械臂可能會超出XYLim設定的範圍。在此模式下，XYLim不適用於脈衝動作。

指定XYLIM_STRICT時，XYLim中設定的動作範圍，對動作命令的目標坐標與起點位置到目標坐標的動作軌跡有效。因此，當機械臂的動作範圍超過XYLim設定的範圍時，則會出現錯誤。在此模式下，XYLim適用於脈衝動作。但是，如果起點位置在XYLim範圍外，目標坐標在XYLim內，則不會報錯。

為了防止機器人與周邊裝置的干涉，推薦使用XYLIM_STRICT。

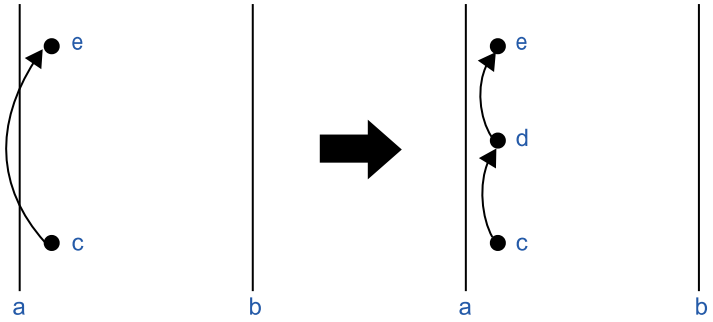
可以在Epson RC+的控制器首選項中，更改控制器啟動時候監視方法的預設值。XYLimMode命令設置的值僅在控制器重新啟動之前有效。重新啟動控制器時，XYLim的監視方法將重置為控制器首選項中指定的監視方法。

> ⚠ 注意
>
> 設置XYLIM_STANDARD時，機械臂可能會移動到XYLim設定的範圍外。此時設置XYLim需要考慮與周圍障礙物留出一定的餘量，或者以低速模式檢查XYLim邊界附近的動作軌跡，確認機器人和設備不會有干擾。。

常見錯誤

- 在XYLim邊界附近執行PTP動作

  執行Go命令等PTP動作時，會如下圖左邊所示，從起點位置按照箭頭的軌跡移動到目標坐標。在XYLIM_STRICT中，雖然目標坐標在設定的範圍內，因為運動軌跡超過了範圍會出現錯誤。這種情況下，請在起點位置和目標坐標之間，添加中間坐標，如下圖右邊所示。避免運動軌跡超過XYLim範圍。



符號	說明
a	XYLim下限位置
b	XYLim上限位置
c	起點位置
d	中間坐標
e	目標坐標

- 使用過去使用過的專案

  如果使用的專案是在控制器的韌體版本低於Ver 7.5.2.0或7.5.52.0的情況下，在XYLim監視方法為XYLIM_STRICT的控制器上執行專案，可能會因為動作軌跡超過設定範圍而出現錯誤。這種情況下，請修改程式，例如添加中間坐標，來使動作軌跡不超過XYLim。

參照

XYLim

XYLimMode命令範例

以下是使用XYLimMode的程式。用XYLIM_STANDARD從起點位置移動到P1，用XYLIM_STRICT從P1移動到P2。

```
Function XYLimMode_sample
Motor On
XYLimMode XYLIM_STANDARD
Go P1  'XYLim僅適用於目標坐標
XYLimMode XYLIM_STRICT
Go P2  'XYLim適用於目標坐標與動作軌跡
Fend
```

以下是在命令視窗使用XYLimMode的程式。顯示當前XYLim的監視方法。

```
> XYLimMode
1
```

3.27.9 XYLimMode函數

獲取設定的XYLim監視方法。

格式

XYLimMode

傳回值

傳回設定的XYLim監視方法。

- 0 = 將XYLim應用於動作命令的目標坐標。(不適用於脈衝運動)
- 1 = 將XYLim應用於動作命令的目標左邊和動作軌跡。(適用於脈衝運動)

參照

XYLimMode

XYLimMode函數範例

以下是使用XYLimMode的程式。可以用變量獲取XYLim的監視方法，然後顯示出來。

```
Function XYLimMode_sample
Integer iVar

iVar = XYLimMode
Print iVar

Fend
```

4. Appendix A：SPEL+命令 使用條件一覽

4.1 Appendix A：SPEL+命令 使用條件一覽

- 命令視窗：可在命令視窗中使用。
- 程式：可在SPEL+程式中用作陳述式。
- 函數：可用作函數。

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
A	AbortMotion	○	○	○	-
	Abs	-	-	○	○
	Accel	○	○	○	○
	AccelMax	-	-	○	○
	AccelR	○	○	○	○
	AccelS	○	○	○	○
	Acos	-	-	○	○
	Agl	-	-	○	○
	AglToPls	-	-	○	○
	AIO_In	○	○	-	○
	AIO_InW	○	○	-	○
	AIO_Out	○	○	○	○
	AIO_OutW	○	○	○	○
	AIO_Set	○	○	○	○
	AIO_TrackingSet	○	○	○	-
	AIO_TrackingStart	-	-	○	-
	AIO_TrackingEnd	-	-	○	-
	AIO_TrackingOn	○	○	○	○
	Align	-	-	○	○
	AlignECP	-	-	○	○
	And	-	-	○	-
	AOpen	○	○	○	-
	Arc	○	○	○	-
	Arc3	○	○	○	-
	Arch	○	○	○	○
	AreaCorrection	○	○	○	○

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	AreaCorrectionClr	○	○	○	-
	AreaCorrectionDef	○	○	○	○
	AreaCorrectionInv	○	○	○	○
	AreaCorrectionOffset	○	○	○	○
	AreaCorrectionSet	○	○	○	-
	Arm	○	○	○	○
	ArmClr	○	○	○	-
	ArmDef	-	-	○	○
	ArmSet	○	○	○	○
	Asc	-	-	○	○
	ArmCalib	○	○	○	○
	ArmCalibClr	○	○	○	-
	ArmCalibDef	-	-	○	○
	ArmCalibSet	○	○	○	○
	Asin	-	-	○	○
	Atan	-	-	○	○
	Atan2	-	-	○	○
	ATCLR	○	○	○	-
	AtHome	-	-	○	○
	ATRQ	○	○	○	○
	AutoLJM	○	○	○	○
	AvoidSingularity	○	○	○	○
B	Base	○	○	○	○
	BClr	-	-	○	○
	BClr64	-	-	○	○
	BGo	○	○	○	-
	BMove	○	○	○	-
	Boolean	-	-	○	-
	BOpen	○	○	○	-
	Box	○	○	○	○

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	BoxClr	○	○	○	-
	BoxDef	-	-	○	○
	Brake	○	○	○(僅限函數)	○
	BSet	-	-	○	○
	BSet64	-	-	○	○
	BTst	-	-	○	○
	BTst64	-	-	○	○
	Byte	-	-	○	-
C	Calib	○	○	○	-
	Call	-	-	○	-
	CalPls	○	○	○	○
	ChDir	○	○	○	-
	ChDisk	○	○	○	-
	ChDrive	○	○	○	-
	ChkCom	-	-	○	○
	ChkNet	-	-	○	○
	Chr$	-	-	○	○
	ClearHistory	○	-	-	-
	ClearPoints	○	○	○	-
	Close	○	○	○	-
	CloseCom	○	○	○	-
	CloseDB	○	○	○	-
	CloseNet	○	○	○	-
	Cls	○	○	○	-
	Cnv_AbortTrack	○	○	○	-
	Cnv_Accel	○	○	○	○
	Cnv_AccelLim	○	○	○	○
	Cnv_Adjust	○	○	○	-
	Cnv_AdjustClear	○	○	○	-
	Cnv_AdjustGet	○	○	○	○

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	Cnv_AdjustSet	○	○	○	-
	Cnv_Downstream	○	○	○	○
	Cnv_Fine	○	○	○	○
	Cnv_LPulse	-	-	○	○
	Cnv_Mode	○	○	○	○
	Cnv_Name$	-	-	○	○
	Cnv_Number	-	-	○	○
	Cnv_OffsetAngle	○	○	○	○
	Cnv_Point	-	-	○	○
	Cnv_PosErr	-	-	○	○
	Cnv_PosErrOffset	-	-	○	○
	Cnv_Pulse	-	-	○	○
	Cnv_QueAdd	○	○	○	-
	Cnv_QueGet	-	-	○	○
	Cnv_QueLen	-	-	○	○
	Cnv_QueList	○	○	-	-
	Cnv_QueMove	○	○	○	-
	Cnv_QueReject	○	○	○	○
	Cnv_QueRemove	○	○	○	-
	Cnv_QueUserData	○	○	○	○
	Cnv_RobotConveyor	-	-	○	○
	Cnv_Speed	-	-	○	○
	Cnv_Trigger	○	○	○	-
	Cnv_Upstream	○	○	○	○
	CollisionDetect	○	○	○	○
	Cont	○	-	○	-
	Copy	○	○	○	-
	Cos	-	-	○	○
	CP	○	○	○	○
	Ctr	-	-	○	○

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	CTReset	○	○	○	-
	CtrlDev	-	-	○	○
	CtrlInfo	-	-	○	○
	CurDir$	-	-	○	○
	CurDisk$	-	-	○	○
	CurDrive$	-	-	○	○
	CurPos	-	-	○	○
	Curve	○	○	○	-
	CVMove	○	○	○	-
	CP_Offset	○	○	○	○
	CR	○	○	○	○
	CS	○	○	○	○
	CT	○	○	○	○
	CU	○	○	○	○
	CV	○	○	○	○
	CW	○	○	○	○
	CX	○	○	○	○
	CY	○	○	○	○
	CZ	○	○	○	○
D	Date	○	○	○	-
	Date$	-	-	○	○
	Declare	-	-	○	-
	DegToRad	-	-	○	○
	Del	○	○	○	-
	DeleteDB	○	○	○	-
	DiffPoint	○	○	○	○
	DispDev	○	○	○	○
	Dist	-	-	○	○
	Do...Loop	-	-	○	-
	Double	-	-	○	-

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
E	ECP	○	○	○	○
	ECPClr	○	○	○	-
	ECPDef	-	-	○	○
	ECPSet	○	○	○	○
	ElapsedTime	-	-	○	○
	Elbow	○	○	○	○
	EncTemper	○	○	○	○
	Eof	-	-	○	○
	Era	-	-	○	○
	EResume	○	○	○	-
	Erf$	-	-	○	○
	Erl	-	-	○	○
	Err	-	-	○	○
	Errb	○	○	○	○
	ErrMsg$	-	-	○	○
	Error	○	○	○	-
	ErrorOn	-	-	○	○
	Ert	-	-	○	○
	EStopOn	-	-	○	○
	Eval	-	-	○	○
	Exit	-	-	○	-
	ExportPoints	○	○	○	-
F	FbusIO_GetBusStatus	-	-	○	○
	FbusIO_GetDeviceStatus	-	-	○	○
	FbusIO_SendMsg	○	○	○	-
	FileDataTime$	-	-	○	○
	FileExists	-	-	○	○
	FileLen	-	-	○	○
	Find	○	○	○	-
	FindPos	-	-	○	○

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	Fine	○	○	○	○
	FineDist	○	○	○	○
	FineStatus	○	○	○	○
	Fix	-	-	○	○
	Flush	○	○	○	-
	FmtStr	○	○	○	-
	FmtStr$	-	-	○	○
	FolderExists	-	-	○	○
	For...Next	-	-	○	-
	FreeFile	-	-	○	○
	Function...Fend	-	-	○	-
G	GClose	○	○	○	-
	GetRobotInsideBox	-	-	○	○
	GetRobotInsidePlane	-	-	○	○
	GGet	○	○	○	-
	Global	-	-	○	-
	Go	○	○	○	-
	Gosub...Return	-	-	○	-
	Goto	-	-	○	-
	GSet	○	○	○	-
	GShow	-	-	○	-
	GShowDialog	-	-	○	○
H	Halt	-	-	○	-
	Hand	○	○	○	○
	HealthCalcPeriod	○	○	○	○
	HealthCtrlAlarmOn	○	○	○	○
	HealthCtrlInfo	○	○	○	○
	HealthCtrlRateOffset	○	○	○	-
	HealthCtrlReset	○	○	○	-
	HealthCtrlWarningEnable	○	○	○	○

