SIEMENS

SINUMERIK

SINUMERIK ONE
PLC Basic Program plus

功能手册

前言 1

基本安全说明 2

概述 3

PLC 项目准备工作 4

编程流程 5

快速入门 6

SINUMERIK 用户报警 7

NC 事件和过程报警 8

从 PLC 访问 NC 变量 9

配置/组态 10

编译和加载块 11

针对 PLC Basic Program Plus 的参考 12

缩写列表 A

适用于
SINUMERIK ONE
SINUMERIK CNC 软件，版本 6.21
SINUMERIK STEP 7 Toolbox V18
SINUMERIK PLC Basic Program plus V2.0

01/2023
A5E51148763F AC

法律资讯

警告提示系统

为了您的人身安全以及避免财产损失，必须注意本手册中的提示。人身安全的提示用一个警告三角表示，仅与财产损失有关的提示不带警告三角。警告提示根据危险等级由高到低如下表示。

⚠ 危险
表示如果不采取相应的小心措施，**将会**导致死亡或者严重的人身伤害。

⚠ 警告
表示如果不采取相应的小心措施，**可能**导致死亡或者严重的人身伤害。

⚠ 小心
表示如果不采取相应的小心措施，可能导致轻微的人身伤害。

注意
表示如果不采取相应的小心措施，可能导致财产损失。

当出现多个危险等级的情况下，每次总是使用最高等级的警告提示。如果在某个警告提示中带有警告可能导致人身伤害的警告三角，则可能在该警告提示中另外还附带有可能导致财产损失的警告。

合格的专业人员

本文件所属的产品/系统只允许由符合各项工作要求的**合格人员**进行操作。其操作必须遵照各自附带的文件说明，特别是其中的安全及警告提示。 由于具备相关培训及经验，合格人员可以察觉本产品/系统的风险，并避免可能的危险。

按规定使用 Siemens 产品

请注意下列说明：

⚠ 警告
Siemens 产品只允许用于目录和相关技术文件中规定的使用情况。如果要使用其他公司的产品和组件，必须得到 Siemens 推荐和允许。正确的运输、储存、组装、装配、安装、调试、操作和维护是产品安全、正常运行的前提。必须保证允许的环境条件。必须注意相关文件中的提示。

商标

所有带有标记符号 ® 的都是 Siemens AG 的注册商标。本印刷品中的其他符号可能是一些其他商标。若第三方出于自身目的使用这些商标，将侵害其所有者的权利。

责任免除

我们已对印刷品中所述内容与硬件和软件的一致性作过检查。然而不排除存在偏差的可能性，因此我们不保证印刷品中所述内容与硬件和软件完全一致。印刷品中的数据都按规定经过检测，必要的修正值包含在下一版本中。

Siemens AG
Digital Industries
Postfach 48 48
90026 NÜRNBERG
德国

A5E51148763F AC
Ⓟ 01/2023 本公司保留更改的权利

Copyright © Siemens AG 2021 - 2023.
保留所有权利

目录

1 前言 33

1.1 关于 SINUMERIK 33

1.2 关于本手册 33

1.3 网上文档 34

1.3.1 SINUMERIK ONE 文档一览 34

1.3.2 SINUMERIK 操作组件文档一览 35

1.4 技术文档反馈 35

1.5 mySupport 文档 35

1.6 服务与支持 36

1.7 OpenSSL 的使用 38

1.8 遵守通用数据保护条例 38

2 基本安全说明 39

2.1 一般安全说明 39

2.2 应用示例的质保规定 39

2.3 安全性信息 39

3 概述 41

3.1 概述 41

4 PLC 项目准备工作 43

4.1 说明 43

4.2 基本程序准备工作 43

4.3 基本程序的使用 44

4.4 嵌入 PLC 基本程序块 45

4.5 复制程序块时的冲突 48

4.6 嵌入 Basic Program plus 的 PLC 数据类型 48

4.7 嵌入 Basic Program 的 PLC 数据类型 50

4.8 嵌入 PLC 指令 52

4.9 指令可用性和 NCK 相关数据类型 53

4.10 TIA Portal 项目的升级 55

4.11 块和库操作 56

5 编程流程 57
5.1 顺序结构 57
5.2 数据类型、访问功能和程序运行 58
5.3 PLC 基本程序的掉电保持行为 59
6 快速入门 61
6.1 概览 61
6.2 创建对象实例的数据块 61
6.3 为与 HMI 交换数据创建数据块 63
6.4 在启动过程中初始化和注册 64
6.5 循环交换数据 66
6.6 以太网机床控制面板和 HT 10 通讯组态 68
6.7 NCK 信号的传输和应答时间 71
6.8 编程技巧与建议 72
7 SINUMERIK 用户报警 75
7.1 消息文本 75
7.1.1 通过 TIA 管理消息文本 77
7.1.2 在“*.ts”文件中管理消息文本 82
7.2 报警类别（消息类别） 82
7.3 应答 83
7.4 重新初始化 83
7.5 消息文本中的关联值 84
7.5.1 报警编号“alarmNumber” 84
7.5.2 报警索引“alarmIndex” 85
7.5.3 关联值“associatedValue” 85
7.5.4 关联值概述 85
7.5.5 关联值的字符串数据类型 87
7.6 时间戳 88
7.7 报警计数器 88
7.8 规模 89
7.9 定义 SINUMERIK 用户报警 89
7.10 掉电保持 89
8 NC 事件和过程报警 91
8.1 从用户程序访问 NC 信号 91

8.2 定义 NCK 过程报警事件 92
9 从 PLC 访问 NC 变量 95
9.1 变量组态概述 95
9.2 在 TIA 博途中组态 NC 变量 95
9.2.1 相关对象和资源 95
9.2.2 使用 NC 变量表工作 97
9.2.2.1 创建和管理用户自定义变量表 97
9.2.2.2 管理变量表中的条目 99
9.2.2.3 调整变量表视图 100
9.2.3 使用包含可用 NC 变量的列表工作 100
9.2.3.1 搜索和筛选 NC 变量 100
9.2.3.2 将 NC 变量传输到“组态的 NC 变量”变量表中 102
9.2.4 组态选择的 NC 变量实例 104
9.2.5 嵌入 PLC 程序中 107
9.2.5.1 在 PLC 程序中读/写 NC 变量 107
9.2.5.2 使用 NC 变量表创建数据块 108
9.2.5.3 将组态的 NC 变量复制到现有 DB/FB 中 110
9.2.5.4 NCK 和 PLC 的数据类型声明对应关系 111
9.2.6 管理库中的 NC 变量 114
10 配置/组态 115
10.1 出现 NC 故障时的处理方式 115
10.2 机床控制面板和操作组件 115
10.2.1 一般信息 115
10.2.2 通过 PROFINET/PROFIBUS 联结 115
10.2.3 通过工业以太网联结 116
10.3 来自/到达操作软件 HMI 的控制信号 117
10.3.1 概述 117
10.3.2 使用 SINUMERIK Operate 时的信号传输 119
10.3.3 机床坐标系 (MCS) 或工件坐标系 (WCS) 中的轴实际值显示 119
10.3.4 运行方式组和增量值的选择 120
10.3.5 通道中的程序控制 123
10.3.6 选择加工轴/几何轴（通过 HMI 选择 JOG 轴） 125
10.3.7 选择手轮 127
10.3.8 干涉保护 129
11 编译和加载块 133
11.1 使用其他设备类型中的专有技术保护程序块/库 133
11.2 激活块的模拟能力 133
12 针对 PLC Basic Program Plus 的参考 135
12.1 引言 135

12.2 NCK 相关的数据类型（简要描述） 135
12.2.1 概览 135
12.2.2 NCK_typeAxis 136
12.2.3 NCK_typeChan 182
12.2.4 NCK_typeModeGroup 306
12.2.5 NCK_typeNC 312
12.2.6 NCK_typeToolholder 388
12.2.7 NCK_typeRelocate 407
12.2.8 NCK_typeLoadingPoint 413
12.2.9 NCK_typeToaUnit 419
12.2.10 NCK_typeHmi 420
12.2.11 NCK_typeHmiAxis 429
12.2.12 NCK_typeHmiChan 430
12.2.13 NCK_typeHmiModeGroup 435
12.2.14 NCK_typeHmiNc 437
12.3 NCK 相关的数据类型（详细说明） 439
12.3.1 概览 439
12.3.2 NCK_typeAxis 440
12.3.3 <Axis>.basic.in 441
12.3.3.1 actToolBasePos 441
12.3.3.2 alarmActive 442
12.3.3.3 clampingToleranceExceeded 443
12.3.3.4 contTraversActive 443
12.3.3.5 disabled 444
12.3.3.6 dynBacklashCompActive 445
12.3.3.7 emergencyRetractionActive 445
12.3.3.8 enc1Active 445
12.3.3.9 enc1FreqLimitExceeded 446
12.3.3.10 enc1Restored 447
12.3.3.11 enc1Synchronized 448
12.3.3.12 enc2Active 449
12.3.3.13 enc2FreqLimitExceeded 450
12.3.3.14 enc2Restored 451
12.3.3.15 enc2Synchronized 451
12.3.3.16 fixedStopForceLimited 452
12.3.3.17 fixedStopReached 453
12.3.3.18 fixedStopRequest 454
12.3.3.19 followUpModeActive 454
12.3.3.20 handwheelAssignedNumber 456
12.3.3.21 handwheelDirInverted 456
12.3.3.22 handwheelOvrActive 457
12.3.3.23 inc1Active 458
12.3.3.24 inc10Active 459
12.3.3.25 inc100Active 460

12.3.3.26 inc1000Active 461
12.3.3.27 inc10000Active 463
12.3.3.28 incVarActive 464
12.3.3.29 indexAxisInPos 465
12.3.3.30 isPathAxis 466
12.3.3.31 isPosAxis 467
12.3.3.32 isStationary 468
12.3.3.33 jogFixedPointReachedNumber 468
12.3.3.34 jogFixedPointSelectedNumber 469
12.3.3.35 jogPosReached 470
12.3.3.36 jogToPosActive 471
12.3.3.37 lubricationPulseActive 472
12.3.3.38 measurementActive 472
12.3.3.39 moduloLimitSwitchActive 473
12.3.3.40 posAxisFeedrateValue 473
12.3.3.41 posCoarseReached 474
12.3.3.42 posCtrlActive 475
12.3.3.43 posFineReached 476
12.3.3.44 ready 477
12.3.3.45 reposDelayAck 478
12.3.3.46 reposOffsetActive 480
12.3.3.47 reposOffsetValid 481
12.3.3.48 revolutionalFeedrateActive 483
12.3.3.49 rotAxisInPos 484
12.3.3.50 speedCtrlActive 484
12.3.3.51 spindleModeActive 485
12.3.3.52 swCamsActive 486
12.3.3.53 traversMinusCommandActive 486
12.3.3.54 traversMinusRequestActive 487
12.3.3.55 traversPlusCommandActive 489
12.3.3.56 traversPlusRequestActive 490
12.3.3.57 velocityReduced 491
12.3.4 <Axis>.basic.out 491
12.3.4.1 acceptExternalWorkOffset 491
12.3.4.2 clampingInProgress 492
12.3.4.3 contTravers 493
12.3.4.4 disable 493
12.3.4.5 disableParameterSetSwitchover 496
12.3.4.6 disableTraversKeys 497
12.3.4.7 dynBacklashComp 497
12.3.4.8 enableCtrl 498
12.3.4.9 enableModuloLimitSwitch 500
12.3.4.10 enableOvr 501
12.3.4.11 enc1Activation 502
12.3.4.12 enc2Activation 504

12.3.4.13 feedrateOvrFactor 507
12.3.4.14 feedStopSpindleStop 510
12.3.4.15 fixedFeedrateSelection 511
12.3.4.16 fixedStopAckReached 513
12.3.4.17 fixedStopEnable 514
12.3.4.18 fixedStopSensor 515
12.3.4.19 followUpMode 516
12.3.4.20 handwheelInvertDir 518
12.3.4.21 handwheelSelection 519
12.3.4.22 hwLimitSwitchMinus 520
12.3.4.23 hwLimitSwitchPlus 520
12.3.4.24 inc1 521
12.3.4.25 inc10 522
12.3.4.26 inc100 523
12.3.4.27 inc1000 523
12.3.4.28 inc10000 523
12.3.4.29 incVar 523
12.3.4.30 jogFixedPointSelection 523
12.3.4.31 jogToPos 524
12.3.4.32 limitSpeed 525
12.3.4.33 posCtrlParameterSetSelection 526
12.3.4.34 progTestEnableCtrl 527
12.3.4.35 progTestRequest 527
12.3.4.36 progTestSuppression 528
12.3.4.37 rapidTraversOvr 529
12.3.4.38 refPointApproachCam 530
12.3.4.39 refPointSelection 530
12.3.4.40 reposDelay 531
12.3.4.41 reposPtp 533
12.3.4.42 resetMovement 533
12.3.4.43 swCams 535
12.3.4.44 swLimitSwitch2Minus 536
12.3.4.45 swLimitSwitch2Plus 536
12.3.4.46 traversKeyMinus 537
12.3.4.47 traversKeyPlus 538
12.3.5 <Axis>.drive.in 540
12.3.5.1 autonomousMotionActive 540
12.3.5.2 classCAlarmActive 541
12.3.5.3 currentCtrlActive 541
12.3.5.4 driveCtrlActive 542
12.3.5.5 driveDataSetNumber 543
12.3.5.6 driveTestTraversRequested 544
12.3.5.7 enabled 544
12.3.5.8 esrDcVoltageAlarmActive 545
12.3.5.9 esrGeneratorOperationActive 545

12.3.5.10 esrSpeedBelowMin 545
12.3.5.11 heatSinkTemperatureWarning 546
12.3.5.12 holdingBrakeReleased 547
12.3.5.13 motorDataSetNumber 547
12.3.5.14 motorTemperatureWarning 548
12.3.5.15 numberOfDriveDataSets 549
12.3.5.16 numberOfDriveDataSetsValid 550
12.3.5.17 posCtrlParameterSetNumber 551
12.3.5.18 pulsesEnabled 551
12.3.5.19 rapidStopActive 552
12.3.5.20 ready 553
12.3.5.21 speedAtSetpoint 554
12.3.5.22 speedBelowMin 554
12.3.5.23 speedBelowThreshold 555
12.3.5.24 speedCtrlIntegratorDisabled 556
12.3.5.25 speedSetpointReached 556
12.3.5.26 torqueBelowThreshold 557
12.3.5.27 varSignalingFunc1Active 558
12.3.5.28 varSignalingFunc2Active 559
12.3.5.29 varSignalingFunc3Active 560
12.3.6 <Axis>.drive.out 561
12.3.6.1 disableSpeedCtrlIntegrator 561
12.3.6.2 driveDataSetSelection 561
12.3.6.3 enableDriveTestTravers 563
12.3.6.4 enablePulses 563
12.3.6.5 motorChangeoverDone 564
12.3.6.6 motorDataSetSelection 565
12.3.6.7 rapidStop 566
12.3.6.8 releaseHoldingBrake 567
12.3.6.9 requestDriveCtrl 568
12.3.7 <Axis>.interchange.in 568
12.3.7.1 assignedChanNumber 568
12.3.7.2 assignedToPlc 569
12.3.7.3 neutralAxis 570
12.3.7.4 possible 571
12.3.7.5 requested 571
12.3.8 <Axis>.interchange.out 572
12.3.8.1 assignToChanSelection 572
12.3.8.2 assignToPlc 573
12.3.8.3 requireSwitchOver 574
12.3.8.4 reset 575
12.3.9 <Axis>.spindle.in 577
12.3.9.1 constantCuttingSpeedActive 577
12.3.9.2 ctrlModeActive 577
12.3.9.3 cwRotActive 578

12.3.9.4 gearChangeoverRequested 579
12.3.9.5 inPos 580
12.3.9.6 mFuncNumber 581
12.3.9.7 oscillModeActive 582
12.3.9.8 posModeActive 582
12.3.9.9 requestedGearStageNumber 583
12.3.9.10 rigidTappingActive 584
12.3.9.11 sFuncValue 585
12.3.9.12 speedLimitationExceeded 586
12.3.9.13 speedSetpointAboveMax 587
12.3.9.14 speedSetpointBelowMin 589
12.3.9.15 speedSetpointReached 590
12.3.9.16 syncModeActive 591
12.3.9.17 toolWithDynLimitationActive 592
12.3.10 <Axis>.spindle.in.smi24 592
12.3.10.1 sensorS1Present 592
12.3.10.2 sensorS1StateNumber 593
12.3.10.3 sensorS1Value 594
12.3.10.4 sensorS4Actuated 595
12.3.10.5 sensorS4Present 596
12.3.10.6 sensorS5Actuated 596
12.3.10.7 sensorS5Present 597
12.3.10.8 sensorS6Present 598
12.3.10.9 sensorsPresent 598
12.3.10.10 speedLimited 599
12.3.11 <Axis>.spindle.out 599
12.3.11.1 autoGearStageChange 599
12.3.11.2 deleteSValue 600
12.3.11.3 disableSpeedMonitor 600
12.3.11.4 gearChangeoverDone 601
12.3.11.5 gearStageSelection 602
12.3.11.6 invertM3M4 603
12.3.11.7 oscillEnable 604
12.3.11.8 oscillRequest 605
12.3.11.9 oscillSetDirCcw 606
12.3.11.10 oscillSetDirCw 607
12.3.11.11 ovrFactor 608
12.3.11.12 resyncEnc1 611
12.3.11.13 resyncEnc1ForPos 612
12.3.11.14 resyncEnc2 612
12.3.11.15 resyncEnc2ForPos 613
12.3.11.16 startCcw 614
12.3.11.17 startCw 614
12.3.11.18 startPosOperation 615
12.3.11.19 stop 615

12.3.11.20 useFeedrateOvr 616
12.3.12 <Axis>.oscillation.in 616
12.3.12.1 active 616
12.3.12.2 disabled 617
12.3.12.3 error 617
12.3.12.4 motionActive 618
12.3.12.5 externalReversalActive 618
12.3.12.6 sparkingOutActive 619
12.3.13 <Axis>.oscillation.out 620
12.3.13.1 activateReversalPoint 620
12.3.13.2 externalReversal 621
12.3.13.3 selectReversalPoint 621
12.3.13.4 stopAtNextReversalPoint 622
12.3.14 <Axis>.safety.in 623
12.3.14.1 brakeTestModeEnabled 623
12.3.14.2 infoCtrlChanActive 623
12.3.15 <Axis>.safety.out 623
12.3.15.1 activateBrakeRampForSs2e 623
12.3.15.2 enableBrakeTestMode 624
12.3.15.3 setpointLimitSelection 624
12.3.16 <Axis>.safetyInfoChan.in 624
12.3.16.1 acceptanceTestModeSelected 624
12.3.16.2 acceptanceTestSlpSelected 625
12.3.16.3 activeBrake 625
12.3.16.4 brakeTest 626
12.3.16.5 brakeTestActive 626
12.3.16.6 brakeTestCompleted 626
12.3.16.7 brakeTestResult 627
12.3.16.8 currentLoadSign 627
12.3.16.9 esrRetractRequested 628
12.3.16.10 externalBrakeRequested 628
12.3.16.11 safetyFaultActive 628
12.3.16.12 safetyMessageActive 629
12.3.16.13 sdiNegativeSelected 629
12.3.16.14 sdiPositiveSelected 630
12.3.16.15 setpointInput 630
12.3.16.16 slaActive 630
12.3.16.17 slpRangeSelected 631
12.3.16.18 slpSelectedAndUserAgreement 631
12.3.16.19 slsActive 632
12.3.16.20 slsBit0Active 632
12.3.16.21 slsBit1Active 632
12.3.16.22 slsLimit 633
12.3.16.23 slsSelected 633
12.3.16.24 sosActive 634

12.3.16.25 sosSelected 634
12.3.16.26 ss1Active 634
12.3.16.27 ss2Active 635
12.3.16.28 ss2eActive 635
12.3.16.29 ss2esrActive 636
12.3.16.30 stoActive 636
12.3.16.31 testStopActive 636
12.3.16.32 testStopRequired 637
12.3.17 <Axis>.safetyCtrlChan.out 637
12.3.17.1 brakeSelection 637
12.3.17.2 closeBrake 638
12.3.17.3 externalBrakeClosed 638
12.3.17.4 selectBrakeTest 638
12.3.17.5 selectDirectionOfRotation 639
12.3.17.6 selectTestSequence 639
12.3.17.7 startBrakeTest 640
12.3.17.8 testStop 640
12.3.18 <Axis>.coupling.in 641
12.3.18.1 AccelReached 641
12.3.18.2 AccelThresholdReached 641
12.3.18.3 correctionActive 642
12.3.18.4 correctionDone 642
12.3.18.5 followingSpindleActive 643
12.3.18.6 gantryActive 644
12.3.18.7 gantryCutOffLimitExceeded 644
12.3.18.8 gantryLeadingActive 645
12.3.18.9 gantrySyncOk 645
12.3.18.10 gantrySyncRunReady 645
12.3.18.11 gantryWarningLimitExceeded 646
12.3.18.12 leadingSpindleActive 646
12.3.18.13 maxAccelReached 647
12.3.18.14 maxSpeedReached 647
12.3.18.15 mcsCouplingActive 648
12.3.18.16 mcsFollowingAxisModeActive 648
12.3.18.17 mcsMirroringActive 648
12.3.18.18 mcsOffsetChanged 649
12.3.18.19 motionOverlaid 649
12.3.18.20 msActive 650
12.3.18.21 msCompCtrlActive 651
12.3.18.22 msSpeedDifferenceCoarseReached 651
12.3.18.23 msSpeedDifferenceFineReached 652
12.3.18.24 processActive 653
12.3.18.25 SpeedThresholdReached 653
12.3.18.26 syncCoarse2Reached 654
12.3.18.27 syncCoarseReached 655

12.3.18.28 syncFine2Reached 656
12.3.18.29 syncFineReached 656
12.3.18.30 syncSpindleCtrlMode 657
12.3.19 <Axis>.coupling.out 658
12.3.19.1 enableFollowOverlay 658
12.3.19.2 gantryDisableSync 659
12.3.19.3 gantryStartSyncRun 659
12.3.19.4 mcsCollisionProtection 660
12.3.19.5 mcsDecouple 660
12.3.19.6 msActivate 660
12.3.19.7 msTorqueComp 661
12.3.19.8 syncCorrection 662
12.3.19.9 syncResetCorrection 663
12.3.19.10 syncSpindleResync 663
12.3.19.11 syncSpindleSyncDisable 664
12.3.20 <Axis>.plcCtrl.in 665
12.3.20.1 plcCtrlActive 665
12.3.20.2 plcCtrlPermanent 665
12.3.20.3 resetDone 666
12.3.20.4 stopped 667
12.3.21 <Axis>.plcCtrl.out 668
12.3.21.1 chanAssignmentSelection 668
12.3.21.2 requestPlcCtrl 669
12.3.21.3 resume 670
12.3.21.4 stop 671
12.3.22 NCK_typeChan 673
12.3.23 <Chan>.basic.in 675
12.3.23.1 allAxesReferenced 675
12.3.23.2 allAxesStationary 676
12.3.23.3 asupActive 676
12.3.23.4 asupSilentActive 677
12.3.23.5 asupStopStateActive 677
12.3.23.6 blockSearchActionActive 678
12.3.23.7 blockSearchActive 679
12.3.23.8 blockSearchApproachActive 680
12.3.23.9 blockSearchLastActionActive 680
12.3.23.10 blockSearchSeruproActive 681
12.3.23.11 circularJogTraversActive 682
12.3.23.12 clcActive 682
12.3.23.13 clcStopAtLowerLimitActive 683
12.3.23.14 clcStopAtUpperLimitActive 684
12.3.23.15 collisionStopActive 685
12.3.23.16 configuredStopAtBlockEnd 686
12.3.23.17 configuredStopFuncActive 686
12.3.23.18 contourHandwheelAssignedNumber 687

12.3.23.19 contourHandwheelDirInverted 688
12.3.23.20 disableReadInIgnored 688
12.3.23.21 dryRunFeedrateActive 689
12.3.23.22 executionFromExternalActive 690
12.3.23.23 externalProgLanguageModeActive 690
12.3.23.24 handwheelOvrActive 691
12.3.23.25 interruptProcessingActive 691
12.3.23.26 jogRetractActive 692
12.3.23.27 jogRetractDataAvailable 693
12.3.23.28 m00m01Active 693
12.3.23.29 m00m01AssociatedAuxFuncActive 694
12.3.23.30 m02m30Active 695
12.3.23.31 ncAlarmActive 696
12.3.23.32 ncAlarmWithProgStopActive 697
12.3.23.33 ncAlarmWithStopActive 698
12.3.23.34 orientableToolholderActive 699
12.3.23.35 overStoreActive 699
12.3.23.36 productionTargetReached 699
12.3.23.37 progStateAborted 700
12.3.23.38 progStateInterrupted 701
12.3.23.39 progStateRunning 703
12.3.23.40 progStateStopped 704
12.3.23.41 progStateWaiting 706
12.3.23.42 progTestSelected 707
12.3.23.43 protAreasNotGuaranteed 708
12.3.23.44 ptpTraversActive 708
12.3.23.45 ready 709
12.3.23.46 reducedDynamicsActive 710
12.3.23.47 refActive 710
12.3.23.48 reposDelayActive 711
12.3.23.49 reposModeChangeAck 712
12.3.23.50 reposModeSelectedNumber 714
12.3.23.51 retraceSupportActive 715
12.3.23.52 retraceSupportModeActive 716
12.3.23.53 revolutionalFeedrateActive 717
12.3.23.54 singleBlockStopAtBlockEnd 717
12.3.23.55 singleBlockStopIgnored 718
12.3.23.56 stateActive 719
12.3.23.57 stateInterrupted 720
12.3.23.58 stateReset 721
12.3.23.59 stopConditionActive 722
12.3.23.60 toffonActive 722
12.3.23.61 toffonMotionActive 723
12.3.23.62 transformationActive 724
12.3.23.63 transformationActiveNumber 725

12.3.24 <Chan>.basic.out 726
12.3.24.1 abortProgLevel 726
12.3.24.2 blockSearchAckPlcActionDone 726
12.3.24.3 circularJogTravers 728
12.3.24.4 clcOvr 729
12.3.24.5 configuredStopFunc 730
12.3.24.6 contourHandwheelInvertDir 730
12.3.24.7 contourHandwheelSelection 731
12.3.24.8 contourHandwheelSim 732
12.3.24.9 contourHandwheelSimInvertDir 732
12.3.24.10 deactivateClc 733
12.3.24.11 deleteDistanceToGo 733
12.3.24.12 deleteSubProgRepetition 734
12.3.24.13 disableFeed 735
12.3.24.14 disableNcStart 736
12.3.24.15 disableReadIn 737
12.3.24.16 dryRunFeedrate 737
12.3.24.17 enableGoToStart 738
12.3.24.18 enableProtAreas 739
12.3.24.19 geoAxesTraversBcsWcsSelection 740
12.3.24.20 handwheelOffsetDrf 740
12.3.24.21 lockAllSynActs 741
12.3.24.22 m01 742
12.3.24.23 m01AssociatedAuxFunc 742
12.3.24.24 ncStart 743
12.3.24.25 ncStop 744
12.3.24.26 ncStopAxesAndSpindles 745
12.3.24.27 ncStopBlockEnd 746
12.3.24.28 pathFeedrateLimit 747
12.3.24.29 pathFeedrateOvr 748
12.3.24.30 pathFeedrateOvrFactor 749
12.3.24.31 pathFixedFeedrateSelection 751
12.3.24.32 pathRapidTraversOvr 752
12.3.24.33 pathRapidTraversOvrFactor 753
12.3.24.34 progTest 755
12.3.24.35 ptpTravers 757
12.3.24.36 refPoint 757
12.3.24.37 reposModeActivationRequest 758
12.3.24.38 reposModeSelection 760
12.3.24.39 reset 761
12.3.24.40 retraceSupportStart 762
12.3.24.41 retraceSupportTraversBackwards 763
12.3.24.42 singleBlock 763
12.3.24.43 skipBlockLevel0..9 763
12.3.24.44 suppressDisableNcStart 764

12.3.25 <Chan>.gCommandGroup.in 765
12.3.25.1 activeValues 765
12.3.25.2 g00Active 766
12.3.26 <Chan>.geoAxis1.in 766
12.3.26.1 contManualTraversActive 766
12.3.26.2 handwheelAssignedNumber 767
12.3.26.3 handwheelDirInverted 768
12.3.26.4 inc1Active 769
12.3.26.5 inc10Active 770
12.3.26.6 inc100Active 771
12.3.26.7 inc1000Active 772
12.3.26.8 inc10000Active 773
12.3.26.9 incVarActive 774
12.3.26.10 traversMinusCommand 775
12.3.26.11 traversMinusRequest 776
12.3.26.12 traversPlusCommand 777
12.3.26.13 traversPlusRequest 778
12.3.27 <Chan>.geoAxis1.out 779
12.3.27.1 contManualTravers 779
12.3.27.2 disableFeed 780
12.3.27.3 disableTraversKey 781
12.3.27.4 handwheelInvertDir 781
12.3.27.5 handwheelSelection 782
12.3.27.6 inc1 783
12.3.27.7 inc10 784
12.3.27.8 inc100 786
12.3.27.9 inc1000 787
12.3.27.10 inc10000 789
12.3.27.11 incVar 790
12.3.27.12 rapidTraversOvr 792
12.3.27.13 traversKeyMinus 793
12.3.27.14 traversKeyPlus 794
12.3.28 <Chan>.orientationAxis1.in 796
12.3.28.1 contManualTraversActive 796
12.3.28.2 handwheelAssignedNumber 797
12.3.28.3 handwheelDirInverted 798
12.3.28.4 inc1Active 798
12.3.28.5 inc10Active 799
12.3.28.6 inc100Active 800
12.3.28.7 inc1000Active 801
12.3.28.8 inc10000Active 802
12.3.28.9 incVarActive 803
12.3.28.10 traversMinusRequest 804
12.3.28.11 traversPlusRequest 805
12.3.28.12 traversMinusCommand 806

12.3.28.13 traversPlusCommand 807
12.3.29 <Chan>.orientationAxis1.out 808
12.3.29.1 contManualTravers 808
12.3.29.2 disableFeed 809
12.3.29.3 disableTraversKey 810
12.3.29.4 handwheelSelection 810
12.3.29.5 handwheelInvertDir 811
12.3.29.6 inc1 812
12.3.29.7 inc10 813
12.3.29.8 inc100 815
12.3.29.9 inc1000 816
12.3.29.10 inc10000 818
12.3.29.11 incVar 819
12.3.29.12 rapidTraversOvr 821
12.3.29.13 traversKeyMinus 822
12.3.29.14 traversKeyPlus 823
12.3.30 <Chan>.toolMgmt.in 825
12.3.30.1 lastReplacementToolReached 825
12.3.30.2 lastTNumber 826
12.3.30.3 newReplacementToolActive 826
12.3.30.4 newTNumber 827
12.3.30.5 toolLimitTNumber 827
12.3.30.6 toolLimitValueReached 828
12.3.30.7 toolMissing 829
12.3.30.8 toolPrewarnLimitReached 829
12.3.30.9 toolPrewarnLimitTNumber 830
12.3.31 <Chan>.toolMgmt.out 830
12.3.31.1 deactivateWearMonitoring 830
12.3.31.2 deactivateWorkpieceCounter 831
12.3.31.3 ignoreDisabledSate 832
12.3.31.4 toolLifeMonitoring 833
12.3.32 <Chan>.auxMDynFuncs.in 834
12.3.32.1 m00Decoded - m99Decoded 834
12.3.33 <Chan>.auxMFuncs[x].in 835
12.3.33.1 extendedAddress 835
12.3.33.2 fastAckActive 836
12.3.33.3 hasChanged 836
12.3.33.4 notDecoded 837
12.3.33.5 value 837
12.3.34 <Chan>.auxSFuncs[x].in 838
12.3.34.1 extendedAddress 838
12.3.34.2 fastAckActive 839
12.3.34.3 hasChanged 839
12.3.34.4 value 840
12.3.35 <Chan>.auxTFuncs[x].in 841

12.3.35.1 extendedAddress 841
12.3.35.2 fastAckActive 841
12.3.35.3 hasChanged 842
12.3.35.4 value 842
12.3.36 <Chan>.auxDFuncs[x].in 844
12.3.36.1 extendedAddress 844
12.3.36.2 fastAckActive 844
12.3.36.3 hasChanged 845
12.3.36.4 value 845
12.3.37 <Chan>.auxDLFuncs[x].in 846
12.3.37.1 extendedAddress 846
12.3.37.2 fastAckActive 847
12.3.37.3 hasChanged 847
12.3.37.4 value 848
12.3.38 <Chan>.auxHFuncs[x].in 848
12.3.38.1 extendedAddress 848
12.3.38.2 fastAckActive 849
12.3.38.3 hasChanged 849
12.3.38.4 value 850
12.3.38.5 valueTypeDint 851
12.3.39 <Chan>.auxFFuncs[x].in 852
12.3.39.1 extendedAddress 852
12.3.39.2 fastAckActive 853
12.3.39.3 hasChanged 853
12.3.39.4 value 853
12.3.40 <Chan>.protAreas[x].in 855
12.3.40.1 chanPreactivated 855
12.3.40.2 chanViolated 856
12.3.40.3 ncPreactivated 857
12.3.40.4 ncViolated 858
12.3.41 <Chan>.protAreas[x].out 859
12.3.41.1 chan 859
12.3.41.2 nc 860
12.3.42 <Chan>.progEvent.in 861
12.3.42.1 blockSearchActive 861
12.3.42.2 ncStartActive 862
12.3.42.3 partProgramEndActive 862
12.3.42.4 powerOnActive 863
12.3.42.5 resetActive 863
12.3.43 <Chan>.synActs[x].in 864
12.3.43.1 lockable 864
12.3.44 <Chan>.synActs[x].out 865
12.3.44.1 lock 865
12.3.45 <Chan>.grinding.in 865
12.3.45.1 requested 865

12.3.45.2 signalsEnabled 866
12.3.46 <Chan>.grinding.out 866
12.3.46.1 ackFuncsExecution 866
12.3.46.2 disableSignals 867
12.3.46.3 set 867
12.3.47 <Chan>.nibbelingAndPunching.in 868
12.3.47.1 strokeManualInitiationAck 868
12.3.47.2 strokeInitiationActive 868
12.3.48 <Chan>.nibbelingAndPunching.out 869
12.3.48.1 delayStroke 869
12.3.48.2 enableStroke 870
12.3.48.3 manualStrokeInterface1 870
12.3.48.4 manualStrokeInterface2 871
12.3.48.5 noStroke 872
12.3.48.6 suppressStroke 873
12.3.49 <Chan>.cycles.in 873
12.3.49.1 signals 873
12.3.50 <Chan>.cycles.out 874
12.3.50.1 signals 874
12.3.51 NCK_typeModeGroup 874
12.3.52 <ModeGroup>.basic.in 875
12.3.52.1 autoModeActive 875
12.3.52.2 contTraversActive 875
12.3.52.3 inc1Active 876
12.3.52.4 inc10Active 877
12.3.52.5 inc100Active 879
12.3.52.6 inc1000Active 880
12.3.52.7 inc10000Active 882
12.3.52.8 incVarActive 883
12.3.52.9 jogModeActive 885
12.3.52.10 mdaModeActive 886
12.3.52.11 nckInternalJogActive 886
12.3.52.12 ready 887
12.3.52.13 refActive 888
12.3.52.14 reposActive 888
12.3.52.15 resetAllChanDone 889
12.3.52.16 resetDone 889
12.3.52.17 teachInActive 890
12.3.53 <ModeGroup>.basic.out 891
12.3.53.1 autoMode 891
12.3.53.2 contTravers 892
12.3.53.3 disableModeChangeover 893
12.3.53.4 inc1 894
12.3.53.5 inc10 895
12.3.53.6 inc100 896

12.3.53.7 inc1000 898
12.3.53.8 inc10000 899
12.3.53.9 incVar 901
12.3.53.10 jogMode 902
12.3.53.11 mdaMode 903
12.3.53.12 refPoint 904
12.3.53.13 repos 904
12.3.53.14 reset 905
12.3.53.15 singleBlockTypeA 906
12.3.53.16 singleBlockTypeB 907
12.3.53.17 stopAll 908
12.3.53.18 stopAxesOnly 910
12.3.53.19 teachIn 911
12.3.54 NCK_typeNc 912
12.3.55 <Nc>.basic.in 913
12.3.55.1 collisionMonitoringActive 913
12.3.55.2 drivesCyclicOperation 914
12.3.55.3 drivesReady 914
12.3.55.4 emergencyStopActive 915
12.3.55.5 imperialMetricChangeCounter 915
12.3.55.6 imperialUnitsActive 916
12.3.55.7 measuringProbe1Actuated 916
12.3.55.8 measuringProbe2Actuated 917
12.3.55.9 ncAlarmActive 917
12.3.55.10 ncReady 918
12.3.55.11 ncuAirTemperatureAlarmActive 919
12.3.55.12 ncuHeatSinkAlarmActive 920
12.3.55.13 sinuOperateReady 920
12.3.55.14 swCams[1..32].minus 920
12.3.55.15 swCams[1..32].plus 921
12.3.56 <Nc>.basic.out 922
12.3.56.1 emergencyStop 922
12.3.56.2 emergencyStopAck 923
12.3.56.3 enableReducedDynamics 924
12.3.56.4 keyOperatedSwitchPos0 924
12.3.56.5 keyOperatedSwitchPos1 925
12.3.56.6 keyOperatedSwitchPos2 926
12.3.56.7 keyOperatedSwitchPos3 926
12.3.56.8 selectIncFromModeGroup 927
12.3.56.9 transferActToolBasePos 928
12.3.57 <Nc>.handwheels[x].in 928
12.3.57.1 movedCounter 928
12.3.57.2 stationary 929
12.3.58 <Nc>.digitalInputs[x].in 929
12.3.58.1 active 929

12.3.59 <Nc>.digitalInputs[x].out 930
12.3.59.1 lock 930
12.3.59.2 set 931
12.3.60 <Nc>.digitalOutputs[x].in 932
12.3.60.1 setpointState 932
12.3.61 <Nc>.digitalOutputs[x].out 933
12.3.61.1 lock 933
12.3.61.2 overwrite 934
12.3.61.3 settingState 935
12.3.61.4 target 936
12.3.62 <Nc>.analogInputs[x].in 937
12.3.62.1 actualValue 937
12.3.63 <Nc>.analogInputs[x].out 938
12.3.63.1 lock 938
12.3.63.2 target 939
12.3.63.3 settingValue 940
12.3.64 <Nc>.analogOutputs[x].in 941
12.3.64.1 setpointValue 941
12.3.65 <Nc>.analogOutputs[x].out 942
12.3.65.1 settingValue 942
12.3.65.2 lock 943
12.3.65.3 overwrite 944
12.3.65.4 target 945
12.3.66 <Nc>.collisionProtAreas.in 946
12.3.66.1 active 946
12.3.67 <Nc>.collisionProtAreas.out 948
12.3.67.1 activate 948
12.3.68 <Nc>.collision.out 949
12.3.68.1 deactivateFixtureAreasAuto 949
12.3.68.2 deactivateFixtureAreasJog 950
12.3.68.3 deactivateMachineAreasAuto 951
12.3.68.4 deactivateMachineAreasJog 953
12.3.68.5 deactivateToolAreasAuto 954
12.3.68.6 deactivateToolAreasJog 955
12.3.68.7 deactivateWorkpieceAreasAuto 956
12.3.68.8 deactivateWorkpieceAreasJog 957
12.3.69 NCK_typeHmi 958
12.3.70 <Hmi>.basic.in 959
12.3.70.1 activeOperatingAreaNumber 959
12.3.70.2 cancelKeyPressed 960
12.3.70.3 currentChanNumber 960
12.3.70.4 currentScreenNumber 960
12.3.70.5 dataTransferActive 961
12.3.70.6 directKeyAddress 961
12.3.70.7 inputDeviceActive 962

12.3.70.8 mcpAddress....962
12.3.70.9 mcsWcsSwitched....962
12.3.70.10 screenChangeActive....963
12.3.70.11 screenDark....963
12.3.70.12 simActive....964
12.3.70.13 switchoverTcuDisabled....964
12.3.70.14 tcuIndex....965
12.3.71 <hmi>.basic.inOut....965
12.3.71.1 cancelAlarmsCleared....965
12.3.71.2 plcHardkeySelection....965
12.3.71.3 recallAlarmsCleared....966
12.3.72 <Hmi>.basic.out....966
12.3.72.1 analogSpindle1Percentage....966
12.3.72.2 analogSpindle2Percentage....967
12.3.72.3 brightenScreen....967
12.3.72.4 clearCancelAlarms....967
12.3.72.5 clearRecallAlarms....968
12.3.72.6 darkenScreen....968
12.3.72.7 disableKeys....969
12.3.72.8 disableTeachInTransfer....970
12.3.72.9 displayWcsValues....971
12.3.72.10 permitRemoteMonitoringOnly....971
12.3.72.11 rejectRemoteAccess....972
12.3.72.12 saveActionLog....972
12.3.73 <Hmi>.plcFunc.in....973
12.3.73.1 busy....973
12.3.73.2 errorCode....973
12.3.74 <Hmi>.plcFunc.out....974
12.3.74.1 parameter1....974
12.3.74.2 selection....975
12.3.74.3 requestStrobe....976
12.3.75 <Hmi>.progSelection.in....976
12.3.75.1 completed....976
12.3.75.2 identified....977
12.3.75.3 error....978
12.3.75.4 errorCode....980
12.3.75.5 selected....981
12.3.76 <Hmi>.progSelection.out....982
12.3.76.1 progList....982
12.3.76.2 progNumber....983
12.3.76.3 request....985
12.3.77 <Hmi>.mappingTable.in....985
12.3.77.1 error....985
12.3.77.2 ready....985
12.3.77.3 status....986

12.3.78 <Hmi>.mappingTable.out 986
12.3.78.1 numberOfTableLines 986
12.3.78.2 status 986
12.3.78.3 tableLines 986
12.3.78.4 version 987
12.3.79 <Hmi>.userKeys.in 987
12.3.79.1 pressed 987
12.3.80 <Hmi>.userKeys.out 987
12.3.80.1 activateLed 987
12.3.80.2 disable 987
12.3.81 NCK_typeHmiAxis 988
12.3.82 <HmiAxis>.basic.in 988
12.3.82.1 handwheelSelected 988
12.3.82.2 progTestRequested 989
12.3.82.3 progTestSuppressionRequested 990
12.3.83 NCK_typeHmiChan 991
12.3.84 <HmiChan>.basic.in 992
12.3.84.1 atStopSetOvrZeroSelected 992
12.3.84.2 configuredStopFuncSelected 992
12.3.84.3 contourHandwheelSelected 993
12.3.84.4 dryRunSelected 994
12.3.84.5 handwheelOffsetDrfSelected 995
12.3.84.6 m01AssociatedAuxFuncSelected 996
12.3.84.7 m01Selected 996
12.3.84.8 progTestSelected 997
12.3.84.9 rapidTraversOvrSelected 998
12.3.84.10 reposModeSelectedNumber 999
12.3.84.11 reposModeChanged 1000
12.3.84.12 skipBlockLevel0...9 1001
12.3.85 <HmiChan>.axisSelection.in 1001
12.3.85.1 selectedKeyActuated 1001
12.3.85.2 selectedNumber 1002
12.3.85.3 selectedType 1003
12.3.86 <HmiChan>.axisSelection.out 1004
12.3.86.1 activeNumber 1004
12.3.86.2 activeType 1005
12.3.87 <HmiChan>.geoAxis.in 1006
12.3.87.1 handwheelSelected 1006
12.3.88 <HmiChan>.orientationAxis.in 1007
12.3.88.1 handwheelSelected 1007
12.3.89 NCK_typeHmiModeGroup 1008
12.3.90 <HmiModeGroup>.basic.in 1008
12.3.90.1 autoModeRequested 1008
12.3.90.2 contTraversRequested 1009
12.3.90.3 mdaModeRequested 1009

12.3.90.4 inc1Requested 1010
12.3.90.5 inc10Requested 1011
12.3.90.6 inc100Requested 1012
12.3.90.7 inc1000Requested 1013
12.3.90.8 inc10000Requested 1014
12.3.90.9 incVarRequested 1015
12.3.90.10 jogModeRequested 1016
12.3.90.11 reposRequested 1017
12.3.90.12 refRequested 1017
12.3.90.13 teachInRequested 1018
12.3.91 <HmiModeGroup>basic.out 1019
12.3.91.1 disableModeChangeover 1019
12.3.92 NCK_typeHmiNC 1019
12.3.93 <HmiNc>.collision.in 1019
12.3.93.1 collision.in 1019
12.3.94 NCK_typeToaUnit 1022
12.3.95 NCK_typeLoadingPoint 1023
12.3.96 <LoadingPoint>.config 1024
12.3.96.1 loadingLocation 1024
12.3.96.2 loadingMagazine 1024
12.3.97 <LoadingPoint>.ack.in 1025
12.3.97.1 busy 1025
12.3.97.2 done 1025
12.3.97.3 error 1025
12.3.97.4 status 1026
12.3.98 <LoadingPoint>.ack.out 1026
12.3.98.1 execute 1026
12.3.98.2 multitoolLocation 1026
12.3.98.3 status 1026
12.3.98.4 toolLocation 1027
12.3.98.5 toolMagazine 1027
12.3.98.6 totalComplete 1027
12.3.98.7 totalFailed 1028
12.3.99 <LoadingPoint>.command.in 1028
12.3.99.1 multitoolRequest 1028
12.3.99.2 request 1028
12.3.99.3 userParameters 1029
12.3.100 <LoadingPoint>.command.in.basic 1029
12.3.100.1 chan 1029
12.3.100.2 load 1029
12.3.100.3 position 1030
12.3.100.4 toolSourceLocation 1030
12.3.100.5 toolSourceMagazine 1030
12.3.100.6 toolTargetLocation 1031
12.3.100.7 toolTargetMagazine 1031

12.3.100.8 unload 1031
12.3.101 <LoadingPoint>.command.in.options 1032
12.3.101.1 byNcProg 1032
12.3.102 <LoadingPoint>.command.in.toolProperties 1032
12.3.102.1 toolLocationType 1032
12.3.102.2 toolSizeBottom 1032
12.3.102.3 toolSizeLeft 1032
12.3.102.4 toolSizeRight 1033
12.3.102.5 toolSizeTop 1033
12.3.103 <LoadingPoint>.command.in.toolStatus 1033
12.3.103.1 active 1033
12.3.103.2 changeActive 1034
12.3.103.3 changeOneToOne 1034
12.3.103.4 disabled 1034
12.3.103.5 disableStateIgnored 1035
12.3.103.6 enabled 1035
12.3.103.7 fixedLocation 1035
12.3.103.8 manualTool 1036
12.3.103.9 masterTool 1036
12.3.103.10 measured 1036
12.3.103.11 prewarning 1037
12.3.103.12 toBeLoaded 1037
12.3.103.13 toBeUnloaded 1037
12.3.103.14 toolAtDisabledLocation 1038
12.3.103.15 toolInBuffer 1038
12.3.103.16 used 1038
12.3.104 <LoadingPoint>.command.in.multitoolProperties 1039
12.3.104.1 distance 1039
12.3.104.2 distanceType 1039
12.3.104.3 numberOfLocations 1039
12.3.105 NCK_typeRelocate 1040
12.3.106 <Relocate>.ack.in 1040
12.3.106.1 busy 1040
12.3.106.2 完成 1041
12.3.106.3 error 1041
12.3.106.4 status 1041
12.3.107 <Relocate>.ack.out 1041
12.3.107.1 execute 1041

12.3.107.2 status 1042
12.3.107.3 toolLocation 1042
12.3.107.4 toolMagazine 1042
12.3.107.5 totalComplete 1043
12.3.107.6 totalFailed 1043
12.3.108 <Relocate>.command.in 1043
12.3.108.1 multitoolRequest 1043
12.3.108.2 request 1044
12.3.108.3 userParameters 1044
12.3.109 <Relocate>.command.in.basic 1044
12.3.109.1 chan 1044
12.3.109.2 relocate 1045
12.3.109.3 tool 1045
12.3.109.4 toolSourceLocation 1045
12.3.109.5 toolSourceMagazine 1045
12.3.109.6 toolTargetLocation 1046
12.3.109.7 toolTargetMagazine 1046
12.3.110 <Relocate>.command.in.options 1046
12.3.110.1 byNcProg 1046
12.3.111 <Relocate>.command.in.toolProperties 1047
12.3.111.1 toolLocationType 1047
12.3.111.2 toolSizeBottom 1047
12.3.111.3 toolSizeLeft 1047
12.3.111.4 toolSizeRight 1048
12.3.111.5 toolSizeTop 1048
12.3.112 <Relocate>.command.in.toolStatus 1048
12.3.112.1 active 1048
12.3.112.2 changeActive 1049
12.3.112.3 changeOneToOne 1049
12.3.112.4 disabled 1049
12.3.112.5 disableStateIgnored 1050
12.3.112.6 enabled 1050
12.3.112.7 fixedLocation 1050
12.3.112.8 manualTool 1051
12.3.112.9 masterTool 1051
12.3.112.10 measured 1051
12.3.112.11 prewarning 1052
12.3.112.12 toBeLoaded 1052
12.3.112.13 toBeUnloaded 1052
12.3.112.14 toolAtDisabledLocation 1053

12.3.112.15 toolInBuffer 1053
12.3.112.16 used 1053
12.3.113 <Relocate>.command.in.multitoolProperties 1054
12.3.113.1 distance 1054
12.3.113.2 distanceType 1054
12.3.113.3 numberOfLocations 1054
12.3.114 NCK_typeToolholder 1055
12.3.115 <Toolholder>.config 1056
12.3.115.1 bufferLocation 1056
12.3.115.2 bufferMagazine 1056
12.3.116 <Toolholder>.ack.in 1057
12.3.116.1 busy 1057
12.3.116.2 完成 1057
12.3.116.3 error 1058
12.3.116.4 status 1058
12.3.117 <Toolholder>.ack.out 1058
12.3.117.1 execute 1058
12.3.117.2 newMultitoolLocation 1058
12.3.117.3 newToolLocation 1059
12.3.117.4 newToolMagazine 1059
12.3.117.5 oldToolLocation 1059
12.3.117.6 oldToolMagazine 1059
12.3.117.7 status 1060
12.3.117.8 totalComplete 1060
12.3.117.9 totalFailed 1061
12.3.118 <Toolholder>.command.in 1061
12.3.118.1 multitoolRequest 1061
12.3.118.2 request 1061
12.3.118.3 userParameters 1061
12.3.119 <Toolholder>.command.in.basic 1062
12.3.119.1 chan 1062
12.3.119.2 newTool 1062
12.3.119.3 newToolSourceLocation 1062
12.3.119.4 newToolSourceMagazine 1062
12.3.119.5 newToolTargetLocation 1063
12.3.119.6 newToolTargetMagazine 1063
12.3.119.7 oldToolSourceLocation 1063
12.3.119.8 oldToolSourceMagazine 1063
12.3.119.9 oldToolTargetLocation 1064
12.3.119.10 oldToolTargetMagazine 1064
12.3.119.11 perform 1064

12.3.119.1 position 1064
12.3.119.1 prepare 1065
12.3.120 <Toolholder>.command.in.options 1065
12.3.120.1 byNcProg 1065
12.3.120.2 newManualTool 1065
12.3.120.3 noNewTool 1065
12.3.120.4 oldManualTool 1066
12.3.120.5 oldToolNotInToolholder 1066
12.3.120.6 toolRemainsInToolholder 1066
12.3.121 <Toolholder>.command.in.toolProperties 1066
12.3.121.1 toolLocationType 1066
12.3.121.2 toolOriginLocation 1067
12.3.121.3 toolOriginMagazine 1067
12.3.121.4 toolSizeBottom 1067
12.3.121.5 toolSizeLeft 1067
12.3.121.6 toolSizeRight 1068
12.3.121.7 toolSizeTop 1068
12.3.122 <Toolholder>.command.in.toolStatus 1068
12.3.122.1 active 1068
12.3.122.2 changeActive 1069
12.3.122.3 changeOneToOne 1069
12.3.122.4 disabled 1069
12.3.122.5 disableStateIgnored 1070
12.3.122.6 enabled 1070
12.3.122.7 fixedLocation 1070
12.3.122.8 manualTool 1071
12.3.122.9 masterTool 1071
12.3.122.10 measured 1071
12.3.122.11 prewarning 1072
12.3.122.12 toBeLoaded 1072
12.3.122.13 toBeUnloaded 1072
12.3.122.14 toolAtDisabledLocation 1073
12.3.122.15 toolInBuffer 1073
12.3.122.16 used 1073
12.3.123 <Toolholder>.command.in.multitoolCommand 1074
12.3.123.1 newTool 1074

12.3.123.2 newToolLocation 1074
12.3.123.3 oldTool 1074
12.3.123.4 oldToolLocation 1075
12.3.123.5 position 1075
12.3.124 <Toolholder>.command.in.multitoolProperties 1075
12.3.124.1 distance 1075
12.3.124.2 distanceType 1075
12.3.124.3 numberOfLocations 1076
12.3.124.4 toolLocationType 1076
12.3.124.5 toolOriginLocation 1076
12.3.124.6 toolOriginMagazine 1076
12.3.124.7 toolSizeBottom 1077
12.3.124.8 toolSizeLeft 1077
12.3.124.9 toolSizeRight 1077
12.3.124.10 toolSizeTop 1077
12.3.125 <Toolholder>.command.in.multitoolStatus 1078
12.3.125.1 active 1078
12.3.125.2 changeActive 1078
12.3.125.3 changeOneToOne 1078
12.3.125.4 disabled 1079
12.3.125.5 disableStateIgnored 1079
12.3.125.6 enabled 1079
12.3.125.7 fixedLocation 1080
12.3.125.8 manualTool 1080
12.3.125.9 masterTool 1080
12.3.125.10 measured 1081
12.3.125.11 prewarning 1081
12.3.125.12 toBeLoaded 1081
12.3.125.13 toBeUnloaded 1082
12.3.125.14 toolAtDisabledLocation 1082
12.3.125.15 toolInBuffer 1082
12.3.125.16 used 1083

12.4 PLC 指令 SINU_* 1083
12.4.1 概览 1083
12.4.2 SINU_Init 1087
12.4.3 SINU_Update 1089

12.4.4 SINU_CompleteRegistration 1090
12.4.5 SINU_ProcessNckEvent 1091
12.4.6 SINU_AsyncTransfer 1092
12.4.7 SINU_RequestAsup 1094
12.4.8 SINU_PublishUserProgramVersion 1096
12.4.9 Registration 1098
12.4.9.1 SINU_RegisterNc 1098
12.4.9.2 SINU_RegisterNcWithHmi 1099
12.4.9.3 SINU_ReRegisterNc 1101
12.4.9.4 SINU_RegisterModeGroup 1102
12.4.9.5 SINU_RegisterModeGroupWithHmi 1103
12.4.9.6 SINU_ReRegisterModeGroup 1105
12.4.9.7 SINU_RegisterChan 1106
12.4.9.8 SINU_RegisterChanWithHmi 1107
12.4.9.9 SINU_ReRegisterChan 1109
12.4.9.10 SINU_RegisterAxis 1110
12.4.9.11 SINU_RegisterAxisWithHmi 1111
12.4.9.12 SINU_RegisterAxisByInst 1113
12.4.9.13 SINU_RegisterAxisByInstWithHmi 1115
12.4.9.14 SINU_RegisterAxisByName 1117
12.4.9.15 SINU_RegisterAxisByNameWithHmi 1119
12.4.9.16 SINU_ReRegisterAxis 1120
12.4.9.17 SINU_RegisterToaUnit 1121
12.4.9.18 SINU_RegisterLoadingPoint 1123
12.4.9.19 SINU_RegisterToolholder 1124
12.4.9.20 SINU_RegisterRelocate 1126
12.4.9.21 SINU_ReRegisterLoadingPoint 1127
12.4.9.22 SINU_ReRegisterToolholder 1128
12.4.9.23 SINU_ReRegisterRelocate 1129
12.4.10 NC commands 1130
12.4.10.1 SINU_NcLogin 1130
12.4.10.2 SINU_NcLogout 1133
12.4.10.3 SINU_NcCancelAlarms 1134
12.4.10.4 SINU_NcReset 1136
12.4.10.5 SINU_NcReconfigureMachineData 1138
12.4.10.6 SINU_NcAssignInterrupt 1140
12.4.10.7 SINU_NcSelectPartProgram 1144
12.4.10.8 SINU_NcActivateBlockSearch 1148
12.4.10.9 SINU_NcSetUserDataActive 1150
12.4.10.10 SINU_NcSetSemaphore 1152
12.4.10.11 SINU_NcResetMonitoringValues 1157
12.4.10.12 SINU_NcCreateTool 1159
12.4.10.13 SINU_NcCreateToolByName 1161
12.4.10.14 SINU_NcGetToolNumber 1164
12.4.10.15 SINU_NcSearchTool 1167

12.4.10.16 SINU_NcRetractTool 1171
12.4.10.17 SINU_NcMoveTool 1173
12.4.10.18 SINU_NcResetActiveToolStatus 1177
12.4.10.19 SINU_NcDeleteTool 1179
12.4.10.20 SINU_NcCreateMultiTool 1181
12.4.10.21 SINU_NcPositionMultitool 1184
12.4.10.22 SINU_NcSearchMultitoolLocation 1187
12.4.10.23 SINU_NcDeleteMultiTool 1190
12.4.10.24 SINU_NcCreateCuttingEdge 1193
12.4.10.25 SINU_NcCreateCuttingEdgeWithD 1195
12.4.10.26 SINU_NcDeleteCuttingEdge 1197
12.4.10.27 SINU_NcPositionMagazine 1199
12.4.10.28 SINU_NcSearchLocation 1202
12.4.10.29 SINU_NcSearchLoadingLocation 1206
12.4.10.30 SINU_NcSetWorkpieceCounterInc 1209
12.4.10.31 SINU_NcSetSemaphoreByToa 1212
12.4.11 轴与主轴功能 1216
12.4.11.1 SINU_TraversePosAxis 1216
12.4.11.2 SINU_TraverseIndexingAxis 1223
12.4.11.3 SINU_OscillateSpindle 1230
12.4.11.4 SINU_PositionSpindle 1236
12.4.11.5 SINU_RotateSpindle 1241
12.4.11.6 SINU_CtrlStarDelta 1247
12.4.11.7 中断轴的运动，删除剩余行程 1251
12.4.12 NC 数据传输 1252
12.4.12.1 SINU_ReadUserDataInt 1252
12.4.12.2 SINU_ReadUserDataHex 1254
12.4.12.3 SINU_ReadUserDataFloat 1256
12.4.12.4 SINU_ReadCompileCycleData 1259
12.4.12.5 SINU_WriteCompileCycleData 1261
12.4.13 操作单元 1263
12.4.13.1 SINU_WriteRecordOverEthernet 1263
12.4.13.2 SINU_UpdateEthernetMcp 1271
12.4.13.3 SINU_TransferHandwheelData 1279

12.5 PLC 数据类型：全局库 1282
12.5.1 概览 1282
12.5.2 SINU_typeHandwheelData 1282

12.6 PLC 块 LSINU_* 全局库 1283
12.6.1 概览 1283
12.6.2 LSINU_GenerateSinuUserAlarm 1283
12.6.3 LSINU_GetReinitSinuUserAlarms 1289
12.6.4 LSINU_GetSinuUserAlarmCounters 1291
12.6.5 LSINU_Powerride 1294
12.6.5.1 LED 色板 1299

12.6.5.2 示例：为通过机床控制面板集成 Powerride 倍率开关做好准备 1299
12.6.5.3 示例：调用 LSINU_Powerride 1301
12.6.6 LSINU_PowerrideConfig 1304
12.6.7 LSINU_ReadGUD 1305
12.6.8 LSINU_ReadVar 1313
12.6.8.1 LSINU_ReadVar 1313
12.6.8.2 生成用于访问 NC 变量的数据块 1321
12.6.9 LSINU_TransferSelData 1322
12.6.9.1 功能 3、4：快速数据交换 PLC-NC 1323
12.6.10 LSINU_WriteVar 1331

12.7 PLC 数据类型：全局库 1339
12.7.1 概览 1339
12.7.2 LSINU_typeAlarmInternalData 1340
12.7.3 LSINU_typePowerrideConfig 1340
12.7.4 LSINU_typePowerrideConfigPos 1342
12.7.5 LSINU_typePowerrideInputs 1343
12.7.6 LSINU_typePowerrideOutputs 1344

12.8 错误代码和状态代码 1345

A 缩写列表 1349

A.1 缩略语列表 1349

A.2 块、数据类型、指令和变量缩写 1355

索引 1361

前言 1

1.1 关于 SINUMERIK

无论是普及型数控机床，还是标准型机床，或者是模块化高端机床，SINUMERIK 数控系统都能为不同类型的机床提供最佳解决方案。无论是单件生产还是批量生产、简单工件还是复杂工件，对于从样品和工具制造、模具制造乃至大批量生产的所有制造领域而言，SINUMERIK 自始至终都是高生产率的自动化解决方案。

详细信息请访问网页 SINUMERIK (https://www.siemens.com/sinumerik)。

1.2 关于本手册

目标使用人群

本文档面向的对象是设计人员、编程人员和调试人员。

用途

文本档主要介绍 SINUMERIK ONE Basic Program plus (BPP) 这个基本程序。因此文档中的“基本程序”，都是指 SINUMERIK Basic Programm plus。如果在介绍 SINUMERIK ONE Basic Program (BP) 的章节提到基本程序这个概念，会补充 BP 字样。比如，使用 SINUMERIK ONE Basic Program 中的数据类型时。

本文档全面介绍使用 Basic Program plus 的 SINUMERIK ONE PLC 的各项功能和编程相关内容。

优点

基于本文档，目标使用人群将能够在将现有功能范畴考虑在内的情况下实现 SINUMERIK ONE PLC 项目。

标准功能范畴

本文档描述了标准功能范畴。该描述可能和交付的系统的功能有所不同。交付的系统的功能仅以订购资料为准。

在系统中也可能会运行本文档中未说明的功能，但这并不表示在交付系统时必须提供这些功能以及相关的维修服务。

为使文档简明清晰，本文档并不包含所有产品类型的所有详细信息，也无法对安装、运行和维护中可能出现的各种情况逐一进行说明。

机床制造商在产品上增添或者更改的功能，由机床制造商进行说明。

第三方网页

本文档可能包含第三方网页链接。西门子对此类网页的内容不承担任何责任，也不会声明或认可此类网页或其内容为西门子所有。西门子并不能控制此类网页上的信息，也不对上述网页的内容和信息负责。使用上述网页的风险由用户承担。

1.3 网上文档

1.3.1 SINUMERIK ONE 文档一览

有关 SINUMERIK ONE（自版本 6.13 起）功能的全部文档，请参见 SINUMERIK ONE 文档一览 (https://support.industry.siemens.com/cs/ww/en/view/109768483)。

您可以直接打开文档或者下载 PDF 和 HTML5 格式。

文档分为以下几个类别：

- 用户：操作
- 用户：编程
- 制造商/服务：功能
- 制造商/服务：硬件
- 制造商/服务：配置/调试

- 制造商/服务：Safety Integrated
- 介绍和培训
- 制造商/服务：SINAMICS

1.3.2 SINUMERIK 操作组件文档一览

有关 SINUMERIK 操作组件的全部文档，请参见 SINUMERIK 操作组件文档一览 (https://support.industry.siemens.com/cs/document/109783841/technische-dokumentation-zu-sinumerik-bedienkomponenten?dti=0&lc=en-WW)。

您可以直接打开文档或者下载 PDF 和 HTML5 格式。

文档分为以下几个类别：

- 操作面板
- 机床控制面板
- 机床按钮面板
- 手持单元/微型手持单元
- 其他操作组件

有关“SINUMERIK”的重要文档、文章和链接，请参见 SINUMERIK 专题页 (https://support.industry.siemens.com/cs/document/109766201/sinumerik-an-overview-of-the-most-important-documents-and-links?dti=0&lc=en-WW)。

1.4 技术文档反馈

对于西门子工业在线支持上发布的任何技术文档，如有疑问、建议或改进意见，请点击文章末尾的链接“发送反馈”。

1.5 mySupport 文档

使用网页版“mySupport 文档”可以自由组合西门子文档内容，创建自己的文档。

在 SiePortal 页面“mySupport 链接和工具” (https://support.industry.siemens.com/cs/cn/zh/my)上点击“我的文档”，便可启动应用：

配置的手册可以 RTF、PDF 或 XML 格式导出。

说明

在链接“配置”下可以查看网页版“mySupport 文档”支持的西门子文档内容。

1.6 服务与支持

产品支持

有关产品的详细信息请访问网址：

产品支持 (https://support.industry.siemens.com/cs/cn/zh/)

在该网址下可以提供：

- 最新产品信息（产品公告）
- FAQ（常见问题与解答）
- 手册
- 下载链接
- 持续提供产品最新信息的新闻。
- “技术论坛”，供全球用户和专家交流经验、分享信息
- “联系人”，提供全球联系人信息，方便查找本地联系人
- “售后服务”，提供现场服务、维修、备件等信息

技术支持

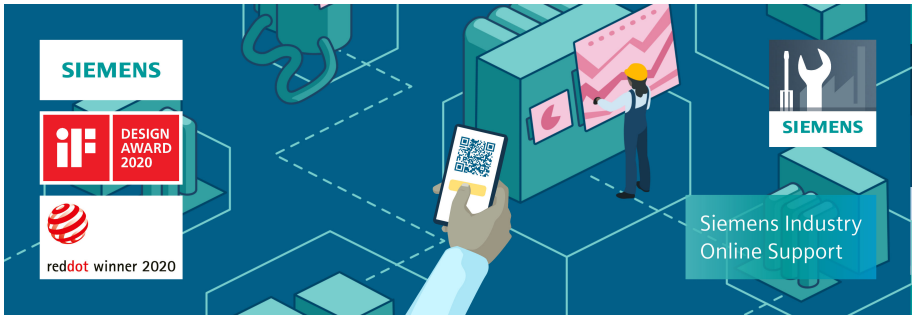
访问网址 (https://support.industry.siemens.com/cs/cn/zh/sc/4868)下的“联系方式”，便可以获取各个国家技术支持的电话号码。

如需咨询技术疑问，请使用“支持请求”一栏下的在线表格。

培训

访问网址 (https://www.siemens.com/sitrain)，可获取有关 SITRAIN 的相关信息。SITRAIN 为西门子的驱动和自动化产品、系统和解决方案提供培训。

无论在何处都能得到最佳的支持



使用荣获大奖的“西门子工业在线支持”App，您可以随时随地查看超过 30 万份的西门子工业领域的产品文件。该应用也可帮助您：

- 解决项目实施中出现的问题
- 排除故障
- 进行设备扩展或重新规划

此外，您还可以登录技术论坛，查看我们的专家为您撰写的其他文章：

- 常见问题与解答
- 应用实例
- 手册
- 证书
- 产品公告等

“西门子工业在线支持”App 提供 Apple iOS 版和安卓版。

铭牌上的二维码

铭牌上的二维码包含了各设备的数据。使用任一智能手机通过“西门子工业在线支持”App 扫描该二维码，便可获取相应设备的技术信息。

1.7 OpenSSL 的使用

本产品可包含以下软件：

- 由 OpenSSL 项目开发并应用在 OpenSSL Toolkit 中的软件
- 由 Eric Young 开发的加密软件
- 由 Eric Young 开发的软件

详细信息请访问网址：

- OpenSSL (https://www.openssl.org)
- Cryptsoft (https://www.cryptsoft.com)

1.8 遵守通用数据保护条例

西门子遵守通用数据保护条例，特别是隐私保护设计（privacy by design）规定。

对于本产品意味着：

产品不会处理或保存个人相关数据，只会处理或保存技术功能数据（例如：时间戳）。用户如果将此类技术功能数据与其他数据（例如：排班表）关联或者将个人相关数据存储在同一介质（例如：硬盘）上而产生个人相关性，则应由用户自行确保遵循数据保护法规。

基本安全说明 2

2.1 一般安全说明

> **⚠ 警告**
>
> **未遵循安全说明和遗留风险可引发生命危险**
>
> 忽视随附硬件文档中的安全说明和遗留风险会导致重伤或死亡。
> - 遵守硬件文档中的安全说明。
> - 进行风险评估时应考虑到遗留风险。

> **⚠ 警告**
>
> **因参数设置错误或修改参数设置引起机器故障**
>
> 参数设置错误可导致机器出现故障，从而导致人员重伤或死亡。
> - 采取保护措施，防止未经授权的参数设置。
> - 采取适当措施（如驻停或急停）处理可能出现的故障。

2.2 应用示例的质保规定

应用示例在组态和配置以及各种突发事件方面对设备没有强制约束力，无需一一遵循。应用示例不会提供客户专用的解决方案，仅在典型任务设置中提供保护。

用户自行负责上述产品的规范运行事宜。应用示例并没有解除您在应用、安装、运行和维护时确保安全环境的责任。

2.3 安全性信息

Siemens 为其产品及解决方案提供了工业信息安全功能，以支持工厂、系统、机器和网络的安全运行。

为了防止工厂、系统、机器和网络受到网络攻击，需要实施并持续维护先进且全面的工业信息安全保护机制。Siemens 的产品和解决方案构成此类概念的其中一个要素。

客户负责防止其工厂、系统、机器和网络受到未经授权的访问。只有在有必要连接时并仅在采取适当安全措施（例如，防火墙和/或网络分段）的情况下，才能将该等系统、机器和组件连接到企业网络或 Internet。

关于可采取的工业信息安全措施的更多信息，请访问 https://www.siemens.com/industrialsecurity

Siemens 不断对产品和解决方案进行开发和完善以提高安全性。Siemens 强烈建议您及时更新产品并始终使用最新产品版本。如果使用的产品版本不再受支持，或者未能应用最新的更新程序，客户遭受网络攻击的风险会增加。

要及时了解有关产品更新的信息，请订阅 Siemens 工业信息安全 RSS 源，网址为 https://www.siemens.com/cert

其他信息请上网查找：

工业安全功能选型手册 (https://support.industry.siemens.com/cs/cn/zh/view/108862708/en)

⚠ 警告

篡改软件会引起不安全的驱动状态

篡改软件（如：病毒、木马、蠕虫等）可使设备处于不安全的运行状态，从而可能导致死亡、重伤和财产损失。

- 总是使用最新版本的软件。
- 将自动化和驱动组件集成到设备或机器上的整套先进工业信息安全方案中。
- 全面考虑整套工业信息安全方案中使用的所有产品。
- 采取相应的保护措施（如：使用杀毒软件）防止移动存储设备中的文件受到恶意软件的破坏。
- 在调试结束后，检查所有和安全相关的设置。

概述 3

3.1 概述

PLC 基本程序

PLC 基本程序可在 TIA Portal 中提供一个包含块 (LSINU_*)、PLC 数据类型的系统库，以及包含指令 (SINU_*) 的系统库（全局库），用于在 NCK 和 PLC 用户程序之间循环交换数据。在基本程序 PLC Basic Program plus（用于通过 NCK 和 HMI 进行数据交换）中，未像以往一样使用固定的接口信号数据块进行数据交换，而是将接口信号作为 NCK 相关的数据类型 (NCK_type*) 进行提供。

系统库的块和 PLC 数据类型在“库”选项卡中显示为复制模板。NC 相关指令的系统库保存在“指令”选项卡的“选件包”部分。

您可在 TIA Portal 中创建设备、创建程序并将其加载到控制器中。但要使用 SINUMERIK 设备和软件模块，还必须在 TIA Portal V18（或更高版本）中安装下列 SINUMERIK Toolbox 工具箱软件。

- SINUMERIK STEP 7 Toolbox V18（或更高版本）

更多信息

更多关于使用库和指令工作的信息，请参阅 TIA Portal 在线技术支持界面的“使用库”和“PLC 编程 > 指令”部分内容。

PLC 项目准备工作

4

4.1 说明

基本程序支持在 NC 和 PLC 之间交换数据，参与 PLC 项目的所有机床都必须集成基本程序。基本程序块“LSINU_*”和指令“SINU_*”可提供多种功能。比如可以注册对象和请求升级等。

机床程序/用户程序

用户程序不但提供连接 NC 的接口，还包含逻辑链路和机床流程。机床制造商可以借助基本程序提供的块和指令创建用户程序。

4.2 基本程序准备工作

在 TIA Portal 中，按照下列操作步骤创建 PLC 程序：

- 添加硬件
- 将所需 PLC 基本程序块嵌入程序中
  建议：嵌入 SINUMERIK ONE PLC Basic Program plus 系统库中的所有块
- 将所需 PLC 数据类型嵌入项目中
  建议：嵌入 SINUMERIK ONE PLC Basic Program plus 系统库中的所有 PLC 数据类型
- 从 SINUMERIK ONE PLC Basic Programm V6.x 系统库嵌入所需的 PLC 数据类型
  使用经过调整的 SINUMERIK ONE PLC Basic Program plus 的块时，需要使用这些数据类型。

  说明

  SINUMERIK ONE PLC Basic Program plus 的哪些块需要这些数据类型，参见各块的相关说明。

- 创建 PLC 用户程序，编辑用户块和对指令进行编程
- 将 TIA Portal 项目载入 PLC

TIA Portal 的全局 SINUMERIK 系统库中，不但有用作复制模板的 SINUMERIK ONE PLC Basic Program plus 块和数据类型，还有包含各种指令的库。您可通过拖放操作，将所需的块拖动

至项目的“程序块”文件夹中。还可通过拖放，将“指令”选项卡中的指令拖动至块编辑器中使用。

说明

安装/升级

SINUMERIK ONE NCU 的 PLC 只能在 TIA Portal 中投入运行。

启动装有 SINUMERIK STEP 7 Toolbox 工具箱软件的 TIA Portal 并创建一个项目后，可在 TIA Portal 中使用 SINUMERIK ONE NCU 的各项功能。

4.3 基本程序的使用

可在一个 TIA Portal 项目中创建多个设备或机床，使用基本程序的不同指令进行组态。

备注

项目结构以及项目和用户程序的创建步骤，参见相关 TIA Portal 文档。

基本程序块操作步骤

在 TIA Portal 中通过系统库复制基本程序块。复制模板在“库”选项卡的“全局库”菜单项下。

在各子文件夹中，可以查看 PLC 基本程序的下列结构化组成部分：

文件夹	用途	内容
复制模板 > SINUMERIK ONE PLC Basic Program plus > SINUMERIK ONE PLC BP plus	要使用 PLC 基本程序 Basic Program plus 对 SINUMERIK ONE 进行编程，需要用到这个复制模板文件夹。	包含基本程序的所有块。
复制模板 > SINUMERIK ONE PLC Basic Program plus > SINUMERIK ONE PLC BP plus data types	该复制模板文件夹的内容由 PLC 基本程序使用，并且必须复制到 PLC 数据类型文件夹下。	包含 PLC 基本程序所使用的所有 PLC 数据类型。

块复制模板和 PLC 数据类型复制模板可以通过拖放操作，单个或与文件夹结构一起复制和嵌入到项目中。

指令操作步骤

在程序编辑器中使用时，再从“指令”选项卡复制 PLC 基本程序的专用 SINUMERIK 指令。

“指令”选项卡	用途	内容
选件包 > SINUMERIK ONE	使用 Basic Program plus 对 SINUMERIK ONE 进行编程时，需要用到这些指令。	包含 PLC 基本程序的专用 SINUMERIK 指令。

4.4 嵌入 PLC 基本程序块

前提条件

- 已添加 NCU/PPU 17x0。

步骤

按如下步骤，将基本程序块从复制模板中复制到项目导航的程序块文件夹下：

1. 切换至“库”选项卡，然后打开“SINUMERIK ONE PLC Basic Program plus”系统库
2. 将 PLC 基本程序块复制到项目中有多种方式：
   – 通过拖放将相应的“SINUMERIK ONE PLC BP plus”文件夹拖动至“程序块”文件夹下，例如：“CNC_1 > PLC_1 > 程序块”。

     说明

     复制此文件夹会改变用户自定义文件夹结构

     在复制“SINUMERIK ONE PLC BP plus”文件夹时，系统会在“程序块”文件夹下自动创建新的分组（子文件夹），并将现有程序块移动至此处。

     若已通过自有分组对 PLC 基本程序进行过组织，并希望保留该结构，则从下一级文件夹复制单独的程序块复制模板作为替代。

   – 打开复制模板文件夹以及可能有的子文件夹，手动选中要复制的程序块，并通过拖放将程序块拖动至程序块文件夹中。
     如要选中连续的多个对象，可以点击第一个对象，然后按住上档键并点击最后一个对象。
     如要选中不连续的多个对象，可以按住 Ctrl 键并依次点击这些对象。

程序块被复制到项目。此时，系统会检查是否有重复的块名称，如果有，会显示“复制时存在冲突 (页 48)”对话框。

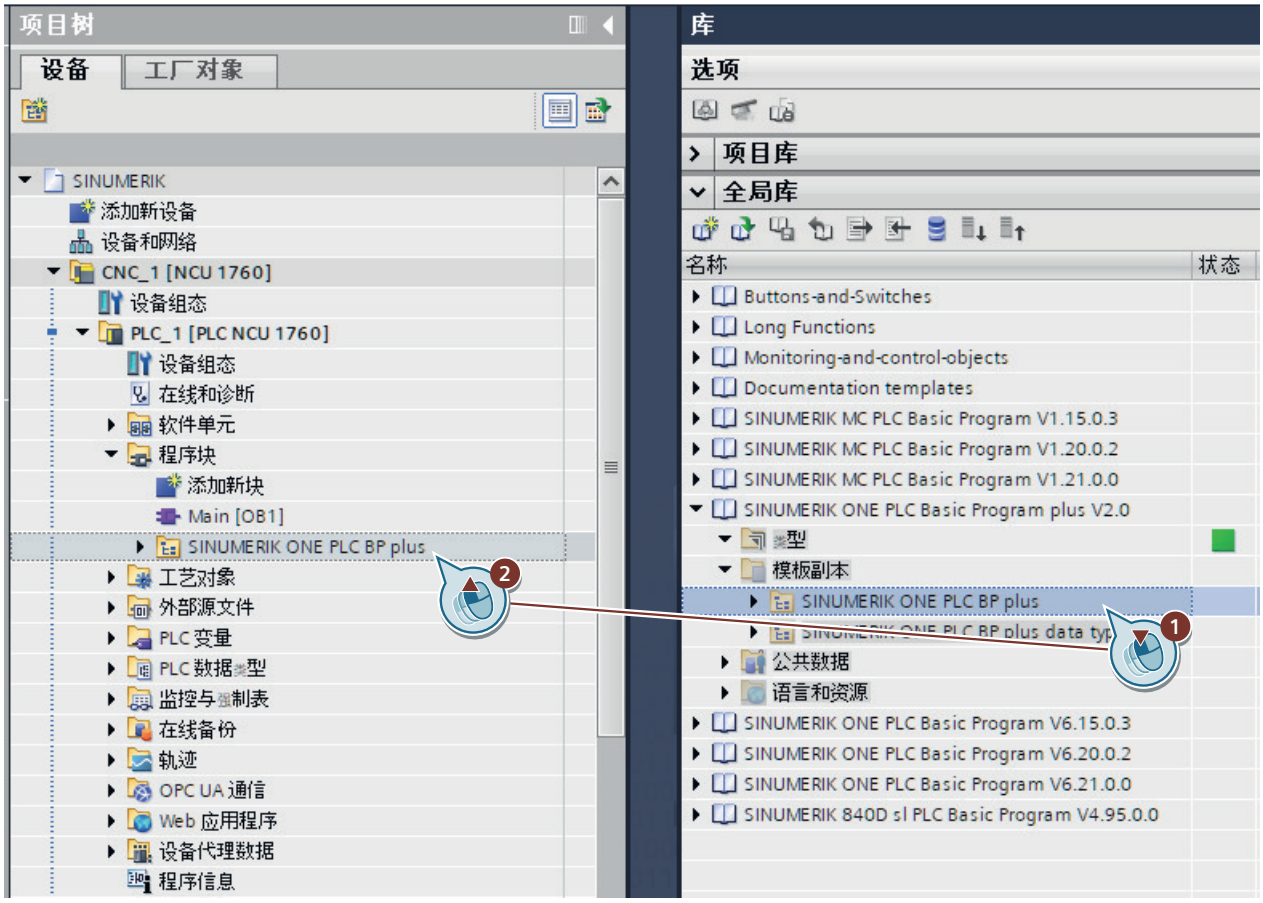


图 4-1　嵌入 PLC 基本程序块

结果

SINUMERIK PLC 基本程序块被复制到项目中。复制模板可能带有的文件夹也被复制。

此时可对所复制的程序块进行检查，手动删除不需要的 PLC 基本程序块。
若项目中存在不需要的程序块，会延长操作过程的持续时间，例如：加载、创建和读入 SINUMERIK 存档时。