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	HealthRateCtrlInfo	○	○	○	○
	HealthRateRBInfo	○	○	○	○
	HealthRBAlarmOn	○	○	○	○
	HealthRBAnalysis	○	○	○	○
	HealthRBDistance	○	○	○	○
	HealthRBInfo	○	○	○	○
	HealthRBRateOffset	○	○	○	-
	HealthRBReset	○	○	○	-
	HealthRBSpeed	○	○	○	○
	HealthRBStart	○	○	○	-
	HealthRBStop	○	○	○	-
	HealthRBTRQ	○	○	○	○
	HealthRBWarningEnable	○	○	○	○
	Here	○	○	○	○
	Hex$	-	-	○	○
	History	○	-	-	-
	Hofs	○	○	○	○
	HofsJointAccuracy	○	○	○	-
	Home	○	○	○	-
	HomeClr	○	○	○	-
	HomeDef	-	-	○	○
	HomeSet	○	○	○	○
	Hordr	○	○	○	○
	Hour	○	○	○	○
I	If...Then..Else…EndIf	-	-	○	-
	ImportPoints	○	○	○	-
	In	-	-	○	○
	InBCD	-	-	○	○
	Inertia	○	○	○	○
	InPos	-	-	○	○

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	Input	○	○	○	-
	Input#	○	○	○	-
	InputBox	○	○	○	-
	InReal	-	-	○	○
	InsideBox	-	-	○	○
	InsidePlane	-	-	○	○
	InStr	-	-	○	○
	Int	-	-	○	○
	Int32	-	-	○	-
	Integer	-	-	○	-
	InW	-	-	○	○
	IODef	-	-	○	○
	IOLabel$	-	-	○	○
	IONumber	-	-	○	○
J	J1Angle	○	○	○	○
	J4Angle	○	○	○	○
	J1Flag	○	○	○	○
	J2Flag	○	○	○	○
	J4Flag	○	○	○	○
	J6Flag	○	○	○	○
	JA	-	-	○	○
	Joint	○	○	○	-
	JointAccuracy	○	○	○	○
	JRange	○	○	○	○
	JS	-	-	○	○
	JT	-	-	○	○
	JTran	○	○	○	-
	Jump	○	○	○	-
	Jump3	○	○	○	-
	Jump3CP	○	○	○	-

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	JumpTLZ	○	○	○	-
L	LatchEnable	○	○	○	-
	LatchPos	-	-	○	○
	LatchState	-	-	○	○
	LCase$	-	-	○	○
	Left$	-	-	○	○
	Len	-	-	○	○
	LimitTorque	○	○	○	○
	LimitTorqueLP	○	○	○	○
	LimitTorqueStop	○	○	○	○
	LimitTorqueStopLP	○	○	○	○
	LimZ	○	○	○	○
	LimZMargin	○	○	○	○
	Line Input	○	○	○	-
	Line Input#	○	○	○	-
	LJM	-	-	○	○
	LoadPoints	○	○	○	-
	Local	○	○	○	○
	LocalClr	○	○	○	-
	LocalDef	-	-	○	○
	Lof	-	-	○	○
	LogIn	-	-	○	○
	Long	-	-	○	-
	LSet$	-	-	○	○
	LShift	-	-	○	○
	LShift64	-	-	○	○
	LTrim$	-	-	○	○
M	Mask	-	-	○	-
	MCal	○	○	○	-
	MCalComplete	-	-	○	○

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	MCordr	○	○	○	○
	MemIn	-	-	○	○
	MemInW	-	-	○	○
	MemOff	○	○	○	-
	MemOn	○	○	○	-
	MemOut	○	○	○	-
	MemOutW	○	○	○	-
	MemSw	-	-	○	○
	MHour	-	-	○	○
	Mid$	-	-	○	○
	MkDir	○	○	○	-
	Mod	-	-	○	-
	Motor	○	○	○	○
	Move	○	○	○	-
	MsgBox	○	○	○	○
	MyTask	-	-	○	○
N	Next	-	-	○	-
	Not	-	-	○	-
O	Off	○	○	○	-
	OLAccel	○	○	○	○
	OLRate	○	○	○	○
	On	○	○	○	-
	OnErr	-	-	○	-
	OpBCD	○	○	○	-
	OpenCom	○	○	○	○
	OpenDB	○	○	○	-
	OpenNet	○	○	○	○
	Oport	-	-	○	○
	Or	-	-	○	-
	Out	○	○	○	○

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	OutReal	○	○	○	○
	OutW	○	○	○	○
P	P#	○	○	○	-
	PAgl	-	-	○	○
	Pallet	○	○	○	○
	PalletClr	○	○	○	-
	ParseStr	○	○	○	○
	Pass	○	○	○	-
	Pause	-	-	○	-
	PauseOn	-	-	○	○
	PDescription	○	○	○	-
	PDescription$	○	○	-	○
	PDef	-	-	○	○
	PDel	○	○	○	-
	PerformMode	○	○	○	○
	PG_FastStop	○	○	○	-
	PG_LSpeed	○	○	○	○
	PG_Scan	○	○	○	-
	PG_SlowStop	○	○	○	-
	PLabel	○	○	○	-
	PLabel$	-	-	○	○
	Plane	○	○	○	○
	PlaneClr	○	○	○	-
	PlaneDef	-	-	○	○
	PList	○	○	○	-
	PLocal	○	○	○	○
	Pls	-	-	○	○
	PNumber	-	-	○	○
	PosFound	-	-	○	○
	Power	○	○	○	○

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	PPls	-	-	○	○
	Preserve	-	-	○	-
	Print	○	○	○	-
	Print#	○	○	○	-
	PTCLR	○	○	○	-
	PTPBoost	○	○	○	○
	PTPBoostOK	-	-	○	○
	PTPTime	-	-	○	○
	PTran	○	○	○	-
	PTRQ	○	○	○	○
	Pulse	○	○	○	○
Q	QP	○	○	○	-
	QPDecelR	○	○	○	○
	QPDecelS	○	○	○	○
	Quit	-	-	○	-
R	RadToDeg	-	-	○	○
	Randmize	○	○	○	-
	Range	○	○	○	-
	Read	○	○	○	-
	ReadBin	○	○	○	-
	Real	-	-	○	-
	RealAccel	-	-	○	○
	RealPls	-	-	○	○
	RealPos	-	-	○	○
	RealTorque	-	-	○	○
	Recover	○	-	○	○
	RecoverPos	-	-	○	○
	Redim	○	○	○	-
	Rename	○	○	○	-
	RenDir	○	○	○	-

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	Reset	○	○	○	-
	ResetElapsedTime	○	○	○	-
	Restart	○	-	○	-
	Resume	-	-	○	-
	Return	-	-	○	-
	Right$	-	-	○	○
	RmDir	○	○	○	-
	Rnd	-	-	○	○
	Robot	○	○	○	○
	RobotInfo	-	-	○	○
	RobotInfo$	-	-	○	○
	RobotModel$	-	-	○	○
	RobotName$	-	-	○	○
	RobotSerial$	-	-	○	○
	RobotType	-	-	○	○
	ROpen	○	○	○	-
	ROTOK	○	○	○	○
	RSet$	-	-	○	○
	RShift64	-	-	○	○
	RShift	-	-	○	○
	RTrim$	-	-	○	○
	RunDialog	-	-	○	-
S	SafetyOn	-	-	○	○
	SavePoints	○	○	○	-
	Seek	○	○	○	-
	Select...Send	-	-	○	-
	SelectDB	○	○	○	○
	Sense	○	○	○	-
	SetCom	○	○	○	-
	SetIn	○	○	○	-

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	SetInReal	○	○	○	-
	SetInW	○	○	○	-
	SetLatch	○	○	○	-
	SetNet	○	○	○	-
	SetSw	○	○	○	-
	SF_GetParam	○	○	○	○
	SF_GetParam$	○	○	○	○
	SF_GetStatus	○	○	○	○
	SF_LimitSpeedS	○	○	○	○
	SF_LimitSpeedSEnable	○	○	○	○
	SF_RealSpeedS	○	○	○	○
	SF_PeakSpeedS	○	○	○	○
	SF_PeakSpeedSClear	○	○	○	-
	SFree	○	○	○	○
	Sgn	-	-	○	○
	Short	-	-	○	-
	Shutdown	○	○	○	○
	Signal	○	○	○	-
	SimGet	-	-	○	-
	SimSet	○	○	○	-
	Sin	-	-	○	○
	SingularityAngle	○	○	○	○
	SingularityDist	○	○	○	○
	SingularitySpeed	○	○	○	○
	SLock	○	○	○	-
	SoftCP	○	○	○	○
	Space$	-	-	○	○
	Speed	○	○	○	○
	SpeedFactor	○	○	○	○
	SpeedR	○	○	○	○

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	SpeedRLimitation	○	○	○	○
	SpeedS	○	○	○	○
	SPELCom_Event	○	○	○	-
	Sqr	-	-	○	○
	ST	-	-	○	○
	StartMain	-	-	○	-
	Stat	-	-	○	○
	Str$	-	-	○	○
	String	-	-	○	-
	Sw	-	-	○	○
	SyncLock	-	-	○	-
	SyncUnlock	-	-	○	-
	SyncRobots	○	○	○	○
	SysConfig	○	○	-	-
	SysErr	-	-	○	○
T	Tab$	-	-	○	○
	Tan	-	-	○	○
	TargetOK	-	-	○	○
	TaskDone	-	-	○	○
	TaskInfo	-	-	○	○
	TaskInfo$	-	-	○	○
	TaskState	○	○	○	○
	TaskWait	○	○	○	-
	TC	○	○	○	-
	TCLim	○	○	○	○
	TCPSpeed	-	-	○	○
	TCSpeed	○	○	○	○
	TeachOn	-	-	○	○
	TGo	○	○	○	-
	Till	○	○	○	-

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	TillOn	-	-	○	○
	Time	○	○	○	○
	Time$	-	-	○	○
	TLClr	○	○	○	-
	TLDef	-	-	○	○
	TLSet	○	○	○	○
	TMOut	○	○	○	-
	TMove	○	○	○	-
	Tmr	-	-	○	○
	TmReset	○	○	○	-
	Toff	○	○	○	-
	Ton	○	○	○	-
	Tool	○	○	○	○
	Trap	-	-	○	-
	Trim$	-	-	○	○
	TW	-	-	○	○
U	UBound	-	-	○	○
	UByte	-	-	○	-
	UCase$	-	-	○	○
	UInt32	-	-	○	-
	UOpen	○	○	○	-
	UpdateDB	○	○	○	-
	UShort	-	-	○	-
V	Val	-	-	○	○
	VCal	○	○	○	-
	VCalPoints	○	○	○	-
	VCls	-	-	○	-
	VCreateCalibration	-	-	○	-
	VCreateObject	-	-	○	-
	VCreateSequence	-	-	○	-

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	VDefArm	-	-	○	-
	VDefGetMotionRange	○	-	○	-
	VDefLocal	-	-	○	-
	VDefSetMotionRange	○	-	○	-
	VDefTool	-	-	○	-
	VDeleteCalibration	-	-	○	-
	VDeleteObject	-	-	○	-
	VDeleteSequence	-	-	○	-
	VGet	-	-	○	-
	VGoCenter	-	-	○	-
	VisCalib	-	-	-	-
	VisCalInfo	-	-	-	○
	VisCalLoad	-	-	-	-
	VisCalSave	-	-	-	-
	VisTrans	-	-	-	○
	VLoad	-	-	○	-
	VLoadModel	-	-	○	-
	VRun	-	-	○	-
	VSave	-	-	○	-
	VSaveImage	-	-	○	-
	VSaveModel	-	-	○	-
	VSD	○	○	○	○
	VSet	-	-	○	-
	VShowModel	○	-	○	-
	VStatsReset	-	-	○	-
	VStatsResetAll	-	-	○	-
	VStatsSave	-	-	○	-
	VStatsShow	○	○	○	-
	VTeach	-	-	○	-
	VTrain	○	-	○	-

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	VxCalib	○	○	○	-
	VxCalDelete	○	○	○	-
	VxCalLoad	○	○	○	-
	VxCalInfo	-	-	○	○
	VxCalSave	○	○	○	-
	VxTrans	-	-	○	○
W	Wait	○	○	○	-
	WaitNet	○	○	○	-
	WaitPos	○	○	○	-
	WaitSig	○	○	○	-
	Weight	○	○	○	○
	Where	○	○	-	-
	WindowStatus	-	-	○	○
	WorkQue_Add	○	○	○	-
	WorkQue_AutoRemove	○	○	○	○
	WorkQue_Get	○	○	○	○
	WorkQue_Len	○	○	○	○
	WorkQue_List	○	○	-	-
	WorkQue_Reject	○	○	○	○
	WorkQue_Remove	○	○	○	-
	WorkQue_Sort	○	○	○	○
	WorkQue_UserData	○	○	○	○
	WOpen	○	○	○	-
	Wrist	○	○	○	○
	Write	○	○	○	-
	WriteBin	○	○	○	-
X	Xor	-	-	○	-
	Xqt	-	-	○	-
	XY	-	-	○	○
	XYLim	○	○	○	○

命令		命令視窗		程式	函數
		RC+	TP3/TP4		
	XYLimClr	○	○	○	-
	XYLimDef	-	-	○	○
	XYLimMode	○	○	○	○

5. Appendix B：相容性相關注意事項

5.1 B-1：與Epson RC+ 6.0的相容性相關注意事項

5.1.1 概要

Epson RC+ 8.0與Epson RC+7.0相容。

在此匯總了透過RC620控制器使用Epson RC+ 6.0的使用者在透過RC800系列控制器使用Epson RC+ 7.0、Epson RC+ 8.0時所需閱讀並瞭解的內容。

使用Epson RC+ 7.0、Epson RC+ 8.0時，可能會因硬體、支援的機械手、關節數以及軟體執行環境的不同，而有不同於Epson RC+ 6.0的部分。請事先理解相關內容，安全地使用機器人。

設計Epson RC+ 7.0、Epson RC+ 8.0時，盡可能地保留與以往產品的相容性進行改善，並且有導入先進的軟體技術。然而，為了專用於機器人控制器並使其更容易使用，部分內容並未完全滿足與Epson RC+ 6.0的相容性或被刪除。

如下所述的相容性相關描述，是以Epson RC+ 6.0為基礎，並與Epson RC+ 7.0、Epson RC+ 8.0進行比較。

5.1.2 一般差異

如下所述為Epson RC+ 6.0和Epson RC+ 7.0、Epson RC+ 8.0的一般差異。

項目	Epson RC+ 7.0, Epson RC+ 8.0	Epson RC+ 6.0
工作數量	最多32個工作 (背景工作時最多16個)	最多32個工作 (背景工作時最多16個)
工作種類	可指定NoPouse工作 可指定NoEmgAbort工作 可指定背景工作	可指定NoPouse工作 可指定NoEmgAbort工作 可指定背景工作
TRAP ERROR等特殊TRAP	支援	支援
以TRAP編號啟動的工作	專用工作編號	專用工作編號
多軸機械手	支援	支援
機器人編號	1～16	1～16
Real型的有效位數	6位	6位
Double型的有效位數	14位	14位
陣列元素數	非字串變數 本地變數：2,000 全域變數：100,000 模組變數：100,000 備份變數：4,000	非字串變數 本地變數：2,000 全域變數：100,000 模組變數：100,000 備份變數：4,000
	字串變數 本地變數：200 全域變數：10,000 模組變數：10,000 備份變數：400	字串變數 本地變數：200 全域變數：10,000 模組變數：10,000 備份變數：400

項目	Epson RC+ 7.0, Epson RC+ 8.0	Epson RC+ 6.0
裝置編號	21：PC 22：遠端 20：TP4	21：PC 22：遠端 28：LCD
控制裝置	遠端I/O PC 遠端Ethernet TP4	遠端I/O PC
計時器編號的範圍	0～63	0～63
程式容量	8MB	8MB
SyncLock、SyncUnlock用信號編號的範圍	0～63	0～63
WaitSig、Signal用信號編號的範圍	0～63	0～63
記憶體I/O點數	1024	1024
I/O連接埠編號	與Epson RC+ 6.0通用	
乙太網路的連接埠編號	201～216	201～216
遠端輸入/輸出分配	已分配功能(標配)	無預設
RS-232C通訊連接埠編號 SPEL+控制部分	1～8, 1001～1008	1～8, 1001, 1002
執行RS-232C通訊連接埠的OpenCom	必須	必須
輸入/輸出到檔案	支援	支援
存取檔案的檔案編號	30～63	30～63
資料庫存取編號	501～508	501～508
VisionGuide	智慧型相機型 影像擷取卡型	智慧型相機型 影像擷取卡型
輸送帶追蹤	支援	支援
PG機器人	支援	支援
OCR	支援	支援
安全性	支援	支援
VB Guide 6.0 (RC+ API)	支援	支援
現場匯流排I/O的使用	使用一般I/O命令	使用一般I/O命令
現場匯流排 主控	不保證回應性	不保證回應性
現場匯流排 從控	保證回應性	保證回應性
GUI Builder	支援	支援
錯誤編號	與Epson RC+6.0通用	

5.1.3 命令相容性一覽

- +：有擴充並變更功能但維持高階相容性的命令
- -：未變更的命令
- !：功能或語法有變更且需注意的命令
- !!：有較大變更且需注意的命令
- x：已刪除的命令

命令		相容性	備註
A	Abs函數	-	
	Accel	-	
	Accel函數	-	
	AccelMax函數	-	
	AccelR	-	
	AccelR函數	-	
	AccelS	-	
	AccelS函數	-	
	Acos函數	-	
	AglToPls函數	-	
	Agl函數	-	
	AlignECP函數	-	
	Align函數	-	
	And	-	
	Arc	-	
	Arc3	-	
	Arch	-	
	Arch函數	-	
	Arm	-	
	ArmClr	-	
	ArmDef函數	-	
	ArmSet	-	
	ArmSet函數	-	
	Arm函數	-	
	Asc函數	-	
	Asin函數	-	
	Atan2函數	-	

命令		相容性	備註
	Atan函數	-	
	ATCLR	-	
	ATRQ	-	
	ATRQ函數	-	
B	Base	-	
	Base函數	-	
	BClr函數	-	
	BGo	+	新增動作模式指定
	BMove	-	
	Boolean	-	
	Box	+	新增遠端輸出邏輯設定的指定
	Box函數	-	
	BoxClr函數	-	
	BoxDef函數	-	
	Brake	-	
	Brake函數	-	
	BSet函數	-	
	BTst函數	-	
	Byte	-	
C	Call	-	
	ChkCom函數	-	
	ChkNet函數	-	
	Chr$函數	-	
	ClearPoints	-	
	CloseCom	-	
	CloseNet	-	
	Cls	-	
	Cos函數	-	
	CP	-	
	CP函數	-	
	CTReset	-	

命令		相容性	備註
	CtrlDev函數	-	
	CtrlInfo函數	-	
	Ctr函數	-	
	CurPos函數	-	
	Curve	-	
	CVMove	-	
	CX～CW	-	
	CX～CW函數	-	
D	Date	-	
	Date$函數	-	
	DegToRad函數	-	
	DispDev	-	
	DispDev函數	-	
	Dist函數	-	
	Do...Loop	-	
	Double	-	
E	ECP	-	
	ECPClr	-	
	EcpDef函數	-	
	ECPSet	-	
	ECPSet函數	-	
	ECP函數	-	
	Elbow	-	
	Elbow函數	-	
	Era函數	-	
	EResume	-	
	Erf$函數	-	
	Erl函數	-	
	ErrMsg$函數	-	
	Error	-	
	ErrorOn函數	-	

命令		相容性	備註
	Err函數	-	
	Ert函數	-	
	EStopOn函數	-	
	Exit	-	
	Find	-	
	FindPos函數	-	
	Fine	-	
	Fine函數	-	
	Fix函數	-	
	FmtStr$	-	
	For...Next	-	
	Function...Fend	-	
G	Global	-	
	Go	+	新增動作模式指定
	Gosub...Return	-	
	Goto	-	
H	Halt	-	
	Hand	-	
	Hand函數	-	
	Here	-	
	Here函數	-	
	Hex$函數	-	
	Home	-	
	HomeClr	-	
	HomeDef函數	-	
	HomeSet	-	
	HomeSet函數	-	
	HOrdr	-	
	HOrdr函數	-	
	Hour	-	
	Hour函數	-	

命令		相容性	備註
I	If...EndIf	-	
	In函數	-	
	InBCD函數	-	
	Inertia	-	
	Inertia函數	-	
	InPos函數	-	
	Input	-	
	Input#	-	
	InsideBox函數	-	
	InsidePlane函數	-	
	InStr函數	-	
	Integer	-	
	Int函數	-	
	InW函數	-	
	IOLabel$函數	-	
	IONumber函數	-	
J	J1Flag	-	
	J1Flag函數	-	
	J2Flag	-	
	J2Flag函數	-	
	J4Flag	-	
	J4Flag函數	-	
	J6Flag	-	
	J6Flag函數	-	
	JA函數	-	
	Joint	-	
	JRange	-	
	JRange函數	-	
	JS函數	-	
	JTran	-	
	JT函數	-	

命令		相容性	備註
	Jump	+	新增動作模式指定
	Jump3	-	
	Jump3CP	-	
L	LCase$函數	-	
	Left$函數	-	
	Len函數	-	
	LimZ	-	
	LimZ函數	-	
	Line Input	-	
	Line Input#	-	
	LJM函數	-	
	LoadPoints	-	
	Local	-	
	LocalClr	-	
	LocalDef函數	-	
	Local函數	-	
	Lof函數	-	
	Long	-	
	LSet$函數	-	
	LShift函數	-	
	LTrim$函數	-	
M	Mask	-	
	MemInW函數	-	
	MemIn函數	-	
	MemOff	-	
	MemOn	-	
	MemOut	-	
	MemOutW	-	
	MemSw函數	-	
	Mid$函數	-	
	Mod	-	

命令		相容性	備註
	Motor	-	
	Motor函數	-	
	Move	-	
	MyTask函數	-	
N	Not	-	
O	Off	-	
	OLAccel	-	
	OLAccel函數	-	
	OLRate	-	
	OLRate函數	-	
	On	-	
	OnErr	-	
	OpBCD	-	
	OpenCom	-	
	OpenNet	-	
	Oport函數	-	
	Or	-	
	Out	-	
	OutW	-	
	OutW函數	-	
	Out函數	-	
P	P#	-	
	PAgl函數	-	
	Pallet	+	新增坐標值指定
	Pallet函數	-	
	ParseStr	-	
	ParseStr函數	-	
	Pass	-	
	Pause	-	
	PauseOn函數	-	
	PDef函數	-	

命令		相容性	備註
	PDel	-	
	PLabel	-	
	PLabel$函數	-	
	Plane	-	
	Plane函數	-	
	PlaneClr	-	
	PlaneDef函數	-	
	PList	-	
	PLocal	-	
	PLocal函數	-	
	Pls函數	-	
	PNumber函數	-	
	PosFound函數	-	
	Power	-	
	Power函數	-	
	PPls函數	-	
	Print	-	
	Print#	-	
	PTCLR	-	
	PTPBoost	-	
	PTPBoostOK函數	-	
	PTPBoost函數	-	
	PTPTime函數	-	
	PTran	-	
	PTRQ	-	
	PTRQ函數	-	
	Pulse	-	
	Pulse函數	-	
Q	QP	-	
	Quit	-	
R	RadToDeg函數	-	

命令		相容性	備註
	Randmize	-	
	Range	-	
	Read	-	
	ReadBin	-	
	Real	-	
	RealPls函數	-	
	RealPos函數	-	
	RealTorque	-	
	Redim	-	
	Reset	-	
	Resume	-	
	Return	-	
	RobotInfo函數	-	
	RobotInfo$函數	-	
	RobotModel$函數	-	
	RobotName$函數	-	
	RobotSerial$函數	-	
	RobotType函數	-	
	RSet$函數	-	
	RShift函數	-	
	RTrim$函數	-	
S	SafetyOn函數	-	
	SavePoints	-	
	Select...Send	-	
	Sense	-	
	SetCom	-	
	SetInW	-	
	SetIn	-	
	SetNet	-	
	SetSw	-	
	SFree	-	

命令		相容性	備註
	SFree函數	-	
	Sgn函數	-	
	Signal	-	
	Sin函數	-	
	SLock	-	
	SoftCP	-	
	SoftCP函數	-	
	Space$函數	-	
	Speed	-	
	SpeedR	-	
	SpeedR函數	-	
	SpeedS	-	
	SpeedS函數	-	
	Speed函數	-	
	SPELCom_Event	-	
	Sqr函數	-	
	Stat函數	-	
	Str$函數	-	
	String	-	
	Sw函數	-	
	SyncLock	-	
	SyncUnlock	-	
	SysConfig	-	
	SysErr函數	-	
T	Tab$函數	-	
	Tan函數	-	
	TargetOK函數	-	
	TaskDone函數	-	
	TaskInfo函數	-	
	TaskInfo$函數	-	
	TaskState	-	

命令		相容性	備註
	TaskState函數	-	
	TaskWait	-	
	TC	-	
	TCLim	-	
	TCLim函數	-	
	TCSpeed	-	
	TCSpeed函數	-	
	TGo	+	新增動作模式指定
	TillOn函數	-	
	Time	-	
	Time函數	-	
	Time$函數	-	
	TLClr	-	
	TlDef函數	-	
	TLSet	-	
	TLSet函數	-	
	TMOut	-	
	TMove	-	
	TmReset	-	
	Tmr函數	-	
	Toff	-	
	Ton	-	
	Tool	-	
	Tool函數	-	
	Trap	-	
	Trim$函數	-	
	Tw函數	-	
U	UBound函數	-	
	UCase$函數	-	
V	Val函數	-	
W	Wait	-	