4.5 复制程序块时的冲突

复制和嵌入程序块时，检查**块的符号名称是否重复**。如果发现重复使用符号名称引发冲突，系统会显示一条消息。

该对话框提供了以下几种排除冲突的操作选项：

- 取消复制操作，在项目导航区找到并删除已存在的程序块，然后重新进行复制。此时请在对话框中点击“取消”。
- 仍然将复制模板复制到项目中，并覆盖已存在的程序块，无需知道是哪些程序块导致了冲突。此时请在对话框中选择“新对象替换旧对象”并点击“OK”确认。

在这种情况下，对话框指出的另一个操作选项“重命名并插入对象”不能排除冲突。

4.6 嵌入 Basic Program plus 的 PLC 数据类型

基本程序的 PLC 接口数据块可实现从 PLC 程序访问警报/消息和 Powerride。

在 SINUMERIK ONE PLC 基本程序的系统库中提供了用于定义数据块内容的 PLC 数据类型。

步骤

按如下步骤，将 PLC 数据类型复制到项目中：

1. 切换至“库”选项卡，然后打开“SINUMERIK ONE PLC Basic Program plus”系统库
2. 通过拖放操作，将“复制模板 > SINUMERIK ONE PLC Basic Program plus > SINUMERIK ONE PLC BP plus data types”文件夹拖动至项目的 PLC 数据类型文件夹下（比如“CNC_1 > PLC_1 > PLC 数据类型”下）。

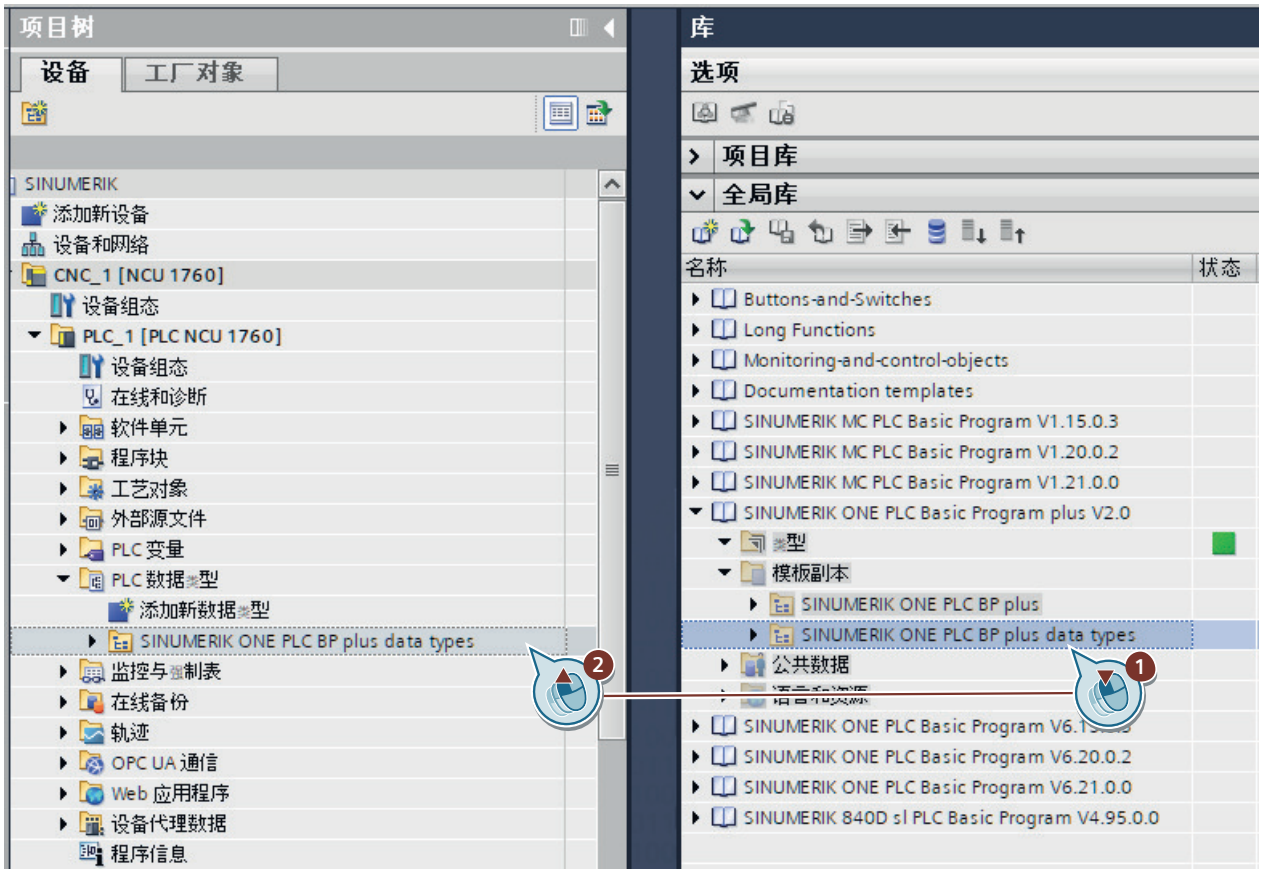


图 4-2　嵌入 PLC 数据类型

3. 在项目导航中右击“程序块”，然后在右键菜单“编译”中选择指令“软件（完全编译程序块）”。
   系统对 PLC 程序进行编译，此时会检查重复存在的程序块地址。

结果

数据类型已复制到项目中并分配给相应的 PLC 基本程序块。

说明

删除不使用的数据类型

检查文件夹“PLC 数据类型”中是否有未被程序块使用的数据类型。

4.7 嵌入 Basic Program 的 PLC 数据类型

“SINUMERIK ONE PLC Basic Programm V6.x”全局系统库提供使用 NC 变量和操作元件等时所需的 PLC 数据类型。这些数据类型必须额外嵌入“PLC 数据类型”文件夹。

步骤

按如下步骤，将 PLC 数据类型复制到项目中：

1. 切换到“库”选项卡，打开系统库“SINUMERIK ONE PLC Basic Program V6.x”。
2. 通过拖放操作，将“复制模板 > SINUMERIK ONE PLC Basic Program > SINUMERIK ONE PLC BP data types”中的下列文件夹拖动至项目的 PLC 数据类型文件夹下（比如“CNC_1 > PLC_1 > PLC 数据类型”下）。
   - 模块化 MCP 数据类型
   - NC 变量数据类型

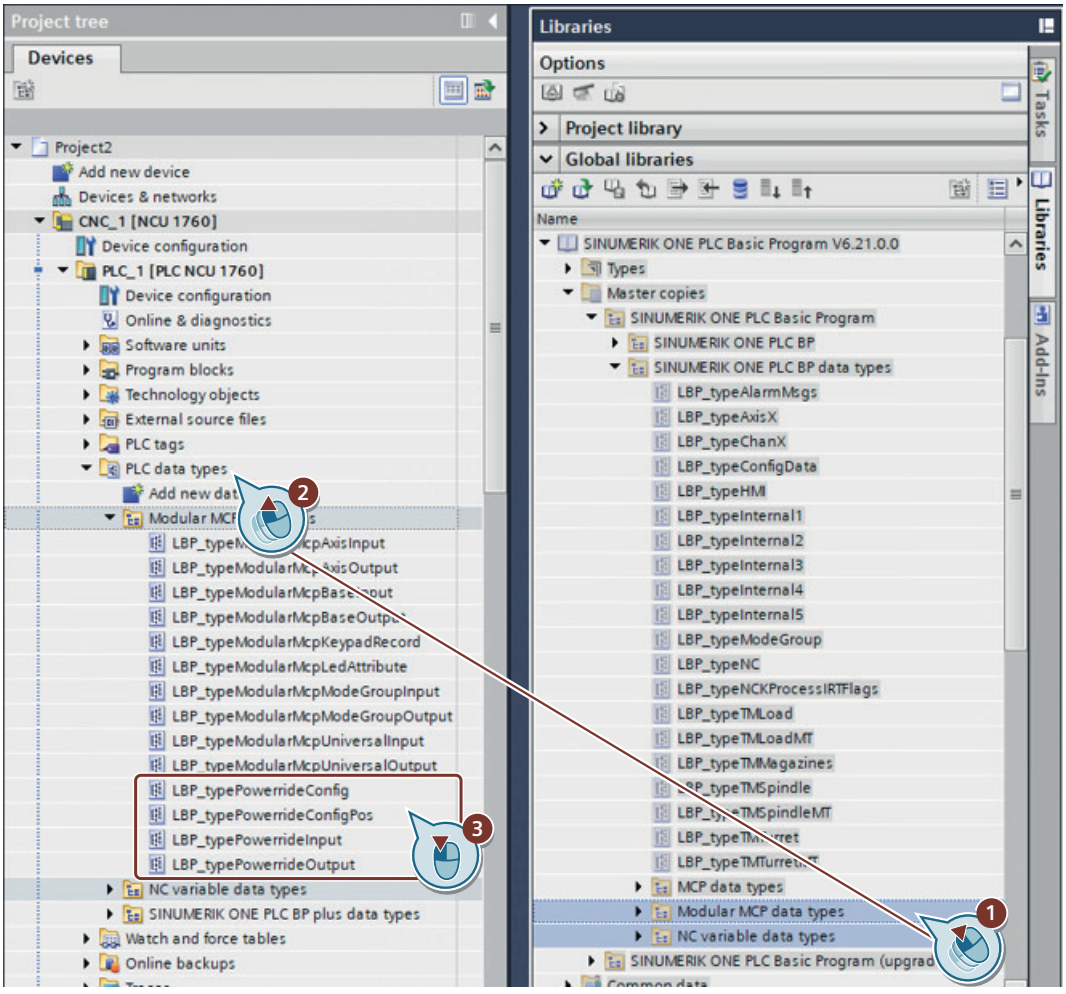


图 4-3　嵌入选择的 Basic Programm（比如 V6.xx 等）PLC 数据类型

3. 完成嵌入后，您可在“Modular MCP data types”（模块化 MCP 数据类型）子文件夹内删除下列条目的 PLC 数据类型：
   - LBP_typePowerride*
     为 Powerride 使用 SINUMERIK ONE Basic Program plus data types 的 PLC 数据类型。

结果

数据类型已复制到项目中并分配给相应的 PLC 基本程序块。

说明

删除不使用的数据类型

检查文件夹"PLC 数据类型"中是否有未被程序块使用的数据类型。

4.8 嵌入 PLC 指令

前提条件

- 已嵌入 PLC 基本程序块和数据类型
- 已在程序编辑器中打开一个块
- 项目中必须有 SINUMERIK ONE NCU V6.20（或更高版本）。只有在 SINUMERIK ONE NCU V6.20（或更高版本）下的块中，才能嵌入指令。

步骤

说明

包含指令的数据库版本和版本 CNC-SW（固件版本）

在 TIA-Portal V18 中，对于 CNC-SW V6.20 推荐系统库版本 V10.1，对于 CNC-SW V6.21 推荐系统库版本 V11.0。

按下列步骤操作，可在用户程序中使用基本程序的各项指令：

1. 切换至"指令"选项卡，然后在"选件包"部分打开"SINUMERIK ONE"系统库。
2. 选择指令的系统库版本。根据所选系统库版本调整指令的版本。
   您可以在"指令可用性 (页 53)"一章中找到有关系统库版本的**更多信息**。

3. 将相关指令从选项卡拖动至打开的程序编辑器中。
   指令嵌入所打开块的设置编程语言中。

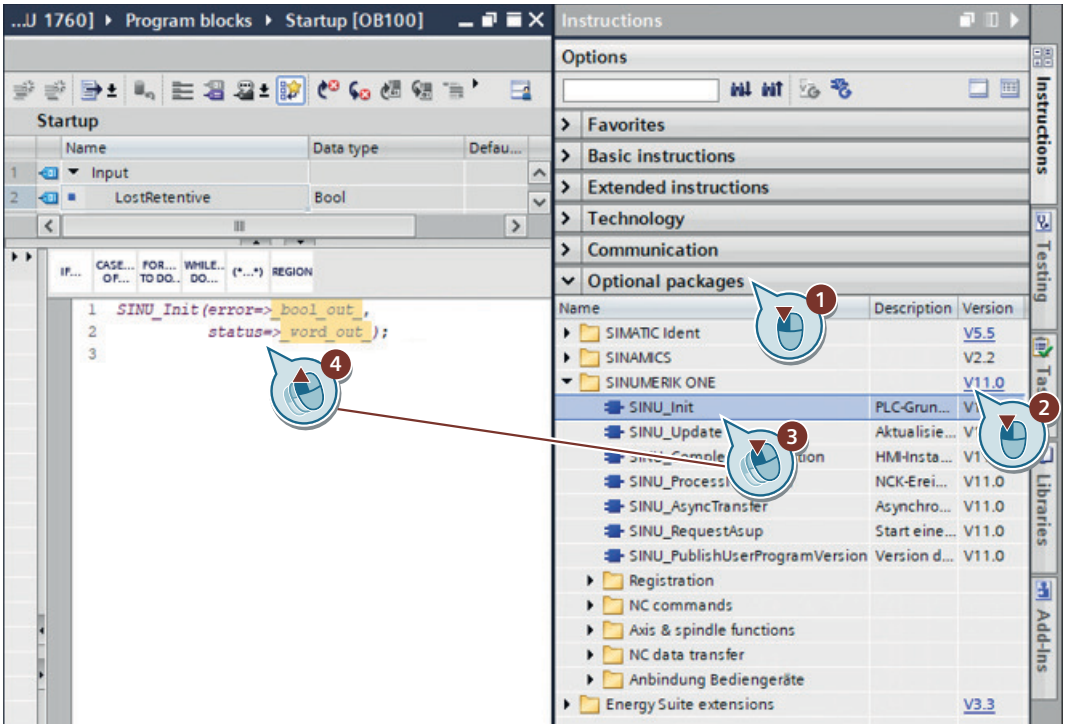


图 4-4　以"SINU_Init"为例，在 SCL 编程语言中嵌入指令

4. 为用户程序补充其他指令并对其进行编程。

结果

SINUMERIK PLC 基本程序的指令成功嵌入用户程序中。

4.9 指令可用性和 NCK 相关数据类型

说明

自 CNC-SW（固件）V6.20 起可用

自 SINUMERIK CNC 软件版本（固件版本）V6.20 起，PLC Basic Program Plus 的指令系统库可用于 SINUMERIK ONE。不支持旧版 SINUMERIK CNC 软件版本。

TIA Portal 项目中的指令包含固件版本 < V6.20

- 指令在系统库中以灰显形式展示且不能添加至 PLC 程序。
- 如果在 PLC 程序中使用 PLC Basic Programm plus 的指令或 NCK 相关数据类型（例如通过导入外部源），PLC 程序的编译可能出错。

系统库版本和在固件版本中的可用性

以下表格展示了指令的系统库版本以及在 NCU 固件版本中的可用性。

设置的系统库版本	TIA Portal 项目中的 NCU 固件版本	可用性和备注
SINUMERIK ONE V10.1		
	自 V6.20 起	**可用** 系统库 **SINUMERIK ONE V10.1** 的指令和 NCK 相关数据类型可在 PLC 程序中使用。
SINUMERIK ONE V11.0		
	V6.20	**不可用** 系统库 **SINUMERIK ONE V11.0** 的指令和 NCK 相关数据类型**不**可在 PLC 程序中使用。
	V6.21	**可用** 系统库 **SINUMERIK ONE V11.0** 的指令和 NCK 相关数据类型可在 PLC 程序中使用。

说明

使用来自不可用的数据库版本的指令

如果在 PLC 程序中使用 SINUMERIK ONE 的指令或 NCK 相关数据类型，但其对于已设置的 NCU 固件版本不可用（例如通过导入外部源），PLC 程序的编译可能出错。

4.10 TIA Portal 项目的升级

在从 TIA Portal V18 以下版本打开包含 PLC Basic plus 用户程序的 SINUMERIK 项目时，系统会要求升级项目。使项目可以在 TIA Portal 的新版本中继续使用。已设置的 NCU 设备版本（例如 V6.20）不变。

说明

升级包含 SINUMERIK ONE PLC BP 的固件 V6.20 以下版本

如果您想要将包含 SINUMERIK ONE PLC BP 的 TIA Portal 项目从旧版本升级且想要使用 SINUMERIK ONE PLC BP plus，则通常必须将 PLC 用户程序全部重新创建。PLC 基本程序 SINUMERIK ONE PLC BP 和 SINUMERIK ONE PLC BP plus 不兼容且不能组合使用。

更多信息

有关 TIA Portal 项目和固件版本升级的更多信息参见*调试手册之组态与调试工序*的章节“升级机床项目”。

理论上的项目更新过程和指令

1. 打开较旧版本的 TIA Portal 项目时，项目将根据要求自动升级至当前 TIA Portal 版本。
2. 保存并编译项目。
3. 通过更换设备将 SINUMERIK ONE NCU 更换至包含新固件版本的 NCU
4. 保存并编译项目。
5. 删除您的 TIA Portal 项目中的 SINUMERIK ONE PLC BP plus 数据类型和 SINUMERIK ONE PLC BP plus 块副本。数据类型和块是 SINUMERIK ONE PLC BP plus 系统库的副本。

   说明

   LSINU_Deprecated* 块自 V2.0 起不再是系统库的组成部分

   块 LSINU_DeprecatedConfigOpUnitComm、LSINU_DeprecatedHandwheels、LSINU_DeprecatedOpUnitComm、LSINU_DeprecatedConfigOpUnitCommData、LSINU_DeprecatedInternal1、LSINU_DeprecatedInternalProc2 不再是 SINUMERIK ONE PLC BP plus V2.0 供货范围的组成部分，而是通过指令分离。

   如果保留 TIA Portal 项目中的 PLC 块直至 PLC 程序转换为新的指令，则在此步不要将块移出项目。

   为将来能更好地继续工作，应将 PLC 用户程序中的 LSINU_Deprecated* 块转换为新的指令。

6. 从 SINUMERIK ONE PLC BP plus V2.0 系统库中复制 SINUMERIK ONE PLC BP plus 数据类型和块至 TIA Portal 项目。
7. 不要将版本为 V6.20 的 SINUMERIK ONE PLC BP 数据类型升级至版本 V6.21，可在 PLC 项目中保留。

8. 转入选项卡“指令”，将指令库的 SINUMERIK ONE 系统库设为版本 V11.0。
9. 保存并编译项目（完全编译软件）。
   升级 PLC 项目中的指令至设置的版本 V11.0。
10. 删除 PLC 用户程序中的从 PLC BP plus V1.0 调用 LSINU_Deprecated* 块，用当前的指令系统库指令替换它。

编译时出现故障报告及解决方法

故障报告	解决办法
数据类型“xyz”在此 CPU 上不受支持或者不受当前库版本支持。	SINUMERIK ONE（指令）系统库版本必须设置为与 NCU 固件版本相匹配。
块“xyz”不受所使用的 CPU 或库版本支持。	SINUMERIK ONE（指令）系统库版本必须设置为与 NCU 固件版本相匹配。

更多信息

关于 TIA Portal 项目升级的更多信息您可在 SINUMERIK STEP 7 Toolbox 工具箱软件安装媒介上自述文件中找到 (..\Documents\Readme\)。

4.11 块和库操作

可对块进行分组等操作，也可从其他项目应用或复制块。

在 TIA Portal 和 SINUMERIK STEP 7 Toolbox 工具箱软件的在线帮助中您可以找到更多关于如何处理块的信息。

编程流程

5

5.1 顺序结构

简介

初始化、注册和升级数据所需的基本程序块和指令，可在相应的执行层级调用。执行层级可以是 OB（组织块），也可以是自有 FB（功能块）或 FC。

通过在启动 OB 等中注册，可以将数据交换变量的位置告知基本程序。循环数据交换（更新所有实例）通过“SINU_Update”指令实现。

流程原理

使用 SINUMERIK ONE PLC Basic Programm plus 基本程序的块时，遵循下列流程原理。此流程涉及块的正确调用。

1. 使用 NCK 相关数据类型，对基本程序执行一次初始化 (Init)
   有两种初始化方法
   – 通常可以在启动 OB 中进行初始化。
   – 如果需要，在“SINU_CompleteRegistration”之前的循环操作中也可以进行初始化。
2. 块的一次性登记（注册）
   可通过以下两种方式进行注册
   – 通常可以在启动 OB 中进行注册。
   – 如果需要，在“SINU_CompleteRegistration”之前的循环操作中也可以进行注册。

   说明

   NCK 实例

   注册后可以立即访问 NCK 实例的值。
3. 结束注册，包括为 SINUMERIK Operate (HMI) 提供注册信息 ("SINU_CompleteRegistration")。

   说明

   HMI 实例

   只有在执行了 SINU_CompleteRegistration 后，才可以访问 HMI 实例的值。
4. 使用“SINU_Update”循环更新所有实例。
5. 在“SINU_Update”发布输出信号 "reRegisterNeeded" = TRUE 发出请求后，重新关联注册的实例 ("SINU_ReRegister*")。
6. 出现 NC 过程报警后，使用“SINU_ProcessNckEvent”处理 NCK 事件。

参见

快速入门 (页 61)

5.2 数据类型、访问功能和程序运行

说明

一般情况下，在引导启动过程中对所需的 NCK 相关数据类型的实例执行一次初始化 ① 和注册 ②。

说明

如有必要，也可在完成引导启动后，在循环运行中进行初始化和注册。注意必须在开始执行循环数据交换前执行。

在循环运行中，通过调用指令“SINU_Update”，更新注册的所有用户实例 ③。

如果“SINU_Update”发布输出信号 "reRegisterNeeded" = "TRUE"，需要使用“SINU_ReRegister<Object>”和“SINU_Update”对注册的所有用户实例执行一次重新关联 ④。

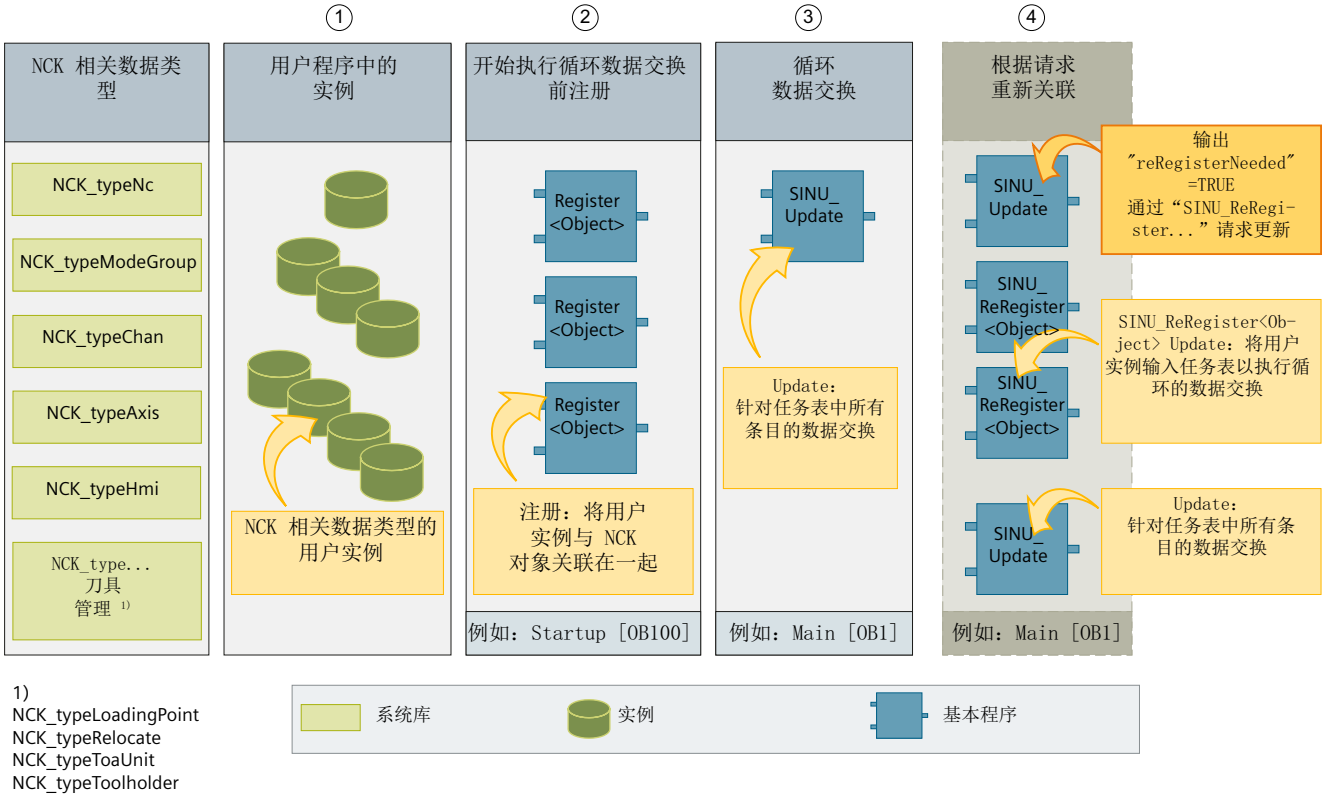


图 5-1　SINUMERIK ONE PLC Basic Program plus 的数据类型和功能

5.3　PLC 基本程序的掉电保持行为

掉电保持是为系统定制预设的设置，无法在 NCK 相关数据类型的变量中进行变更。

参见

掉电保持 (页 89)

快速入门 6

6.1 概览

本章通过举例的方式，介绍如何使用基本程序在 PLC 和 NC 之间进行数据交换。

示例是一台具有下列对象的简单机床：

- 一个 NC (nc)
- 一个通道 (chan1)
- 一个运行方式组 (modeGroup1)
- 三个轴（轴 X1、Y1、Z1）
- 示例：使用刀具管理 (toa, toolholder, loadingPoint, relocate)

前提条件

- 已安装基本程序库和数据类型。
- 已创建一个使用 SINUMERIK ONE 的 TIA 博途项目。
- 已在项目中应用全局库的块和数据类型。
- 如果是“Create MyVirtual Machine”项目，必须额外激活块的可模拟性 (页 133)。

基本编程流程

1. 以相应的数据结构，为所有对象实例创建一个全局数据块。也可为每个对象实例单独创建数据块。
2. 在 PLC 项目中创建一个数据类型为“NCK_typeHmi”的“LSINU_Hmi”全局数据块。
3. 初始化基本程序“SINU_Init”并注册要使用的 NC 对象实例“SINU_Register<对象>”。
4. 在一个循环 OB（比如 OB1）中组态 PLC 和 NC 之间的数据交换，然后使用“SINU_Update”更新 NC 对象实例。
5. 创建 PLC 用户程序。

6.2 创建对象实例的数据块

此基本程序应用于 PLC 和 NCK 或 HMI 间的数据交换，包含“NCK_type”类型使用者实例。例如在一个数据块中声明所有对象。

步骤

按下列步骤创建数据块：

1. 创建一个数据块，例如名称为 "NckInst"。
2. 在静态局部数据（“Static”部分）下，声明块接口中所需的变量。

说明

您也可为每个对象单独创建数据块，以此实现模块化编程，比如包含轴 X1 的“X1”等。

下表所示，是在“Static”部分为所有需要的轴、通道、运行方式组、NC 接口创建的变量，以及为刀具管理变量示例创建的变量（TOA 单元、刀夹、装载位、移刀）。“NCK_typeHmi”类型对象仅在项目中使用 HMI (SINUMERIK Operate) 时必需。

名称	NCK 数据类型	注释
axisX1	NCK_typeAxis	用于机床轴 X1 的 NCK 对象实例
axisY1	NCK_typeAxis	用于机床轴 Y1 的 NCK 对象实例
axisZ1	NCK_typeAxis	用于机床轴 Z1 的 NCK 对象实例
hmiX1	NCK_typeHmiAxis	用于机床轴 X1 的 NCK 对象实例，含 HMI 连接
hmiY1	NCK_typeHmiAxis	用于机床轴 Y1 的 NCK 对象实例，含 HMI 连接
hmiZ1	NCK_typeHmiAxis	用于机床轴 Z1 的 NCK 对象实例，含 HMI 连接
chan1	NCK_typeChan	用于一个通道的 NCK 对象实例
hmiChan1	NCK_typeHmiChan	用于一个通道的 NCK 对象实例，含 HMI 连接
modeGroup1	NCK_typeModeGroup	用于一个运行方式组的 NCK 对象实例
hmiModeGroup1	NCK_typeHmiModeGroup	用于一个运行方式组的 NCK 对象实例，含 HMI 连接
nc	NCK_typeNc	用于 NC 的 NCK 对象实例
hmiNc	NCK_typeHmiNc	用于 NC 的 NCK 对象实例，含 HMI 连接
toa	NCK_typeToaUnit	用于 TO 单位的 NCK 对象实例
toolholder	NCK_typeToolholder	用于刀夹的 NCK 对象实例
loadingPoint	NCK_typeLoadingPoint	用于装载位的 NCK 对象实例
relocate	NCK_typeRelocate	用于“移刀”数据的 NCK 对象实例

NckInst

	Name	Data type	Start valu	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Supe...
1	Static								
2	nc	NCK_typeNc							
3	Base	NCK_typeBase							
4	Input								
5	Output								
6	InOut								
7	Static								
8	config	NCK_typeNcConfig							
9	basic	NCK_typeNcBasic							
10	handwheels	Array[1..6] of NCK_typeNcHandwheel							
11	digitalInputs	Array[1..40] of NCK_typeNcDigitalInput							
12	digitalOutputs	Array[1..40] of NCK_typeNcDigitalOutput							
13	analogInputs	Array[1..8] of NCK_typeNcAnalogInput							
14	analogOutputs	Array[1..8] of NCK_typeNcAnalogOutput							
15	robot	NCK_typeNcRobot							
16	collisionProtAreas	Array[0..63] of NCK_typeNcCollisionProtArea							
17	collision	NCK_typeNcCollision							
18	hmiNc	NCK_typeHmiNc							
19	modeGroup1	NCK_typeModeGroup							
20	hmiModeGroup1	NCK_typeHmiModeGroup							
21	chan1	NCK_typeChan							
22	hmiChan1	NCK_typeHmiChan							
23	axisX1	NCK_typeAxis							
24	axisY1	NCK_typeAxis							
25	axisZ1	NCK_typeAxis							
26	hmiX1	NCK_typeHmiAxis							
27	hmiY1	NCK_typeHmiAxis							
28	hmiZ1	NCK_typeHmiAxis							

图 6-1 “NckInst”数据块示例

其他步骤

下一步，为与 HMI 交换创建数据块“LSINU_Hmi”。

6.3 为与 HMI 交换数据创建数据块

为了让 HMI SINUMERIK Operate 能使用接口信号，必须在项目中创建名称为“LSINU_Hmi”、数据类型为“NCK_typeHmi”的全局数据块。使用 HMI 时需要用到这个全局数据块。

步骤

按下列步骤创建数据块：

1. 生成一个数据类型为 "NCK_typeHmi".，名为 "LSINU_Hmi" 的全局数据块。

注意
名称和数据类型“LSINU_Hmi” 数据块名称**必须**为“LSINU_Hmi”，且数据类型必须为“NCK_typeHmi”。不允许使用其他名称和数据类型！

6.4 在启动过程中初始化和注册

开始调用所有其他功能之前，必须先使用"SINU_Init"指令对基本程序进行初始化。

背景数据块中创建的各个变量，必须针对轴、通道、运行方式组、NC 接口、TOA 单元、刀夹、装载位和移刀进行注册。

注册时，出于分配 NC 对象的目的，将数据结构的存储情况告知基本程序。所有实例都完成注册后，必须结束注册。

在启动 OB 中注册

在组织块 Startup [OB100] 中，调用初始化指令、用于注册的指令，以及用于结束注册的“SINU_CompleteRegistration”。

下例介绍如何在启动过程中进行注册调用。

SCL

```
// tempError: Bool
// tempStatus: Word
...
SINU_Init(error => #tempError,
          status => #tempStatus);
SINU_RegisterNcWithHmi(error => #tempError,
                       status => #tempStatus,
                       nc := "NckInst".nc,
                       hmiNc := "NckInst".hmiNc);
SINU_RegisterModeGroupWithHmi(number := 1,
                              error => #tempError,
                              status => #tempStatus,
                              modeGroup := "NckInst".modeGroup1,
                              hmiModeGroup := "NckInst".hmiModeGroup1
                              );
```

SCL

```
SINU_RegisterChanWithHmi(number := 1,
                         error => #tempError,
                         status => #tempStatus,
                         chan := "NckInst".chan1,
                         hmiChan := "NckInst".hmiChan1
                         );
SINU_RegisterAxisWithHmi(number := 1,
                         error => #tempError,
                         status => #tempStatus,
                         axis := "NckInst".axisX1,
                         hmiAxis := "NckInst".hmiX1
                         );
SINU_RegisterAxisWithHmi(number := 2,
                         error => #tempError,
                         status => #tempStatus,
                         axis := "NckInst".axisY1,
                         hmiAxis := "NckInst".hmiY1
                         );
SINU_RegisterAxisWithHmi(number := 3,
                         error => #tempError,
                         status => #tempStatus,
                         axis := "NckInst".axisZ1,
                         hmiAxis := "NckInst".hmiZ1
                         );
SINU_RegisterToaUnit(number := 1,
                     error => #tempError,
                     status => #tempStatus,
                     toaUnit := "NckInst".toa
                     );
SINU_RegisterToolholder(number := 1,
                        error => #tempError,
                        status => #tempStatus,
                        toolholder := "NckInst".toolholder,
                        toaUnit := "NckInst".toa
                        );
SINU_RegisterLoadingPoint(number := 1,
                          error => #tempError,
                          status => #tempStatus,
                          loadingPoint := "NckInst".loadingpoint,
                          toaUnit := "NckInst".toa
                          );
SINU_RegisterRelocate(error => #tempError,
                      status => #tempStatus,
                      relocate := "NckInst".relocate,
                      toaUnit := "NckInst".toa
                      );
```

SCL

```
SINU_CompleteRegistration(error => #tempError,
                          status => #tempStatus,
                          hmi := "LSINU_Hmi"
                          );
...
```

说明

示例备注

也可使用"SINU_RegisterAxisByInst"和"SINU_RegisterAxisByName"指令进行轴注册。

6.5 循环交换数据

背景数据块中的变量通过循环调用进行更新。

循环 OB 中的数据交换

在循环OB（比如OB1）中调用“SINU_Update”。“SINU_Update”发布信号“reRegisterNeeded” = “TRUE”时（比如运行中完成下载后），必须先重新关联注册的所有实例 ("SINU_ReRegister...")。注册后再次调用“SINU_Update”。调用“SINU_Update”后，可以编程设置另外一个用户程序。

下例介绍循环更新调用。

SCL

```
// tempNcCpuReady: Bool
// tempReRegisterNeeded: Bool
// tempError: Bool
// tempStatus: Word
...
SINU_Update(ncCpuReady => #tempNcCpuReady,
            reRegisterNeeded => #tempReRegisterNeeded,
            error => #tempError,
            status => #tempStatus
            );
IF #tempReRegisterNeeded THEN
   SINU_ReRegisterNc(error => #tempError,
                     status => #tempStatus,
                     nc := "NckInst".nc
                     );
```

SCL

```
   SINU_ReRegisterModeGroup(error => #tempError,
                            status => #tempStatus,
                            modeGroup := "NckInst".modeGroup1
                            );
   SINU_ReRegisterChan(error => #tempError,
                       status => #tempStatus,
                       chan := "NckInst".chan1
                       );
   SINU_ReRegisterAxis(error => #tempError,
                       status => #tempStatus,
                       axis := "NckInst".axisX1
                       );
   SINU_ReRegisterAxis(error => #tempError,
                       status => #tempStatus,
                       axis := "NckInst".axisY1
                       );
   SINU_ReRegisterAxis(error => #tempError,
                       status => #tempStatus,
                       axis := "NckInst".axisZ1
                       );
   SINU_ReRegisterToolholder(error => #tempError,
                             status => #tempStatus,
                             toolholder := "NckInst".toolholder
                             );
   SINU_ReRegisterLoadingPoint(error => #tempError,
                               status => #tempStatus,
                               loadingPoint := "NckInst".loadingpoint
                               );
   SINU_ReRegisterRelocate(error => #tempError,
                           status => #tempStatus,
                           relocate := "NckInst".relocate
                           );
   // SINU_ReRegister* 后再次调用 SINU_Update
   SINU_Update(ncCpuReady => #tempNcCpuReady,
               reRegisterNeeded => #tempReRegisterNeeded,
               error => #tempError,
               status => #tempStatus
               );
END_IF;
...
```

6.6 以太网机床控制面板和 HT 10 通讯组态

下面的示例介绍了通过以太网创建模块化机床控制面板通信的方法。机床控制面板由以下模块组成：

- 1 个基本模块
- 4 个按键模块
- 2 个 SINUMERIK Powerride
- 2 个手轮
- 此外，还使用了一个带手轮的 HT 10 手持操作设备。

编程流程

1. 为了建立与各以太网 HMI 设备的循环通信，在单独实例中分别针对模块化的机床控制面板和 HT 10 调用"SINU_UpdateEthernetMcp"指令。
   更多信息参见 "SINU_UpdateEthernetMcp (页 1271)"一章。
2. 为了将两个 HMI 设备的手轮数据传输到 NCK 中，收集模块化 MCP（2 个手轮）和 HT 10（1 个手轮）的手轮数据并通过指令"SINU_TransferHandwheelData"将其传输到 NCK 中。
   更多信息参见 "SINU_TransferHandwheelData (页 1279)"一章。

3. 为了存储机床控制面板的输入和输出数据以及手轮数据，创建一个全局数据块“DataMcp”。为创建模块化机床控制面板的输入和输出映像，请使用 SINUMERIK STEP 7 Toolbox 软件中随附的 PLC 数据类型。
更多信息参见“插入 PLC 数据类型基础程序 (页 50)”一章。

说明

手轮数据的数据类型为“SI (页 1282)”。该数据类型不在全局库中，但属于 TIA 系统库的一部分，必须输入到“数据类型”一栏中。

	Name	Datentyp	Startwert	Kommentar
16	ModularMcpData	Struct		
17	toMcp	Struct		
18	base	"LBP_typeModularMcpBaseOutput"		Ausgangsabbild von Basissignalen für modulare MCP
19	keypad1	"LBP_typeModularMcpUniversalOut...		Ausgangsabbild des Universal-Bedienfelds für modulare MCP
20	keypad2	"LBP_typeModularMcpUniversalOut...		Ausgangsabbild des Universal-Bedienfelds für modulare MCP
21	keypad3	"LBP_typeModularMcpUniversalOut...		Ausgangsabbild des Universal-Bedienfelds für modulare MCP
22	keypad4	"LBP_typeModularMcpUniversalOut...		Ausgangsabbild des Universal-Bedienfelds für modulare MCP
23	powerride1	"LBP_typePowerrideOutput"		Ausgangsabbild von Powerride
24	powerride2	"LBP_typePowerrideOutput"		Ausgangsabbild von Powerride
25	fromMcp	Struct		
26	base	"LBP_typeModularMcpBaseInput"		Eingangsabbild von Basissignalen für modulare MCP
27	keypad1	"LBP_typeModularMcpUniversalInp...		Eingangsabbild des Universal-Bedienfelds für modulare MCP
28	keypad2	"LBP_typeModularMcpUniversalInp...		Eingangsabbild des Universal-Bedienfelds für modulare MCP
29	keypad3	"LBP_typeModularMcpUniversalInp...		Eingangsabbild des Universal-Bedienfelds für modulare MCP
30	keypad4	"LBP_typeModularMcpUniversalInp...		Eingangsabbild des Universal-Bedienfelds für modulare MCP
31	powerride1	"LBP_typePowerrideInput"		Eingangsabbild von Powerride
32	powerride2	"LBP_typePowerrideInput"		Eingangsabbild von Powerride
33	handwheel1	"SINU_typeHandwheelData"		Daten für Handradimpulse, die an der Anweisung SINU_UpdateE
34	active	Bool	false	TRUE: Handrad ist aktiv vorhanden
35	recent	Bool	false	TRUE: neue Daten vom Handrad vorhanden
36	stopped	Bool	false	TRUE: Handrad steht
37	pulses	Int	0	Differenz der Impulse von der letzten Übertragung an den NCK
38	handwheel2	"SINU_typeHandwheelData"		Daten für Handradimpulse, die an der Anweisung SINU_UpdateE
39	HT10Data	Struct		
40	toMcp	Array[1..8] of Byte		
41	fromMcp	Array[1..8] of Byte		
42	handwheel1	"SINU_typeHandwheelData"		Daten für Handradimpulse, die an der Anweisung SINU_UpdateE

图 6-2　数据块中包括了输入和输出映像及手轮数据

4. 在块（用于调用指令）的临时或静态区域定义所有其他的输入和输出参数。

示例

以下示例显示了对通信（与机床控制面板和手持操作设备进行的通信）的工程组态的调用。

SCL

```
// InstUpdateEthernetMcpMod : SINU_UpdateEthernetMcp
// InstUpdateEthernetMcpHT10 : SINU_UpdateEthernetMcp
// tempMcpReady : Bool
// tempNewData : Bool
// tempIdentified : Bool
// tempMcpType : Byte
// tempLengthIn : USInt
// tempLengthOut : USInt
// tempBusy : Bool
// tempWarning : Bool
// tempError : Bool
// tempStatus : Word

// 调用 SINU_UpdateEthernetMcp，以与模块化机床控制面板通信
#InstUpdateEthernetMcpMod(enable := TRUE,
                          connectionId := 1001, // PLC 上的连接标识 1001
                          interfaceId := 262, // 网络接口的接口 ID
                          mcpReady => #tempMcpReady,
                          newData => #tempNewData,
                          identified => #tempIdentified,
                          mcpType => #tempMcpType,
                          lengthIn => #tempLengthIn,
                          lengthOut => #tempLengthOut,
                          handwheel1 => "DataMcp".ModularMcpData.handwheel1,
                          handwheel2 => "DataMcp".ModularMcpData.handwheel2,
                          busy => #tempBusy,
                          warning => #tempWarning,
                          error => #tempError,
                          status => #tempStatus,
                          fromMcp := "DataMcp".ModularMcpData.fromMcp,
                          toMcp := "DataMcp".ModularMcpData.toMcp);
```

SCL

```
// 调用 SINU_UpdateEthernetMcp，以与 HT10 通信
#InstUpdateEthernetMcpHT10(enable := TRUE,
                                address := 14, // HT 10 地址 14
                                udpPort := 16002, // PLC 上的端口 16002
                                connectionId := 1002, // PLC 上的连接标识 1002
                                interfaceId := 262, // 网络接口的接口 ID
                                mcpReady => #tempMcpReady,
                                newData => #tempNewData,
                                identified => #tempIdentified,
                                mcpType => #tempMcpType,
                                lengthIn => #tempLengthIn,
                                lengthOut => #tempLengthOut,
                                handwheel1 => "DataMcp".HT10Data.handwheel1,
                                busy => #tempBusy,
                                warning => #tempWarning,
                                error => #tempError,
                                status => #tempStatus,
                                fromMcp := "DataMcp".HT10Data.fromMcp,
                                toMcp := "DataMcp".HT10Data.toMcp);

// 调用 SINU_TransferHandwheelData，以将手轮数据传输到 NCK 中。
"SINU_TransferHandwheelData"(handwheel1:="DataMcp".ModularMcpData.handwheel1, //机床控制面板
的手轮 1
                             handwheel2:="DataMcp".ModularMcpData.handwheel2, // 机床控制面板
的手轮 2
                             handwheel3:="DataMcp".HT10Data.handwheel1, // HT 10 的手轮
                             handwheel4:=NULL,
                             handwheel5:=NULL,
                             handwheel6:=NULL,
                             error=>#tempError,
                             status=>#tempStatus);
```

后续步骤

此步骤仅设置设备通讯。机床仍无法操作。在用户程序中还必须编程机床控制面板 (MCP) 和 NC/PLC 接口（NC、运行方式组、通道、轴、HMI）之间的信号。MCP（按键等）的信号传递至 NC/PLC 接口，NC/PLC 接口的信号传递至 MCP （LED 灯）。

6.7 NCK 信号的传输和应答时间

NC (NCK) 中的不同事件会传输到 PLC。视事件类型而定，可在不同时间，在 PLC 用户程序中访问这些事件，并于不同时间在 NCK 上应答。

在过程报警 OB 中处理事件

在 NCK 事件的过程报警 OB（比如 OB40）中调用“SINU_ProcessNckEvent”，参见 NC 事件和过程报警 (页 91)章节。无需调用其他块。在后续循环中提供数据。

示例

下例介绍用于传输事件控制的 NCK 信号的调用。

SCL

```
// tempError: Bool
// tempStatus: Word
SINU_ProcessNckEvent(error => #tempError,
                     status => #tempStatus);
```

更多信息

关于 NCK 事件处理的详细信息和示例，参见“SINU_ProcessNckEvent (页 1091)”章节。

6.8 编程技巧与建议

本章主要介绍 PLC 程序组态和编程的技巧与建议。

组态提示

在块的声明部分使用 NCK 相关数据类型，FC、FB

- Input, Output, InOut
  不支持 NCK 数据类型阵列，只能使用单个 NCK 数据类型
- Temp, Const
  不支持 NCK 数据类型
- Static
  支持 NCK 数据类型阵列和所有 NCK 数据类型
- STRUCT，PLC 数据类型
  不支持 NCK 数据类型

编程提示

用户程序结构

- 既可在单独的数据块中为每个对象声明 NCK 相关数据类型，也可在一个数据块中为多个对象声明数据类型。这样可以在一个数据块中声明各对象的逻辑关系。
  数据块举例：
  - 全局 DB（数据块）“通道 1”具有通道 1 的对象示例（包括 HMI）以及通道 1 下属各轴。
  - 用于更换刀具的全局 DB（换刀装置通道，下属各轴...）
- 必须先用“SINU_Init”对基本程序进行初始化，否则无法注册对象。
- 动态注册 NC 对象：
  - 将 NC 对象注册为第一个
  - 根据 <NC>.config.activeModeGroups 注册运行方式组对象
  - 根据 <NC>.config.activeChans 注册通道对象
  - 根据 <NC>.config.activeAxes 注册轴对象

运行中下载

在运行中下载时，注意数据类型、访问功能和程序运行 (页 58)章节的相关信息。

编译导入的块/源时出错

当不同版本的数据（例如所使用的选项卡指令）通过全局库或者 VCI 导出/导入时，导入源/块后编译 PLC 块时出现错误。

底部

通过 Openness 或使用 VCI（版本控制接口）从 SCL 或 XML 文件导入代码后，在编译导入的 PLC 块时可能会出现错误。如果数据块或用户程序中的变量使用了特定变量名称，并且在前一次导出时在“选件包 > SINUMERIK ONE”下为指令设置的版本与导入时不同，就会出现这种情况。

编译器会将该结构解释为以下变量名的分片访问，并自动进行转换。

- <Tag>.**X**<0..63>
- <Tag>.**B**<0..7>
- <Tag>.**W**<0..3>
- <Tag>.**D**<0..1>

示例：

"MyInstDB".X1 变为 "MyInstDB".%X1

由于没有定义访问目标，在编译过程中会输出一个错误。

解决办法

- 在导出和随后导入时，请使用“选件包 > SINUMERIK ONE”下相同版本的指令。
- 在导入和编译后显示的出现错误的位置，删除所插入的 % 符号。
- 不要使用 X<数字>、B<数字>、D<数字>、W<数字> 的组合作为数据块子结构的第一个元素。

错误	正确
"MyInstDB".X1	"MyInstDB".axisX1
"MyInstDB".X1	"MyInstDB".X
"MyInstDB".myAxis.X1	"MyInstDB".myAxis.axisX1

- X<数字>、B<数字>、D<数字>、W<数字> 的组合仅可直接作为的数据块名称，但不能在子结构中使用。

错误	正确
"MyInstDB".X1	"X1"
"MyInstDB".B1	"B1"

SINUMERIK 用户报警 7

SINUMERIK 消息借助 SINUMERIK Basic Program plus 供货时随附的块触发。消息包括 SINUMERIK 用户报警等。

在新款 PLC 基本程序 PLC Basic Program plus 中，老版 SINUMERIK DB2 报警升级为新报警体系“SINUMERIK 用户报警”。新报警体系不但支持模块化程序结构，还用全局 DB 替换了呆板的编程操作。

用户可以在 TIA Portal 的“SINUMERIK_UserAlarmsTextLib”文本列表中，维护 SINUMERIK 用户报警文本。针对 SINUMERIK 用户报警，可通过 #alarmCategory 选择生成哪种类型的消息。此外，SINUMERIK 用户报警还支持自行编程设置应答，以及规定哪些信号作用于哪些需要应答的报警。

下文为大家介绍 SINUMERIK 用户报警的特性和使用方法。

说明

在 SINUMERIK 用户报警的 AppNote 中，您可以找到 SINUMERIK 用户报警的一些定义示例。AppNote 请参见西门子工业在线支持 (https://support.industry.siemens.com/cs/ww/en/view/109812364)页面。

7.1 消息文本

消息文本管理方法

原则上消息文本有两种管理方式：

- 在 TIA 博途的“SINUMERIK_UserAlarmsTextLib”文本列表中，管理所有语言编写的消息文本。
- 与以前一样，仍然在 TIA 博途项目外部，通过“*.ts”文件管理 SINUMERIK 用户报警的文本。

说明

- 两种管理方式不可混用，必须二选一。
- 在 TIA 博途项目中，除了用于 SINUMERIK 用户报警的可选文本之外，还有其他文本（比如 PLC1500 的系统诊断文本等）。TIA 博途中已知的文本传输到 SINUMERIK Operate 后，导出数据中也会包含在 TIA 博途项目中导出数据时，所包含的 SINUMERIK 用户报警文本。如果 TIA 博途项目中包含 SINUMERIK 用户报警文本，则在 SINUMERIK Operate 中生效的会是这些文本，而非可能附加保存的“*.ts”文件中的文本。

在 TIA 博途中管理消息文本

在 TIA 博途的“SINUMERIK_UserAlarmsTextLib”文本列表中，管理所有语言编写的消息文本。消息文本可通过 PLC 快捷菜单“为 SINUMERIK Operate 导出报警和文本列表…”导出。导出时，将消息文本导出至“PLCAlarms.xml”和“PLCTextLists.xml”文件，然后通过将文件保存在“/oem/sinumerik/hmi/lng”目录下，达到传输至 SINUMERIK Operate 的目的。

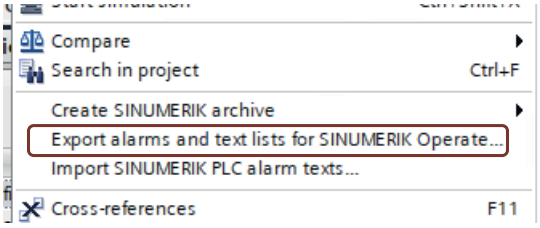


图 7-1　PLC 快捷菜单

在 TIA 博途外部管理消息文本

与以前一样，仍然在 TIA 博途项目外部，通过“*.ts”文件管理 SINUMERIK 用户报警的文本，然后将其传输到 SINUMERIK Operate。

7.1.1 通过 TIA 管理消息文本

特性

- 在 TIA 博途项目导航区的“语言和资源”下，选择所需的项目语言。如果选择多种语言，TIA 博途项目的规模会相应变大。

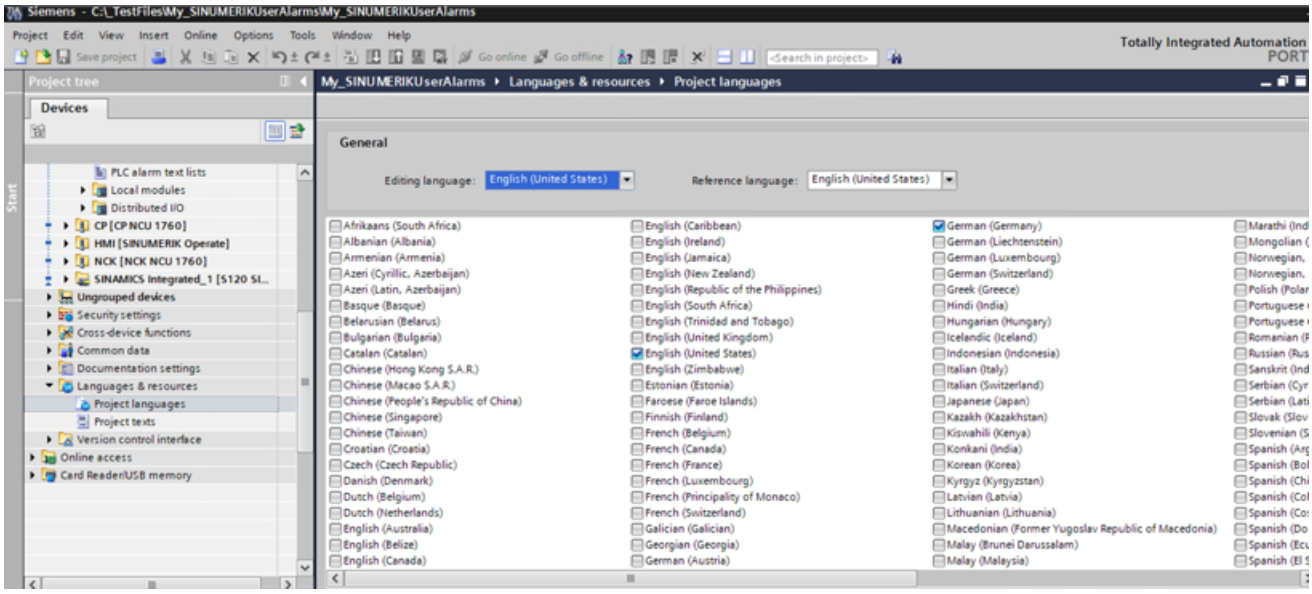


图 7-2 TIA 博途中的消息文本项目语言

- SINUMERIK 用户报警文本可在 TIA 博途项目中与 PLC 项目一起读取和编辑；文本列表及其下属文本可以导出或导入（*.xlsx 格式），或者作为复制模板保存在 TIA 博途库中。
- 在 TIA 博途或 PLC Web 服务器等的报警显示界面，消息文本为已知数据。

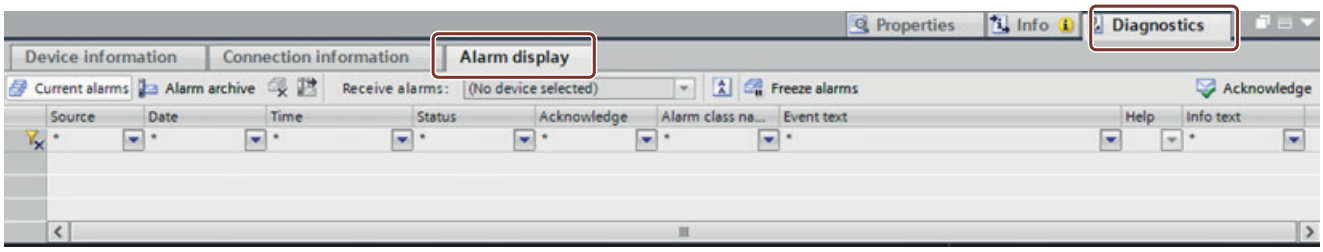


图 7-3 报警显示界面的消息文本

- 不可手动调整 SINUMERIK Operate 的“*.ts”和“*.cfg”文件。
- 无法直接从老版“*.ts”文件导入消息文本。TIA 博途快捷菜单中的导入功能“导入 SINUMERIK PLC 报警文本...”，不适用于 SINUMERIK 用户报警，因为导入的文本会导入到“SYSTEM_SINUMERIK_DB2Alarms TextLib”文本列表中，而此文本列表只可用于 SINUMERIK DB2 报警，不支持 SINUMERIK 用户报警。SINUMERIK 用户报警的关联值也与 SINUMERIK DB2 报警不同。

编辑消息文本

- 在“CNC_1 > PLC_1 > PLC 消息文本列表”等路径下，创建名为“SINUMERIK_UserAlarmsTextLib”的消息文本列表。
- 在此文本列表中，为 TIA 博途需要的所有语言输入所需文本。操作时，必须在“范围从”和“范围至”两列输入 SINUMERIK 报警编号（参见“LSINU_GenerateSinuUserAlarm”函数的“alarmNumber”参数）。

  说明

  也可为某些连续的 SINUMERIK 报警编号输入共用文本。

- 要在消息文本中嵌入动态关联值，可以使用 SIMATIC 句法，参见关联值章节。

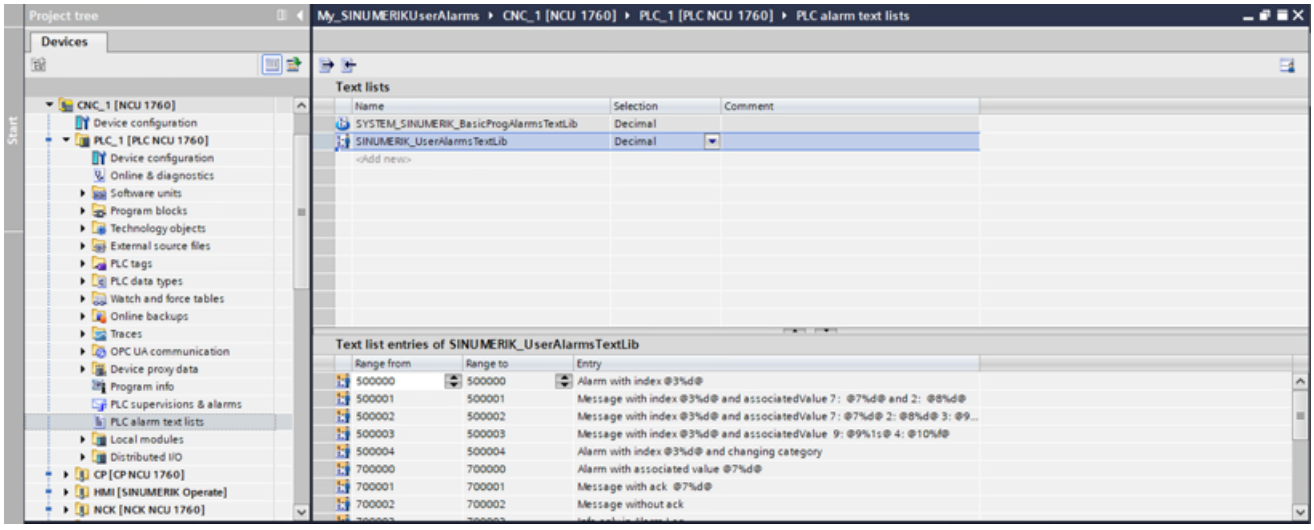


图 7-4　消息文本中的关联值

通过"*.xlsx"文件导出和导入消息文本

您也可以通过"*.xlsx"文件，为 TIA 博途项目导入 SINUMERIK 用户报警文本，或从项目中导出。文本列表名称必须是"SINUMERIK_UserAlarmsTextLib"。

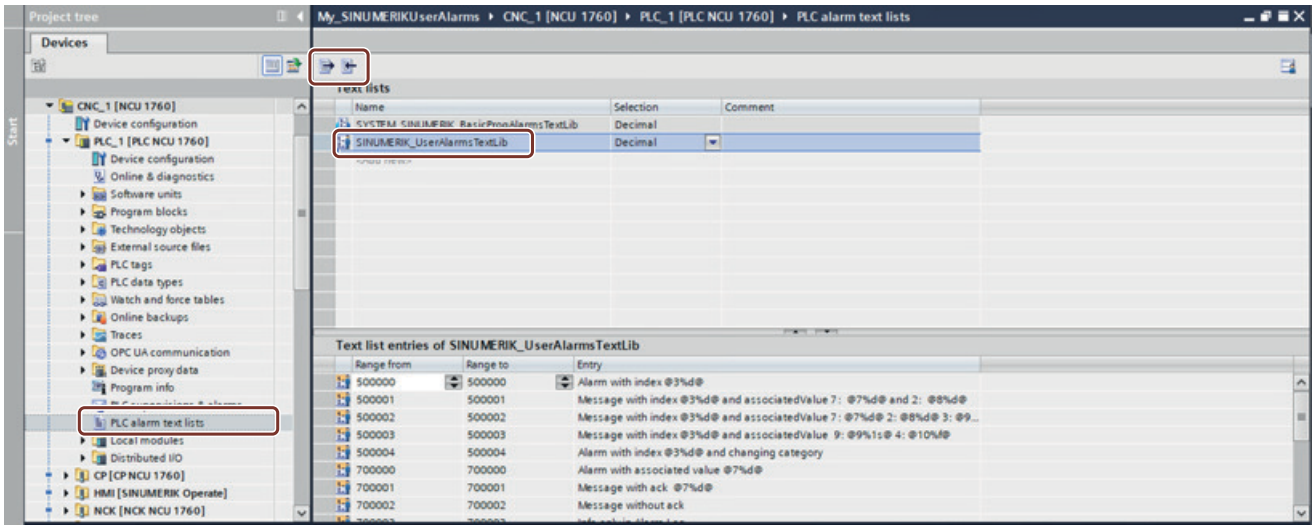


图 7-5　通过"*.xlsx"文件导入和导出

"*.xlsx"文件格式由 TIA 博途指定，包括两个工作表：

- 第一个工作表"TextList"列举文件中包含的所有文本列表，示例中只包含导出时选择的文本列表"SINUMERIK_UserAlarmsTextLib"。

	A	B	C	D
1	Name	ListRange	Comment [de-DE]	Comment [en-US]
2	SINUMERIK_UserAlarmsTextLib	Decimal	<No value>	<No value>
3				
4				

TextList　TextListEntry

图 7-6　"*.xlsx"文件文本列表

- 第二个工作表"TextListEntry"包含各 SINUMERIK 用户报警（或报警编号部分）的文本，并按照"Parent"列对显示的文本列表进行排序。

	A	B	C	D	
1	Parent	From	To	Text [de-DE]	Text [en-US]
2	SINUMERIK_UserAlarmsTextLib	500000	500000	Alarm mit Index @3%d@	Alarm with index @3%d@
3	SINUMERIK_UserAlarmsTextLib	500001	500001	Meldung mit Index @3%d@ u	Message with index @3
4	SINUMERIK_UserAlarmsTextLib	500002	500002	Meldung mit Index @3%d@ u	Message with index @3

TextList　TextListEntry

图 7-7　"*.xlsx"文件中的文本列表条目

针对项目中使用的不同项目语言，每种语言占用一列。

向 SINUMERIK Operate 传输消息文本

可以通过快捷菜单“为 SINUMERIK Operate 导出报警和文本列表...”，导出 SINUMERIK 用户报警文本（与 TIA 博途项目中的其他报警描述和文本一起导出）。接下来弹出的对话框，包含“如何将文本传输至 SINUMERIK Operate”的相关说明。

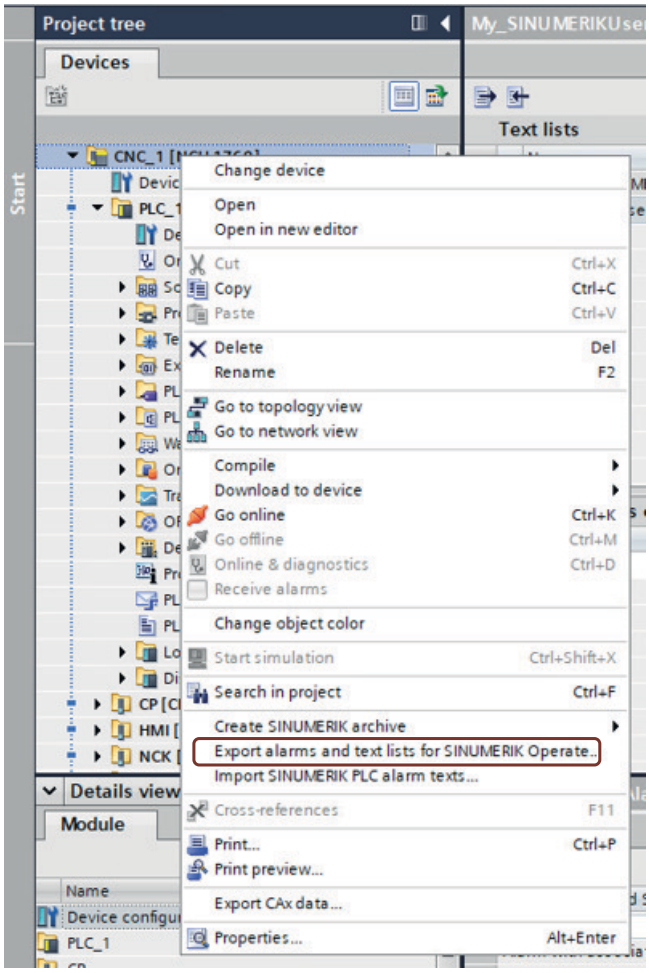


图 7-8　将 SINUMERIK 用户报警文本传输至 SINUMERIK Operate

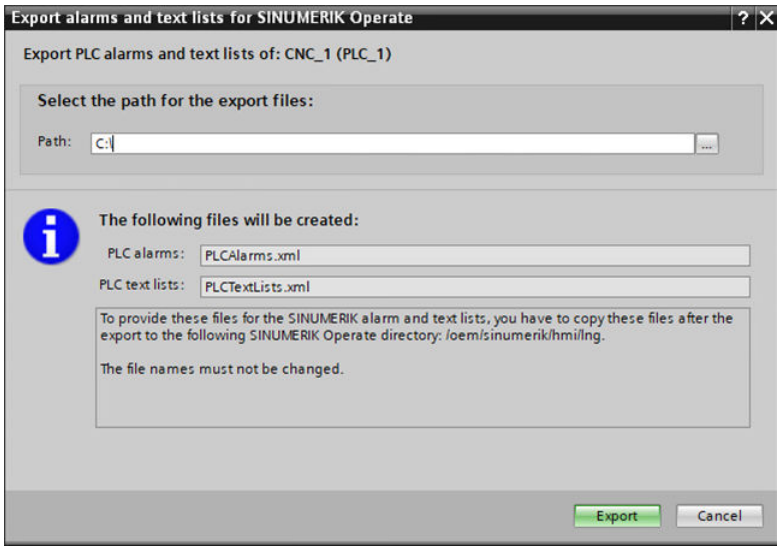


图 7-9　报警和文本列表导出对话框

传输 SINUMERIK 档案中的消息文本

通过快捷菜单“创建 SINUMERIK 档案 > 硬件和所有程序块...”为 SINUMERIK ONE 创建 SINUMERIK 档案时，TIA 博途根据 PLC 配置存储文本。通过 SINUMERIK Operate 读入 SINUMERIK 档案（“*.dsf”）时存储文本，重启 SINUMERIK Operate 后生效。

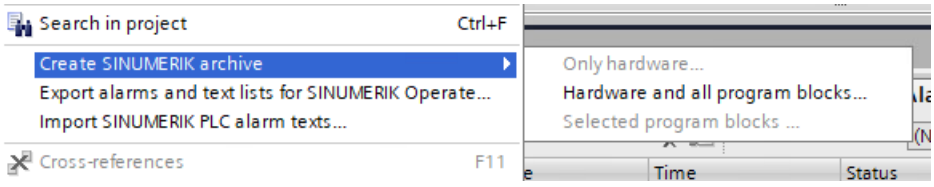


图 7-10　SINUMERIK 档案中的文本

7.1.2 在“*.ts”文件中管理消息文本

特性

- 支持管理多语言文本，可以按照所需的语言范围管理文本。“*.ts”文件存储在 SINUMERIK Operate 文件系统中。文本和语言规模不会导致 TIA 博途项目变大。
- 无法在 TIA 博途中读取或编辑“*.ts”文件中的文本，而且这些文本对于 PLC Web 服务器来说是未知数据。

  说明

  无法直接从老版“*.ts”文件导入消息文本。TIA 博途快捷菜单中的导入功能“导入 SINUMERIK PLC 报警文本...”，不适用于 SINUMERIK 用户报警，因为导入的文本会导入到“SYSTEM_SINUMERIK_DB2Alarms TextLib”文本列表中，而此文本列表只可用于 SINUMERIK DB2 报警，不支持 SINUMERIK 用户报警。SINUMERIK 用户报警的关联值也与 SINUMERIK DB2 报警不同。

- 您可以在“*.ts”文件中编辑文本，而且必须相应调整 SINUMERIK Operate 的“*.cfg”文件。

7.2 报警类别（消息类别）

SINUMERIK 报警划分为以下几个类别（LSINU_GenerateSinuUserAlarm：块参数“alarmCategory”）：

名称	在 HMI Operate 中的显示	消息类别
故障	报警列表	SINUMERIK error messages (ErrM)
需要应答的消息	消息列表	SINUMERIK operating messages with acknowledgement (OpMAck)
不需要应答的消息	消息列表	SINUMERIK operating messages (OpM)
信息	报警日志， 只记录，只录入输入的报警	SINUMERIK operating messages (OpM) +“仅信息”

SINUMERIK 中的消息类别

在 SINUMERIK Operate 中，消息类别原则上属于未知数据。这一点也适用于 SINUMERIK 消息类别。借助 SINUMERIK 消息类别，比如可以在 WinCC 中影响基于相关消息类别的消息的显示方式。因此为 SINUMERIK 报警的各种消息类型分配了不同的报警类别。

在 SINUMERIK Operate 中的显示位置，不通过消息类别进行控制。

在 WinCC 中，显示位置也不通过消息类别进行控制，而是通过组态相应的控制实现目的。

更多信息

详细信息请访问 TIA 博途在线帮助进行查询，或者参阅 *组态 SINUMERIK 报警*手册的“SINUMERIK 消息和其他消息在 SINUMERIK Operate 中的显示位置”章节。

7.3 应答

必须在用户程序中，通过 PLC 信号应答 SINUMERIK 用户报警。不支持通过软键、快捷菜单的菜单项、WinCC 报告控制等其他机制进行应答。

7.4 重新初始化

RUN 模式中的文本更改和下载

在 TIA Portal 中更改报警文本列表并随后在 RUN 模式中下载程序块后，基于不同的系统，所有待处理的报警可能会被删除。

为识别该状态，可以循环调用块“LSINU_GetReinitSinuUserAlarms”。可以通过循环的“reinitNeeded”信号重新发送所有报警。如果设置了“alarmCome”，则删除报警的内部状态，并且必须重新发送报警。

说明

如果在用户程序中不考虑上述操作，则必须通过重启 PLC 强制实现报警显示和 PLC 数据块的同步。

7.5 消息文本中的关联值

在报警和消息文本中，可以为 SINUMERIK 用户报警的动态内容使用占位符，等到运行时将其替换成关联值、报警参数、变量、关键词或文本列表/索引文件条目。

为此，SINUMERIK ONE 的 TS 文件中设有 2 种书写方式，即以 % 符号作为前缀的 SINUMERIK (Operate) 书写方式和以 @ 符号作为前缀的 SIMATIC（TIA 博途）书写方式。

- 必须为用户自定义报警文件或消息文本文件指定其中一种书写方式。
- 两种文件类型都能导入 SINUMERIK Operate 和 TIA 博途中，导入时书写方式会转换成相应的标准句法。

要在消息文本中嵌入动态关联值，可以使用 SIMATIC 句法。

可以从 TIA 博途已知的基本数据类型中，为各关联值自定义选择数据类型。

SINUMERIK 用户报警为用户提供 484 = 512 – 4 – 24 字节用于指令的关联值。在最大值 512 字节中，有 4 个字节被内部使用的报警指令预留和占用，另有 24 个字节被基本程序内部预留和占用。

关联值作为参数传递到“LSINU_GenerateSinuUserAlarm”块上。

7.5.1 报警编号“alarmNumber”

报警编号作为参数传输到“LSINU_GenerateSinuUserAlarm”，而且在 PLC 范围内必须具有唯一性。报警编号固定与报警一起，作为关联值 1 进行传输。

数据类型为 UDInt。

有效值域：500.000 - 999.999

说明

- 使用 GRAPH 编程语言时，切不可从 8xx.xxx 部分生成报警。8xx.xxx 部分是 GRAPH 的预留部分。
- 一般不允许出现两个报警使用相同报警编号的情况，除非它们的报警类别或报警索引存在差异。对于定义的报警来说，这样可以提高或变更报警的紧急程度。始终显示相同的文本。新出现的报警运行时，会检查报警编号。这时会相对于同一类别和同一索引下已激活的报警进行检查。

参见

关联值概述 (页 85)

7.5.2 报警索引“alarmIndex”

报警索引固定与报警一起，作为关联值 3 进行传输；此外，还可作为参数应用在报警文本中。数据类型为 UInt。

生成报警时，会检查是否存在具有相同编号、相同类别和相同索引的报警。如果存在，会报错。

如果报警具有相同编号和相同索引，但类别不同，这种情况是允许出现的。出于在不同设备部分使用相同报警等目的，可以使用包含索引的报警。

参见

关联值概述 (页 85)

7.5.3 关联值“associatedValue”

通过关联值扩展报警文本“associatedValue”

通过参数“associatedValue7”...“associatedValue10”，最多可为报警关联四个关联值。这些关联值在显示报警时嵌入报警文本中。

参见

关联值概述 (页 85)

7.5.4 关联值概述

关联值概述

文本列表条目中的关联值基准化：

块参数 LSINU_GenerateSinuUserAlarm	文本列表条目中的参考
alarmNumber	@1%u@
alarmIndex	@3%u@
associatedValue7	@7%<format>@
associatedValue8	@8%<format>@

块参数 LSINU_GenerateSinuUserAlarm	文本列表条目中的参考
associatedValue9	@9%<format>@
associatedValue10	@10%<format>@

更多关于关联值结构的信息，请参阅 TIA 博途在线技术支持界面，搜索“关联值结构”。

格式 <format>	说明
%[i]X	i 位十六进制数
%[i]u	i 位无符号十进制数
%[i]d	i 位有符号十进制数
%[i]b	i 位二进制数
%[i][.y]f	有符号的浮点数，小数点后有 y 位，总位数为 i 位
%[i]s	i 位字符串（ANSI 字符串），字符打印至第一个 0 字节 (00hex)。
%t#<文本库名称>	访问文本库

可以选择在“[i]”位指定一个数字 i，以此来设置作为输出宽度的位数。

表中的中括号意为“可以选择指定位数”。实际应用中没有中括号。

在 ProDiag 报警文本等中，句法与辅助关联值的句法相同。这种句法同样适用于 S7-300 CPU 和基于 WinCC 的 HMI。HMI Operate 也支持这种句法。

示例

格式示例

- @7%u@ 代表显示关联值 7（= 块参数 associatedValue7），无符号 (u)，位数视需要而定。
- @7%6u@ 代表显示关联值 7（= 块参数 associatedValue7）的 6 位，无符号 (u)。

文本列表示例

变量值	附值	说明	输出
文本库内容 <TextLib1>: Text 1:Text="0°" Text 2:Text="90°" Text 3:Text="180°" Text 4:Text="270°" associatedValue9 的值：2	@9%t#TextLib1@	数据类型为 WORD 的块参数 associatedValue9 的关联值，是引用所用文本库“TextLib1”的索引。	90°

文本列表条目中预留的标识符（关联值）

以下所列，为预留的标识符，切不可在文本列表中使用。

文本列表条目中的参考	说明
@2%<format>@	预留用于西门子
@4%<format>@	预留用于西门子
@5%<format>@	预留用于西门子
@6%<format>@	预留用于西门子

7.5.5 关联值的字符串数据类型

使用字符串数据类型作为关联值时注意：检查所需总长度时，始终计数字符串的声明长度，而非实际占用的长度。

因此，请始终为这类关联值指定实际长度作为数据类型（比如 STRING [20]），这是因为指定字符串数据类型时的默认长度为 254，大多数情况下此长度并不合理。一旦超过允许的最大长度，块 LSINU_GenerateSinuUserAlarm 会生成错误消息。这种情况下块提供下列返回值“statusId == 12; status == 0x8004”。错误信息的详细介绍，请参阅**块的参考说明**。

说明

可以使用字符串作为关联值。建议动态链接其他文本列表。文本列表的优点在于，无需使用为关联值预留的存储区。而且，文本列表于语言编译更为有利。

相关详细信息参见 TIA 博途在线技术支持界面。

关联值作为字符串或文本列表的应用情况

对于关联值是不受语言影响或受语言影响的文本的情况，为您提供以下功能：

- 字符串是不受语言影响的文本（一种语言）
  →（声明的最大）字符串长度记为关联值长度极限值
- 引用文本列表，文本列表包含采用组态语言的文本（多语言）
  →只有“选择器”大小记为关联值长度极限值

通过文本列表将文本保存为组态的所有语言。这可能使用 TIA 博途项目的内存，而且必须满足“存在采用所有语言的文本”这个前提条件（TIA 博途中的文本列表或“*.ts”文件）。

对于不受语言影响的文本，建议使用字符串，因为在多语言项目中，必须在所有语言的文本列表中统一输入不受语言影响的文本。

7.6 时间戳

所有消息（输入/输出）都会打时间戳：

- 切换信号时默认使用 PLC 的当前系统时间（参数“timestamp”的缺省值）。
- 如果要自定义设置时间戳，可以使用参数“timestamp”创建。
- 时间值务必要设为系统时间（即 UTC），因为在设备范围内执行时间同步时采用这个时间。
- 如果要以本地时间给消息打时间戳，必须预先连接一个能将本地时间换算成系统时间的转换块。只有这样才能保证消息显示中正确显示时间戳。

如要使用 CPU 的当前系统时间，需将参数“timestamp”设为缺省值 (LDT#1970-01-01-00:00:00.0)。

说明

如要使用当前系统时间，需在 OB1 中查询时间并将此时间用于所有报警。如此操作既可缩短运行时间，又能使同一 OB1 循环中的所有报警拥有相同的时间戳。

7.7 报警计数器

SINUMERIK 用户报警共有 16 个计数器可以使用。

如果通过“LSINU_GenerateSinUserAlarm”生成 SINUMERIK 用户报警，这种情况下通过参数“alarmCounter”指定在报警输入或输出时，增加或减少 16 个可用计数器中的哪个。如果参数设为 0，则所有计数器都不记录报警数据。

使用 “LSINU_GetSinuUserAlarmCounters”函数，可以读取 16 个计数器的值。

7.8 规模

最多可同时存在 100 个 SINUMERIK 用户报警。

说明

为了避免 PLC 循环时间出现剧烈波动，一个 PLC 循环中触发的 SINUMERIK 用户报警不要超过 50 个。

7.9 定义 SINUMERIK 用户报警

说明

在 SINUMERIK 用户报警的 AppNote 中，您可以找到 SINUMERIK 用户报警的一些定义示例。AppNote 请参见西门子工业在线支持 (https://support.industry.siemens.com/cs/ww/en/view/109812364)页面。

说明

存在报警时，不可更改“alarmNumber”、“alarmCategory”、“alarmCounter”和“alarmIndex”等参数。因此，必要时可借助起始值定义“报警编号”、“类别”、“计数器”和“报警索引”。