命令		相容性	備註
	WaitNet	-	
	WaitPos	-	
	WaitSig	-	
	Weight	-	
	Weight函數	-	
	Where	-	
	Wrist	-	
	Wrist函數	-	
	Write	-	
	WriteBin	-	
X	Xor	-	
	Xqt	-	
	XY函數	-	
	XYLim	-	
	XYLim函數	-	
	XYLimClr	-	
	XYLimDef	-	
	XYLimDef函數	-	

5.2 B-2：與Epson RC+ 5.0的相容性相關注意事項

5.2.1 概要

Epson RC+ 8.0與Epson RC+7.0相容。

在此匯總了透過RC180控制器使用Epson RC+ 5.0的使用者在透過RC800系列使用Epson RC+ 7.0、Epson RC+ 8.0時所需閱讀並瞭解的內容。

使用Epson RC+ 7.0、Epson RC+ 8.0時，可能會因硬體、支援的機械手、關節數以及軟體執行環境的不同，而有不同於Epson RC+ 5.0的部分。請事先理解相關內容，安全地使用機器人。

設計Epson RC+ 7.0、Epson RC+ 8.0時，盡可能地保留與以往產品的相容性進行改善，並且有導入先進的軟體技術。然而，為了專用於機器人控制器並使其更容易使用，部分內容並未完全滿足與Epson RC+ 5.0的相容性或被刪除。

如下所述的相容性相關描述，是以Epson RC+ 5.0為基礎，並與Epson RC+ 7.0、Epson RC+ 8.0進行比較。

5.2.2 一般差異

如下所述為Epson RC+ 5.0和Epson RC+ 7.0、Epson RC+ 8.0的一般差異。

項目	Epson RC+ 7.0, Epson RC+ 8.0	Epson RC+ 5.0
工作數量	最多32個工作 (背景工作時最多16個)	最多16個工作
工作種類	可指定NoPouse工作 可指定NoEmgAbort工作 可指定背景工作	可指定NoPouse工作 可指定NoEmgAbort工作
TRAP ERROR等特殊TRAP	支援	不支援
以TRAP編號啟動的工作	專用工作編號	專用工作編號
多軸機械手	支援	不支援
機器人編號	1～16	1
Real型的有效位數	6位	6位
Double型的有效位數	14位	14位
陣列元素數	非字串變數 本地變數：2,000 全域變數：100,000 模組變數：100,000 備份變數：4,000	非字串變數 本地變數：1,000 全域變數：10,000 模組變數：10,000 備份變數：1,000
	字串變數 本地變數：200 全域變數：10,000 模組變數：10,000 備份變數：400	字串變數 本地變數：100 全域變數：1,000 模組變數：1,000 備份變數：100

項目	Epson RC+ 7.0, Epson RC+ 8.0	Epson RC+ 5.0
裝置編號	21：PC 22：遠端 20：TP4	21：PC 22：遠端 23：OP
控制裝置	遠端I/O PC 遠端Ethernet TP4	遠端I/O PC OP1 遠端Ethernet
計時器編號的範圍	0～63	0～15
程式容量	8MB	4MB
SyncLock、SyncUnlock用信號編號的範圍	0～63	0～15
WaitSig、Signal用信號編號的範圍	0～63	0～5
記憶體I/O點數	1024	256
I/O連接埠編號	與Epson RC+ 5.0通用	
乙太網路的連接埠編號	201～216	201～208
遠端輸入/輸出分配	已分配功能(標配)	已分配功能(標配)
RS-232C通訊連接埠編號 SPEL+控制部分	1～8, 1001～1008	1～8
執行RS-232C通訊連接埠的OpenCom	必須	必須
輸入/輸出到檔案	支援	不支援
存取檔案的檔案編號	30～63	不支援
資料庫存取編號	501～508	不支援
VisionGuide	智慧型相機型 影像擷取卡型	智慧型相機型
輸送帶追蹤	支援	不支援
PG機器人	支援	不支援
OCR	支援	不支援
安全性	支援	不支援
VBGuide 5.0 (RC+ API)	支援	支援Lite版
現場匯流排I/O的使用	使用一般I/O命令	使用一般I/O命令
現場匯流排 主控	不保證回應性	不支援
現場匯流排 從控	保證回應性	保證回應性
GUI Builder	支援	不支援
錯誤編號	與Epson RC+ 5.0通用	

5.2.3 命令相容性一覽

- +：有擴充並變更功能但維持高階相容性的命令
- -：未變更的命令
- !：功能或語法有變更且需注意的命令
- !!：有較大變更且需注意的命令
- x：已刪除的命令

命令		相容性	備註
A	Abs函數	-	
	Accel	-	
	Accel函數	-	
	AccelMax函數	-	
	AccelR	-	
	AccelR函數	-	
	AccelS	-	
	AccelS函數	-	
	Acos函數	-	
	AglToPls函數	-	
	Agl函數	-	
	AlignECP函數	-	
	Align函數	-	
	And	-	
	Arc	-	
	Arc3	-	
	Arch	-	
	Arch函數	-	
	Arm	-	
	ArmClr	-	
	ArmDef函數	-	
	ArmSet	-	
	ArmSet函數	-	
	Arm函數	-	
	Asc函數	-	
	Asin函數	-	
	Atan2函數	-	

命令		相容性	備註
	Atan函數	-	
	ATCLR	-	
	ATRQ	-	
	ATRQ函數	-	
B	Base	-	
	Base函數	-	
	BClr函數	-	
	BGo	+	新增動作模式指定
	BMove	-	
	Boolean	-	
	Box	+	新增機器人編號指定
	Box函數	+	新增機器人編號指定
	BoxClr函數	+	新增機器人編號指定
	BoxDef函數	+	新增機器人編號指定
	Brake	-	
	Brake函數	-	
	BSet函數	-	
	BTst函數	-	
	Byte	-	
C	Call	+	叫用外部函式
	ChkCom函數	-	
	ChkNet函數	-	
	Chr$函數	-	
	ClearPoints	-	
	CloseCom	-	
	CloseNet	-	
	Cls	-	
	Cos函數	-	
	CP	-	
	CP函數	-	
	CTReset	-	

命令		相容性	備註
	CtrlDev函數	!	變更控制器裝置
	CtrlInfo函數	-	
	Ctr函數	-	
	CurPos函數	-	
	Curve	-	
	CVMove	-	
	CX～CW	+	新增CR、CS、CT
	CX～CW函數	+	新增CR、CS、CT
D	Date	!	僅顯示
	Date$函數	-	
	DegToRad函數	-	
	DispDev	-	
	DispDev函數	-	
	Dist函數	-	
	Do...Loop	-	
	Double	-	
E	ECP	-	
	ECPClr	-	
	EcpDef函數	-	
	ECPSet	-	
	ECPSet函數	-	
	ECP函數	-	
	ElapsedTime函數	-	
	Elbow	-	
	Elbow函數	-	
	Era函數	-	
	EResume	-	
	Erf$函數	-	
	Erl函數	-	
	ErrMsg$函數	-	
	Error	-	

命令		相容性	備註
	ErrorOn函數	-	
	Err函數	-	
	Ert函數	-	
	EStopOn函數	-	
	Exit	-	
	Find	-	
	FindPos函數	-	
	Fine	-	
	Fine函數	-	
	Fix函數	-	
	FmtStr$	-	
	For...Next	-	
	Function...Fend	-	
G	Global	-	
	Go	+	新增動作模式指定
	Gosub...Return	-	
	Goto	-	
H	Halt	-	
	Hand	-	
	Hand函數	-	
	Here	-	
	Here函數	-	
	Hex$函數	-	
	Home	-	
	HomeClr	-	
	HomeDef函數	-	
	HomeSet	-	
	HomeSet函數	-	
	HOrdr	-	
	HOrdr函數	-	
	Hour	-	

命令		相容性	備註
	Hour函數	-	
I	If...EndIf	-	
	In函數	-	
	InBCD函數	-	
	Inertia	-	
	Inertia函數	-	
	InPos函數	-	
	Input	-	
	Input#	+	新增裝置編號
	InsideBox函數	!	新增機器人編號、All指定，無法進行Wait等待
	InsidePlane函數	!	新增機器人編號、All指定，無法進行Wait等待
	InStr函數	-	
	Integer	-	
	Int函數	-	
	InW函數	-	
	IOLabel$函數	-	
	IONumber函數	-	
J	J1Flag	-	
	J1Flag函數	-	
	J2Flag	-	
	J2Flag函數	-	
	J4Flag	-	
	J4Flag函數	-	
	J6Flag	-	
	J6Flag函數	-	
	JA函數	-	
	Joint	-	
	JRange	-	
	JRange函數	-	
	JS函數	-	
	JTran	-	

命令		相容性	備註
	JT函數	-	
	Jump	+	新增動作模式指定
	Jump3	+	
	Jump3CP	+	
L	LCase$函數	-	
	Left$函數	-	
	Len函數	-	
	LimZ	-	
	LimZ函數	-	
	Line Input	-	
	Line Input#	+	新增裝置編號
	LJM函數	-	
	LoadPoints	-	
	Local	-	
	LocalClr	-	
	LocalDef函數	-	
	Local函數	-	
	Lof函數	-	
	Long	-	
	LSet$函數	-	
	LShift函數	-	
	LTrim$函數	-	
M	Mask	-	
	MemInW函數	-	
	MemIn函數	-	
	MemOff	-	
	MemOn	-	
	MemOut	-	
	MemOutW	-	
	MemSw函數	-	
	Mid$函數	-	

命令		相容性	備註
	Mod	-	
	Motor	-	
	Motor函數	-	
	Move	-	
	MyTask函數	-	
N	Not	-	
O	Off	-	
	OLAccel	-	
	OLAccel函數	-	
	OLRate	-	
	OLRate函數	-	
	On	-	
	OnErr	-	
	OpBCD	-	
	OpenCom	-	
	OpenNet	-	
	Oport函數	-	
	Or	-	
	Out	-	
	OutW	-	
	OutW函數	-	
	Out函數	-	
P	P#	-	
	PAgl函數	-	
	Pallet	-	新增坐標值指定
	Pallet函數	-	
	ParseStr	-	
	ParseStr函數	-	
	Pass	+	
	Pause	-	
	PauseOn函數	-	

命令		相容性	備註
	PDef函數	-	
	PDel	-	
	PLabel	-	
	PLabel$函數	-	
	Plane	+	新增機器人編號指定
	Plane函數	+	新增機器人編號指定
	PlaneClr	+	新增機器人編號指定
	PlaneDef函數	+	新增機器人編號指定
	PList	!	變更顯示形式
	PLocal	-	
	PLocal函數	-	
	Pls函數	-	
	PNumber函數	-	
	PosFound函數	-	
	Power	-	
	Power函數	-	
	PPls函數	-	
	Print	-	
	Print#	+	變更裝置編號
	PTCLR	-	
	PTPBoost	-	
	PTPBoostOK函數	-	
	PTPBoost函數	-	
	PTPTime函數	-	
	PTran	-	
	PTRQ	-	
	PTRQ函數	-	
	Pulse	-	
	Pulse函數	-	
Q	QP	-	
	Quit	-	

命令		相容性	備註
R	RadToDeg函數	-	
	Randmize	-	
	Range	-	
	Read	-	
	ReadBin	-	
	Real	-	
	RealPls函數	-	
	RealPos函數	-	
	RealTorque	-	
	Redim	-	
	Reset	-	
	ResetElapsedTime	-	
	Resume	-	
	Return	-	
	RobotInfo函數	+	新增資訊
	RobotInfo$函數	+	新增預設點檔名顯示
	RobotModel$函數	-	
	RobotName$函數	-	
	RobotSerial$函數	-	
	RobotType函數	-	
	RSet$函數	-	
	RShift函數	-	
	RTrim$函數	-	
S	SafetyOn函數	-	
	SavePoints	-	
	Select...Send	-	
	Sense	-	
	SetCom	-	
	SetInW	-	
	SetIn	-	
	SetNet	-	

命令		相容性	備註
	SetSw	-	
	SFree	-	
	SFree函數	-	
	Sgn函數	-	
	Signal	-	
	Sin函數	-	
	SLock	-	
	SoftCP	-	
	SoftCP函數	-	
	Space$函數	-	
	Speed	-	
	SpeedR	-	
	SpeedR函數	-	
	SpeedS	-	
	SpeedS函數	-	
	Speed函數	-	
	SPELCom_Event	-	
	Sqr函數	-	
	Stat函數	+	新增資訊
	Str$函數	-	
	String	-	
	Sw函數	-	
	SyncLock	-	
	SyncUnlock	-	
	SysConfig	+	新增資訊
	SysErr函數	+	新增警告取得功能
T	Tab$函數	-	
	Tan函數	-	
	TargetOK函數	-	
	TaskDone函數	-	
	TaskInfo函數	-	