7.10 掉电保持

您可将 SINUMERIK 用户报警声明为掉电保持和非掉电保持。

对于标记掉电保持的报警，注意在引导启动 PLC 后必须调用“LSINU_GenerateSinuUserAlarm”块，以重新生成 SINUMERIK 用户报警。

NC 事件和过程报警

8

8.1 从用户程序访问 NC 信号

NC 信号列表

下表列举介绍可在 PLC 用户程序中访问的所有 NC 信号。

信号类型	传输至用户	NCK 应答	备注
标准辅助功能	循环 OB： “SINU_Update”	下一个循环 OB 中的 “SINU_Update”	-
快速辅助功能 （快速应答的用户自定义辅助功能）	循环 OB： “SINU_Update”	过程报警 OB： “SINU_ProcessNckEvent”（见备注）	快速辅助功能（快速应答的用户自定义辅助功能）是指在将数据传输到用户接口之前，就已在“SINU_ProcessNckEvent”中应答的辅助功能。这种情况下，应答不确认 PLC 用户功能执行完成。 当多个连续零件程序段输出快速辅助功能时，尤其无法保证 PLC 用户程序中的响应与程序段同步进行。这类情况下立即在“SINU_ProcessNckEvent”中应答第一个快速辅助功能。以下快速辅助功能在下一个循环 OB 才进行应答：“SINU_Update”。
G 指令	循环 OB： “SINU_Update”	前往“SINU_Update”开头	-
辅助功能轴信号	循环 OB： “SINU_Update”	前往“SINU_Update”开头	-

信号类型	传输至用户	NCK 应答	备注
定位轴信号	下一次调用下列指令： • "SINU_PositionSpindle" • "SINU_RotateSpindle" • "SINU_OscillateSpindle" • "SINU_TraverseIndexingAxis" • "SINU_TraversePosAxis"	前往“SINU_Update”开头	-
已执行异步子程序中断	下一次调用“SINU_RequestAsup”	前往“SINU_Update”开头	特殊形态。无功能，只确认已执行异步子程序中断。

更多信息

关于 NCK 事件处理的详细信息和示例，参见“SINU_ProcessNckEvent (页 1091)”章节。

8.2 定义 NCK 过程报警事件

在 PLC 用户程序中，通过“SINU_ProcessNckEvent” (页 1091)指令相对于 NCK 处理辅助功能及其应答（传输和程序段转换应答）。

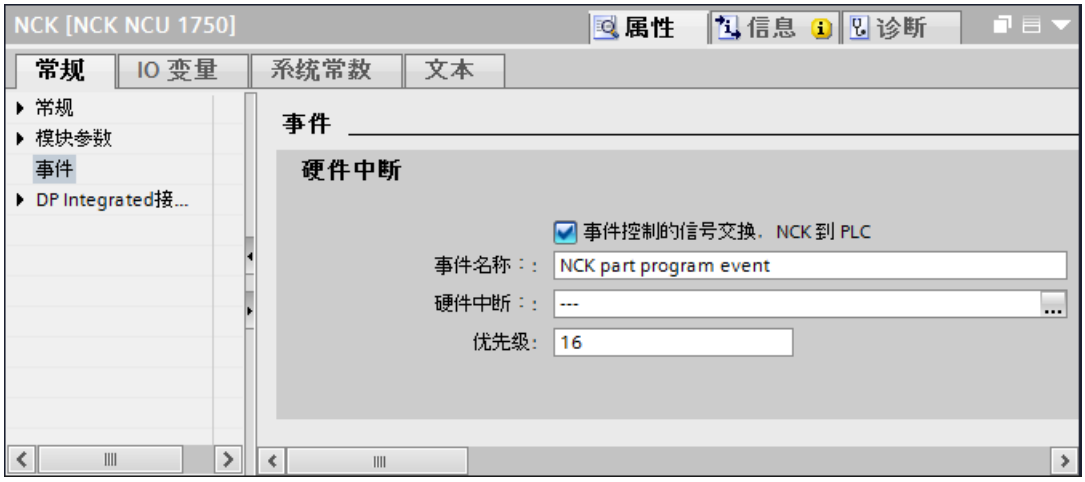
前提条件

- 已添加 NCU/PPU 17x0。

步骤

为了在 NCK 的属性中配置过程报警，按如下步骤操作：

1. 在网络视图或设备视图中点击 NCK。
2. 在检视窗口“属性”中选择条目“事件”。
   显示过程报警设置。事件控制的从 NCK 至 PLC 的信号交换通过预设激活。



3. 进行以下设置：

设置	用途
事件名称	名称，用于例如在与启动事件对应的过程报警 OB 的列表以及系统常量的列表中识别事件。 预设是语言相关的事件名称。可任意调整事件名称，且例如选择与语言无关的名称。
过程报警	从列表选择现有的过程报警 OB 并通过勾号按钮确定，或者，点击“添加”，用以添加新程序块并将其分配给事件。 而在特殊应用情况下（比如模块化机床），适合在用户程序运行期间，使用 Attach 指令分配过程报警。
优先级	定义这个事件相对于其他过程报警 OB 的优先级或等待队列中的优先级。 优先级的有效取值范围参见提示框中的说明。 **此时适用：** • 较小的数值 → 低优先级（例如 2） • 较大的数值 → 高优先级（例如 26）

结果

完成 NCK 过程报警事件属性定义和过程报警 OB 分配。此外，事件在所选择的过程报警 OB 的属性中列出在“启动事件”下。添加了对应的系统常量。

从 PLC 访问 NC 变量 9

9.1 变量组态概述

编程设置 SINUMERIK PLC 时，可以使用 TIA Portal 中的下列变量：

- PLC 变量
  按照 TIA Portal 中的默认操作步骤，声明和使用 PLC 变量。
- NC 变量
  您可在 PLC 程序中，通过"LSINU_ReadVar"和"LSINU_WriteVar"访问已添加到 NC 变量 DB 的 NC 变量。

在 TIA Portal 中使用变量工作时，注意 PLC 变量和 NC 变量的不同数据类型声明。

更多信息

详细信息可在 TIA Portal 在线技术支持界面的“组态 SINUMERIK NC 变量”部分查阅。

参见

NCK 和 PLC 的数据类型声明对应关系 (页 111)

9.2 在 TIA 博途中组态 NC 变量

9.2.1 相关对象和资源

为了在 PLC 用户程序中访问 NC 变量，必须在项目中对所需对象进行配置和组态。为此需要在 TIA Portal 中使用下列资源：

- 用于 SINUMERIK PLC 基本程序 V6.x，具有复制模板的系统库，包含访问所需的 PLC 数据类型。必须将此系统库嵌入您的项目中。
- 借助 NC 变量表，不但能选择和配置 NC 变量，还可对变量进行自定义组织，以及将其添加到 NC 变量 DB。视 NC 变量的组态步骤而定，在概念方面可能存在差异。
- 您可在程序编辑器中创建具有所需 NC 变量访问权限的用户程序。

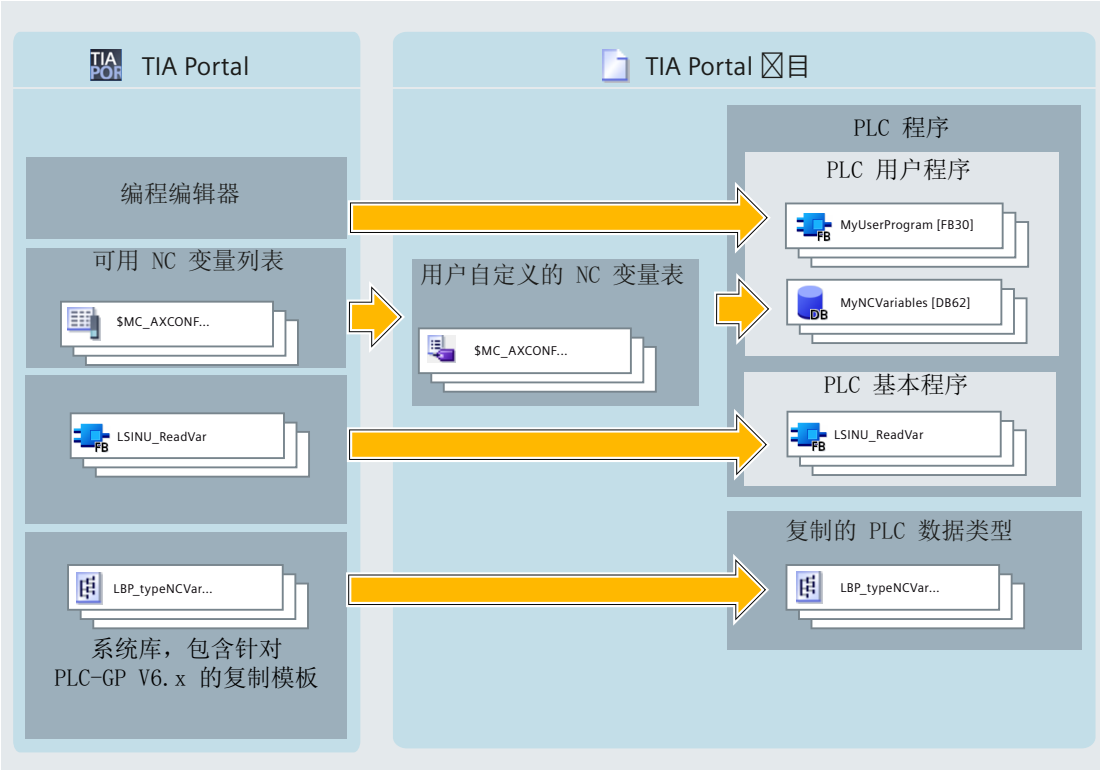
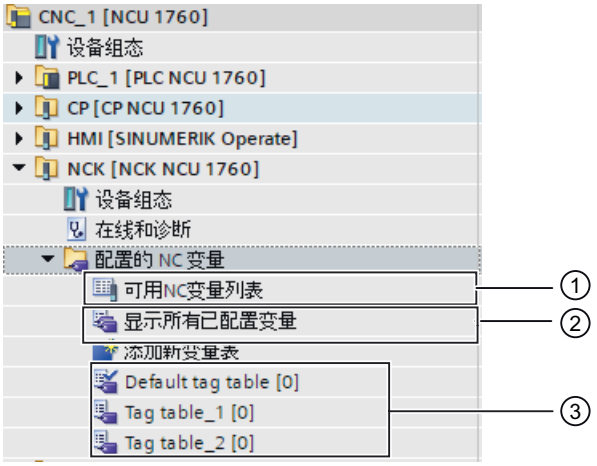


图 9-1　组态 NC 变量

对于在项目中创建的所有 NCU，项目导航区 NCK 条目下都设有“组态的 NC 变量”文件夹，其内包含多个 NC 变量表。

下图所示，为“组态的 NC 变量”文件夹内包含的变量表：



① “可用 NC 变量列表”表

② “组态的所有变量”表

③ 用户自定义变量表

TIA Portal 中的 NC 变量特征

在 TIA Portal 中，NC 变量受当前组态步骤与属性影响，在不同的时间点具有不同的特征（名称和属性不同）：

	常规（可用）NC 变量	组态的 NC 变量	DB/FB 中的 NC 变量
位置	在可用 NC 变量列表中	在用户自定义变量表和可用 NC 变量列表中	在 DB 或 FB 中
名称	NC 上下文中的变量名称，即下属 NC 变量和系统变量（可能有）的名称。	用户自定义名称	用户自定义名称
数据类型	NC 声明中的规定	PLC 声明中的规定	与 PLC 基本程序的 PLC 数据类型关联在一起
组态	不输入特殊信息。	组态有特殊信息，比如所需 NCK 对象、通道或轴等的相关信息。	有从 PLC 程序访问用的特殊令牌。
	-	-	变量值元素

9.2.2 使用 NC 变量表工作

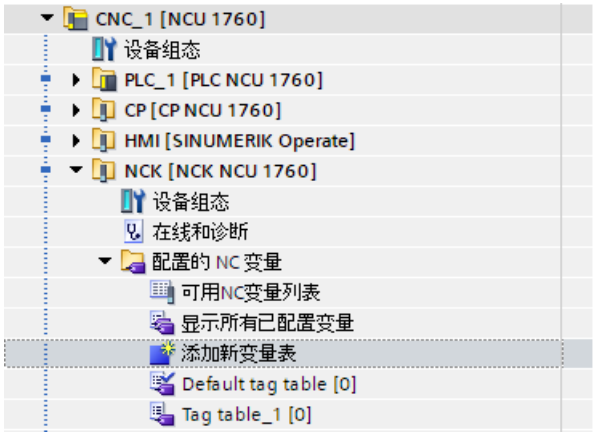
9.2.2.1 创建和管理用户自定义变量表

创建变量表

按下列步骤新建 NC 变量表：

1. 在项目导航区的"CNC_1 > NCK > 组态的 NC 变量”等路径下，打开“组态的 NC 变量”文件夹。
2. 双击“添加新变量表”指令。

3. 为新变量表输入具有唯一性的名称。



4. 按下软键“确认”确认输入。

打开变量表

按下列步骤打开 NC 变量表：

1. 在项目导航区的“CNC_1 > NCK > 组态的 NC 变量”等路径下，打开“组态的 NC 变量”文件夹。
2. 在文件夹中双击 NC 变量表。

复制变量表

按下列步骤复制 NC 变量表：

1. 选择要复制的 NC 变量表。
2. 在快捷菜单中点击“复制”指令。
3. 将插入光标定位在要插入 NC 变量表的位置。
   可将 NC 变量表插入同一 NCU 或其他 NCU 的“组态的 NC 变量”文件夹下。
4. 在快捷菜单中点击“插入”指令。
   - 如果将 NC 变量表插入同一文件夹，会插入添加“_<连续编号>”扩展名的副本。
   - 如果将 NC 变量表插入其他文件夹，而此文件夹内已经有同名变量表，这种情况下插入的变量表会添加“_<连续编号>”扩展名。

也可按住 <Ctrl> 键拖放，以此达到复制 NC 变量表的目的。

9.2.2.2 管理变量表中的条目

复制条目

说明

兼容性检查

将变量复制到其他系统的变量表中时，插入时会检查是否兼容，如不兼容则取消插入。

按下列步骤复制变量：

1. 选择要复制的变量。
   按住 <Ctrl> 键依次点击多个变量，或者按住 <Shift> 键点击第一个和最后一个变量，可以选中多个变量。
2. 在快捷菜单中点击“复制”指令。
3. 将插入光标定位在要插入变量的位置。
4. 在快捷菜单中点击“插入”指令。

或者

1. 标记变量。
2. 按住左键。
3. 同时按下 <Ctrl> 键。
4. 将变量拖动至目标位置。

删除条目

按下列步骤删除元素：

1. 选中待删除元素所在的行。按住 <Ctrl> 键依次点击多个行，或者按住 <Shift> 键点击第一行和最后一行，可以选中多个行。
2. 在快捷菜单中点击“删除”指令。

9.2.2.3 调整变量表视图

行排序

按下列步骤给变量表的行排序：

1. 选择排序依据列。
2. 点击列的表头。
   列升序显示。
   箭头向上，说明是升序排列。
   - 点击箭头可以变更排序。
     列降序显示。箭头向下，说明是降序排列。
   - 第三次点击列的表头，可以恢复初始排序。

显示或隐藏列

按下列步骤显示或隐藏变量表的列：

1. 点击某个列的表头。
2. 在快捷菜单中点击“显示/隐藏”指令。
   显示可用列的选项。
   - 勾选列的复选框，可以显示列。
   - 禁用列的复选框，可以隐藏列。
   - 点击“更多”，在“显示/隐藏”对话框中勾选或禁用相应列的复选框，可以显示或隐藏多个列。

9.2.3 使用包含可用 NC 变量的列表工作

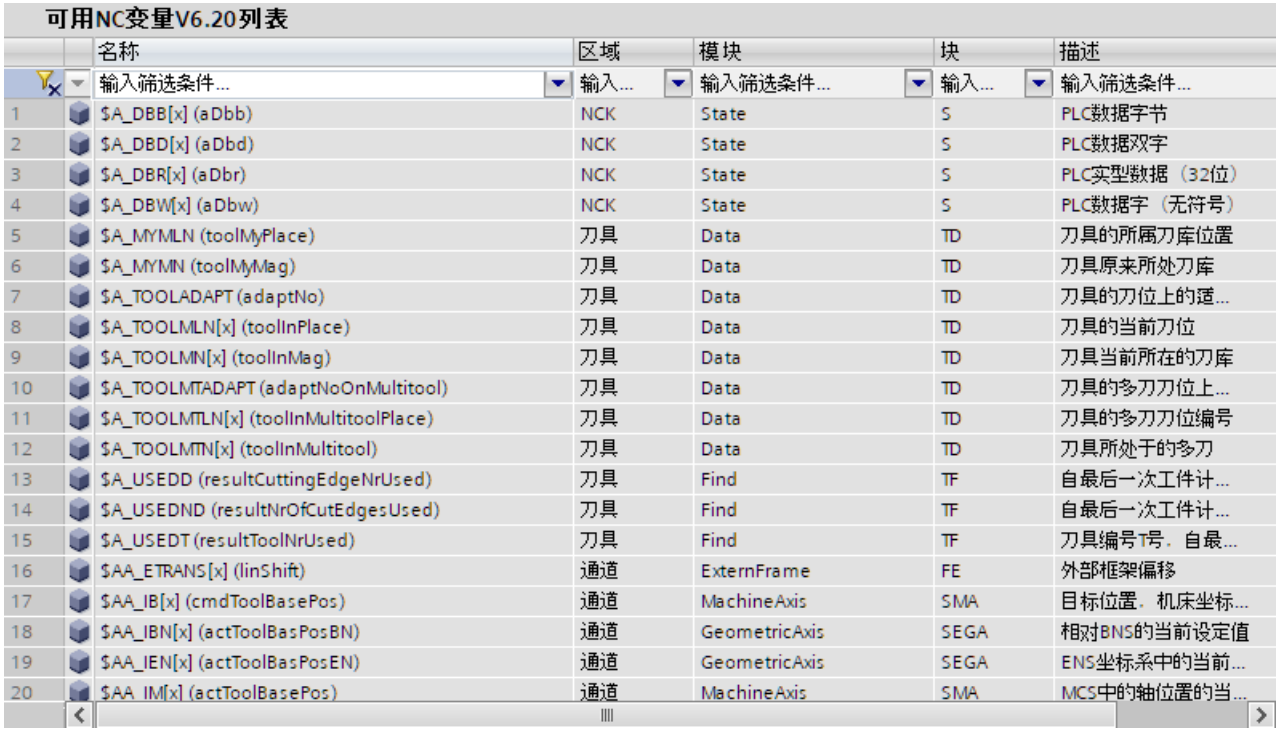
9.2.3.1 搜索和筛选 NC 变量

在项目导航区内 NCK 下找到“可用 NC 变量列表”，此列表包含所有可与组态的 NCU 固件组合使用的 NC 变量。

默认所有可用变量都显示在 NC 变量列表中。这个包含完整数据的列表不但具备筛选功能，也支持搜索，可以帮助您轻松找到所需变量，然后复制到自己的变量表中。

筛选

您可以设置不同的筛选条件，在 NC 变量列表中筛选 PLC 程序需要的 NC 变量。为此，请点击各列的“输入筛选器”。



可用NC变量V6.20列表

	名称	区域	模块	块	描述
	输入筛选条件...	输入...	输入筛选条件...	输入...	输入筛选条件...
1	$A_DBB[x] (aDbb)	NCK	State	S	PLC数据字节
2	$A_DBD[x] (aDbd)	NCK	State	S	PLC数据双字
3	$A_DBR[x] (aDbr)	NCK	State	S	PLC实型数据（32位）
4	$A_DBW[x] (aDbw)	NCK	State	S	PLC数据字（无符号）
5	$A_MYMLN (toolMyPlace)	刀具	Data	TD	刀具的所属刀库位置
6	$A_MYMN (toolMyMag)	刀具	Data	TD	刀具原来所处刀库
7	$A_TOOLADAPT (adaptNo)	刀具	Data	TD	刀具的刀位上的适...
8	$A_TOOLMLN[x] (toolInPlace)	刀具	Data	TD	刀具的当前刀位
9	$A_TOOLMN[x] (toolInMag)	刀具	Data	TD	刀具当前所在的刀库
10	$A_TOOLMTADAPT (adaptNoOnMultitool)	刀具	Data	TD	刀具的多刀刀位上...
11	$A_TOOLMTLN[x] (toolInMultitoolPlace)	刀具	Data	TD	刀具的多刀刀位编号
12	$A_TOOLMTN[x] (toolInMultitool)	刀具	Data	TD	刀具所处于的多刀
13	$A_USEDD (resultCuttingEdgeNrUsed)	刀具	Find	TF	自最后一次工件计...
14	$A_USEDND (resultNrOfCutEdgesUsed)	刀具	Find	TF	自最后一次工件计...
15	$A_USEDT (resultToolNrUsed)	刀具	Find	TF	刀具编号T号，自最...
16	$AA_ETRANS[x] (linShift)	通道	ExternFrame	FE	外部框架偏移
17	$AA_IB[x] (cmdToolBasePos)	通道	MachineAxis	SMA	目标位置，机床坐标...
18	$AA_IBN[x] (actToolBasPosBN)	通道	GeometricAxis	SEGA	相对BNS的当前设定值
19	$AA_IEN[x] (actToolBasPosEN)	通道	GeometricAxis	SEGA	ENS坐标系中的当前...
20	$AA_IM[x] (actToolBasePos)	通道	MachineAxis	SMA	MCS中的轴位置的当...

视列类型而定，有下面几种筛选方式：

筛选条件	步骤
名称	输入用户自定义筛选文本并按回车键确认。 这个字段可能包含同一变量的多个名称（系统变量名称或 NC 变量名称），搜索时需要注意这一点。 注意：通过 NC 变量名称分配 NC 变量，无法始终保证明确性。必要时附加显示“区域”列和“模块”列。
区域	在筛选单元格的下拉列表中选择所需区域。
模块	输入用户自定义筛选文本并按回车键确认。
描述	输入包含在待筛选描述中的用户自定义筛选文本，然后按回车键确认。 使用通配符 *，可以筛选多个不连续的字词。

筛选条件	步骤
数据类型	在筛选单元格的下拉列表中选择一种数据类型（NCK 声明）进行筛选。
访问权限	在筛选单元格的下拉列表中选择一种访问权限类型进行筛选：只读、读/写或 *（全部）。

您可以在不同列定义多个同时有效的筛选条件。

选中文本，按删除键，再按回车键确认，即可删除筛选条件。

显示或隐藏列

按下列步骤显示或隐藏变量表的列：

1. 点击某个列的表头。
2. 在快捷菜单中点击“显示/隐藏”指令。
   显示可用列的选项。
   - 勾选列的复选框，可以显示列。
   - 禁用列的复选框，可以隐藏列。
   - 点击“更多”，在“显示/隐藏”对话框中勾选或禁用相应列的复选框，可以显示或隐藏多个列。

行排序

按下列步骤给变量表的行排序：

1. 选择排序依据列。
2. 点击列的表头。
   列升序显示。
   箭头向上，说明是升序排列。
3. 点击箭头可以变更排序。
   列降序显示。箭头向下，说明是降序排列。
4. 第三次点击列的表头，可以恢复初始排序。

9.2.3.2 将 NC 变量传输到“组态的 NC 变量”变量表中

在项目导航区内 NCK 下找到“可用 NC 变量列表”，此列表包含所有可与组态的 NCU 固件组合使用的 NC 变量。

您可以使用项目中的用户自定义 NC 变量表，管理需要在 PLC 用户程序中访问的 NC 变量。

前提条件

- 有用户自定义 NC 变量表 (页 97)（或者使用标准变量表）。

步骤

1. 在“CNC_1 > NCK > 组态的 NC 变量”等路径下双击“可用 NC 变量列表”。
   列表打开。

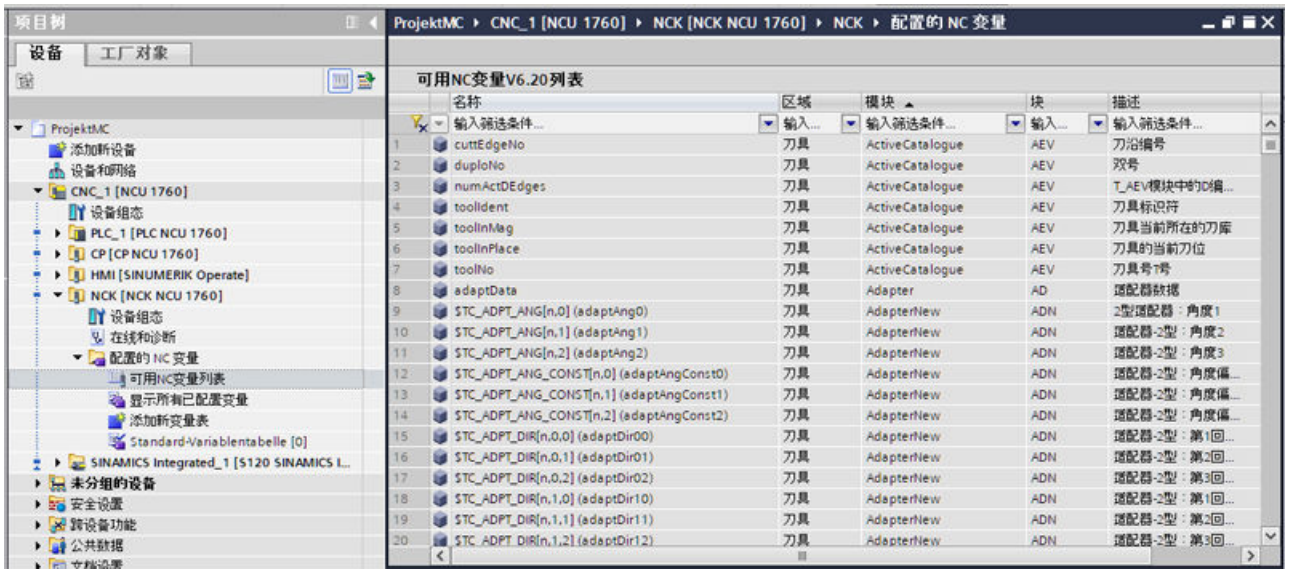
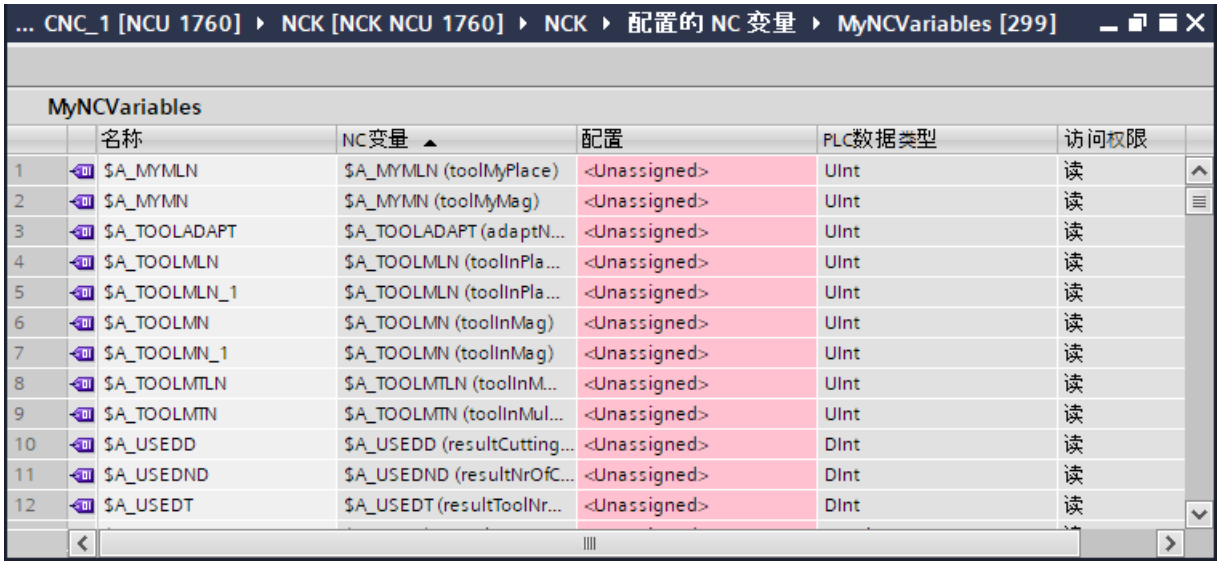


图 9-2　可用 NC 变量列表

2. 选择所需的 NC 变量。
   – 可以在 NC 变量列表中筛选和搜索变量 (页 100)，以此来查找所需变量。
   – 可以操作 Shift 键或 Ctrl 键选择多个 NC 变量以同时复制。
3. 通过复制和插入，或者拖放操作，将选中的 NC 变量复制到变量表中。

结果

选择的 NC 变量已汇总在用户自定义变量表中，随后可进行组态 (页 104)。



	名称	NC变量	配置	PLC数据类型	访问权限
1	$A_MYMLN	$A_MYMLN (toolMyPlace)	<Unassigned>	UInt	读
2	$A_MYMN	$A_MYMN (toolMyMag)	<Unassigned>	UInt	读
3	$A_TOOLADAPT	$A_TOOLADAPT (adaptN...	<Unassigned>	UInt	读
4	$A_TOOLMLN	$A_TOOLMLN (toolInPla...	<Unassigned>	UInt	读
5	$A_TOOLMLN_1	$A_TOOLMLN (toolInPla...	<Unassigned>	UInt	读
6	$A_TOOLMN	$A_TOOLMN (toolInMag)	<Unassigned>	UInt	读
7	$A_TOOLMN_1	$A_TOOLMN (toolInMag)	<Unassigned>	UInt	读
8	$A_TOOLMTLN	$A_TOOLMTLN (toolInM...	<Unassigned>	UInt	读
9	$A_TOOLMTN	$A_TOOLMTN (toolInMul...	<Unassigned>	UInt	读
10	$A_USEDD	$A_USEDD (resultCutting...	<Unassigned>	DInt	读
11	$A_USEDND	$A_USEDND (resultNrOfC...	<Unassigned>	DInt	读
12	$A_USEDT	$A_USEDT (resultToolNr...	<Unassigned>	DInt	读

参见

将组态的 NC 变量复制到现有 DB/FB 中 (页 110)

9.2.4 组态选择的 NC 变量实例

将所有可用 NC 变量列表中的变量传输到用户自定义变量表之后，对选中的 NC 变量进行必要和可选（如有必要）的设置。如果缺少变量设置或设置错误，表格会以彩色形式高亮显示相关位置。

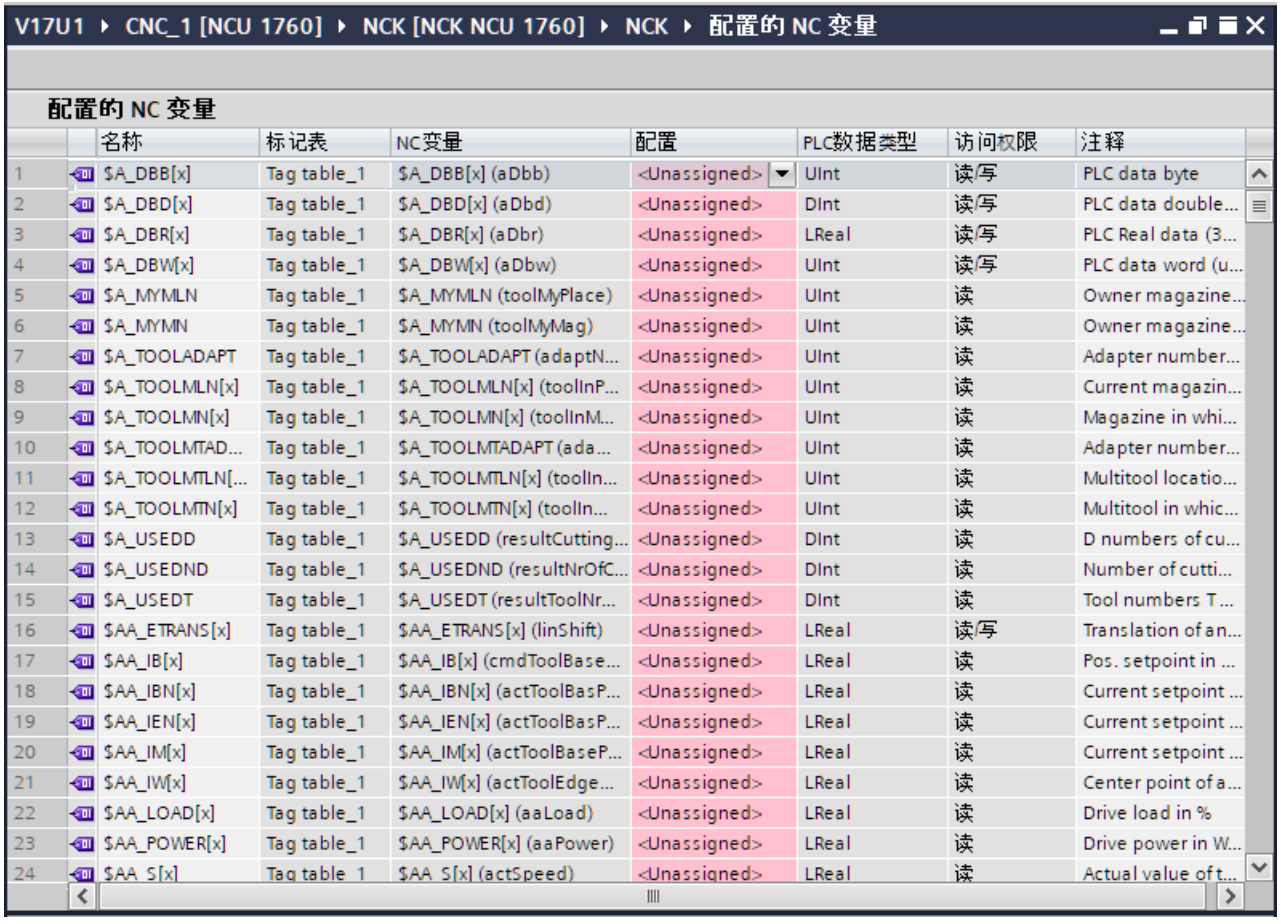
前提条件

- 待组态变量示例已汇总到用户自定义 NC 变量表 (页 97)中。

步骤

按下列步骤组态 NC 变量：

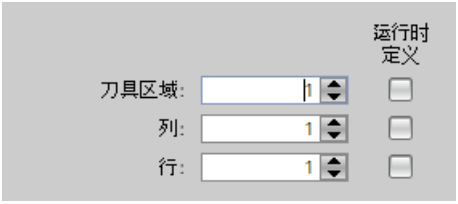
1. 在项目导航区内“组态的 NC 变量”文件夹下双击“显示组态的所有变量”指令，比如“CNC_1 > NCK > 组态的 NC 变量 > 显示组态的所有变量”。



	名称	标记表	NC变量	配置	PLC数据类型	访问权限	注释
1	$A_DBB[x]	Tag table_1	$A_DBB[x] (aDbb)	<Unassigned>	UInt	读/写	PLC data byte
2	$A_DBD[x]	Tag table_1	$A_DBD[x] (aDbd)	<Unassigned>	DInt	读/写	PLC data double...
3	$A_DBR[x]	Tag table_1	$A_DBR[x] (aDbr)	<Unassigned>	LReal	读/写	PLC Real data (3...
4	$A_DBW[x]	Tag table_1	$A_DBW[x] (aDbw)	<Unassigned>	UInt	读/写	PLC data word (u...
5	$A_MYMLN	Tag table_1	$A_MYMLN (toolMyPlace)	<Unassigned>	UInt	读	Owner magazine...
6	$A_MYMN	Tag table_1	$A_MYMN (toolMyMag)	<Unassigned>	UInt	读	Owner magazine...
7	$A_TOOLADAPT	Tag table_1	$A_TOOLADAPT (adaptN...	<Unassigned>	UInt	读	Adapter number...
8	$A_TOOLMLN[x]	Tag table_1	$A_TOOLMLN[x] (toolInP...	<Unassigned>	UInt	读	Current magazin...
9	$A_TOOLMN[x]	Tag table_1	$A_TOOLMN[x] (toolInM...	<Unassigned>	UInt	读	Magazine in whi...
10	$A_TOOLMTAD...	Tag table_1	$A_TOOLMTADAPT (ada...	<Unassigned>	UInt	读	Adapter number...
11	$A_TOOLMTLN[...	Tag table_1	$A_TOOLMTLN[x] (toolIn...	<Unassigned>	UInt	读	Multitool locatio...
12	$A_TOOLMTN[x]	Tag table_1	$A_TOOLMTN[x] (toolIn...	<Unassigned>	UInt	读	Multitool in whic...
13	$A_USEDD	Tag table_1	$A_USEDD (resultCutting...	<Unassigned>	DInt	读	D numbers of cu...
14	$A_USEDND	Tag table_1	$A_USEDND (resultNrOfC...	<Unassigned>	DInt	读	Number of cutti...
15	$A_USEDT	Tag table_1	$A_USEDT (resultToolNr...	<Unassigned>	DInt	读	Tool numbers T...
16	$AA_ETRANS[x]	Tag table_1	$AA_ETRANS[x] (linShift)	<Unassigned>	LReal	读/写	Translation of an...
17	$AA_IB[x]	Tag table_1	$AA_IB[x] (cmdToolBase...	<Unassigned>	LReal	读	Pos. setpoint in ...
18	$AA_IBN[x]	Tag table_1	$AA_IBN[x] (actToolBasP...	<Unassigned>	LReal	读	Current setpoint ...
19	$AA_IEN[x]	Tag table_1	$AA_IEN[x] (actToolBasP...	<Unassigned>	LReal	读	Current setpoint ...
20	$AA_IM[x]	Tag table_1	$AA_IM[x] (actToolBaseP...	<Unassigned>	LReal	读	Current setpoint ...
21	$AA_IW[x]	Tag table_1	$AA_IW[x] (actToolEdge...	<Unassigned>	LReal	读	Center point of a...
22	$AA_LOAD[x]	Tag table_1	$AA_LOAD[x] (aaLoad)	<Unassigned>	LReal	读	Drive load in %
23	$AA_POWER[x]	Tag table_1	$AA_POWER[x] (aaPower)	<Unassigned>	LReal	读	Drive power in W...
24	$AA_S[x]	Tag table_1	$AA_S[x] (actSpeed)	<Unassigned>	LReal	读	Actual value of t...

2. 依次选择变量并进行必要的组态，必要时执行其他调整：

属性	说明
名称	可自定义更改的符号名称（假名），您可通过此名称在 PLC 程序中对变量进行寻址。 默认设置为机床数据编号、系统变量名称或 OPI 名称，具体取决于变量类型。 如果多次插入同一 NC 变量，默认会在名称后面添加一个数字。
NC 变量	NCK 中的变量，PLC 变量示例以 NC 变量为基础。 显示系统变量名称或 NC 变量名称，具体取决于变量类型。

属性	说明
组态 （必要时为必选项）	打开适用于所选 NC 变量或相关 NCK 对象，包含必要设置的对话框。  必须根据变量类型，在这里指定变量基于哪个通道、哪个轴、哪个刀具范围或刀具编号。 这些数据可在组态时作为常量值输入，也可勾选复选框，在运行期间通过 LSINU_ReadVar (页 1313) 或 LSINU_WriteVar (页 1331) 中的变量寻址形式参数进行定义。 可以直接在单元格中编辑单元格内容。 各变量详细信息参见相关帮助界面。
PLC 数据类型	显示 PLC 声明中变量的数据类型。
访问权限	显示变量可否写入，还是只能读取。 注意：只在 NCK 侧检查访问权限，PLC 侧不检查。在 PLC 程序中，即便相关 NC 变量无法写入，也会将未授权的写入请求传输到接口。
注释	变量简要信息。 可出于变量应用归档等目的进行自定义编辑。

结果

NC 变量完成组态。可根据底层变量表生成新 DB，或将 NC 变量添加到现有 DB/FB。

参见

将组态的 NC 变量复制到现有 DB/FB 中 (页 110)

9.2.5 嵌入 PLC 程序中

9.2.5.1 在 PLC 程序中读/写 NC 变量

要在用户程序中使用 PLC 基本程序的 NC 变量接口，需要在用户程序中嵌入相应的 PLC 基本程序块调用：

- LSINU_ReadVar
- LSINU_WriteVar

在块调用的形式参数中，指定一个或多个之前组态的访问令牌，以及读/写访问的数据目标，必要时指定其他参数。

从图中两个示例可以看出，所需的用户程序组成部分既可分布在同一块内，也可分布在不同块内。详细信息参见相关 PLC 基本程序块 LSINU_ReadVar (页 1313) 和 LSINU_WriteVar (页 1331) 的描述内容。

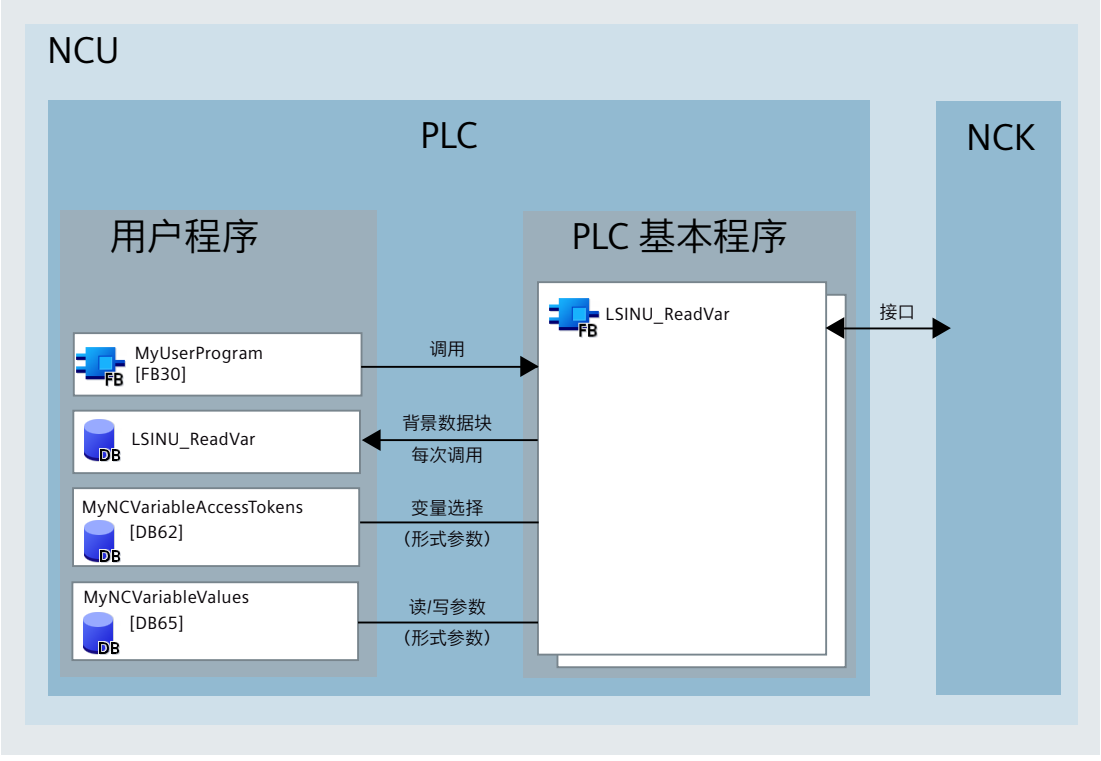


图 9-3 通过 PLC 基本程序访问 NC 变量 - 多 DB 示例

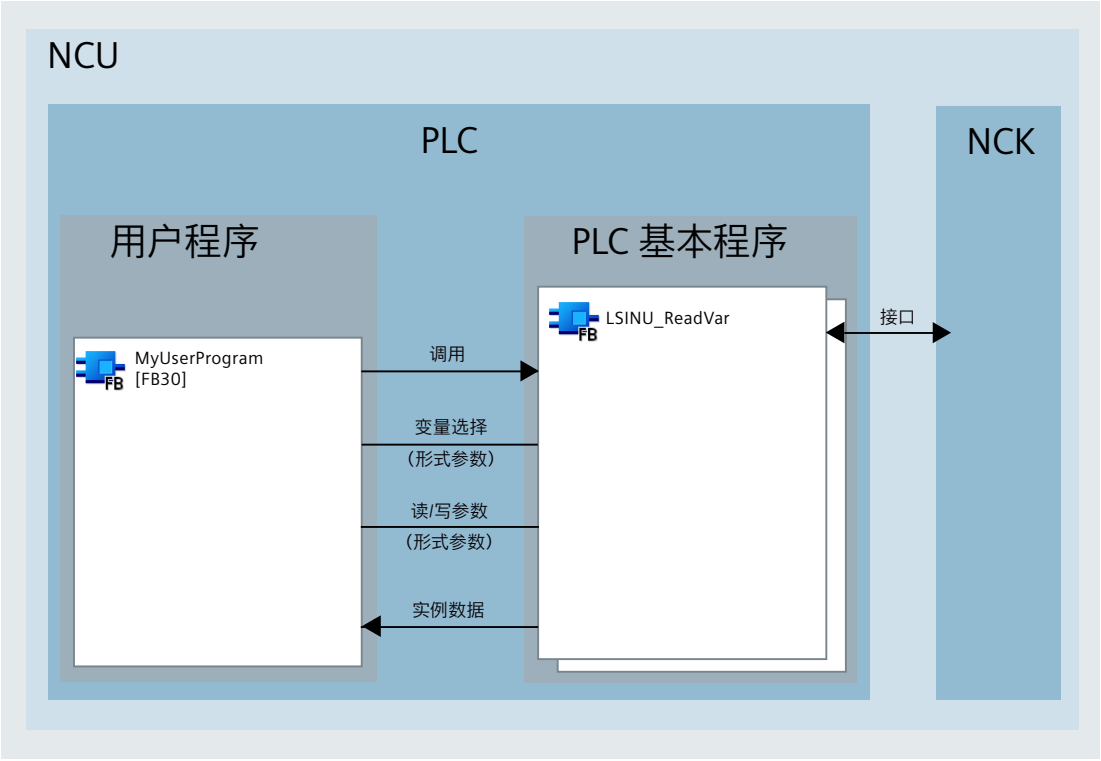


图 9-4　通过 PLC 基本程序访问 NC 变量 - 单 DB 示例

9.2.5.2　使用 NC 变量表创建数据块

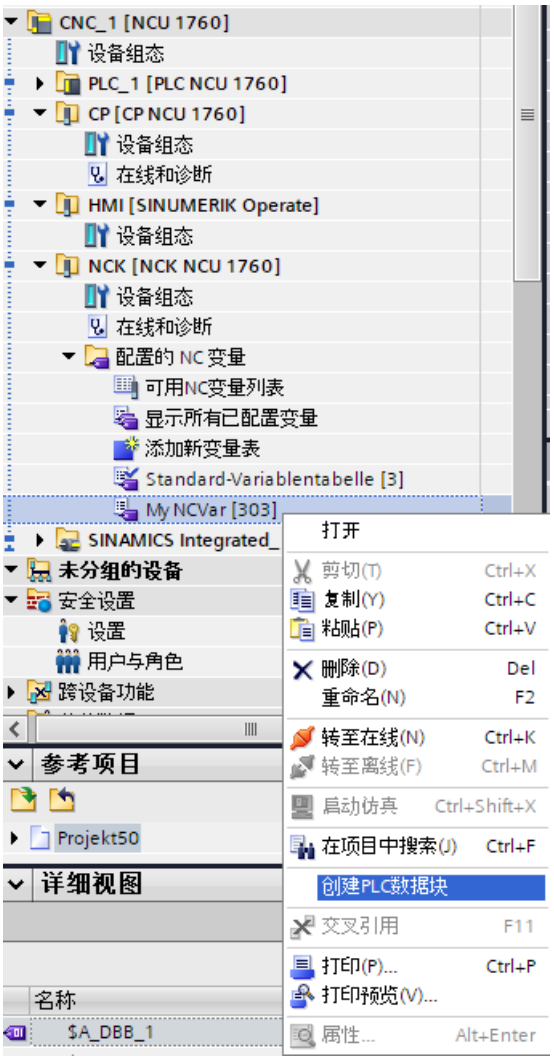
前提条件

- 已有一个用户自定义 NC 变量表。
- 用户自定义 NC 变量表中的所有 NC 变量完整且已正确组态。
- 已添加 NC 变量的数据类型。

步骤

按下列步骤，用 NC 变量表生成数据块：

1. 在项目导航区打开"NC 变量"文件夹，路径示例：CNC_1 > NCK > NC 变量。
2. 右击变量表并选择"创建 PLC 数据块"指令。

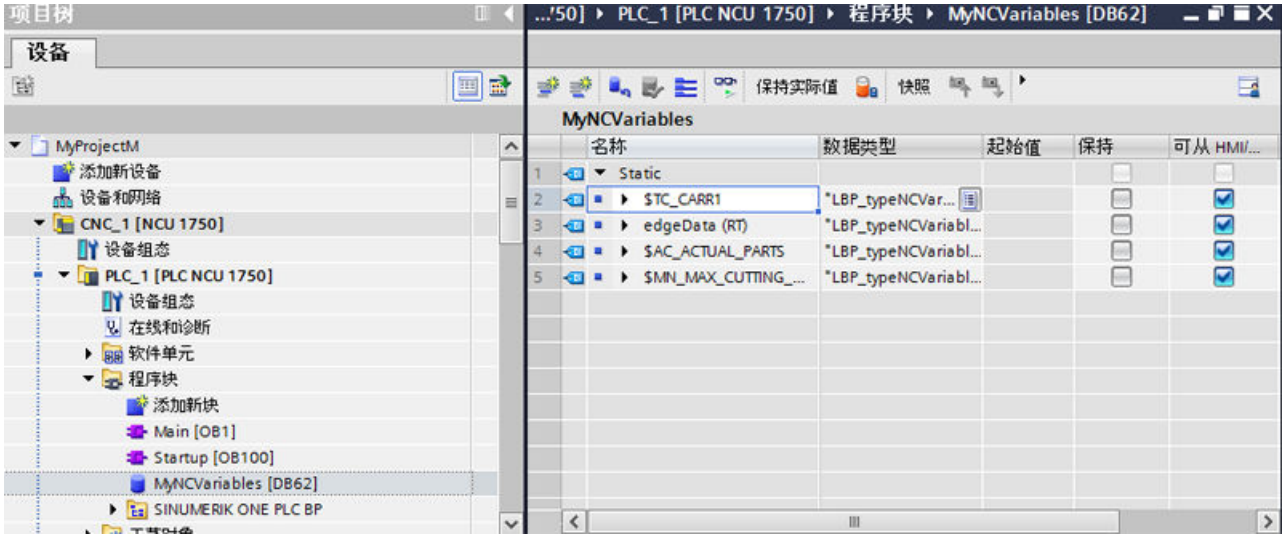


说明

PLC 数据块名称具有唯一性。如果已经存在同名 PLC 数据块，此数据块会被覆盖。覆盖前需要您确认。

结果

生成数据块并添加到程序块文件夹内。使用底层变量表的名称作为块名称。



现在您可以使用生成的 DB，在 PLC 程序中为汇总的变量进行寻址 (页 107)，并通过块 LSINU_WriteVar (页 1331) 和 LSINU_ReadVar (页 1313) 读写变量。

9.2.5.3 将组态的 NC 变量复制到现有 DB/FB 中

前提条件

- 待嵌入 NC 变量完整且已正确组态。
- 已添加 NC 变量的数据类型。
- 已设置具有写访问权限的 DB/FB 作为 NC 变量的目标位置。
- 目标系统与待嵌入 NC 变量兼容。

说明

文本块接口无初始值（令牌常量）

如果为 SCL 设置了可激活的块接口文本显示模式，则无法为 FB 中嵌入的 NC 变量生成令牌常量。

也可使用表格显示模式。

步骤

按下列步骤将完成组态的 NC 变量嵌入现有 DB 或 FB 中：

1. 在项目导航区打开相应的用户自定义变量表，或者点击“显示组态的所有变量”，路径示例：CNC_1 > NCK > NC 变量。
2. 选择要复制的所有变量。
3. 将变量嵌入 DB/FB 步骤示例：
   – 右击某个选中的 NC 变量，点击“复制”，然后在项目导航区右击 DB/FB，点击“插入”。
   – 通过拖放操作，将选中的变量拖动至项目导航区的 DB/FB 处。

结果

NC 变量嵌入 DB/FB 中。

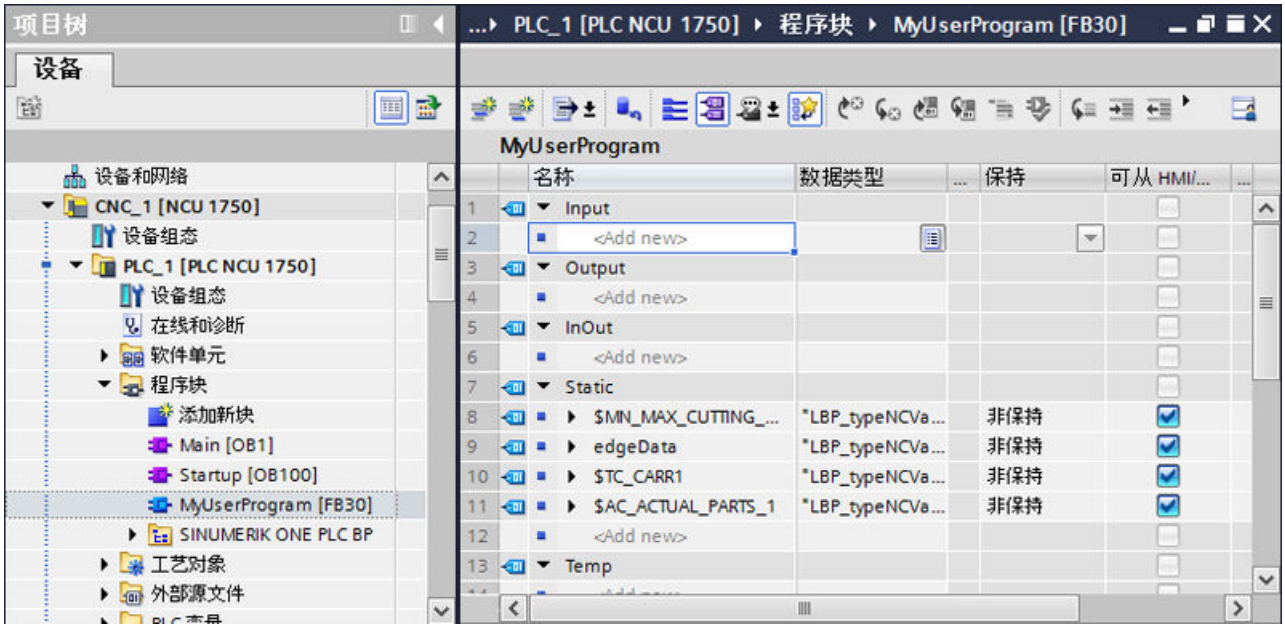


图 9-5　包含 NC 变量的 MyUserProgram [FB30] 的块接口示例

现在您可以使用此 DB/FB，在 PLC 程序中为汇总的变量进行寻址 (页 107)，并通过块 LSINU_WriteVar (页 1331) 和 LSINU_ReadVar (页 1313) 读写变量。

9.2.5.4　NCK 和 PLC 的数据类型声明对应关系

数据类型决定数据属性，比如内容显示模式和允许的存储区等。

NC 变量使用其他数据类型声明作为 STEP 7 变量（PLC 程序）。NC 变量列表显示 NCK 声明中的数据类型。如果在变量表中插入 NC 变量并在 PLC 程序中使用，这种情况下必须使用 PLC 声明。

表格 9-1　NCK 和 STEP 7 (PLC) 数据类型声明分配

数据类型		**格式**	**取值范围**	**简要说明**
NCK 声明	**PLC 声明**			
BOOL	BOOL	无符号整数（十进制）	0 或 1，或者 FALSE 或 TRUE	一个 BOOL 型操作数占一个比特位。
		二进制数	2#0 或 2#1	
		八进制数	8#0 或 8#1	
		十六进制数	16#0 或 16#1	
BYTE	USINT	无符号整数（十进制）	0 至 255	USINT 型操作数长度为 8 比特位，包含无符号数值。
		二进制数	2#0 至 2#01111_1111	
		八进制数	8#0 至 8#377	
		十六进制数	16#0 至 16#FF	
DATETIME	DT (DATE_AND_TIME)	日期和时间 （年-月-日-时:分:秒:毫秒）	最小： DT#1990-01-01-00:00:00.000 最大： DT#2089-12-31-23:59:59.999	数据类型 DT (DATE_AND_TIME) 以 BCD 格式保存日期和时间信息。
DOUBLE	LREAL	IEEE754 定义的浮点数	-1.7976931348623157e+308 至 -2.2250738585072014e-308 ±0.0 +2.2250738585072014e-308 至 +1.7976931348623157e+308	LREAL 型操作数长度为 64 比特位，用于显示浮点数。

数据类型		格式	取值范围	简要说明
NCK 声明	PLC 声明			
INTEGER	DINT	带符号整数（十进制）	-2_147_483_648 至 +2_147_483_647	DINT（双整数）型操作数长度为 32 比特位，由两部分构成：符号和数值（二进制数补码形式）。
		二进制数（仅正数）	2#0 至 2#0111_1111_1111_1111_1111_1111_1111_1111	
		八进制数（仅正数）	8#0 至 8#177_7777_7777	
		十六进制数	16#0 至 16#7FFF_FFFF	
字符串	字符串	ASCII 字符串含特殊字符	0 至 254 个字符	STRING 型操作数在一个字符串中保存多个字符，字符串最多可由 254 个字符构成。
DWORD	UDINT	无符号整数（十进制）	0 至 4_294_967_295	UDINT 型操作数长度为 32 比特位，包含无符号数值。
		二进制数	2#0 至 2#1111_1111_1111_1111_1111_1111_1111_1111	
		八进制数	8#0 至 8#377_7777_7777	
		十六进制数	16#0 至 16#FFFF_FFFF	

数据类型		格式	取值范围	简要说明
NCK 声明	PLC 声明			
UINT_16	UINT	无符号整数（十进制）	0 至 65_535	UINT（无符号整数）型操作数长度为 16 比特位，包含无符号数值。在内存中，一个 UINT 型操作数占用两个字节。
		二进制数	2#0 至 2#1111_1111_1111_1111	
		八进制数	8#0 至 8#17_7777	
		十六进制数	16#0 至 16#FFFF	

另见：LSINU_ReadVar (页 1313)、LSINU_WriteVar (页 1331)、STEP 7 帮助中的有效数据类型概览相关内容。

参见

使用 NC 变量表工作 (页 97)

9.2.6 管理库中的 NC 变量

NC 变量与全局库兼容。您可以将下列包含 NC 变量的对象作为复制模板存储在库中：

- 含 NC 变量的 DB/FB
- 用户自定义 NC 变量表

这样就可以在其他项目中继续使用用户自定义组态的 NC 变量实例。插入复制模板时，检查与目标设备中的新控制器是否兼容。

配置/组态 10

10.1 出现 NC 故障时的处理方式

简介

循环运行期间，PLC 基本程序会通过查询生命符号对 NC 就绪状态进行持续监控。只有当 NC 和 PLC 处于循环运行中，交换过生命符号且 PLC 程序未发生变更时，才会进行更新。如果不是这种情况，所有实例中的数据都会失效，而且不会设置“ncCpuReady”输出。

如果 NC 不响应，会对 NC/PLC 接口进行初始化，同时将“SINU_Update”的“ncCpuReady”输出端设为 FALSE。此外，从 NC 传输至 PLC 和从 PLC 传输至 NC 的信号也被设为初始状态。

PLC 自身保持生效，从而可继续控制机床功能。但此时用户程序将获得将机床切换至安全状态的权限。

从 PLC 发送至 NC 的任务：

用于向 NC 传输任务的 FB 和 FC 不允许通过 PLC 用户程序处理，因为有时会出现错误的反馈信息。引导启动控制器时必须在用户程序中等待，直至“SINU_Update”的“ncCpuReady”输出端设为 TRUE 后才能激活任务（例如：读取 NC 数据）。

10.2 机床控制面板和操作组件

10.2.1 一般信息

通过 PROFINET (PN)、PROFIBUS 或工业以太网 (IE) 协议与机床控制面板 (MCP, MPP) 通信。并非所有 SINUMERIK 设备都支持 PROFIBUS 通信，具体取决于设备。手持操作设备 (HT) 和直接键模块 (OpKey) 通过工业以太网 (IE) 进行通信。

操作组件（比如按键）的输入数据传输到 PLC。PLC（比如 LED）的输出数据传输到操作组件。这些数据通过用户程序进行处理。

10.2.2 通过 PROFINET/PROFIBUS 联结

在 TIA 博途的硬件和网络编辑器中联结机床控制面板。MCP/MPP 模块可在硬件目录下找到，也可从“西门子工业支持中心”下载。

通过 PLC 操作系统配置和建立与操作组件的 PROFINET 连接。

说明

在虚拟 SINUMERIK 中模拟采用标准输入和输出信号的 PROFIBUS-MCP，信号固定在 E0.0 / A0.0 上组态。

示例 PROFINET-MCP

在设备配置中创建 PROFINET-MCP。

借助 PLC 变量表，可将输入及输入映像与对应的 MCP 数据类型关联。这样便实现一种用于以符号方式访问 MCP 的输入和输出的舒适的途径。

10.2.3 通过工业以太网联结

通过工业以太网 (IE) 与机床控制面板 (MCP)、手持操作设备 (HT) 和直接键模块 (OpKey) 通信时，需要使用"SINU_UpdateEthernetMcp"指令。必须为每个需进行通信的操作组件调用"SINU_UpdateEthernetMcp"指令。

详细信息参见*操作组件手册*。

说明

连接参数 MCP 地址、连接 ID、接口 ID 和本地 UDP 端口都是输入参数的一部分。此外，可在输入端上指定其是否是直接键模块。连接参数和输入端"directKeys"只有在 enable = FALSE 时才能更改。可以周期性地更改输入端"disableDataToMcp"，借此不再向操作组件发送数据。

在循环式数据交换中，将参数"toMcp2"的数据发送到操作组件中，从操作组件接收的数据将被写入参数"fromMcp"。数据以设定的时间间隔进行发送和接收。每次有新的数据包可用时，均将设置输出端"newData"。

模块化机床控制面板或其他 HMI 设备提供的数据类型可用于在"fromMcp"和"toMcp"参数中构建输入和输出映像（参见 PLC 数据类型基础程序 (页 50)）。自有类型或变量也可以进行参数化。

循环调用

该指令在每个 PLC 循环内对每个实例调用一次（例如在 Main 组织块中）。该调用应在访问输入数据之前进行。以便为后续程序提供操作组件的当前数据。

10.3 来自/到达操作软件 HMI 的控制信号

10.3.1 概述

SINUMERIK Operate 和 NCK 相关数据类型

SINUMERIK Operate 提供了通过现有的操作掩码实现各种功能的可能性。例如，可以使用 PLC Basic Program plus 中的 NCK 相关数据类型，通过 SINUMERIK Operate 传输或评估运行方式、程序控制、手轮偏移、轴预选等信号。

更多信息

NCK 相关数据类型可用信号的概览请参见“参考 PLC Basic Program plus”一章。

SINUMERIK Operate 可用信号

SINUMERIK Operate 上的一些软键/按键操作可用于约 300 ms 的输入信号范围 (In-Signale)，且可以在 PLC 用户程序中进行处理。下表中列出了必须在 PLC 用户程序中进行重置且不会自动重置的信号。

信号名称	需要在 PLC 用户程序中重置
HMI 概述	
• <Hmi>.basic.in.cancelKeyPressed	X
• <Hmi>.basic.in.mcsWcsSwitched	X
选择运行方式	
• <HmiModeGroup>.basic.in.autoModeRequested	X
• <HmiModeGroup>.basic.in.mdaModeRequested	X
• <HmiModeGroup>.basic.in.jogModeRequested	X
• <HmiModeGroup>.basic.in.teachInRequested	X
• <HmiModeGroup>.basic.in.reposRequested	X
• <HmiModeGroup>.basic.in.refRequested	X
• <HmiModeGroup>.basic.in.inc1Requested .. inc10000Requested	X
• <HmiModeGroup>.basic.in.incVarRequested	X
• <HmiModeGroup>.basic.in.contTraversRequested	X

信号名称	需要在 PLC 用户程序中重置
程序控制	
• <HmiChan>.basic.in.progTestSelected	--
• <HmiChan>.basic.in.configuredStopFuncSelected	--
• <HmiChan>.basic.in.atStopSetOvrZeroSelected	--
• <HmiChan>.basic.in.atStopSetOvrZeroSelected	--
• <HmiChan>.basic.in.handwheelOffsetDrfSelected	--
• <HmiChan>.basic.in.m01AssociatedAuxFuncSelected	--
• <HmiChan>.basic.in.m01Selected	--
• <HmiChan>.basic.in.dryRunSelected	--
• <HmiChan>.basic.in.rapidTraversOvrSelected	--
• <HmiChan>.basic.in.skipBlockLevel0 ..9	--
• <HmiAxis>.basic.in.progTestSuppressionRequested	--
• <HmiAxis>.basic.in.progTestRequested	--
通过 HMI 选择 JOG 轴	
• <hmiChan>.axisSelection.in.selectedKeyActuated	--
干涉保护	
• <hmiNc>.collision.in.machineAreasAutoDeactivated	--
• <hmiNc>.collision.in.toolAreasAutoDeactivated	--
• <hmiNc>.collision.in.fixtureAreasAutoDeactivated	--
• <hmiNc>.collision.in.workpieceAreasAutoDeactivated	--
• <hmiNc>.collision.in.machineAreasJogDeactivated	--
• <hmiNc>.collision.in.toolAreasJogDeactivated	--
• <hmiNc>.collision.in.fixtureAreasJogDeactivated	--
• <hmiNc>.collision.in.workpieceAreasJogDeactivated	--

参见

NCK 相关的数据类型（简要描述） (页 135)

10.3.2 使用 SINUMERIK Operate 时的信号传输

注册和数据块"LSINU_Hmi"

为了在 SINUMERIK Operate 之间进行信号传输，PLC 用户程序中必须有所有与 NCK 相关的数据类型。通过调用全局数据块"LSINU_Hmi"中的指令"SINU_CompleteRegistration"，SINUMERIK Operate 可以获得已注册实例的信息。

更多信息

详细信息请见章节 "SINU_CompleteRegistration (页 1090)"。

10.3.3 机床坐标系 (MCS) 或工件坐标系 (WCS) 中的轴实际值显示

机床或工件坐标系

在 SINUMERIK Operate 的机床基本画面中，可以在机床或工件坐标系中显示轴的位置。通过软键 "机床坐标系实际值" 可请求查看该视图。可以在 PLC 用户程序中进行切换。

在 PLC 用户程序中，切换过程需等待至输入信号"<hmi>.basic.in.mcsWcsSwitched"的上升沿出现，然后将输出信号设为"<Hmi>.basic.out.displayWcsValues" = TRUE。输出信号状态为 TRUE 时，SINUMERIK Operate 中的轴实际值显示切换为工件坐标系。

SINUMERIK Operate 关于所激活的实际值显示的反馈将通过软键或在轴位置窗口中显示。

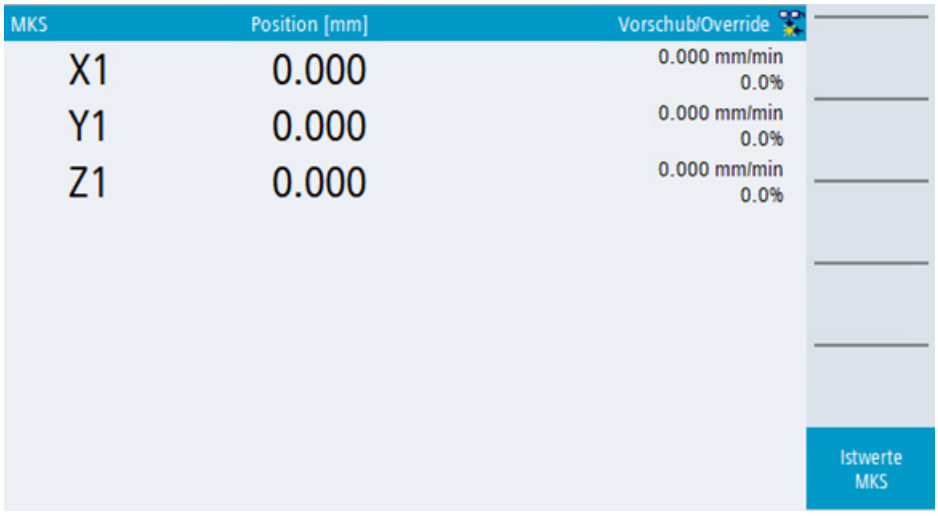


图 10-1 所显示的机床坐标系 (MCS)

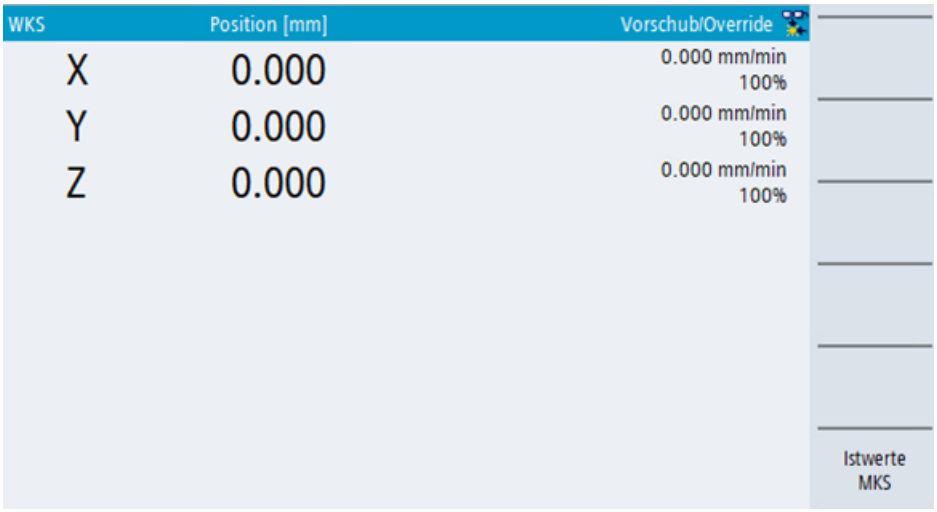


图 10-2　所显示的工件坐标系 (WCS)

说明

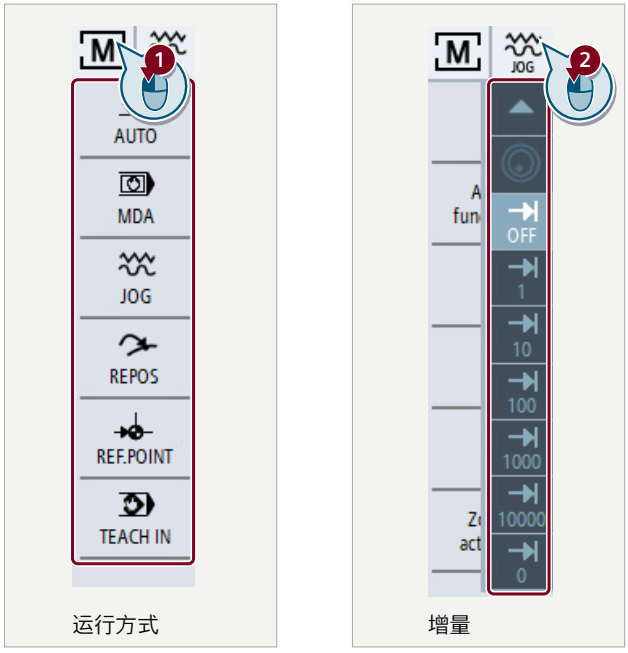
不同实际值之前显示的切换也与“通过 HMI 选择 JOG 轴”功能有关。只能在“工件坐标系”视图中选择几何轴。

10.3.4　运行方式组和增量值的选择

NC 运行方式和针对轴运动的增量

在 SINUMERIK Operate 的机床基本画面中，可以选择 NC 运行方式和轴移动时的增量值。

为了处理这些信号，必须在 PLC 用户项目中创建 NCK 相关数据类型“NCK_typeHmiModeGroup”和“NCK_typeModeGroup”的实例，然后注册并通过指令“SINU_CompleteRegistration”使它们能够被系统识别。NCK 相关数据类型的周期性更新是通过 PLC 项目中的“SINU_Update”指令完成的。



① 可以通过按下“运行方式显示”在“已激活的操作区域和运行方式”中调用相应运行方式。

② 可以通过按下“运行方式显示”在“已激活的操作区域和运行方式”中调用相应增量。

图 10-3 选择运行方式和针对点动模式的增量值

详细信息参见 *SINUMERIK ONE Universal 操作手册*。

编程原则

通过选择运行方式或增量，相应运行方式或增量的请求信号将被传送到实例“NCK_typeHmiModeGroup”的输入端接口上。该请求可以在 PLC 用户程序中与可能存在的 MCP 输入信号并行联结。

一旦 NC 激活了运行方式，SINUMERIK Operate 中的显示也将被激活。

说明

在激活运行方式后，SINUMERIK Operate 的输入信号（按下按钮）必须由 PLC 用户程序进行复位。

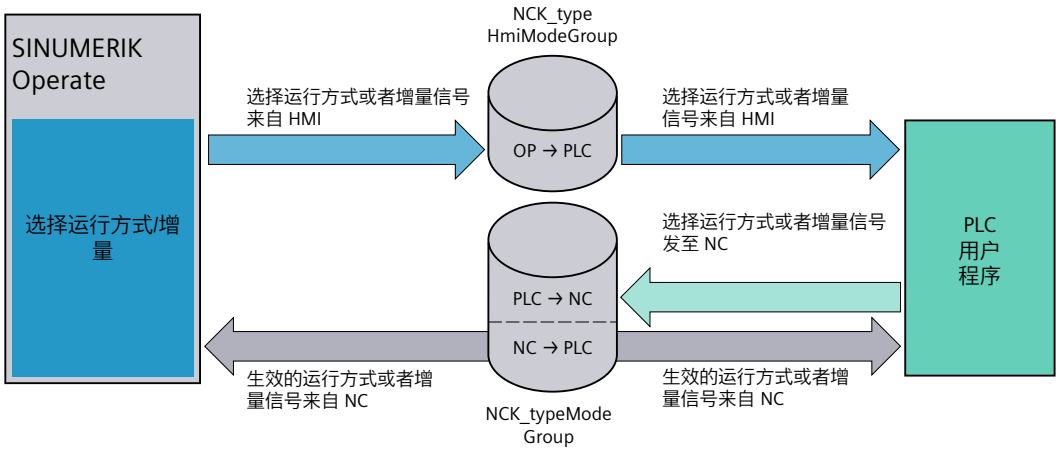


图 10-4　选择运行方式和增量原理图

示例

MDA 运行方式需通过 SINUMERIK Operate 来激活。

已注册运行方式实例的以下输入信号与 PLC 用户程序有关：

信号名称	含义和当前值	信号方向
<HmiModeGroup>.basic.in.mdaModeRequested	请求 MDA 运行方式 （按下 MDA 按键）	OP → PLC
<ModeGroup>.basic.in.mdaModeActive	MDI 运行方式生效	NC → PLC

已注册的运行方式实例的下述输出信号对 PLC 用户程序很重要：

信号名称	含义和当前值	信号方向
<ModeGroup>.basic.out.mdaMode	请求切换至 MDA 运行方式	PLC → NC

输入信号"<HmiModeGroup>.basic.in.mdaModeRequested"直接切换为 PLC 用户程序的输出信号"<ModeGroup>.basic.out.mdaMode"。通过循环调用"SINU_Update"指令，确保所请求的运行方式能够被传送到 NC 上。NC 完成所有检查后（例如运行方式转换抑制），MDA 运行方式将被激活（输入信号"<ModeGroup>.basic.in.mdaModeActive"的信号状态为 TRUE）。应用 SINUMERIK Operate 中的显示。

以与该示例相似的方式，也可以对增量值的选择进行编程。

10.3.5 通道中的程序控制

AUTO 和 MDA 中的程序控制

为了支持 NC 程序的“运行”，SINUMERIK Operate 在程序控制条目下提供了各种选项。

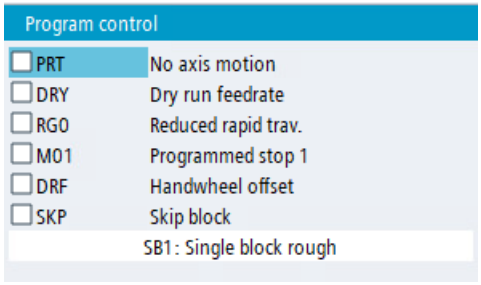


图 10-5 程序控制方案

程序控制的各个功能只有在 SINUMERIK Operate 的“AUTO”（自动）或“MDA”（手动数据输入）运行方式下才有效。

详细信息参见*功能手册：基础功能*中的“程序控制”一章。

编程原则

为了处理这些信号，必须在 PLC 用户项目中针对所需的编辑通道创建 NCK 相关数据类型“NCK_typeHmiChan”和“NCK_typeChan”的实例，然后注册并通过指令“SINU_CompleteRegistration”使它们能够被系统识别。NCK 相关数据类型的周期性更新是通过 PLC 项目中的“SINU_Update”指令完成的。

为了使它在 NC 过程中生效，在 PLC 用户程序中处理来自 SINUMERIK Operate 的 NCK 相关数据类型“NCK_typeHmiChan”的输入信号。将输入数据传输到数据类型为“NCK_typeChan”的输出信号范围中。所选选项只有在 NCK 处理信号后并通过数据类型“NCK_typeChan”的输入信号激活反馈后才会被激活。

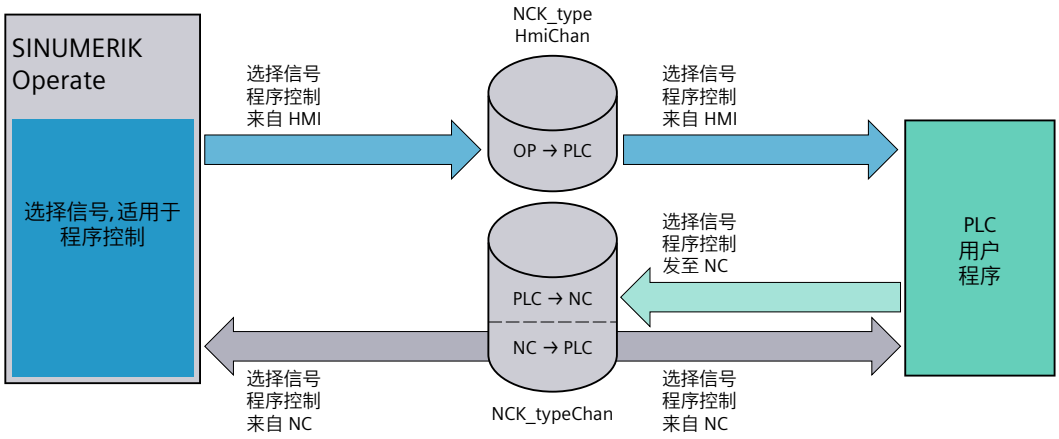


图 10-6　通道中的程序控制原理图

示例

通过 SINUMERIK Operate，将针对第 1 条通道激活程序控制“试运行进给率”(DRY)。

所注册实例（通道 1）的以下输入信号与 PLC 用户程序有关：

信号名称	含义和当前值	信号方向
<HmiChan1>.basic.in.dryRunSelected	已选试运行进给率 (DRY) （重新按下按键）	OP → PLC
<Chan1>.basic.in.dryRunFeedrateActive	试运行进给率处于生效状态	NC → PLC