命令		相容性	備註
	TaskInfo$函數	-	
	TaskState	+	新增背景工作顯示
	TaskState函數	-	
	TaskWait	-	
	TC	-	
	TCLim	-	
	TCLim函數	-	
	TCSpeed	-	
	TCSpeed函數	-	
	TGo	+	新增動作模式指定
	TillOn函數	-	
	Time	!	僅顯示
	Time函數	-	
	Time$函數	-	
	TLClr	-	
	TlDef函數	-	
	TLSet	-	
	TLSet函數	-	
	TMOut	-	
	TMove	-	
	TmReset	-	
	Tmr函數	-	
	Toff	-	
	Ton	-	
	Tool	-	
	Tool函數	-	
	Trap	!	新增將控制器狀態作為觸發的Trap
	Trim$函數	-	
	Tw函數	-	
U	UBound函數	-	
	UCase$函數	-	

命令		相容性	備註
V	Val函數	-	
W	Wait	!	將全域變數等新增到等待條件中
	WaitNet	-	
	WaitPos	-	
	WaitSig	-	
	Weight	+	新增S、T指定
	Weight函數	+	新增S、T指定
	Where	-	
	Wrist	-	
	Wrist函數	-	
	Write	-	
	WriteBin	-	
X	Xor	-	
	Xqt	-	
	XY函數	-	
	XYLim	-	
	XYLim函數	-	
	XYLimClr	-	
	XYLimDef	-	
	XYLimDef函數	-	

5.2.4 Epson RC+ Ver.4.*的轉移命令一覽(未支援Epson RC+5.0)

AOpen	BOpen	Calib
CalPls	ChDir	ChDrive
Close	Cnv_AbortTrack	Cnv_Downstream
Cnv_Fine	Cnv_Fine函數	Cnv_Name$函數
Cnv_Number函數	Cnv_Point函數	Cnv_PosErr函數
Cnv_Pulse函數	Cnv_QueAdd	Cnv_QueGet函數
Cnv_QueLen函數	Cnv_QueList	Cnv_QueMove
Cnv_QueReject	Cnv_QueReject函數	Cnv_QueRemove
Cnv_QueUserData	Cnv_QueUserData函數	Cnv_RobotConveyor函數

Cnv_Speed函數	Cnv_Trigger	Cnv_Upstream函數
Cont	Copy	CurDir$函數
CurDrive$函數	Declare	Del
Eof函數	Eval函數	FbusIO_GetBusStatus函數
FbusIO_GetDeviceStatus函數	FbusIO_SendMsg	FileDateTime$函數
FileExists函數	FileLen函數	FolderExists函數
FreeFile函數	GetCurrentUser$	Hofs
Hofs函數	ImportPoints	InputBox
LogIn函數	MCalComplete函數	MCal
MCordr	MCordr函數	MKDir
MsgBox	Recover函數	Rename
RenDir	Restart	RmDir
Robot	Robot函數	ROpen
RunDialog	Seek	Shutdown
UOpen	WOpen	

6. Appendix C：Epson RC+ 8.0的命令

6.1 C-1：Epson RC+ 4.0以後版本增加的命令一覽

A		
AbortMotion	AccelMax函數	AglToPls函數
AIO_Out	AIO_Out函數	AIO_OutW
AIO_OutW函數	AIO_Set	AIO_Set函數
AIO_TrackingSet	AIO_TrackingStart	AIO_TrackingEnd
AIO_TrackingOn函數	AIO_In函數	AIO_InW函數
Align函數	AlignECP函數	AreaCorrection函數
AreaCorrectionClr	AreaCorrectionDef函數	AreaCorrectionInv函數
AreaCorrectionOffset函數	AreaCorrectionSet	ArmCalib
ArmCalib函數	ArmCalibSet	ArmCalibSet函數
ArmCalibClr	ArmCalibDef函數	ArmDef函數
ATCLR	AtHome函數	ATRQ
ATRQ函數	AutoLJM	AutoLJM函數
AvoidSingularity	AvoidSingularity函數	

B		
BClr函數	BClr64函數	Box
Box函數	BoxClr函數	BoxDef函數
Brake函數	BSet函數	BSet64函數
BTst函數	BTst64函數	

C		
ChDisk	ChkCom函數	ChkNet函數
CloseCom	CloseDB	CloseNet
Cls	CP	CP函數
CP_Offset	CP_Offset函數	CR
CR函數	CS	CS函數
CT	CT函數	CtrlDev函數
Curve	CVMove	Cnv_Accel
Cnv_Accel函數	Cnv_AccelLim	Cnv_AccelLim函數
Cnv_Adjust	Cnv_AdjustClear	Cnv_AdjustGet
Cnv_AdjustSet	Cnv_DownStream	Cnv_Mode

C		
Cnv_Mode函數	Cnv_OffsetAngle	Cnv_OffsetAngle函數
Cnv_PosErrOffset	Cnv_Upstream	CollisionDetect
CollisionDetect函數		

D		
DegToRad函數	DeleteDB	DiffPoint函數
DispDev	DispDev函數	Dist函數

E		
EcpDef函數	ElapsedTime函數	EResume
Errb函數	ErrorOn函數	Error
EStopOn函數	Exit	ExportPoints

F		
FindPos函數	Find	FineDist
FineDist函數	FineStatus函數Fix函數	Flush
Fmtstr		

G	
GetRobotInsideBox函數	GetRobotInsidePlane函數

H		
Hand_On	Hand_On函數	Hand_Off
Hand_Off函數	Hand_TW函數	Hand_Def函數
Hand_Type函數	Hand_Label$函數	Hand_Number函數
HealthCalcPeriod	HealthCalcPeriod函數	HealthCtrlAlarmOn函數
HealthCtrlInfo	HealthCtrlInfo函數	HealthCtrlRateOffset
HealthCtrlReset	HealthCtrlWarningEnable	HealthCtrlWarningEnable函數
HealthRateCtrlInfo函數	HealthRateRBInfo函數	HealthRBAlarmOn函數
HealthRBAnalysis	HealthRBAnalysis函數	HealthRBDistance
HealthRBDistance函數	HealthRBInfo	HealthRBInfo函數
HealthRBRateOffset	HealthRBReset	HealthRBSpeed
HealthRBSpeed函數	HealthRBStart	HealthRBStop
HealthRBTRQ	HealthRBTRQ函數	HealthRBWarningEnable

H		
HealthRBWarningEnable函數	Here	Here函數
Hex$函數	HofsJointAccuracy	HomeClr
HomeDef函數		

I		
InReal函數	InsideBox函數	InsidePlane函數
InStr函數	IODef函數	IOLabel$函數
IONumber函數		

J		
J1Angle	J1Angle函數	J4Angle
JA函數	Joint	JointAccuracy
JointAccuracy函數	JumpTLZ	JTran

L		
LatchEnable	LatchState函數	LatchPos函數
LimZMargin	LimZMargin函數	LimitTorque
LimitTorque函數	LimitTorqueLP	LimitTorqueLP函數
LimitTorqueStop	LimitTorqueStop函數	LimitTorqueStopLP
LimitTorqueStopLP函數	LJM函數	LocalDef函數
LShift64函數		

M		
MemInW函數	MemOutW	MHour函數

O		
OLAccel	OLAccel函數	OpenCom
OpenCom函數	OpenDB	OpenNet
OpenNet函數	OutReal	OutReal函數

P		
P#	PalletClr	PauseOn函數
PDef函數	PDel	PDescription
PDescription函數	PerformMode	PerformMode函數
PG_FastStop	PG_LSpeed	PG_LSpeed函數

P		
PG_Scan	PG_SlowStop	PLabel
PLabel$函數	PlaneClr	PlaneDef
Plane	Plane函數	PList
PLocal	PLocal函數	PNumber函數
PosFound函數	PTCLR	PTPBoostOK函數
PTPTime函數	PTran	PTRQ
PTRQ函數		

Q		
QPDECELR	QPDECELR函數	QPDECELS
QPDECELS函數		

R		
RadToDeg函數	Randomize	ReadBin
Read	RealAccel函數	RealPls函數
RealPos函數	RealTorque函數	RecoverPos函數
Recover	Redim	ResetElapsedTime
Rnd函數	RobotInfo函數	RobotInfo$函數
RobotModel$函數	RobotName$函數	RobotSerial$函數
RobotType函數	ROTOK函數	RShift64函數

S		
SafetyOn函數	SelectDB	SetCom
SetInW	SetIn	SetNet
SetSw	SF_GetParam函數	SF_GetParam$函數
SF_GetStatus函數	SF_LimitSpeedS	SF_LimitSpeedS函數
SF_LimitSpeedEnable	SF_LimitSpeedEnable函數	SF_PeakSpeedS
SF_PeakSpeedS函數	SF_PeakSpeedSClear	SF_RealSpeedS
SF_RealSpeedS函數	Shutdown函數	SimGet
SimSet	SingularityAngle	SingularityAngle函數
SingularityDist	SingularityDist函數	SingularitySpeed
SingularitySpeed函數	SoftCP	SoftCP函數
SpeedFactor	SpeedFactor函數	StartMain

S		
SyncRobots	SyncRobots函數	SysErr函數

T		
Tab$函數	TargetOK函數	TaskDone函數
TaskInfo函數	TaskInfo$函數	TaskState
TaskState函數	TaskWait	TC
TCLim	TCLim函數	TCSpeed
TCSpeed函數	TeachOn函數	TillOn函數
TlDef函數	Toff	Ton

U	
UBound函數	UpdateDB

V		
VDefArm	VDefLocal	VDefSetMotionRange
VDefGetMotionRange	VDefTool	VGoCenter
VSD	VSD函數	VxCalib
VxCalDelete	VxCalLoad	VxCalInfo函數
VxCalSave		

W		
WaitNet	WaitPos	Where
WindosStatus函數	WorkQue_Add	WorkQue_AutoRemove
WorkQue_AutoRemove函數	WorkQue_Get函數	WorkQue_Len函數
WorkQue_List	WorkQue_Reject	WorkQue_Reject函數
WorkQue_Remove	WorkQue_Sort	WorkQue_Sort函數
WorkQue_UserData	WorkQue_UserData函數	WriteBin
Write		

X		
XYLimClr	XYLimDef	XY函數

6.2 C-2：Epson RC+ 8.0各版本增加的命令一覽

Epson RC+ 8.0版本	新增的命令
Ver.8.0.0	ClearHistory, ContManualRecover, EncTemper, EncTemper函數 , History, SpeedRLimitation, SpeedRLimitation函數

6.3 C-3：EPSON RC+ 7.0各版本增加的命令一覽

EPSON RC+ 6.0、5.0、4.0通用

EPSON RC+ 7.0版本	新增的命令
Ver.7.5.4	Cnv_AccelLim, Cnv_AccelLim函數 SF_GetParam函數, SF_GetParam$函數 SF_GetStatus函數, SF_LimitSpeedS SF_LimitSpeedS函數, SF_LimitSpeedEnable SF_LimitSpeedEnable函數, SF_PeakSpeedS SF_PeakSpeedS函數, SF_PeakSpeedSClear SF_RealSpeedS, SF_RealSpeedS函數
Ver.7.5.3	AreaCorrection函數, AreaCorrectionClr AreaCorrectionDef函數, AreaCorrectionInv函數 AreaCorrectionOffset函數, AreaCorrectionSet Cnv_PosErrOffset
Ver.7.5.2	XYLimMode, XYLimMode函數
Ver.7.5.1	ArmCalib, ArmCalib函數 ArmCalibSet, ArmCalibSet函數 ArmCalibClr, ArmCalibDef函數 JointAccuracy, JointAccuracy函數 HofsJointAccuracy, Hand_On Hand_On函數, Hand_Off Hand_Off函數, Hand_TW函數 Hand_Def函數, Hand_Type函數 Hand_Label$函數, Hand_Number函數 Cnv_Adjust, Cnv_AdjustClear Cnv_AdjustGet, Cnv_AdjustSet DiffPoint函數, ROTOK函數
Ver.7.4.3	AIO_TrackingSet, AIO_TrackingStart AIO_TrackingEnd, AIO_TrackingOn函數
Ver.7.4.1	AutoOrientationFlag, AutoOrientationFlag函數
Ver.7.3.4	SimGet, SimSet
Ver.7.3.3	HealthCtrlWarningEnable, HealthCtrlWarningEnable函數 HealthRBWarningEnable, HealthRBWarningEnable函數
Ver.7.3.2	PDescription, PDescription函數