所注册实例（通道 1）的以下输出信号与 PLC 用户程序有关：

信号名称	含义和当前值	信号方向
<Chan1>.basic.out.dryRunFeedrate	请求激活试运行进给率 (DRY)	PLC → NC

输入信号“<HmiChan1>.basic.in.dryRunSelected”直接切换为 PLC 用户程序的输出信号“<Chan1>.basic.out.dryRunFeedrate”。通过循环调用“SINU_Update”指令，确保所请求的程序控制能够被传送到 NC 上。NC 完成所有检查后，针对通道 1 的程序控制将被激活（输入信号“<Chan1>.basic.in.dryRunFeedrate”的信号状态为 log. "1"）。

SINUMERIK Operate 中的复选框已激活。

图 10-7　试运行进给率有效

以与该示例相似的方式，也可以对程序控制的其它选件进行编程。

说明

请注意，程序控制要根据具体的通道来实现。

10.3.6 选择加工轴/几何轴（通过 HMI 选择 JOG 轴）

设置通道专用机床数据

为在 SINUMERIK Operate 中使用该功能，请设置以下通道专用的机床数据：

- 激活选件 $MCS_FUNCTION_MASK_DISP Bit 16 (MD52210)
- 加工轴的分配 $MC_DISPLAY_AXIS (MD20098)

详细信息，请参阅 SINUMERIK Operate 的文档和在线帮助 *“调试 SINUMERIK Operate”*。

编程原则

说明

HMI 和 PLC 之间针对轴预选的通信是直接进行的，即不需要与 OB1 节拍同步或通过调用指令“SINU_Update”实现同步。用于轴预选的输入信号（按下按钮）可供 PLC 用户程序使用，时间最长为 300 ms，该时间结束后 SINUMERIK Operate 会自动复位。

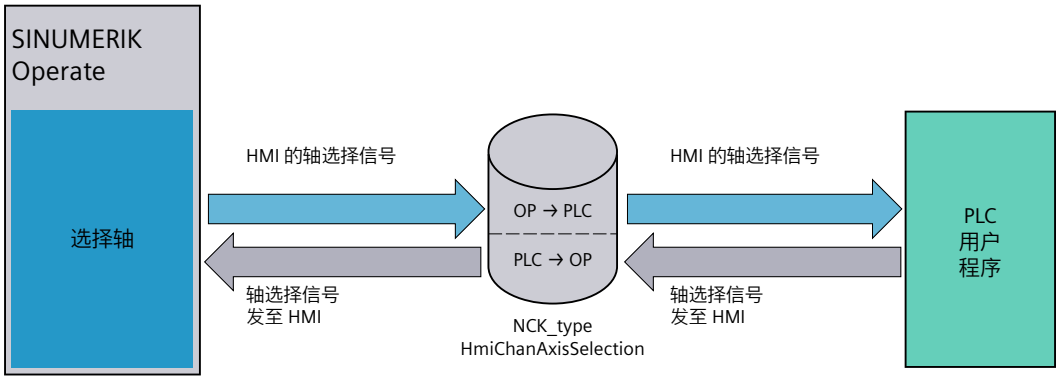


图 10-8 选择加工轴和几何轴原理图

NCK 相关数据类型“NCK_typeHmiChanAxisSelection”所创建的实例中的输入和输出数据的更新不是由块“SINU_Update”完成的。

编程流程

在用户项目中生成并注册一个 NCK 相关数据类型"NCK_typeHmiChan"的实例，并通过指令"SINU_CompleteRegistration"使其能够被系统所识别。一旦在 SINUMERIK Operate 中选择了一个轴，其数据就会存储在已注册实例的输入区域中。可以利用 PLC 用户程序处理和评估这些数据。通过已注册实例输出区域中的信号进行确认。

说明

已注册通道实例的输入和输出信号必须在 PLC 用户程序中根据具体的通道进行处理。即 SINUMERIK Operate 中的轴预选将根据所激活的通道被传输到相应的通道实例中。SINUMERIK Operate 中的按键操作被传输到通道实例中，时间最长为 300ms。

示例

利用"通过 HMI 选择 JOG 轴"功能选择通道 1 中的几何轴。

已注册通道实例的以下输入信号与 PLC 用户程序有关：

信号名称	含义和当前值	信号方向
<HmiChan1>.axisSelection.in.selectedType	轴选择类型： 1：几何轴	OP → PLC
<HmiChan1>.axisSelection.in.selectedNumber	轴编号： 3：几何轴号	OP → PLC
<HmiChan1>.axisSelection.in.selectedKeyActuated	轴选择"按键"（触摸键）已按下	OP → PLC

已注册通道实例的以下输出信号与 PLC 用户程序有关：

信号名称	含义和当前值	信号方向
<HmiChan1>.axisSelection.out.activeType	轴选择类型： 1：几何轴	PLC → OP
<HmiChan1>.axisSelection.out.activeNumber	轴编号： 3：几何轴号	PLC → OP

一旦输入信号"<HmiChan1>.axisSelection.in.selectedKeyActuated"通过上升沿值变成 TRUE，PLC 用户程序就可以检查并应用信号"<HmiChan1>.axisSelection.in.selectedType"和"<HmiChan1>.axisSelection.in.selectedNumber"的值。

在这个示例中，SINUMERIK Operate 预选了编号为 3（轴号=3）的几何轴（轴类型=1）。

借助输出信号"<HmiChan1>.axisSelection.out.activeType"= 1 和"<HmiChan1>.axisSelection.out.activeNumber"= 3 的反馈通过 PLC 用户程序应用和激活几何轴。

说明

通过输入信号"<HmiChan1>.axisSelection.in.selectedKeyActuated"的上升沿重新应用轴预选。

10.3.7 选择手轮

为轴分配手轮

通过 SINUMERIK Operate 可以将所连接的手轮分配给加工轴、几何轴、轮廓轴和定向轴。一个控制器中最多可以有 6 个手轮的参数。

说明

为了在与 SINUMERIK Operate 的交互中实现无误的功能，进行手轮编号的参数设置时必须始终采用二进制编码（将 MD11324 $MN_HANDWH_VDI_REPRESENTATION 设置为数值 1）。

更多详细信息，请参见*功能手册：进给轴和主轴*中的章节“手动方法 > 调试：手轮”。

编程原则

HMI 和 PLC 之间针对手轮分配的通信是直接进行的，即不需要与 OB1 节拍同步或通过调用指令"SINU_Update"实现同步。

为了使轴的手轮编号在 NC 中生效，必须通过 PLC 用户程序将出自 SINUMERIK Operate 的输入信号处理为相应的 NCK 相关数据类型，并将其传输到相关数据类型的输出信号区域内。

有四种手轮类型

- 加工轴手轮 → 所分配轴 <Axis> 的 NCK 相关数据类型
- 轮廓轴手轮 → 所分配轴 <Axis> 的 NCK 相关数据类型
- 几何轴手轮 → 所属通道 <Chan> 的 NCK 相关数据类型
- 定向轴手轮 → 所属通道 <Chan> 的 NCK 相关数据类型

由 NC 处理信号并将手轮信号分配给指定的轴。只有在完成分配后，才能用手轮移动轴。

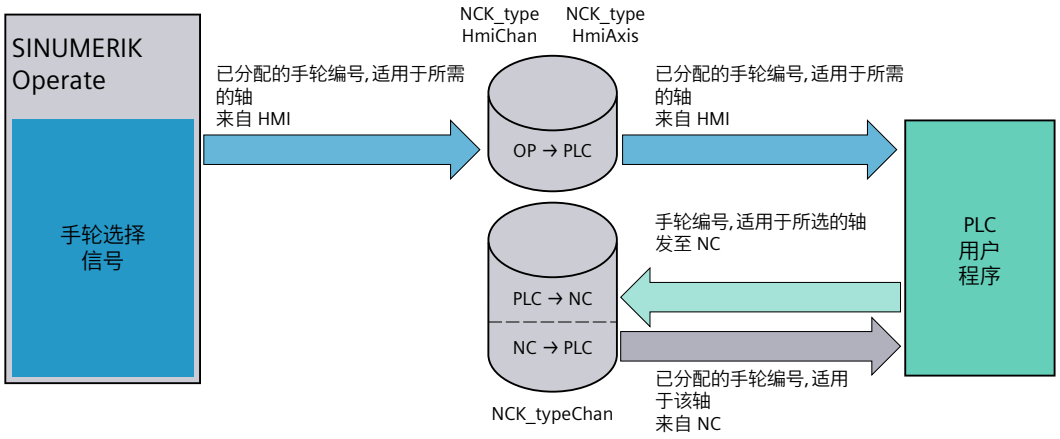


图 10-9　选择手轮原理图

SINUMERIK Operate 通过 OPI 变量直接从 NC 中确定已激活的手轮分配。

示例

将手轮 1 分配给加工轴 X（使用默认机床数据时）。

已注册 X 轴的以下输入信号与 PLC 用户程序有关：

信号名称	含义和当前值	信号方向
<HmiAxis>.basic.in.handwheelSelected	HMI：选择界面上的手轮编号 1：手轮 1	OP → PLC
<Axis>.basic.in.handwheelAssignedNumber	当前所分配手轮的编号： 1：手轮 1	NC → PLC

已注册 X 轴的以下输出信号与 PLC 用户程序有关：

信号名称	含义和当前值	信号方向
<Axis>.basic.out.handwheelSelection	所使用轴的手轮编号 1：手轮 1	PLC → NC

输入信号“<HmiAxis>.basic.in.handwheelSelected”直接切换为 PLC 用户程序的输出信号“<Axis>.basic.out.handwheelSelection”。通过循环调用“SINU_Update”指令，确保所分配的

手轮编号能够被传送到 NC 上。由 NC 完成所有检查后，将应用并分配用于 X 轴的手轮编号（输入信号“<Axis>.basic.in.handwheelAssignedNumber”的数值为“1”）。

说明

请注意，NC 和 SINUMERIK Operate 不会检查手轮的分配是否正确。这在 PLC 用户程序（在 SINUMERIK Operate 中该程序将所选的手轮编号传输到的另一个轴实例中）中是不可见的。

10.3.8 干涉保护

干涉保护

“干涉保护”功能用于防止机床部件以及刀具刀沿在机床轴移行期间发生碰撞。

详细信息请参见*功能手册：监控与补偿*中的“内部干涉保护”章节。

编程原则

为了处理这些信号，必须在 PLC 用户项目中针对所需保护范围的管理创建 NCK 相关数据类型“NCK_typeHmiNc”和“NCK_typeNc”的实例，然后注册并通过指令“SINU_CompleteRegistration”使它们能够被系统识别。NCK 相关数据类型的周期性更新是通过 PLC 项目中的“SINU_Update”指令完成的。

为了使该信号在 NC 过程中生效，在 PLC 用户程序中处理来自 SINUMERIK Operate 的 NCK 相关数据类型“NCK_typeHmiNc”的输入信号。将输入数据传输到数据类型为“NCK_typeNc”的输出信号范围中。此外，您可以激活或锁定单独的保护范围。

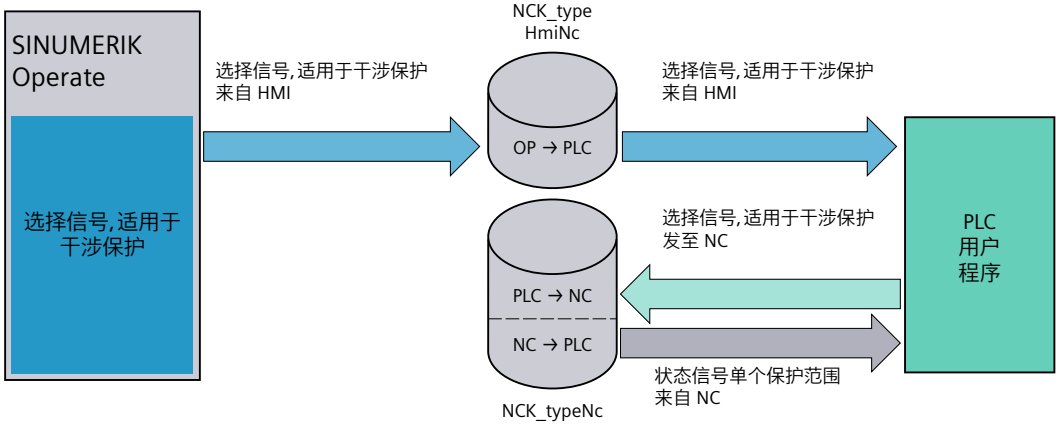


图 10-10 干涉保护原理图

SINUMERIK Operate 通过 OPI 变量直接从 NC 中确定已激活的配置。

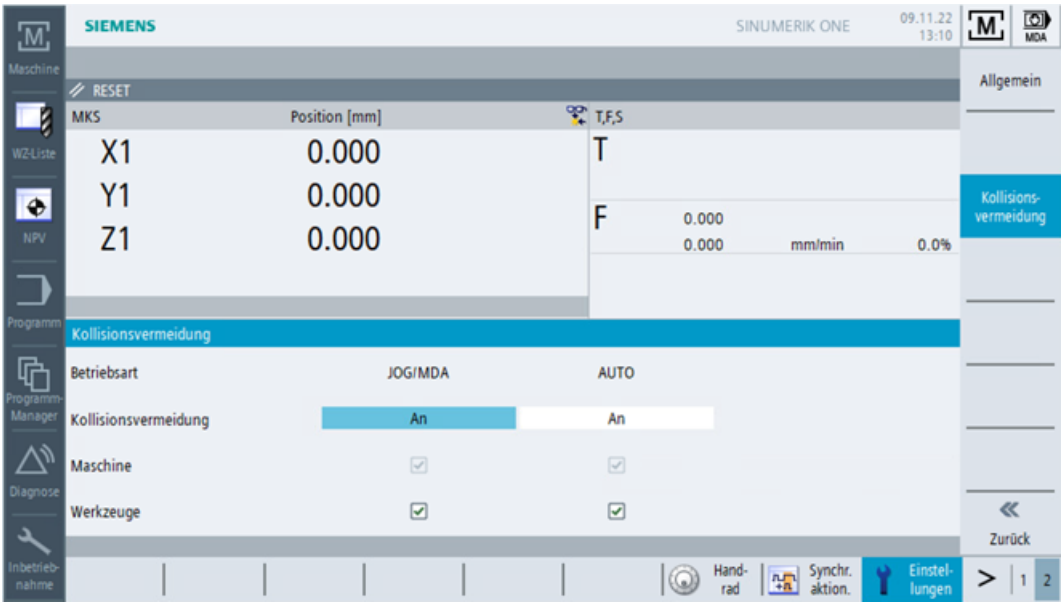


图 10-11　JOG/MDA 和 AUTO 中的干涉保护已激活

示例

在 JOG/MDA 运行方式下，将禁用整个保护范围的监控。针对该示例，需假定已经根据*功能手册：监控与补偿*中的示例设置了干涉保护的参数化。

在 PLC 用户程序中，将 NCK 相关数据类型 <hmiNc>.collision.in 的输入信号传输到输出信号 <nc>.collision.out 中。下表举例介绍了分配方式。

输入信号		输出信号
<hmiNc>.collision.in.machineAreasAutoDeactivated	→	<nc>.collision.out.deactivateMachineAreasAuto
<hmiNc>.collision.in.deactivateToolAreasAuto	→	<nc>.collision.out.deactivateToolAreasAuto
<hmiNc>.collision.in.deactivateFixtureAreasAuto	→	<nc>.collision.out.deactivateFixtureAreasAuto
<hmiNc>.collision.in.deactivateWorkpieceAreasAuto	→	<nc>.collision.out.deactivateWorkpieceAreasAuto
<hmiNc>.collision.in.deactivateMachineAreasJog	→	<nc>.collision.out.deactivateMachineAreasJog
<hmiNc>.collision.in.deactivateToolAreasJog	→	<nc>.collision.out.deactivateToolAreasJog
<hmiNc>.collision.in.deactivateFixtureAreasJog	→	<nc>.collision.out.deactivateFixtureAreasJog
<hmiNc>.collision.in.deactivateWorkpieceAreasJog	→	<nc>.collision.out.deactivateWorkpieceAreasJog

进行干涉保护时，针对 JOG/MDA 运行方式应选择“关闭”，SINUMERIK Operate 将切换至“已禁用干涉保护”状态。

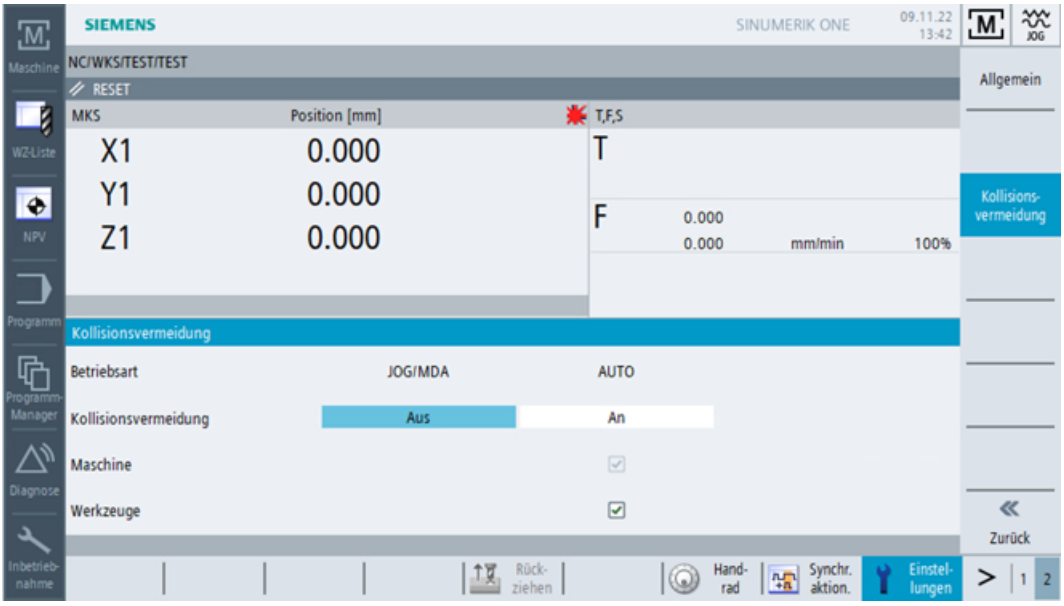


图 10-12　在 JOG/MDA 运行方式中已禁用干涉保护

编译和加载块 11

11.1 使用其他设备类型中的专有技术保护程序块/库

专有技术保护程序块或库始终会绑定至特定的设备类型（专有技术保护的一部分）。该绑定是额外存在的，与专有技术保护的可选序列号绑定无关。对于 S7-1500 程序块的设备类型绑定，下列控制器类型之间存在差异：

- 硬件 PLC，例如：CPU 1518F-4 PN/DP
- 软件 PLC，例如：CPU 1505 SP (F/T/TF)
- 数控系统，例如：SINUMERIK ONE
- 模拟环境（例如：SINUMERIK Create MyVirtual Machine）中在模拟的控制器类型上会有一个设备类型绑定并且要求额外生效的模拟性能。

设置专有技术保护和编译块时，系统会自动输入并/或更新设备类型连接。需要密码才能编译块或更改专有技术保护和/或设备类型连接，因此，该功能通常预留给创建该块的编程人员。

- 如果要将该专有技术保护的组织块用于类型与原来不同的控制器，则必须将该块复制到该类型的控制器中并重新进行编译。为此，您需要有待编译程序块的密码。
- 为了能够使用其他人创建的专有技术保护程序块或库，您需要针对您的控制器类型编译的版本。

如果没有密码就开始编译此类程序块，则会出现以下错误消息，过程被中断：“其他 CPU 系列中设置了专有技术保护。只有获得有效密码才能编译该程序块。”

参见

激活块的模拟能力 (页 133)

11.2 激活块的模拟能力

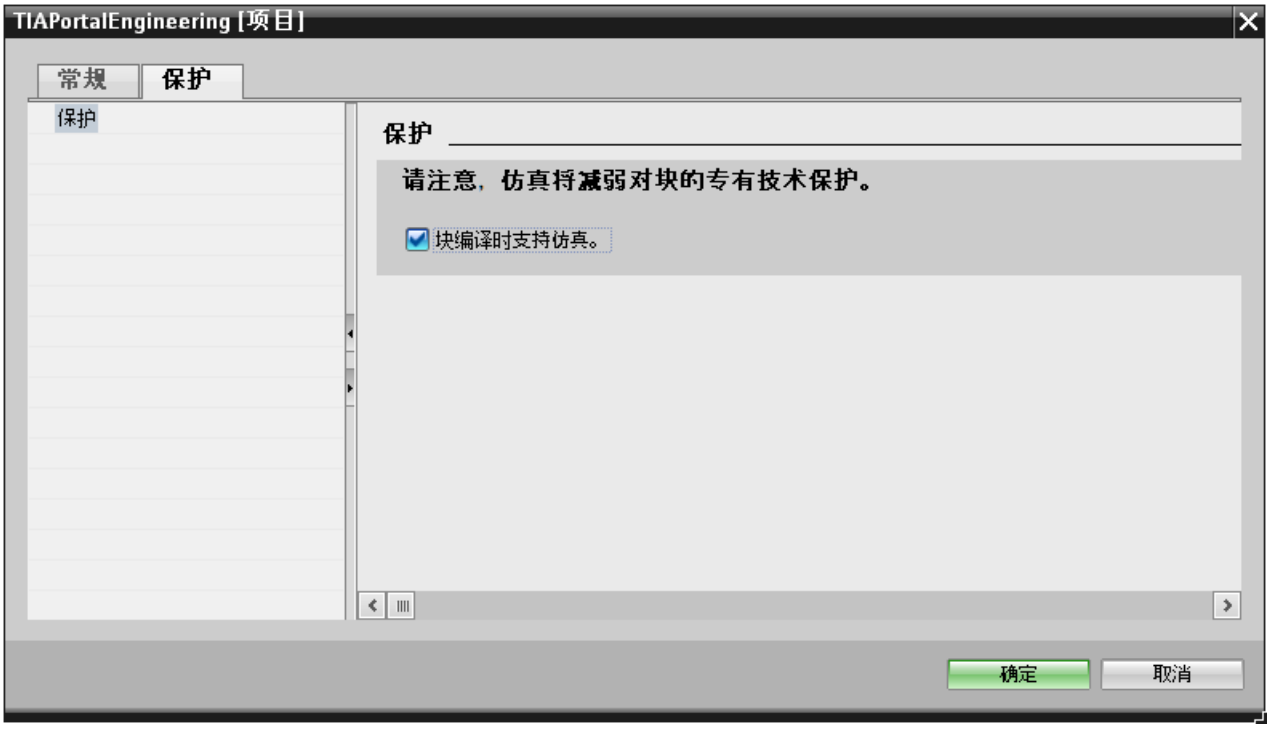
如要在模拟环境 (Create MyVirtual Machine) 下使用一个 TIA Portal 项目，则必须在编译项目前，在项目属性中激活模拟能力。该设置默认不激活，因其会影响专有技术保护。

将程序块载入模拟时需要检查它们是否可被模拟。如果发生错误，则会显示以下消息：“'MyKhpBlock [FC30]'无法被模拟。”。如果该程序块来自于库，则使用支持模拟的库。否则，在项目属性中激活选项“编译支持模拟的程序块时并重新编译程序块”。

步骤

请按以下步骤，在项目中激活程序块的模拟能力：

1. 在项目树中右击项目并选择“属性”。
2. 在“保护”选项卡下激活“支持用于编译的程序块模拟能力”选项。单击“确定”进行确认。



针对 PLC Basic Program Plus 的参考 12

12.1 引言

“PLC 基本程序参考”主要介绍 SINUMERIK 基本程序的库中所包含的块、指令和数据类型。

描述和示例基于使用 SCL 编程语言的符号编程方式展开。

12.2 NCK 相关的数据类型（简要描述）

12.2.1 概览

下表主要介绍基本数据类型。

数据类型	说明
NCK_typeAxis (页 136)	用户程序和轴/主轴之间的接口 包含来自/发送至轴 x/主轴 x 的信号
NCK_typeChan (页 182)	用户程序和通道之间的接口 包含来自/发送至通道 x 的信号
NCK_typeModeGroup (页 306)	用户程序和运行方式组之间的接口 包含来自/发送至 NC 和操作软件的信号
NCK_typeNc (页 312)	用户程序和 NC 之间的接口 包含来自/发送至 NC、PLC 和操作软件的信号
NCK_typeToolholder (页 388)	用户程序和刀具管理之间的接口 包含刀夹信号
NCK_typeRelocate (页 407)	用户程序和刀具管理之间的接口 包含移刀信号
NCK_typeLoadingPoint (页 413)	用户程序和刀具管理之间的接口 包含装载位信号
NCK_typeToaUnit (页 419)	用户程序和刀具管理之间的接口 包含 TO 区域信号

数据类型	说明
NCK_typeHmi (页 420)	用户程序和 HMI 之间的接口 包含来自/发送至 HMI 的一般信号
NCK_typeHmiAxis (页 429)	用户程序和 HMI 之间的接口 包含来自 HMI，用于轴的信号
NCK_typeHmiChan (页 430)	用户程序和 HMI 之间的接口 包含来自 HMI，用于通道的信号
NCK_typeHmiModeGroup (页 435)	用户程序和 HMI 之间的接口 包含来自 HMI，用于运行方式组的信号
NCK_typeHmiNc (页 437)	用户程序和 HMI 之间的接口 包含来自 HMI，用于 NC 的信号

与老款 NCK 接口数据块相比，当前 NCK 相关数据类型的结构有所不同 -从功能性和信号方向两方面着眼进行结构化。

上述每种数据类型都具有一个“config”结构，该结构包含与数据类型有关的组态数据，具体含义见下文。

只要合理，所有其他结构都包含两个子结构“in”和“out”。这两个子结构包含各自的专用信号。通过这种方式，立刻就能识别出从 PLC 角度来看，信号是输入信号 (in) 还是输出信号 (out)。

功能性结构将用途一致的信号结合在一起。

说明

创建“NCK_typeHmi*”数据类型的实例时，必须设置“可从 HMI/OPC UA/Web API 到达”和“可从 HMI/OPC UA/Web API 写入”两种属性。如果不设置，HMI Operate 无法写入信号。

12.2.2 NCK_typeAxis

<Axis>.config
轴的配置数据

<Axis>.basic
轴的通用信号

<Axis>.drive
驱动系统的信号

<Axis>.spindle	
主轴的信号	
<Axis>.interchange	
轴切换信号	
<Axis>.coupling	
轴的耦合信号	
<Axis>.plcCtrl	
PLC 控制轴的信号	
<Axis>.oscillation	
轴的往复信号	
<Axis>.safety	
轴的安全专用信号	
<Axis>.safetyInfoChan	
从驱动发送给 PLC 的信号(SIC)	
<Axis>.safetyCtrlChan	
从 PLC 发送给驱动的信号(SCC)	
<Axis>.basic.in	**NC → PLC**
从轴收到的通用信号	
<Axis>.basic.out	**PLC → NC**
发送给轴的通用信号	
<Axis>.drive.in	**NC → PLC**
从驱动系统收到的信号	
<Axis>.drive.out	**PLC → NC**
发送给驱动系统的信号	

<Axis>.spindle.in	**NC → PLC**
从主轴收到的信号	

<Axis>.spindle.out	**PLC → NC**
发送给主轴的信号	

<Axis>.coupling.in	**NC → PLC**
从轴收到的耦合信号	

<Axis>.coupling.out	**PLC → NC**
发送给轴的耦合信号	

<Axis>.plcCtrl.in	**NC → PLC**
从 PLC 控制轴收到的状态信号	

<Axis>.plcCtrl.out	**PLC → NC**
发送给 PLC 控制轴的控制信号	

<Axis>.oscillation.in	**NC → PLC**
从轴收到的往复信号	

<Axis>.oscillation.out	**PLC → NC**
发送给轴的往复信号	

<Axis>.interchange.in	**NC → PLC**
用于轴切换的输入信号	

<Axis>.interchange.out	**PLC → NC**
用于轴切换的输出信号	

<Axis>.safety.in	**NC → PLC**
输入的轴安全专用信号	

<Axis>.safety.out	**PLC → NC**
输出的轴安全专用信号	

<Axis>.safetyInfoChan.in	**NC → PLC**	
从轴发出的 SIC 信号		

<Axis>.safetyCtrlChan.out	**PLC → NC**	
发送给轴的 SCC 信号		

<Axis>.config.registered	**NC → PLC**	**Bool**
轴已注册		

<Axis>.config.axisActive	**NC → PLC**	**Bool**
轴已生效		

<Axis>.config.axisNumber	**NC → PLC**	**USInt**
轴编号		

<Axis>.basic.in.disabled	**NC → PLC**	**Bool**
轴已被禁用，无法移动		
SINUMERIK ONE Basic Program	LBP_Axis*.E_AxSpDisable	
SINUMERIK 840D sl	DB31,... .DBX63.3	

<Axis>.basic.in.ready	**NC → PLC**	**Bool**
轴已运行就绪		
SINUMERIK ONE Basic Program	LBP_Axis*.E_AxisReady	
SINUMERIK 840D sl	DB31,... .DBX61.2	

<Axis>.basic.in.alarmActive	**NC → PLC**	**Bool**
轴上有报警没有应答		
SINUMERIK ONE Basic Program	LBP_Axis*.E_AxialAlarm	
SINUMERIK 840D sl	DB31,... .DBX61.1	

<Axis>.basic.in.isStationary	**NC → PLC**	**Bool**
轴达到静止转速(n<n_min)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_Stat	
SINUMERIK 840D sl	DB31,... .DBX61.4	

<Axis>.basic.in.enc1Active		**NC → PLC**	**Bool**
位置测量系统 1 已生效			
SINUMERIK ONE Basic Program	LBP_Axis*.E_PosMeas1		
SINUMERIK 840D sl	DB31,... .DBX102.5		

<Axis>.basic.in.enc1Synchronized		**NC → PLC**	**Bool**
机床轴位置测量系统 1 已回参考点或同步			
SINUMERIK ONE Basic Program	LBP_Axis*.E_RefSyn1		
SINUMERIK 840D sl	DB31,... .DBX60.4		

<Axis>.basic.in.enc1FreqLimitExceeded		**NC → PLC**	**Bool**
超出了测量系统 1 的编码器频率限值			
SINUMERIK ONE Basic Program	LBP_Axis*.E_EncodFreq1		
SINUMERIK 840D sl	DB31,... .DBX60.2		

<Axis>.basic.in.enc1Restored		**NC → PLC**	**Bool**
机床轴位置测量系统 1 的位置已恢复			
SINUMERIK ONE Basic Program	LBP_Axis*.E_POS_RESTORED1		
SINUMERIK 840D sl	DB31,... .DBX71.4		

<Axis>.basic.in.enc2Active		**NC → PLC**	**Bool**
位置测量系统 2 已生效			
SINUMERIK ONE Basic Program	LBP_Axis*.E_PosMeas2		
SINUMERIK 840D sl	DB31,... .DBX102.6		

<Axis>.basic.in.enc2Synchronized		**NC → PLC**	**Bool**
机床轴位置测量系统 2 已回参考点或同步			
SINUMERIK ONE Basic Program	LBP_Axis*.E_RefSyn2		
SINUMERIK 840D sl	DB31,... .DBX60.5		

<Axis>.basic.in.enc2FreqLimitExceeded		**NC → PLC**	**Bool**
超出了测量系统 2 的编码器频率限值			
SINUMERIK ONE Basic Program	LBP_Axis*.E_EncodFreq2		
SINUMERIK 840D sl	DB31,... .DBX60.3		

<Axis>.basic.in.enc2Restored	**NC → PLC**	**Bool**
机床轴位置测量系统 2 的位置已恢复		
SINUMERIK ONE Basic Program	LBP_Axis*.E_POS_RESTORED2	
SINUMERIK 840D sl	DB31,... .DBX71.5	

<Axis>.basic.in.clampingToleranceExceeded	**NC → PLC**	**Bool**
超出夹紧公差 (MD36050 $MA_CLAMP_POS_TOL)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_ClampTolerance	
SINUMERIK 840D sl	DB31,... .DBX102.3	

<Axis>.basic.in.posCoarseReached	**NC → PLC**	**Bool**
程序位置已到达“粗准停”范围(MD36000 $MA_STOP_LIMIT_COARSE)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_ExactCoarse	
SINUMERIK 840D sl	DB31,... .DBX60.6	

<Axis>.basic.in.posFineReached	**NC → PLC**	**Bool**
程序位置已到达“精准停”范围 (MD36150 $MA_STOP_LIMIT_FINE)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_ExactFine	
SINUMERIK 840D sl	DB31,... .DBX60.7	

<Axis>.basic.in.followUpModeActive	**NC → PLC**	**Bool**
跟踪模式已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.E_Follow_upMode	
SINUMERIK 840D sl	DB31,... .DBX61.3	

<Axis>.basic.in.posCtrlActive	**NC → PLC**	**Bool**
位置控制器已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.E_PositContr	
SINUMERIK 840D sl	DB31,... .DBX61.5	

<Axis>.basic.in.speedCtrlActive	**NC → PLC**	**Bool**
转速控制器已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SpeedContr	
SINUMERIK 840D sl	DB31,... .DBX61.6	

<Axis>.basic.in.swCamsActive	**NC → PLC**	**Bool**
软件挡块使用已生效，当达到某软件挡块时，会向 PLC 发出信号。		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SWCam	
SINUMERIK 840D sl	DB31,... .DBX62.0	

<Axis>.basic.in.revolutionalFeedrateActive	**NC → PLC**	**Bool**
功能 G95 已选中：旋转进给速度生效，进给速度单位为毫米/转。		
SINUMERIK ONE Basic Program	LBP_Axis*.E_RevFD	
SINUMERIK 840D sl	DB31,... .DBX62.2	

<Axis>.basic.in.measurementActive	**NC → PLC**	**Bool**
“测量”功能已生效（比如：测量工件或刀具）		
SINUMERIK ONE Basic Program	LBP_Axis*.E_MeasAct	
SINUMERIK 840D sl	DB31,... .DBX62.3	

<Axis>.basic.in.handwheelAssignedNumber	**NC → PLC**	**USInt**
当前分配手轮的编号：1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）		
SINUMERIK ONE Basic Program	LBP_Axis*.E_HW1..3	
SINUMERIK 840D sl	DB31,... .DBX64.0..2	

<Axis>.basic.in.handwheelDirInverted	**NC → PLC**	**Bool**
手轮转向已取反		
SINUMERIK ONE Basic Program	LBP_Axis*.E_InvHWDirOfRot	
SINUMERIK 840D sl	DB31,... .DBX67.0	

<Axis>.basic.in.handwheelOvrActive	**NC → PLC**	**Bool**
手轮倍率生效：手轮可用于移动轴		
SINUMERIK ONE Basic Program	LBP_Axis*.E_HWOverlay	
SINUMERIK 840D sl	DB31,... .DBX62.1	

<Axis>.basic.in.traversMinusRequestActive	**NC → PLC**	**Bool**
负向移动已请求		
SINUMERIK ONE Basic Program	LBP_Axis*.E_TReqMinus	
SINUMERIK 840D sl	DB31,... .DBX64.4	

<Axis>.basic.in.traversPlusRequestActive		**NC → PLC**	**Bool**
正向移动已请求			
SINUMERIK ONE Basic Program	LBP_Axis*.E_TReqPlus		
SINUMERIK 840D sl	DB31,... .DBX64.5		

<Axis>.basic.in.traversMinusCommandActive		**NC → PLC**	**Bool**
负向移动指令已生效(MD17900 $MN_VDI_FUNCTION_MASK)			
SINUMERIK ONE Basic Program	LBP_Axis*.E_TCMinus		
SINUMERIK 840D sl	DB31,... .DBX64.6		

<Axis>.basic.in.traversPlusCommandActive		**NC → PLC**	**Bool**
正向移动指令已生效(MD17900 $MN_VDI_FUNCTION_MASK)			
SINUMERIK ONE Basic Program	LBP_Axis*.E_TCPlus		
SINUMERIK 840D sl	DB31,... .DBX64.7		

<Axis>.basic.in.inc1Active		**NC → PLC**	**Bool**
“收到请求时移动轴 1 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Axis*.E_INC1		
SINUMERIK 840D sl	DB31,... .DBX65.0		

<Axis>.basic.in.inc10Active		**NC → PLC**	**Bool**
“收到请求时移动轴 10 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Axis*.E_INC10		
SINUMERIK 840D sl	DB31,... .DBX65.1		

<Axis>.basic.in.inc100Active		**NC → PLC**	**Bool**
“收到请求时移动轴 100 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Axis*.E_INC100		
SINUMERIK 840D sl	DB31,... .DBX65.2		

<Axis>.basic.in.inc1000Active		**NC → PLC**	**Bool**
“收到请求时移动轴 1 000 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Axis*.E_INC1000		
SINUMERIK 840D sl	DB31,... .DBX65.3		

<Axis>.basic.in.inc10000Active		**NC → PLC**	**Bool**
“收到请求时移动轴 10 000 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Axis*.E_INC10000		
SINUMERIK 840D sl	DB31,... .DBX65.4		

<Axis>.basic.in.incVarActive		**NC → PLC**	**Bool**
“收到请求时移动轴用户自定义数量的增量”已生效			
SINUMERIK ONE Basic Program	LBP_Axis*.E_INCVar		
SINUMERIK 840D sl	DB31,... .DBX65.5		

<Axis>.basic.in.contTraversActive		**NC → PLC**	**Bool**
连续移动已生效			
SINUMERIK ONE Basic Program	LBP_Axis*.E_ContManTravelAct		
SINUMERIK 840D sl	DB31,... .DBX65.6		

<Axis>.basic.in.moduloLimitSwitchActive		**NC → PLC**	**Bool**
模数旋转轴的旋转范围限制功能已生效（软限位、工作区限制）			
SINUMERIK ONE Basic Program	LBP_Axis*.E_ModLimEnAct		
SINUMERIK 840D sl	DB31,... .DBX74.4		

<Axis>.basic.in.jogFixedPointSelectedNumber		**NC → PLC**	**USInt**
移动到 JOG 固定点(1..4)			
SINUMERIK ONE Basic Program	LBP_Axis*.E_JogFixPPos0..2Act		
SINUMERIK 840D sl	DB31,... .DBX75.0..2		

<Axis>.basic.in.jogFixedPointReachedNumber		**NC → PLC**	**USInt**
已按“精准停”方式到达目标 JOG 固定点(1..4)			
SINUMERIK ONE Basic Program	LBP_Axis*.E_JogFixPPos0..2		
SINUMERIK 840D sl	DB31,... .DBX75.3..5		

<Axis>.basic.in.jogToPosActive		**NC → PLC**	**Bool**
“JOG 定位”功能已生效			
SINUMERIK ONE Basic Program	LBP_Axis*.E_JogToPos		
SINUMERIK 840D sl	DB31,... .DBX75.6		

<Axis>.basic.in.jogPosReached		**NC → PLC**	**Bool**
在"JOG 定位"中，轴已经到达 SD43320"精准停"定义的位置			
SINUMERIK ONE Basic Program	LBP_Axis*.E_JogPos		
SINUMERIK 840D sl	DB31,... .DBX75.7		

<Axis>.basic.in.lubricationPulseActive		**NC → PLC**	**Bool**
已按机床数据 MD33050 $MA_LUBRICATION_DIST 定义的距离移动			
SINUMERIK ONE Basic Program	LBP_Axis*.E_ScratchPulse		
SINUMERIK 840D sl	DB31,... .DBX76.0		

<Axis>.basic.in.isPathAxis		**NC → PLC**	**Bool**
轴是轨迹轴			
SINUMERIK ONE Basic Program	LBP_Axis*.E_PathAxis		
SINUMERIK 840D sl	DB31,... .DBX76.4		

<Axis>.basic.in.isPosAxis		**NC → PLC**	**Bool**
轴是定位轴			
SINUMERIK ONE Basic Program	LBP_Axis*.E_PosAxis		
SINUMERIK 840D sl	DB31,... .DBX76.5		

<Axis>.basic.in.spindleModeActive		**NC → PLC**	**Bool**
主轴运行生效方式：控制运行、往复运行、定位运行、同步运行或带补偿夹具的攻丝			
SINUMERIK ONE Basic Program	LBP_Axis*.E_SpNA		
SINUMERIK 840D sl	DB31,... .DBX60.0		

<Axis>.basic.in.indexAxisInPos		**NC → PLC**	**Bool**
分度轴就位			
SINUMERIK ONE Basic Program	LBP_Axis*.E_IndexAxisPos		
SINUMERIK 840D sl	DB31,... .DBX76.6		

<Axis>.basic.in.rotAxisInPos		**NC → PLC**	**Bool**
旋转轴就位			
SINUMERIK ONE Basic Program	LBP_Axis*.E_CurvePos		
SINUMERIK 840D sl	DB31,... .DBX76.7		

<Axis>.basic.in.velocityReduced	**NC → PLC**	**Bool**
当前轴移动速度被“碰撞检测”功能下调		
SINUMERIK ONE Basic Program	LBP_Axis*.E_CollCheckRedSpeed	
SINUMERIK 840D sl	DB31,... .DBX77.0	

<Axis>.basic.in.posAxisFeedrateValue	**NC → PLC**	**Real**
轴的进给速度，轴作为定位轴移动		
SINUMERIK ONE Basic Program	LBP_Axis*.E_FPosAxis	
SINUMERIK 840D sl	DB31,... .DBD78	

<Axis>.basic.in.emergencyRetractionActive	**NC → PLC**	**Bool**
紧急回退已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.E_EmRetr	
SINUMERIK 840D sl	DB31,... .DBX98.7	

<Axis>.basic.in.fixedStopRequest	**NC → PLC**	**Bool**
请求启用“运行到固定挡块”		
SINUMERIK ONE Basic Program	LBP_Axis*.E_ActTravFStop	
SINUMERIK 840D sl	DB31,... .DBX62.4	

<Axis>.basic.in.fixedStopReached	**NC → PLC**	**Bool**
已到达固定挡块		
SINUMERIK ONE Basic Program	LBP_Axis*.E_FixedStop	
SINUMERIK 840D sl	DB31,... .DBX62.5	

<Axis>.basic.in.fixedStopForceLimited	**NC → PLC**	**Bool**
固定挡块夹紧力受到限制		
SINUMERIK ONE Basic Program	LBP_Axis*.E_ForceFixedStop	
SINUMERIK 840D sl	DB31,... .DBX62.6	

<Axis>.basic.in.reposOffsetActive	**NC → PLC**	**Bool**
REPOS 偏移已应用到轴上		
SINUMERIK ONE Basic Program	LBP_Axis*.E_ReposSh	
SINUMERIK 840D sl	DB31,... .DBX70.0	

<Axis>.basic.in.reposOffsetValid		**NC → PLC**	**Bool**
REPOS 偏移已作为有效值计算			
SINUMERIK ONE Basic Program	LBP_Axis*.E_ReposShValid		
SINUMERIK 840D sl	DB31,... .DBX70.1		

<Axis>.basic.in.reposDelayAck		**NC → PLC**	**Bool**
REPOS 延迟已应答			
SINUMERIK ONE Basic Program	LBP_Axis*.E_ReposDelayQuit		
SINUMERIK 840D sl	DB31,... .DBX70.2		

<Axis>.basic.in.dynBacklashCompActive		**NC → PLC**	**Bool**
动态反向间隙补偿生效			
SINUMERIK ONE Basic Program	LBP_Axis*.E_DynBComp		
SINUMERIK 840D sl	DB31,... .DBX102.0		

<Axis>.basic.in.actToolBasePos		**NC → PLC**	**LReal**
轴的当前 MCS 设定值			

<Axis>.basic.out.disable		**PLC → NC**	**Bool**
请求禁止进给轴/主轴			
SINUMERIK ONE Basic Program	LBP_Axis*.A_ASpDisable		
SINUMERIK 840D sl	DB31,... .DBX1.3		

<Axis>.basic.out.enableCtrl		**PLC → NC**	**Bool**
请求控制器使能			
SINUMERIK ONE Basic Program	LBP_Axis*.A_ContrEnable		
SINUMERIK 840D sl	DB31,... .DBX2.1		

<Axis>.basic.out.feedrateOvrFactor		**PLC → NC**	**USInt**
进给倍率的标定系数可以是采用二进制或格雷码 (MD12000 $MN_OVR_AX_IS_GRAY_CODE)			
SINUMERIK ONE Basic Program	LBP_Axis*.A_FD_OR		
SINUMERIK 840D sl	DB31,... .DBB0		

<Axis>.basic.out.enableOvr		**PLC → NC**	**Bool**
激活进给倍率/主轴倍率			
SINUMERIK ONE Basic Program	LBP_Axis*.A_ORactive		
SINUMERIK 840D sl	DB31,... .DBX1.7		

<Axis>.basic.out.followUpMode		**PLC → NC**	**Bool**
激活跟踪模式			
SINUMERIK ONE Basic Program	LBP_Axis*.A_Follow_upMode		
SINUMERIK 840D sl	DB31,... .DBX1.4		

<Axis>.basic.out.enc1Activation		**PLC → NC**	**Bool**
激活位置测量系统 1			
SINUMERIK ONE Basic Program	LBP_Axis*.A_PosMeas1		
SINUMERIK 840D sl	DB31,... .DBX1.5		

<Axis>.basic.out.enc2Activation		**PLC → NC**	**Bool**
激活位置测量系统 2			
SINUMERIK ONE Basic Program	LBP_Axis*.A_PosMeas2		
SINUMERIK 840D sl	DB31,... .DBX1.6		

<Axis>.basic.out.swCams		**PLC → NC**	**Bool**
激活正向/负向软件挡块信号的输出			
SINUMERIK ONE Basic Program	LBP_Axis*.A_SWCam		
SINUMERIK 840D sl	DB31,... .DBX2.0		

<Axis>.basic.out.resetMovement		**PLC → NC**	**Bool**
出现 0 → 1 上升沿：“删除剩余行程”已请求，“主轴复位”已请求			
SINUMERIK ONE Basic Program	LBP_Axis*.A_DelDTGSpReset		
SINUMERIK 840D sl	DB31,... .DBX2.2		

<Axis>.basic.out.clampingInProgress		**PLC → NC**	**Bool**
轴正在夹紧，激活监控			
SINUMERIK ONE Basic Program	LBP_Axis*.A_Clamp		
SINUMERIK 840D sl	DB31,... .DBX2.3		

<Axis>.basic.out.handwheelSelection		**PLC → NC**	**USInt**
分配手轮 1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）			
SINUMERIK ONE Basic Program	LBP_Axis*.A_HW1..3		
SINUMERIK 840D sl	DB31,... .DBX4.0..2		

<Axis>.basic.out.handwheelInvertDir		**PLC → NC**	**Bool**
取反手轮转向			
SINUMERIK ONE Basic Program	LBP_Axis*.A_InvHWDirOfRot		
SINUMERIK 840D sl	DB31,... .DBX7.0		

<Axis>.basic.out.feedStopSpindleStop		**PLC → NC**	**Bool**
进给停止/主轴停止已生效			
SINUMERIK ONE Basic Program	LBP_Axis*.A_FDSpStop		
SINUMERIK 840D sl	DB31,... .DBX4.3		

<Axis>.basic.out.disableTraversKeys		**PLC → NC**	**Bool**
禁用“正向”和“负向”按键			
SINUMERIK ONE Basic Program	LBP_Axis*.A_Disable		
SINUMERIK 840D sl	DB31,... .DBX4.4		

<Axis>.basic.out.rapidTraversOvr		**PLC → NC**	**Bool**
激活 JOG 模式中的快速移动速度(MD32010 $MA_JOG_VELO_RAPID)			
SINUMERIK ONE Basic Program	LBP_Axis*.A_RapidTrOR		
SINUMERIK 840D sl	DB31,... .DBX4.5		

<Axis>.basic.out.traversKeyMinus		**PLC → NC**	**Bool**
在 JOG 方式中负向移动机床轴(MD32020 $MA_JOG_VELO)			
SINUMERIK ONE Basic Program	LBP_Axis*.A_Minus		
SINUMERIK 840D sl	DB31,... .DBX4.6		

<Axis>.basic.out.traversKeyPlus		**PLC → NC**	**Bool**
在 JOG 方式中正向移动机床轴 (MD32020 $MA_JOG_VELO)			
SINUMERIK ONE Basic Program	LBP_Axis*.A_Plus		
SINUMERIK 840D sl	DB31,... .DBX4.7		

<Axis>.basic.out.inc1	**PLC → NC**	**Bool**
激活“收到请求时移动轴 1 个增量”		
SINUMERIK ONE Basic Program	LBP_Axis*.A_INC1	
SINUMERIK 840D sl	DB31,... .DBX5.0	

<Axis>.basic.out.inc10	**PLC → NC**	**Bool**
激活“收到请求时移动轴 10 个增量”		
SINUMERIK ONE Basic Program	LBP_Axis*.A_INC10	
SINUMERIK 840D sl	DB31,... .DBX5.1	

<Axis>.basic.out.inc100	**PLC → NC**	**Bool**
激活“收到请求时移动轴 100 个增量”		
SINUMERIK ONE Basic Program	LBP_Axis*.A_INC100	
SINUMERIK 840D sl	DB31,... .DBX5.2	

<Axis>.basic.out.inc1000	**PLC → NC**	**Bool**
激活“收到请求时移动轴 1 000 个增量”		
SINUMERIK ONE Basic Program	LBP_Axis*.A_INC1000	
SINUMERIK 840D sl	DB31,... .DBX5.3	

<Axis>.basic.out.inc10000	**PLC → NC**	**Bool**
激活“收到请求时移动轴 10 000 个增量”		
SINUMERIK ONE Basic Program	LBP_Axis*.A_INC10000	
SINUMERIK 840D sl	DB31,... .DBX5.4	

<Axis>.basic.out.incVar	**PLC → NC**	**Bool**
激活“收到请求时移动轴用户自定义数量的增量”		
SINUMERIK ONE Basic Program	LBP_Axis*.A_INCVar	
SINUMERIK 840D sl	DB31,... .DBX5.5	

<Axis>.basic.out.contTravers	**PLC → NC**	**Bool**
激活连续手动移动（默认）		
SINUMERIK ONE Basic Program	LBP_Axis*.A_ContManTravel	
SINUMERIK 840D sl	DB31,... .DBX5.6	

<Axis>.basic.out.jogFixedPointSelection		**PLC → NC**	**USInt**
选择 JOG 固定点 1..4(MD30600 $MA_FIX_POINT_POS)			
SINUMERIK ONE Basic Program	LBP_Axis*.A_JogFixPPos0..2		
SINUMERIK 840D sl	DB31,... .DBX13.0..2		

<Axis>.basic.out.jogToPos		**PLC → NC**	**Bool**
功能"JOG 定位"已请求(SD43320 $SA_JOG_POSITION)			
SINUMERIK ONE Basic Program	LBP_Axis*.A_JogToPos		
SINUMERIK 840D sl	DB31,... .DBX13.3		

<Axis>.basic.out.hwLimitSwitchMinus		**PLC → NC**	**Bool**
负向硬限位已动作			
SINUMERIK ONE Basic Program	LBP_Axis*.A_HWLimitMinus		
SINUMERIK 840D sl	DB31,... .DBX12.0		

<Axis>.basic.out.hwLimitSwitchPlus		**PLC → NC**	**Bool**
正向硬限位已动作			
SINUMERIK ONE Basic Program	LBP_Axis*.A_HWLimitPlus		
SINUMERIK 840D sl	DB31,... .DBX12.1		

<Axis>.basic.out.swLimitSwitch2Minus		**PLC → NC**	**Bool**
激活第二个负向软限位			
SINUMERIK ONE Basic Program	LBP_Axis*.A_SWLimit2Minus		
SINUMERIK 840D sl	DB31,... .DBX12.2		

<Axis>.basic.out.swLimitSwitch2Plus		**PLC → NC**	**Bool**
激活第二个正向软限位			
SINUMERIK ONE Basic Program	LBP_Axis*.A_SWLimit2Plus		
SINUMERIK 840D sl	DB31,... .DBX12.3		

<Axis>.basic.out.enableModuloLimitSwitch		**PLC → NC**	**Bool**
激活模数旋转轴的旋转范围限制（比如：软限位、工作区限制）			
SINUMERIK ONE Basic Program	LBP_Axis*.A_ModuloLimitEn		
SINUMERIK 840D sl	DB31,... .DBX12.4		

<Axis>.basic.out.refPointApproachCam	**PLC → NC**	**Bool**
已达到参考挡块，然后以下调后的转速朝参考点移动		
SINUMERIK ONE Basic Program	LBP_Axis*.A_DelayRef	
SINUMERIK 840D sl	DB31,... .DBX12.7	

<Axis>.basic.out.refPointSelection	**PLC → NC**	**USInt**
参考点选择(1..4)：可以最多使用四个参考点 (MD34100 $MA_REFP_SET_POS)		
SINUMERIK ONE Basic Program	LBP_Axis*.A_RefVal1..4	
SINUMERIK 840D sl	DB31,... .DBX2.4..7	

<Axis>.basic.out.progTestRequest	**PLC → NC**	**Bool**
请求激活功能“程序测试”（转速设定值不生效）		
SINUMERIK ONE Basic Program	LBP_Axis*.A_ProgtestActivate	
SINUMERIK 840D sl	DB31,... .DBX14.1	

<Axis>.basic.out.progTestSuppression	**PLC → NC**	**Bool**
请求关闭功能“程序测试”（转速设定值生效）		
SINUMERIK ONE Basic Program	LBP_Axis*.A_ProgtestSuppress	
SINUMERIK 840D sl	DB31,... .DBX14.0	

<Axis>.basic.out.progTestEnableCtrl	**PLC → NC**	**Bool**
程序测试：请求控制器使能		
SINUMERIK ONE Basic Program	LBP_Axis*.A_PrgtestAxRel	
SINUMERIK 840D sl	DB31,... .DBX3.7	

<Axis>.basic.out.reposPtp	**PLC → NC**	**Bool**
REPOS 重新定位，用于实现点到点(PTP)的坐标转换		
SINUMERIK 840D sl	DB31,... .DBX31.3	

<Axis>.basic.out.reposDelay	**PLC → NC**	**Bool**
REPOS 延迟已请求		
SINUMERIK ONE Basic Program	LBP_Axis*.A_REPOSDelay	
SINUMERIK 840D sl	DB31,... .DBX10.0	

<Axis>.basic.out.fixedStopEnable	**PLC → NC**	**Bool**
确认激活固定挡块的请求		
SINUMERIK ONE Basic Program	LBP_Axis*.A_EnabTravFixedStop	
SINUMERIK 840D sl	DB31,... .DBX3.1	

<Axis>.basic.out.fixedStopSensor	**PLC → NC**	**Bool**
已到达固定挡块(MD37060 $MA_FIXED_STOP_ACKN_MASK)		
SINUMERIK ONE Basic Program	LBP_Axis*.A_SensorFixedStop	
SINUMERIK 840D sl	DB31,... .DBX1.2	

<Axis>.basic.out.fixedStopAckReached	**PLC → NC**	**Bool**
应答“已到达固定挡块”，继续程序执行		
SINUMERIK ONE Basic Program	LBP_Axis*.A_AckFixedStop	
SINUMERIK 840D sl	DB31,... .DBX1.1	

<Axis>.basic.out.fixedFeedrateSelection	**PLC → NC**	**USInt**
JOG 模式中的固定进给率选择(1..4；0 表示无选择)，(MD12202 $MN_PERMANENT_FEED / MD12204 $MN_PERMANENT_ROT_AX_FEED)		
SINUMERIK ONE Basic Program	LBP_Axis*.A_FixedFD1..4	
SINUMERIK 840D sl	DB31,... .DBX3.2..5	

<Axis>.basic.out.acceptExternalWorkOffset	**PLC → NC**	**Bool**
应用外部零点偏移值已请求 (MD28082 $MC_MM_SYSTEM_FRAME_MASK)		
SINUMERIK ONE Basic Program	LBP_Axis*.A_ExtZO	
SINUMERIK 840D sl	DB31,... .DBX3.0	

<Axis>.basic.out.limitSpeed	**PLC → NC**	**Bool**
限制主轴速度/转速 (MD35160 $MA_SPIND_EXTERN_VELO_LIMIT)		
SINUMERIK ONE Basic Program	LBP_Axis*.A_VelocSpeedLimit	
SINUMERIK 840D sl	DB31,... .DBX3.6	

<Axis>.basic.out.posCtrlParameterSetSelection	**PLC → NC**	**USInt**
选择位置控制器参数组 (1..8)		
SINUMERIK ONE Basic Program	LBP_Axis*.A_Para_A..C	
SINUMERIK 840D sl	DB31,... .DBX9.0..2	

<Axis>.basic.out.disableParameterSetSwitchover		**PLC → NC**	**Bool**
禁止切换参数组			
SINUMERIK ONE Basic Program	LBP_Axis*.A_LockParSetDef		
SINUMERIK 840D sl	DB31,... .DBX9.3		

<Axis>.basic.out.dynBacklashComp		**PLC → NC**	**Bool**
激活动态反向间隙补偿			
SINUMERIK ONE Basic Program	LBP_Axis*.A_DynBComp		
SINUMERIK 840D sl	DB31,... .DBX25.0		

<Axis>.drive.in.enabled		**NC → PLC**	**Bool**
驱动模块运行已使能			
SINUMERIK ONE Basic Program	LBP_Axis*.E_DriveRunEn		
SINUMERIK 840D sl	DB31,... .DBX92.7		

<Axis>.drive.in.ready		**NC → PLC**	**Bool**
驱动已就绪			
SINUMERIK ONE Basic Program	LBP_Axis*.E_MotOK		
SINUMERIK 840D sl	DB31,... .DBX93.5		

<Axis>.drive.in.pulsesEnabled		**NC → PLC**	**Bool**
驱动脉冲已使能			
SINUMERIK ONE Basic Program	LBP_Axis*.E_PulseEnable		
SINUMERIK 840D sl	DB31,... .DBX93.7		

<Axis>.drive.in.classCAlarmActive		**NC → PLC**	**Bool**
驱动发出 C 类报警/警告			
SINUMERIK 840D sl	DB31,... .DBX95.7		

<Axis>.drive.in.motorTemperatureWarning		**NC → PLC**	**Bool**
电机温度超出了驱动中定义的报警阈值 (p0604)。			
SINUMERIK ONE Basic Program	LBP_Axis*.E_MTempWarn		
SINUMERIK 840D sl	DB31,... .DBX94.0		

<Axis>.drive.in.heatSinkTemperatureWarning	**NC → PLC**	**Bool**
功率半导体的散热器温度超出了设定的报警阈值(p0294)。		
SINUMERIK ONE Basic Program	LBP_Axis*.E_HTempWarn	
SINUMERIK 840D sl	DB31,... .DBX94.1	

<Axis>.drive.in.rapidStopActive	**NC → PLC**	**Bool**
斜坡函数发生器已禁用（以转速设定值 0 快速停止）		
SINUMERIK ONE Basic Program	LBP_Axis*.E_RUEncDisabl	
SINUMERIK 840D sl	DB31,... .DBX92.1	

<Axis>.drive.in.speedCtrlIntegratorDisabled	**NC → PLC**	**Bool**
驱动转速控制器的积分元件已禁用		
SINUMERIK ONE Basic Program	LBP_Axis*.E_IntegratDisable	
SINUMERIK 840D sl	DB31,... .DBX93.6	

<Axis>.drive.in.holdingBrakeReleased	**NC → PLC**	**Bool**
电机抱闸已松开		
SINUMERIK ONE Basic Program	LBP_Axis*.E_HoldBrakeOpened	
SINUMERIK 840D sl	DB31,... .DBX92.5	

<Axis>.drive.in.autonomousMotionActive	**NC → PLC**	**Bool**
由驱动自行控制的运动已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.E_RLIact	
SINUMERIK 840D sl	DB31,... .DBX92.4	

<Axis>.drive.in.driveDataSetNumber	**NC → PLC**	**USInt**
当前生效的驱动数据组 (DDS) 的显示接口		
SINUMERIK ONE Basic Program	LBP_Axis*.E_ParA..C	
SINUMERIK 840D sl	DB31,... .DBX93.0..2	

<Axis>.drive.in.motorDataSetNumber	**NC → PLC**	**USInt**
当前生效的电机数据组 (MDS) 的显示接口		
SINUMERIK ONE Basic Program	LBP_Axis*.E_MotA..B	
SINUMERIK 840D sl	DB31,... .DBX93.3..4	

<Axis>.drive.in.numberOfDriveDataSets	**NC → PLC**	**USInt**
当前生效的电机数据组/驱动数据组(MDS/DDS)的编号		
SINUMERIK 840D sl	DB31,... .DBX130.0..4	

<Axis>.drive.in.numberOfDriveDataSetsValid	**NC → PLC**	**Bool**
当前生效的电机数据组/驱动数据组(MDS/DDS)的编号有效		
SINUMERIK 840D sl	DB31,... .DBX130.7	

<Axis>.drive.in.speedSetpointReached	**NC → PLC**	**Bool**
转速实际值(n_实际)达到程序中的转速设定值(n_设定)；并且在指定时间(p2166)内没有超出公差范围(p2164)时，该信号生效。在程序中写入新的转速设定值后信号失效。		
SINUMERIK ONE Basic Program	LBP_Axis*.E_RUComplete	
SINUMERIK 840D sl	DB31,... .DBX94.2	

<Axis>.drive.in.speedAtSetpoint	**NC → PLC**	**Bool**
n_实际 = n_设定: 转速实际值(n_实际)达到转速设定值(n_设定)；并且在指定时间(p2167)内没有超出公差范围(p2163)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_NactNset	
SINUMERIK 840D sl	DB31,... .DBX94.6	

<Axis>.drive.in.speedBelowMin	**NC → PLC**	**Bool**
n_实际 < n_最小：转速实际值(n_实际)低于指定的最小转速(n_最小，p2161)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_NactNmin	
SINUMERIK 840D sl	DB31,... .DBX94.4	

<Axis>.drive.in.speedBelowThreshold	**NC → PLC**	**Bool**
n_实际 < n_x：转速实际值(n_实际)低于指定的阈值(n_x，p2155)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_NactNx	
SINUMERIK 840D sl	DB31,... .DBX94.5	

<Axis>.drive.in.torqueBelowThreshold	**NC → PLC**	**Bool**
M_d < M_dx: 转矩实际利用率(M_d)低于指定阈值(M_dx, p2194)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_MdMdx	
SINUMERIK 840D sl	DB31,... .DBX94.3	

<Axis>.drive.in.varSignalingFunc1Active	**NC → PLC**	**Bool**
变量报告功能 1 (r3294.0)，指定的驱动变量超出了指定阈值，含回差		
SINUMERIK ONE Basic Program	LBP_Axis*.E_Message	
SINUMERIK 840D sl	DB31,... .DBX94.7	

<Axis>.drive.in.varSignalingFunc2Active	**NC → PLC**	**Bool**
变量报告功能 2 (r3294.1)，指定的驱动变量超出了指定阈值，含回差		
SINUMERIK ONE Basic Program	LBP_Axis*.E_Message2	
SINUMERIK 840D sl	DB31,... .DBX95.4	

<Axis>.drive.in.varSignalingFunc3Active	**NC → PLC**	**Bool**
变量报告功能 3 (r3294.2)，指定的驱动变量超出了指定阈值，含回差		
SINUMERIK ONE Basic Program	LBP_Axis*.E_Message3	
SINUMERIK 840D sl	DB31,... .DBX95.5	

<Axis>.drive.in.esrDcVoltageAlarmActive	**NC → PLC**	**Bool**
扩展停止和回退(ESR)：直流母线电压低于指定的最小值 (p1248)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_UDClessAlThreshold	
SINUMERIK 840D sl	DB31,... .DBX95.1	

<Axis>.drive.in.esrGeneratorOperationActive	**NC → PLC**	**Bool**
扩展停止和回退(ESR)：再生模式已生效		
SINUMERIK 840D sl	DB31,... .DBX95.2	

<Axis>.drive.in.esrSpeedBelowMin	**NC → PLC**	**Bool**
扩展停止和回退(ESR)：实际转速低于指定的最低转速(p2161) - 轴作为再生轴工作		
SINUMERIK 840D sl	DB31,... .DBX95.3	

<Axis>.drive.in.posCtrlParameterSetNumber	**NC → PLC**	**USInt**
当前生效的位置控制器参数组编号		
SINUMERIK ONE Basic Program	LBP_Axis*.E_ParS_A	
SINUMERIK 840D sl	DB31,... .DBX69.0..2	

<Axis>.drive.in.driveTestTraversRequested	**NC → PLC**	**Bool**
轴移动前提条件已满足，因此已请求移动使能，以开展驱动测试		
SINUMERIK ONE Basic Program	LBP_Axis*.E_TravRequ	
SINUMERIK 840D sl	DB31,... .DBX61.0	

<Axis>.drive.in.driveCtrlActive	**NC → PLC**	**Bool**
转速设定值切换：轴已接管驱动的控制		
SINUMERIK ONE Basic Program	LBP_Axis*.E_BitValCtrlOutChang0	
SINUMERIK 840D sl	DB31,... .DBX96.5	

<Axis>.drive.in.currentCtrlActive	**NC → PLC**	**Bool**
电流控制器已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.E_CurrentContr	
SINUMERIK 840D sl	DB31,... .DBX61.7	

<Axis>.drive.out.enablePulses	**PLC → NC**	**Bool**
使能驱动的控制脉冲		
SINUMERIK ONE Basic Program	LBP_Axis*.A_PulseEnable	
SINUMERIK 840D sl	DB31,... .DBX21.7	

<Axis>.drive.out.releaseHoldingBrake	**PLC → NC**	**Bool**
松开电机抱闸		
SINUMERIK ONE Basic Program	LBP_Axis*.A_HoldBrakeToOpen	
SINUMERIK 840D sl	DB31,... .DBX20.5	

<Axis>.drive.out.rapidStop	**PLC → NC**	**Bool**
停止，无斜坡函数发生器（以转速设定值 0 快速停止）		
SINUMERIK 840D sl	DB31,... .DBX20.1	

<Axis>.drive.out.disableSpeedCtrlIntegrator	**PLC → NC**	**Bool**
转速控制器：禁用积分元件(I)，只有比例元件(P)生效 (PI → P)		
SINUMERIK ONE Basic Program	LBP_Axis*.A_IntegratDisable	
SINUMERIK 840D sl	DB31,... .DBX21.6	

<Axis>.drive.out.driveDataSetSelection	**PLC → NC**	**USInt**
选择驱动数据组 (1..maxNumDataSet)		
SINUMERIK ONE Basic Program	LBP_Axis*.A_ParA..C; A_MotA..B	
SINUMERIK 840D sl	DB31,... .DBX21.0..2	

<Axis>.drive.out.motorDataSetSelection	**PLC → NC**	**USInt**
选择电机数据组 (1..maxNumDataSet)		
SINUMERIK 840D sl	DB31,... .DBX21.3..4	

<Axis>.drive.out.motorChangeoverDone	**PLC → NC**	**Bool**
所有必要的电气和/或机械升级改造工作（比如：更换刀头）都已结束		
SINUMERIK ONE Basic Program	LBP_Axis*.A_MotOK	
SINUMERIK 840D sl	DB31,... .DBX21.5	

<Axis>.drive.out.enableDriveTestTravers	**PLC → NC**	**Bool**
驱动测试用移动使能已发出		
SINUMERIK ONE Basic Program	LBP_Axis*.A_DriveEnable	
SINUMERIK 840D sl	DB31,... .DBX1.0	

<Axis>.drive.out.requestDriveCtrl	**PLC → NC**	**Bool**
请求获取驱动控制		
SINUMERIK ONE Basic Program	LBP_Axis*.A_ChangSetpOutAss0	
SINUMERIK 840D sl	DB31,... .DBX24.5	

<Axis>.spindle.in.mFuncNumber	**NC → PLC**	**UInt**
专用 M 功能 (INT)：考虑的功能有：3/4 - 主轴正转/反转; 5 - 主轴停止; 19 - 主轴以指定角度停止; 70 - 主轴采用进给轴模式		
SINUMERIK ONE Basic Program	LBP_Axis*.E_MFunct	
SINUMERIK 840D sl	DB31,... .DBW86	

<Axis>.spindle.in.sFuncValue	**NC → PLC**	**Real**
专用 S 功能值(REAL)：主轴转速，转/分钟 (r/min) 或主轴恒定切削速度，转/分钟，取决于选中的运行方式（G97 或 G96）		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SFunct	
SINUMERIK 840D sl	DB31,... .DBD88	

<Axis>.spindle.in.requestedGearStageNumber	**NC → PLC**	**USInt**
目标齿轮档的编号 (1..5)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SetpGearA..C	
SINUMERIK 840D sl	DB31,... .DBX82.0..2	

<Axis>.spindle.in.gearChangeoverRequested	**NC → PLC**	**Bool**
齿轮档切换已请求		
SINUMERIK ONE Basic Program	LBP_Axis*.E_GearChange	
SINUMERIK 840D sl	DB31,... .DBX82.3	

<Axis>.spindle.in.speedLimitationExceeded	**NC → PLC**	**Bool**
转速实际值超出了指定限值(MD35100 $MA_SPIND_VELO_LIMIT)和公差(MD35150 $MA_SPIND_DES_VELO_TOL)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SpeedLimit	
SINUMERIK 840D sl	DB31,... .DBX83.0	

<Axis>.spindle.in.speedSetpointBelowMin	**NC → PLC**	**Bool**
转速设定值低于程序指令 G25 最小转速限值(SD43210 $SA_SPIND_MIN_VELO_G25)或低于为每个齿轮档设置的最小转速限值(MD35140 $MA_GEAR_STEP_MIN_VELO_LIMIT)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SetSpeedIncreased	
SINUMERIK 840D sl	DB31,... .DBX83.2	

<Axis>.spindle.in.speedSetpointAboveMax	**NC → PLC**	**Bool**
转速设定值超出了程序指令 G26 最大转速限值 (SD43220 $SA_SPIND_MAX_VELO_G26)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SetSpeedLimit	
SINUMERIK 840D sl	DB31,... .DBX83.1	

<Axis>.spindle.in.speedSetpointReached	**NC → PLC**	**Bool**
达到转速设定值公差范围(MD35150 $MA_SPIND_DES_VELO_TOL)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SetRange	
SINUMERIK 840D sl	DB31,... .DBX83.5	

<Axis>.spindle.in.cwRotActive	**NC → PLC**	**Bool**
实际旋转方向: 0 - 逆时针; 1 - 顺时针		
SINUMERIK ONE Basic Program	LBP_Axis*.E_ActRotRight	
SINUMERIK 840D sl	DB31,... .DBX83.7	

<Axis>.spindle.in.constantCuttingSpeedActive	**NC → PLC**	**Bool**
恒定切削速度：主轴转速被控制，以补偿工件直径的变化		
SINUMERIK ONE Basic Program	LBP_Axis*.E_ConstCuttSpeed	
SINUMERIK 840D sl	DB31,... .DBX84.0	

<Axis>.spindle.in.rigidTappingActive	**NC → PLC**	**Bool**
无补偿夹具的攻丝（刚性攻丝）生效		
SINUMERIK ONE Basic Program	LBP_Axis*.E_Tapping	
SINUMERIK 840D sl	DB31,... .DBX84.3	

<Axis>.spindle.in.syncModeActive	**NC → PLC**	**Bool**
主轴处于同步模式		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SyncMode	
SINUMERIK 840D sl	DB31,... .DBX84.4	

<Axis>.spindle.in.posModeActive	**NC → PLC**	**Bool**
主轴处于定位模式		
SINUMERIK ONE Basic Program	LBP_Axis*.E_PosMode	
SINUMERIK 840D sl	DB31,... .DBX84.5	

<Axis>.spindle.in.oscillModeActive	**NC → PLC**	**Bool**
主轴处于往复模式		
SINUMERIK ONE Basic Program	LBP_Axis*.E_OscillMode	
SINUMERIK 840D sl	DB31,... .DBX84.6	

<Axis>.spindle.in.ctrlModeActive	**NC → PLC**	**Bool**
主轴进入转速控制（由比如 M03/M04/M05 等指令启动）		
SINUMERIK ONE Basic Program	LBP_Axis*.E_ContrMode	
SINUMERIK 840D sl	DB31,... .DBX84.7	

<Axis>.spindle.in.toolWithDynLimitationActive	**NC → PLC**	**Bool**
当前装夹的刀具超出动态响应限值		
SINUMERIK ONE Basic Program	LBP_Axis*.E_ToolDynLimit	
SINUMERIK 840D sl	DB31,... .DBX85.0	

<Axis>.spindle.in.inPos		**NC → PLC**	**Bool**
主轴就位			
SINUMERIK ONE Basic Program	LBP_Axis*.E_SpInPosition		
SINUMERIK 840D sl	DB31,... .DBX85.5		

<Axis>.spindle.in.smi24		**NC → PLC**	
输入的 SMI24 传感器信号（WEISS 主轴）			

<Axis>.spindle.in.smi24.sensorsPresent		**NC → PLC**	**Bool**
SMI24: 传感器已存在			
SINUMERIK ONE Basic Program	LBP_Axis*.E_SensorAv		
SINUMERIK 840D sl	DB31,... .DBX132.0		

<Axis>.spindle.in.smi24.speedLimited		**NC → PLC**	**Bool**
SMI24: 模拟量传感器经过模数转换 (r5001)且转速受到限制(p5043)			
SINUMERIK 840D sl	DB31,... .DBX133.2		

<Axis>.spindle.in.smi24.sensorS1Present		**NC → PLC**	**Bool**
SMI24: 传感器 S1（夹紧系统的模拟量传感器）存在			
SINUMERIK ONE Basic Program	LBP_Axis*.E_S1aMeasARod		
SINUMERIK 840D sl	DB31,... .DBX132.1		

<Axis>.spindle.in.smi24.sensorS1Value		**NC → PLC**	**UInt**
SMI24: 传感器 S1（夹紧系统模拟量传感器）原始值			
SINUMERIK ONE Basic Program	LBP_Axis*.E_AnalogSensor		
SINUMERIK 840D sl	DB31,... .DBW136		

<Axis>.spindle.in.smi24.sensorS1StateNumber		**NC → PLC**	**UInt**
SMI24: 传感器 S1（夹紧系统模拟量传感器）模数转换后的状态 (0..11)			
SINUMERIK ONE Basic Program	LBP_Axis*.E_ClampingState		
SINUMERIK 840D sl	DB31,... .DBW134		

<Axis>.spindle.in.smi24.sensorS4Present		**NC → PLC**	**Bool**
SMI24: 传感器 S4（活塞进入最终位置）已存在			
SINUMERIK ONE Basic Program	LBP_Axis*.E_S4dPistonStopP		
SINUMERIK 840D sl	DB31,... .DBX132.4		

<Axis>.spindle.in.smi24.sensorS4Actuated	**NC → PLC**	**Bool**
SMI24: 传感器 S4（活塞进入最终位置）已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.E_S4PistonStopP	
SINUMERIK 840D sl	DB31,... .DBX138.4	

<Axis>.spindle.in.smi24.sensorS5Present	**NC → PLC**	**Bool**
SMI24: 传感器 S5（电机轴就位）已存在		
SINUMERIK ONE Basic Program	LBP_Axis*.E_S5dAngleShaft	
SINUMERIK 840D sl	DB31,... .DBX132.5	

<Axis>.spindle.in.smi24.sensorS5Actuated	**NC → PLC**	**Bool**
SMI24: 传感器 S5（电机轴就位）已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.E_S5AngleShaft	
SINUMERIK 840D sl	DB31,... .DBX138.5	

<Axis>.spindle.in.smi24.sensorS6Present	**NC → PLC**	**Bool**
SMI24: 传感器 S6（温度）已存在		
SINUMERIK ONE Basic Program	LBP_Axis*.E_S6Temp	
SINUMERIK 840D sl	DB31,... .DBX132.6	

<Axis>.oscillation.in.active	**NC → PLC**	**Bool**
往复模式已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.E_Oscill	
SINUMERIK 840D sl	DB31,... .DBX100.7	

<Axis>.oscillation.in.disabled	**NC → PLC**	**Bool**
禁止切换到往复模式		
SINUMERIK ONE Basic Program	LBP_Axis*.E_StopOscill	
SINUMERIK 840D sl	DB31,... .DBX100.3	

<Axis>.oscillation.in.error	**NC → PLC**	**Bool**
往复模式出错		
SINUMERIK ONE Basic Program	LBP_Axis*.E_ErrorOscill	
SINUMERIK 840D sl	DB31,... .DBX100.4	

<Axis>.oscillation.in.motionActive		**NC → PLC**	**Bool**
往复运动已生效			
SINUMERIK ONE Basic Program	LBP_Axis*.E_OscillMotion		
SINUMERIK 840D sl	DB31,... .DBX100.6		

<Axis>.oscillation.in.externalReversalActive		**NC → PLC**	**Bool**
从外部反向已生效			
SINUMERIK ONE Basic Program	LBP_Axis*.E_OscillAxExtRev		
SINUMERIK 840D sl	DB31,... .DBX100.2		

<Axis>.oscillation.in.sparkingOutActive		**NC → PLC**	**Bool**
光磨已生效			
SINUMERIK ONE Basic Program	LBP_Axis*.E_SparkOut		
SINUMERIK 840D sl	DB31,... .DBX100.5		

<Axis>.spindle.out.ovrFactor		**PLC → NC**	**USInt**
倍率系数，用于控制主轴转速，可通过 MCP 主轴倍率旋钮等装置确定。			
SINUMERIK ONE Basic Program	LBP_Axis*.A_SpOR		
SINUMERIK 840D sl	DB31,... .DBB19.0		

<Axis>.spindle.out.gearStageSelection		**PLC → NC**	**USInt**
实际齿轮档选择(1..5)			
SINUMERIK ONE Basic Program	LBP_Axis*.A_ActGearA..C		
SINUMERIK 840D sl	DB31,... .DBX16.0..2		

<Axis>.spindle.out.gearChangeoverDone		**PLC → NC**	**Bool**
齿轮档已切换			
SINUMERIK ONE Basic Program	LBP_Axis*.A_GearChangeOv		
SINUMERIK 840D sl	DB31,... .DBX16.3		

<Axis>.spindle.out.resyncEnc1		**PLC → NC**	**Bool**
请求重新同步主轴行程系统 1			
SINUMERIK ONE Basic Program	LBP_Axis*.A_Sp1Syn		
SINUMERIK 840D sl	DB31,... .DBX16.4		

<Axis>.spindle.out.resyncEnc2		**PLC → NC**	**Bool**
请求重新同步主轴行程系统 2			
SINUMERIK ONE Basic Program	LBP_Axis*.A_Sp2Syn		
SINUMERIK 840D sl	DB31,... .DBX16.5		

<Axis>.spindle.out.resyncEnc1ForPos		**PLC → NC**	**Bool**
请求在定位主轴前重新同步主轴行程系统 1			
SINUMERIK ONE Basic Program	LBP_Axis*.A_Sp1SynP		
SINUMERIK 840D sl	DB31,... .DBX17.4		

<Axis>.spindle.out.resyncEnc2ForPos		**PLC → NC**	**Bool**
请求在定位主轴前重新同步主轴行程系统 2			
SINUMERIK ONE Basic Program	LBP_Axis*.A_Sp2SynP		
SINUMERIK 840D sl	DB31,... .DBX17.5		

<Axis>.spindle.out.disableSpeedMonitor		**PLC → NC**	**Bool**
在切换齿轮档期间禁止转速监控			
SINUMERIK ONE Basic Program	LBP_Axis*.A_NoMonitorGear		
SINUMERIK 840D sl	DB31,... .DBX16.6		

<Axis>.spindle.out.deleteSValue		**PLC → NC**	**Bool**
删除 S 值：主轴沿着指定制动斜坡制动到静止			
SINUMERIK ONE Basic Program	LBP_Axis*.A_DeleteS		
SINUMERIK 840D sl	DB31,... .DBX16.7		

<Axis>.spindle.out.useFeedrateOvr		**PLC → NC**	**Bool**
为主轴转速控制使用进给倍率			
SINUMERIK ONE Basic Program	LBP_Axis*.A_FD_ORSp		
SINUMERIK 840D sl	DB31,... .DBX17.0		