EPSON RC+ 7.0版本	新增的命令
Ver.7.3.1	AIO_Out, AIO_Out函數 AIO_OutW, AIO_OutW函數 AIO_Set, AIO_Set函數AIO_In函數 AIO_InW函數, HealthCalcPeriod HealthCalcPeriod函數
Ver.7.3.0	VDefTool, VDefArm VDefLocal, VGoCenter VDefSetMotionRange, VDefGetMotionRange
Ver.7.2.0	CP_Offset, CP_Offset函數 HealthCtrlAlarmOn函數, HealthCtrlInfo HealthCtrlInfo函數, HealthCtrlRateOffset HealthCtrlReset, HealthRateCtrlInfo函數 HealthRateRBInfo函數, HealthRBAlarmOn函數 HealthRBAnalysis, HealthRBAnalysis函數 HealthRBDistance, HealthRBDistance函數 HealthRBInfo, HealthRBInfo函數 HealthRBRateOffset, HealthRBReset HealthRBSpeed HealthRBSpeed函數, HealthRBStart HealthRBStop, HealthRBTRQ HealthRBTRQ函數, J4Angle JumpTLZ, LimitTorqueLP LimitTorqueLP函數, LimitTorqueStop LimitTorqueStop函數, LimitTorqueStopLP LimitTorqueStopLP函數, VSD VSD函數
Ver.7.1.4	CollisionDetect函數
Ver.7.1.3	CollisionDetect, MHour函數
Ver.7.1.2	SingularityDist, SingularityDist函數 ExportPoints
Ver.7.1.0	BClr64函數, BSet64函數 BTst64函數, FineDist FineDist函數, FineStatus函數 Fmtstr, IODef函數 LShift64函數, RealAccel函數 RShift64函數, WorkQue_Add WorkQue_AutoRemove, WorkQue_AutoRemove函數 WorkQue_Get函數, WorkQue_Len函數 WorkQue_List, WorkQue_Reject WorkQue_Reject函數, WorkQue_Remove WorkQue_Sort, WorkQue_Sort函數 WorkQue_UserData, WorkQue_UserData函數
Ver.7.0.3	PerformMode, PerformMode函數

提示

若為Epson RC+7.0 Ver.7.0.0，

Epson RC+ 6.0、5.0、4.0所附加的命令並不相同。

Epson RC+ 6.0、5.0新增的命令

Epson RC+ 7.0版本	Epson RC+ 6.0	Epson RC+ 5.0
Ver.7.0.0	AutoLJM AutoLJM函數 AvoidSingularity AvoidSingularity函數	AbortMotion AutoLJM AutoLJM函數 AvoidSingularity AvoidSingularity函數
	Cnv_Accel Cnv_Accel函數 Cnv_DownStream Cnv_Mode Cnv_Mode函數 Cnv_Upstream	ChDisk CloseDB Cnv_Accel Cnv_Accel函數 Cnv_DownStream Cnv_Mode Cnv_Mode函數 Cnv_Upstream
		CR CR函數 CS CS函數 CT CT函數
	DeleteDB	DeleteDB
	ElapsedTime函數 Errb函數	Errb函數 Flush GetRobotInsideBox函數 GetRobotInsidePlane函數 J1Angle J1Angle函數
	LimZMargin LimZMargin函數 LimitTorque LimitTorque函數	LimZMargin LimZMargin函數 LimitTorque LimitTorque函數 OpenDB
	PalletClr	PalletClr PG_FastStop PG_LSpeed PG_LSpeed函數 PG_Scan PG_SlowStop QPDECELR QPDECELR函數

Epson RC+ 7.0版本	Epson RC+ 6.0	Epson RC+ 5.0
		QPDECELS QPDECELS函數 RecoverPos函數 Recover
	ResetElapsedTime	SelectDB Shutdown函數
	SingularityAngle SingularityAngle函數 SingularitySpeed SingularitySpeed函數 SpeedFactor SpeedFactor函數	SingularityAngle SingularityAngle函數 SingularitySpeed SingularitySpeed函數 SpeedFactor SpeedFactor函數
		StartMain SyncRobots SyncRobots函數 TeachOn函數
	UpdateDB	UpdateDB WindosStatus函數

Epson RC+ 4.0新增的命令

Epson RC+ 7.0版本	Epson RC+ 4.0
Ver.7.0.0	AbortMotion, AccelMax函數 AglToPls函數, Align函數 AlignECP函數, ArmDef函數 ATCLR, AtHome函數 ATRQ, ATRQ函數 AutoLJM, AutoLJM函數 AvoidSingularity, AvoidSingularity函數
	BClr函數, Box Box函數, BoxClr函數 BoxDef函數, Brake函數 BSet函數, BTst 函數
	ChDisk, ChkCom函數 ChkNet函數, CloseCom CloseDB, CloseNet Cls, CP CP函數, CR CR函數, CS CS函數, CT CT函數, CtrlDev函數 Curve, CVMove Cnv_Accel, Cnv_Accel函數Cnv_DownStream Cnv_Mode Cnv_Mode函數, Cnv_OffsetAngle Cnv_OffsetAngle函數, Cnv_Upstream
	DegToRad函數, DeleteDB DispDev, DispDev函數 Dist函數
	EcpDef函數, EResume Errb函數, ErrorOn函數 Error, EStopOn函數 Exit
	FindPos函數, Find FineStatus函數, Fix函數 Flush
	GetRobotInsideBox函數, GetRobotInsidePlane函數
	Here, Here函數 Hex$函數, HomeClr HomeDef函數
	InReal函數, InsideBox函數 InsidePlane函數, InStr函數 IOLabel$函數, IONumber函數
	J1Angle, J1Angle函數 JA函數, Joint JTran

Epson RC+ 7.0版本	Epson RC+ 4.0
	LatchEnable, LatchState函數 LatchPos函數, LimZMargin LimZMargin函數, LimitTorque LimitTorque函數, LJM函數 LocalDef函數
	MemInW函數, MemOutW
	OLAccel, OLAccel函數 OpenCom, OpenCom函數 OpenDB, OpenNet OpenNet函數, OutReal OutReal函數
	P#, PalletClr PauseOn函數, PDef函數 PDel, PG_FastStop PG_LSpeed, PG_LSpeed函數 PG_Scan, PG_SlowStop PLabel, PLabel$函數 PlaneClr, PlaneDef Plane, Plane函數 PList, PLocal PLocal函數, PNumber函數 PosFound函數, PTCLR PTPBoostOK函數, PTPTime函數 PTran, PTRQ PTRQ函數
	QPDECELR, QPDECELR函數 QPDECELS, QPDECELS函數
	RadToDeg函數, Randomize ReadBin, Read RealPls函數, RealPos函數 RealTorque函數, RecoverPos函數 Recover, Redim RobotInfo函數, RobotInfo$ 函數RobotModel$函數, RobotName$函數 RobotSerial$函數, RobotType函數
	SafetyOn函數, SelectDB SetCom, SetInW SetIn, SetNet SetSw, Shutdown函數 SingularityAngle, SingularityAngle函數 SingularitySpeed, SingularitySpeed函數 SoftCP, SoftCP函數 SpeedFactor, SpeedFactor函數 StartMain, SyncRobots SyncRobots函數, SysErr函數

Epson RC+ 7.0版本	Epson RC+ 4.0
	Tab$函數, TargetOK函數 TaskDone函數, TaskInfo函數 TaskInfo$函數, TaskState TaskState函數, TaskWait TC, TCLim TCLim函數, TCSpeed TCSpeed函數, TeachOn函數 TillOn函數, TlDef函數 Toff, Ton
	UBound函數, UpdateDB
	VxCalib, VxCalDelete VxCalLoad, VxCalInfo函數 VxCalSave, VxTrans函數
	WaitNet, WaitPos Where, WindosStatus函數 WriteBin, Write
	XYLimClr, XYLimDef XY函數

6.4 C-4：EPSON RC+ 7.0各版本刪除的命令一覽

EPSON RC+ 6.0、5.0、4.0刪除的命令

EPSON RC+ 7.0版本	EPSON RC+ 6.0	EPSON RC+ 5.0	EPSON RC+ 4.0
Ver.7.1.2	SetLCD	SetLCD	SetLCD
Ver.7.0.0	Dir Type	-	Dir Type

7. Appendix D：常數

7.1 Appendix D：常數

SPEL+程式中提供了可使用的常數。專案創建使用常數值。

常數名稱	值	用法
TRUE	-1	邏輯式
FALSE	0	邏輯式
High	1	
Low	0	
Off	0	
On	1	
Above	1	
Below	2	
NoFlip	1	
若省略Flip	2	
Righty	1	
若省略Lefty	2	
J1	1	
J2	2	
J3	4	
J4	8	
J5	16	
J6	32	
J7	64	
CR	CHR$(13)	
CRLF	CHR$(13)+CHR$(10)	
LF	CHR$(10)	
MB_OK	0	MsgBox旗標
MB_OKCANCEL	1	MsgBox旗標
MB_ABORTRETRYIGNORE	2	MsgBox旗標
MB_YESNOCANCEL	3	MsgBox旗標
MB_YESNO	4	MsgBox旗標
MB_RETRYCANCEL	5	MsgBox旗標
MB_ICONSTOP	16	MsgBox旗標

常數名稱	值	用法
MB_ICONQUESTION	32	MsgBox旗標
MB_ICONEXCLAMATION	48	MsgBox旗標
MB_ICONINFORMATION	64	MsgBox旗標
MB_DEFBUTTON1	0	MsgBox旗標
MB_DEFBUTTON2	256	MsgBox旗標
IDOK	1	MsgBox傳回值
IDCANCEL	2	MsgBox傳回值
IDABORT	3	MsgBox傳回值
IDRETRY	4	MsgBox傳回值
IDIGNORE	5	MsgBox傳回值
IDYES	6	MsgBox傳回值
IDNO	7	MsgBox傳回值
ARC_SHORT	0	Arc, Arc3
ARC_LONG	1	Arc, Arc3
ARC_PLUS	1	Arc
ARC_MINUS	-1	Arc
BACKCOLORMODE_VISUALSTYLE	0	適用於GUI Builder
BACKCOLORMODE_USER	1	適用於GUI Builder
BORDERSTYLE_NONE	0	適用於GUI Builder
BORDERSTYLE_FIXEDSINGLE	1	適用於GUI Builder
BORDERSTYLE_FIXED3D	2	適用於GUI Builder
CNV_QUELEN_ALL	0	Cnv_QueLen
CNV_QUELEN_UPSTREAM	1	Cnv_QueLen
CNV_QUELEN_PICKUPAREA	2	Cnv_QueLen
CNV_QUELEN_DOWNSTREAM	3	Cnv_QueLen
DEVID_SELF	21	CLS
DEVID_TP	24	CLS
DEVID_TP3	30	CLS
DIALOGRESULT_NOE	0	適用於GUI Builder
DIALOGRESULT_OK	1	適用於GUI Builder
DIALOGRESULT_CANCEL	2	適用於GUI Builder

常數名稱	值	用法
DLG_IOMON	102	RunDialog
DLG_ROBOTMNG	100	RunDialog
DLG_VGUIDE	110	RunDialog
DOCK_NONE	0	適用於GUI Builder
DOCK_TOP	1	適用於GUI Builder
DOCK_BOTTOM	2	適用於GUI Builder
DOCK_LEFT	3	適用於GUI Builder
DOCK_RIGHT	4	適用於GUI Builder
DOCK_FILL	5	適用於GUI Builder
DROPDOWNSTYLE_SIMPLE	0	適用於GUI Builder
DROPDOWNSTYLE_DROPDOWN	1	適用於GUI Builder
DROPDOWNSTYLE_DROPDOWNLIST	2	適用於GUI Builder
ERROR_DOINGMOTION	2999	適用於GUI Builder
ERROR_NOMOTION	2998	適用於GUI Builder
EVENTTASKTYPE_NORMAL	0	適用於GUI Builder
EVENTTASKTYPE_NOPAUSE	1	適用於GUI Builder
EVENTTASKTYPE_NOEMGABORT	2	適用於GUI Builder
FORMBORDERSTYLE_NONE	0	適用於GUI Builder
FORMBORDERSTYLE_FIXEDSINGLE	1	適用於GUI Builder
FORMBORDERSTYLE_FIXED3D	2	適用於GUI Builder
FORMBORDERSTYLE_FIXEDDIALOG	3	適用於GUI Builder
FORMBORDERSTYLE_SIZABLE	4	適用於GUI Builder
IMAGEALIGN_TOPLEFT	1	適用於GUI Builder
IMAGEALIGN_TOPCENTER	2	適用於GUI Builder
IMAGEALIGN_TOPRIGHT	3	適用於GUI Builder
IMAGEALIGN_MIDDLELEFT	4	適用於GUI Builder
IMAGEALIGN_MIDDLECENTER	5	適用於GUI Builder
IMAGEALIGN_MIDDLERIGHT	6	適用於GUI Builder
IMAGEALIGN_BOTTOMLEFT	7	適用於GUI Builder
IMAGEALIGN_BOTTOMCENTER	8	適用於GUI Builder
IMAGEALIGN_BOTTOMRIGHT	9	適用於GUI Builder

常數名稱	值	用法
IOTYPE_INPUT	0	IOLabel函數
IOTYPE_OUTPUT	1	IOLabel函數
IOTYPE_MEMORY	2	IOLabel函數
IOSIZE_BIT	1	IOLabel函數
IOSIZE_BYTE	8	IOLabel函數
IOSIZE_WORD	16	IOLabel函數
LANGID_ENGLISH	0	ErrMsg$
LANGID_JAPANESE	1	ErrMsg$
LANGID_GERMAN	2	ErrMsg$
LANGID_FRENCH	3	ErrMsg$
LANGID_SIMPLIFIED_CHINESE	4	ErrMsg$
LANGID_TRADITIONAL_CHINESE	5	ErrMsg$
MODE_STANDARD	1	PerformMode
MODE_HIGH_SPEED	2	PerformMode
MODE_LOW_OSCILLATION	3	PerformMode
ORIENT_HORIZONTAL	0	適用於GUI Builder
ORIENT_VERTICAL	1	適用於GUI Builder
PROGRESSBAR_STYLE_BLOCKS	0	適用於GUI Builder
PROGRESSBAR_STYLE_CONT	1	適用於GUI Builder
PROGRESSBAR_STYLE_MARQUEE	2	適用於GUI Builder
SCROLLBARS_NONE	0	適用於GUI Builder
SCROLLBARS_HORIZ	1	適用於GUI Builder
SCROLLBARS_VERT	2	適用於GUI Builder
SCROLLBARS_BOTH	3	適用於GUI Builder
SETLATCH_PORT_CU_0	24	SetLatch
SETLATCH_PORT_CU_1	25	SetLatch
SETLATCH_PORT_CU_2	26	SetLatch
SETLATCH_PORT_CU_3	27	SetLatch
SETLATCH_TRIGGERMODE_LEADINGEDGE	1	SetLatch
SETLATCH_TRIGGERMODE_TRAILINGEDGE	0	SetLatch
SHUTDOWN_ALL	0	Shutdown

常數名稱	值	用法
SHUTDOWN_RESTART	1	Shutdown
SHUTDOWN_EPSONRC	2	Shutdown
SING_NONE	0	AvoidSingularity
SING_THRU	1	AvoidSingularity
SING_THRUROT	2	AvoidSingularity
SING_VSD	3	AvoidSingularity
SING_AUTO	4	AvoidSingularity
SIZEMODE_NORMAL	0	適用於GUI Builder
SIZEMODE_STRETCHIMAGE	1	適用於GUI Builder
SIZEMODE_AUTOSIZE	2	適用於GUI Builder
SIZEMODE_CENTERIMAGE	3	適用於GUI Builder
SIZEMODE_ZOOM	4	適用於GUI Builder
STARTPOSITION_MANUAL	0	適用於GUI Builder
STARTPOSITION_CENTERSCREEN	1	適用於GUI Builder
STARTPOSITION_CENTERPARENT	2	適用於GUI Builder
TEXTALIGN_LEFT	1	適用於GUI Builder
TEXTALIGN_CENTER	2	適用於GUI Builder
TEXTALIGN_RIGHT	3	適用於GUI Builder
TEXTALIGN_TOPLEFT	1	適用於GUI Builder
TEXTALIGN_TOPCENTER	2	適用於GUI Builder
TEXTALIGN_TOPRIGHT	3	適用於GUI Builder
TEXTALIGN_MIDDLELEFT	4	適用於GUI Builder
TEXTALIGN_MIDDLECENTER	5	適用於GUI Builder
TEXTALIGN_MIDDLERIGHT	6	適用於GUI Builder
TEXTALIGN_BOTTOMLEFT	7	適用於GUI Builder
TEXTALIGN_BOTTOMCENTER	8	適用於GUI Builder
TEXTALIGN_BOTTOMRIGHT	9	適用於GUI Builder
TICKSTYLE_NONE	0	適用於GUI Builder
TICKSTYLE_TOPLEFT	1	適用於GUI Builder
TICKSTYLE_BOTTOMRIGHT	2	適用於GUI Builder
TICKSTYLE_BOTH	3	適用於GUI Builder

常數名稱	值	用法
VISION_SORT_NONE	0	適用於Vision Guide
VISION_SORT_PIXELX	1	適用於Vision Guide
VISION_SORT_PIXELY	2	適用於Vision Guide
VISION_SORT_PIXELXY	3	適用於Vision Guide
VISION_SORT_CAMERAX	4	適用於Vision Guide
VISION_SORT_CAMERAY	5	適用於Vision Guide
VISION_SORT_CAMERAXY	6	適用於Vision Guide
VISION_SORT_ROBOTX	7	適用於Vision Guide
VISION_SORT_ROBOTY	8	適用於Vision Guide
VISION_SORT_ROBOTXY	9	適用於Vision Guide
VISION_SIZETOFIND_ANY	0	適用於Vision Guide
VISION_SIZETOFIND_LARGEST	1	適用於Vision Guide
VISION_SIZETOFIND_SMALLEST	2	適用於Vision Guide
VISION_BACKCOLOR_NONE	0	適用於Vision Guide
VISION_BACKCOLOR_BLACK	1	適用於Vision Guide
VISION_BACKCOLOR_WHITE	2	適用於Vision Guide
VISION_CAMORIENT_STANDALONE	1	適用於Vision Guide
VISION_CAMORIENT_FIXEDDOWN	2	適用於Vision Guide
VISION_CAMORIENT_FIXEDUP	3	適用於Vision Guide
VISION_CAMORIENT_MOBILEJ2	4	適用於Vision Guide
VISION_CAMORIENT_MOBILEJ4	5	適用於Vision Guide
VISION_CAMORIENT_MOBILEJ5	6	適用於Vision Guide
VISION_CAMORIENT_MOBILEJ6	7	適用於Vision Guide
VISION_FOUNDCOLOR_LIGHTGREEN	1	適用於Vision Guide
VISION_FOUNDCOLOR_DARKGREEN	2	適用於Vision Guide
VISION_GRAPHICS_ALL	1	適用於Vision Guide
VISION_GRAPHICS_POSONLY	2	適用於Vision Guide
VISION_GRAPHICS_NONE	3	適用於Vision Guide
VISION_OPERATION_OPEN	1	適用於Vision Guide
VISION_OPERATION_CLOSE	2	適用於Vision Guide
VISION_OPERATION_ERODE	3	適用於Vision Guide

常數名稱	值	用法
VISION_OPERATION_DILATE	4	適用於Vision Guide
VISION_OPERATION_SMOOTH	5	適用於Vision Guide
VISION_OPERATION_SHARPEN1	6	適用於Vision Guide
VISION_OPERATION_SHARPEN2	7	適用於Vision Guide
VISION_OPERATION_HORIZEDGE	8	適用於Vision Guide
VISION_OPERATION_VERTEDGE	9	適用於Vision Guide
VISION_OPERATION_EDGEDETECT1	10	適用於Vision Guide
VISION_OPERATION_EDGEDETECT2	11	適用於Vision Guide
VISION_OPERATION_LAPLACE1	12	適用於Vision Guide
VISION_OPERATION_LAPLACE2	13	適用於Vision Guide
VISION_OPERATION_THIN	14	適用於Vision Guide
VISION_OPERATION_THICKEN	15	適用於Vision Guide
VISION_OPERATION_BINARIZE	16	適用於Vision Guide
VISION_OPERATION_ROTATE	17	適用於Vision Guide
VISION_OPERATION_FLIPHORIZ	18	適用於Vision Guide
VISION_OPERATION_FLIPVERT	19	適用於Vision Guide
VISION_OPERATION_FLIPBOTH	20	適用於Vision Guide
VISION_OPERATION_COLORFILTER	21	適用於Vision Guide
VISION_OPERATION_SUBTRACTABS	22	適用於Vision Guide
VISION_OPERATION_ZOOM	23	適用於Vision Guide
VISION_ACQUIRE_NONE	0	適用於Vision Guide
VISION_ACQUIRE_STATIONARY	1	適用於Vision Guide
VISION_ACQUIRE_STROBED	2	適用於Vision Guide
VISION_TRIGGERMODE_LEADINGEDGE	1	適用於Vision Guide
VISION_TRIGGERMODE_TRAILINGEDGE	2	適用於Vision Guide
VISION_THRESHCOLOR_BLACK	1	適用於Vision Guide
VISION_THRESHCOLOR_WHITE	2	適用於Vision Guide
VISION_OBJTYPE_CORRELATIO	1	適用於Vision Guide
VISION_OBJTYPE_BLOB	2	適用於Vision Guide
VISION_OBJTYPE_EDGE	3	適用於Vision Guide
VISION_OBJTYPE_POLAR	4	適用於Vision Guide

常數名稱	值	用法
VISION_OBJTYPE_LINE	5	適用於Vision Guide
VISION_OBJTYPE_POINT	6	適用於Vision Guide
VISION_OBJTYPE_FRAME	7	適用於Vision Guide
VISION_OBJTYPE_IMAGEOP	8	適用於Vision Guide
VISION_OBJTYPE_OCR	9	適用於Vision Guide
VISION_OBJTYPE_CODEREADER	10	適用於Vision Guide
VISION_OBJTYPE_GEOMETRIC	11	適用於Vision Guide
VISION_DETAILLEVEL_MEDIUM	1	適用於Vision Guide
VISION_DETAILLEVEL_HIGH	2	適用於Vision Guide
VISION_DETAILLEVEL_VERYHIGH	3	適用於Vision Guide
VISION_IMAGESOURCE_CAMERA	1	適用於Vision Guide
VISION_IMAGESOURCE_FILE	2	適用於Vision Guide
VISION_CODETYPE_AUTO	0	適用於Vision Guide
VISION_CODETYPE_EAN13	2	適用於Vision Guide
VISION_CODETYPE_CODE39	3	適用於Vision Guide
VISION_CODETYPE_INTERLEAVED25	4	適用於Vision Guide
VISION_CODETYPE_CODE128	5	適用於Vision Guide
VISION_CODETYPE_CODABAR	6	適用於Vision Guide
VISION_CODETYPE_PDF417	8	適用於Vision Guide
VISION_CODETYPE_QR	10	適用於Vision Guide
VISION_CODETYPE_EAN8	13	適用於Vision Guide
VISION_CODETYPE_UPCA	18	適用於Vision Guide
VISION_CODETYPE_UPCE	19	適用於Vision Guide
VISION_CODETYPE_UPC	20	適用於Vision Guide
VISION_EDGETYPE_SINGLE	1	適用於Vision Guide
VISION_EDGETYPE_PAIR	2	適用於Vision Guide
VISION_IMAGECOLOR_ALL	1	適用於Vision Guide
VISION_IMAGECOLOR_RED	2	適用於Vision Guide
VISION_IMAGECOLOR_GREEN	3	適用於Vision Guide
VISION_IMAGECOLOR_BLUE	4	適用於Vision Guide
VISION_IMAGECOLOR_GRAYSCALE	5	適用於Vision Guide

常數名稱	值	用法
VISION_POINTTYPE_POINT	0	適用於Vision Guide
VISION_POINTTYPE_ENDPOINT	1	適用於Vision Guide
VISION_POINTTYPE_MIDPOINT	2	適用於Vision Guide
VISION_POINTTYPE_PERPTOLINE	3	適用於Vision Guide
VISION_POINTTYPE_STARTPOINT	4	適用於Vision Guide
VISION_POINTTYPE_PERPTOSTARTPOINT	5	適用於Vision Guide
VISION_POINTTYPE_PERPTOMIDPOINT	6	適用於Vision Guide
VISION_POINTTYPE_PERPTOENDPOINT	7	適用於Vision Guide
VISION_REFTYPE_TAUGHTPOINTS	1	適用於Vision Guide
VISION_REFTYPE_UPWARDCAMERA	2	適用於Vision Guide
VISION_IMAGESIZE_320X240	1	適用於Vision Guide
VISION_IMAGESIZE_640X480	2	適用於Vision Guide
VISION_IMAGESIZE_800X600	3	適用於Vision Guide
VISION_IMAGESIZE_1024X768	4	適用於Vision Guide
VISION_IMAGESIZE_1280X1024	5	適用於Vision Guide
VISION_IMAGESIZE_1600X1200	6	適用於Vision Guide
VISION_IMAGESIZE_2048X1536	7	適用於Vision Guide
VISION_IMAGESIZE_2560X1920	8	適用於Vision Guide
VISION_WINTYPE_RECTANGLE	1	適用於Vision Guide
VISION_WINTYPE_ROTATEDRECT	2	適用於Vision Guide
VISION_WINTYPE_CIRCLE	3	適用於Vision Guide
VISION_ORIENT_BOTH	1	適用於Vision Guide
VISION_ORIENT_HORIZ	2	適用於Vision Guide
VISION_ORIENT_VERT	3	適用於Vision Guide
VISION_DIRECTION_INSIDEOUT	1	適用於Vision Guide
VISION_DIRECTION_OUTSIDEIN	2	適用於Vision Guide
VISION_POLARITY_DARK	1	適用於Vision Guide
VISION_POLARITY_LIGHT	2	適用於Vision Guide
VISION_PASSTYPE_SOMEFOUND	1	適用於Vision Guide
VISION_PASSTYPE_ALLFOUND	2	適用於Vision Guide
VISION_PASSTYPE_SOMENOTFOUND	3	適用於Vision Guide