<Axis>.spindle.out.invertM3M4		**PLC → NC**	**Bool**
M03/M04 取反：电机在相反方向上旋转			
SINUMERIK ONE Basic Program	LBP_Axis*.A_M3M4Inv		
SINUMERIK 840D sl	DB31,... .DBX17.6		

<Axis>.spindle.out.stop		**PLC → NC**	**Bool**
停止主轴运动			
SINUMERIK ONE Basic Program	LBP_Axis*.A_StopSpindle		
SINUMERIK 840D sl	DB31,... .DBX30.0		

<Axis>.spindle.out.startCw		**PLC → NC**	**Bool**
启动主轴顺时针旋转			
SINUMERIK ONE Basic Program	LBP_Axis*.A_StartCW		
SINUMERIK 840D sl	DB31,... .DBX30.1		

<Axis>.spindle.out.startCcw		**PLC → NC**	**Bool**
启动主轴逆时针旋转			
SINUMERIK ONE Basic Program	LBP_Axis*.A_StartCCW		
SINUMERIK 840D sl	DB31,... .DBX30.2		

<Axis>.spindle.out.autoGearStageChange		**PLC → NC**	**Bool**
激活自动齿轮档切换			
SINUMERIK ONE Basic Program	LBP_Axis*.A_AutoGearStep		
SINUMERIK 840D sl	DB31,... .DBX30.3		

<Axis>.spindle.out.startPosOperation		**PLC → NC**	**Bool**
主轴开始定位			
SINUMERIK ONE Basic Program	LBP_Axis*.A_PosSpindle		
SINUMERIK 840D sl	DB31,... .DBX30.4		

<Axis>.spindle.out.oscillEnable		**PLC → NC**	**Bool**
激活主轴的往复运动			
SINUMERIK ONE Basic Program	LBP_Axis*.A_OscilSpeed		
SINUMERIK 840D sl	DB31,... .DBX18.5		

<Axis>.spindle.out.oscillRequest		**PLC → NC**	**Bool**
请求 PLC 对往复运动的控制			
SINUMERIK ONE Basic Program	LBP_Axis*.A_OscilPLC		
SINUMERIK 840D sl	DB31,... .DBX18.4		

<Axis>.spindle.out.oscillSetDirCcw	**PLC → NC**	**Bool**
将往复方向设为“逆时针”		
SINUMERIK ONE Basic Program	LBP_Axis*.A_SetRotDirectLeft	
SINUMERIK 840D sl	DB31,... .DBX18.7	

<Axis>.spindle.out.oscillSetDirCw	**PLC → NC**	**Bool**
将往复方向设为“顺时针”		
SINUMERIK ONE Basic Program	LBP_Axis*.A_SetRotDirectRight	
SINUMERIK 840D sl	DB31,... .DBX18.6	

<Axis>.oscillation.out.externalReversal	**PLC → NC**	**Bool**
停止往复运动，在相反方向上移动往复轴		
SINUMERIK ONE Basic Program	LBP_Axis*.A_OscillAxExtRev	
SINUMERIK 840D sl	DB31,... .DBX28.0	

<Axis>.oscillation.out.selectReversalPoint	**PLC → NC**	**Bool**
选择返向点： 0 - 第一个返向点； 1 - 第二个返向点		
SINUMERIK ONE Basic Program	LBP_Axis*.A_SetRP	
SINUMERIK 840D sl	DB31,... .DBX28.3	

<Axis>.oscillation.out.activateReversalPoint	**PLC → NC**	**Bool**
激活选中的返向点		
SINUMERIK ONE Basic Program	LBP_Axis*.A_ChangeRP	
SINUMERIK 840D sl	DB31,... .DBX28.4	

<Axis>.oscillation.out.stopAtNextReversalPoint	**PLC → NC**	**Bool**
停在下一个返向点上		
SINUMERIK ONE Basic Program	LBP_Axis*.A_StopRP	
SINUMERIK 840D sl	DB31,... .DBX28.5	

<Axis>.coupling.in.gantryActive	**NC → PLC**	**Bool**
龙门轴：轴是龙门轴系统的一部分。		
SINUMERIK ONE Basic Program	LBP_Axis*.E_GantAxis	
SINUMERIK 840D sl	DB31,... .DBX101.7	

<Axis>.coupling.in.gantryLeadingActive	**NC → PLC**	**Bool**
龙门轴：轴是主动轴		
SINUMERIK ONE Basic Program	LBP_Axis*.E_GantLeadAxis	
SINUMERIK 840D sl	DB31,... .DBX101.6	

<Axis>.coupling.in.gantrySyncRunReady	**NC → PLC**	**Bool**
龙门轴：可以开始同步		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SynRunStart	
SINUMERIK 840D sl	DB31,... .DBX101.4	

<Axis>.coupling.in.gantrySyncOk	**NC → PLC**	**Bool**
龙门轴：该系统的所有轴都已经同步		
SINUMERIK ONE Basic Program	LBP_Axis*.E_GantGroupSyn	
SINUMERIK 840D sl	DB31,... .DBX101.5	

<Axis>.coupling.in.gantryWarningLimitExceeded	**NC → PLC**	**Bool**
龙门轴：超出报警限值		
SINUMERIK ONE Basic Program	LBP_Axis*.E_GantWarn	
SINUMERIK 840D sl	DB31,... .DBX101.3	

<Axis>.coupling.in.gantryCutOffLimitExceeded	**NC → PLC**	**Bool**
龙门轴：超出关闭限值		
SINUMERIK ONE Basic Program	LBP_Axis*.E_GantCOLimit	
SINUMERIK 840D sl	DB31,... .DBX101.2	

<Axis>.coupling.in.msActive	**NC → PLC**	**Bool**
主从耦合：耦合已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.E_MS	
SINUMERIK 840D sl	DB31,... .DBX96.7	

<Axis>.coupling.in.msCompCtrlActive	**NC → PLC**	**Bool**
主从耦合：转矩平衡控制器已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.E_MSCompContr	
SINUMERIK 840D sl	DB31,... .DBX96.4	

<Axis>.coupling.in.msSpeedDifferenceCoarseReached	**NC → PLC**	**Bool**
主从耦合：转速差位于粗公差范围内 (MD37270 $MS_VELO_TOL_COARSE)		
SINUMERIK ONE Basic Program LBP_Axis*.E_MSCoarse		
SINUMERIK 840D sl DB31,... .DBX96.3		

<Axis>.coupling.in.msSpeedDifferenceFineReached	**NC → PLC**	**Bool**
主从耦合：转速差位于精公差范围内(MD37272 $MS_VELO_TOL_FINE)		
SINUMERIK ONE Basic Program LBP_Axis*.E_MSFine		
SINUMERIK 840D sl DB31,... .DBX96.2		

<Axis>.coupling.in.mcsCouplingActive	**NC → PLC**	**Bool**
机床坐标系耦合：耦合已生效		
SINUMERIK 840D sl DB31,... .DBX97.1		

<Axis>.coupling.in.mcsFollowingAxisModeActive	**NC → PLC**	**Bool**
MCS 耦合：轴为从动轴		
SINUMERIK 840D sl DB31,... .DBX97.0		

<Axis>.coupling.in.mcsMirroringActive	**NC → PLC**	**Bool**
机床坐标系耦合：镜像已生效		
SINUMERIK 840D sl DB31,... .DBX97.2		

<Axis>.coupling.in.mcsOffsetChanged	**NC → PLC**	**Bool**
机床坐标系耦合：偏移已更改		
SINUMERIK 840D sl DB31,... .DBX97.3		

<Axis>.coupling.in.leadingSpindleActive	**NC → PLC**	**Bool**
通用耦合：主轴为主动主轴		
SINUMERIK ONE Basic Program LBP_Axis*.E_MasterSp		
SINUMERIK 840D sl DB31,... .DBX99.0		

<Axis>.coupling.in.followingSpindleActive	**NC → PLC**	**Bool**
通用耦合：主轴为从动主轴		
SINUMERIK ONE Basic Program LBP_Axis*.E_SlaveSp		
SINUMERIK 840D sl DB31,... .DBX99.1		

<Axis>.coupling.in.syncSpindleCtrlMode	**NC → PLC**	**Bool**
同步主轴耦合：从动主轴由主动主轴控制：0 - 设定值，1- 实际值 (MD21310 $MC_COUPLING_MODE_1)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_ActValCoupling	
SINUMERIK 840D sl	DB31,... .DBX98.2	

<Axis>.coupling.in.syncCoarseReached	**NC → PLC**	**Bool**
通用耦合：实际值与设定值（位置或转速）之差已达到“粗公差范围”(MD37200 $MA_COUPLE_POS_TOL_COARSE, MD37220 $MA_COUPLE_VELO_TOL_COARSE)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SyncronCoarse	
SINUMERIK 840D sl	DB31,... .DBX98.1	

<Axis>.coupling.in.syncFineReached	**NC → PLC**	**Bool**
通用耦合：实际值与设定值（位置或转速）之差已达到“精公差范围”(MD37210 $MA_COUPLE_POS_TOL_FINE, MD37230 $MA_COUPLE_VELO_TOL_FINE)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SyncronFine	
SINUMERIK 840D sl	DB31,... .DBX98.0	

<Axis>.coupling.in.syncCoarse2Reached	**NC → PLC**	**Bool**
通用耦合：用于实际位置控制的同步运行已达到“粗公差范围”(MD37202 $MA_COUPLE_POS_TOL_COARSE_2)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_Syncron2Coarse	
SINUMERIK 840D sl	DB31,... .DBX103.5	

<Axis>.coupling.in.syncFine2Reached	**NC → PLC**	**Bool**
通用耦合：用于实际位置控制的同步运行已达到“精公差范围”(MD37212 $MA_COUPLE_POS_TOL_FINE_2)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_Syncron2Fine	
SINUMERIK 840D sl	DB31,... .DBX103.4	

<Axis>.coupling.in.motionOverlaid	**NC → PLC**	**Bool**
通用耦合：从动主轴上的附加叠加生效（例如：主动主轴与从动主轴之间的起始角偏移）		
SINUMERIK ONE Basic Program	LBP_Axis*.E_OverlayMotion	
SINUMERIK 840D sl	DB31,... .DBX98.4	

<Axis>.coupling.in.correctionActive	**NC → PLC**	**Bool**
通用耦合：同步运行补偿生效		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SyncRunCorrIncl	
SINUMERIK 840D sl	DB31,... .DBX103.0	

<Axis>.coupling.in.correctionDone	**NC → PLC**	**Bool**
通用耦合：跟踪运行补偿 ($AA_COUP_CORR)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SyncRunCorrDOut	
SINUMERIK 840D sl	DB31,... .DBX99.2	

<Axis>.coupling.in.processActive	**NC → PLC**	**Bool**
通用耦合：同步生效		
SINUMERIK ONE Basic Program	LBP_Axis*.E_Sync	
SINUMERIK 840D sl	DB31,... .DBX99.4	

<Axis>.coupling.in.maxSpeedReached	**NC → PLC**	**Bool**
通用耦合：已达到最大转速		
SINUMERIK ONE Basic Program	LBP_Axis*.E_MSR	
SINUMERIK 840D sl	DB31,... .DBX99.5	

<Axis>.coupling.in.maxAccelReached	**NC → PLC**	**Bool**
通用耦合：已达到最大加速度		
SINUMERIK ONE Basic Program	LBP_Axis*.E_MAR	
SINUMERIK 840D sl	DB31,... .DBX99.6	

<Axis>.coupling.in.speedThresholdReached	**NC → PLC**	**Bool**
通用耦合：从动主轴已达到或超出转速报警阈值		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SWL	
SINUMERIK 840D sl	DB31,... .DBX98.5	

<Axis>.coupling.in.accelThresholdReached	**NC → PLC**	**Bool**
通用耦合：从动主轴已达到或超出了加速度报警阈值		
SINUMERIK ONE Basic Program	LBP_Axis*.E_AWL	
SINUMERIK 840D sl	DB31,... .DBX98.6	

<Axis>.coupling.in.accelReached	**NC → PLC**	**Bool**
通用耦合：从动轴达到或超出了为“加速轴”显示的最小值 (MD37560 $MA_EG_ACC_TOL)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_AxisAccel	
SINUMERIK 840D sl	DB31,... .DBX99.3	

<Axis>.coupling.out.gantryStartSyncRun	**PLC → NC**	**Bool**
龙门轴：开始同步		
SINUMERIK ONE Basic Program	LBP_Axis*.A_StartGantSynRun	
SINUMERIK 840D sl	DB31,... .DBX29.4	

<Axis>.coupling.out.gantryDisableSync	**PLC → NC**	**Bool**
龙门轴：禁止自动同步		
SINUMERIK ONE Basic Program	LBP_Axis*.A_NoAutoSync	
SINUMERIK 840D sl	DB31,... .DBX29.5	

<Axis>.coupling.out.msActivate	**PLC → NC**	**Bool**
主从耦合：激活耦合		
SINUMERIK ONE Basic Program	LBP_Axis*.A_MS	
SINUMERIK 840D sl	DB31,... .DBX24.7	

<Axis>.coupling.out.msTorqueComp	**PLC → NC**	**Bool**
主从耦合：请求激活转矩平衡控制器		
SINUMERIK ONE Basic Program	LBP_Axis*.A_TorqComp	
SINUMERIK 840D sl	DB31,... .DBX24.4	

<Axis>.coupling.out.mcsDecouple	**PLC → NC**	**Bool**
机床坐标系耦合：请求关闭耦合		
SINUMERIK 840D sl	DB31,... .DBX24.2	

<Axis>.coupling.out.mcsCollisionProtection	**PLC → NC**	**Bool**
机床坐标系耦合：请求轴对的碰撞保护		
SINUMERIK 840D sl	DB31,... .DBX24.3	

<Axis>.coupling.out.syncSpindleResync	**PLC → NC**	**Bool**
同步主轴耦合：重新同步：恢复所编写的偏移 ($P_COUP_OFFS)		
SINUMERIK ONE Basic Program	LBP_Axis*.A_Resynchronize	
SINUMERIK 840D sl	DB31,... .DBX31.4	

<Axis>.coupling.out.syncSpindleSyncDisable	**PLC → NC**	**Bool**
机床坐标系耦合：PLC 禁止在程序偏移指令中指定的从动主轴同步运动量。		
SINUMERIK ONE Basic Program	LBP_Axis*.A_DisSync	
SINUMERIK 840D sl	DB31,... .DBX31.5	

<Axis>.coupling.out.syncCorrection	**PLC → NC**	**Bool**
通用耦合：请求同步补偿		
SINUMERIK ONE Basic Program	LBP_Axis*.A_Tracksync	
SINUMERIK 840D sl	DB31,... .DBX31.6	

<Axis>.coupling.out.syncResetCorrection	**PLC → NC**	**Bool**
通用耦合：删除同步补偿		
SINUMERIK ONE Basic Program	LBP_Axis*.A_DelSyncRunCorr	
SINUMERIK 840D sl	DB31,... .DBX31.7	

<Axis>.coupling.out.enableFollowOverlay	**PLC → NC**	**Bool**
通用耦合：使能从动轴叠加		
SINUMERIK ONE Basic Program	LBP_Axis*.A_FollowAxOv	
SINUMERIK 840D sl	DB31,... .DBX26.4	

<Axis>.plcCtrl.in.plcCtrlActive	**NC → PLC**	**Bool**
PLC 控制轴：轴由 PLC 控制进行单轴自控运行		
SINUMERIK ONE Basic Program	LBP_Axis*.E_PLCCtrlAx	
SINUMERIK 840D sl	DB31,... .DBX63.1	

<Axis>.plcCtrl.in.stopped	**NC → PLC**	**Bool**
PLC 控制轴：轴已停止		
SINUMERIK ONE Basic Program	LBP_Axis*.E_AxStop	
SINUMERIK 840D sl	DB31,... .DBX63.2	

<Axis>.plcCtrl.in.resetDone	**NC → PLC**	**Bool**
PLC 控制轴：轴复位已执行		
SINUMERIK ONE Basic Program	LBP_Axis*.E_AxReset	
SINUMERIK 840D sl	DB31,... .DBX63.0	

<Axis>.plcCtrl.in.plcCtrlPermanent	**NC → PLC**	**Bool**
轴始终由 PLC 控制 $MA_BASE_FUNCTION_MASK = 0x20 位 5=1		
SINUMERIK ONE Basic Program	LBP_Axis*.E_PLCAxDedic	
SINUMERIK 840D sl	DB31,... .DBX71.7	

<Axis>.plcCtrl.out.requestPlcCtrl	**PLC → NC**	**Bool**
请求系统由 PLC 控制单轴自控运行		
SINUMERIK ONE Basic Program	LBP_Axis*.A_PLCAxis	
SINUMERIK 840D sl	DB31,... .DBX28.7	

<Axis>.plcCtrl.out.stop	**PLC → NC**	**Bool**
PLC 控制轴：沿着制动斜坡停止轴		
SINUMERIK ONE Basic Program	LBP_Axis*.A_Stop	
SINUMERIK 840D sl	DB31,... .DBX28.6	

<Axis>.plcCtrl.out.reset	**PLC → NC**	**Bool**
PLC 控制轴：请求复位		
SINUMERIK ONE Basic Program	LBP_Axis*.A_AxReset	
SINUMERIK 840D sl	DB31,... .DBX28.1	

<Axis>.plcCtrl.out.resume	**PLC → NC**	**Bool**
PLC 控制轴：请求继续被中断的运行		
SINUMERIK ONE Basic Program	LBP_Axis*.A_AxResume	
SINUMERIK 840D sl	DB31,... .DBX28.2	

<Axis>.safetyInfoChan.in.slaActive	**NC → PLC**	**Bool**
安全功能“Safely Limited Acceleration”已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.SLA_Sel	
SINUMERIK 840D sl	DB31,... .DBX164.0	

<Axis>.safetyInfoChan.in.slsBit0Active	**NC → PLC**	**Bool**
四个 SLS 限值中有一个生效；和位 0 配合工作		
SINUMERIK ONE Basic Program	LBP_Axis*.SLS_LimBit0_Sel	
SINUMERIK 840D sl	DB31,... .DBX164.1	

<Axis>.safetyInfoChan.in.slsBit1Active	**NC → PLC**	**Bool**
四个 SLS 限值中有一个生效；和位 1 配合工作		
SINUMERIK ONE Basic Program	LBP_Axis*.SLS_LimBit1_Sel	
SINUMERIK 840D sl	DB31,... .DBX164.2	

<Axis>.safetyInfoChan.in.esrRetractRequested	**NC → PLC**	**Bool**
ESR 回退已请求		
SINUMERIK ONE Basic Program	LBP_Axis*.ESR_Req	
SINUMERIK 840D sl	DB31,... .DBX164.6	

<Axis>.safetyInfoChan.in.safetyMessageActive	**NC → PLC**	**Bool**
存在安全消息		
SINUMERIK ONE Basic Program	LBP_Axis*.SF_Message_Act	
SINUMERIK 840D sl	DB31,... .DBX164.7	

<Axis>.safetyInfoChan.in.stoActive	**NC → PLC**	**Bool**
安全功能"Safe Torque Off"已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.STO_Act	
SINUMERIK 840D sl	DB31,... .DBX165.0	

<Axis>.safetyInfoChan.in.ss1Active	**NC → PLC**	**Bool**
安全功能"Safe Stop 1 "已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.SS1_Act	
SINUMERIK 840D sl	DB31,... .DBX165.1	

<Axis>.safetyInfoChan.in.ss2Active	**NC → PLC**	**Bool**
安全功能"Safe Stop 2 "已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.SS2_Act	
SINUMERIK 840D sl	DB31,... .DBX165.2	

<Axis>.safetyInfoChan.in.sosActive	**NC → PLC**	**Bool**
安全功能"Safe Operating Stop"已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.SOS_Act	
SINUMERIK 840D sl	DB31,... .DBX165.3	

<Axis>.safetyInfoChan.in.slsActive	**NC → PLC**	**Bool**
安全功能"Safely Limited Speed"已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.SLS_Act	
SINUMERIK 840D sl	DB31,... .DBX165.4	

<Axis>.safetyInfoChan.in.sosSelected	**NC → PLC**	**Bool**
安全功能"Safe Operating Stop"已选中，但还没有生效		
SINUMERIK ONE Basic Program	LBP_Axis*.SOS_Sel	
SINUMERIK 840D sl	DB31,... .DBX165.5	

<Axis>.safetyInfoChan.in.slsSelected	**NC → PLC**	**Bool**
安全功能"Safely Limited Speed"已选中，但还没有生效		
SINUMERIK ONE Basic Program	LBP_Axis*.SLS_Sel	
SINUMERIK 840D sl	DB31,... .DBX165.6	

<Axis>.safetyInfoChan.in.safetyFaultActive	**NC → PLC**	**Bool**
发生内部事件（SINAMICS 中安全性错误）		
SINUMERIK ONE Basic Program	LBP_Axis*.InternalEvent	
SINUMERIK 840D sl	DB31,... .DBX165.7	

<Axis>.safetyInfoChan.in.sdiPositiveSelected	**NC → PLC**	**Bool**
安全功能"Safe Direction +"已选中，但还没有生效		
SINUMERIK ONE Basic Program	LBP_Axis*.SDI_P_Sel	
SINUMERIK 840D sl	DB31,... .DBX166.0	

<Axis>.safetyInfoChan.in.sdiNegativeSelected	**NC → PLC**	**Bool**
安全功能"Safe Direction -"已选中，但还没有生效		
SINUMERIK ONE Basic Program	LBP_Axis*.SDI_N_Sel	
SINUMERIK 840D sl	DB31,... .DBX166.1	

<Axis>.safetyInfoChan.in.testStopActive	**NC → PLC**	**Bool**
强制潜在故障检查已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.TestStopAct	
SINUMERIK 840D sl	DB31,... .DBX166.4	

<Axis>.safetyInfoChan.in.testStopRequired	**NC → PLC**	**Bool**
强制潜在故障检查已请求		
SINUMERIK ONE Basic Program	LBP_Axis*.TestStopReq	
SINUMERIK 840D sl	DB31,... .DBX166.5	

<Axis>.safetyInfoChan.in.slpRangeSelected	**NC → PLC**	**Bool**
TRUE: SLP 位置范围 2 已选中; FALSE: SLP 位置范围 1 已选中		
SINUMERIK ONE Basic Program	LBP_Axis*.SLP_LimBit0_Sel	
SINUMERIK 840D sl	DB31,... .DBX167.4	

<Axis>.safetyInfoChan.in.slpSelectedAndUserAgreement	**NC → PLC**	**Bool**
SLP 已选中，用户确认已设置		
SINUMERIK ONE Basic Program	LBP_Axis*.SLP_Sel	
SINUMERIK 840D sl	DB31,... .DBX167.7	

<Axis>.safetyInfoChan.in.slsLimit	**NC → PLC**	**DInt**
SLS 转速限值， 32 位分辨率，带正负号位		

<Axis>.safetyInfoChan.in.ss2eActive	**NC → PLC**	**Bool**
安全功能 SS2E 已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.SLS_LimBit0_Sel	
SINUMERIK 840D sl	DB31,... .DBX172.3	

<Axis>.safetyInfoChan.in.ss2esrActive	**NC → PLC**	**Bool**
Safe Stop 2 扩展停止和回退 (SS2ESR) 已生效		
SINUMERIK ONE Basic Program	LBP_Axis*.SLS_LimBit1_Sel	
SINUMERIK 840D sl	DB31,... .DBX172.4	

<Axis>.safetyInfoChan.in.acceptanceTestSlpSelected	**NC → PLC**	**Bool**
SLP 验收测试已选中		
SINUMERIK ONE Basic Program	LBP_Axis*.ESR_Req	
SINUMERIK 840D sl	DB31,... .DBX172.6	

<Axis>.safetyInfoChan.in.acceptanceTestModeSelected	**NC → PLC**	**Bool**
验收测试模式已选中		
SINUMERIK ONE Basic Program	LBP_Axis*.SF_Message_Act	
SINUMERIK 840D sl	DB31,... .DBX172.7	

<Axis>.safetyInfoChan.in.brakeTest	**NC → PLC**	**Bool**
制动测试已选中		
SINUMERIK ONE Basic Program	LBP_Axis*.STO_Act	
SINUMERIK 840D sl	DB31,... .DBX173.0	

<Axis>.safetyInfoChan.in.setpointInput	**NC → PLC**	**Bool**
驱动设定值的指定方式：由 SBT 功能指定或由外部信号源（控制器）指定		
SINUMERIK ONE Basic Program	LBP_Axis*.SS1_Act	
SINUMERIK 840D sl	DB31,... .DBX173.1	

<Axis>.safetyInfoChan.in.activeBrake	**NC → PLC**	**Bool**
TRUE: 制动器 2 测试生效; FALSE: 制动器 1 测试生效		
SINUMERIK ONE Basic Program	LBP_Axis*.SS2_Act	
SINUMERIK 840D sl	DB31,... .DBX173.2	

<Axis>.safetyInfoChan.in.brakeTestActive	**NC → PLC**	**Bool**
制动测试生效		
SINUMERIK ONE Basic Program	LBP_Axis*.SOS_Act	
SINUMERIK 840D sl	DB31,... .DBX173.3	

<Axis>.safetyInfoChan.in.brakeTestResult	**NC → PLC**	**Bool**
TRUE: 制动测试成功; FALSE: 制动测试失败		
SINUMERIK ONE Basic Program	LBP_Axis*.SLS_Act	
SINUMERIK 840D sl	DB31,... .DBX173.4	

<Axis>.safetyInfoChan.in.brakeTestCompleted	**NC → PLC**	**Bool**
TRUE: 制动测试已结束; FALSE: 制动测试不完整		
SINUMERIK ONE Basic Program	LBP_Axis*.SOS_Sel	
SINUMERIK 840D sl	DB31,... .DBX173.5	

<Axis>.safetyInfoChan.in.externalBrakeRequested	**NC → PLC**	**Bool**
TRUE: 制动器闭合; FALSE: 制动器打开		
SINUMERIK ONE Basic Program	LBP_Axis*.SLS_Sel	
SINUMERIK 840D sl	DB31,... .DBX173.6	

<Axis>.safetyInfoChan.in.currentLoadSign	**NC → PLC**	**Bool**
当前负载的正负号: TRUE: 负; FALSE: 正		
SINUMERIK ONE Basic Program	LBP_Axis*.InternalEvent	
SINUMERIK 840D sl	DB31,... .DBX173.7	

<Axis>.safetyCtrlChan.out.testStop	**PLC → NC**	**Bool**
强制潜在故障检查已选中		
SINUMERIK ONE Basic Program	LBP_Axis*.TeststopStart	
SINUMERIK 840D sl	DB31,... .DBX140.0	

<Axis>.safetyCtrlChan.out.closeBrake	**PLC → NC**	**Bool**
闭合内部制动器		
SINUMERIK ONE Basic Program	LBP_Axis*.CloseBrake	
SINUMERIK 840D sl	DB31,... .DBX140.5	

<Axis>.safetyCtrlChan.out.selectBrakeTest	**PLC → NC**	**Bool**
制动测试已选中		
SINUMERIK ONE Basic Program	LBP_Axis*.SelectionSBT	
SINUMERIK 840D sl	DB31,... .DBX143.0	

<Axis>.safetyCtrlChan.out.startBrakeTest	**PLC → NC**	**Bool**
制动测试启动已请求		
SINUMERIK ONE Basic Program	LBP_Axis*.BT_Start	
SINUMERIK 840D sl	DB31,... .DBX143.1	

<Axis>.safetyCtrlChan.out.brakeSelection		**PLC → NC**	**Bool**
TRUE: 测试制动器 2 已选中; FALSE: 测试制动器 1 已选中			
SINUMERIK ONE Basic Program	LBP_Axis*.Brake_1_2		
SINUMERIK 840D sl	DB31,... .DBX143.2		

<Axis>.safetyCtrlChan.out.selectDirectionOfRotation		**PLC → NC**	**Bool**
TRUE: 负向已选中; FALSE: 正向已选中			
SINUMERIK ONE Basic Program	LBP_Axis*.DirOfROT		
SINUMERIK 840D sl	DB31,... .DBX143.3		

<Axis>.safetyCtrlChan.out.selectTestSequence		**PLC → NC**	**Bool**
TRUE: 测试序列 2 已选中; FALSE: 测试序列 1 已选中			
SINUMERIK ONE Basic Program	LBP_Axis*.Testseq_1_2		
SINUMERIK 840D sl	DB31,... .DBX143.4		

<Axis>.safetyCtrlChan.out.externalBrakeClosed		**PLC → NC**	**Bool**
TRUE: 外部制动器已闭合; FALSE: 外部制动器已打开			
SINUMERIK ONE Basic Program	LBP_Axis*.BrakeClosed		
SINUMERIK 840D sl	DB31,... .DBX143.5		

<Axis>.interchange.in.requested		**NC → PLC**	**Bool**
已请求交换进给轴/主轴			
SINUMERIK ONE Basic Program	LBP_Axis*.E_PLCType		
SINUMERIK 840D sl	DB31,... .DBB68.4		

<Axis>.interchange.in.possible		**NC → PLC**	**Bool**
允许交换进给轴/主轴			
SINUMERIK ONE Basic Program	LBP_Axis*.E_ChPoss		
SINUMERIK 840D sl	DB31,... .DBB68.5		

<Axis>.interchange.in.neutralAxis		**NC → PLC**	**Bool**
中立进给轴/主轴			
SINUMERIK ONE Basic Program	LBP_Axis*.E_NeutrASp		
SINUMERIK 840D sl	DB31,... .DBB68.6		

<Axis>.interchange.in.assignedToPlc	**NC → PLC**	**Bool**
PLC 进给轴/主轴		
SINUMERIK ONE Basic Program	LBP_Axis*.E_PLCASp	
SINUMERIK 840D sl	DB31,... .DBB68.7	

<Axis>.interchange.in.assignedChanNumber	**NC → PLC**	**USInt**
NC 轴被分配给的通道的编号（通道号或 0：中性轴）		
SINUMERIK ONE Basic Program	LBP_Axis*.E_NCASpChanA..D	
SINUMERIK 840D sl	DB31,... .DBB68.0..3	

<Axis>.interchange.out.assignToChanSelection	**PLC → NC**	**USInt**
为 NC 轴分配通道（通道号，0 表示中立轴）		
SINUMERIK ONE Basic Program	LBP_Axis*.A_NCASpChanA..D	
SINUMERIK 840D sl	DB31,... .DBB8.0..3	

<Axis>.interchange.out.assignToPlc	**PLC → NC**	**Bool**
请求将轴/主轴分配给 PLC		
SINUMERIK ONE Basic Program	LBP_Axis*.A_PLCASp	
SINUMERIK 840D sl	DB31,... .DBB8.7	

<Axis>.interchange.out.requireSwitchOver	**PLC → NC**	**Bool**
脉冲沿切换 0 → 1：激活切换		
SINUMERIK ONE Basic Program	LBP_Axis*.A_NCASpStrobe	
SINUMERIK 840D sl	DB31,... .DBB8.4	

<Axis>.safety.in.infoCtrlChanActive	**NC → PLC**	**Bool**
通过 SIC 集成已在 NCK 中激活 (MD37950 $MA_SAFE_INFO_ENABLE，位 0)		
SINUMERIK ONE Basic Program	LBP_Axis*.E_SI_Drv_SICSCC	
SINUMERIK 840D sl	DB31,... .DBX70.5	

<Axis>.safety.in.brakeTestModeEnabled	**NC → PLC**	**Bool**
轴已使能进行制动测试		
SINUMERIK ONE Basic Program	LBP_Axis*.E_BrakeTest	
SINUMERIK 840D sl	DB31,... .DBX71.0	

<Axis>.safety.out.enableBrakeTestMode	**PLC → NC**	**Bool**
Safety Integrated (SI): 开始制动测试		
SINUMERIK ONE Basic Program	LBP_Axis*.A_StartBrakeTest	
SINUMERIK 840D sl	DB31,... .DBX11.0	

<Axis>.safety.out.activateBrakeRampForSs2e	**PLC → NC**	**Bool**
选择 SS2E 时通过 NCK 斜坡制动 (MD36959 MA_SAFE_BRAKE_RAMP_TIME)		
SINUMERIK ONE Basic Program	LBP_Axis*.A_ActivateBrakeRampForSs2e	
SINUMERIK 840D sl	DB31,... .DBX34.3	

<Axis>.safety.out.setpointLimitSelection	**PLC → NC**	**USInt**
选择设定速度限值的计算系数 (0..3) (MD36933 $MA_SAFE_DES_VELO_LIMIT)		
SINUMERIK ONE Basic Program	LBP_Axis*.A_SI_SetpLimBit0	
SINUMERIK 840D sl	DB31,... .DBX34.0..1	

12.2.3 NCK_typeChan

<Chan>.config
通道配置数据

<Chan>.basic
通道的通用信号

<Chan>.gCommandGroup	**NC → PLC**
G 指令组信号	

<Chan>.geoAxis1	
几何轴 1 信号	
SINUMERIK ONE Basic Program	LBP_Chan*.E_Geo[1]

<Chan>.geoAxis2	
几何轴 2 信号	
SINUMERIK ONE Basic Program	LBP_Chan*.E_Geo[2]

<Chan>.geoAxis3	
几何轴 3 信号	
SINUMERIK ONE Basic Program LBP_Chan*.E_Geo[3]	

<Chan>.grinding	
通道的磨削信号	

<Chan>.orientationAxis1	
摆动轴 1 的信号	
SINUMERIK ONE Basic Program LBP_Chan*.E_Ori[1]	

<Chan>.orientationAxis2	
摆动轴 2 的信号	
SINUMERIK ONE Basic Program LBP_Chan*.E_Ori[2]	

<Chan>.orientationAxis3	
摆动轴 3 的信号	
SINUMERIK ONE Basic Program LBP_Chan*.E_Ori[3]	

<Chan>.toolMgmt	
刀具管理信号	

<Chan>.auxMDynFuncs	**NC → PLC**
动态 M 功能解码的信号	

<Chan>.auxMFuncs	**NC → PLC**
M 功能信号[1..5]	

<Chan>.auxSFuncs	**NC → PLC**
S 功能信号[1..3]	

<Chan>.auxTFuncs	**NC → PLC**
T 功能信号[1..3]	

<Chan>.auxDFuncs	**NC → PLC**
D 功能信号[1..3]	

<Chan>.auxDLFuncs	**NC → PLC**
DL 功能信号[1..1]	

<Chan>.auxHFuncs	**NC → PLC**
H 功能信号[1..3]	

<Chan>.auxFFuncs	**NC → PLC**
F 功能信号[1..6]	

<Chan>.protAreas
保护区信号 [1..10]

<Chan>.nibbelingAndPunching
冲压操作信号

<Chan>.cycles
循环信号

<Chan>.progEvent
PROG_EVENT 信号

<Chan>.synActs
同步动作信号

<Chan>.nibbelingAndPunching.in	**NC → PLC**
收到的冲压信号	

<Chan>.nibbelingAndPunching.out	**PLC → NC**
发送的冲压信号	

<Chan>.cycles.in	**NC → PLC**
收到的循环信号	

<Chan>.cycles.out	**PLC → NC**
发送的循环信号	

<Chan>.basic.in	**NC → PLC**
从通道收到的通用信号	

<Chan>.basic.out	**PLC → NC**
发送给通道的通用信号	

<Chan>.auxMFuncs.in	**NC → PLC**
收到的 M 功能信号 [1..5]	

<Chan>.auxSFuncs.in	**NC → PLC**
收到的 S 功能信号 [1..3]	

<Chan>.auxTFuncs.in	**NC → PLC**
收到的 T 功能信号 [1..3]	

<Chan>.auxDFuncs.in	**NC → PLC**
收到的 D 功能信号 [1..3]	

<Chan>.auxDLFuncs.in	**NC → PLC**
收到的 DL 功能信号 [1..1]	

<Chan>.auxHFuncs.in	**NC → PLC**
收到的 H 功能信号 [1..3]	

<Chan>.auxFFuncs.in	**NC → PLC**
收到的 F 功能信号 [1..6]	

<Chan>.auxMDynFuncs.in	**NC → PLC**
收到的动态 M 功能解码信号	

<Chan>.protAreas.in	**NC → PLC**
收到的保护区[1..10]信号	

<Chan>.protAreas.out	**PLC → NC**
发送的保护区域信号[1..10]	

<Chan>.synActs.in	**NC → PLC**
收到的同步动作信号	

<Chan>.synActs.out	**PLC → NC**
发送的同步动作信号	

<Chan>.toolMgmt.in	**NC → PLC**
收到的刀具管理信号	

<Chan>.toolMgmt.out	**PLC → NC**
发送的刀具管理信号	

<Chan>.progEvent.in	**NC → PLC**
收到的 PROG_EVENT 信号	

<Chan>.gCommandGroup.in	**NC → PLC**
收到的 G 指令组信号	

<Chan>.geoAxis.in	**NC → PLC**
收到的几何轴信号 SINUMERIK ONE Basic Program LBP_Chan*.E_Geo	

<Chan>.geoAxis.out	**PLC → NC**
发送的几何轴信号 SINUMERIK ONE Basic Program LBP_Chan*.A_Geo	

<Chan>.grinding.in	**NC → PLC**
从通道发出的磨削信号	

<Chan>.grinding.out	**PLC → NC**
发送给通道的磨削信号	

<Chan>.orientationAxis.in	**NC → PLC**
收到的摆动轴信号 SINUMERIK ONE Basic Program LBP_Chan*.E_Ori	

<Chan>.orientationAxis.out	**PLC → NC**	
发送的摆动轴信号		
SINUMERIK ONE Basic Program	LBP_Chan*.A_Ori	

<Chan>.config.registered	**NC → PLC**	**Bool**
通道已注册		

<Chan>.config.chanActive	**NC → PLC**	**Bool**
通道已生效		

<Chan>.config.chanNumber	**NC → PLC**	**USInt**
通道号		

<Chan>.basic.in.ready	**NC → PLC**	**Bool**
通道已运行就绪		
SINUMERIK ONE Basic Program	LBP_Chan*.E_ChanRO	
SINUMERIK 840D sl	DB21,... .DBX36.5	

<Chan>.basic.in.stateActive	**NC → PLC**	**Bool**
通道状态“生效”		
SINUMERIK ONE Basic Program	LBP_Chan*.E_ChanActive	
SINUMERIK 840D sl	DB21,... .DBX35.5	

<Chan>.basic.in.stateInterrupted	**NC → PLC**	**Bool**
通道状态“中断”		
SINUMERIK ONE Basic Program	LBP_Chan*.E_ChanInterrupt	
SINUMERIK 840D sl	DB21,... .DBX35.6	

<Chan>.basic.in.stateReset	**NC → PLC**	**Bool**
通道状态“复位”		
SINUMERIK ONE Basic Program	LBP_Chan*.E_ChanReset	
SINUMERIK 840D sl	DB21,... .DBX35.7	

<Chan>.basic.in.progStateRunning		**NC → PLC**	**Bool**
程序状态“正在运行”			
SINUMERIK ONE Basic Program	LBP_Chan*.E_ProgRunning		
SINUMERIK 840D sl	DB21,... .DBX35.0		

<Chan>.basic.in.progStateWaiting		**NC → PLC**	**Bool**
程序状态“等待”			
SINUMERIK ONE Basic Program	LBP_Chan*.E_ProgWait		
SINUMERIK 840D sl	DB21,... .DBX35.1		

<Chan>.basic.in.progStateStopped		**NC → PLC**	**Bool**
程序状态“停止”			
SINUMERIK ONE Basic Program	LBP_Chan*.E_ProgStop		
SINUMERIK 840D sl	DB21,... .DBX35.2		

<Chan>.basic.in.progStateInterrupted		**NC → PLC**	**Bool**
程序状态“中断”			
SINUMERIK ONE Basic Program	LBP_Chan*.E_ProgInterrupt		
SINUMERIK 840D sl	DB21,... .DBX35.3		

<Chan>.basic.in.progStateAborted		**NC → PLC**	**Bool**
程序状态“终止”			
SINUMERIK ONE Basic Program	LBP_Chan*.E_ProgAborted		
SINUMERIK 840D sl	DB21,... .DBX35.4		

<Chan>.basic.in.allAxesReferenced		**NC → PLC**	**Bool**
该通道的所有轴（线性轴和旋转轴）都已回参考点			
SINUMERIK ONE Basic Program	LBP_Chan*.E_AxesRef		
SINUMERIK 840D sl	DB21,... .DBX36.2		

<Chan>.basic.in.ncStartRequested		**NC → PLC**	**Bool**
已请求 NC 启动			
SINUMERIK ONE Basic Program	LBP_Chan*.E_StartReq		
SINUMERIK 840D sl	DB21,... .DBX36.0		

<Chan>.basic.in.ncStopRequested		**NC → PLC**	**Bool**
已请求 NC 停止			
SINUMERIK ONE Basic Program	LBP_Chan*.E_StopReq		
SINUMERIK 840D sl	DB21,... .DBX36.1		

<Chan>.basic.in.retraceSupportModeActive		**NC → PLC**	**Bool**
沿轮廓恢复加工功能(RESU)：加工中断后可以沿着轮廓恢复加工			
SINUMERIK ONE Basic Program	LBP_Chan*.f32_1		
SINUMERIK 840D sl	DB21,... .DBX32.1		

<Chan>.basic.in.retraceSupportActive		**NC → PLC**	**Bool**
沿轮廓恢复加工功能(RESU)：机床退到断点，从该点继续向前移动			
SINUMERIK ONE Basic Program	LBP_Chan*.f32_2		
SINUMERIK 840D sl	DB21,... .DBX32.2		

<Chan>.basic.in.executionFromExternalActive		**NC → PLC**	**Bool**
“从外部执行程序”激活：此时，子程序或程序比如可直接从闪存盘执行			
SINUMERIK ONE Basic Program	LBP_Chan*.E_ExecExtern		
SINUMERIK 840D sl	DB21,... .DBX32.0		

<Chan>.basic.in.blockSearchActive		**NC → PLC**	**Bool**
程序段搜索：数据采集执行中			
SINUMERIK ONE Basic Program	LBP_Chan*.E_BlockSearch		
SINUMERIK 840D sl	DB21,... .DBX33.4		

<Chan>.basic.in.blockSearchActionActive		**NC → PLC**	**Bool**
程序段搜索：动作程序段执行中			
SINUMERIK ONE Basic Program	LBP_Chan*.E_ActBlock		
SINUMERIK 840D sl	DB21,... .DBX32.3		

<Chan>.basic.in.blockSearchLastActionActive		**NC → PLC**	**Bool**
程序段搜索：最后一条动作程序段执行中			
SINUMERIK ONE Basic Program	LBP_Chan*.E_LastActBlock		
SINUMERIK 840D sl	DB21,... .DBX32.6		

<Chan>.basic.in.blockSearchApproachActive	**NC → PLC**	**Bool**
程序段搜索：启动程序段“程序段搜索，带沿轮廓计算”执行中		
SINUMERIK ONE Basic Program	LBP_Chan*.E_BegBlock	
SINUMERIK 840D sl	DB21,... .DBX32.4	

<Chan>.basic.in.blockSearchSeruproActive	**NC → PLC**	**Bool**
程序测试期间的程序段搜索(SERUPRO)：找到目标程序段后，该信号有效，程序状态变为“停止”		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SearchAct	
SINUMERIK 840D sl	DB21,... .DBX318.1	

<Chan>.basic.in.m00m01Active	**NC → PLC**	**Bool**
M00 / M01:“程序停止/有条件停止”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_M01	
SINUMERIK 840D sl	DB21,... .DBX32.5	

<Chan>.basic.in.m00m01AssociatedAuxFuncActive	**NC → PLC**	**Bool**
M00 / M01:对应的辅助功能“程序停止/有条件停止”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_AssM01	
SINUMERIK 840D sl	DB21,... .DBX318.5	

<Chan>.basic.in.m02m30Active	**NC → PLC**	**Bool**
M02 / M30: “程序结束”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_M30	
SINUMERIK 840D sl	DB21,... .DBX33.5	

<Chan>.basic.in.refActive	**NC → PLC**	**Bool**
“回参考点”功能已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_Ref	
SINUMERIK 840D sl	DB21,... .DBX33.0	

<Chan>.basic.in.orientableToolholderActive	**NC → PLC**	**Bool**
摆动式刀柄已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_OrieToolholder	
SINUMERIK 840D sl	DB21,... .DBX33.1	

<Chan>.basic.in.revolutionalFeedrateActive	**NC → PLC**	**Bool**
旋转进给率 (G95)已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_RevFD	
SINUMERIK 840D sl	DB21,... .DBX33.2	

<Chan>.basic.in.handwheelOvrActive	**NC → PLC**	**Bool**
“自动方式中的手轮倍率”生效：可以为程序中的轨迹轴指定附加的进给倍率		
SINUMERIK ONE Basic Program	LBP_Chan*.E_HWOverlay	
SINUMERIK 840D sl	DB21,... .DBX33.3	

<Chan>.basic.in.transformationActive	**NC → PLC**	**Bool**
坐标转换已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_Transform	
SINUMERIK 840D sl	DB21,... .DBX33.6	

<Chan>.basic.in.transformationActiveNumber	**NC → PLC**	**USInt**
坐标转换的编号		
SINUMERIK ONE Basic Program	LBP_Chan*.E_TransformNo	
SINUMERIK 840D sl	DB21,... .DBB388	

<Chan>.basic.in.progTestSelected	**NC → PLC**	**Bool**
程序测试(PRT)已选中		
SINUMERIK ONE Basic Program	LBP_Chan*.E_ProgTest	
SINUMERIK 840D sl	DB21,... .DBX33.7	

<Chan>.basic.in.allAxesStationary	**NC → PLC**	**Bool**
该通道的所有轴都停止，没有后续运动需要执行。		
SINUMERIK ONE Basic Program	LBP_Chan*.E_AxesStop	
SINUMERIK 840D sl	DB21,... .DBX36.3	

<Chan>.basic.in.interruptProcessingActive	**NC → PLC**	**Bool**
中断加工已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_IR_Activ	
SINUMERIK 840D sl	DB21,... .DBX36.4	

<Chan>.basic.in.ncAlarmActive	**NC → PLC**	**Bool**
通道中至少有一条 NC 报警待处理		
SINUMERIK ONE Basic Program	LBP_Chan*.E_NCKalarmChan	
SINUMERIK 840D sl	DB21,... .DBX36.6	

<Chan>.basic.in.ncAlarmWithStopActive	**NC → PLC**	**Bool**
通道中至少有一条导致加工停止的 NC 报警待处理		
SINUMERIK ONE Basic Program	LBP_Chan*.E_NCKalarmStop	
SINUMERIK 840D sl	DB21,... .DBX36.7	

<Chan>.basic.in.ncAlarmWithProgStopActive	**NC → PLC**	**Bool**
通道中至少有一条导致程序停止的 NC 报警待处理（程序执行过程中断）		
SINUMERIK ONE Basic Program	LBP_Chan*.f39_1	
SINUMERIK 840D sl	DB21,... .DBX39.1	

<Chan>.basic.in.clcActive	**NC → PLC**	**Bool**
间隙控制 CLC 生效（编译循环）		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CLC	
SINUMERIK 840D sl	DB21,... .DBX37.3	

<Chan>.basic.in.clcStopAtLowerLimitActive	**NC → PLC**	**Bool**
间隙控制 CLC：轴达到下限位并停止(MD62505 $MC_CLC_SENSOR_LOWER_LIMIT)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CLCStopLL	
SINUMERIK 840D sl	DB21,... .DBX37.4	

<Chan>.basic.in.clcStopAtUpperLimitActive	**NC → PLC**	**Bool**
间隙控制 CLC：轴达到上限位并停止(MD62506 $MC_CLC_SENSOR_UPPER_LIMIT)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CLCStopUL	
SINUMERIK 840D sl	DB21,... .DBX37.5	

<Chan>.basic.in.disableReadInIgnored	**NC → PLC**	**Bool**
“读入禁止”被忽略		
SINUMERIK ONE Basic Program	LBP_Chan*.E_RIEnabIgnored	
SINUMERIK 840D sl	DB21,... .DBX37.6	

<Chan>.basic.in.singleBlockStopIgnored	**NC → PLC**	**Bool**
在单步(SBL)模式总中忽略程序段末尾的停止		
SINUMERIK ONE Basic Program	LBP_Chan*.E_StopBlkEndSBLsuppr	
SINUMERIK 840D sl	DB21,... .DBX37.7	

<Chan>.basic.in.contourHandwheelAssignedNumber	**NC → PLC**	**USInt**
当前分配手轮的编号：1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CHW1	
SINUMERIK 840D sl	DB21,... .DBX37.0..2	

<Chan>.basic.in.contourHandwheelDirInverted	**NC → PLC**	**Bool**
轮廓手轮的旋转方向取反已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_InvCHWDirOfRot	
SINUMERIK 840D sl	DB21,... .DBX39.5	

<Chan>.basic.in.configuredStopFuncActive	**NC → PLC**	**Bool**
自定义暂停已激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CfgStopFuncActive	
SINUMERIK 840D sl	DB21,... .DBX39.6	

<Chan>.basic.in.configuredStopAtBlockEnd	**NC → PLC**	**Bool**
因自定义暂停在程序段末尾停止		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CfgStopAtBlockEnd	
SINUMERIK 840D sl	DB21,... .DBX39.7	

<Chan>.basic.in.singleBlockStopAtBlockEnd	**NC → PLC**	**Bool**
因单步模式在程序段末尾停止		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SblStopAtBlockEnd	
SINUMERIK 840D sl	DB21,... .DBX39.4	

<Chan>.basic.in.externalProgLanguageModeActive	**NC → PLC**	**Bool**
“外部编程语言”运行方式已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_ExtLang	
SINUMERIK 840D sl	DB21,... .DBX317.0	

<Chan>.basic.in.productionTargetReached	**NC → PLC**	**Bool**
已达到目标工件数($AC_ACTUAL_PARTS == $AC_REQUIRED_PARTS)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_WS_Setp	
SINUMERIK 840D sl	DB21,... .DBX317.1	

<Chan>.basic.in.ptpTraversActive	**NC → PLC**	**Bool**
PTP（点到点）移动已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_PTP_Travel	
SINUMERIK 840D sl	DB21,... .DBX317.6	

<Chan>.basic.in.toffonActive	**NC → PLC**	**Bool**
在线刀具长度补偿(TOFFON)：功能已生效(MD19320 TECHNO_FUNCTION_MASK 位 29)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_TOFF	
SINUMERIK 840D sl	DB21,... .DBX318.2	

<Chan>.basic.in.toffonMotionActive	**NC → PLC**	**Bool**
在线刀具长度补偿(TOFFON)：补偿运动已生效(MD21194 TOFF_VELO)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_TOFFmovem	
SINUMERIK 840D sl	DB21,... .DBX318.3	

<Chan>.basic.in.dryRunFeedrateActive	**NC → PLC**	**Bool**
试运行进给速度已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_DRY	
SINUMERIK 840D sl	DB21,... .DBX318.6	

<Chan>.basic.in.overStoreActive	**NC → PLC**	**Bool**
溢出存储已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_Overstore	
SINUMERIK 840D sl	DB21,... .DBX318.7	

<Chan>.basic.in.reposModeChangeAck	**NC → PLC**	**Bool**
应答 REPOS 重新定位方式的切换		
SINUMERIK ONE Basic Program	LBP_Chan*.E_REPOS_EdgeAckn	
SINUMERIK 840D sl	DB21,... .DBX319.0	

<Chan>.basic.in.reposModeSelectedNumber	**NC → PLC**	**USInt**
选中的 REPOS 重新定位： 0 - 无 REPOS 重新定位生效; 1 - 移动到程序段起点(RMBBL); 2 - 移动到程序断点(RMIBL); 3 - 移动到程序段结尾(RMEBL); 4 - 移动到下一个轨迹点 (RMNBL)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_REPOS_PMode0	
SINUMERIK 840D sl	DB21,... .DBX319.1..3	

<Chan>.basic.in.reposDelayActive	**NC → PLC**	**Bool**
REPOS 重新定位方式中的延迟已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_REPOS_DEFERRA	
SINUMERIK 840D sl	DB21,... .DBX319.5	

<Chan>.basic.in.stopConditionActive	**NC → PLC**	**Bool**
程序执行时的停止条件已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_StopCondition	
SINUMERIK 840D sl	DB21,... .DBX377.1	

<Chan>.basic.in.jogRetractActive	**NC → PLC**	**Bool**
JOG 回退：已选中并生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_JogRetract	
SINUMERIK 840D sl	DB21,... .DBX377.4	

<Chan>.basic.in.jogRetractDataAvailable	**NC → PLC**	**Bool**
JOG 回退：回退数据存在，用于刀具方向的移动		
SINUMERIK ONE Basic Program	LBP_Chan*.E_RetractData	
SINUMERIK 840D sl	DB21,... .DBX377.5	

<Chan>.basic.in.circularJogTraversActive	**NC → PLC**	**Bool**
“JOG 圆弧移动”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_JOG_Circles	
SINUMERIK 840D sl	DB21,... .DBX377.6	

<Chan>.basic.in.asupActive	**NC → PLC**	**Bool**
异步子程序有效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_AnyAsup	
SINUMERIK 840D sl	DB21,... .DBX378.0	

<Chan>.basic.in.asupSilentActive	**NC → PLC**	**Bool**	
“静默”异步子程序已生效：程序段因为 DISPLOF 不显示在屏幕上			
SINUMERIK ONE Basic Program	LBP_Chan*.E_SilentAsup		
SINUMERIK 840D sl	DB21,... .DBX378.1		

<Chan>.basic.in.asupStopStateActive	**NC → PLC**	**Bool**	
在完整执行程序之前，异步子程序被控制系统停止			
SINUMERIK ONE Basic Program	LBP_Chan*.E_ASUP_Stop		
SINUMERIK 840D sl	DB21,... .DBX318.0		

<Chan>.basic.in.collisionStopActive	**NC → PLC**	**Bool**	
碰撞检测触发了通道内的运行停止。			
SINUMERIK ONE Basic Program	LBP_Chan*.E_CollCheckStop		
SINUMERIK 840D sl	DB21,... .DBX377.0		

<Chan>.basic.in.protAreasNotGuaranteed	**NC → PLC**	**Bool**	
运动开始后才激活了监控，因此保护区监控无法确保有效（不属于碰撞检测 CA）			
SINUMERIK ONE Basic Program	LBP_Chan*.E_ProtZoneNOK		
SINUMERIK 840D sl	DB21,... .DBX39.0		

<Chan>.basic.in.reducedDynamicsActive	**NC → PLC**	**Bool**	**起始版本 V11**
“降低动态模式”生效			
SINUMERIK ONE Basic Program	LBP_Chan*.E_ReducedDynamicsActive		
SINUMERIK 840D sl	DB21,... .DBX39.3		

<Chan>.basic.out.disableFeed	**PLC → NC**	**Bool**	
激活通道中的进给禁止			
SINUMERIK ONE Basic Program	LBP_Chan*.A_FDdisable		
SINUMERIK 840D sl	DB21,... .DBX6.0		

<Chan>.basic.out.retraceSupportTraversBackwards	**PLC → NC**	**Bool**	
沿轮廓恢复加工功能(RESU)：沿着轮廓恢复加工，请求“向后移动”			
SINUMERIK ONE Basic Program	LBP_Chan*.f0_1		
SINUMERIK 840D sl	DB21,... .DBX0.1		

<Chan>.basic.out.retraceSupportStart		**PLC → NC**	**Bool**
沿轮廓恢复加工功能(RESU)：出现 0 → 1 上升沿：沿轮廓恢复加工，请求“开始恢复加工”			
SINUMERIK ONE Basic Program	LBP_Chan*.f0_2		
SINUMERIK 840D sl	DB21,... .DBX0.2		

<Chan>.basic.out.singleBlock		**PLC → NC**	**Bool**
请求激活“单步处理”（SBL）方式			
SINUMERIK ONE Basic Program	LBP_Chan*.A_SBL		
SINUMERIK 840D sl	DB21,... .DBX0.4		

<Chan>.basic.out.m01		**PLC → NC**	**Bool**
请求激活程序控制“有条件停止”(M01)			
SINUMERIK ONE Basic Program	LBP_Chan*.A_M01		
SINUMERIK 840D sl	DB21,... .DBX0.5		

<Chan>.basic.out.dryRunFeedrate		**PLC → NC**	**Bool**
请求激活程序测试进给率(DRY)			
SINUMERIK ONE Basic Program	LBP_Chan*.A_DRY		
SINUMERIK 840D sl	DB21,... .DBX0.6		

<Chan>.basic.out.refPoint		**PLC → NC**	**Bool**
出现 0 → 1 上升沿：请求“回参考点”功能			
SINUMERIK ONE Basic Program	LBP_Chan*.A_Ref		
SINUMERIK 840D sl	DB21,... .DBX1.0		

<Chan>.basic.out.deactivateClc		**PLC → NC**	**Bool**
间隙控制 CLC：请求关闭功能，类似于程序指令 CLC_GAIN=0.0			
SINUMERIK ONE Basic Program	LBP_Chan*.A_CLC_Stop		
SINUMERIK 840D sl	DB21,... .DBX1.4		

<Chan>.basic.out.clcOvr		**PLC → NC**	**Bool**
间隙控制 CLC：请求使用通道专用的进给倍率			
SINUMERIK ONE Basic Program	LBP_Chan*.A_CLC_OR		
SINUMERIK 840D sl	DB21,... .DBX1.5		

<Chan>.basic.out.blockSearchAckPlcActionDone	**PLC → NC**	**Bool**
程序段搜索：PLC 动作反馈已结束		
SINUMERIK ONE Basic Program	LBP_Chan*.A_PLC_ActCompl	
SINUMERIK 840D sl	DB21,... .DBX1.6	

<Chan>.basic.out.progTest	**PLC → NC**	**Bool**
请求激活程序测试(PRT)		
SINUMERIK ONE Basic Program	LBP_Chan*.A_ProgTest	
SINUMERIK 840D sl	DB21,... .DBX1.7	

<Chan>.basic.out.pathFeedrateOvrFactor	**PLC → NC**	**USInt**
轨迹进给速度倍率可采用二进制或格雷码(MD12020 $MN_OVR_FEED_IS_GRAY_CODE)		
SINUMERIK ONE Basic Program	LBP_Chan*.A_FD_OR	
SINUMERIK 840D sl	DB21,... .DBB4	

<Chan>.basic.out.pathFeedrateOvr	**PLC → NC**	**Bool**
轨迹进给速度倍率生效		
SINUMERIK ONE Basic Program	LBP_Chan*.A_FD_ORA	
SINUMERIK 840D sl	DB21,... .DBX6.7	

<Chan>.basic.out.pathFeedrateLimit	**PLC → NC**	**Bool**
激活刀具轨迹进给速度限制(MD12100 $MN_OVR_FACTOR_LIMIT_BIN)		
SINUMERIK ONE Basic Program	LBP_Chan*.A_F_Limit	
SINUMERIK 840D sl	DB21,... .DBX6.5	

<Chan>.basic.out.pathRapidTraversOvrFactor	**PLC → NC**	**USInt**
轨迹快速移动速度倍率可采用二进制或格雷码(MD12050 $MN_OVR_FACTOR_RAPID_TRA)。		
SINUMERIK ONE Basic Program	LBP_Chan*.A_RT_OR	
SINUMERIK 840D sl	DB21,... .DBB5	

<Chan>.basic.out.pathRapidTraversOvr	**PLC → NC**	**Bool**
轨迹快速移动速度倍率生效		
SINUMERIK ONE Basic Program	LBP_Chan*.A_RT_ORA	
SINUMERIK 840D sl	DB21,... .DBX6.6	

<Chan>.basic.out.disableReadIn		**PLC → NC**	**Bool**
请求“读入禁止”			
SINUMERIK ONE Basic Program	LBP_Chan*.A_RIdisable		
SINUMERIK 840D sl	DB21,... .DBX6.1		

<Chan>.basic.out.deleteDistanceToGo		**PLC → NC**	**Bool**
出现 0 → 1 上升沿：“删除剩余行程”（通道专用剩余行程）已请求			
SINUMERIK ONE Basic Program	LBP_Chan*.A_DeleteDTG		
SINUMERIK 840D sl	DB21,... .DBX6.2		

<Chan>.basic.out.deleteSubProgRepetition		**PLC → NC**	**Bool**
出现 0 → 1 上升沿：删除剩余的子程序重复执行次数			
SINUMERIK ONE Basic Program	LBP_Chan*.A_SP_Clear		
SINUMERIK 840D sl	DB21,... .DBX6.3		

<Chan>.basic.out.abortProgLevel		**PLC → NC**	**Bool**
出现 0 → 1 上升沿：请求中断当前程序级（子程序级、异步子程序级、数据存储器程序级）的执行			
SINUMERIK ONE Basic Program	LBP_Chan*.A_ProgAbort		
SINUMERIK 840D sl	DB21,... .DBX6.4		

<Chan>.basic.out.disableNcStart		**PLC → NC**	**Bool**
请求“NC 启动禁止”			
SINUMERIK ONE Basic Program	LBP_Chan*.A_NCStartDisabl		
SINUMERIK 840D sl	DB21,... .DBX7.0		

<Chan>.basic.out.ncStart		**PLC → NC**	**Bool**
出现 0 → 1 上升沿：请求启动在通道中选择的程序			
SINUMERIK ONE Basic Program	LBP_Chan*.A_NCStart		
SINUMERIK 840D sl	DB21,... .DBX7.1		

<Chan>.basic.out.ncStop		**PLC → NC**	**Bool**
请求“NC 停止”			
SINUMERIK ONE Basic Program	LBP_Chan*.A_NCStop		
SINUMERIK 840D sl	DB21,... .DBX7.3		

<Chan>.basic.out.ncStopBlockEnd	**PLC → NC**	**Bool**
请求在当前程序段结束处执行 NC 停止		
SINUMERIK ONE Basic Program	LBP_Chan*.A_NCStopBlock	
SINUMERIK 840D sl	DB21,... .DBX7.2	

<Chan>.basic.out.ncStopAxesAndSpindles	**PLC → NC**	**Bool**
请求"NC 停止进给轴和主轴"		
SINUMERIK ONE Basic Program	LBP_Chan*.A_NCStopASp	
SINUMERIK 840D sl	DB21,... .DBX7.4	

<Chan>.basic.out.suppressDisableNcStart	**PLC → NC**	**Bool**
请求撤销 NC 启动禁止		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SuppStartLock	
SINUMERIK 840D sl	DB21,... .DBX7.5	

<Chan>.basic.out.configuredStopFunc	**PLC → NC**	**Bool**
激活自定义暂停		
SINUMERIK ONE Basic Program	LBP_Chan*.A_CfgStopFunc	
SINUMERIK 840D sl	DB21,... .DBX7.6	

<Chan>.basic.out.reset	**PLC → NC**	**Bool**
通道复位已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.A_Reset	
SINUMERIK 840D sl	DB21,... .DBX7.7	

<Chan>.basic.out.pathFixedFeedrateSelection	**PLC → NC**	**USInt**
为轨迹轴/几何轴激活固定进给速度 1..4（0 表示不选择）。该接口信号可以在自动运行方式中激活固定进给速度，而不使用程序中的进给速度或参数定义的 JOG 速度。		
SINUMERIK ONE Basic Program	LBP_Chan*.A_FixedFD1	
SINUMERIK 840D sl	DB21,... .DBX29.0..3	

<Chan>.basic.out.ptpTravers	**PLC → NC**	**Bool**
激活 PTP（点到点）移动		
SINUMERIK ONE Basic Program	LBP_Chan*.A_PTP_Travel	
SINUMERIK 840D sl	DB21,... .DBX29.4	

<Chan>.basic.out.handwheelOffsetDrf	**PLC → NC**	**Bool**
激活手轮偏移(DRF)已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.A_DRF	
SINUMERIK 840D sl	DB21,... .DBX0.3	

<Chan>.basic.out.contourHandwheelSelection	**PLC → NC**	**USInt**
分配手轮 1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）		
SINUMERIK ONE Basic Program	LBP_Chan*.A_ContHW1	
SINUMERIK 840D sl	DB21,... .DBX30.0..2	

<Chan>.basic.out.contourHandwheelSim	**PLC → NC**	**Bool**
激活轮廓手轮仿真		
SINUMERIK ONE Basic Program	LBP_Chan*.A_ContHWsim	
SINUMERIK 840D sl	DB21,... .DBX30.3	

<Chan>.basic.out.contourHandwheelInvertDir	**PLC → NC**	**Bool**
为选中的轮廓手轮请求切换旋转方向		
SINUMERIK ONE Basic Program	LBP_Chan*.A_InvCHWDirOfRot	
SINUMERIK 840D sl	DB21,... .DBX31.5	

<Chan>.basic.out.contourHandwheelSimInvertDir	**PLC → NC**	**Bool**
在轮廓手轮仿真中，手轮逆着程序指定方向沿着轮廓移动		
SINUMERIK ONE Basic Program	LBP_Chan*.A_ContHWneg	
SINUMERIK 840D sl	DB21,... .DBX30.4	

<Chan>.basic.out.m01AssociatedAuxFunc	**PLC → NC**	**Bool**
请求激活 M0/M1“关联辅助功能”		
SINUMERIK ONE Basic Program	LBP_Chan*.A_NCKrelatedM01	
SINUMERIK 840D sl	DB21,... .DBX30.5	

<Chan>.basic.out.circularJogTravers	**PLC → NC**	**Bool**
请求“JOG 圆弧移动”(SD42690 到 42694 $SC_JOG_CIRCLE_...)		
SINUMERIK ONE Basic Program	LBP_Chan*.A_JOG_Circles	
SINUMERIK 840D sl	DB21,... .DBX30.6	

<Chan>.basic.out.reposModeSelection	**PLC → NC**	**USInt**
REPOS 重新定位方式： 0 - 无 REPOS 重新定位; 1 - 移动到程序段起点(RMBBL); 2 - 移动到程序断点(RMIBL); 3 - 移动到程序段结尾(RMEBL); 4 - 移动到下一个轨迹点 (RMNBL)		
SINUMERIK ONE Basic Program	LBP_Chan*.A_REPOSPM_0	
SINUMERIK 840D sl	DB21,... .DBX31.0..2	

<Chan>.basic.out.reposModeActivationRequest	**PLC → NC**	**Bool**
请求为当前主处理程序段激活"REPOS"方式		
SINUMERIK ONE Basic Program	LBP_Chan*.A_REPOSMode	
SINUMERIK 840D sl	DB21,... .DBX31.4	

<Chan>.basic.out.enableGoToStart	**PLC → NC**	**Bool**
请求激活 GOTOS（跳转到当前程序开头处）		
SINUMERIK ONE Basic Program	LBP_Chan*.A_ProgJump	
SINUMERIK 840D sl	DB21,... .DBX384.0	

<Chan>.basic.out.geoAxesTraversBcsWcsSelection	**PLC → NC**	**USInt**
为几何轴的手动运行和手轮倍率(DRF)选择坐标系： 0 - 未选择; 1 - BCS; 2 - WCS; 3 - TCS		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CART_JOG_MODE	
SINUMERIK 840D sl	DB21,... .DBB392	

<Chan>.basic.out.enableProtAreas	**PLC → NC**	**Bool**
激活保护区（不属于碰撞检测 CA）		
SINUMERIK ONE Basic Program	LBP_Chan*.A_ProtZone	
SINUMERIK 840D sl	DB21,... .DBX1.1	

<Chan>.basic.out.lockAllSynActs	**PLC → NC**	**Bool**
禁止所有支持的同步动作(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynOff	
SINUMERIK 840D sl	DB21,... .DBX1.2	

<Chan>.basic.out.skipBlockLevel0	**PLC → NC**	**Bool**
请求激活跳转级 0 的零件程序段跳转功能(Skip, SKP)		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SKP0	
SINUMERIK 840D sl	DB21,... .DBX2.0	

<Chan>.basic.out.skipBlockLevel1		**PLC → NC**	**Bool**
请求激活跳转级 1 的零件程序段跳转功能(Skip, SKP)			
SINUMERIK ONE Basic Program	LBP_Chan*.A_SKP1		
SINUMERIK 840D sl	DB21,... .DBX2.1		

<Chan>.basic.out.skipBlockLevel2		**PLC → NC**	**Bool**
请求激活跳转级 2 的零件程序段跳转功能(Skip, SKP)			
SINUMERIK ONE Basic Program	LBP_Chan*.A_SKP2		
SINUMERIK 840D sl	DB21,... .DBX2.2		

<Chan>.basic.out.skipBlockLevel3		**PLC → NC**	**Bool**
请求激活跳转级 3 的零件程序段跳转功能(Skip, SKP)			
SINUMERIK ONE Basic Program	LBP_Chan*.A_SKP3		
SINUMERIK 840D sl	DB21,... .DBX2.3		

<Chan>.basic.out.skipBlockLevel4		**PLC → NC**	**Bool**
请求激活跳转级 4 的零件程序段跳转功能(Skip, SKP)			
SINUMERIK ONE Basic Program	LBP_Chan*.A_SKP4		
SINUMERIK 840D sl	DB21,... .DBX2.4		

<Chan>.basic.out.skipBlockLevel5		**PLC → NC**	**Bool**
请求激活跳转级 5 的零件程序段跳转功能(Skip, SKP)			
SINUMERIK ONE Basic Program	LBP_Chan*.A_SKP5		
SINUMERIK 840D sl	DB21,... .DBX2.5		

<Chan>.basic.out.skipBlockLevel6		**PLC → NC**	**Bool**
请求激活跳转级 6 的零件程序段跳转功能(Skip, SKP)			
SINUMERIK ONE Basic Program	LBP_Chan*.A_SKP6		
SINUMERIK 840D sl	DB21,... .DBX2.6		

<Chan>.basic.out.skipBlockLevel7		**PLC → NC**	**Bool**
请求激活跳转级 7 的零件程序段跳转功能(Skip, SKP)			
SINUMERIK ONE Basic Program	LBP_Chan*.A_SKP7		
SINUMERIK 840D sl	DB21,... .DBX2.7		

<Chan>.basic.out.skipBlockLevel8	**PLC → NC**	**Bool**
请求激活跳转级 8 的零件程序段跳转功能(Skip, SKP)		
SINUMERIK ONE Basic Program LBP_Chan*.A_SKP8		
SINUMERIK 840D sl DB21,... .DBX31.6		

<Chan>.basic.out.skipBlockLevel9	**PLC → NC**	**Bool**
请求激活跳转级 9 的零件程序段跳转功能(Skip, SKP)		
SINUMERIK ONE Basic Program LBP_Chan*.A_SKP9		
SINUMERIK 840D sl DB21,... .DBX31.7		

<Chan>.gCommandGroup.in.activeValues	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program LBP_Chan*.GAct		
SINUMERIK 840D sl DB21,... .DBB208..271		

<Chan>.gCommandGroup.in.activeValues.activeValues[1]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program LBP_Chan*.GAct[1]		
SINUMERIK 840D sl DB21,... .DBB208		

<Chan>.gCommandGroup.in.activeValues.activeValues[2]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program LBP_Chan*.GAct[2]		
SINUMERIK 840D sl DB21,... .DBB209		

<Chan>.gCommandGroup.in.activeValues.activeValues[3]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program LBP_Chan*.GAct[3]		
SINUMERIK 840D sl DB21,... .DBB210		

<Chan>.gCommandGroup.in.activeValues.activeValues[4]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[4]	
SINUMERIK 840D sl	DB21,... .DBB211	

<Chan>.gCommandGroup.in.activeValues.activeValues[5]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[5]	
SINUMERIK 840D sl	DB21,... .DBB212	

<Chan>.gCommandGroup.in.activeValues.activeValues[6]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[6]	
SINUMERIK 840D sl	DB21,... .DBB213	

<Chan>.gCommandGroup.in.activeValues.activeValues[7]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[7]	
SINUMERIK 840D sl	DB21,... .DBB214	

<Chan>.gCommandGroup.in.activeValues.activeValues[8]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[8]	
SINUMERIK 840D sl	DB21,... .DBB215	

<Chan>.gCommandGroup.in.activeValues.activeValues[9]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[9]	
SINUMERIK 840D sl	DB21,... .DBB216	

<Chan>.gCommandGroup.in.activeValues.activeValues[10]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[10]	
SINUMERIK 840D sl	DB21,... .DBB217	

<Chan>.gCommandGroup.in.activeValues.activeValues[11]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[11]	
SINUMERIK 840D sl	DB21,... .DBB218	

<Chan>.gCommandGroup.in.activeValues.activeValues[12]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[12]	
SINUMERIK 840D sl	DB21,... .DBB219	

<Chan>.gCommandGroup.in.activeValues.activeValues[13]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[13]	
SINUMERIK 840D sl	DB21,... .DBB220	

<Chan>.gCommandGroup.in.activeValues.activeValues[14]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[14]	
SINUMERIK 840D sl	DB21,... .DBB221	

<Chan>.gCommandGroup.in.activeValues.activeValues[15]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[15]	
SINUMERIK 840D sl	DB21,... .DBB222	

<Chan>.gCommandGroup.in.activeValues.activeValues[16]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[16]	
SINUMERIK 840D sl	DB21,... .DBB223	

<Chan>.gCommandGroup.in.activeValues.activeValues[17]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[17]	
SINUMERIK 840D sl	DB21,... .DBB224	

<Chan>.gCommandGroup.in.activeValues.activeValues[18]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[18]	
SINUMERIK 840D sl	DB21,... .DBB225	

<Chan>.gCommandGroup.in.activeValues.activeValues[19]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[19]	
SINUMERIK 840D sl	DB21,... .DBB226	

<Chan>.gCommandGroup.in.activeValues.activeValues[20]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[20]	
SINUMERIK 840D sl	DB21,... .DBB227	

<Chan>.gCommandGroup.in.activeValues.activeValues[21]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[21]	
SINUMERIK 840D sl	DB21,... .DBB228	

<Chan>.gCommandGroup.in.activeValues.activeValues[22]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[22]	
SINUMERIK 840D sl	DB21,... .DBB229	

<Chan>.gCommandGroup.in.activeValues.activeValues[23]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[23]	
SINUMERIK 840D sl	DB21,... .DBB230	

<Chan>.gCommandGroup.in.activeValues.activeValues[24]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[24]	
SINUMERIK 840D sl	DB21,... .DBB231	

<Chan>.gCommandGroup.in.activeValues.activeValues[25]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[25]	
SINUMERIK 840D sl	DB21,... .DBB232	

<Chan>.gCommandGroup.in.activeValues.activeValues[26]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[26]	
SINUMERIK 840D sl	DB21,... .DBB233	

<Chan>.gCommandGroup.in.activeValues.activeValues[27]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[27]	
SINUMERIK 840D sl	DB21,... .DBB234	

<Chan>.gCommandGroup.in.activeValues.activeValues[28]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[28]	
SINUMERIK 840D sl	DB21,... .DBB235	

<Chan>.gCommandGroup.in.activeValues.activeValues[29]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[29]	
SINUMERIK 840D sl	DB21,... .DBB236	

<Chan>.gCommandGroup.in.activeValues.activeValues[30]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[30]	
SINUMERIK 840D sl	DB21,... .DBB237	

<Chan>.gCommandGroup.in.activeValues.activeValues[31]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[31]	
SINUMERIK 840D sl	DB21,... .DBB238	

<Chan>.gCommandGroup.in.activeValues.activeValues[32]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[32]	
SINUMERIK 840D sl	DB21,... .DBB239	

<Chan>.gCommandGroup.in.activeValues.activeValues[33]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[33]	
SINUMERIK 840D sl	DB21,... .DBB240	

<Chan>.gCommandGroup.in.activeValues.activeValues[34]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[34]	
SINUMERIK 840D sl	DB21,... .DBB241	

<Chan>.gCommandGroup.in.activeValues.activeValues[35]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[35]	
SINUMERIK 840D sl	DB21,... .DBB242	

<Chan>.gCommandGroup.in.activeValues.activeValues[36]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[36]	
SINUMERIK 840D sl	DB21,... .DBB243	

<Chan>.gCommandGroup.in.activeValues.activeValues[37]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[37]	
SINUMERIK 840D sl	DB21,... .DBB244	

<Chan>.gCommandGroup.in.activeValues.activeValues[38]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[38]	
SINUMERIK 840D sl	DB21,... .DBB245	

<Chan>.gCommandGroup.in.activeValues.activeValues[39]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[39]	
SINUMERIK 840D sl	DB21,... .DBB246	

<Chan>.gCommandGroup.in.activeValues.activeValues[40]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[40]	
SINUMERIK 840D sl	DB21,... .DBB247	

<Chan>.gCommandGroup.in.activeValues.activeValues[41]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[41]	
SINUMERIK 840D sl	DB21,... .DBB248	

<Chan>.gCommandGroup.in.activeValues.activeValues[42]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[42]	
SINUMERIK 840D sl	DB21,... .DBB249	

<Chan>.gCommandGroup.in.activeValues.activeValues[43]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[43]	
SINUMERIK 840D sl	DB21,... .DBB250	

<Chan>.gCommandGroup.in.activeValues.activeValues[44]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[44]	
SINUMERIK 840D sl	DB21,... .DBB251	

<Chan>.gCommandGroup.in.activeValues.activeValues[45]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[45]	
SINUMERIK 840D sl	DB21,... .DBB252	

<Chan>.gCommandGroup.in.activeValues.activeValues[46]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[46]	
SINUMERIK 840D sl	DB21,... .DBB253	

<Chan>.gCommandGroup.in.activeValues.activeValues[47]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[47]	
SINUMERIK 840D sl	DB21,... .DBB254	

<Chan>.gCommandGroup.in.activeValues.activeValues[48]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[48]	
SINUMERIK 840D sl	DB21,... .DBB255	

<Chan>.gCommandGroup.in.activeValues.activeValues[49]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[49]	
SINUMERIK 840D sl	DB21,... .DBB256	

<Chan>.gCommandGroup.in.activeValues.activeValues[50]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[50]	
SINUMERIK 840D sl	DB21,... .DBB257	

<Chan>.gCommandGroup.in.activeValues.activeValues[51]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[51]	
SINUMERIK 840D sl	DB21,... .DBB258	

<Chan>.gCommandGroup.in.activeValues.activeValues[52]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[52]	
SINUMERIK 840D sl	DB21,... .DBB259	

<Chan>.gCommandGroup.in.activeValues.activeValues[53]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[53]	
SINUMERIK 840D sl	DB21,... .DBB260	

<Chan>.gCommandGroup.in.activeValues.activeValues[54]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[54]	
SINUMERIK 840D sl	DB21,... .DBB261	

<Chan>.gCommandGroup.in.activeValues.activeValues[55]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[55]	
SINUMERIK 840D sl	DB21,... .DBB262	

<Chan>.gCommandGroup.in.activeValues.activeValues[56]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[56]	
SINUMERIK 840D sl	DB21,... .DBB263	

<Chan>.gCommandGroup.in.activeValues.activeValues[57]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[57]	
SINUMERIK 840D sl	DB21,... .DBB264	

<Chan>.gCommandGroup.in.activeValues.activeValues[58]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[58]	
SINUMERIK 840D sl	DB21,... .DBB265	

<Chan>.gCommandGroup.in.activeValues.activeValues[59]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[59]	
SINUMERIK 840D sl	DB21,... .DBB266	

<Chan>.gCommandGroup.in.activeValues.activeValues[60]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[60]	
SINUMERIK 840D sl	DB21,... .DBB267	

<Chan>.gCommandGroup.in.activeValues.activeValues[61]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[61]	
SINUMERIK 840D sl	DB21,... .DBB268	

<Chan>.gCommandGroup.in.activeValues.activeValues[62]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[62]	
SINUMERIK 840D sl	DB21,... .DBB269	

<Chan>.gCommandGroup.in.activeValues.activeValues[63]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[63]	
SINUMERIK 840D sl	DB21,... .DBB270	

<Chan>.gCommandGroup.in.activeValues.activeValues[64]	**NC → PLC**	**USInt**
G 指令组 1..64 内当前生效的 G 指令的内部编号		
SINUMERIK ONE Basic Program	LBP_Chan*.GAct[64]	
SINUMERIK 840D sl	DB21,... .DBB271	

<Chan>.gCommandGroup.in.g00Active	**NC → PLC**	**Bool**
G00 快速移动速度已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_G00	
SINUMERIK 840D sl	DB21,... .DBB316	

<Chan>.geoAxis1.in	**NC → PLC**	
SINUMERIK ONE Basic Program	LBP_Chan*.E_Geo[1]	
SINUMERIK 840D sl	DB21,... .DBX40.0..45.7	

<Chan>.geoAxis.in.handwheelAssignedNumber	**NC → PLC**	**USInt**
当前分配手轮的编号：1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）		
SINUMERIK ONE Basic Program	LBP_Chan*.HW1	
SINUMERIK 840D sl	DB21,... .DBX40.0..2	

<Chan>.geoAxis.in.handwheelDirInverted	**NC → PLC**	**Bool**
分给几何轴的手轮的旋转方向取反已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.InvHWDirOfRot	
SINUMERIK 840D sl	DB21,... .DBX43.0	

<Chan>.geoAxis.in.traversMinusRequest	**NC → PLC**	**Bool**
几何轴负向移动已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.TReqMinus	
SINUMERIK 840D sl	DB21,... .DBX40.4	

<Chan>.geoAxis.in.traversPlusRequest	**NC → PLC**	**Bool**
几何轴正向移动已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.TReqPlus	
SINUMERIK 840D sl	DB21,... .DBX40.5	

<Chan>.geoAxis.in.traversMinusCommand	**NC → PLC**	**Bool**
几何轴负向移动指令已生效(MD17900 $MN_VDI_FUNCTION_MASK)		
SINUMERIK ONE Basic Program	LBP_Chan*.TCMinus	
SINUMERIK 840D sl	DB21,... .DBX40.6	

<Chan>.geoAxis.in.traversPlusCommand	**NC → PLC**	**Bool**
几何轴正向移动指令已生效(MD17900 $MN_VDI_FUNCTION_MASK)		
SINUMERIK ONE Basic Program	LBP_Chan*.TCPlus	
SINUMERIK 840D sl	DB21,... .DBX40.7	

<Chan>.geoAxis.in.inc1Active	**NC → PLC**	**Bool**
“收到请求时移动几何轴 1 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC1	
SINUMERIK 840D sl	DB21,... .DBX41.0	

<Chan>.geoAxis.in.inc10Active	**NC → PLC**	**Bool**
“收到请求时移动几何轴 10 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC10	
SINUMERIK 840D sl	DB21,... .DBX41.1	

<Chan>.geoAxis.in.inc100Active	**NC → PLC**	**Bool**
“收到请求时移动几何轴 100 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC100	
SINUMERIK 840D sl	DB21,... .DBX41.2	

<Chan>.geoAxis.in.inc1000Active	**NC → PLC**	**Bool**
“收到请求时移动几何轴 1 000 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC1000	
SINUMERIK 840D sl	DB21,... .DBX41.3	

<Chan>.geoAxis.in.inc10000Active	**NC → PLC**	**Bool**
“收到请求时移动几何轴 10 000 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC10000	
SINUMERIK 840D sl	DB21,... .DBX41.4	

<Chan>.geoAxis.in.incVarActive		**NC → PLC**	**Bool**
“收到请求时移动几何轴轴用户自定义数量的增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INCVar		
SINUMERIK 840D sl	DB21,... .DBX41.5		

<Chan>.geoAxis.in.contManualTraversActive		**NC → PLC**	**Bool**
几何轴连续移动（标准）			
SINUMERIK 840D sl	DB21,... .DBX41.6		

<Chan>.geoAxis1.out		**PLC → NC**	
SINUMERIK ONE Basic Program	LBP_Chan*.A_Geo[1]		
SINUMERIK 840D sl	DB21,... .DBX12.0..15.7		

<Chan>.geoAxis.out.handwheelSelection		**PLC → NC**	**USInt**
分配手轮 1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）			
SINUMERIK ONE Basic Program	LBP_Chan*.HW1		
SINUMERIK 840D sl	DB21,... .DBX12.0..2		

<Chan>.geoAxis.out.handwheelInvertDir		**PLC → NC**	**Bool**
请求取反分给几何轴的手轮的旋转方向			
SINUMERIK ONE Basic Program	LBP_Chan*.InvHWDirOfRot		
SINUMERIK 840D sl	DB21,... .DBX15.0		

<Chan>.geoAxis.out.disableFeed		**PLC → NC**	**Bool**
禁止几何轴的进给速度			
SINUMERIK ONE Basic Program	LBP_Chan*.FDStop		
SINUMERIK 840D sl	DB21,... .DBX12.3		

<Chan>.geoAxis.out.disableTraversKey		**PLC → NC**	**Bool**
禁止几何轴的移动键			
SINUMERIK ONE Basic Program	LBP_Chan*.Disabl		
SINUMERIK 840D sl	DB21,... .DBX12.4		

<Chan>.geoAxis.out.rapidTraversOvr		**PLC → NC**	**Bool**
为几何轴激活快速移动速度			
SINUMERIK ONE Basic Program	LBP_Chan*.RapidTrOR		
SINUMERIK 840D sl	DB21,... .DBX12.5		

<Chan>.geoAxis.out.traversKeyMinus		**PLC → NC**	**Bool**
请求为几何轴激活“负向”键，使轴在负向移动			
SINUMERIK ONE Basic Program	LBP_Chan*.Minus		
SINUMERIK 840D sl	DB21,... .DBX12.6		

<Chan>.geoAxis.out.traversKeyPlus		**PLC → NC**	**Bool**
请求为几何轴激活“正向”键，使轴在正向移动			
SINUMERIK ONE Basic Program	LBP_Chan*.Plus		
SINUMERIK 840D sl	DB21,... .DBX12.7		

<Chan>.geoAxis.out.inc1		**PLC → NC**	**Bool**
激活“收到请求时移动几何轴 1 个增量”			
SINUMERIK ONE Basic Program	LBP_Chan*.INC1		
SINUMERIK 840D sl	DB21,... .DBX13.0		

<Chan>.geoAxis.out.inc10		**PLC → NC**	**Bool**
“收到请求时移动几何轴 10 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC10		
SINUMERIK 840D sl	DB21,... .DBX13.1		

<Chan>.geoAxis.out.inc100		**PLC → NC**	**Bool**
“收到请求时移动几何轴 100 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC100		
SINUMERIK 840D sl	DB21,... .DBX13.2		

<Chan>.geoAxis.out.inc1000		**PLC → NC**	**Bool**
激活“收到请求时移动几何轴 1 000 个增量”			
SINUMERIK ONE Basic Program	LBP_Chan*.INC1000		
SINUMERIK 840D sl	DB21,... .DBX13.3		

<Chan>.geoAxis.out.inc10000		**PLC → NC**	**Bool**
激活“收到请求时移动几何轴 10 000 个增量”			
SINUMERIK ONE Basic Program	LBP_Chan*.INC10000		
SINUMERIK 840D sl	DB21,... .DBX13.4		

<Chan>.geoAxis.out.incVar		**PLC → NC**	**Bool**
激活“收到请求时移动几何轴用户自定义数量的增量”			
SINUMERIK ONE Basic Program	LBP_Chan*.INCVar		
SINUMERIK 840D sl	DB21,... .DBX13.5		

<Chan>.geoAxis.out.contManualTravers		**PLC → NC**	**Bool**
激活几何轴的连续移动（标准）			
SINUMERIK ONE Basic Program	LBP_Chan*.ContManTravel		
SINUMERIK 840D sl	DB21,... .DBX13.6		

<Chan>.geoAxis2.in		**NC → PLC**	
SINUMERIK ONE Basic Program	LBP_Chan*.E_Geo[2]		
SINUMERIK 840D sl	DB21,... .DBX46.0..51.7		

<Chan>.geoAxis2.in.handwheelAssignedNumber		**NC → PLC**	**USInt**
当前分配手轮的编号：1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）			
SINUMERIK ONE Basic Program	LBP_Chan*.HW1		
SINUMERIK 840D sl	DB21,... .DBX46.0..2		

<Chan>.geoAxis2.in.handwheelDirInverted		**NC → PLC**	**Bool**
分给几何轴的手轮的旋转方向取反已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.InvHWDirOfRot		
SINUMERIK 840D sl	DB21,... .DBX49.0		

<Chan>.geoAxis2.in.traversMinusRequest		**NC → PLC**	**Bool**
几何轴负向移动已请求			
SINUMERIK ONE Basic Program	LBP_Chan*.TReqMinus		
SINUMERIK 840D sl	DB21,... .DBX46.4		

<Chan>.geoAxis2.in.traversPlusRequest	**NC → PLC**	**Bool**
几何轴正向移动已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.TReqPlus	
SINUMERIK 840D sl	DB21,... .DBX46.5	

<Chan>.geoAxis2.in.traversMinusCommand	**NC → PLC**	**Bool**
几何轴负向移动指令已生效(MD17900 $MN_VDI_FUNCTION_MASK)		
SINUMERIK ONE Basic Program	LBP_Chan*.TCMinus	
SINUMERIK 840D sl	DB21,... .DBX46.6	

<Chan>.geoAxis2.in.traversPlusCommand	**NC → PLC**	**Bool**
几何轴正向移动指令已生效(MD17900 $MN_VDI_FUNCTION_MASK)		
SINUMERIK ONE Basic Program	LBP_Chan*.TCPlus	
SINUMERIK 840D sl	DB21,... .DBX46.7	

<Chan>.geoAxis2.in.inc1Active	**NC → PLC**	**Bool**
“收到请求时移动几何轴 1 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC1	
SINUMERIK 840D sl	DB21,... .DBX47.0	

<Chan>.geoAxis2.in.inc10Active	**NC → PLC**	**Bool**
“收到请求时移动几何轴 10 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC10	
SINUMERIK 840D sl	DB21,... .DBX47.1	

<Chan>.geoAxis2.in.inc100Active	**NC → PLC**	**Bool**
“收到请求时移动几何轴 100 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC100	
SINUMERIK 840D sl	DB21,... .DBX47.2	

<Chan>.geoAxis2.in.inc1000Active	**NC → PLC**	**Bool**
“收到请求时移动几何轴 1 000 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC1000	
SINUMERIK 840D sl	DB21,... .DBX47.3	

<Chan>.geoAxis2.in.inc10000Active	**NC → PLC**	**Bool**
“收到请求时移动几何轴 10 000 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC10000	
SINUMERIK 840D sl	DB21,... .DBX47.4	

<Chan>.geoAxis2.in.incVarActive	**NC → PLC**	**Bool**
“收到请求时移动几何轴轴用户自定义数量的增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INCVar	
SINUMERIK 840D sl	DB21,... .DBX47.5	

<Chan>.geoAxis2.in.contManualTraversActive	**NC → PLC**	**Bool**
几何轴连续移动（标准）		
SINUMERIK 840D sl	DB21,... .DBX47.6	

<Chan>.geoAxis2.out	**PLC → NC**	
SINUMERIK ONE Basic Program	LBP_Chan*.A_Geo[2]	
SINUMERIK 840D sl	DB21,... .DBX16.0..19.7	

<Chan>.geoAxis2.out.handwheelSelection	**PLC → NC**	**USInt**
分配手轮 1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）		
SINUMERIK ONE Basic Program	LBP_Chan*.HW1	
SINUMERIK 840D sl	DB21,... .DBX16.0..2	

<Chan>.geoAxis2.out.handwheelInvertDir	**PLC → NC**	**Bool**
请求取反分给几何轴的手轮的旋转方向		
SINUMERIK ONE Basic Program	LBP_Chan*.InvHWDirOfRot	
SINUMERIK 840D sl	DB21,... .DBX19.0	

<Chan>.geoAxis2.out.disableFeed	**PLC → NC**	**Bool**
禁止几何轴的进给速度		
SINUMERIK ONE Basic Program	LBP_Chan*.FDStop	
SINUMERIK 840D sl	DB21,... .DBX16.3	

<Chan>.geoAxis2.out.disableTraversKey		**PLC → NC**	**Bool**
禁止几何轴的移动键			
SINUMERIK ONE Basic Program	LBP_Chan*.Disabl		
SINUMERIK 840D sl	DB21,... .DBX16.4		

<Chan>.geoAxis2.out.rapidTraversOvr		**PLC → NC**	**Bool**
为几何轴激活快速移动速度			
SINUMERIK ONE Basic Program	LBP_Chan*.RapidTrOR		
SINUMERIK 840D sl	DB21,... .DBX16.5		

<Chan>.geoAxis2.out.traversKeyMinus		**PLC → NC**	**Bool**
请求为几何轴激活“负向”键，使轴在负向移动			
SINUMERIK ONE Basic Program	LBP_Chan*.Minus		
SINUMERIK 840D sl	DB21,... .DBX16.6		

<Chan>.geoAxis2.out.traversKeyPlus		**PLC → NC**	**Bool**
请求为几何轴激活“正向”键，使轴在正向移动			
SINUMERIK ONE Basic Program	LBP_Chan*.Plus		
SINUMERIK 840D sl	DB21,... .DBX16.7		

<Chan>.geoAxis2.out.inc1		**PLC → NC**	**Bool**
激活“收到请求时移动几何轴 1 个增量”			
SINUMERIK ONE Basic Program	LBP_Chan*.INC1		
SINUMERIK 840D sl	DB21,... .DBX17.0		

<Chan>.geoAxis2.out.inc10		**PLC → NC**	**Bool**
“收到请求时移动几何轴 10 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC10		
SINUMERIK 840D sl	DB21,... .DBX17.1		

<Chan>.geoAxis2.out.inc100		**PLC → NC**	**Bool**
“收到请求时移动几何轴 100 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC100		
SINUMERIK 840D sl	DB21,... .DBX17.2		

<Chan>.geoAxis2.out.inc1000		**PLC → NC**	**Bool**
激活“收到请求时移动几何轴 1 000 个增量”			
SINUMERIK ONE Basic Program	LBP_Chan*.INC1000		
SINUMERIK 840D sl	DB21,... .DBX17.3		

<Chan>.geoAxis2.out.inc10000		**PLC → NC**	**Bool**
激活“收到请求时移动几何轴 10 000 个增量”			
SINUMERIK ONE Basic Program	LBP_Chan*.INC10000		
SINUMERIK 840D sl	DB21,... .DBX17.4		

<Chan>.geoAxis2.out.incVar		**PLC → NC**	**Bool**
激活“收到请求时移动几何轴用户自定义数量的增量”			
SINUMERIK ONE Basic Program	LBP_Chan*.INCVar		
SINUMERIK 840D sl	DB21,... .DBX17.5		

<Chan>.geoAxis2.out.contManualTravers		**PLC → NC**	**Bool**
激活几何轴的连续移动（标准）			
SINUMERIK ONE Basic Program	LBP_Chan*.ContManTravel		
SINUMERIK 840D sl	DB21,... .DBX17.6		

<Chan>.geoAxis3.in		**NC → PLC**	
SINUMERIK ONE Basic Program	LBP_Chan*.E_Geo[3]		
SINUMERIK 840D sl	DB21,... .DBX52.0..56.7		

<Chan>.geoAxis3.in.handwheelAssignedNumber		**NC → PLC**	**USInt**
当前分配手轮的编号：1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）			
SINUMERIK ONE Basic Program	LBP_Chan*.HW1		
SINUMERIK 840D sl	DB21,... .DBX52.0..2		

<Chan>.geoAxis3.in.handwheelDirInverted		**NC → PLC**	**Bool**
分给几何轴的手轮的旋转方向取反已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.InvHWDirOfRot		
SINUMERIK 840D sl	DB21,... .DBX55.0		

<Chan>.geoAxis3.in.traversMinusRequest		**NC → PLC**	**Bool**
几何轴负向移动已请求			
SINUMERIK ONE Basic Program	LBP_Chan*.TReqMinus		
SINUMERIK 840D sl	DB21,... .DBX52.4		

<Chan>.geoAxis3.in.traversPlusRequest		**NC → PLC**	**Bool**
几何轴正向移动已请求			
SINUMERIK ONE Basic Program	LBP_Chan*.TReqPlus		
SINUMERIK 840D sl	DB21,... .DBX52.5		

<Chan>.geoAxis3.in.traversMinusCommand		**NC → PLC**	**Bool**
几何轴负向移动指令已生效(MD17900 $MN_VDI_FUNCTION_MASK)			
SINUMERIK ONE Basic Program	LBP_Chan*.TCMinus		
SINUMERIK 840D sl	DB21,... .DBX52.6		

<Chan>.geoAxis3.in.traversPlusCommand		**NC → PLC**	**Bool**
几何轴正向移动指令已生效(MD17900 $MN_VDI_FUNCTION_MASK)			
SINUMERIK ONE Basic Program	LBP_Chan*.TCPlus		
SINUMERIK 840D sl	DB21,... .DBX52.7		

<Chan>.geoAxis3.in.inc1Active		**NC → PLC**	**Bool**
“收到请求时移动几何轴 1 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC1		
SINUMERIK 840D sl	DB21,... .DBX53.0		

<Chan>.geoAxis3.in.inc10Active		**NC → PLC**	**Bool**
“收到请求时移动几何轴 10 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC10		
SINUMERIK 840D sl	DB21,... .DBX53.1		

<Chan>.geoAxis3.in.inc100Active		**NC → PLC**	**Bool**
“收到请求时移动几何轴 100 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC100		
SINUMERIK 840D sl	DB21,... .DBX53.2		

<Chan>.geoAxis3.in.inc1000Active	**NC → PLC**	**Bool**
“收到请求时移动几何轴 1 000 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC1000	
SINUMERIK 840D sl	DB21,... .DBX53.3	

<Chan>.geoAxis3.in.inc10000Active	**NC → PLC**	**Bool**
“收到请求时移动几何轴 10 000 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC10000	
SINUMERIK 840D sl	DB21,... .DBX53.4	

<Chan>.geoAxis3.in.incVarActive	**NC → PLC**	**Bool**
“收到请求时移动几何轴轴用户自定义数量的增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INCVar	
SINUMERIK 840D sl	DB21,... .DBX53.5	

<Chan>.geoAxis3.in.contManualTraversActive	**NC → PLC**	**Bool**
几何轴连续移动（标准）		
SINUMERIK 840D sl	DB21,... .DBX53.6	

<Chan>.geoAxis3.out	**PLC → NC**	
SINUMERIK ONE Basic Program	LBP_Chan*.A_Geo[3]	
SINUMERIK 840D sl	DB21,... .DBX20.0..23.7	

<Chan>.geoAxis3.out.handwheelSelection	**PLC → NC**	**USInt**
分配手轮 1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）		
SINUMERIK ONE Basic Program	LBP_Chan*.HW1	
SINUMERIK 840D sl	DB21,... .DBX20.0..2	

<Chan>.geoAxis3.out.handwheelInvertDir	**PLC → NC**	**Bool**
请求取反分给几何轴的手轮的旋转方向		
SINUMERIK ONE Basic Program	LBP_Chan*.InvHWDirOfRot	
SINUMERIK 840D sl	DB21,... .DBX23.0	

<Chan>.geoAxis3.out.disableFeed		**PLC → NC**	**Bool**
禁止几何轴的进给速度			
SINUMERIK ONE Basic Program	LBP_Chan*.FDStop		
SINUMERIK 840D sl	DB21,... .DBX20.3		

<Chan>.geoAxis3.out.disableTraversKey		**PLC → NC**	**Bool**
禁止几何轴的移动键			
SINUMERIK ONE Basic Program	LBP_Chan*.Disabl		
SINUMERIK 840D sl	DB21,... .DBX20.4		

<Chan>.geoAxis3.out.rapidTraversOvr		**PLC → NC**	**Bool**
为几何轴激活快速移动速度			
SINUMERIK ONE Basic Program	LBP_Chan*.RapidTrOR		
SINUMERIK 840D sl	DB21,... .DBX20.5		

<Chan>.geoAxis3.out.traversKeyMinus		**PLC → NC**	**Bool**
请求为几何轴激活“负向”键，使轴在负向移动			
SINUMERIK ONE Basic Program	LBP_Chan*.Minus		
SINUMERIK 840D sl	DB21,... .DBX20.6		

<Chan>.geoAxis3.out.traversKeyPlus		**PLC → NC**	**Bool**
请求为几何轴激活“正向”键，使轴在正向移动			
SINUMERIK ONE Basic Program	LBP_Chan*.Plus		
SINUMERIK 840D sl	DB21,... .DBX20.7		

<Chan>.geoAxis3.out.inc1		**PLC → NC**	**Bool**
激活“收到请求时移动几何轴 1 个增量”			
SINUMERIK ONE Basic Program	LBP_Chan*.INC1		
SINUMERIK 840D sl	DB21,... .DBX21.0		

<Chan>.geoAxis3.out.inc10		**PLC → NC**	**Bool**
“收到请求时移动几何轴 10 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC10		
SINUMERIK 840D sl	DB21,... .DBX21.1		

<Chan>.geoAxis3.out.inc100		**PLC → NC**	**Bool**
“收到请求时移动几何轴 100 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC100		
SINUMERIK 840D sl	DB21,... .DBX21.2		

<Chan>.geoAxis3.out.inc1000		**PLC → NC**	**Bool**
激活“收到请求时移动几何轴 1 000 个增量”			
SINUMERIK ONE Basic Program	LBP_Chan*.INC1000		
SINUMERIK 840D sl	DB21,... .DBX21.3		

<Chan>.geoAxis3.out.inc10000		**PLC → NC**	**Bool**
激活“收到请求时移动几何轴 10 000 个增量”			
SINUMERIK ONE Basic Program	LBP_Chan*.INC10000		
SINUMERIK 840D sl	DB21,... .DBX21.4		

<Chan>.geoAxis3.out.incVar		**PLC → NC**	**Bool**
激活“收到请求时移动几何轴用户自定义数量的增量”			
SINUMERIK ONE Basic Program	LBP_Chan*.INCVar		
SINUMERIK 840D sl	DB21,... .DBX21.5		

<Chan>.geoAxis3.out.contManualTravers		**PLC → NC**	**Bool**
激活几何轴的连续移动（标准）			
SINUMERIK ONE Basic Program	LBP_Chan*.ContManTravel		
SINUMERIK 840D sl	DB21,... .DBX21.6		

<Chan>.orientationAxis1.in		**NC → PLC**	
SINUMERIK ONE Basic Program	LBP_Chan*.E_Ori[1]		
SINUMERIK 840D sl	DB21,... .DBX332.0..335.7		

<Chan>.orientationAxis.in.handwheelAssignedNumber		**NC → PLC**	**USInt**
当前分配手轮的编号：1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）			
SINUMERIK ONE Basic Program	LBP_Chan*.HW1		
SINUMERIK 840D sl	DB21,... .DBX332.0..2		

<Chan>.orientationAxis.in.handwheelDirInverted	**NC → PLC**	**Bool**
分给摆动轴的手轮的旋转方向取反已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.InvHWDirOfRot	
SINUMERIK 840D sl	DB21,... .DBX335.0	

<Chan>.orientationAxis.in.traversMinusRequest	**NC → PLC**	**Bool**
摆动轴负向移动已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.TReqMinus	
SINUMERIK 840D sl	DB21,... .DBX332.4	

<Chan>.orientationAxis.in.traversPlusRequest	**NC → PLC**	**Bool**
摆动轴正向移动已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.TReqPlus	
SINUMERIK 840D sl	DB21,... .DBX332.5	

<Chan>.orientationAxis.in.traversMinusCommand	**NC → PLC**	**Bool**
摆动轴负向移动指令已生效(MD17900 $MN_VDI_FUNCTION_MASK)		
SINUMERIK ONE Basic Program	LBP_Chan*.TCMinus	
SINUMERIK 840D sl	DB21,... .DBX332.6	

<Chan>.orientationAxis.in.traversPlusCommand	**NC → PLC**	**Bool**
摆动轴正向移动指令已生效(MD17900 $MN_VDI_FUNCTION_MASK)		
SINUMERIK ONE Basic Program	LBP_Chan*.TCPlus	
SINUMERIK 840D sl	DB21,... .DBX332.7	

<Chan>.orientationAxis.in.inc1Active	**NC → PLC**	**Bool**
“收到请求时移动摆动轴 1 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC1	
SINUMERIK 840D sl	DB21,... .DBX333.0	

<Chan>.orientationAxis.in.inc10Active	**NC → PLC**	**Bool**
“收到请求时移动摆动轴 10 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC10	
SINUMERIK 840D sl	DB21,... .DBX333.1	

<Chan>.orientationAxis.in.inc100Active	**NC → PLC**	**Bool**
“收到请求时移动摆动轴 100 个增量”已生效		
SINUMERIK ONE Basic Program LBP_Chan*.INC100		
SINUMERIK 840D sl DB21,... .DBX333.2		

<Chan>.orientationAxis.in.inc1000Active	**NC → PLC**	**Bool**
“收到请求时移动摆动轴 1 000 个增量”已生效		
SINUMERIK ONE Basic Program LBP_Chan*.INC1000		
SINUMERIK 840D sl DB21,... .DBX333.3		

<Chan>.orientationAxis.in.inc10000Active	**NC → PLC**	**Bool**
“收到请求时移动摆动轴 10 000 个增量”已生效		
SINUMERIK ONE Basic Program LBP_Chan*.INC10000		
SINUMERIK 840D sl DB21,... .DBX333.4		

<Chan>.orientationAxis.in.incVarActive	**NC → PLC**	**Bool**
“收到请求时移动摆动轴用户自定义数量的增量”已生效		
SINUMERIK ONE Basic Program LBP_Chan*.INCVar		
SINUMERIK 840D sl DB21,... .DBX333.5		

<Chan>.orientationAxis.in.contManualTraversActive	**NC → PLC**	**Bool**
连续移动摆动轴（标准）		
SINUMERIK 840D sl DB21,... .DBX333.6		

<Chan>.orientationAxis1.out	**PLC → NC**	
SINUMERIK ONE Basic Program LBP_Chan*.A_Ori[1]		
SINUMERIK 840D sl DB21,... .DBX320.0..323.7		

<Chan>.orientationAxis.out.handwheelSelection	**PLC → NC**	**USInt**
分配手轮 1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）		
SINUMERIK ONE Basic Program LBP_Chan*.HW1		
SINUMERIK 840D sl DB21,... .DBX320.0..2		

<Chan>.orientationAxis.out.handwheelInvertDir	**PLC → NC**	**Bool**
请求取反分给摆动轴的手轮的旋转方向		
SINUMERIK ONE Basic Program	LBP_Chan*.InvHWDirOfRot	
SINUMERIK 840D sl	DB21,... .DBX323.0	

<Chan>.orientationAxis.out.disableFeed	**PLC → NC**	**Bool**
为摆动轴禁用进给速度		
SINUMERIK ONE Basic Program	LBP_Chan*.FDStop	
SINUMERIK 840D sl	DB21,... .DBX320.3	

<Chan>.orientationAxis.out.disableTraversKey	**PLC → NC**	**Bool**
为摆动轴禁用移动键		
SINUMERIK ONE Basic Program	LBP_Chan*.Disabl	
SINUMERIK 840D sl	DB21,... .DBX320.4	

<Chan>.orientationAxis.out.rapidTraversOvr	**PLC → NC**	**Bool**
为摆动轴激活快速移动速度(MD32010)		
SINUMERIK ONE Basic Program	LBP_Chan*.RapidTrOR	
SINUMERIK 840D sl	DB21,... .DBX320.5	

<Chan>.orientationAxis.out.traversKeyMinus	**PLC → NC**	**Bool**
请求为摆动轴激活“负向”键，使轴在负向移动		
SINUMERIK ONE Basic Program	LBP_Chan*.Minus	
SINUMERIK 840D sl	DB21,... .DBX320.6	

<Chan>.orientationAxis.out.traversKeyPlus	**PLC → NC**	**Bool**
请求为摆动轴激活“正向”键，使轴在正向移动		
SINUMERIK ONE Basic Program	LBP_Chan*.Plus	
SINUMERIK 840D sl	DB21,... .DBX320.7	

<Chan>.orientationAxis.out.inc1	**PLC → NC**	**Bool**
激活“收到请求时移动摆动轴 1 个增量”		
SINUMERIK ONE Basic Program	LBP_Chan*.INC1	
SINUMERIK 840D sl	DB21,... .DBX321.0	

<Chan>.orientationAxis.out.inc10		**PLC → NC**	**Bool**
“收到请求时移动摆动轴 10 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC10		
SINUMERIK 840D sl	DB21,... .DBX321.1		

<Chan>.orientationAxis.out.inc100		**PLC → NC**	**Bool**
“收到请求时移动摆动轴 100 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC100		
SINUMERIK 840D sl	DB21,... .DBX321.2		

<Chan>.orientationAxis.out.inc1000		**PLC → NC**	**Bool**
“收到请求时移动摆动轴 1 000 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC1000		
SINUMERIK 840D sl	DB21,... .DBX321.3		

<Chan>.orientationAxis.out.inc10000		**PLC → NC**	**Bool**
“收到请求时移动摆动轴 10 000 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC10000		
SINUMERIK 840D sl	DB21,... .DBX321.4		

<Chan>.orientationAxis.out.incVar		**PLC → NC**	**Bool**
激活“收到请求时移动摆动轴用户自定义数量的增量”			
SINUMERIK ONE Basic Program	LBP_Chan*.INCVar		
SINUMERIK 840D sl	DB21,... .DBX321.5		

<Chan>.orientationAxis.out.contManualTravers		**PLC → NC**	**Bool**
激活摆动轴的连续移动（标准）			
SINUMERIK 840D sl	DB21,... .DBX321.6		

<Chan>.orientationAxis2.in		**NC → PLC**	
SINUMERIK ONE Basic Program	LBP_Chan*.E_Ori[2]		
SINUMERIK 840D sl	DB21,... .DBX336.0..339.7		

<Chan>.orientationAxis2.in.handwheelAssignedNumber	**NC → PLC**	**USInt**
当前分配手轮的编号：1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）		
SINUMERIK ONE Basic Program	LBP_Chan*.HW1	
SINUMERIK 840D sl	DB21,... .DBX336.0..2	

<Chan>.orientationAxis2.in.handwheelDirInverted	**NC → PLC**	**Bool**
分给摆动轴的手轮的旋转方向取反已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.InvHWDirOfRot	
SINUMERIK 840D sl	DB21,... .DBX339.0	

<Chan>.orientationAxis2.in.traversMinusRequest	**NC → PLC**	**Bool**
摆动轴负向移动已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.TReqMinus	
SINUMERIK 840D sl	DB21,... .DBX336.4	

<Chan>.orientationAxis2.in.traversPlusRequest	**NC → PLC**	**Bool**
摆动轴正向移动已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.TReqPlus	
SINUMERIK 840D sl	DB21,... .DBX336.5	

<Chan>.orientationAxis2.in.traversMinusCommand	**NC → PLC**	**Bool**
摆动轴负向移动指令已生效(MD17900 $MN_VDI_FUNCTION_MASK)		
SINUMERIK ONE Basic Program	LBP_Chan*.TCMinus	
SINUMERIK 840D sl	DB21,... .DBX336.6	

<Chan>.orientationAxis2.in.traversPlusCommand	**NC → PLC**	**Bool**
摆动轴正向移动指令已生效(MD17900 $MN_VDI_FUNCTION_MASK)		
SINUMERIK ONE Basic Program	LBP_Chan*.TCPlus	
SINUMERIK 840D sl	DB21,... .DBX336.7	

<Chan>.orientationAxis2.in.inc1Active	**NC → PLC**	**Bool**
“收到请求时移动摆动轴 1 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC1	
SINUMERIK 840D sl	DB21,... .DBX337.0	

<Chan>.orientationAxis2.in.inc10Active	**NC → PLC**	**Bool**
“收到请求时移动摆动轴 10 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC10	
SINUMERIK 840D sl	DB21,... .DBX337.1	

<Chan>.orientationAxis2.in.inc100Active	**NC → PLC**	**Bool**
“收到请求时移动摆动轴 100 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC100	
SINUMERIK 840D sl	DB21,... .DBX337.2	

<Chan>.orientationAxis2.in.inc1000Active	**NC → PLC**	**Bool**
“收到请求时移动摆动轴 1 000 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC1000	
SINUMERIK 840D sl	DB21,... .DBX337.3	

<Chan>.orientationAxis2.in.inc10000Active	**NC → PLC**	**Bool**
“收到请求时移动摆动轴 10 000 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC10000	
SINUMERIK 840D sl	DB21,... .DBX337.4	

<Chan>.orientationAxis2.in.incVarActive	**NC → PLC**	**Bool**
“收到请求时移动摆动轴用户自定义数量的增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INCVar	
SINUMERIK 840D sl	DB21,... .DBX337.5	

<Chan>.orientationAxis2.in.contManualTraversActive	**NC → PLC**	**Bool**
连续移动摆动轴（标准）		
SINUMERIK 840D sl	DB21,... .DBX337.6	

<Chan>.orientationAxis2.out	**PLC → NC**	
SINUMERIK ONE Basic Program	LBP_Chan*.A_Ori[2]	
SINUMERIK 840D sl	DB21,... .DBX324.0..327.7	

<Chan>.orientationAxis2.out.handwheelSelection	**PLC → NC**	**USInt**
分配手轮 1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）		
SINUMERIK ONE Basic Program	LBP_Chan*.HW1	
SINUMERIK 840D sl	DB21,... .DBX324.0..2	

<Chan>.orientationAxis2.out.handwheelInvertDir	**PLC → NC**	**Bool**
请求取反分给摆动轴的手轮的旋转方向		
SINUMERIK ONE Basic Program	LBP_Chan*.InvHWDirOfRot	
SINUMERIK 840D sl	DB21,... .DBX327.0	

<Chan>.orientationAxis2.out.disableFeed	**PLC → NC**	**Bool**
为摆动轴禁用进给速度		
SINUMERIK ONE Basic Program	LBP_Chan*.FDStop	
SINUMERIK 840D sl	DB21,... .DBX324.3	

<Chan>.orientationAxis2.out.disableTraversKey	**PLC → NC**	**Bool**
为摆动轴禁用移动键		
SINUMERIK ONE Basic Program	LBP_Chan*.Disabl	
SINUMERIK 840D sl	DB21,... .DBX324.4	

<Chan>.orientationAxis2.out.rapidTraversOvr	**PLC → NC**	**Bool**
为摆动轴激活快速移动速度(MD32010)		
SINUMERIK ONE Basic Program	LBP_Chan*.RapidTrOR	
SINUMERIK 840D sl	DB21,... .DBX324.5	

<Chan>.orientationAxis2.out.traversKeyMinus	**PLC → NC**	**Bool**
请求为摆动轴激活“负向”键，使轴在负向移动		
SINUMERIK ONE Basic Program	LBP_Chan*.Minus	
SINUMERIK 840D sl	DB21,... .DBX324.6	

<Chan>.orientationAxis2.out.traversKeyPlus	**PLC → NC**	**Bool**
请求为摆动轴激活“正向”键，使轴在正向移动		
SINUMERIK ONE Basic Program	LBP_Chan*.Plus	
SINUMERIK 840D sl	DB21,... .DBX324.7	

<Chan>.orientationAxis2.out.inc1		**PLC → NC**	**Bool**
激活“收到请求时移动摆动轴 1 个增量”			
SINUMERIK ONE Basic Program	LBP_Chan*.INC1		
SINUMERIK 840D sl	DB21,... .DBX325.0		

<Chan>.orientationAxis2.out.inc10		**PLC → NC**	**Bool**
“收到请求时移动摆动轴 10 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC10		
SINUMERIK 840D sl	DB21,... .DBX325.1		

<Chan>.orientationAxis2.out.inc100		**PLC → NC**	**Bool**
“收到请求时移动摆动轴 100 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC100		
SINUMERIK 840D sl	DB21,... .DBX325.2		

<Chan>.orientationAxis2.out.inc1000		**PLC → NC**	**Bool**
“收到请求时移动摆动轴 1 000 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC1000		
SINUMERIK 840D sl	DB21,... .DBX325.3		

<Chan>.orientationAxis2.out.inc10000		**PLC → NC**	**Bool**
“收到请求时移动摆动轴 10 000 个增量”已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.INC10000		
SINUMERIK 840D sl	DB21,... .DBX325.4		

<Chan>.orientationAxis2.out.incVar		**PLC → NC**	**Bool**
激活“收到请求时移动摆动轴用户自定义数量的增量”			
SINUMERIK ONE Basic Program	LBP_Chan*.INCVar		
SINUMERIK 840D sl	DB21,... .DBX325.5		

<Chan>.orientationAxis2.out.contManualTravers		**PLC → NC**	**Bool**
激活摆动轴的连续移动（标准）			
SINUMERIK 840D sl	DB21,... .DBX325.6		

<Chan>.orientationAxis3.in	**NC → PLC**
SINUMERIK ONE Basic Program	LBP_Chan*.E_Ori[3]
SINUMERIK 840D sl	DB21,... .DBX340.0..343.7

<Chan>.orientationAxis3.in.handwheelAssignedNumber	**NC → PLC**	**USInt**
当前分配手轮的编号：1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）		
SINUMERIK ONE Basic Program	LBP_Chan*.HW1	
SINUMERIK 840D sl	DB21,... .DBX340.0..2	

<Chan>.orientationAxis3.in.handwheelDirInverted	**NC → PLC**	**Bool**
分给摆动轴的手轮的旋转方向取反已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.InvHWDirOfRot	
SINUMERIK 840D sl	DB21,... .DBX343.0	

<Chan>.orientationAxis3.in.traversMinusRequest	**NC → PLC**	**Bool**
摆动轴负向移动已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.TReqMinus	
SINUMERIK 840D sl	DB21,... .DBX340.4	

<Chan>.orientationAxis3.in.traversPlusRequest	**NC → PLC**	**Bool**
摆动轴正向移动已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.TReqPlus	
SINUMERIK 840D sl	DB21,... .DBX340.5	

<Chan>.orientationAxis3.in.traversMinusCommand	**NC → PLC**	**Bool**
摆动轴负向移动指令已生效(MD17900 $MN_VDI_FUNCTION_MASK)		
SINUMERIK ONE Basic Program	LBP_Chan*.TCMinus	
SINUMERIK 840D sl	DB21,... .DBX340.6	

<Chan>.orientationAxis3.in.traversPlusCommand	**NC → PLC**	**Bool**
摆动轴正向移动指令已生效(MD17900 $MN_VDI_FUNCTION_MASK)		
SINUMERIK ONE Basic Program	LBP_Chan*.TCPlus	
SINUMERIK 840D sl	DB21,... .DBX340.7	

<Chan>.orientationAxis3.in.inc1Active	**NC → PLC**	**Bool**
“收到请求时移动摆动轴 1 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC1	
SINUMERIK 840D sl	DB21,... .DBX341.0	

<Chan>.orientationAxis3.in.inc10Active	**NC → PLC**	**Bool**
“收到请求时移动摆动轴 10 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC10	
SINUMERIK 840D sl	DB21,... .DBX341.1	

<Chan>.orientationAxis3.in.inc100Active	**NC → PLC**	**Bool**
“收到请求时移动摆动轴 100 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC100	
SINUMERIK 840D sl	DB21,... .DBX341.2	

<Chan>.orientationAxis3.in.inc1000Active	**NC → PLC**	**Bool**
“收到请求时移动摆动轴 1 000 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC1000	
SINUMERIK 840D sl	DB21,... .DBX341.3	

<Chan>.orientationAxis3.in.inc10000Active	**NC → PLC**	**Bool**
“收到请求时移动摆动轴 10 000 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC10000	
SINUMERIK 840D sl	DB21,... .DBX341.4	

<Chan>.orientationAxis3.in.incVarActive	**NC → PLC**	**Bool**
“收到请求时移动摆动轴用户自定义数量的增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INCVar	
SINUMERIK 840D sl	DB21,... .DBX341.5	

<Chan>.orientationAxis3.in.contManualTraversActive	**NC → PLC**	**Bool**
连续移动摆动轴（标准）		
SINUMERIK 840D sl	DB21,... .DBX341.6	

<Chan>.orientationAxis3.out	**PLC → NC**
SINUMERIK ONE Basic Program	LBP_Chan*.A_Ori[3]
SINUMERIK 840D sl	DB21,... .DBX328.0..331.7

<Chan>.orientationAxis3.out.handwheelSelection	**PLC → NC**	**USInt**
分配手轮 1..6；0 表示无分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，二进制）		
SINUMERIK ONE Basic Program	LBP_Chan*.HW1	
SINUMERIK 840D sl	DB21,... .DBX328.0..2	

<Chan>.orientationAxis3.out.handwheelInvertDir	**PLC → NC**	**Bool**
请求取反分给摆动轴的手轮的旋转方向		
SINUMERIK ONE Basic Program	LBP_Chan*.InvHWDirOfRot	
SINUMERIK 840D sl	DB21,... .DBX331.0	

<Chan>.orientationAxis3.out.disableFeed	**PLC → NC**	**Bool**
为摆动轴禁用进给速度		
SINUMERIK ONE Basic Program	LBP_Chan*.FDStop	
SINUMERIK 840D sl	DB21,... .DBX328.3	

<Chan>.orientationAxis3.out.disableTraversKey	**PLC → NC**	**Bool**
为摆动轴禁用移动键		
SINUMERIK ONE Basic Program	LBP_Chan*.Disabl	
SINUMERIK 840D sl	DB21,... .DBX328.4	

<Chan>.orientationAxis3.out.rapidTraversOvr	**PLC → NC**	**Bool**
为摆动轴激活快速移动速度(MD32010)		
SINUMERIK ONE Basic Program	LBP_Chan*.RapidTrOR	
SINUMERIK 840D sl	DB21,... .DBX328.5	

<Chan>.orientationAxis3.out.traversKeyMinus	**PLC → NC**	**Bool**
请求为摆动轴激活“负向”键，使轴在负向移动		
SINUMERIK ONE Basic Program	LBP_Chan*.Minus	
SINUMERIK 840D sl	DB21,... .DBX328.6	

<Chan>.orientationAxis3.out.traversKeyPlus	**PLC → NC**	**Bool**
请求为摆动轴激活“正向”键，使轴在正向移动		
SINUMERIK ONE Basic Program	LBP_Chan*.Plus	
SINUMERIK 840D sl	DB21,... .DBX328.7	

<Chan>.orientationAxis3.out.inc1	**PLC → NC**	**Bool**
激活“收到请求时移动摆动轴 1 个增量”		
SINUMERIK ONE Basic Program	LBP_Chan*.INC1	
SINUMERIK 840D sl	DB21,... .DBX329.0	

<Chan>.orientationAxis3.out.inc10	**PLC → NC**	**Bool**
“收到请求时移动摆动轴 10 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC10	
SINUMERIK 840D sl	DB21,... .DBX329.1	

<Chan>.orientationAxis3.out.inc100	**PLC → NC**	**Bool**
“收到请求时移动摆动轴 100 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC100	
SINUMERIK 840D sl	DB21,... .DBX329.2	

<Chan>.orientationAxis3.out.inc1000	**PLC → NC**	**Bool**
“收到请求时移动摆动轴 1 000 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC1000	
SINUMERIK 840D sl	DB21,... .DBX329.3	

<Chan>.orientationAxis3.out.inc10000	**PLC → NC**	**Bool**
“收到请求时移动摆动轴 10 000 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.INC10000	
SINUMERIK 840D sl	DB21,... .DBX329.4	

<Chan>.orientationAxis3.out.incVar	**PLC → NC**	**Bool**
激活“收到请求时移动摆动轴用户自定义数量的增量”		
SINUMERIK ONE Basic Program	LBP_Chan*.INCVar	
SINUMERIK 840D sl	DB21,... .DBX329.5	

<Chan>.orientationAxis3.out.contManualTravers	**PLC → NC**	**Bool**
激活摆动轴的连续移动（标准）		
SINUMERIK 840D sl	DB21,... .DBX329.6	

<Chan>.toolMgmt.in.toolMissing	**NC → PLC**	**Bool**
程序指定的刀具缺少，或无法使用		
SINUMERIK ONE Basic Program	LBP_Chan*.E_ToolMissing	
SINUMERIK 840D sl	DB21,... .DBX317.7	

<Chan>.toolMgmt.in.toolPrewarnLimitReached	**NC → PLC**	**Bool**
达到刀具预警限值。超出预警限值刀具的 T 号在“ toolPrewarnLimitTNumber”中指出		
SINUMERIK ONE Basic Program	LBP_Chan*.E_TPrewarnLimit	
SINUMERIK 840D sl	DB21,... .DBX344.0	

<Chan>.toolMgmt.in.toolLimitValueReached	**NC → PLC**	**Bool**
达到刀具限值。超限刀具的 T 号在“ toolLimitTNumber”中指出		
SINUMERIK ONE Basic Program	LBP_Chan*.E_TLimit	
SINUMERIK 840D sl	DB21,... .DBX344.1	

<Chan>.toolMgmt.in.newReplacementToolActive	**NC → PLC**	**Bool**
已过渡到新的姊妹刀具。新姊妹刀具的 T 号位于 newReplacementToolTNumber 中		
SINUMERIK ONE Basic Program	LBP_Chan*.E_TNewRepl	
SINUMERIK 840D sl	DB21,... .DBX344.2	

<Chan>.toolMgmt.in.lastReplacementToolReached	**NC → PLC**	**Bool**
达到刀具组最后一个姊妹刀具(MD20310)。刀具组最后一把姊妹刀具的 T 号在“lastReplacementToolTNumber”中指明		
SINUMERIK ONE Basic Program	LBP_Chan*.E_TLastRepl	
SINUMERIK 840D sl	DB21,... .DBX344.3	

<Chan>.toolMgmt.in.toolPrewarnLimitTNumber	**NC → PLC**	**DInt**
超出预警限值的刀具的 T 号		
SINUMERIK ONE Basic Program	LBP_Chan*.E_TNoPrewarnLimit	
SINUMERIK 840D sl	DB21,... .DBD348	

<Chan>.toolMgmt.in.toolLimitTNumber	**NC → PLC**	**DInt**
超限刀具的 T 号		
SINUMERIK ONE Basic Program	LBP_Chan*.E_TNoLimit	
SINUMERIK 840D sl	DB21,... .DBD352	

<Chan>.toolMgmt.in.newTNumber	**NC → PLC**	**DInt**
新姊妹刀具的 T 号		
SINUMERIK ONE Basic Program	LBP_Chan*.E_TNoNewRepl	
SINUMERIK 840D sl	DB21,... .DBD356	

<Chan>.toolMgmt.in.lastTNumber	**NC → PLC**	**DInt**
上一把姊妹刀具的 T 号		
SINUMERIK ONE Basic Program	LBP_Chan*.E_TNoLastRepl	
SINUMERIK 840D sl	DB21,... .DBD360	

<Chan>.toolMgmt.out.toolLifeMonitoring	**PLC → NC**	**Bool**
确定刀具使用寿命，刀具管理(MD20310 位 17)		
SINUMERIK ONE Basic Program	LBP_Chan*.A_TM_TimOut	
SINUMERIK 840D sl	DB21,... .DBX1.3	

<Chan>.toolMgmt.out.deactivateWorkpieceCounter	**PLC → NC**	**Bool**
刀具管理：工件计数监控关闭已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.A_DisablWP_Counter	
SINUMERIK 840D sl	DB21,... .DBX29.5	

<Chan>.toolMgmt.out.deactivateWearMonitoring	**PLC → NC**	**Bool**
刀具管理：磨损监控关闭已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.A_DisablWearMon	
SINUMERIK 840D sl	DB21,... .DBX29.6	

<Chan>.toolMgmt.out.ignoreDisabledSate	**PLC → NC**	**Bool**
刀具管理：“刀具禁用”未生效		
SINUMERIK ONE Basic Program	LBP_Chan*.A_DontDisablTool	
SINUMERIK 840D sl	DB21,... .DBX29.7	

<Chan>.auxMDynFuncs.in.m00Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 00 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[0]	
SINUMERIK 840D sl	DB21,... .DBX194.0	

<Chan>.auxMDynFuncs.in.m01Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 01 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[1]	
SINUMERIK 840D sl	DB21,... .DBX194.1	

<Chan>.auxMDynFuncs.in.m02Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 02 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[2]	
SINUMERIK 840D sl	DB21,... .DBX194.2	

<Chan>.auxMDynFuncs.in.m03Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 03 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[3]	
SINUMERIK 840D sl	DB21,... .DBX194.3	

<Chan>.auxMDynFuncs.in.m04Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 04 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[4]	
SINUMERIK 840D sl	DB21,... .DBX194.4	

<Chan>.auxMDynFuncs.in.m05Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 05 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[5]	
SINUMERIK 840D sl	DB21,... .DBX194.5	

<Chan>.auxMDynFuncs.in.m06Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 06 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[6]	
SINUMERIK 840D sl	DB21,... .DBX194.6	

<Chan>.auxMDynFuncs.in.m07Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 07 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[7]	
SINUMERIK 840D sl	DB21,... .DBX194.7	

<Chan>.auxMDynFuncs.in.m08Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 08 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[8]	
SINUMERIK 840D sl	DB21,... .DBX195.0	

<Chan>.auxMDynFuncs.in.m09Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 09 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[9]	
SINUMERIK 840D sl	DB21,... .DBX195.1	

<Chan>.auxMDynFuncs.in.m10Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 10 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[10]	
SINUMERIK 840D sl	DB21,... .DBX195.2	

<Chan>.auxMDynFuncs.in.m11Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 11 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[11]	
SINUMERIK 840D sl	DB21,... .DBX195.3	

<Chan>.auxMDynFuncs.in.m12Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 12 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[12]	
SINUMERIK 840D sl	DB21,... .DBX195.4	

<Chan>.auxMDynFuncs.in.m13Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 13 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[13]	
SINUMERIK 840D sl	DB21,... .DBX195.5	

<Chan>.auxMDynFuncs.in.m14Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 14 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[14]		
SINUMERIK 840D sl DB21,... .DBX195.6		

<Chan>.auxMDynFuncs.in.m15Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 15 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[15]		
SINUMERIK 840D sl DB21,... .DBX195.7		

<Chan>.auxMDynFuncs.in.m16Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 16 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[16]		
SINUMERIK 840D sl DB21,... .DBX196.0		

<Chan>.auxMDynFuncs.in.m17Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 17 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[17]		
SINUMERIK 840D sl DB21,... .DBX196.1		

<Chan>.auxMDynFuncs.in.m18Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 18 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[18]		
SINUMERIK 840D sl DB21,... .DBX196.2		

<Chan>.auxMDynFuncs.in.m19Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 19 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[19]		
SINUMERIK 840D sl DB21,... .DBX196.3		

<Chan>.auxMDynFuncs.in.m20Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 20 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[20]		
SINUMERIK 840D sl DB21,... .DBX196.4		

<Chan>.auxMDynFuncs.in.m21Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 21 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[21]	
SINUMERIK 840D sl	DB21,... .DBX196.5	

<Chan>.auxMDynFuncs.in.m22Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 22 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[22]	
SINUMERIK 840D sl	DB21,... .DBX196.6	

<Chan>.auxMDynFuncs.in.m23Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 23 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[23]	
SINUMERIK 840D sl	DB21,... .DBX196.7	

<Chan>.auxMDynFuncs.in.m24Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 24 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[24]	
SINUMERIK 840D sl	DB21,... .DBX197.0	

<Chan>.auxMDynFuncs.in.m25Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 25 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[25]	
SINUMERIK 840D sl	DB21,... .DBX197.1	

<Chan>.auxMDynFuncs.in.m26Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 26 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[26]	
SINUMERIK 840D sl	DB21,... .DBX197.2	

<Chan>.auxMDynFuncs.in.m27Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 27 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[27]	
SINUMERIK 840D sl	DB21,... .DBX197.3	

<Chan>.auxMDynFuncs.in.m28Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 28 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[28]		
SINUMERIK 840D sl DB21,... .DBX197.4		

<Chan>.auxMDynFuncs.in.m29Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 29 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[29]		
SINUMERIK 840D sl DB21,... .DBX197.5		

<Chan>.auxMDynFuncs.in.m30Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 30 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[30]		
SINUMERIK 840D sl DB21,... .DBX197.6		

<Chan>.auxMDynFuncs.in.m31Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 31 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[31]		
SINUMERIK 840D sl DB21,... .DBX197.7		

<Chan>.auxMDynFuncs.in.m32Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 32 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[32]		
SINUMERIK 840D sl DB21,... .DBX198.0		

<Chan>.auxMDynFuncs.in.m33Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 33 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[33]		
SINUMERIK 840D sl DB21,... .DBX198.1		

<Chan>.auxMDynFuncs.in.m34Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 34 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[34]		
SINUMERIK 840D sl DB21,... .DBX198.2		

<Chan>.auxMDynFuncs.in.m35Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 35 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[35]	
SINUMERIK 840D sl	DB21,... .DBX198.3	

<Chan>.auxMDynFuncs.in.m36Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 36 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[36]	
SINUMERIK 840D sl	DB21,... .DBX198.4	

<Chan>.auxMDynFuncs.in.m37Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 37 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[37]	
SINUMERIK 840D sl	DB21,... .DBX198.5	

<Chan>.auxMDynFuncs.in.m38Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 38 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[38]	
SINUMERIK 840D sl	DB21,... .DBX198.6	

<Chan>.auxMDynFuncs.in.m39Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 39 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[39]	
SINUMERIK 840D sl	DB21,... .DBX198.7	

<Chan>.auxMDynFuncs.in.m40Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 40 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[40]	
SINUMERIK 840D sl	DB21,... .DBX199.0	

<Chan>.auxMDynFuncs.in.m41Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 41 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[41]	
SINUMERIK 840D sl	DB21,... .DBX199.1	

<Chan>.auxMDynFuncs.in.m42Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 42 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[42]	
SINUMERIK 840D sl	DB21,... .DBX199.2	

<Chan>.auxMDynFuncs.in.m43Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 43 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[43]	
SINUMERIK 840D sl	DB21,... .DBX199.3	

<Chan>.auxMDynFuncs.in.m44Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 44 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[44]	
SINUMERIK 840D sl	DB21,... .DBX199.4	

<Chan>.auxMDynFuncs.in.m45Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 45 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[45]	
SINUMERIK 840D sl	DB21,... .DBX199.5	

<Chan>.auxMDynFuncs.in.m46Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 46 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[46]	
SINUMERIK 840D sl	DB21,... .DBX199.6	

<Chan>.auxMDynFuncs.in.m47Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 47 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[47]	
SINUMERIK 840D sl	DB21,... .DBX199.7	

<Chan>.auxMDynFuncs.in.m48Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 48 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[48]	
SINUMERIK 840D sl	DB21,... .DBX210.0	

<Chan>.auxMDynFuncs.in.m49Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 49 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[49]	
SINUMERIK 840D sl	DB21,... .DBX210.1	

<Chan>.auxMDynFuncs.in.m50Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 50 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[50]	
SINUMERIK 840D sl	DB21,... .DBX210.2	

<Chan>.auxMDynFuncs.in.m51Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 51 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[51]	
SINUMERIK 840D sl	DB21,... .DBX210.3	

<Chan>.auxMDynFuncs.in.m52Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 52 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[52]	
SINUMERIK 840D sl	DB21,... .DBX210.4	

<Chan>.auxMDynFuncs.in.m53Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 53 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[53]	
SINUMERIK 840D sl	DB21,... .DBX210.5	

<Chan>.auxMDynFuncs.in.m54Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 54 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[54]	
SINUMERIK 840D sl	DB21,... .DBX210.6	

<Chan>.auxMDynFuncs.in.m55Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 55 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[55]	
SINUMERIK 840D sl	DB21,... .DBX210.7	

<Chan>.auxMDynFuncs.in.m56Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 56 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[56]	
SINUMERIK 840D sl	DB21,... .DBX211.0	

<Chan>.auxMDynFuncs.in.m57Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 57 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[57]	
SINUMERIK 840D sl	DB21,... .DBX211.1	

<Chan>.auxMDynFuncs.in.m58Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 58 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[58]	
SINUMERIK 840D sl	DB21,... .DBX211.2	

<Chan>.auxMDynFuncs.in.m59Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 59 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[59]	
SINUMERIK 840D sl	DB21,... .DBX211.3	

<Chan>.auxMDynFuncs.in.m60Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 60 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[60]	
SINUMERIK 840D sl	DB21,... .DBX211.4	

<Chan>.auxMDynFuncs.in.m61Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 61 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[61]	
SINUMERIK 840D sl	DB21,... .DBX211.5	

<Chan>.auxMDynFuncs.in.m62Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 62 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[62]	
SINUMERIK 840D sl	DB21,... .DBX211.6	

<Chan>.auxMDynFuncs.in.m63Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 63 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[63]	
SINUMERIK 840D sl	DB21,... .DBX211.7	

<Chan>.auxMDynFuncs.in.m64Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 64 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[64]	
SINUMERIK 840D sl	DB21,... .DBX212.0	

<Chan>.auxMDynFuncs.in.m65Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 65 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[65]	
SINUMERIK 840D sl	DB21,... .DBX212.1	

<Chan>.auxMDynFuncs.in.m66Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 66 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[66]	
SINUMERIK 840D sl	DB21,... .DBX212.2	

<Chan>.auxMDynFuncs.in.m67Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 67 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[67]	
SINUMERIK 840D sl	DB21,... .DBX212.3	

<Chan>.auxMDynFuncs.in.m68Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 68 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[68]	
SINUMERIK 840D sl	DB21,... .DBX212.4	

<Chan>.auxMDynFuncs.in.m69Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 69 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[69]	
SINUMERIK 840D sl	DB21,... .DBX212.5	

<Chan>.auxMDynFuncs.in.m70Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 70 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[70]		
SINUMERIK 840D sl DB21,... .DBX212.6		

<Chan>.auxMDynFuncs.in.m71Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 71 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[71]		
SINUMERIK 840D sl DB21,... .DBX212.7		

<Chan>.auxMDynFuncs.in.m72Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 72 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[72]		
SINUMERIK 840D sl DB21,... .DBX213.0		

<Chan>.auxMDynFuncs.in.m73Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 73 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[73]		
SINUMERIK 840D sl DB21,... .DBX213.1		

<Chan>.auxMDynFuncs.in.m74Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 74 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[74]		
SINUMERIK 840D sl DB21,... .DBX213.2		

<Chan>.auxMDynFuncs.in.m75Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 75 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[75]		
SINUMERIK 840D sl DB21,... .DBX213.3		

<Chan>.auxMDynFuncs.in.m76Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 76 已解码		
SINUMERIK ONE Basic Program LBP_Chan*.MDyn[76]		
SINUMERIK 840D sl DB21,... .DBX213.4		

<Chan>.auxMDynFuncs.in.m77Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 77 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[77]	
SINUMERIK 840D sl	DB21,... .DBX213.5	

<Chan>.auxMDynFuncs.in.m78Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 78 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[78]	
SINUMERIK 840D sl	DB21,... .DBX213.6	

<Chan>.auxMDynFuncs.in.m79Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 79 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[79]	
SINUMERIK 840D sl	DB21,... .DBX213.7	

<Chan>.auxMDynFuncs.in.m80Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 80 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[80]	
SINUMERIK 840D sl	DB21,... .DBX214.0	

<Chan>.auxMDynFuncs.in.m81Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 81 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[81]	
SINUMERIK 840D sl	DB21,... .DBX214.1	

<Chan>.auxMDynFuncs.in.m82Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 82 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[82]	
SINUMERIK 840D sl	DB21,... .DBX214.2	

<Chan>.auxMDynFuncs.in.m83Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 83 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[83]	
SINUMERIK 840D sl	DB21,... .DBX214.3	

<Chan>.auxMDynFuncs.in.m84Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 84 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[84]	
SINUMERIK 840D sl	DB21,... .DBX214.4	

<Chan>.auxMDynFuncs.in.m85Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 85 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[85]	
SINUMERIK 840D sl	DB21,... .DBX214.5	

<Chan>.auxMDynFuncs.in.m86Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 86 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[86]	
SINUMERIK 840D sl	DB21,... .DBX214.6	

<Chan>.auxMDynFuncs.in.m87Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 87 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[87]	
SINUMERIK 840D sl	DB21,... .DBX214.7	

<Chan>.auxMDynFuncs.in.m88Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 88 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[88]	
SINUMERIK 840D sl	DB21,... .DBX215.0	

<Chan>.auxMDynFuncs.in.m89Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 89 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[89]	
SINUMERIK 840D sl	DB21,... .DBX215.1	

<Chan>.auxMDynFuncs.in.m90Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 90 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[90]	
SINUMERIK 840D sl	DB21,... .DBX215.2	

<Chan>.auxMDynFuncs.in.m91Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 91 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[91]	
SINUMERIK 840D sl	DB21,... .DBX215.3	

<Chan>.auxMDynFuncs.in.m92Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 92 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[92]	
SINUMERIK 840D sl	DB21,... .DBX215.4	

<Chan>.auxMDynFuncs.in.m93Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 93 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[93]	
SINUMERIK 840D sl	DB21,... .DBX215.5	

<Chan>.auxMDynFuncs.in.m94Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 94 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[94]	
SINUMERIK 840D sl	DB21,... .DBX215.6	

<Chan>.auxMDynFuncs.in.m95Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 95 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[95]	
SINUMERIK 840D sl	DB21,... .DBX215.7	

<Chan>.auxMDynFuncs.in.m96Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 96 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[96]	
SINUMERIK 840D sl	DB21,... .DBX216.0	

<Chan>.auxMDynFuncs.in.m97Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 97 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[97]	
SINUMERIK 840D sl	DB21,... .DBX216.1	

<Chan>.auxMDynFuncs.in.m98Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 98 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[98]	
SINUMERIK 840D sl	DB21,... .DBX216.2	

<Chan>.auxMDynFuncs.in.m99Decoded	**NC → PLC**	**Bool**
出现 0 → 1 上升沿：M 功能 99 已解码		
SINUMERIK ONE Basic Program	LBP_Chan*.MDyn[99]	
SINUMERIK 840D sl	DB21,... .DBX216.3	

<Chan>.auxMFuncs.auxMFuncs[1].in	**NC → PLC**

<Chan>.auxMFuncs[1].in.hasChanged	**NC → PLC**	**Bool**
M 功能已变化，信号表明，对应值有效，信号持续一个 PLC 周期		
SINUMERIK ONE Basic Program	LBP_Chan*.M1Change - M5Change	
SINUMERIK 840D sl	DB21,... .DBX58.1..4	

<Chan>.auxMFuncs[1].in.notDecoded	**NC → PLC**	**Bool**
M 功能不位于解码列表中，它持续一个 PLC 周期		
SINUMERIK ONE Basic Program	LBP_Chan*.M1NDec- M5NDec	
SINUMERIK 840D sl	DB21,... .DBX59.1..4	

<Chan>.auxMFuncs[1].in.fastAckActive	**NC → PLC**	**Bool**
M 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.M1Quick - M5Quick	
SINUMERIK 840D sl	DB21,... .DBX66.1..4	

<Chan>.auxMFuncs[1].in.extendedAddress	**NC → PLC**	**Int**
扩展 M 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtM1 - ExtM5	
SINUMERIK 840D sl	DB21,... .DBW68, .DBW74, .DBW80, .DBW86, .DBW92	

<Chan>.auxMFuncs[1].in.value		**NC → PLC**	**DInt**
M 功能值			
SINUMERIK ONE Basic Program	LBP_Chan*.M1 - M5		
SINUMERIK 840D sl	DB21,... .DBD70, .DBD76, .DBD82, .DBD88, .DBD94		

<Chan>.auxMFuncs.auxMFuncs[2].in	**NC → PLC**

<Chan>.auxMFuncs[2].in.hasChanged		**NC → PLC**	**Bool**
M 功能已变化，信号表明，对应值有效，信号持续一个 PLC 周期			
SINUMERIK ONE Basic Program	LBP_Chan*.M2Change		
SINUMERIK 840D sl	DB21,... .DBX58.1		

<Chan>.auxMFuncs[2].in.notDecoded		**NC → PLC**	**Bool**
M 功能不位于解码列表中，它持续一个 PLC 周期			
SINUMERIK ONE Basic Program	LBP_Chan*.M2NDec		
SINUMERIK 840D sl	DB21,... .DBX59.1		

<Chan>.auxMFuncs[2].in.fastAckActive		**NC → PLC**	**Bool**
M 功能快速应答已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.M2Quick		
SINUMERIK 840D sl	DB21,... .DBX66.1		

<Chan>.auxMFuncs[2].in.extendedAddress		**NC → PLC**	**Int**
扩展 M 功能地址			
SINUMERIK ONE Basic Program	LBP_Chan*.ExtM2		
SINUMERIK 840D sl	DB21,... .DBW74		

<Chan>.auxMFuncs[2].in.value		**NC → PLC**	**DInt**
M 功能值			
SINUMERIK ONE Basic Program	LBP_Chan*.M2		
SINUMERIK 840D sl	DB21,... .DBD76		

<Chan>.auxMFuncs.auxMFuncs[3].in	**NC → PLC**

<Chan>.auxMFuncs[3].in.hasChanged	**NC → PLC**	**Bool**
M 功能已变化，信号表明，对应值有效，信号持续一个 PLC 周期		
SINUMERIK ONE Basic Program	LBP_Chan*.M3Change	
SINUMERIK 840D sl	DB21,... .DBX58.2	

<Chan>.auxMFuncs[3].in.notDecoded	**NC → PLC**	**Bool**
M 功能不位于解码列表中，它持续一个 PLC 周期		
SINUMERIK ONE Basic Program	LBP_Chan*.M3NDec	
SINUMERIK 840D sl	DB21,... .DBX59.2	

<Chan>.auxMFuncs[3].in.fastAckActive	**NC → PLC**	**Bool**
M 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.M3Quick	
SINUMERIK 840D sl	DB21,... .DBX66.2	

<Chan>.auxMFuncs[3].in.extendedAddress	**NC → PLC**	**Int**
扩展 M 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtM3	
SINUMERIK 840D sl	DB21,... .DBW80	

<Chan>.auxMFuncs[3].in.value	**NC → PLC**	**DInt**
M 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.M3	
SINUMERIK 840D sl	DB21,... .DBD82	

<Chan>.auxMFuncs.auxMFuncs[4].in	**NC → PLC**

<Chan>.auxMFuncs[4].in.hasChanged	**NC → PLC**	**Bool**
M 功能已变化，信号表明，对应值有效，信号持续一个 PLC 周期		
SINUMERIK ONE Basic Program	LBP_Chan*.M4Change	
SINUMERIK 840D sl	DB21,... .DBX58.3	

<Chan>.auxMFuncs[4].in.notDecoded	**NC → PLC**	**Bool**
M 功能不位于解码列表中，它持续一个 PLC 周期		
SINUMERIK ONE Basic Program	LBP_Chan*.M4NDec	
SINUMERIK 840D sl	DB21,... .DBX59.3	

<Chan>.auxMFuncs[4].in.fastAckActive	**NC → PLC**	**Bool**
M 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.M4Quick	
SINUMERIK 840D sl	DB21,... .DBX66.3	

<Chan>.auxMFuncs[4].in.extendedAddress	**NC → PLC**	**Int**
扩展 M 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtM4	
SINUMERIK 840D sl	DB21,... .DBW86	

<Chan>.auxMFuncs[4].in.value	**NC → PLC**	**DInt**
M 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.M4	
SINUMERIK 840D sl	DB21,... .DBD88	

<Chan>.auxMFuncs.auxMFuncs[5].in	**NC → PLC**	

<Chan>.auxMFuncs[5].in.hasChanged	**NC → PLC**	**Bool**
M 功能已变化，信号表明，对应值有效，信号持续一个 PLC 周期		
SINUMERIK ONE Basic Program	LBP_Chan*.M5Change	
SINUMERIK 840D sl	DB21,... .DBX58.4	

<Chan>.auxMFuncs[5].in.notDecoded	**NC → PLC**	**Bool**
M 功能不位于解码列表中，它持续一个 PLC 周期		
SINUMERIK ONE Basic Program	LBP_Chan*.M5NDec	
SINUMERIK 840D sl	DB21,... .DBX59.4	

<Chan>.auxMFuncs[5].in.fastAckActive	**NC → PLC**	**Bool**
M 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.M5Quick	
SINUMERIK 840D sl	DB21,... .DBX66.4	

<Chan>.auxMFuncs[5].in.extendedAddress	**NC → PLC**	**Int**
扩展 M 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtM5	
SINUMERIK 840D sl	DB21,... .DBW92	

<Chan>.auxMFuncs[5].in.value	**NC → PLC**	**DInt**
M 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.M5	
SINUMERIK 840D sl	DB21,... .DBD94	

<Chan>.auxSFuncs.auxSFuncs[1].in	**NC → PLC**

<Chan>.auxSFuncs[1].in.hasChanged	**NC → PLC**	**Bool**
S 功能已变化，信号表明，对应值有效		
SINUMERIK ONE Basic Program	LBP_Chan*.S1Change - S3Change	
SINUMERIK 840D sl	DB21,... .DBX60.0..2	

<Chan>.auxSFuncs[1].in.fastAckActive	**NC → PLC**	**Bool**
S 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.S1Quick - S3Quick	
SINUMERIK 840D sl	DB21,... .DBX60.4..6	

<Chan>.auxSFuncs[1].in.extendedAddress	**NC → PLC**	**Int**
扩展 S 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtS1 - ExtS3	
SINUMERIK 840D sl	DB21,... .DBW98, .DBW104, .DBW110	

<Chan>.auxSFuncs[1].in.value		**NC → PLC**	**Real**
S 功能值			
SINUMERIK ONE Basic Program	LBP_Chan*.S1 - S3		
SINUMERIK 840D sl	DB21,... .DBD100, .DBD106, .DBD112		

<Chan>.auxSFuncs.auxSFuncs[2].in	**NC → PLC**

<Chan>.auxSFuncs[2].in.hasChanged		**NC → PLC**	**Bool**
S 功能已变化，信号表明，对应值有效			
SINUMERIK ONE Basic Program	LBP_Chan*.S2Change		
SINUMERIK 840D sl	DB21,... .DBX60.1		

<Chan>.auxSFuncs[2].in.fastAckActive		**NC → PLC**	**Bool**
S 功能快速应答已生效			
SINUMERIK ONE Basic Program	LBP_Chan*.S2Quick		
SINUMERIK 840D sl	DB21,... .DBX60.5		

<Chan>.auxSFuncs[2].in.extendedAddress		**NC → PLC**	**Int**
扩展 S 功能地址			
SINUMERIK ONE Basic Program	LBP_Chan*.ExtS2		
SINUMERIK 840D sl	DB21,... .DBW104		

<Chan>.auxSFuncs[2].in.value		**NC → PLC**	**Real**
S 功能值			
SINUMERIK ONE Basic Program	LBP_Chan*.S2		
SINUMERIK 840D sl	DB21,... .DBD106		

<Chan>.auxSFuncs.auxSFuncs[3].in	**NC → PLC**

<Chan>.auxSFuncs[3].in.hasChanged		**NC → PLC**	**Bool**
S 功能已变化，信号表明，对应值有效			
SINUMERIK ONE Basic Program	LBP_Chan*.S3Change		
SINUMERIK 840D sl	DB21,... .DBX60.2		

<Chan>.auxSFuncs[3].in.fastAckActive	**NC → PLC**	**Bool**
S 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.S3Quick	
SINUMERIK 840D sl	DB21,... .DBX60.6	

<Chan>.auxSFuncs[3].in.extendedAddress	**NC → PLC**	**Int**
扩展 S 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtS3	
SINUMERIK 840D sl	DB21,... .DBW110	

<Chan>.auxSFuncs[3].in.value	**NC → PLC**	**Real**
S 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.S3	
SINUMERIK 840D sl	DB21,... .DBD112	

<Chan>.auxTFuncs.auxTFuncs[1].in	**NC → PLC**

<Chan>.auxTFuncs[1].in.hasChanged	**NC → PLC**	**Bool**
T 功能已变化，信号表明，对应值有效		
SINUMERIK ONE Basic Program	LBP_Chan*.T1Change - T3Change	
SINUMERIK 840D sl	DB21,... .DBX61.0..2	

<Chan>.auxTFuncs[1].in.fastAckActive	**NC → PLC**	**Bool**
T 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.T1Quick - T3Quick	
SINUMERIK 840D sl	DB21,... .DBX61.4..6	

<Chan>.auxTFuncs[1].in.extendedAddress	**NC → PLC**	**Int**
扩展 T 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtT1 - ExtT3	
SINUMERIK 840D sl	DB21,... .DBW116, .DBW120, .DBW124	

<Chan>.auxTFuncs[1].in.value	**NC → PLC**	**Int**
T 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.T1 - T3	
SINUMERIK 840D sl	DB21,... .DBW118, .DBW122, .DBW126	

<Chan>.auxTFuncs.auxTFuncs[2].in	**NC → PLC**	

<Chan>.auxTFuncs[2].in.hasChanged	**NC → PLC**	**Bool**
T 功能已变化，信号表明，对应值有效		
SINUMERIK ONE Basic Program	LBP_Chan*.T2Change	
SINUMERIK 840D sl	DB21,... .DBX61.1	

<Chan>.auxTFuncs[2].in.fastAckActive	**NC → PLC**	**Bool**
T 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.T2Quick	
SINUMERIK 840D sl	DB21,... .DBX61.5	

<Chan>.auxTFuncs[2].in.extendedAddress	**NC → PLC**	**Int**
扩展 T 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtT2	
SINUMERIK 840D sl	DB21,... .DBW120	

<Chan>.auxTFuncs[2].in.value	**NC → PLC**	**Int**
T 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.T2	
SINUMERIK 840D sl	DB21,... .DBW122	

<Chan>.auxTFuncs.auxTFuncs[3].in	**NC → PLC**	

<Chan>.auxTFuncs[3].in.hasChanged	**NC → PLC**	**Bool**
T 功能已变化，信号表明，对应值有效		
SINUMERIK ONE Basic Program	LBP_Chan*.T3Change	
SINUMERIK 840D sl	DB21,... .DBX61.2	

<Chan>.auxTFuncs[3].in.fastAckActive	**NC → PLC**	**Bool**
T 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.T3Quick	
SINUMERIK 840D sl	DB21,... .DBX61.6	

<Chan>.auxTFuncs[3].in.extendedAddress	**NC → PLC**	**Int**
扩展 T 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtT3	
SINUMERIK 840D sl	DB21,... .DBW124	

<Chan>.auxTFuncs[3].in.value	**NC → PLC**	**Int**
T 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.T3	
SINUMERIK 840D sl	DB21,... .DBW126	

<Chan>.auxDFuncs.auxDFuncs[1].in	**NC → PLC**

<Chan>.auxDFuncs[1].in.hasChanged	**NC → PLC**	**Bool**
D 功能已变化，信号表明，对应值有效。		
SINUMERIK ONE Basic Program	LBP_Chan*.D1Change - D3Change	
SINUMERIK 840D sl	DB21,... .DBX62.0..2	

<Chan>.auxDFuncs[1].in.fastAckActive	**NC → PLC**	**Bool**
D 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.D1Quick - D3Quick	
SINUMERIK 840D sl	DB21,... .DBX62.4..6	

<Chan>.auxDFuncs[1].in.extendedAddress	**NC → PLC**	**Int**
扩展 D 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtD1 - ExtD3	
SINUMERIK 840D sl	DB21,... .DBB128, .DBB130, .DBB132	

<Chan>.auxDFuncs[1].in.value	**NC → PLC**	**USInt**
D 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.D1 - D3	
SINUMERIK 840D sl	DB21,... .DBB129, .DBB131, .DBB133	

<Chan>.auxDFuncs.auxDFuncs[2].in	**NC → PLC**	

<Chan>.auxDFuncs[2].in.hasChanged	**NC → PLC**	**Bool**
D 功能已变化，信号表明，对应值有效。		
SINUMERIK ONE Basic Program	LBP_Chan*.D2Change	
SINUMERIK 840D sl	DB21,... .DBX62.1	

<Chan>.auxDFuncs[2].in.fastAckActive	**NC → PLC**	**Bool**
D 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.D2Quick	
SINUMERIK 840D sl	DB21,... .DBX62.5	

<Chan>.auxDFuncs[2].in.extendedAddress	**NC → PLC**	**Int**
扩展 D 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtD2	
SINUMERIK 840D sl	DB21,... .DBB130	

<Chan>.auxDFuncs[2].in.value	**NC → PLC**	**USInt**
D 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.D2	
SINUMERIK 840D sl	DB21,... .DBB131	

<Chan>.auxDFuncs.auxDFuncs[3].in	**NC → PLC**	

<Chan>.auxDFuncs[3].in.hasChanged	**NC → PLC**	**Bool**
D 功能已变化，信号表明，对应值有效。		
SINUMERIK ONE Basic Program	LBP_Chan*.D3Change	
SINUMERIK 840D sl	DB21,... .DBX62.2	

<Chan>.auxDFuncs[3].in.fastAckActive	**NC → PLC**	**Bool**
D 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.D3Quick	
SINUMERIK 840D sl	DB21,... .DBX62.6	

<Chan>.auxDFuncs[3].in.extendedAddress	**NC → PLC**	**Int**
扩展 D 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtD3	
SINUMERIK 840D sl	DB21,... .DBB132	

<Chan>.auxDFuncs[3].in.value	**NC → PLC**	**USInt**
D 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.D3	
SINUMERIK 840D sl	DB21,... .DBB133	

<Chan>.auxDLFuncs.auxDLFuncs[1]	**NC → PLC**	

<Chan>.auxDLFuncs[1].in.hasChanged	**NC → PLC**	**Bool**
DL 功能已变化，信号表明，对应值有效。		
SINUMERIK ONE Basic Program	LBP_Chan*.EChange	
SINUMERIK 840D sl	DB21,... .DBX63.0	

<Chan>.auxDLFuncs[1].in.fastAckActive	**NC → PLC**	**Bool**
DL 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.EQuick	
SINUMERIK 840D sl	DB21,... .DBX63.4	

<Chan>.auxDLFuncs[1].in.extendedAddress	**NC → PLC**	**Int**
扩展 DL 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtE	
SINUMERIK 840D sl	DB21,... .DBW134	

<Chan>.auxDLFuncs[1].in.value	**NC → PLC**	**Real**
DL 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.E	
SINUMERIK 840D sl	DB21,... .DBD136	

<Chan>.auxHFuncs.auxHFuncs[1].in	**NC → PLC**	

<Chan>.auxHFuncs[1].in.hasChanged	**NC → PLC**	**Bool**
H 功能已变化，信号表明，对应值有效。		
SINUMERIK ONE Basic Program	LBP_Chan*.H1Change - H3Change	
SINUMERIK 840D sl	DB21,... .DBX64.0..2	

<Chan>.auxHFuncs[1].in.fastAckActive	**NC → PLC**	**Bool**
H 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.H1Quick - H3Quick	
SINUMERIK 840D sl	DB21,... .DBX64.4..6	

<Chan>.auxHFuncs[1].in.extendedAddress	**NC → PLC**	**Int**
扩展 H 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtH1 - ExtH3	
SINUMERIK 840D sl	DB21,... .DBW140, .DBW146, .DBW152	

<Chan>.auxHFuncs[1].in.value	**NC → PLC**	**Real**
H 功能值(REAL 型, MD22110 $MC_AUXFU_H_TYPE_INT=0)		
SINUMERIK ONE Basic Program	LBP_Chan*.H1 - H3	
SINUMERIK 840D sl	DB21,... .DBD142, .DBD148, .DBD154	

<Chan>.auxHFuncs[1].in.valueTypeDint	**NC → PLC**	**DInt**
H 功能值(DINT 型, MD22110 $MC_AUXFU_H_TYPE_INT=1)		
SINUMERIK ONE Basic Program	LBP_Chan*.H1_DInt	
SINUMERIK 840D sl	DB21,... .DBD396	

<Chan>.auxHFuncs.auxHFuncs[2].in	**NC → PLC**	

<Chan>.auxHFuncs[2].in.hasChanged	**NC → PLC**	**Bool**
H 功能已变化，信号表明，对应值有效。		
SINUMERIK ONE Basic Program	LBP_Chan*.H2Change	
SINUMERIK 840D sl	DB21,... .DBX64.1	

<Chan>.auxHFuncs[2].in.fastAckActive	**NC → PLC**	**Bool**
H 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.H2Quick	
SINUMERIK 840D sl	DB21,... .DBX64.5	