常數名稱	值	用法
VISION_PASSTYPE_ALLNOTFOUND	4	適用於Vision Guide
WIN_IOMON	-1	適用於GUI Builder
WIN_TASKMGR	-2	適用於GUI Builder
WIN_FORCEMON	-3	適用於GUI Builder
WIN_SIMULATOR	-4	適用於GUI Builder
WINDOWSTATE_NORMAL	0	WindowsStatus
WINDOWSTATE_MINIMIZED	1	WindowsStatus
WINDOWSTATE_MAXIMIZED	2	WindowsStatus
WithMove	0	Recover
WithoutMove	1	Recover
DRYRUNOFF	1	SF_GetParam函數
SLS_1_HAND_EN	2	SF_GetParam函數
SLS_1_SPEED	3	SF_GetParam函數
SLS_1_ELBOW_EN	4	SF_GetParam函數
SLS_1_JOINT_EN	5	SF_GetParam函數
SLS_1_JOINTSPEED	6	SF_GetParam函數
SLS_1_WRIST_EN	7	SF_GetParam函數
SLS_1_SHOULDER_EN	8	SF_GetParam函數
SLS_2_HAND_EN	9	SF_GetParam函數
SLS_2_SPEED	10	SF_GetParam函數
SLS_2_ELBOW_EN	11	SF_GetParam函數
SLS_2_JOINT_EN	12	SF_GetParam函數
SLS_2_JOINTSPEED	13	SF_GetParam函數
SLS_2_WRIST_EN	14	SF_GetParam函數
SLS_2_SHOULDER_EN	15	SF_GetParam函數
SLS_3_HAND_EN	16	SF_GetParam函數
SLS_3_SPEED	17	SF_GetParam函數
SLS_3_ELBOW_EN	18	SF_GetParam函數
SLS_3_JOINT_EN	19	SF_GetParam函數
SLS_3_JOINTSPEED	20	SF_GetParam函數
SLS_3_WRIST_EN	21	SF_GetParam函數

常數名稱	值	用法
SLS_3_SHOULDER_EN	22	SF_GetParam函數
SLS_T2_HAND_EN	23	SF_GetParam函數
SLS_T2_SPEED	24	SF_GetParam函數
SLS_T2_ELBOW_EN	25	SF_GetParam函數
SLS_T2_JOINT_EN	26	SF_GetParam函數
SLS_T2_JOINTSPEED	27	SF_GetParam函數
SLS_T2_WRIST_EN	28	SF_GetParam函數
SLS_T2_SHOULDER_EN	29	SF_GetParam函數
SLS_T_SPEED	30	SF_GetParam函數
SLS_T_JOINT_EN	31	SF_GetParam函數
SLS_T_JOINTSPEED	32	SF_GetParam函數
SLS_HAND_OFS_X	33	SF_GetParam函數
SLS_HAND_OFS_Y	34	SF_GetParam函數
SLS_HAND_OFS_Z	35	SF_GetParam函數
SLS_1_DELAY	36	SF_GetParam函數
SLS_2_DELAY	37	SF_GetParam函數
SLS_3_DELAY	38	SF_GetParam函數
SLS_JOINT_POS_EN	39	SF_GetParam函數
SLS_JOINT_POS_ANGLE	40	SF_GetParam函數
SLP_A_XU_EN	41	SF_GetParam函數
SLP_A_XU_POS	42	SF_GetParam函數
SLP_A_XL_EN	43	SF_GetParam函數
SLP_A_XL_POS	44	SF_GetParam函數
SLP_A_YU_EN	45	SF_GetParam函數
SLP_A_YU_POS	46	SF_GetParam函數
SLP_A_YL_EN	47	SF_GetParam函數
SLP_A_YL_POS	48	SF_GetParam函數
SLP_A_ZU_EN	49	SF_GetParam函數
SLP_A_ZU_POS	50	SF_GetParam函數
SLP_A_ZL_EN	51	SF_GetParam函數
SLP_A_ZL_POS	52	SF_GetParam函數

常數名稱	值	用法
SLP_B_XU_EN	53	SF_GetParam函數
SLP_B_XU_POS	54	SF_GetParam函數
SLP_B_XL_EN	55	SF_GetParam函數
SLP_B_XL_POS	56	SF_GetParam函數
SLP_B_YU_EN	57	SF_GetParam函數
SLP_B_YU_POS	58	SF_GetParam函數
SLP_B_YL_EN	59	SF_GetParam函數
SLP_B_YL_POS	60	SF_GetParam函數
SLP_B_ZU_EN	61	SF_GetParam函數
SLP_B_ZU_POS	62	SF_GetParam函數
SLP_B_ZL_EN	63	SF_GetParam函數
SLP_B_ZL_POS	64	SF_GetParam函數
SLP_C_XU_EN	65	SF_GetParam函數
SLP_C_XU_POS	66	SF_GetParam函數
SLP_C_XL_EN	67	SF_GetParam函數
SLP_C_XL_POS	68	SF_GetParam函數
SLP_C_YU_EN	69	SF_GetParam函數
SLP_C_YU_POS	70	SF_GetParam函數
SLP_C_YL_EN	71	SF_GetParam函數
SLP_C_YL_POS	72	SF_GetParam函數
SLP_C_ZU_EN	73	SF_GetParam函數
SLP_C_ZU_POS	74	SF_GetParam函數
SLP_C_ZL_EN	75	SF_GetParam函數
SLP_C_ZL_POS	76	SF_GetParam函數
SLP_J2_MON_RAD	77	SF_GetParam函數
SLP_J3_MON_RAD	78	SF_GetParam函數
SLP_J5_MON_RAD	79	SF_GetParam函數
SLP_J6_MON_RAD	80	SF_GetParam函數
SLP_J1_RANGE_MAX	81	SF_GetParam函數
SLP_J1_RANGE_MIN	82	SF_GetParam函數
SLP_J2_RANGE_MAX	83	SF_GetParam函數

常數名稱	值	用法
SLP_J2_RANGE_MIN	84	SF_GetParam函數
SLP_J3_RANGE_MAX	85	SF_GetParam函數
SLP_J3_RANGE_MIN	86	SF_GetParam函數
SLP_J4_RANGE_MAX	87	SF_GetParam函數
SLP_J4_RANGE_MIN	88	SF_GetParam函數
SLP_J5_RANGE_MAX	89	SF_GetParam函數
SLP_J5_RANGE_MIN	90	SF_GetParam函數
SLP_J6_RANGE_MAX	91	SF_GetParam函數
SLP_J6_RANGE_MIN	92	SF_GetParam函數
SIN_1_SLS_1_EN	93	SF_GetParam函數
SIN_1_SLS_2_EN	94	SF_GetParam函數
SIN_1_SLS_3_EN	95	SF_GetParam函數
SIN_1_SLP_A_EN	96	SF_GetParam函數
SIN_1_SLP_B_EN	97	SF_GetParam函數
SIN_1_SLP_C_EN	98	SF_GetParam函數
SIN_1_SG_EN	99	SF_GetParam函數
SIN_1_ESTOP_EN	100	SF_GetParam函數
SIN_2_SLS_1_EN	101	SF_GetParam函數
SIN_2_SLS_2_EN	102	SF_GetParam函數
SIN_2_SLS_3_EN	103	SF_GetParam函數
SIN_2_SLP_A_EN	104	SF_GetParam函數
SIN_2_SLP_B_EN	105	SF_GetParam函數
SIN_2_SLP_C_EN	106	SF_GetParam函數
SIN_2_SG_EN	107	SF_GetParam函數
SIN_2_ESTOP_EN	108	SF_GetParam函數
SIN_3_SLS_1_EN	109	SF_GetParam函數
SIN_3_SLS_2_EN	110	SF_GetParam函數
SIN_3_SLS_3_EN	111	SF_GetParam函數
SIN_3_SLP_A_EN	112	SF_GetParam函數
SIN_3_SLP_B_EN	113	SF_GetParam函數
SIN_3_SLP_C_EN	114	SF_GetParam函數

常數名稱	值	用法
SIN_3_SG_EN	115	SF_GetParam函數
SIN_3_ESTOP_EN	116	SF_GetParam函數
SIN_4_SLS_1_EN	117	SF_GetParam函數
SIN_4_SLS_2_EN	118	SF_GetParam函數
SIN_4_SLS_3_EN	119	SF_GetParam函數
SIN_4_SLP_A_EN	120	SF_GetParam函數
SIN_4_SLP_B_EN	121	SF_GetParam函數
SIN_4_SLP_C_EN	122	SF_GetParam函數
SIN_4_SG_EN	123	SF_GetParam函數
SIN_4_ESTOP_EN	124	SF_GetParam函數
SIN_5_SLS_1_EN	125	SF_GetParam函數
SIN_5_SLS_2_EN	126	SF_GetParam函數
SIN_5_SLS_3_EN	127	SF_GetParam函數
SIN_5_SLP_A_EN	128	SF_GetParam函數
SIN_5_SLP_B_EN	129	SF_GetParam函數
SIN_5_SLP_C_EN	130	SF_GetParam函數
SIN_5_SG_EN	131	SF_GetParam函數
SIN_5_ESTOP_EN	132	SF_GetParam函數
SOUT_1_STO	133	SF_GetParam函數
SOUT_1_SLS_1	134	SF_GetParam函數
SOUT_1_SLS_2	135	SF_GetParam函數
SOUT_1_SLS_3	136	SF_GetParam函數
SOUT_1_SLS_T2	137	SF_GetParam函數
SOUT_1_SLS_T	138	SF_GetParam函數
SOUT_1_SLP_A	139	SF_GetParam函數
SOUT_1_SLP_B	140	SF_GetParam函數
SOUT_1_SLP_C	141	SF_GetParam函數
SOUT_1_EP_RC	142	SF_GetParam函數
SOUT_1_EP_TP	143	SF_GetParam函數
SOUT_1_EN_SW	144	SF_GetParam函數
SOUT_2_STO	145	SF_GetParam函數

常數名稱	值	用法
SOUT_2_SLS_1	146	SF_GetParam函數
SOUT_2_SLS_2	147	SF_GetParam函數
SOUT_2_SLS_3	148	SF_GetParam函數
SOUT_2_SLS_T2	149	SF_GetParam函數
SOUT_2_SLS_T	150	SF_GetParam函數
SOUT_2_SLP_A	151	SF_GetParam函數
SOUT_2_SLP_B	152	SF_GetParam函數
SOUT_2_SLP_C	153	SF_GetParam函數
SOUT_2_EP_RC	154	SF_GetParam函數
SOUT_2_EP_TP	155	SF_GetParam函數
SOUT_2_EN_SW	156	SF_GetParam函數
SOUT_3_STO	157	SF_GetParam函數
SOUT_3_SLS_1	158	SF_GetParam函數
SOUT_3_SLS_2	159	SF_GetParam函數
SOUT_3_SLS_3	160	SF_GetParam函數
SOUT_3_SLS_T2	161	SF_GetParam函數
SOUT_3_SLS_T	162	SF_GetParam函數
SOUT_3_SLP_A	163	SF_GetParam函數
SOUT_3_SLP_B	164	SF_GetParam函數
SOUT_3_SLP_C	165	SF_GetParam函數
SOUT_3_EP_RC	166	SF_GetParam函數
SOUT_3_EP_TP	167	SF_GetParam函數
SOUT_3_EN_SW	168	SF_GetParam函數
POS_ROT_U	169	SF_GetParam函數
POS_ROT_V	170	SF_GetParam函數
POS_ROT_W	171	SF_GetParam函數
POS_OFS_X	172	SF_GetParam函數
POS_OFS_Y	173	SF_GetParam函數
POS_OFS_Z	174	SF_GetParam函數
SF_TOOLVERSION	1	SF_GetParam$函數
SF_CHECKSUM	2	SF_GetParam$函數

常數名稱	值	用法
SF_LAST_MODIFIED	3	SF_GetParam$函數
SF_ROBOT_MODEL_NAME	4	SF_GetParam$函數
SF_ROBOT_CHECKSUM	5	SF_GetParam$函數
SF_HOFS	6	SF_GetParam$函數
SF_HOFS_LAST_MODIFIED	7	SF_GetParam$函數
SLS_1	1	SF_LimitSpeedS, SF_LimitSpeedSEnable
SLS_2	2	SF_LimitSpeedS, SF_LimitSpeedSEnable
SLS_3	3	SF_LimitSpeedS, SF_LimitSpeedSEnable
SLS_T	9	SF_LimitSpeedS, SF_LimitSpeedSEnable
SLS_T2	10	SF_LimitSpeedS, SF_LimitSpeedSEnable