<Chan>.auxHFuncs[2].in.extendedAddress	**NC → PLC**	**Int**
扩展 H 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtH2	
SINUMERIK 840D sl	DB21,... .DBW146	

<Chan>.auxHFuncs[2].in.value	**NC → PLC**	**Real**
H 功能值(REAL 型, MD22110 $MC_AUXFU_H_TYPE_INT=0)		
SINUMERIK ONE Basic Program	LBP_Chan*.H2	
SINUMERIK 840D sl	DB21,... .DBD148	

<Chan>.auxHFuncs[2].in.valueTypeDint	**NC → PLC**	**DInt**
H 功能值(DINT 型, MD22110 $MC_AUXFU_H_TYPE_INT=1)		
SINUMERIK ONE Basic Program	LBP_Chan*.H2_DInt	
SINUMERIK 840D sl	DB21,... .DBD400	

<Chan>.auxHFuncs.auxHFuncs[3].in	**NC → PLC**

<Chan>.auxHFuncs[3].in.hasChanged	**NC → PLC**	**Bool**
H 功能已变化，信号表明，对应值有效。		
SINUMERIK ONE Basic Program	LBP_Chan*.H3Change	
SINUMERIK 840D sl	DB21,... .DBX64.2	

<Chan>.auxHFuncs[3].in.fastAckActive	**NC → PLC**	**Bool**
H 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.H3Quick	
SINUMERIK 840D sl	DB21,... .DBX64.6	

<Chan>.auxHFuncs[3].in.extendedAddress	**NC → PLC**	**Int**
扩展 H 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtH3	
SINUMERIK 840D sl	DB21,... .DBW152	

<Chan>.auxHFuncs[3].in.value	**NC → PLC**	**Real**
H 功能值(REAL 型, MD22110 $MC_AUXFU_H_TYPE_INT=0)		
SINUMERIK ONE Basic Program	LBP_Chan*.H3	
SINUMERIK 840D sl	DB21,... .DBD154	

<Chan>.auxHFuncs[3].in.valueTypeDint	**NC → PLC**	**DInt**
H 功能值(DINT 型, MD22110 $MC_AUXFU_H_TYPE_INT=1)		
SINUMERIK ONE Basic Program	LBP_Chan*.H3_DInt	
SINUMERIK 840D sl	DB21,... .DBD404	

<Chan>.auxFFuncs.auxFFuncs[1].in	**NC → PLC**	

<Chan>.auxFFuncs[1].in.hasChanged	**NC → PLC**	**Bool**
F 功能已变化，信号显示对应值有效。		
SINUMERIK ONE Basic Program	LBP_Chan*.F1Change - F6Change	
SINUMERIK 840D sl	DB21,... .DBX65.0..5	

<Chan>.auxFFuncs[1].in.fastAckActive	**NC → PLC**	**Bool**
F 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.F1Quick - F6Quick	
SINUMERIK 840D sl	DB21,... .DBX67.0..5	

<Chan>.auxFFuncs[1].in.extendedAddress	**NC → PLC**	**Int**
扩展 F 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtF1 - ExtF6	
SINUMERIK 840D sl	DB21,... .DBW158, .DBW164, .DBW170, .DBW176, .DBW182, .DBW188	

<Chan>.auxFFuncs[1].in.value	**NC → PLC**	**Real**
F 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.F1 - F6	
SINUMERIK 840D sl	DB21,... .DBD160, .DBD166, .DBD172, .DBD178, .DBD184, .DBD190	

<Chan>.auxFFuncs.auxFFuncs[2].in	**NC → PLC**

<Chan>.auxFFuncs[2].in.hasChanged	**NC → PLC**	**Bool**
F 功能已变化，信号显示对应值有效。		
SINUMERIK ONE Basic Program	LBP_Chan*.F2Change	
SINUMERIK 840D sl	DB21,... .DBX65.1	

<Chan>.auxFFuncs[2].in.fastAckActive	**NC → PLC**	**Bool**
F 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.F2Quick	
SINUMERIK 840D sl	DB21,... .DBX67.1	

<Chan>.auxFFuncs[2].in.extendedAddress	**NC → PLC**	**Int**
扩展 F 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtF2	
SINUMERIK 840D sl	DB21,... .DBW164	

<Chan>.auxFFuncs[2].in.value	**NC → PLC**	**Real**
F 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.F2	
SINUMERIK 840D sl	DB21,... .DBD166	

<Chan>.auxFFuncs.auxFFuncs[3].in	**NC → PLC**

<Chan>.auxFFuncs[3].in.hasChanged	**NC → PLC**	**Bool**
F 功能已变化，信号显示对应值有效。		
SINUMERIK ONE Basic Program	LBP_Chan*.F3Change	
SINUMERIK 840D sl	DB21,... .DBX65.2	

<Chan>.auxFFuncs[3].in.fastAckActive	**NC → PLC**	**Bool**
F 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.F3Quick	
SINUMERIK 840D sl	DB21,... .DBX67.2	

<Chan>.auxFFuncs[3].in.extendedAddress	**NC → PLC**	**Int**
扩展 F 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtF3	
SINUMERIK 840D sl	DB21,... .DBW170	

<Chan>.auxFFuncs[3].in.value	**NC → PLC**	**Real**
F 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.F3	
SINUMERIK 840D sl	DB21,... .DBD172	

<Chan>.auxFFuncs.auxFFuncs[4].in	**NC → PLC**

<Chan>.auxFFuncs[4].in.hasChanged	**NC → PLC**	**Bool**
F 功能已变化，信号显示对应值有效。		
SINUMERIK ONE Basic Program	LBP_Chan*.F4Change	
SINUMERIK 840D sl	DB21,... .DBX65.3	

<Chan>.auxFFuncs[4].in.fastAckActive	**NC → PLC**	**Bool**
F 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.F4Quick	
SINUMERIK 840D sl	DB21,... .DBX67.3	

<Chan>.auxFFuncs[4].in.extendedAddress	**NC → PLC**	**Int**
扩展 F 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtF4	
SINUMERIK 840D sl	DB21,... .DBW176	

<Chan>.auxFFuncs[4].in.value	**NC → PLC**	**Real**
F 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.F4	
SINUMERIK 840D sl	DB21,... .DBD178	

<Chan>.auxFFuncs.auxFFuncs[5].in	**NC → PLC**

<Chan>.auxFFuncs[5].in.hasChanged	**NC → PLC**	**Bool**
F 功能已变化，信号显示对应值有效。		
SINUMERIK ONE Basic Program	LBP_Chan*.F5Change	
SINUMERIK 840D sl	DB21,... .DBX65.4	

<Chan>.auxFFuncs[5].in.fastAckActive	**NC → PLC**	**Bool**
F 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.F5Quick	
SINUMERIK 840D sl	DB21,... .DBX67.4	

<Chan>.auxFFuncs[5].in.extendedAddress	**NC → PLC**	**Int**
扩展 F 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtF5	
SINUMERIK 840D sl	DB21,... .DBW182	

<Chan>.auxFFuncs[5].in.value	**NC → PLC**	**Real**
F 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.F5	
SINUMERIK 840D sl	DB21,... .DBD184	

<Chan>.auxFFuncs.auxFFuncs[6].in	**NC → PLC**

<Chan>.auxFFuncs[6].in.hasChanged	**NC → PLC**	**Bool**
F 功能已变化，信号显示对应值有效。		
SINUMERIK ONE Basic Program	LBP_Chan*.F6Change	
SINUMERIK 840D sl	DB21,... .DBX65.5	

<Chan>.auxFFuncs[6].in.fastAckActive	**NC → PLC**	**Bool**
F 功能快速应答已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.F6Quick	
SINUMERIK 840D sl	DB21,... .DBX67.5	

<Chan>.auxFFuncs[6].in.extendedAddress	**NC → PLC**	**Int**
扩展 F 功能地址		
SINUMERIK ONE Basic Program	LBP_Chan*.ExtF6	
SINUMERIK 840D sl	DB21,... .DBW188	

<Chan>.auxFFuncs[6].in.value	**NC → PLC**	**Real**
F 功能值		
SINUMERIK ONE Basic Program	LBP_Chan*.F6	
SINUMERIK 840D sl	DB21,... .DBD190	

<Chan>.protAreas.in.protAreas.in[1]	**NC → PLC**	

<Chan>.protAreas[1].in.ncPreactivated	**NC → PLC**	**Bool**
机床专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPactivated[1] - [10]	
SINUMERIK 840D sl	DB21,... .DBX272.0..273.1	

<Chan>.protAreas[1].in.ncViolated	**NC → PLC**	**Bool**
进入了机床专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPViolated[1] - [10]	
SINUMERIK 840D sl	DB21,... .DBX276.0..277.1	

<Chan>.protAreas[1].in.chanPreactivated	**NC → PLC**	**Bool**
通道专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPactivated[1] - [10]	
SINUMERIK 840D sl	DB21,... .DBX274.0..275.1	

<Chan>.protAreas[1].in.chanViolated	**NC → PLC**	**Bool**
进入了通道专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPViolated[1] - [10]	
SINUMERIK 840D sl	DB21,... .DBX278.0..279.1	

<Chan>.protAreas[1].out.nc	**PLC → NC**	**Bool**
激活机床专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.A_MRP[1] - [10]	
SINUMERIK 840D sl	DB21,... .DBX8.0..9.1	

<Chan>.protAreas[1].out.chan	**PLC → NC**	**Bool**
激活通道专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.A_CSP[1] - [10]	
SINUMERIK 840D sl	DB21,... .DBX10.0..11.1	

<Chan>.protAreas[2].in.ncPreactivated	**NC → PLC**	**Bool**
机床专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPactivated[2]	
SINUMERIK 840D sl	DB21,... .DBX272.1	

<Chan>.protAreas[2].in.ncViolated	**NC → PLC**	**Bool**
进入了机床专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPViolated[2]	
SINUMERIK 840D sl	DB21,... .DBX276.1	

<Chan>.protAreas[2].in.chanPreactivated	**NC → PLC**	**Bool**
通道专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPactivated[2]	
SINUMERIK 840D sl	DB21,... .DBX274.1	

<Chan>.protAreas[2].in.chanViolated	**NC → PLC**	**Bool**
进入了通道专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPViolated[2]	
SINUMERIK 840D sl	DB21,... .DBX278.1	

<Chan>.protAreas[2].out.nc	**PLC → NC**	**Bool**
激活机床专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.A_MRP[2]	
SINUMERIK 840D sl	DB21,... .DBX8.1	

<Chan>.protAreas[2].out.chan	**PLC → NC**	**Bool**
激活通道专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.A_CSP[2]	
SINUMERIK 840D sl	DB21,... .DBX10.1	

<Chan>.protAreas[3].in.ncPreactivated	**NC → PLC**	**Bool**
机床专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPactivated[3]	
SINUMERIK 840D sl	DB21,... .DBX272.2	

<Chan>.protAreas[3].in.ncViolated	**NC → PLC**	**Bool**
进入了机床专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPViolated[3]	
SINUMERIK 840D sl	DB21,... .DBX276.2	

<Chan>.protAreas[3].in.chanPreactivated	**NC → PLC**	**Bool**
通道专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPactivated[3]	
SINUMERIK 840D sl	DB21,... .DBX274.2	

<Chan>.protAreas[3].in.chanViolated	**NC → PLC**	**Bool**
进入了通道专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPViolated[3]	
SINUMERIK 840D sl	DB21,... .DBX278.2	

<Chan>.protAreas[3].out.nc		**PLC → NC**	**Bool**
激活机床专用保护区			
SINUMERIK ONE Basic Program	LBP_Chan*.A_MRP[3]		
SINUMERIK 840D sl	DB21,... .DBX8.2		

<Chan>.protAreas[3].out.chan		**PLC → NC**	**Bool**
激活通道专用保护区			
SINUMERIK ONE Basic Program	LBP_Chan*.A_CSP[3]		
SINUMERIK 840D sl	DB21,... .DBX10.2		

<Chan>.protAreas[4].in.ncPreactivated		**NC → PLC**	**Bool**
机床专用保护区已预激活			
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPactivated[4]		
SINUMERIK 840D sl	DB21,... .DBX272.3		

<Chan>.protAreas[4].in.ncViolated		**NC → PLC**	**Bool**
进入了机床专用保护区			
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPViolated[4]		
SINUMERIK 840D sl	DB21,... .DBX276.3		

<Chan>.protAreas[4].in.chanPreactivated		**NC → PLC**	**Bool**
通道专用保护区已预激活			
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPactivated[4]		
SINUMERIK 840D sl	DB21,... .DBX274.3		

<Chan>.protAreas[4].in.chanSpecificViolated		**NC → PLC**	**Bool**
进入了通道专用保护区			
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPViolated[4]		
SINUMERIK 840D sl	DB21,... .DBX278.3		

<Chan>.protAreas[4].out.nc		**PLC → NC**	**Bool**
激活机床专用保护区			
SINUMERIK ONE Basic Program	LBP_Chan*.A_MRP[4]		
SINUMERIK 840D sl	DB21,... .DBX8.3		

<Chan>.protAreas[4].out.chan		**PLC → NC**	**Bool**
激活通道专用保护区			
SINUMERIK ONE Basic Program	LBP_Chan*.A_CSP[4]		
SINUMERIK 840D sl	DB21,... .DBX10.3		

<Chan>.protAreas[5].in.ncPreactivated		**NC → PLC**	**Bool**
机床专用保护区已预激活			
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPactivated[5]		
SINUMERIK 840D sl	DB21,... .DBX272.4		

<Chan>.protAreas[5].in.ncViolated		**NC → PLC**	**Bool**
进入了机床专用保护区			
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPViolated[5]		
SINUMERIK 840D sl	DB21,... .DBX276.4		

<Chan>.protAreas[5].in.chanSpecificPreactivated		**NC → PLC**	**Bool**
通道专用保护区已预激活			
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPactivated[5]		
SINUMERIK 840D sl	DB21,... .DBX274.4		

<Chan>.protAreas[5].in.chanViolated		**NC → PLC**	**Bool**
进入了通道专用保护区			
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPViolated[5]		
SINUMERIK 840D sl	DB21,... .DBX278.4		

<Chan>.protAreas[5].out.nc		**PLC → NC**	**Bool**
激活机床专用保护区			
SINUMERIK ONE Basic Program	LBP_Chan*.A_MRP[5]		
SINUMERIK 840D sl	DB21,... .DBX8.4		

<Chan>.protAreas[5].out.chan		**PLC → NC**	**Bool**
激活通道专用保护区			
SINUMERIK ONE Basic Program	LBP_Chan*.A_CSP[5]		
SINUMERIK 840D sl	DB21,... .DBX10.4		

<Chan>.protAreas[6].in.ncPreactivated	**NC → PLC**	**Bool**
机床专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPactivated[6]	
SINUMERIK 840D sl	DB21,... .DBX272.5	

<Chan>.protAreas[6].in.ncViolated	**NC → PLC**	**Bool**
进入了机床专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPViolated[6]	
SINUMERIK 840D sl	DB21,... .DBX276.5	

<Chan>.protAreas[6].in.chanPreactivated	**NC → PLC**	**Bool**
通道专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPactivated[6]	
SINUMERIK 840D sl	DB21,... .DBX274.5	

<Chan>.protAreas[6].in.chanViolated	**NC → PLC**	**Bool**
进入了通道专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPViolated[6]	
SINUMERIK 840D sl	DB21,... .DBX278.5	

<Chan>.protAreas[6].out.nc	**PLC → NC**	**Bool**
激活机床专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.A_MRP[6]	
SINUMERIK 840D sl	DB21,... .DBX8.5	

<Chan>.protAreas[6].out.chan	**PLC → NC**	**Bool**
激活通道专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.A_CSP[6]	
SINUMERIK 840D sl	DB21,... .DBX10.5	

<Chan>.protAreas[7].in.ncPreactivated	**NC → PLC**	**Bool**
机床专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPactivated[7]	
SINUMERIK 840D sl	DB21,... .DBX272.6	

<Chan>.protAreas[7].in.ncViolated	**NC → PLC**	**Bool**
进入了机床专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPViolated[7]	
SINUMERIK 840D sl	DB21,... .DBX276.6	

<Chan>.protAreas[7].in.chanPreactivated	**NC → PLC**	**Bool**
通道专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPactivated[7]	
SINUMERIK 840D sl	DB21,... .DBX274.6	

<Chan>.protAreas[7].in.chanViolated	**NC → PLC**	**Bool**
进入了通道专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPViolated[7]	
SINUMERIK 840D sl	DB21,... .DBX278.6	

<Chan>.protAreas[7].out.nc	**PLC → NC**	**Bool**
激活机床专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.A_MRP[7]	
SINUMERIK 840D sl	DB21,... .DBX8.6	

<Chan>.protAreas[7].out.chan	**PLC → NC**	**Bool**
激活通道专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.A_CSP[7]	
SINUMERIK 840D sl	DB21,... .DBX10.6	

<Chan>.protAreas[8].in.ncPreactivated	**NC → PLC**	**Bool**
机床专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPactivated[8]	
SINUMERIK 840D sl	DB21,... .DBX272.7	

<Chan>.protAreas[8].in.ncViolated	**NC → PLC**	**Bool**
进入了机床专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPViolated[8]	
SINUMERIK 840D sl	DB21,... .DBX276.7	

<Chan>.protAreas[8].in.chanPreactivated	**NC → PLC**	**Bool**
通道专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPactivated[8]	
SINUMERIK 840D sl	DB21,... .DBX274.7	

<Chan>.protAreas[8].in.chanViolated	**NC → PLC**	**Bool**
进入了通道专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPViolated[8]	
SINUMERIK 840D sl	DB21,... .DBX278.7	

<Chan>.protAreas[8].out.nc	**PLC → NC**	**Bool**
激活机床专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.A_MRP[8]	
SINUMERIK 840D sl	DB21,... .DBX8.7	

<Chan>.protAreas[8].out.chan	**PLC → NC**	**Bool**
激活通道专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.A_CSP[8]	
SINUMERIK 840D sl	DB21,... .DBX10.7	

<Chan>.protAreas[9].in.ncPreactivated	**NC → PLC**	**Bool**
机床专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPactivated[9]	
SINUMERIK 840D sl	DB21,... .DBX273.0	

<Chan>.protAreas[9].in.ncViolated	**NC → PLC**	**Bool**
进入了机床专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPViolated[9]	
SINUMERIK 840D sl	DB21,... .DBX277.0	

<Chan>.protAreas[9].in.chanPreactivated	**NC → PLC**	**Bool**
通道专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPactivated[9]	
SINUMERIK 840D sl	DB21,... .DBX275.0	

<Chan>.protAreas[9].in.chanViolated	**NC → PLC**	**Bool**
进入了通道专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPViolated[9]	
SINUMERIK 840D sl	DB21,... .DBX279.0	

<Chan>.protAreas[9].out.nc	**PLC → NC**	**Bool**
激活机床专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.A_MRP[9]	
SINUMERIK 840D sl	DB21,... .DBX9.0	

<Chan>.protAreas[9].out.chan	**PLC → NC**	**Bool**
激活通道专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.A_CSP[9]	
SINUMERIK 840D sl	DB21,... .DBX11.0	

<Chan>.protAreas[10].in.ncPreactivated	**NC → PLC**	**Bool**
机床专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPactivated[10]	
SINUMERIK 840D sl	DB21,... .DBX273.1	

<Chan>.protAreas[10].in.ncViolated	**NC → PLC**	**Bool**
进入了机床专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MRPViolated[10]	
SINUMERIK 840D sl	DB21,... .DBX277.1	

<Chan>.protAreas[10].in.chanPreactivated	**NC → PLC**	**Bool**
通道专用保护区已预激活		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPactivated[10]	
SINUMERIK 840D sl	DB21,... .DBX275.1	

<Chan>.protAreas[10].in.chanViolated	**NC → PLC**	**Bool**
进入了通道专用保护区		
SINUMERIK ONE Basic Program	LBP_Chan*.E_CSPViolated[10]	
SINUMERIK 840D sl	DB21,... .DBX279.1	

<Chan>.protAreas[10].out.nc		**PLC → NC**	**Bool**
激活机床专用保护区			
SINUMERIK ONE Basic Program	LBP_Chan*.A_MRP[10]		
SINUMERIK 840D sl	DB21,... .DBX9.1		

<Chan>.protAreas[10].out.chan		**PLC → NC**	**Bool**
激活通道专用保护区			
SINUMERIK ONE Basic Program	LBP_Chan*.A_CSP[10]		
SINUMERIK 840D sl	DB21,... .DBX11.1		

<Chan>.progEvent.in.ncStartActive		**NC → PLC**	**Bool**
PROG_EVENT: NC 从通道状态“复位”中启动			
SINUMERIK ONE Basic Program	LBP_Chan*.E_ProgEvent_Start		
SINUMERIK 840D sl	DB21,... .DBX376.0		

<Chan>.progEvent.in.partProgramEndActive		**NC → PLC**	**Bool**
PROG_EVENT: 程序结束复位(比如：M30)			
SINUMERIK ONE Basic Program	LBP_Chan*.E_ProgEvent_M30		
SINUMERIK 840D sl	DB21,... .DBX376.1		

<Chan>.progEvent.in.resetActive		**NC → PLC**	**Bool**
PROG_EVENT: 通道复位			
SINUMERIK ONE Basic Program	LBP_Chan*.E_ProgEvent_Reset		
SINUMERIK 840D sl	DB21,... .DBX376.2		

<Chan>.progEvent.in.powerOnActive		**NC → PLC**	**Bool**
PROG_EVENT: 上电			
SINUMERIK ONE Basic Program	LBP_Chan*.E_ProgEvent_PowerOn		
SINUMERIK 840D sl	DB21,... .DBX376.3		

<Chan>.progEvent.in.blockSearchActive		**NC → PLC**	**Bool**
PROG_EVENT: 程序段搜索后的第 1. 次 NC 启动			
SINUMERIK ONE Basic Program	LBP_Chan*.E_ProgEvent_SearchRun		
SINUMERIK 840D sl	DB21,... .DBX376.4		

<Chan>.synActs[1].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[1] - [64]	
SINUMERIK 840D sl	DB21,... .DBX308.0..315.7	

<Chan>.synActs[1].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[1] - [64]	
SINUMERIK 840D sl	DB21,... .DBX300.0..307.7	

<Chan>.synActs[2].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[2]	
SINUMERIK 840D sl	DB21,... .DBX308.1	

<Chan>.synActs[2].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[2]	
SINUMERIK 840D sl	DB21,... .DBX300.1	

<Chan>.synActs[3].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[3]	
SINUMERIK 840D sl	DB21,... .DBX308.2	

<Chan>.synActs[3].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[3]	
SINUMERIK 840D sl	DB21,... .DBX300.2	

<Chan>.synActs[4].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[4]	
SINUMERIK 840D sl	DB21,... .DBX308.3	

<Chan>.synActs[4].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[4]	
SINUMERIK 840D sl	DB21,... .DBX300.3	

<Chan>.synActs[5].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[5]	
SINUMERIK 840D sl	DB21,... .DBX308.4	

<Chan>.synActs[5].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[5]	
SINUMERIK 840D sl	DB21,... .DBX300.4	

<Chan>.synActs[6].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[6]	
SINUMERIK 840D sl	DB21,... .DBX308.5	

<Chan>.synActs[6].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[6]	
SINUMERIK 840D sl	DB21,... .DBX300.5	

<Chan>.synActs[7].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[7]	
SINUMERIK 840D sl	DB21,... .DBX308.6	

<Chan>.synActs[7].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[7]	
SINUMERIK 840D sl	DB21,... .DBX300.6	

<Chan>.synActs[8].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[8]	
SINUMERIK 840D sl	DB21,... .DBX308.7	

<Chan>.synActs[8].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[8]	
SINUMERIK 840D sl	DB21,... .DBX300.7	

<Chan>.synActs[9].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[9]	
SINUMERIK 840D sl	DB21,... .DBX309.0	

<Chan>.synActs[9].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[9]	
SINUMERIK 840D sl	DB21,... .DBX301.0	

<Chan>.synActs[10].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[10]	
SINUMERIK 840D sl	DB21,... .DBX309.1	

<Chan>.synActs[10].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[10]	
SINUMERIK 840D sl	DB21,... .DBX301.1	

<Chan>.synActs[11].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[11]	
SINUMERIK 840D sl	DB21,... .DBX309.2	

<Chan>.synActs[11].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[11]	
SINUMERIK 840D sl	DB21,... .DBX301.2	

<Chan>.synActs[12].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[12]	
SINUMERIK 840D sl	DB21,... .DBX309.3	

<Chan>.synActs[12].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[12]	
SINUMERIK 840D sl	DB21,... .DBX301.3	

<Chan>.synActs[13].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[13]	
SINUMERIK 840D sl	DB21,... .DBX309.4	

<Chan>.synActs[13].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[13]	
SINUMERIK 840D sl	DB21,... .DBX301.4	

<Chan>.synActs[14].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[14]	
SINUMERIK 840D sl	DB21,... .DBX309.5	

<Chan>.synActs[14].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[14]	
SINUMERIK 840D sl	DB21,... .DBX301.5	

<Chan>.synActs[15].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[15]	
SINUMERIK 840D sl	DB21,... .DBX309.6	

<Chan>.synActs[15].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[15]	
SINUMERIK 840D sl	DB21,... .DBX301.6	

<Chan>.synActs[16].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[16]	
SINUMERIK 840D sl	DB21,... .DBX309.7	

<Chan>.synActs[16].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[16]	
SINUMERIK 840D sl	DB21,... .DBX301.7	

<Chan>.synActs[17].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[17]	
SINUMERIK 840D sl	DB21,... .DBX310.0	

<Chan>.synActs[17].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[17]	
SINUMERIK 840D sl	DB21,... .DBX302.0	

<Chan>.synActs[18].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[18]	
SINUMERIK 840D sl	DB21,... .DBX310.1	

<Chan>.synActs[18].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[18]	
SINUMERIK 840D sl	DB21,... .DBX302.1	

<Chan>.synActs[19].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[19]	
SINUMERIK 840D sl	DB21,... .DBX310.2	

<Chan>.synActs[19].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[19]	
SINUMERIK 840D sl	DB21,... .DBX302.2	

<Chan>.synActs[20].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[20]	
SINUMERIK 840D sl	DB21,... .DBX310.3	

<Chan>.synActs[20].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[20]	
SINUMERIK 840D sl	DB21,... .DBX302.3	

<Chan>.synActs[21].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[21]	
SINUMERIK 840D sl	DB21,... .DBX310.4	

<Chan>.synActs[21].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[21]	
SINUMERIK 840D sl	DB21,... .DBX302.4	

<Chan>.synActs[22].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[22]	
SINUMERIK 840D sl	DB21,... .DBX310.5	

<Chan>.synActs[22].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[22]	
SINUMERIK 840D sl	DB21,... .DBX302.5	

<Chan>.synActs[23].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[23]	
SINUMERIK 840D sl	DB21,... .DBX310.6	

<Chan>.synActs[23].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[23]	
SINUMERIK 840D sl	DB21,... .DBX302.6	

<Chan>.synActs[24].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[24]	
SINUMERIK 840D sl	DB21,... .DBX310.7	

<Chan>.synActs[24].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[24]	
SINUMERIK 840D sl	DB21,... .DBX302.7	

<Chan>.synActs[25].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[25]	
SINUMERIK 840D sl	DB21,... .DBX311.0	

<Chan>.synActs[25].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[25]	
SINUMERIK 840D sl	DB21,... .DBX303.0	

<Chan>.synActs[26].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[26]	
SINUMERIK 840D sl	DB21,... .DBX311.1	

<Chan>.synActs[26].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[26]	
SINUMERIK 840D sl	DB21,... .DBX303.1	

<Chan>.synActs[27].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[27]	
SINUMERIK 840D sl	DB21,... .DBX311.2	

<Chan>.synActs[27].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[27]	
SINUMERIK 840D sl	DB21,... .DBX303.2	

<Chan>.synActs[28].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[28]	
SINUMERIK 840D sl	DB21,... .DBX311.3	

<Chan>.synActs[28].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[28]	
SINUMERIK 840D sl	DB21,... .DBX303.3	

<Chan>.synActs[29].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[29]	
SINUMERIK 840D sl	DB21,... .DBX311.4	

<Chan>.synActs[29].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[29]	
SINUMERIK 840D sl	DB21,... .DBX303.4	

<Chan>.synActs[30].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[30]	
SINUMERIK 840D sl	DB21,... .DBX311.5	

<Chan>.synActs[30].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[30]	
SINUMERIK 840D sl	DB21,... .DBX303.5	

<Chan>.synActs[31].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[31]	
SINUMERIK 840D sl	DB21,... .DBX311.6	

<Chan>.synActs[31].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[31]	
SINUMERIK 840D sl	DB21,... .DBX303.6	

<Chan>.synActs[32].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[32]	
SINUMERIK 840D sl	DB21,... .DBX311.7	

<Chan>.synActs[32].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[32]	
SINUMERIK 840D sl	DB21,... .DBX303.7	

<Chan>.synActs[33].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[33]	
SINUMERIK 840D sl	DB21,... .DBX312.0	

<Chan>.synActs[33].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[33]	
SINUMERIK 840D sl	DB21,... .DBX304.0	

<Chan>.synActs[34].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[34]	
SINUMERIK 840D sl	DB21,... .DBX312.1	

<Chan>.synActs[34].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[34]	
SINUMERIK 840D sl	DB21,... .DBX304.1	

<Chan>.synActs[35].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[35]	
SINUMERIK 840D sl	DB21,... .DBX312.2	

<Chan>.synActs[35].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[35]	
SINUMERIK 840D sl	DB21,... .DBX304.2	

<Chan>.synActs[36].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[36]	
SINUMERIK 840D sl	DB21,... .DBX312.3	

<Chan>.synActs[36].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[36]	
SINUMERIK 840D sl	DB21,... .DBX304.3	

<Chan>.synActs[37].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[37]	
SINUMERIK 840D sl	DB21,... .DBX312.4	

<Chan>.synActs[37].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[37]	
SINUMERIK 840D sl	DB21,... .DBX304.4	

<Chan>.synActs[38].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[38]	
SINUMERIK 840D sl	DB21,... .DBX312.5	

<Chan>.synActs[38].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[38]	
SINUMERIK 840D sl	DB21,... .DBX304.5	

<Chan>.synActs[39].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[39]	
SINUMERIK 840D sl	DB21,... .DBX312.6	

<Chan>.synActs[39].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[39]	
SINUMERIK 840D sl	DB21,... .DBX304.6	

<Chan>.synActs[40].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[40]	
SINUMERIK 840D sl	DB21,... .DBX312.7	

<Chan>.synActs[40].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[40]	
SINUMERIK 840D sl	DB21,... .DBX304.7	

<Chan>.synActs[41].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[41]	
SINUMERIK 840D sl	DB21,... .DBX313.0	

<Chan>.synActs[41].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[41]	
SINUMERIK 840D sl	DB21,... .DBX305.0	

<Chan>.synActs[42].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[42]	
SINUMERIK 840D sl	DB21,... .DBX313.1	

<Chan>.synActs[42].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[42]	
SINUMERIK 840D sl	DB21,... .DBX305.1	

<Chan>.synActs[43].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[43]	
SINUMERIK 840D sl	DB21,... .DBX313.2	

<Chan>.synActs[43].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[43]	
SINUMERIK 840D sl	DB21,... .DBX305.2	

<Chan>.synActs[44].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[44]	
SINUMERIK 840D sl	DB21,... .DBX313.3	

<Chan>.synActs[44].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[44]	
SINUMERIK 840D sl	DB21,... .DBX305.3	

<Chan>.synActs[45].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[45]	
SINUMERIK 840D sl	DB21,... .DBX313.4	

<Chan>.synActs[45].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[45]	
SINUMERIK 840D sl	DB21,... .DBX305.4	

<Chan>.synActs[46].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[46]	
SINUMERIK 840D sl	DB21,... .DBX313.5	

<Chan>.synActs[46].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[46]	
SINUMERIK 840D sl	DB21,... .DBX305.5	

<Chan>.synActs[47].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[47]	
SINUMERIK 840D sl	DB21,... .DBX313.6	

<Chan>.synActs[47].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[47]	
SINUMERIK 840D sl	DB21,... .DBX305.6	

<Chan>.synActs[48].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[48]	
SINUMERIK 840D sl	DB21,... .DBX313.7	

<Chan>.synActs[48].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[48]	
SINUMERIK 840D sl	DB21,... .DBX305.7	

<Chan>.synActs[49].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[49]	
SINUMERIK 840D sl	DB21,... .DBX314.0	

<Chan>.synActs[49].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[49]	
SINUMERIK 840D sl	DB21,... .DBX306.0	

<Chan>.synActs[50].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[50]	
SINUMERIK 840D sl	DB21,... .DBX314.1	

<Chan>.synActs[50].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[50]	
SINUMERIK 840D sl	DB21,... .DBX306.1	

<Chan>.synActs[51].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[51]	
SINUMERIK 840D sl	DB21,... .DBX314.2	

<Chan>.synActs[51].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[51]	
SINUMERIK 840D sl	DB21,... .DBX306.2	

<Chan>.synActs[52].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[52]	
SINUMERIK 840D sl	DB21,... .DBX314.3	

<Chan>.synActs[52].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[52]	
SINUMERIK 840D sl	DB21,... .DBX306.3	

<Chan>.synActs[53].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[53]	
SINUMERIK 840D sl	DB21,... .DBX314.4	

<Chan>.synActs[53].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[53]	
SINUMERIK 840D sl	DB21,... .DBX306.4	

<Chan>.synActs[54].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[54]	
SINUMERIK 840D sl	DB21,... .DBX314.5	

<Chan>.synActs[54].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[54]	
SINUMERIK 840D sl	DB21,... .DBX306.5	

<Chan>.synActs[55].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[55]	
SINUMERIK 840D sl	DB21,... .DBX314.6	

<Chan>.synActs[55].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[55]	
SINUMERIK 840D sl	DB21,... .DBX306.6	

<Chan>.synActs[56].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[56]	
SINUMERIK 840D sl	DB21,... .DBX314.7	

<Chan>.synActs[56].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[56]	
SINUMERIK 840D sl	DB21,... .DBX306.7	

<Chan>.synActs[57].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[57]	
SINUMERIK 840D sl	DB21,... .DBX315.0	

<Chan>.synActs[57].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[57]	
SINUMERIK 840D sl	DB21,... .DBX307.0	

<Chan>.synActs[58].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[58]	
SINUMERIK 840D sl	DB21,... .DBX315.1	

<Chan>.synActs[58].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[58]	
SINUMERIK 840D sl	DB21,... .DBX307.1	

<Chan>.synActs[59].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[59]	
SINUMERIK 840D sl	DB21,... .DBX315.2	

<Chan>.synActs[59].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[59]	
SINUMERIK 840D sl	DB21,... .DBX307.2	

<Chan>.synActs[60].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[60]	
SINUMERIK 840D sl	DB21,... .DBX315.3	

<Chan>.synActs[60].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[60]	
SINUMERIK 840D sl	DB21,... .DBX307.3	

<Chan>.synActs[61].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[61]	
SINUMERIK 840D sl	DB21,... .DBX315.4	

<Chan>.synActs[61].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[61]	
SINUMERIK 840D sl	DB21,... .DBX307.4	

<Chan>.synActs[62].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[62]	
SINUMERIK 840D sl	DB21,... .DBX315.5	

<Chan>.synActs[62].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[62]	
SINUMERIK 840D sl	DB21,... .DBX307.5	

<Chan>.synActs[63].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[63]	
SINUMERIK 840D sl	DB21,... .DBX315.6	

<Chan>.synActs[63].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[63]	
SINUMERIK 840D sl	DB21,... .DBX307.6	

<Chan>.synActs[64].in.lockable	**NC → PLC**	**Bool**
同步动作 ID/IDS 可以禁止，不受 NCK 保护(MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_SynDisabled[64]	
SINUMERIK 840D sl	DB21,... .DBX315.7	

<Chan>.synActs[64].out.lock	**PLC → NC**	**Bool**
禁止同步动作 ID/IDS		
SINUMERIK ONE Basic Program	LBP_Chan*.A_SynDisabl[64]	
SINUMERIK 840D sl	DB21,... .DBX307.7	

<Chan>.grinding.in.signalsEnabled	**NC → PLC**	**Bool**
磨削工艺：PLC 和 NCK 的输入信号已启用($AC_IN_KEY_G_ISENABLE[])		
SINUMERIK ONE Basic Program	LBP_Chan*.E_InKeyGIsEn1-8	
SINUMERIK 840D sl	DB21,... .DBX390.0..7	

<Chan>.grinding.in.requested	**NC → PLC**	**Bool**
NCK 启动的某操作请求启用磨削信号($AC_IN_KEY_G_RUN_OUT[])		
SINUMERIK ONE Basic Program	LBP_Chan*.E_InKeyGRunOut1-8	
SINUMERIK 840D sl	DB21,... .DBX391.0..7	

<Chan>.grinding.out.set	**PLC → NC**	**Bool**
启用后的 PLC 输入信号($AC_IN_KEY_G[])		
SINUMERIK ONE Basic Program	LBP_Chan*.A_InKeyG<n>	
SINUMERIK 840D sl	DB21,... .DBX385.0..7	

<Chan>.grinding.out.disableSignals	**PLC → NC**	**Bool**
禁用磨削信号		
SINUMERIK ONE Basic Program	LBP_Chan*.A_InKeyGEn<n>	
SINUMERIK 840D sl	DB21,... .DBX386.0..7	

<Chan>.grinding.out.ackFuncsExecution	**PLC → NC**	**Bool**
应答磨削功能的执行 ($AC_IN_KEY_G_RUN_IN[])		
SINUMERIK ONE Basic Program	LBP_Chan*.A_InKeyGRunIn<n>	
SINUMERIK 840D sl	DB21,... .DBX387.0..7	

<Chan>.nibbelingAndPunching.in.strokeInitiationActive	**NC → PLC**	**Bool**
冲程释放已生效		
SINUMERIK ONE Basic Program	LBP_Chan*.E_StrokeEA	
SINUMERIK 840D sl	DB21,... .DBX38.0	

<Chan>.nibbelingAndPunching.in.strokeManualInitiationAck	**NC → PLC**	**Bool**
手动冲压已释放		
SINUMERIK ONE Basic Program	LBP_Chan*.E_AcknManStrokEnab	
SINUMERIK 840D sl	DB21,... .DBX38.1	

<Chan>.nibbelingAndPunching.out.enableStroke	**PLC → NC**	**Bool**
激活冲程的执行		
SINUMERIK ONE Basic Program	LBP_Chan*.A_StrokEnab	
SINUMERIK 840D sl	DB21,... .DBX3.0	

<Chan>.nibbelingAndPunching.out.suppressStroke	**PLC → NC**	**Bool**
激活“冲程封”		
SINUMERIK ONE Basic Program	LBP_Chan*.A_StrokSup	
SINUMERIK 840D sl	DB21,... .DBX3.2	

<Chan>.nibbelingAndPunching.out.delayStroke	**PLC → NC**	**Bool**
激活选项“延时冲程” (MD26018 $MC_NIBPUNCH_PRE_START_TIME; SD42402 $SC_NIB_PRE_START_TIME)		
SINUMERIK ONE Basic Program	LBP_Chan*.A_StrokDelayed	
SINUMERIK 840D sl	DB21,... .DBX3.3	

<Chan>.nibbelingAndPunching.out.noStroke	**PLC → NC**	**Bool**
冲程未生效		
SINUMERIK ONE Basic Program	LBP_Chan*.A_StrokStop	
SINUMERIK 840D sl	DB21,... .DBX3.4	

<Chan>.nibbelingAndPunching.out.manualStrokeInterface1	**PLC → NC**	**Bool**
在接口 1 上执行手动冲压		
SINUMERIK ONE Basic Program	LBP_Chan*.A_ManStrokEnab	
SINUMERIK 840D sl	DB21,... .DBX3.1	

<Chan>.nibbelingAndPunching.out.manualStrokeInterface2	**PLC → NC**	**Bool**
在接口 2 上执行手动冲压		
SINUMERIK ONE Basic Program	LBP_Chan*.A_ManRelStroke2	
SINUMERIK 840D sl	DB21,... .DBX3.5	

<Chan>.cycles.in.signals	**NC → PLC**	**Bool**
收到的循环信号		
SINUMERIK ONE Basic Program	LBP_Chan*.E_Cycle_Sig	
SINUMERIK 840D sl	DB21,... .DBX364.0..365.7	

<Chan>.cycles.out.signals	**PLC → NC**	**Bool**
发送的循环信号		
SINUMERIK ONE Basic Program	LBP_Chan*.A_Cycle_Sig	
SINUMERIK 840D sl	DB21,... .DBX366.0..367.7	

12.2.4 NCK_typeModeGroup

<ModeGroup>.config
运行方式组的配置数据

<ModeGroup>.basic
运行方式组的通用信号

<ModeGroup>.basic.in	**NC → PLC**
从运行方式组收到的通用信号	

<ModeGroup>.basic.out	**PLC → NC**
发送给运行方式组的通用信号	

<ModeGroup>.config.registered	**NC → PLC**	**Bool**
运行方式组已注册		

<ModeGroup>.config.modeGroupActive	**NC → PLC**	**Bool**
运行方式组已生效		

<ModeGroup>.config.modeGroupNumber	**NC → PLC**	**USInt**
运行方式组编号		

<ModeGroup>.basic.in.ready	**NC → PLC**	**Bool**
运行方式组已就绪		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_MGOK	
SINUMERIK 840D sl	DB11.DBX6.3	

<ModeGroup>.basic.in.resetDone	**NC → PLC**	**Bool**
运行方式组的复位已执行		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_MGreseted	
SINUMERIK 840D sl	DB11.DBX6.4	

<ModeGroup>.basic.in.resetAllChanDone	**NC → PLC**	**Bool**
该运行方式组内的所有通道都处于“复位”状态		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_ChanReset	
SINUMERIK 840D sl	DB11.DBX6.7	

<ModeGroup>.basic.in.autoModeActive	**NC → PLC**	**Bool**
自动运行方式已生效		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_AUTO	
SINUMERIK 840D sl	DB11.DBX6.0	

<ModeGroup>.basic.in.mdaModeActive	**NC → PLC**	**Bool**
MDI 运行方式已生效		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_MDA	
SINUMERIK 840D sl	DB11.DBX6.1	

<ModeGroup>.basic.in.jogModeActive	**NC → PLC**	**Bool**
JOG 运行方式已生效		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_JOG	
SINUMERIK 840D sl	DB11.DBX6.2	

<ModeGroup>.basic.in.nckInternalJogActive	**NC → PLC**	**Bool**
NCK 内部 JOG 方式已生效		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_NCKintJOG	
SINUMERIK 840D sl	DB11.DBX6.5	

<ModeGroup>.basic.in.teachInActive	**NC → PLC**	**Bool**
机床功能 TEACH IN 已生效		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_TEACHIN	
SINUMERIK 840D sl	DB11.DBX7.0	

<ModeGroup>.basic.in.reposActive	**NC → PLC**	**Bool**
机床功能 REPOS 已生效		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_REPOS	
SINUMERIK 840D sl	DB11.DBX7.1	

<ModeGroup>.basic.in.refActive	**NC → PLC**	**Bool**
机床功能 REF 已生效		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_REF	
SINUMERIK 840D sl	DB11.DBX7.2	

<ModeGroup>.basic.in.inc1Active	**NC → PLC**	**Bool**
“根据请求移动 1 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_INC1	
SINUMERIK 840D sl	DB11.DBX8.0	

<ModeGroup>.basic.in.inc10Active	**NC → PLC**	**Bool**
“根据请求移动 10 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_INC10	
SINUMERIK 840D sl	DB11.DBX8.1	

<ModeGroup>.basic.in.inc100Active	**NC → PLC**	**Bool**
“根据请求移动 100 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_INC100	
SINUMERIK 840D sl	DB11.DBX8.2	

<ModeGroup>.basic.in.inc1000Active	**NC → PLC**	**Bool**
“根据请求移动 1 000 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_INC1000	
SINUMERIK 840D sl	DB11.DBX8.3	

<ModeGroup>.basic.in.inc10000Active	**NC → PLC**	**Bool**
“根据请求移动 10 000 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_INC10000	
SINUMERIK 840D sl	DB11.DBX8.4	

<ModeGroup>.basic.in.incVarActive	**NC → PLC**	**Bool**
“根据请求移动用户自定义数量的增量”已生效		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_INCVar	
SINUMERIK 840D sl	DB11.DBX8.5	

<ModeGroup>.basic.in.contTraversActive	**NC → PLC**	**Bool**
持续移动（缺省）已生效		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_ContManTravelAct	
SINUMERIK 840D sl	DB11.DBX8.6	

<ModeGroup>.basic.out.reset	**PLC → NC**	**Bool**
请求复位运行方式组		
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_MGReset	
SINUMERIK 840D sl	DB11.DBX0.7	

<ModeGroup>.basic.out.stopAll	**PLC → NC**	**Bool**
停止运行方式组（进给轴和主轴）		
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_MGStopASp	
SINUMERIK 840D sl	DB11.DBX0.6	

<ModeGroup>.basic.out.stopAxesOnly	**PLC → NC**	**Bool**
请求停止运行方式组（仅进给轴）		
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_MGStop	
SINUMERIK 840D sl	DB11.DBX0.5	

<ModeGroup>.basic.out.disableModeChangeover	**PLC → NC**	**Bool**
一个运行方式组内的运行方式切换已禁止		
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_MCDisable	
SINUMERIK 840D sl	DB11.DBX0.4	

<ModeGroup>.basic.out.autoMode	**PLC → NC**	**Bool**
请求切换到自动方式		
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_AUTO	
SINUMERIK 840D sl	DB11.DBX0.0	

<ModeGroup>.basic.out.mdaMode	**PLC → NC**	**Bool**
请求切换到 MDI 方式		
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_MDA	
SINUMERIK 840D sl	DB11.DBX0.1	

<ModeGroup>.basic.out.jogMode	**PLC → NC**	**Bool**
请求切换到 JOG 方式		
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_JOG	
SINUMERIK 840D sl	DB11.DBX0.2	

<ModeGroup>.basic.out.teachIn	**PLC → NC**	**Bool**
请求机床功能 TEACH IN		
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_TEACHIN	
SINUMERIK 840D sl	DB11.DBX1.0	

<ModeGroup>.basic.out.repos	**PLC → NC**	**Bool**
请求机床功能 REPOS		
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_REPOS	
SINUMERIK 840D sl	DB11.DBX1.1	

<ModeGroup>.basic.out.refPoint		**PLC → NC**	**Bool**
请求机床功能 REF			
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_REF		
SINUMERIK 840D sl	DB11.DBX1.2		

<ModeGroup>.basic.out.singleBlockTypeA		**PLC → NC**	**Bool**
选择"A 型单步执行"运行方式			
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_SingleBlock_A		
SINUMERIK 840D sl	DB11.DBX1.7		

<ModeGroup>.basic.out.singleBlockTypeB		**PLC → NC**	**Bool**
选择"B 型单步执行"运行方式			
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_SingleBlock_B		
SINUMERIK 840D sl	DB11.DBX1.6		

<ModeGroup>.basic.out.inc1		**PLC → NC**	**Bool**
激活"根据请求移动 1 个增量"			
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_INC1		
SINUMERIK 840D sl	DB11.DBX2.0		

<ModeGroup>.basic.out.inc10		**PLC → NC**	**Bool**
"根据请求移动 10 个增量"已生效			
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_INC10		
SINUMERIK 840D sl	DB11.DBX2.1		

<ModeGroup>.basic.out.inc100		**PLC → NC**	**Bool**
"根据请求移动 100 个增量"已生效			
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_INC100		
SINUMERIK 840D sl	DB11.DBX2.2		

<ModeGroup>.basic.out.inc1000		**PLC → NC**	**Bool**
"根据请求移动 1 000 个增量"已生效			
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_INC1000		
SINUMERIK 840D sl	DB11.DBX2.3		

<ModeGroup>.basic.out.inc10000	**PLC → NC**	**Bool**
“根据请求移动 10 000 个增量”已生效		
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_INC10000	
SINUMERIK 840D sl	DB11.DBX2.4	

<ModeGroup>.basic.out.incVar	**PLC → NC**	**Bool**
激活“根据请求移动用户自定义数量的增量”		
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_INCVar	
SINUMERIK 840D sl	DB11.DBX2.5	

<ModeGroup>.basic.out.contTravers	**PLC → NC**	**Bool**
激活持续移动（默认）		
SINUMERIK ONE Basic Program	LBP_ModeGroup.A_ContManTravel	
SINUMERIK 840D sl	DB11.DBX2.6	

12.2.5 NCK_typeNC

<Nc>.config
NC 配置数据

<Nc>.basic
NC 的通用信号

<Nc>.handwheels
手轮信号

<Nc>.digitalInputs
NC 数字量输入信号

<Nc>.digitalOutputs
NC 数字量输出信号

<Nc>.analogInputs
NC 模拟量输入信号

<Nc>.analogOutputs	
NC 模拟量输出信号	
<Nc>.collisionProtAreas	
保护区信号 [0..63]	
<Nc>.collision	
碰撞检测信号	
<Nc>.basic.in	**NC → PLC**
从 NCK 收到的通用信号	
<Nc>.basic.out	**PLC → NC**
发送给 NCK 的通用信号	
<Nc>.digitalInputs.in	**NC → PLC**
收到的 NC 硬件数字量输入信号的状态信号	
<Nc>.digitalInputs.out	**PLC → NC**
发送的 NC 软件数字量输入值的控制信号	
<Nc>.digitalOutputs.in	**NC → PLC**
收到的 NC 软件数字量输出信号的状态信号	
<Nc>.digitalOutputs.out	**PLC → NC**
发送的 NC 硬件数字量输出信号的控制信号	
<Nc>.analogInputs.in	**NC → PLC**
收到的 NC 硬件模拟量输入值	
<Nc>.analogInputs.out	**PLC → NC**
发送的软件模拟量输入值的控制信号	
<Nc>.analogOutputs.in	**NC → PLC**
收到的 NC 软件模拟量输出值	

<Nc>.analogOutputs.out	**PLC → NC**	
发送的 NC 硬件模拟量输出值的控制信号		

<Nc>.handwheels.in	**NC → PLC**	
从手轮收到的信号 (1..6)		

<Nc>.collisionProtAreas.in	**NC → PLC**	
收到的保护区状态信号[0..63]		

<Nc>.collisionProtAreas.out	**PLC → NC**	
发送的保护区控制信号[0..63]		

<Nc>.collision.out	**PLC → NC**	
发送的碰撞检测信号		

<Nc>.config.registered	**NC → PLC**	**Bool**
NC 已注册		

<Nc>.config.isVirtual	**NC → PLC**	**Bool**
NC 是虚拟系统（数字化双胞胎）		

<Nc>.config.maxModeGroup	**NC → PLC**	**UInt**
生效的运行方式组的数量		

<Nc>.config.maxChan	**NC → PLC**	**UInt**
生效的最高通道号		

<Nc>.config.maxAxis	**NC → PLC**	**UInt**
生效的最高轴号		

<Nc>.config.activeModeGroups	**NC → PLC**	**Bool**
生效的运行方式组		

<Nc>.config.activeChans	**NC → PLC**	**Bool**
生效的通道		

<Nc>.config.activeAxes	**NC → PLC**	**Bool**
生效的轴		

<Nc>.config.ncCycleTimeout	**NC → PLC**	**Time**
NC 循环寿命监控时间		

<Nc>.config.ncRunupTimeout	**NC → PLC**	**Time**
NC 启动监控时间		

<Nc>.basic.in.emergencyStopActive	**NC → PLC**	**Bool**
急停已生效		
SINUMERIK ONE Basic Program	LBP_NC.E_EMERGENCY	
SINUMERIK 840D sl	DB10.DBX106.1	

<Nc>.basic.in.measuringProbe1Actuated	**NC → PLC**	**Bool**
测头 1 已动作		
SINUMERIK ONE Basic Program	LBP_NC.E_InspProbe1	
SINUMERIK 840D sl	DB10.DBX107.0	

<Nc>.basic.in.measuringProbe2Actuated	**NC → PLC**	**Bool**
测头 2 已动作		
SINUMERIK ONE Basic Program	LBP_NC.E_InspProbe2	
SINUMERIK 840D sl	DB10.DBX107.1	

<Nc>.basic.in.ncuLinkActive	**NC → PLC**	**Bool**
NCU-Link 通讯已生效		
SINUMERIK ONE Basic Program	LBP_NC.E_NCU_LinkActive	
SINUMERIK 840D sl	DB10.DBX107.6	

<Nc>.basic.in.imperialUnitsActive	**NC → PLC**	**Bool**
所有目标值和实际值都作为英制单位值计算或使用。		
SINUMERIK ONE Basic Program	LBP_NC.E_SystemInchDim	
SINUMERIK 840D sl	DB10.DBX107.7	

<Nc>.basic.in.imperialMetricChangeCounter	**NC → PLC**	**USInt**
英制/公制单位切换计数器		
SINUMERIK ONE Basic Program	LBP_NC.E_CounterInchMetr	
SINUMERIK 840D sl	DB10.DBB71	

<Nc>.basic.in.sinuOperateReady	**NC → PLC**	**Bool**
SINUMERIK Operate 运行就绪，和 NC 循环通讯		
SINUMERIK ONE Basic Program	LBP_NC.E_MMCBTSSready	
SINUMERIK 840D sl	DB10.DBX108.3	

<Nc>.basic.in.drivesCyclicOperation	**NC → PLC**	**Bool**
NC 所有机床轴对应的驱动都已进入循环运行，即：驱动和 NC 循环地交换 PROFIdrive 报文。		
SINUMERIK ONE Basic Program	LBP_NC.E_DrivesInCylOp	
SINUMERIK 840D sl	DB10.DBX108.5	

<Nc>.basic.in.drivesReady	**NC → PLC**	**Bool**
NC 所有机床轴对应的驱动都已就绪		
SINUMERIK ONE Basic Program	LBP_NC.E_611Dready	
SINUMERIK 840D sl	DB10.DBX108.6	

<Nc>.basic.in.ncReady	**NC → PLC**	**Bool**
NC 就绪：控制系统进入循环运行		
SINUMERIK ONE Basic Program	LBP_NC.E_NCready	
SINUMERIK 840D sl	DB10.DBX108.7	

<Nc>.basic.in.ncAlarmActive	**NC → PLC**	**Bool**
目前至少有一条 NC 报警有待处理		
SINUMERIK ONE Basic Program	LBP_NC.E_NCKalarm	
SINUMERIK 840D sl	DB10.DBX109.0	

<Nc>.basic.in.ncuHeatSinkAlarmActive	**NC → PLC**	**Bool**
超出了 NCU 散热器温度限值，无法保证 NCU 可以持续运行。		
SINUMERIK ONE Basic Program	LBP_NC.E_HTempAlarm	
SINUMERIK 840D sl	DB10.DBX109.5	

<Nc>.basic.in.ncuAirTemperatureAlarmActive	**NC → PLC**	**Bool**
NC 环境温度监控和/或风扇监控已响应		
SINUMERIK ONE Basic Program	LBP_NC.E_ATempAlarm	
SINUMERIK 840D sl	DB10.DBX109.6	

<Nc>.basic.in.ncuBatteryAlarmActive	**NC → PLC**	**Bool**
NC 电池电压监控功能已响应		
SINUMERIK ONE Basic Program	LBP_NC.E_BattAlarm	
SINUMERIK 840D sl	DB10.DBX109.7	

<Nc>.basic.in.collisionMonitoringActive	**NC → PLC**	**Bool**
碰撞检测：保护区监控已生效		
SINUMERIK ONE Basic Program	LBP_NC.E_CollCheckOff	
SINUMERIK 840D sl	DB10.DBX106.0	

<Nc>.basic.in.swCams	**NC → PLC**
线性轴 - 当轴在同一个方向上越过挡块时，挡块信号从 0 变为 1；模数旋转轴 - 当轴正向移动，越过负向挡块时，正向挡块信号从 0 变为 1，作为对该上升沿的响应，负向挡块信号也相应变化。	
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus
SINUMERIK 840D sl	DB10.DBX110.0

<Nc>.basic.in.internalAsupModeGroupNumber	**OP → PLC**	**USInt**
仅内部使用		
SINUMERIK 840D sl	DB19.DBB40	

<Nc>.basic.in.internalAsupChanNumber	**OP → PLC**	**USInt**
仅内部使用		
SINUMERIK 840D sl	DB19.DBB41	

<Nc>.basic.in.internalAsup9StartReq	**OP → PLC**	**Bool**
仅内部使用		
SINUMERIK 840D sl	DB19.DBX42.0	

<Nc>.basic.in.internalAsup10StartReq	**OP → PLC**	**Bool**
仅内部使用		
SINUMERIK 840D sl	DB19.DBX42.1	

<Nc>.basic.in.swCams[1].minus	**NC → PLC**	**Bool**
直线轴负向移动，越过负向挡块 0（和模数旋转轴的方式不同）		
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[0]	
SINUMERIK 840D sl	DB10.DBX110.0	

<Nc>.basic.in.swCams[1].plus	**NC → PLC**	**Bool**
直线轴正向移动，越过正向挡块 0（和模数旋转轴的方式不同）		
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[0]	
SINUMERIK 840D sl	DB10.DBX114.0	

<Nc>.basic.in.swCams[2].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[1]	
SINUMERIK 840D sl	DB10.DBX110.1	

<Nc>.basic.in.swCams[2].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[1]	
SINUMERIK 840D sl	DB10.DBX114.1	

<Nc>.basic.in.swCams[3].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[2]	
SINUMERIK 840D sl	DB10.DBX110.2	

<Nc>.basic.in.swCams[3].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[2]	
SINUMERIK 840D sl	DB10.DBX114.2	

<Nc>.basic.in.swCams[4].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[3]	
SINUMERIK 840D sl	DB10.DBX110.3	

<Nc>.basic.in.swCams[4].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[3]	
SINUMERIK 840D sl	DB10.DBX114.3	

<Nc>.basic.in.swCams[5].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[4]	
SINUMERIK 840D sl	DB10.DBX110.4	

<Nc>.basic.in.swCams[5].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[4]	
SINUMERIK 840D sl	DB10.DBX114.4	

<Nc>.basic.in.swCams[6].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[5]	
SINUMERIK 840D sl	DB10.DBX110.5	

<Nc>.basic.in.swCams[6].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[5]	
SINUMERIK 840D sl	DB10.DBX114.5	

<Nc>.basic.in.swCams[7].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[6]	
SINUMERIK 840D sl	DB10.DBX110.6	

<Nc>.basic.in.swCams[7].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[6]	
SINUMERIK 840D sl	DB10.DBX114.6	

<Nc>.basic.in.swCams[8].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[7]	
SINUMERIK 840D sl	DB10.DBX110.7	

<Nc>.basic.in.swCams[8].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[7]	
SINUMERIK 840D sl	DB10.DBX114.7	

<Nc>.basic.in.swCams[9].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[8]	
SINUMERIK 840D sl	DB10.DBX111.0	

<Nc>.basic.in.swCams[9].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[8]	
SINUMERIK 840D sl	DB10.DBX115.0	

<Nc>.basic.in.swCams[10].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[9]	
SINUMERIK 840D sl	DB10.DBX111.1	

<Nc>.basic.in.swCams[10].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[9]	
SINUMERIK 840D sl	DB10.DBX115.1	

<Nc>.basic.in.swCams[11].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[10]	
SINUMERIK 840D sl	DB10.DBX111.2	

<Nc>.basic.in.swCams[11].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[10]	
SINUMERIK 840D sl	DB10.DBX115.2	

<Nc>.basic.in.swCams[12].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[11]	
SINUMERIK 840D sl	DB10.DBX111.3	

<Nc>.basic.in.swCams[12].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[11]	
SINUMERIK 840D sl	DB10.DBX115.3	

<Nc>.basic.in.swCams[13].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[12]	
SINUMERIK 840D sl	DB10.DBX111.4	

<Nc>.basic.in.swCams[13].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[12]	
SINUMERIK 840D sl	DB10.DBX115.4	

<Nc>.basic.in.swCams[14].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[13]	
SINUMERIK 840D sl	DB10.DBX111.5	

<Nc>.basic.in.swCams[14].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[13]	
SINUMERIK 840D sl	DB10.DBX115.5	

<Nc>.basic.in.swCams[15].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[14]	
SINUMERIK 840D sl	DB10.DBX111.6	

<Nc>.basic.in.swCams[15].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[14]	
SINUMERIK 840D sl	DB10.DBX115.6	

<Nc>.basic.in.swCams[16].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[15]	
SINUMERIK 840D sl	DB10.DBX111.7	

<Nc>.basic.in.swCams[16].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[15]	
SINUMERIK 840D sl	DB10.DBX115.7	

<Nc>.basic.in.swCams[17].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[16]	
SINUMERIK 840D sl	DB10.DBX112.0	

<Nc>.basic.in.swCams[17].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[16]	
SINUMERIK 840D sl	DB10.DBX116.0	

<Nc>.basic.in.swCams[18].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[17]	
SINUMERIK 840D sl	DB10.DBX112.1	

<Nc>.basic.in.swCams[18].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[17]	
SINUMERIK 840D sl	DB10.DBX116.1	

<Nc>.basic.in.swCams[19].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[18]	
SINUMERIK 840D sl	DB10.DBX112.2	

<Nc>.basic.in.swCams[19].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[18]	
SINUMERIK 840D sl	DB10.DBX116.2	

<Nc>.basic.in.swCams[20].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[19]	
SINUMERIK 840D sl	DB10.DBX112.3	

<Nc>.basic.in.swCams[20].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[19]	
SINUMERIK 840D sl	DB10.DBX116.3	

<Nc>.basic.in.swCams[21].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[20]	
SINUMERIK 840D sl	DB10.DBX112.4	

<Nc>.basic.in.swCams[21].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[20]	
SINUMERIK 840D sl	DB10.DBX116.4	

<Nc>.basic.in.swCams[22].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[21]	
SINUMERIK 840D sl	DB10.DBX112.5	

<Nc>.basic.in.swCams[22].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[21]	
SINUMERIK 840D sl	DB10.DBX116.5	

<Nc>.basic.in.swCams[23].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[22]	
SINUMERIK 840D sl	DB10.DBX112.6	

<Nc>.basic.in.swCams[23].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[22]	
SINUMERIK 840D sl	DB10.DBX116.6	

<Nc>.basic.in.swCams[24].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[23]	
SINUMERIK 840D sl	DB10.DBX112.7	

<Nc>.basic.in.swCams[24].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[23]	
SINUMERIK 840D sl	DB10.DBX116.7	

<Nc>.basic.in.swCams[25].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[24]	
SINUMERIK 840D sl	DB10.DBX113.0	

<Nc>.basic.in.swCams[25].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[24]	
SINUMERIK 840D sl	DB10.DBX117.0	

<Nc>.basic.in.swCams[26].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[25]	
SINUMERIK 840D sl	DB10.DBX113.1	

<Nc>.basic.in.swCams[26].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[25]	
SINUMERIK 840D sl	DB10.DBX117.1	

<Nc>.basic.in.swCams[27].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[26]	
SINUMERIK 840D sl	DB10.DBX113.2	

<Nc>.basic.in.swCams[27].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[26]	
SINUMERIK 840D sl	DB10.DBX117.2	

<Nc>.basic.in.swCams[28].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[27]	
SINUMERIK 840D sl	DB10.DBX113.3	

<Nc>.basic.in.swCams[28].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[27]	
SINUMERIK 840D sl	DB10.DBX117.3	

<Nc>.basic.in.swCams[29].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[28]	
SINUMERIK 840D sl	DB10.DBX113.4	

<Nc>.basic.in.swCams[29].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[28]	
SINUMERIK 840D sl	DB10.DBX117.4	

<Nc>.basic.in.swCams[30].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[29]	
SINUMERIK 840D sl	DB10.DBX113.5	

<Nc>.basic.in.swCams[30].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[29]	
SINUMERIK 840D sl	DB10.DBX117.5	

<Nc>.basic.in.swCams[31].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[30]	
SINUMERIK 840D sl	DB10.DBX113.6	

<Nc>.basic.in.swCams[31].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[30]	
SINUMERIK 840D sl	DB10.DBX117.6	

<Nc>.basic.in.swCams[32].minus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamMinus[31]	
SINUMERIK 840D sl	DB10.DBX113.7	

<Nc>.basic.in.swCams[32].plus	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_SWCamPlus[31]	
SINUMERIK 840D sl	DB10.DBX117.7	

<Nc>.basic.out.emergencyStop	**PLC → NC**	**Bool**
急停已请求		
SINUMERIK ONE Basic Program	LBP_NC.A_EMERGENCY	
SINUMERIK 840D sl	DB10.DBX56.1	

<Nc>.basic.out.emergencyStopAck	**PLC → NC**	**Bool**
已请求应答状态“急停”		
SINUMERIK ONE Basic Program	LBP_NC.A_EMERGENCY_Ackn	
SINUMERIK 840D sl	DB10.DBX56.2	

<Nc>.basic.out.keyOperatedSwitchPos0	**PLC → NC**	**Bool**
通过设置钥匙开关位置 0，可启用或禁用对 NC 中某些单元的访问		
SINUMERIK ONE Basic Program	LBP_NC.A_Keyswitch0	
SINUMERIK 840D sl	DB10.DBX56.4	

<Nc>.basic.out.keyOperatedSwitchPos1	**PLC → NC**	**Bool**
通过设置钥匙开关位置 1，可启用或禁用对 NC 中某些单元的访问		
SINUMERIK ONE Basic Program	LBP_NC.A_Keyswitch1	
SINUMERIK 840D sl	DB10.DBX56.5	

<Nc>.basic.out.keyOperatedSwitchPos2	**PLC → NC**	**Bool**
通过设置钥匙开关位置 2，可启用或禁用对 NC 中某些单元的访问		
SINUMERIK ONE Basic Program	LBP_NC.A_Keyswitch2	
SINUMERIK 840D sl	DB10.DBX56.6	

<Nc>.basic.out.keyOperatedSwitchPos3	**PLC → NC**	**Bool**
通过设置钥匙开关位置 3，可启用或禁用对 NC 中某些单元的访问		
SINUMERIK ONE Basic Program	LBP_NC.A_Keyswitch3	
SINUMERIK 840D sl	DB10.DBX56.7	

<Nc>.basic.out.selectIncFromModeGroup	**PLC → NC**	**Bool**
在该运行方式组内，可以选择增量式移动。		
SINUMERIK ONE Basic Program	LBP_NC.A_IncInModeGroup	
SINUMERIK 840D sl	DB10.DBX57.0	

<Nc>.basic.out.internalAsupInfo	**PLC → OP**	**USInt**
仅内部使用		
SINUMERIK 840D sl	DB19.DBB44	

<Nc>.basic.out.internalAsupStatus	**PLC → OP**	**USInt**
仅内部使用		
SINUMERIK 840D sl	DB19.DBB45	

<Nc>.basic.out.internalAsupReference	**PLC → OP**	**USInt**
仅内部使用		
SINUMERIK 840D sl	DB19.DBW46	

<Nc>.basic.out.transferActToolBasePos	**PLC → NC**	**Bool**
传输轴的当前 MCS 设定值		

<Nc>.basic.out.enableReducedDynamics	**PLC → NC**	**Bool**	**起始版本 V11**
激活“降低动态模式”请求			
SINUMERIK ONE Basic Program	LBP_NC.A_EnableReducedDynamics		
SINUMERIK 840D sl	DB10.DBX57.6		

<Nc>.handwheels[1].in.movedCounter	**NC → PLC**	**USInt**
手轮计数时不含方向、刻度和移动量不符。该信号可用于故障查找		
SINUMERIK ONE Basic Program	LBP_NC.E_HW1_moved, E_HW2_moved ...	
SINUMERIK 840D sl	DB10.DBB68..70, .DBB242..244	

<Nc>.handwheels[1].in.stationary	**NC → PLC**	**Bool**
以太网手轮停止		
SINUMERIK ONE Basic Program	LBP_NC.E_EHW1_stands, E_EHW2_stands ...	
SINUMERIK 840D sl	DB10.DBX245.0..245.5	

<Nc>.handwheels[2].in.movedCounter	**NC → PLC**	**USInt**
SINUMERIK ONE Basic Program	LBP_NC.E_HW2_moved	
SINUMERIK 840D sl	DB10.DBB69	

<Nc>.handwheels[2].in.stationary	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_EHW2_stands	
SINUMERIK 840D sl	DB10.DBX245.1	

<Nc>.handwheels[3].in.movedCounter	**NC → PLC**	**USInt**
SINUMERIK ONE Basic Program	LBP_NC.E_HW3_moved	
SINUMERIK 840D sl	DB10.DBB70	

<Nc>.handwheels[3].in.stationary	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_EHW3_stands	
SINUMERIK 840D sl	DB10.DBX245.2	

<Nc>.handwheels[4].in.movedCounter	**NC → PLC**	**USInt**
SINUMERIK ONE Basic Program	LBP_NC.E_HW4_moved	
SINUMERIK 840D sl	DB10.DBB242	

<Nc>.handwheels[4].in.stationary	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_EHW4_stands	
SINUMERIK 840D sl	DB10.DBX245.3	

<Nc>.handwheels[5].in.movedCounter	**NC → PLC**	**USInt**
SINUMERIK ONE Basic Program	LBP_NC.E_HW5_moved	
SINUMERIK 840D sl	DB10.DBB243	

<Nc>.handwheels[5].in.stationary	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_EHW5_stands	
SINUMERIK 840D sl	DB10.DBX245.4	

<Nc>.handwheels[6].in.movedCounter	**NC → PLC**	**USInt**
SINUMERIK ONE Basic Program	LBP_NC.E_HW6_moved	
SINUMERIK 840D sl	DB10.DBB244	

<Nc>.handwheels[6].in.stationary	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_EHW6_stands	
SINUMERIK 840D sl	DB10.DBX245.5	

<Nc>.digitalInputs[1].in.active	**NC → PLC**	**Bool**
NC 硬件数字量输入的实际状态		
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In1 - 40	
SINUMERIK 840D sl	DB10.DBX60.0..7, .DBX186.0..189.7	

<Nc>.digitalInputs[2].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In2	
SINUMERIK 840D sl	DB10.DBX60.1	

<Nc>.digitalInputs[3].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In3	
SINUMERIK 840D sl	DB10.DBX60.2	

<Nc>.digitalInputs[4].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In4	
SINUMERIK 840D sl	DB10.DBX60.3	

<Nc>.digitalInputs[5].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.f60_4	
SINUMERIK 840D sl	DB10.DBX60.4	

<Nc>.digitalInputs[6].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.f60_5	
SINUMERIK 840D sl	DB10.DBX60.5	

<Nc>.digitalInputs[7].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.f60_6	
SINUMERIK 840D sl	DB10.DBX60.6	

<Nc>.digitalInputs[8].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.f60_7	
SINUMERIK 840D sl	DB10.DBX60.7	

<Nc>.digitalInputs[9].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In9	
SINUMERIK 840D sl	DB10.DBX186.0	

<Nc>.digitalInputs[10].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In10	
SINUMERIK 840D sl	DB10.DBX186.1	

<Nc>.digitalInputs[11].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In11	
SINUMERIK 840D sl	DB10.DBX186.2	

<Nc>.digitalInputs[12].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In12	
SINUMERIK 840D sl	DB10.DBX186.3	

<Nc>.digitalInputs[13].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In13	
SINUMERIK 840D sl	DB10.DBX186.4	

<Nc>.digitalInputs[14].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In14	
SINUMERIK 840D sl	DB10.DBX186.5	

<Nc>.digitalInputs[15].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In15	
SINUMERIK 840D sl	DB10.DBX186.6	

<Nc>.digitalInputs[16].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In16	
SINUMERIK 840D sl	DB10.DBX186.7	

<Nc>.digitalInputs[17].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In17	
SINUMERIK 840D sl	DB10.DBX187.0	

<Nc>.digitalInputs[18].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In18	
SINUMERIK 840D sl	DB10.DBX187.1	

<Nc>.digitalInputs[19].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In19	
SINUMERIK 840D sl	DB10.DBX187.2	

<Nc>.digitalInputs[20].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In20	
SINUMERIK 840D sl	DB10.DBX187.3	

<Nc>.digitalInputs[21].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In21	
SINUMERIK 840D sl	DB10.DBX187.4	

<Nc>.digitalInputs[22].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In22	
SINUMERIK 840D sl	DB10.DBX187.5	

<Nc>.digitalInputs[23].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In23	
SINUMERIK 840D sl	DB10.DBX187.6	

<Nc>.digitalInputs[24].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In24	
SINUMERIK 840D sl	DB10.DBX187.7	

<Nc>.digitalInputs[25].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In25	
SINUMERIK 840D sl	DB10.DBX188.0	

<Nc>.digitalInputs[26].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In26	
SINUMERIK 840D sl	DB10.DBX188.1	

<Nc>.digitalInputs[27].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In27	
SINUMERIK 840D sl	DB10.DBX188.2	

<Nc>.digitalInputs[28].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In28	
SINUMERIK 840D sl	DB10.DBX188.3	

<Nc>.digitalInputs[29].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In29	
SINUMERIK 840D sl	DB10.DBX188.4	

<Nc>.digitalInputs[30].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In30	
SINUMERIK 840D sl	DB10.DBX188.5	

<Nc>.digitalInputs[31].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In31	
SINUMERIK 840D sl	DB10.DBX188.6	

<Nc>.digitalInputs[32].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In32	
SINUMERIK 840D sl	DB10.DBX188.7	

<Nc>.digitalInputs[33].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In33	
SINUMERIK 840D sl	DB10.DBX189.0	

<Nc>.digitalInputs[34].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In34	
SINUMERIK 840D sl	DB10.DBX189.1	

<Nc>.digitalInputs[35].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In35	
SINUMERIK 840D sl	DB10.DBX189.2	

<Nc>.digitalInputs[36].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In36	
SINUMERIK 840D sl	DB10.DBX189.3	

<Nc>.digitalInputs[37].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In37	
SINUMERIK 840D sl	DB10.DBX189.4	

<Nc>.digitalInputs[38].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In38	
SINUMERIK 840D sl	DB10.DBX189.5	

<Nc>.digitalInputs[39].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In39	
SINUMERIK 840D sl	DB10.DBX189.6	

<Nc>.digitalInputs[40].in.active	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_In40	
SINUMERIK 840D sl	DB10.DBX189.7	

<Nc>.digitalInputs[1].out.lock	**PLC → NC**	**Bool**
将 NC 软件数字量输入"$A_IN[]""锁定"在指定值 0，当"digitalInputs[].out.set"没有生效时		
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp1 - 40	
SINUMERIK 840D sl	DB10.DBX0.1..7, .DBX122.0..7, .DBX124.0..7, .DBX126.0..7, .DBX128.0..7	

<Nc>.digitalInputs[1].out.set	**PLC → NC**	**Bool**
将 NC 软件数字量输入"$A_IN[]"设为指定值 1		
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp1 - 40	
SINUMERIK 840D sl	DB10.DBX1.1..7, .DBX123.0..7, .DBX125.0..7, .DBX127.0..7, .DBX129.0..7	

<Nc>.digitalInputs[2].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp2	
SINUMERIK 840D sl	DB10.DBX0.1	

<Nc>.digitalInputs[2].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp2	
SINUMERIK 840D sl	DB10.DBX1.1	

<Nc>.digitalInputs[3].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp3	
SINUMERIK 840D sl	DB10.DBX0.2	

<Nc>.digitalInputs[3].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp3	
SINUMERIK 840D sl	DB10.DBX1.2	

<Nc>.digitalInputs[4].out.lock		**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp4		
SINUMERIK 840D sl	DB10.DBX0.3		

<Nc>.digitalInputs[4].out.set		**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp4		
SINUMERIK 840D sl	DB10.DBX1.3		

<Nc>.digitalInputs[5].out.lock		**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp5		
SINUMERIK 840D sl	DB10.DBX0.4		

<Nc>.digitalInputs[5].out.set		**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp5		
SINUMERIK 840D sl	DB10.DBX1.4		

<Nc>.digitalInputs[6].out.lock		**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp6		
SINUMERIK 840D sl	DB10.DBX0.5		

<Nc>.digitalInputs[6].out.set		**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp6		
SINUMERIK 840D sl	DB10.DBX1.5		

<Nc>.digitalInputs[7].out.lock		**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp7		
SINUMERIK 840D sl	DB10.DBX0.6		

<Nc>.digitalInputs[7].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp7	
SINUMERIK 840D sl	DB10.DBX1.6	

<Nc>.digitalInputs[8].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp8	
SINUMERIK 840D sl	DB10.DBX0.7	

<Nc>.digitalInputs[8].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp8	
SINUMERIK 840D sl	DB10.DBX1.7	

<Nc>.digitalInputs[9].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp9	
SINUMERIK 840D sl	DB10.DBX122.0	

<Nc>.digitalInputs[9].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp9	
SINUMERIK 840D sl	DB10.DBX123.0	

<Nc>.digitalInputs[10].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp10	
SINUMERIK 840D sl	DB10.DBX122.1	

<Nc>.digitalInputs[10].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp10	
SINUMERIK 840D sl	DB10.DBX123.1	

<Nc>.digitalInputs[11].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp11	
SINUMERIK 840D sl	DB10.DBX122.2	

<Nc>.digitalInputs[11].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp11	
SINUMERIK 840D sl	DB10.DBX123.2	

<Nc>.digitalInputs[12].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp12	
SINUMERIK 840D sl	DB10.DBX122.3	

<Nc>.digitalInputs[12].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp12	
SINUMERIK 840D sl	DB10.DBX123.3	

<Nc>.digitalInputs[13].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp13	
SINUMERIK 840D sl	DB10.DBX122.4	

<Nc>.digitalInputs[13].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp13	
SINUMERIK 840D sl	DB10.DBX123.4	

<Nc>.digitalInputs[14].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp14	
SINUMERIK 840D sl	DB10.DBX122.5	

<Nc>.digitalInputs[14].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp14	
SINUMERIK 840D sl	DB10.DBX123.5	

<Nc>.digitalInputs[15].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp15	
SINUMERIK 840D sl	DB10.DBX122.6	

<Nc>.digitalInputs[15].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp15	
SINUMERIK 840D sl	DB10.DBX123.6	

<Nc>.digitalInputs[16].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp16	
SINUMERIK 840D sl	DB10.DBX122.7	

<Nc>.digitalInputs[16].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp16	
SINUMERIK 840D sl	DB10.DBX123.7	

<Nc>.digitalInputs[17].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp17	
SINUMERIK 840D sl	DB10.DBX124.0	

<Nc>.digitalInputs[17].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp17	
SINUMERIK 840D sl	DB10.DBX125.0	

<Nc>.digitalInputs[18].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp18	
SINUMERIK 840D sl	DB10.DBX124.1	

<Nc>.digitalInputs[18].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp18	
SINUMERIK 840D sl	DB10.DBX125.1	

<Nc>.digitalInputs[19].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp19	
SINUMERIK 840D sl	DB10.DBX124.2	

<Nc>.digitalInputs[19].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp19	
SINUMERIK 840D sl	DB10.DBX125.2	

<Nc>.digitalInputs[20].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp20	
SINUMERIK 840D sl	DB10.DBX124.3	

<Nc>.digitalInputs[20].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp20	
SINUMERIK 840D sl	DB10.DBX125.3	

<Nc>.digitalInputs[21].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp21	
SINUMERIK 840D sl	DB10.DBX124.4	

<Nc>.digitalInputs[21].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp21	
SINUMERIK 840D sl	DB10.DBX125.4	

<Nc>.digitalInputs[22].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp22	
SINUMERIK 840D sl	DB10.DBX124.5	

<Nc>.digitalInputs[22].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp22	
SINUMERIK 840D sl	DB10.DBX125.5	

<Nc>.digitalInputs[23].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp23	
SINUMERIK 840D sl	DB10.DBX124.6	

<Nc>.digitalInputs[23].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp23	
SINUMERIK 840D sl	DB10.DBX125.6	

<Nc>.digitalInputs[24].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp24	
SINUMERIK 840D sl	DB10.DBX124.7	

<Nc>.digitalInputs[24].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp24	
SINUMERIK 840D sl	DB10.DBX125.7	

<Nc>.digitalInputs[25].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp25	
SINUMERIK 840D sl	DB10.DBX126.0	

<Nc>.digitalInputs[25].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp25	
SINUMERIK 840D sl	DB10.DBX127.0	

<Nc>.digitalInputs[26].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp26	
SINUMERIK 840D sl	DB10.DBX126.1	

<Nc>.digitalInputs[26].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp26	
SINUMERIK 840D sl	DB10.DBX127.1	

<Nc>.digitalInputs[27].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp27	
SINUMERIK 840D sl	DB10.DBX126.2	

<Nc>.digitalInputs[27].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp27	
SINUMERIK 840D sl	DB10.DBX127.2	

<Nc>.digitalInputs[28].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp28	
SINUMERIK 840D sl	DB10.DBX126.3	

<Nc>.digitalInputs[28].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp28	
SINUMERIK 840D sl	DB10.DBX127.3	

<Nc>.digitalInputs[29].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp29	
SINUMERIK 840D sl	DB10.DBX126.4	

<Nc>.digitalInputs[29].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp29	
SINUMERIK 840D sl	DB10.DBX127.4	

<Nc>.digitalInputs[30].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp30	
SINUMERIK 840D sl	DB10.DBX126.5	

<Nc>.digitalInputs[30].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp30	
SINUMERIK 840D sl	DB10.DBX127.5	

<Nc>.digitalInputs[31].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp31	
SINUMERIK 840D sl	DB10.DBX126.6	

<Nc>.digitalInputs[31].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp31	
SINUMERIK 840D sl	DB10.DBX127.6	

<Nc>.digitalInputs[32].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp32	
SINUMERIK 840D sl	DB10.DBX126.7	

<Nc>.digitalInputs[32].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp32	
SINUMERIK 840D sl	DB10.DBX127.7	

<Nc>.digitalInputs[33].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp33	
SINUMERIK 840D sl	DB10.DBX128.0	

<Nc>.digitalInputs[33].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp33	
SINUMERIK 840D sl	DB10.DBX129.0	

<Nc>.digitalInputs[34].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp34	
SINUMERIK 840D sl	DB10.DBX128.1	

<Nc>.digitalInputs[34].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp34	
SINUMERIK 840D sl	DB10.DBX129.1	

<Nc>.digitalInputs[35].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp35	
SINUMERIK 840D sl	DB10.DBX128.2	

<Nc>.digitalInputs[35].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp35	
SINUMERIK 840D sl	DB10.DBX129.2	

<Nc>.digitalInputs[36].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp36	
SINUMERIK 840D sl	DB10.DBX128.3	

<Nc>.digitalInputs[36].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp36	
SINUMERIK 840D sl	DB10.DBX129.3	

<Nc>.digitalInputs[37].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp37	
SINUMERIK 840D sl	DB10.DBX128.4	

<Nc>.digitalInputs[37].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp37	
SINUMERIK 840D sl	DB10.DBX129.4	

<Nc>.digitalInputs[38].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp38	
SINUMERIK 840D sl	DB10.DBX128.5	

<Nc>.digitalInputs[38].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp38	
SINUMERIK 840D sl	DB10.DBX129.5	

<Nc>.digitalInputs[39].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp39	
SINUMERIK 840D sl	DB10.DBX128.6	

<Nc>.digitalInputs[39].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp39	
SINUMERIK 840D sl	DB10.DBX129.6	

<Nc>.digitalInputs[40].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Inp40	
SINUMERIK 840D sl	DB10.DBX128.7	

<Nc>.digitalInputs[40].out.set	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Inp40	
SINUMERIK 840D sl	DB10.DBX129.7	

<Nc>.digitalOutputs[1].in.setpointState	**NC → PLC**	**Bool**
NC 软件数字量输出"$A_OUT[]"的值由 NC 设置，当"digitalOutputs[].out.overwrite"没有生效时		
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out1 - 40	
SINUMERIK 840D sl	DB10.DBX64.1..7, .DBX190.0..193.7	

<Nc>.digitalOutputs[2].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out2	
SINUMERIK 840D sl	DB10.DBX64.1	

<Nc>.digitalOutputs[3].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out3	
SINUMERIK 840D sl	DB10.DBX64.2	

<Nc>.digitalOutputs[4].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out4	
SINUMERIK 840D sl	DB10.DBX64.3	

<Nc>.digitalOutputs[5].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out5	
SINUMERIK 840D sl	DB10.DBX64.4	

<Nc>.digitalOutputs[6].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out6	
SINUMERIK 840D sl	DB10.DBX64.5	

<Nc>.digitalOutputs[7].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out7	
SINUMERIK 840D sl	DB10.DBX64.6	

<Nc>.digitalOutputs[8].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out8	
SINUMERIK 840D sl	DB10.DBX64.7	

<Nc>.digitalOutputs[9].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out9	
SINUMERIK 840D sl	DB10.DBX190.0	

<Nc>.digitalOutputs[10].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out10	
SINUMERIK 840D sl	DB10.DBX190.1	

<Nc>.digitalOutputs[11].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out11	
SINUMERIK 840D sl	DB10.DBX190.2	

<Nc>.digitalOutputs[12].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out12	
SINUMERIK 840D sl	DB10.DBX190.3	

<Nc>.digitalOutputs[13].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out13	
SINUMERIK 840D sl	DB10.DBX190.4	

<Nc>.digitalOutputs[14].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out14	
SINUMERIK 840D sl	DB10.DBX190.5	

<Nc>.digitalOutputs[15].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out15	
SINUMERIK 840D sl	DB10.DBX190.6	

<Nc>.digitalOutputs[16].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out16	
SINUMERIK 840D sl	DB10.DBX190.7	

<Nc>.digitalOutputs[17].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out17	
SINUMERIK 840D sl	DB10.DBX191.0	

<Nc>.digitalOutputs[18].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out18	
SINUMERIK 840D sl	DB10.DBX191.1	

<Nc>.digitalOutputs[19].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out19	
SINUMERIK 840D sl	DB10.DBX191.2	

<Nc>.digitalOutputs[20].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out20	
SINUMERIK 840D sl	DB10.DBX191.3	

<Nc>.digitalOutputs[21].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out21	
SINUMERIK 840D sl	DB10.DBX191.4	

<Nc>.digitalOutputs[22].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out22	
SINUMERIK 840D sl	DB10.DBX191.5	

<Nc>.digitalOutputs[23].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out23	
SINUMERIK 840D sl	DB10.DBX191.6	

<Nc>.digitalOutputs[24].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out24	
SINUMERIK 840D sl	DB10.DBX191.7	

<Nc>.digitalOutputs[25].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out25	
SINUMERIK 840D sl	DB10.DBX192.0	

<Nc>.digitalOutputs[26].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out26	
SINUMERIK 840D sl	DB10.DBX192.1	

<Nc>.digitalOutputs[27].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out27	
SINUMERIK 840D sl	DB10.DBX192.2	

<Nc>.digitalOutputs[28].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out28	
SINUMERIK 840D sl	DB10.DBX192.3	

<Nc>.digitalOutputs[29].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out29	
SINUMERIK 840D sl	DB10.DBX192.4	

<Nc>.digitalOutputs[30].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out30	
SINUMERIK 840D sl	DB10.DBX192.5	

<Nc>.digitalOutputs[31].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out31	
SINUMERIK 840D sl	DB10.DBX192.6	

<Nc>.digitalOutputs[32].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out32	
SINUMERIK 840D sl	DB10.DBX192.7	

<Nc>.digitalOutputs[33].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out33	
SINUMERIK 840D sl	DB10.DBX193.0	

<Nc>.digitalOutputs[34].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out34	
SINUMERIK 840D sl	DB10.DBX193.1	

<Nc>.digitalOutputs[35].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out35	
SINUMERIK 840D sl	DB10.DBX193.2	

<Nc>.digitalOutputs[36].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out36	
SINUMERIK 840D sl	DB10.DBX193.3	

<Nc>.digitalOutputs[37].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out37	
SINUMERIK 840D sl	DB10.DBX193.4	

<Nc>.digitalOutputs[38].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out38	
SINUMERIK 840D sl	DB10.DBX193.5	

<Nc>.digitalOutputs[39].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out39	
SINUMERIK 840D sl	DB10.DBX193.6	

<Nc>.digitalOutputs[40].in.setpointState	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.E_Setpoint_Out40	
SINUMERIK 840D sl	DB10.DBX193.7	

<Nc>.digitalOutputs[1].out.settingState	**PLC → NC**	**Bool**
数值传递给 NC 硬件数字量输出，当“digitalOutputs[].out.overwrite”或“digitalOutputs[].out.target”中任意一个生效时。		
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out1 - 40	
SINUMERIK 840D sl	DB10.DBX6.1..7, .DBX132.0..7, .DBX136.0..7, .DBX140.0..7, .DBX144.0..7	

<Nc>.digitalOutputs[1].out.overwrite	**PLC → NC**	**Bool**
出现 0 → 1 上升沿：将“digitalOutpute[].out.settingState”传送给硬件数字量输出，NC 数字量值“$A_OUT[]”置 0		
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out1 - 40	
SINUMERIK 840D sl	DB10.DBX5.1..7, .DBX131.0..7, .DBX135.0..7, .DBX139.0..7, .DBX143.0..7	

<Nc>.digitalOutputs[1].out.target	**PLC → NC**	**Bool**
将“digitalOutputs[].out.settingState”传送到硬件数字量输出；实际的 NC 软件数字量输出值“$A_OUT[]”保持不变。		
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out1 - 40	
SINUMERIK 840D sl	DB10.DBX7.1..7, .DBX133.0..7, .DBX137.0..7, .DBX141.0..7, .DBX145.0..7	

<Nc>.digitalOutputs[1].out.lock	**PLC → NC**	**Bool**
将 NC 硬件数字量输出“锁定”在指定值 0		
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out1 - 40	
SINUMERIK 840D sl	DB10.DBX4.1..7, .DBX130.0..7, .DBX134.0..7, .DBX138.0..7, .DBX142.0..7	

<Nc>.digitalOutputs[2].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out2	
SINUMERIK 840D sl	DB10.DBX6.1	

<Nc>.digitalOutputs[2].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out2	
SINUMERIK 840D sl	DB10.DBX5.1	

<Nc>.digitalOutputs[2].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out2	
SINUMERIK 840D sl	DB10.DBX7.1	

<Nc>.digitalOutputs[2].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out2	
SINUMERIK 840D sl	DB10.DBX4.1	

<Nc>.digitalOutputs[3].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out3	
SINUMERIK 840D sl	DB10.DBX6.2	

<Nc>.digitalOutputs[3].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out3	
SINUMERIK 840D sl	DB10.DBX5.2	

<Nc>.digitalOutputs[3].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out3	
SINUMERIK 840D sl	DB10.DBX7.2	

<Nc>.digitalOutputs[3].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out3	
SINUMERIK 840D sl	DB10.DBX4.2	

<Nc>.digitalOutputs[4].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out4	
SINUMERIK 840D sl	DB10.DBX6.3	

<Nc>.digitalOutputs[4].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out4	
SINUMERIK 840D sl	DB10.DBX5.3	

<Nc>.digitalOutputs[4].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out4	
SINUMERIK 840D sl	DB10.DBX7.3	

<Nc>.digitalOutputs[4].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out4	
SINUMERIK 840D sl	DB10.DBX4.3	

<Nc>.digitalOutputs[5].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out5	
SINUMERIK 840D sl	DB10.DBX6.4	

<Nc>.digitalOutputs[5].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out5	
SINUMERIK 840D sl	DB10.DBX5.4	

<Nc>.digitalOutputs[5].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out5	
SINUMERIK 840D sl	DB10.DBX7.4	

<Nc>.digitalOutputs[5].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out5	
SINUMERIK 840D sl	DB10.DBX4.4	

<Nc>.digitalOutputs[6].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out6	
SINUMERIK 840D sl	DB10.DBX6.5	

<Nc>.digitalOutputs[6].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out6	
SINUMERIK 840D sl	DB10.DBX5.5	

<Nc>.digitalOutputs[6].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out6	
SINUMERIK 840D sl	DB10.DBX7.5	

<Nc>.digitalOutputs[6].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out6	
SINUMERIK 840D sl	DB10.DBX4.5	

<Nc>.digitalOutputs[7].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out7	
SINUMERIK 840D sl	DB10.DBX6.6	

<Nc>.digitalOutputs[7].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out7	
SINUMERIK 840D sl	DB10.DBX5.6	

<Nc>.digitalOutputs[7].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out7	
SINUMERIK 840D sl	DB10.DBX7.6	

<Nc>.digitalOutputs[7].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out7	
SINUMERIK 840D sl	DB10.DBX4.6	

<Nc>.digitalOutputs[8].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out8	
SINUMERIK 840D sl	DB10.DBX6.7	

<Nc>.digitalOutputs[8].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out8	
SINUMERIK 840D sl	DB10.DBX5.7	

<Nc>.digitalOutputs[8].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out8	
SINUMERIK 840D sl	DB10.DBX7.7	

<Nc>.digitalOutputs[8].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out8	
SINUMERIK 840D sl	DB10.DBX4.7	

<Nc>.digitalOutputs[9].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out9	
SINUMERIK 840D sl	DB10.DBX132.0	

<Nc>.digitalOutputs[9].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out9	
SINUMERIK 840D sl	DB10.DBX131.0	

<Nc>.digitalOutputs[9].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out9	
SINUMERIK 840D sl	DB10.DBX133.0	

<Nc>.digitalOutputs[9].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out9	
SINUMERIK 840D sl	DB10.DBX130.0	

<Nc>.digitalOutputs[10].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out10	
SINUMERIK 840D sl	DB10.DBX132.1	

<Nc>.digitalOutputs[10].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out10	
SINUMERIK 840D sl	DB10.DBX131.1	

<Nc>.digitalOutputs[10].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out10	
SINUMERIK 840D sl	DB10.DBX133.1	

<Nc>.digitalOutputs[10].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out10	
SINUMERIK 840D sl	DB10.DBX130.1	

<Nc>.digitalOutputs[11].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out11	
SINUMERIK 840D sl	DB10.DBX132.2	

<Nc>.digitalOutputs[11].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out11	
SINUMERIK 840D sl	DB10.DBX131.2	

<Nc>.digitalOutputs[11].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out11	
SINUMERIK 840D sl	DB10.DBX133.2	

<Nc>.digitalOutputs[11].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out11	
SINUMERIK 840D sl	DB10.DBX130.2	

<Nc>.digitalOutputs[12].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out12	
SINUMERIK 840D sl	DB10.DBX132.3	

<Nc>.digitalOutputs[12].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out12	
SINUMERIK 840D sl	DB10.DBX131.3	

<Nc>.digitalOutputs[12].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out12	
SINUMERIK 840D sl	DB10.DBX133.3	

<Nc>.digitalOutputs[12].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out12	
SINUMERIK 840D sl	DB10.DBX130.3	

<Nc>.digitalOutputs[13].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out13	
SINUMERIK 840D sl	DB10.DBX132.4	

<Nc>.digitalOutputs[13].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out13	
SINUMERIK 840D sl	DB10.DBX131.4	

<Nc>.digitalOutputs[13].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out13	
SINUMERIK 840D sl	DB10.DBX133.4	

<Nc>.digitalOutputs[13].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out13	
SINUMERIK 840D sl	DB10.DBX130.4	

<Nc>.digitalOutputs[14].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out14	
SINUMERIK 840D sl	DB10.DBX132.5	

<Nc>.digitalOutputs[14].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out14	
SINUMERIK 840D sl	DB10.DBX131.5	

<Nc>.digitalOutputs[14].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out14	
SINUMERIK 840D sl	DB10.DBX133.5	

<Nc>.digitalOutputs[14].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out14	
SINUMERIK 840D sl	DB10.DBX130.5	

<Nc>.digitalOutputs[15].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out15	
SINUMERIK 840D sl	DB10.DBX132.6	

<Nc>.digitalOutputs[15].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out15	
SINUMERIK 840D sl	DB10.DBX131.6	

<Nc>.digitalOutputs[15].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out15	
SINUMERIK 840D sl	DB10.DBX133.6	

<Nc>.digitalOutputs[15].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out15	
SINUMERIK 840D sl	DB10.DBX130.6	

<Nc>.digitalOutputs[16].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out16	
SINUMERIK 840D sl	DB10.DBX132.7	

<Nc>.digitalOutputs[16].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out16	
SINUMERIK 840D sl	DB10.DBX131.7	

<Nc>.digitalOutputs[16].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out16	
SINUMERIK 840D sl	DB10.DBX133.7	

<Nc>.digitalOutputs[16].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out16	
SINUMERIK 840D sl	DB10.DBX130.7	

<Nc>.digitalOutputs[17].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out17	
SINUMERIK 840D sl	DB10.DBX136.0	

<Nc>.digitalOutputs[17].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out17	
SINUMERIK 840D sl	DB10.DBX135.0	

<Nc>.digitalOutputs[17].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out17	
SINUMERIK 840D sl	DB10.DBX137.0	

<Nc>.digitalOutputs[17].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out17	
SINUMERIK 840D sl	DB10.DBX134.0	

<Nc>.digitalOutputs[18].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out18	
SINUMERIK 840D sl	DB10.DBX136.1	

<Nc>.digitalOutputs[18].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out18	
SINUMERIK 840D sl	DB10.DBX135.1	

<Nc>.digitalOutputs[18].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out18	
SINUMERIK 840D sl	DB10.DBX137.1	

<Nc>.digitalOutputs[18].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out18	
SINUMERIK 840D sl	DB10.DBX134.1	

<Nc>.digitalOutputs[19].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out19	
SINUMERIK 840D sl	DB10.DBX136.2	

<Nc>.digitalOutputs[19].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out19	
SINUMERIK 840D sl	DB10.DBX135.2	

<Nc>.digitalOutputs[19].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out19	
SINUMERIK 840D sl	DB10.DBX137.2	

<Nc>.digitalOutputs[19].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out19	
SINUMERIK 840D sl	DB10.DBX134.2	

<Nc>.digitalOutputs[20].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out20	
SINUMERIK 840D sl	DB10.DBX136.3	

<Nc>.digitalOutputs[20].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out20	
SINUMERIK 840D sl	DB10.DBX135.3	

<Nc>.digitalOutputs[20].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out20	
SINUMERIK 840D sl	DB10.DBX137.3	

<Nc>.digitalOutputs[20].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out20	
SINUMERIK 840D sl	DB10.DBX134.3	

<Nc>.digitalOutputs[21].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out21	
SINUMERIK 840D sl	DB10.DBX136.4	

<Nc>.digitalOutputs[21].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out21	
SINUMERIK 840D sl	DB10.DBX135.4	

<Nc>.digitalOutputs[21].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out21	
SINUMERIK 840D sl	DB10.DBX137.4	

<Nc>.digitalOutputs[21].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out21	
SINUMERIK 840D sl	DB10.DBX134.4	

<Nc>.digitalOutputs[22].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out22	
SINUMERIK 840D sl	DB10.DBX136.5	

<Nc>.digitalOutputs[22].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out22	
SINUMERIK 840D sl	DB10.DBX135.5	

<Nc>.digitalOutputs[22].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out22	
SINUMERIK 840D sl	DB10.DBX137.5	

<Nc>.digitalOutputs[22].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out22	
SINUMERIK 840D sl	DB10.DBX134.5	

<Nc>.digitalOutputs[23].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out23	
SINUMERIK 840D sl	DB10.DBX136.6	

<Nc>.digitalOutputs[23].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out23	
SINUMERIK 840D sl	DB10.DBX135.6	

<Nc>.digitalOutputs[23].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out23	
SINUMERIK 840D sl	DB10.DBX137.6	

<Nc>.digitalOutputs[23].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out23	
SINUMERIK 840D sl	DB10.DBX134.6	

<Nc>.digitalOutputs[24].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out24	
SINUMERIK 840D sl	DB10.DBX136.7	

<Nc>.digitalOutputs[24].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out24	
SINUMERIK 840D sl	DB10.DBX135.7	

<Nc>.digitalOutputs[24].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out24	
SINUMERIK 840D sl	DB10.DBX137.7	

<Nc>.digitalOutputs[24].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out24	
SINUMERIK 840D sl	DB10.DBX134.7	

<Nc>.digitalOutputs[25].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out25	
SINUMERIK 840D sl	DB10.DBX140.0	

<Nc>.digitalOutputs[25].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out25	
SINUMERIK 840D sl	DB10.DBX139.0	

<Nc>.digitalOutputs[25].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out25	
SINUMERIK 840D sl	DB10.DBX141.0	

<Nc>.digitalOutputs[25].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out25	
SINUMERIK 840D sl	DB10.DBX138.0	

<Nc>.digitalOutputs[26].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out26	
SINUMERIK 840D sl	DB10.DBX140.1	

<Nc>.digitalOutputs[26].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out26	
SINUMERIK 840D sl	DB10.DBX139.1	

<Nc>.digitalOutputs[26].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out26	
SINUMERIK 840D sl	DB10.DBX141.1	

<Nc>.digitalOutputs[26].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out26	
SINUMERIK 840D sl	DB10.DBX138.1	

<Nc>.digitalOutputs[27].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out27	
SINUMERIK 840D sl	DB10.DBX140.2	

<Nc>.digitalOutputs[27].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out27	
SINUMERIK 840D sl	DB10.DBX139.2	

<Nc>.digitalOutputs[27].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out27	
SINUMERIK 840D sl	DB10.DBX141.2	

<Nc>.digitalOutputs[27].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out27	
SINUMERIK 840D sl	DB10.DBX138.2	

<Nc>.digitalOutputs[28].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out28	
SINUMERIK 840D sl	DB10.DBX140.3	

<Nc>.digitalOutputs[28].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out28	
SINUMERIK 840D sl	DB10.DBX139.3	

<Nc>.digitalOutputs[28].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out28	
SINUMERIK 840D sl	DB10.DBX141.3	

<Nc>.digitalOutputs[28].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out28	
SINUMERIK 840D sl	DB10.DBX138.3	

<Nc>.digitalOutputs[29].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out29	
SINUMERIK 840D sl	DB10.DBX140.4	

<Nc>.digitalOutputs[29].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out29	
SINUMERIK 840D sl	DB10.DBX139.4	

<Nc>.digitalOutputs[29].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out29	
SINUMERIK 840D sl	DB10.DBX141.4	

<Nc>.digitalOutputs[29].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out29	
SINUMERIK 840D sl	DB10.DBX138.4	

<Nc>.digitalOutputs[30].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out30	
SINUMERIK 840D sl	DB10.DBX140.5	

<Nc>.digitalOutputs[30].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out30	
SINUMERIK 840D sl	DB10.DBX139.5	

<Nc>.digitalOutputs[30].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out30	
SINUMERIK 840D sl	DB10.DBX141.5	

<Nc>.digitalOutputs[30].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out30	
SINUMERIK 840D sl	DB10.DBX138.5	

<Nc>.digitalOutputs[31].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out31	
SINUMERIK 840D sl	DB10.DBX140.6	

<Nc>.digitalOutputs[31].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out31	
SINUMERIK 840D sl	DB10.DBX139.6	

<Nc>.digitalOutputs[31].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out31	
SINUMERIK 840D sl	DB10.DBX141.6	

<Nc>.digitalOutputs[31].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out31	
SINUMERIK 840D sl	DB10.DBX138.6	

<Nc>.digitalOutputs[32].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out32	
SINUMERIK 840D sl	DB10.DBX140.7	

<Nc>.digitalOutputs[32].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out32	
SINUMERIK 840D sl	DB10.DBX139.7	

<Nc>.digitalOutputs[32].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out32	
SINUMERIK 840D sl	DB10.DBX141.7	

<Nc>.digitalOutputs[32].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out32	
SINUMERIK 840D sl	DB10.DBX138.7	

<Nc>.digitalOutputs[33].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out33	
SINUMERIK 840D sl	DB10.DBX144.0	

<Nc>.digitalOutputs[33].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out33	
SINUMERIK 840D sl	DB10.DBX143.0	

<Nc>.digitalOutputs[33].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out33	
SINUMERIK 840D sl	DB10.DBX145.0	

<Nc>.digitalOutputs[33].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out33	
SINUMERIK 840D sl	DB10.DBX142.0	

<Nc>.digitalOutputs[34].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out34	
SINUMERIK 840D sl	DB10.DBX144.1	

<Nc>.digitalOutputs[34].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out34	
SINUMERIK 840D sl	DB10.DBX143.1	

<Nc>.digitalOutputs[34].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out34	
SINUMERIK 840D sl	DB10.DBX145.1	

<Nc>.digitalOutputs[34].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out34	
SINUMERIK 840D sl	DB10.DBX142.1	

<Nc>.digitalOutputs[35].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out35	
SINUMERIK 840D sl	DB10.DBX144.2	

<Nc>.digitalOutputs[35].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out35	
SINUMERIK 840D sl	DB10.DBX143.2	

<Nc>.digitalOutputs[35].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out35	
SINUMERIK 840D sl	DB10.DBX145.2	

<Nc>.digitalOutputs[35].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out35	
SINUMERIK 840D sl	DB10.DBX142.2	

<Nc>.digitalOutputs[36].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out36	
SINUMERIK 840D sl	DB10.DBX144.3	

<Nc>.digitalOutputs[36].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out36	
SINUMERIK 840D sl	DB10.DBX143.3	

<Nc>.digitalOutputs[36].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out36	
SINUMERIK 840D sl	DB10.DBX145.3	

<Nc>.digitalOutputs[36].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out36	
SINUMERIK 840D sl	DB10.DBX142.3	

<Nc>.digitalOutputs[37].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out37	
SINUMERIK 840D sl	DB10.DBX144.4	

<Nc>.digitalOutputs[37].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out37	
SINUMERIK 840D sl	DB10.DBX143.4	

<Nc>.digitalOutputs[37].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out37	
SINUMERIK 840D sl	DB10.DBX145.4	

<Nc>.digitalOutputs[37].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out37	
SINUMERIK 840D sl	DB10.DBX142.4	

<Nc>.digitalOutputs[38].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out38	
SINUMERIK 840D sl	DB10.DBX144.5	

<Nc>.digitalOutputs[38].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out38	
SINUMERIK 840D sl	DB10.DBX143.5	

<Nc>.digitalOutputs[38].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out38	
SINUMERIK 840D sl	DB10.DBX145.5	

<Nc>.digitalOutputs[38].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out38	
SINUMERIK 840D sl	DB10.DBX142.5	

<Nc>.digitalOutputs[39].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out39	
SINUMERIK 840D sl	DB10.DBX144.6	

<Nc>.digitalOutputs[39].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out39	
SINUMERIK 840D sl	DB10.DBX143.6	

<Nc>.digitalOutputs[39].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out39	
SINUMERIK 840D sl	DB10.DBX145.6	

<Nc>.digitalOutputs[39].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out39	
SINUMERIK 840D sl	DB10.DBX142.6	

<Nc>.digitalOutputs[40].out.settingState	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Set_Out40	
SINUMERIK 840D sl	DB10.DBX144.7	

<Nc>.digitalOutputs[40].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_Out40	
SINUMERIK 840D sl	DB10.DBX143.7	

<Nc>.digitalOutputs[40].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_Out40	
SINUMERIK 840D sl	DB10.DBX145.7	

<Nc>.digitalOutputs[40].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_Out40	
SINUMERIK 840D sl	DB10.DBX142.7	

<Nc>.analogInputs[1].in.actualValue	**NC → PLC**	**Int**
NC 硬件模拟量输入的实际值在 NC 软件模拟量输入"$A_INA[]"上提供		
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_anaIn1 - 8	
SINUMERIK 840D sl	DB10.DBW194..208	

<Nc>.analogInputs[2].in.actualValue	**NC → PLC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_anaIn2	
SINUMERIK 840D sl	DB10.DBW196	

<Nc>.analogInputs[3].in.actualValue	**NC → PLC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_anaIn3	
SINUMERIK 840D sl	DB10.DBW198	

<Nc>.analogInputs[4].in.actualValue	**NC → PLC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_anaIn4	
SINUMERIK 840D sl	DB10.DBW200	

<Nc>.analogInputs[5].in.actualValue	**NC → PLC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_anaIn5	
SINUMERIK 840D sl	DB10.DBW202	

<Nc>.analogInputs[6].in.actualValue	**NC → PLC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_anaIn6	
SINUMERIK 840D sl	DB10.DBW204	

<Nc>.analogInputs[7].in.actualValue	**NC → PLC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_anaIn7	
SINUMERIK 840D sl	DB10.DBW206	

<Nc>.analogInputs[8].in.actualValue	**NC → PLC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.E_ActVal_anaIn8	
SINUMERIK 840D sl	DB10.DBW208	

<Nc>.analogInputs[1].out.lock	**PLC → NC**	**Bool**
当“analogInputs[].out.target”没有生效时，便将 NC 软件模拟量输入“ $A_INA[]”“锁定”在指定值 0。		
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaIn1 - 8	
SINUMERIK 840D sl	DB10.DBX146.x	

<Nc>.analogInputs[1].out.target	**PLC → NC**	**Bool**
将“analogInputs[].out.settingValue”传送到 NC 软件模拟量输入“$A_INA[]”		
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaIn1 - 8	
SINUMERIK 840D sl	DB10.DBX147.x	

<Nc>.analogInputs[1].out.settingValue	**PLC → NC**	**Int**
当“analogInput[].out.target”生效时，将“analogInputs[].out.settingValue”传送到 NC 软件模拟量输入“$A_INA[]”		
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaIn1 - 8	
SINUMERIK 840D sl	DB10.DBW148..162	

<Nc>.analogInputs[2].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaIn2	
SINUMERIK 840D sl	DB10.DBX146.1	

<Nc>.analogInputs[2].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaIn2	
SINUMERIK 840D sl	DB10.DBX147.1	

<Nc>.analogInputs[2].out.settingValue	**PLC → NC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaIn2	
SINUMERIK 840D sl	DB10.DBW150	

<Nc>.analogInputs[3].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaIn3	
SINUMERIK 840D sl	DB10.DBX146.2	

<Nc>.analogInputs[3].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaIn3	
SINUMERIK 840D sl	DB10.DBX147.2	

<Nc>.analogInputs[3].out.settingValue	**PLC → NC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaIn3	
SINUMERIK 840D sl	DB10.DBW152	

<Nc>.analogInputs[4].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaIn4	
SINUMERIK 840D sl	DB10.DBX146.3	

<Nc>.analogInputs[4].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaIn4	
SINUMERIK 840D sl	DB10.DBX147.3	

<Nc>.analogInputs[4].out.settingValue	**PLC → NC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaIn4	
SINUMERIK 840D sl	DB10.DBW154	

<Nc>.analogInputs[5].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaIn5	
SINUMERIK 840D sl	DB10.DBX146.4	

<Nc>.analogInputs[5].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaIn5	
SINUMERIK 840D sl	DB10.DBX147.4	

<Nc>.analogInputs[5].out.settingValue	**PLC → NC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaIn5	
SINUMERIK 840D sl	DB10.DBW156	

<Nc>.analogInputs[6].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaIn6	
SINUMERIK 840D sl	DB10.DBX146.5	

<Nc>.analogInputs[6].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaIn6	
SINUMERIK 840D sl	DB10.DBX147.5	

<Nc>.analogInputs[6].out.settingValue	**PLC → NC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaIn6	
SINUMERIK 840D sl	DB10.DBW158	

<Nc>.analogInputs[7].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaIn7	
SINUMERIK 840D sl	DB10.DBX146.6	

<Nc>.analogInputs[7].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaIn7	
SINUMERIK 840D sl	DB10.DBX147.6	

<Nc>.analogInputs[7].out.settingValue	**PLC → NC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaIn7	
SINUMERIK 840D sl	DB10.DBW160	

<Nc>.analogInputs[8].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaIn8	
SINUMERIK 840D sl	DB10.DBX146.7	

<Nc>.analogInputs[8].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaIn8	
SINUMERIK 840D sl	DB10.DBX147.7	

<Nc>.analogInputs[8].out.settingValue	**PLC → NC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaIn8	
SINUMERIK 840D sl	DB10.DBW162	

<Nc>.analogOutputs[1].in.setpointValue	**NC → PLC**	**Int**
NC 软件模拟量输出"$A_OUTA[]"的值由 NC 设置，当"改写"功能没有生效时		
SINUMERIK ONE Basic Program	LBP_NC.E_Setval_anaOut1 - 8	
SINUMERIK 840D sl	DB10.DBW210..224	

<Nc>.analogOutputs[2].in.setpointValue	**NC → PLC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.E_Setval_anaOut2	
SINUMERIK 840D sl	DB10.DBW212	

<Nc>.analogOutputs[3].in.setpointValue	**NC → PLC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.E_Setval_anaOut3	
SINUMERIK 840D sl	DB10.DBW214	

<Nc>.analogOutputs[4].in.setpointValue	**NC → PLC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.E_Setval_anaOut4	
SINUMERIK 840D sl	DB10.DBW216	

<Nc>.analogOutputs[5].in.setpointValue	**NC → PLC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.E_Setval_anaOut5	
SINUMERIK 840D sl	DB10.DBW218	

<Nc>.analogOutputs[6].in.setpointValue	**NC → PLC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.E_Setval_anaOut6	
SINUMERIK 840D sl	DB10.DBW220	

<Nc>.analogOutputs[7].in.setpointValue	**NC → PLC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.E_Setval_anaOut7	
SINUMERIK 840D sl	DB10.DBW222	

<Nc>.analogOutputs[8].in.setpointValue	**NC → PLC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.E_Setval_anaOut8	
SINUMERIK 840D sl	DB10.DBW224	

<Nc>.analogOutputs[1].out.settingValue	**PLC → NC**	**Int**
当“analogOutputs[].out.overwrite”或“analogOutputs[].out.target”其中之一生效时，“analogOutputs[].out.settingValue”传送给 NC 硬件模拟量输出 1。		
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaOut1 - 8	
SINUMERIK 840D sl	DB10.DBW170..184	

<Nc>.analogOutputs[1].out.overwrite	**PLC → NC**	**Bool**
出现 0 → 1 上升沿：将“analogOutputs[].out.settingValue”传送给硬件模拟量输出 1，实际的 NC 软件模拟量值 $A_OUTA[1]丢失。		
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_anaOut1 - 8	
SINUMERIK 840D sl	DB10.DBX166.x	

<Nc>.analogOutputs[1].out.target	**PLC → NC**	**Bool**
当“analogOutputs[].out.lock”没有生效时，“analogOutputs[].out.settingValue”传送给硬件模拟量输出 1，实际的 NC 软件模拟量输出值“$A_OUTA[1] ”保持不变。		
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaOut1 - 8	
SINUMERIK 840D sl	DB10.DBX167.x	

<Nc>.analogOutputs[1].out.lock	**PLC → NC**	**Bool**
当“analogOutputs[].out.overwrite”或“analogOutpus[].out.target”其中一个生效时，便 “锁定”NC 硬件模拟量输出 1。		
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaOut1 - 8	
SINUMERIK 840D sl	DB10.DBX168.x	

<Nc>.analogOutputs[2].out.settingValue	**PLC → NC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaOut2	
SINUMERIK 840D sl	DB10.DBW172	

<Nc>.analogOutputs[2].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_anaOut2	
SINUMERIK 840D sl	DB10.DBX166.1	

<Nc>.analogOutputs[2].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaOut2	
SINUMERIK 840D sl	DB10.DBX167.1	

<Nc>.analogOutputs[2].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaOut2	
SINUMERIK 840D sl	DB10.DBX168.1	

<Nc>.analogOutputs[3].out.settingValue	**PLC → NC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaOut3	
SINUMERIK 840D sl	DB10.DBW174	

<Nc>.analogOutputs[3].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_anaOut3	
SINUMERIK 840D sl	DB10.DBX166.2	

<Nc>.analogOutputs[3].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaOut3	
SINUMERIK 840D sl	DB10.DBX167.2	

<Nc>.analogOutputs[3].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaOut3	
SINUMERIK 840D sl	DB10.DBX168.2	

<Nc>.analogOutputs[4].out.settingValue	**PLC → NC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaOut4	
SINUMERIK 840D sl	DB10.DBW176	

<Nc>.analogOutputs[4].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_anaOut4	
SINUMERIK 840D sl	DB10.DBX166.3	

<Nc>.analogOutputs[4].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaOut4	
SINUMERIK 840D sl	DB10.DBX167.3	

<Nc>.analogOutputs[4].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaOut4	
SINUMERIK 840D sl	DB10.DBX168.3	

<Nc>.analogOutputs[5].out.settingValue	**PLC → NC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaOut5	
SINUMERIK 840D sl	DB10.DBW178	

<Nc>.analogOutputs[5].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_anaOut5	
SINUMERIK 840D sl	DB10.DBX166.4	

<Nc>.analogOutputs[5].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaOut5	
SINUMERIK 840D sl	DB10.DBX167.4	

<Nc>.analogOutputs[5].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaOut5	
SINUMERIK 840D sl	DB10.DBX168.4	

<Nc>.analogOutputs[6].out.settingValue	**PLC → NC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaOut6	
SINUMERIK 840D sl	DB10.DBW180	

<Nc>.analogOutputs[6].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_anaOut6	
SINUMERIK 840D sl	DB10.DBX166.5	

<Nc>.analogOutputs[6].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaOut6	
SINUMERIK 840D sl	DB10.DBX167.5	

<Nc>.analogOutputs[6].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaOut6	
SINUMERIK 840D sl	DB10.DBX168.5	

<Nc>.analogOutputs[7].out.settingValue	**PLC → NC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaOut7	
SINUMERIK 840D sl	DB10.DBW182	

<Nc>.analogOutputs[7].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_anaOut7	
SINUMERIK 840D sl	DB10.DBX166.6	

<Nc>.analogOutputs[7].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaOut7	
SINUMERIK 840D sl	DB10.DBX167.6	

<Nc>.analogOutputs[7].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaOut7	
SINUMERIK 840D sl	DB10.DBX168.6	

<Nc>.analogOutputs[8].out.settingValue	**PLC → NC**	**Int**
SINUMERIK ONE Basic Program	LBP_NC.A_Setval_anaOut8	
SINUMERIK 840D sl	DB10.DBW184	

<Nc>.analogOutputs[8].out.overwrite	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_OvMask_anaOut8	
SINUMERIK 840D sl	DB10.DBX166.7	

<Nc>.analogOutputs[8].out.target	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_InMask_anaOut8	
SINUMERIK 840D sl	DB10.DBX167.7	

<Nc>.analogOutputs[8].out.lock	**PLC → NC**	**Bool**
SINUMERIK ONE Basic Program	LBP_NC.A_Disabl_anaOut8	
SINUMERIK 840D sl	DB10.DBX168.7	

<Nc>.collisionProtAreas[0].in.active	**NC → PLC**	**Bool**
所有和该接口信号相连的保护区都已生效		
SINUMERIK ONE Basic Program	LBP_NC.E_CollCheck	
SINUMERIK 840D sl	DB10.DBB226..233	

<Nc>.collisionProtAreas[0].out.activate	**PLC → NC**	**Bool**
激活和该接口信号相连的保护区		
SINUMERIK ONE Basic Program	LBP_NC.A_CollCheck	
SINUMERIK 840D sl	DB10.DBB234..241	

<Nc>.collision.out.deactivateMachineAreasAuto	**PLC → NC**	**Bool**
请求在自动方式中关闭“机床”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.A_CA_DisableAutoMach	
SINUMERIK 840D sl	DB10.DBX58.0	

<Nc>.collision.out.deactivateToolAreasAuto	**PLC → NC**	**Bool**
请求在自动方式中关闭“刀具”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.A_CA_DisableAutoTools	
SINUMERIK 840D sl	DB10.DBX58.1	

<Nc>.collision.out.deactivateFixtureAreasAuto	**PLC → NC**	**Bool**
请求在自动方式中关闭“夹具”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.A_CA_DisableAutoWorkh	
SINUMERIK 840D sl	DB10.DBX58.2	

<Nc>.collision.out.deactivateWorkpieceAreasAuto	**PLC → NC**	**Bool**
请求在自动方式中关闭“工件”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.A_CA_DisableAutoWorkp	
SINUMERIK 840D sl	DB10.DBX58.3	

<Nc>.collision.out.deactivateMachineAreasJog	**PLC → NC**	**Bool**
请求在 JOG 方式中关闭“机床”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.A_CA_DisableJogMach	
SINUMERIK 840D sl	DB10.DBX58.4	

<Nc>.collision.out.deactivateToolAreasJog	**PLC → NC**	**Bool**
请求在 JOG 方式中关闭“刀具”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.A_CA_DisableJogTools	
SINUMERIK 840D sl	DB10.DBX58.5	

<Nc>.collision.out.deactivateFixtureAreasJog	**PLC → NC**	**Bool**
请求在 JOG 方式中关闭“夹具”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.A_CA_DisableJogWorkh	
SINUMERIK 840D sl	DB10.DBX58.6	

<Nc>.collision.out.deactivateWorkpieceAreasJog	**PLC → NC**	**Bool**
请求在 JOG 方式中关闭“工件”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.A_CA_DisableJogWorkp	
SINUMERIK 840D sl	DB10.DBX58.7	

12.2.6 NCK_typeToolholder

<Toolholder>.config
刀柄的配置数据

<Toolholder>.command
刀具指令信号

<Toolholder>.ack
对刀具指令的应答，只有当请求生效时才有效

<Toolholder>.command.in	**NC → PLC**
从刀具管理收到的信号	

<Toolholder>.ack.in	**NC → PLC**
收到的换刀应答信号	

<Toolholder>.ack.out	**PLC → NC**
发送的换刀应答信号	

<Toolholder>.command.in.request	**NC → PLC**	**Bool**
NC 请求换刀，指令信号有效		
SINUMERIK ONE Basic Program	LBP_TMSpindleIFNo	
SINUMERIK ONE Basic Program	LBP_TMTurretIFNo	
SINUMERIK 840D sl	DB72.DBX0.0..1.7	
SINUMERIK 840D sl	DB73.DBX0.0..1.7	

<Toolholder>.command.in.multitoolRequest	**NC → PLC**	**Bool**
已请求多刀换刀，多刀信号有效。只有当“<toolholder>.command.in.request”置位时，该信号才有效		
SINUMERIK ONE Basic Program	LBP_TMSpindleDataInDB1072	

<Toolholder>.command.in.multitoolRequest	**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_TMTurretDataInDB1073	
SINUMERIK 840D sl	DB72.DBX5.7	
SINUMERIK 840D sl	DB73.DBX5.7	

<Toolholder>.command.in.basic	**NC → PLC**
通用指令信号。只有当"<toolholder>.command.in.request"置位时，该信号才生效	

<Toolholder>.command.in.options	**NC → PLC**
指令的选项。只有当"<toolholder>.command.in.request"置位时，它才有效。	

<Toolholder>.command.in.toolProperties	**NC → PLC**
刀柄中新刀具的属性。只有当"<toolholder>.command.in.request"置位时，它才有效。	

<Toolholder>.command.in.toolStatus	**NC → PLC**
刀柄中新刀具的状态。只有当"<toolholder>.command.in.request"置位时，它才有效。	

<Toolholder>.command.in.userParameters	**NC → PLC**	**DInt**
使用指令($P_VDITCP)传递的用户自定义参数。只有当"<toolholder>.command.in.request"置位时，它才有效。		
SINUMERIK ONE Basic Program	LBP_TMSpindleVdiTcp0 - VdiTcp2	
SINUMERIK ONE Basic Program	LBP_TMTurretVdiTcp0 - VdiTcp2	
SINUMERIK 840D sl	DB72.DBD8, .DBD12, .DBD16	
SINUMERIK 840D sl	DB73.DBD8, .DBD812, .DBD816	

<Toolholder>.command.in.multitoolCommand	**NC → PLC**
通用多刀指令信号。只有当"<toolholder>.command.in.multitoolRequest"置位时，该信号才生效	

<Toolholder>.command.in.multitoolProperties	**NC → PLC**
刀柄中新多刀的属性。只有当"<toolholder>.command.in.multitoolRequest"置位时，该信号才有效	

<Toolholder>.command.in.multitoolStatus	**NC → PLC**
刀柄中新多刀的状态。只有当"<toolholder>.command.in.multitoolRequest"置位时，该信号才有效	

<Toolholder>.config.registered	**NC → PLC**	**Bool**
刀柄已注册		

<Toolholder>.config.toolholder	**NC → PLC**	**UInt**
刀柄编号		

<Toolholder>.config.toaUnit	**NC → PLC**	**USInt**
对应 TO 范围的编号		
SINUMERIK ONE Basic Program	LBP_TMSpindleTMNo	
SINUMERIK ONE Basic Program	LBP_TMTurretTMNo	
SINUMERIK 840D sl	DB72.DBB7	
SINUMERIK 840D sl	DB73.DBB7	

<Toolholder>.config.bufferMagazine	**NC → PLC**	**UInt**
刀柄所在刀库的编号，通常是周转刀库		

<Toolholder>.config.bufferLocation	**NC → PLC**	**UInt**
刀库中刀柄的位置编号		

<Toolholder>.command.in.basic.ncCommand	**NC → PLC**	**USInt**
内部 NC 指令编号		

<Toolholder>.command.in.basic.perform	**NC → PLC**	**Bool**
“执行换刀”已请求		
SINUMERIK ONE Basic Program	LBP_TMSpindlePerform	
SINUMERIK ONE Basic Program	LBP_TMTurretPerform	
SINUMERIK 840D sl	DB72.DBX4.1	
SINUMERIK 840D sl	DB73.DBX4.1	

<Toolholder>.command.in.basic.prepare	**NC → PLC**	**Bool**
“准备换刀”已请求		
SINUMERIK ONE Basic Program	LBP_TMSpindlePrepare	
SINUMERIK ONE Basic Program	LBP_TMTurretf0_2	
SINUMERIK 840D sl	DB72.DBX4.2	
SINUMERIK 840D sl	DB73.DBX4.2	

<Toolholder>.command.in.basic.position	**NC → PLC**	**Bool**
已请求“刀库定位”到刀柄上		
SINUMERIK ONE Basic Program	LBP_TMLoadPositioning	

<Toolholder>.command.in.basic.position		**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_TMTurretTO		
SINUMERIK 840D sl	DB71.DBX4.3		
SINUMERIK 840D sl	DB73.DBX4.3		

<Toolholder>.command.in.basic.chan		**NC → PLC**	**USInt**
对应的通道号			
SINUMERIK ONE Basic Program	LBP_TMSpindleChannel		
SINUMERIK ONE Basic Program	LBP_TMTurretChannel		
SINUMERIK 840D sl	DB72.DBB6		
SINUMERIK 840D sl	DB73.DBB6		

<Toolholder>.command.in.basic.newTool		**NC → PLC**	**UInt**
新建刀具的 T 号			
SINUMERIK ONE Basic Program	LBP_TMSpindleTNo		
SINUMERIK ONE Basic Program	LBP_TMTurretTNo		
SINUMERIK 840D sl	DB72.DBW44		
SINUMERIK 840D sl	DB73.DBW44		

<Toolholder>.command.in.basic.newToolTargetMagazine		**NC → PLC**	**Int**
新刀具目标刀库的编号；或是目标刀柄的刀库编号			
SINUMERIK ONE Basic Program	LBP_TMSpindleBufferIdent		
SINUMERIK ONE Basic Program	LBP_TMTurretBufferIdent		
SINUMERIK 840D sl	DB72.DBW20		
SINUMERIK 840D sl	DB73.DBW20		

<Toolholder>.command.in.basic.newToolTargetLocation		**NC → PLC**	**Int**
新刀具目标位置的编号；或是目标刀柄的位置编号			
SINUMERIK ONE Basic Program	LBP_TMSpindleBufferNo		
SINUMERIK ONE Basic Program	LBP_TMTurretBufferNo		
SINUMERIK 840D sl	DB72.DBW22		
SINUMERIK 840D sl	DB73.DBW22		

<Toolholder>.command.in.basic.newToolSourceMagazine	**NC → PLC**	**Int**
新刀具当前刀库的编号；或是需要定位到刀柄的刀库的编号		
SINUMERIK ONE Basic Program	LBP_TMSpindleSMag	
SINUMERIK ONE Basic Program	LBP_TMTurretSMag	
SINUMERIK 840D sl	DB72.DBW24	
SINUMERIK 840D sl	DB73.DBW24	

<Toolholder>.command.in.basic.newToolSourceLocation	**NC → PLC**	**Int**
新刀具当前位置的编号；或是需要定位到刀柄的位置的编号		
SINUMERIK ONE Basic Program	LBP_TMSpindleSLoc	
SINUMERIK ONE Basic Program	LBP_TMTurretSLoc	
SINUMERIK 840D sl	DB72.DBW26	
SINUMERIK 840D sl	DB73.DBW26	

<Toolholder>.command.in.basic.oldToolSourceMagazine	**NC → PLC**	**Int**
旧刀具当前刀库的编号		

<Toolholder>.command.in.basic.oldToolSourceLocation	**NC → PLC**	**Int**
旧刀具当前位置的编号		

<Toolholder>.command.in.basic.oldToolTargetMagazine	**NC → PLC**	**Int**
旧刀具目标刀库的编号		
SINUMERIK ONE Basic Program	LBP_TMSpindleTMag	
SINUMERIK ONE Basic Program	LBP_TMTurretTMag	
SINUMERIK 840D sl	DB72.DBW28	
SINUMERIK 840D sl	DB73.DBW28	

<Toolholder>.command.in.basic.oldToolTargetLocation	**NC → PLC**	**Int**
旧刀具目标位置的编号		
SINUMERIK ONE Basic Program	LBP_TMSpindleTLoc	

<Toolholder>.command.in.basic.oldToolTargetLocation	**NC → PLC**	**Int**
SINUMERIK ONE Basic Program	LBP_TMTurretTLoc	
SINUMERIK 840D sl	DB72.DBW30	
SINUMERIK 840D sl	DB73.DBW30	

<Toolholder>.command.in.options.noNewTool	**NC → PLC**	**Bool**
没有新刀具可执行换刀，程序中写入了 T0		
SINUMERIK ONE Basic Program	LBP_TMSpindleT0	
SINUMERIK ONE Basic Program	LBP_TMTurretT0	
SINUMERIK 840D sl	DB72.DBX4.3	
SINUMERIK 840D sl	DB73.DBX4.3	

<Toolholder>.command.in.options.oldToolNotInToolholder	**NC → PLC**	**Bool**
旧刀具不在位于刀柄中		
SINUMERIK ONE Basic Program	LBP_TMSpindleOldTinDBW42	
SINUMERIK ONE Basic Program	LBP_TMTurretf0_4	
SINUMERIK 840D sl	DB72.DBX4.4	
SINUMERIK 840D sl	DB73.DBX4.4	

<Toolholder>.command.in.options.newManualTool	**NC → PLC**	**Bool**
新刀具须手动装入（手动刀具）		
SINUMERIK ONE Basic Program	LBP_TMSpindleManTIn	
SINUMERIK ONE Basic Program	LBP_TMTurretf0_5	
SINUMERIK 840D sl	DB72.DBX4.5	
SINUMERIK 840D sl	DB73.DBX4.5	

<Toolholder>.command.in.options.oldManualTool	**NC → PLC**	**Bool**
旧刀具须手动更换（手动刀具）		
SINUMERIK ONE Basic Program	LBP_TMSpindleManTOut	
SINUMERIK ONE Basic Program	LBP_TMTurretManTOut	
SINUMERIK 840D sl	DB72.DBX4.6	
SINUMERIK 840D sl	DB73.DBX4.6	

<Toolholder>.command.in.options.byNcProg	**NC → PLC**	**Bool**
一个数控程序触发了换刀请求		
SINUMERIK ONE Basic Program	LBP_TMLoadPosbyProgram	
SINUMERIK 840D sl	DB71.DBX4.4	

<Toolholder>.command.in.options.toolRemainsInToolholder	**NC → PLC**	**Bool**
新刀已位于刀柄中		
SINUMERIK ONE Basic Program	LBP_TMSpindleSpiTRSpi	
SINUMERIK ONE Basic Program	LBP_TMTurretf0_7	
SINUMERIK 840D sl	DB72.DBX4.7	
SINUMERIK 840D sl	DB73.DBX4.7	

<Toolholder>.command.in.options.rejectingToolAllowed	**NC → PLC**	**Bool**
应拒绝系统建议的刀具		

<Toolholder>.command.in.toolProperties.toolLocationType	**NC → PLC**	**UInt**
刀具的位置类型		
SINUMERIK ONE Basic Program	LBP_TMSpindleLocType	
SINUMERIK ONE Basic Program	LBP_TMTurretLocType	
SINUMERIK 840D sl	DB72.DBW32	
SINUMERIK 840D sl	DB73.DBW32	

<Toolholder>.command.in.toolProperties.toolSizeLeft	**NC → PLC**	**UInt**
刀具在左侧占用的半刀位数量		
SINUMERIK ONE Basic Program	LBP_TMSpindleSLeft	
SINUMERIK ONE Basic Program	LBP_TMTurretSLeft	
SINUMERIK 840D sl	DB72.DBW34	
SINUMERIK 840D sl	DB73.DBW34	

<Toolholder>.command.in.toolProperties.toolSizeRight	**NC → PLC**	**UInt**
刀具在右侧占用的半刀位数量		
SINUMERIK ONE Basic Program	LBP_TMSpindleSRight	

<Toolholder>.command.in.toolProperties.toolSizeRight	**NC → PLC**	**UInt**
SINUMERIK ONE Basic Program	LBP_TMTurretSRight	
SINUMERIK 840D sl	DB72.DBW36	
SINUMERIK 840D sl	DB73.DBW36	

<Toolholder>.command.in.toolProperties.toolSizeTop	**NC → PLC**	**UInt**
刀具在顶部占用的半刀位数量		
SINUMERIK ONE Basic Program	LBP_TMSpindleSTop	
SINUMERIK ONE Basic Program	LBP_TMTurretSTop	
SINUMERIK 840D sl	DB72.DBW38	
SINUMERIK 840D sl	DB73.DBW38	

<Toolholder>.command.in.toolProperties.toolSizeBottom	**NC → PLC**	**UInt**
刀具在底部占用的半刀位数量		
SINUMERIK ONE Basic Program	LBP_TMSpindleSBottom	
SINUMERIK ONE Basic Program	LBP_TMTurretSBottom	
SINUMERIK 840D sl	DB72.DBW40	
SINUMERIK 840D sl	DB73.DBW40	

<Toolholder>.command.in.toolProperties.toolOriginMagazine	**NC → PLC**	**UInt**
自有刀库的编号		
SINUMERIK ONE Basic Program	LBP_TMSpindleMyMag	
SINUMERIK 840D sl	DB72.DBW48	

<Toolholder>.command.in.toolProperties.toolOriginLocation	**NC → PLC**	**UInt**
刀库中自有位置的编号		
SINUMERIK ONE Basic Program	LBP_TMSpindleMyPlace	
SINUMERIK 840D sl	DB72.DBW50	

<Toolholder>.command.in.toolStatus.active	**NC → PLC**	**Bool**
刀具生效		
SINUMERIK ONE Basic Program	LBP_TMSpindleActive	

<Toolholder>.command.in.toolStatus.active		**NC → PLC**	**Bool**
SINUMERIK ONE Basic Program	LBP_TMTurretActive		
SINUMERIK 840D sl	DB72.DBX43.0		
SINUMERIK 840D sl	DB73.DBX43.0		

<Toolholder>.command.in.toolStatus.enabled		**NC → PLC**	**Bool**
刀具使能			
SINUMERIK ONE Basic Program	LBP_TMSpindleEnabled		
SINUMERIK ONE Basic Program	LBP_TMTurretEnabled		
SINUMERIK 840D sl	DB72.DBX43.1		
SINUMERIK 840D sl	DB73.DBX43.1		

<Toolholder>.command.in.toolStatus.disabled		**NC → PLC**	**Bool**
刀具禁用			
SINUMERIK ONE Basic Program	LBP_TMSpindleDisabled		
SINUMERIK ONE Basic Program	LBP_TMTurretDisabled		
SINUMERIK 840D sl	DB72.DBX43.2		
SINUMERIK 840D sl	DB73.DBX43.2		

<Toolholder>.command.in.toolStatus.measured		**NC → PLC**	**Bool**
刀具已测量			
SINUMERIK ONE Basic Program	LBP_TMSpindleMeasured		
SINUMERIK ONE Basic Program	LBP_TMTurretMeasured		
SINUMERIK 840D sl	DB72.DBX43.3		
SINUMERIK 840D sl	DB73.DBX43.3		

<Toolholder>.command.in.toolStatus.prewarning		**NC → PLC**	**Bool**
已达到刀具预警限值			
SINUMERIK ONE Basic Program	LBP_TMSpindlePrewarn		
SINUMERIK ONE Basic Program	LBP_TMTurretPrewarn		
SINUMERIK 840D sl	DB72.DBX43.4		
SINUMERIK 840D sl	DB73.DBX43.4		

<Toolholder>.command.in.toolStatus.changeActive	**NC → PLC**	**Bool**
正在换刀		
SINUMERIK ONE Basic Program	LBP_TMSpindleChange	
SINUMERIK ONE Basic Program	LBP_TMTurretChange	
SINUMERIK 840D sl	DB72.DBX43.5	
SINUMERIK 840D sl	DB73.DBX43.5	

<Toolholder>.command.in.toolStatus.fixedLocation	**NC → PLC**	**Bool**
刀具位置固定		
SINUMERIK ONE Basic Program	LBP_TMSpindleFixedLoc	
SINUMERIK ONE Basic Program	LBP_TMTurretFixedLoc	
SINUMERIK 840D sl	DB72.DBX43.6	
SINUMERIK 840D sl	DB73.DBX43.6	

<Toolholder>.command.in.toolStatus.used	**NC → PLC**	**Bool**
刀具已使用		
SINUMERIK ONE Basic Program	LBP_TMSpindleUsed	
SINUMERIK ONE Basic Program	LBP_TMTurretUsed	
SINUMERIK 840D sl	DB72.DBX43.7	
SINUMERIK 840D sl	DB73.DBX43.7	

<Toolholder>.command.in.toolStatus.toolInBuffer	**NC → PLC**	**Bool**
刀具位于周转刀库中		
SINUMERIK ONE Basic Program	LBP_TMSpindleTInBuffer	
SINUMERIK ONE Basic Program	LBP_TMTurretTInBuffer	
SINUMERIK 840D sl	DB72.DBX42.0	
SINUMERIK 840D sl	DB73.DBX42.0	

<Toolholder>.command.in.toolStatus.disableStateIgnored	**NC → PLC**	**Bool**
忽略刀具的禁用状态		
SINUMERIK ONE Basic Program	LBP_TMSpindleIgnoreDisabled	
SINUMERIK ONE Basic Program	LBP_TMTurretIgnoreDisabled	
SINUMERIK 840D sl	DB72.DBX42.1	
SINUMERIK 840D sl	DB73.DBX42.1	

<Toolholder>.command.in.toolStatus.toBeUnloaded	**NC → PLC**	**Bool**
刀具必须卸下		
SINUMERIK ONE Basic Program	LBP_TMSpindleUnload	
SINUMERIK ONE Basic Program	LBP_TMTurretUnload	
SINUMERIK 840D sl	DB72.DBX42.2	
SINUMERIK 840D sl	DB73.DBX42.2	

<Toolholder>.command.in.toolStatus.toBeLoaded	**NC → PLC**	**Bool**
刀具必须装入		
SINUMERIK ONE Basic Program	LBP_TMSpindleLoad	
SINUMERIK ONE Basic Program	LBP_TMTurretLoad	
SINUMERIK 840D sl	DB72.DBX42.3	
SINUMERIK 840D sl	DB73.DBX42.3	

<Toolholder>.command.in.toolStatus.masterTool	**NC → PLC**	**Bool**
刀具为主刀具		
SINUMERIK ONE Basic Program	LBP_TMSpindleMasterTool	
SINUMERIK ONE Basic Program	LBP_TMTurretMasterTool	
SINUMERIK 840D sl	DB72.DBX42.4	
SINUMERIK 840D sl	DB73.DBX42.4	

<Toolholder>.command.in.toolStatus.changeOneToOne	**NC → PLC**	**Bool**
1:1 换刀		
SINUMERIK ONE Basic Program	LBP_TMSpindleExhange_1to1	
SINUMERIK ONE Basic Program	LBP_TMTurretExhange_1to1	
SINUMERIK 840D sl	DB72.DBX42.6	
SINUMERIK 840D sl	DB73.DBX42.6	

<Toolholder>.command.in.toolStatus.manualTool	**NC → PLC**	**Bool**
刀具为手动刀具		
SINUMERIK ONE Basic Program	LBP_TMSpindleHandTool	
SINUMERIK ONE Basic Program	LBP_TMTurretHandTool	
SINUMERIK 840D sl	DB72.DBX42.7	
SINUMERIK 840D sl	DB73.DBX42.7	

<Toolholder>.command.in.toolStatus.toolAtDisabledLocation	**NC → PLC**	**Bool**
刀具位于禁用刀位上		

<Toolholder>.command.in.multitoolCommand.position	**NC → PLC**	**Bool**
多刀定位请求		
SINUMERIK ONE Basic Program	LBP_TMLoadPositioningMT	
SINUMERIK 840D sl	DB71.DBX4.5	

<Toolholder>.command.in.multitoolCommand.newTool	**NC → PLC**	**UInt**
新多刀的 T 号		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTNewMTNo	
SINUMERIK ONE Basic Program	LBP_TMTurretMTNewMTNo	
SINUMERIK 840D sl	DB1072.DBW8	
SINUMERIK 840D sl	DB1073.DBW8	

<Toolholder>.command.in.multitoolCommand.newToolLocation	**NC → PLC**	**Int**
多刀内新刀具的位置编号		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTNewPlace	
SINUMERIK ONE Basic Program	LBP_TMTurretMTNewPlace	
SINUMERIK 840D sl	DB1072.DBW10	
SINUMERIK 840D sl	DB1073.DBW10	

<Toolholder>.command.in.multitoolCommand.oldTool	**NC → PLC**	**UInt**
旧多刀的 T 号		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTOldMTNo	
SINUMERIK ONE Basic Program	LBP_TMTurretMTOldMTNo	
SINUMERIK 840D sl	DB1072.DBW12	
SINUMERIK 840D sl	DB1073.DBW12	

<Toolholder>.command.in.multitoolCommand.oldToolLocation	**NC → PLC**	**Int**
多刀内旧刀具的位置编号		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTOldPlace	
SINUMERIK ONE Basic Program	LBP_TMTurretMTOldPlace	
SINUMERIK 840D sl	DB1072.DBW14	
SINUMERIK 840D sl	DB1073.DBW14	

<Toolholder>.command.in.multitoolProperties.distanceType	**NC → PLC**	**UInt**
多刀的距离编码方式		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTDistanceCoding	
SINUMERIK ONE Basic Program	LBP_TMTurretMTDistanceCoding	
SINUMERIK 840D sl	DB1072.DBW0	
SINUMERIK 840D sl	DB1073.DBW0	

<Toolholder>.command.in.multitoolProperties.distance	**NC → PLC**	**Real**
多刀占据的各个位置的距离采用距离编码方式		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTDistance	
SINUMERIK ONE Basic Program	LBP_TMTurretMTDistance	
SINUMERIK 840D sl	DB1072.DBD4	
SINUMERIK 840D sl	DB1073.DBD4	

<Toolholder>.command.in.multitoolProperties.numberOfLocations	**NC → PLC**	**UInt**
多刀占据的位置数量		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTPlaces	
SINUMERIK ONE Basic Program	LBP_TMTurretMTPlaces	
SINUMERIK 840D sl	DB1072.DBW2	
SINUMERIK 840D sl	DB1073.DBW2	

<Toolholder>.command.in.multitoolProperties.toolLocationType	**NC → PLC**	**UInt**
多刀的位置类型		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTLocType	

<Toolholder>.command.in.multitoolProperties.toolLocationType	**NC → PLC**	**UInt**
SINUMERIK ONE Basic Program	LBP_TMTurretMTLocType	
SINUMERIK 840D sl	DB1072.DBW16	
SINUMERIK 840D sl	DB1073.DBW16	

<Toolholder>.command.in.multitoolProperties.toolSizeLeft	**NC → PLC**	**UInt**
多刀在左侧占用的半刀位数量		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTSleft	
SINUMERIK ONE Basic Program	LBP_TMTurretMTSleft	
SINUMERIK 840D sl	DB1072.DBW18	
SINUMERIK 840D sl	DB1073.DBW18	

<Toolholder>.command.in.multitoolProperties.toolSizeRight	**NC → PLC**	**UInt**
多刀在右侧占用的半刀位数量		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTSRight	
SINUMERIK ONE Basic Program	LBP_TMTurretMTSRight	
SINUMERIK 840D sl	DB1072.DBW20	
SINUMERIK 840D sl	DB1073.DBW20	

<Toolholder>.command.in.multitoolProperties.toolSizeTop	**NC → PLC**	**UInt**
多刀在顶部占用的半刀位数量		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTSTop	
SINUMERIK ONE Basic Program	LBP_TMTurretMTSTop	
SINUMERIK 840D sl	DB1072.DBW22	
SINUMERIK 840D sl	DB1073.DBW22	

<Toolholder>.command.in.multitoolProperties.toolSizeBottom	**NC → PLC**	**UInt**
多刀在底部占用的半刀位数量		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTSBottom	
SINUMERIK ONE Basic Program	LBP_TMTurretMTSBottom	

<Toolholder>.command.in.multitoolProperties.toolSizeBottom		**NC → PLC**	**UInt**
SINUMERIK 840D sl	DB1072.DBW24		
SINUMERIK 840D sl	DB1073.DBW24		

<Toolholder>.command.in.multitoolProperties.toolOriginMagazine		**NC → PLC**	**UInt**
多刀自有刀库的编号			
SINUMERIK ONE Basic Program	LBP_TMSpindleMTMyMag		
SINUMERIK ONE Basic Program	LBP_TMTurretMTMyMag		
SINUMERIK 840D sl	DB1072.DBW32		
SINUMERIK 840D sl	DB1073.DBW32		

<Toolholder>.command.in.multitoolProperties.toolOriginLocation		**NC → PLC**	**UInt**
刀库中自有位置的编号			
SINUMERIK ONE Basic Program	LBP_TMSpindleMTMyPlace		
SINUMERIK ONE Basic Program	LBP_TMTurretMTMyPlace		
SINUMERIK 840D sl	DB1072.DBW34		
SINUMERIK 840D sl	DB1073.DBW34		

<Toolholder>.command.in.multitoolStatus.active		**NC → PLC**	**Bool**
刀具生效			
SINUMERIK ONE Basic Program	LBP_TMSpindleMTActive		
SINUMERIK ONE Basic Program	LBP_TMTurretMTActive		
SINUMERIK 840D sl	DB1072.DBX27.0		
SINUMERIK 840D sl	DB1073.DBX27.0		

<Toolholder>.command.in.multitoolStatus.enabled		**NC → PLC**	**Bool**
刀具使能			
SINUMERIK ONE Basic Program	LBP_TMSpindleMTEnabled		
SINUMERIK ONE Basic Program	LBP_TMTurretMTEnabled		
SINUMERIK 840D sl	DB1072.DBX27.1		
SINUMERIK 840D sl	DB1073.DBX27.1		

<Toolholder>.command.in.multitoolStatus.disabled	**NC → PLC**	**Bool**
刀具禁用		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTDisabled	
SINUMERIK ONE Basic Program	LBP_TMTurretMTDisabled	
SINUMERIK 840D sl	DB1072.DBX27.2	
SINUMERIK 840D sl	DB1073.DBX27.2	

<Toolholder>.command.in.multitoolStatus.measured	**NC → PLC**	**Bool**
刀具已测量		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTMeasured	
SINUMERIK ONE Basic Program	LBP_TMTurretMTMeasured	
SINUMERIK 840D sl	DB1072.DBX27.3	
SINUMERIK 840D sl	DB1073.DBX27.3	

<Toolholder>.command.in.multitoolStatus.prewarning	**NC → PLC**	**Bool**
已达到刀具预警限值		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTPrewarn	
SINUMERIK ONE Basic Program	LBP_TMTurretMTPrewarn	
SINUMERIK 840D sl	DB1072.DBX27.4	
SINUMERIK 840D sl	DB1073.DBX27.4	

<Toolholder>.command.in.multitoolStatus.changeActive	**NC → PLC**	**Bool**
正在换刀		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTChange	
SINUMERIK ONE Basic Program	LBP_TMTurretMTChange	
SINUMERIK 840D sl	DB1072.DBX27.5	
SINUMERIK 840D sl	DB1073.DBX27.5	

<Toolholder>.command.in.multitoolStatus.fixedLocation	**NC → PLC**	**Bool**
刀具位置固定		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTFixedLoc	
SINUMERIK ONE Basic Program	LBP_TMTurretMTFixedLoc	

<Toolholder>.command.in.multitoolStatus.fixedLocation	**NC → PLC**	**Bool**
SINUMERIK 840D sl	DB1072.DBX27.6	
SINUMERIK 840D sl	DB1073.DBX27.6	

<Toolholder>.command.in.multitoolStatus.used	**NC → PLC**	**Bool**
刀具已使用		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTUsed	
SINUMERIK ONE Basic Program	LBP_TMTurretMTUsed	
SINUMERIK 840D sl	DB1072.DBX27.7	
SINUMERIK 840D sl	DB1073.DBX27.7	

<Toolholder>.command.in.multitoolStatus.toolInBuffer	**NC → PLC**	**Bool**
刀具位于周转刀库中		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTTInBuffer	
SINUMERIK ONE Basic Program	LBP_TMTurretMTTInBuffer	
SINUMERIK 840D sl	DB1072.DBX26.0	
SINUMERIK 840D sl	DB1073.DBX26.0	

<Toolholder>.command.in.multitoolStatus.disableStateIgnored	**NC → PLC**	**Bool**
忽略刀具的禁用状态		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTIgnoreDisabled	
SINUMERIK ONE Basic Program	LBP_TMTurretMTIgnoreDisabled	
SINUMERIK 840D sl	DB1072.DBX26.1	
SINUMERIK 840D sl	DB1073.DBX26.1	

<Toolholder>.command.in.multitoolStatus.toBeUnloaded	**NC → PLC**	**Bool**
刀具必须卸下		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTUnload	
SINUMERIK ONE Basic Program	LBP_TMTurretMTUnload	
SINUMERIK 840D sl	DB1072.DBX26.2	
SINUMERIK 840D sl	DB1073.DBX26.2	

<Toolholder>.command.in.multitoolStatus.toBeLoaded	**NC → PLC**	**Bool**
刀具必须装入		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTLoad	
SINUMERIK ONE Basic Program	LBP_TMTurretMTLoad	
SINUMERIK 840D sl	DB1072.DBX26.3	
SINUMERIK 840D sl	DB1073.DBX26.3	

<Toolholder>.command.in.multitoolStatus.masterTool	**NC → PLC**	**Bool**
刀具为主刀具		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTMasterTool	
SINUMERIK ONE Basic Program	LBP_TMTurretMTMasterTool	
SINUMERIK 840D sl	DB1072.DBX26.4	
SINUMERIK 840D sl	DB1073.DBX26.4	

<Toolholder>.command.in.multitoolStatus.changeOneToOne	**NC → PLC**	**Bool**
1:1 换刀		

<Toolholder>.command.in.multitoolStatus.manualTool	**NC → PLC**	**Bool**
刀具为手动刀具		
SINUMERIK ONE Basic Program	LBP_TMSpindleMTHandTool	
SINUMERIK ONE Basic Program	LBP_TMTurretMTHandTool	
SINUMERIK 840D sl	DB1072.DBX26.7	
SINUMERIK 840D sl	DB1073.DBX26.7	

<Toolholder>.command.in.multitoolStatus.toolAtDisabledLocation	**NC → PLC**	**Bool**
刀具位于禁用刀位上		

<Toolholder>.ack.in.busy	**NC → PLC**	**Bool**
正在应答		

<Toolholder>.ack.in.done	**NC → PLC**	**Bool**
应答成功		

<Toolholder>.ack.in.error	**NC → PLC**	**Bool**
应答被拒，查看“状态”		

<Toolholder>.ack.in.status	**NC → PLC**	**Word**
应答失败时的状态信息		

<Toolholder>.ack.out.execute	**PLC → NC**	**Bool**
开始应答换刀		

<Toolholder>.ack.out.totalComplete	**PLC → NC**	**Bool**
发出应答“操作成功”，状态 1 或 5 被使用，刀具位置参数和状态参数被忽略		
SINUMERIK ONE Basic Program	LBP_TMSpindleQuit	
SINUMERIK ONE Basic Program	LBP_TMTurretQuit	
SINUMERIK 840D sl	DB72.DBX2.0..3.7	
SINUMERIK 840D sl	DB73.DBX2.0..3.7	

<Toolholder>.ack.out.totalFailed	**PLC → NC**	**Bool**
发出应答“操作失败”，状态 3 被使用，刀具位置参数和状态参数被忽略		
SINUMERIK ONE Basic Program	LBP_TMSpindleQuit + PerfStatusError	
SINUMERIK ONE Basic Program	LBP_TMTurretQuit + PerfStatusError	
SINUMERIK 840D sl	DB72.DBX5.0	
SINUMERIK 840D sl	DB73.DBX5.0	

<Toolholder>.ack.out.newToolMagazine	**PLC → NC**	**Int**
新刀具当前所在的刀库编号		

<Toolholder>.ack.out.newToolLocation	**PLC → NC**	**Int**
新刀具在刀库中的当前位置编号		

<Toolholder>.ack.out.newMultitoolLocation	**PLC → NC**	**Int**
多刀内新刀具的当前位置编号		

<Toolholder>.ack.out.oldToolMagazine	**PLC → NC**	**Int**
旧刀具当前所在的刀库编号		

<Toolholder>.ack.out.oldToolLocation	**PLC → NC**	**Int**
旧刀具在刀库中的当前位置编号		

<Toolholder>.ack.out.status	**PLC → NC**	**USInt**
换刀状态		

12.2.7 NCK_typeRelocate

<Relocate>.config
移位接口的配置数据

<Relocate>.command
刀具指令信号

<Relocate>.ack
对刀具指令的应答。只有当"<relocate>.command.in.request"置位时，它才有效。

<Relocate>.command.in	**NC → PLC**
从刀具管理收到的信号	

<Relocate>.ack.in	**NC → PLC**
收到的传刀应答信号	

<Relocate>.ack.out	**PLC → NC**
发送的传刀应答信号	

<Relocate>.command.in.request	**NC → PLC**	**Bool**
NC 请求移刀，该指令信号有效		
SINUMERIK ONE Basic Program	LBP_TMLoadIFNo	
SINUMERIK 840D sl	DB71.DBX0.0..1.7	

<Relocate>.command.in.multitoolRequest	**NC → PLC**	**Bool**
已请求多刀移刀，多刀的指令信号有效。只有当"<relocate>.command.in.request"置位时，该信号才有效		
SINUMERIK ONE Basic Program	LBP_TMLoadDataInDB1071	
SINUMERIK 840D sl	DB71.DBX5.7	

<Relocate>.command.in.basic	**NC → PLC**	
通用指令信号。只有当"<relocate>.command.in.request"置位时，该信号才生效		

<Relocate>.command.in.options	**NC → PLC**	
指令的选项。只有当"<relocate>.command.in.request"置位时，它才有效。		

<Relocate>.command.in.toolProperties	**NC → PLC**	
刀具属性。只有当"<relocate>.command.in.request"置位时，该信号才有效。		

<Relocate>.command.in.toolStatus	**NC → PLC**	
刀具状态。只有当"<relocate>.command.in.request"置位时，它才有效。		

<Relocate>.command.in.userParameters	**NC → PLC**	**DInt**
使用指令($P_VDITCP)传递的用户自定义参数。只有当"<relocate>.command.in.request"置位时，它才有效。		
SINUMERIK ONE Basic Program	LBP_TMLoadreserved1 - 3	
SINUMERIK 840D sl	DB71.DBD8, .DBD12, .DBD16	

<Relocate>.command.in.multitoolProperties	**NC → PLC**	
多刀属性，只有当"<relocate>.command.in.request"置位时，该信号才有效		

<Relocate>.config.registered	**NC → PLC**	**Bool**
已注册移刀用接口		

<Relocate>.config.toaUnit	**NC → PLC**	**USInt**
对应 TO 范围的编号		
SINUMERIK ONE Basic Program	LBP_TMLoadTMNo	
SINUMERIK 840D sl	DB71.DBB7	

<Relocate>.command.in.basic.ncCommand	**NC → PLC**	**USInt**
内部 NC 指令编号		

<Relocate>.command.in.basic.relocate	**NC → PLC**	**Bool**
已请求移刀		
SINUMERIK ONE Basic Program	LBP_TMLoadReloading	
SINUMERIK 840D sl	DB71.DBX4.2	

<Relocate>.command.in.basic.chan		**NC → PLC**	**USInt**
对应的通道号			
SINUMERIK ONE Basic Program	LBP_TMLoadChannel		
SINUMERIK 840D sl	DB71.DBB6		

<Relocate>.command.in.basic.tool	**NC → PLC**	**UInt**
刀具的 T 号		

<Relocate>.command.in.basic.toolSourceMagazine		**NC → PLC**	**Int**
刀具当前刀库的编号			
SINUMERIK ONE Basic Program	LBP_TMLoadSMag		
SINUMERIK 840D sl	DB71.DBW24		

<Relocate>.command.in.basic.toolSourceLocation		**NC → PLC**	**Int**
刀具当前位置的编号			
SINUMERIK ONE Basic Program	LBP_TMLoadSLoc		
SINUMERIK 840D sl	DB71.DBW26		

<Relocate>.command.in.basic.toolTargetMagazine		**NC → PLC**	**Int**
刀具目标刀库的编号			
SINUMERIK ONE Basic Program	LBP_TMLoadTMag		
SINUMERIK 840D sl	DB71.DBW28		

<Relocate>.command.in.basic.toolTargetLocation		**NC → PLC**	**Int**
刀具目标位置的编号			
SINUMERIK ONE Basic Program	LBP_TMLoadTLoc		
SINUMERIK 840D sl	DB71.DBW30		

<Relocate>.command.in.options.byNcProg		**NC → PLC**	**Bool**
传刀请求由数控程序发出			
SINUMERIK ONE Basic Program	LBP_TMLoadPosbyProgram		
SINUMERIK 840D sl	DB71.DBX4.4		

<Relocate>.command.in.toolProperties.toolLocationType	**NC → PLC**	**UInt**
刀具的位置类型		

<Relocate>.command.in.toolProperties.toolSizeLeft	**NC → PLC**	**UInt**
刀具在左侧占用的半刀位数量		
<Relocate>.command.in.toolProperties.toolSizeRight	**NC → PLC**	**UInt**
刀具在右侧占用的半刀位数量		
<Relocate>.command.in.toolProperties.toolSizeTop	**NC → PLC**	**UInt**
刀具在顶部占用的半刀位数量		
<Relocate>.command.in.toolProperties.toolSizeBottom	**NC → PLC**	**UInt**
刀具在底部占用的半刀位数量		
<Relocate>.command.in.toolStatus.active	**NC → PLC**	**Bool**
刀具生效		
<Relocate>.command.in.toolStatus.enabled	**NC → PLC**	**Bool**
刀具使能		
<Relocate>.command.in.toolStatus.disabled	**NC → PLC**	**Bool**
刀具禁用		
<Relocate>.command.in.toolStatus.measured	**NC → PLC**	**Bool**
刀具已测量		
<Relocate>.command.in.toolStatus.prewarning	**NC → PLC**	**Bool**
已达到刀具预警限值		
<Relocate>.command.in.toolStatus.changeActive	**NC → PLC**	**Bool**
正在换刀		
<Relocate>.command.in.toolStatus.fixedLocation	**NC → PLC**	**Bool**
刀具位置固定		
<Relocate>.command.in.toolStatus.used	**NC → PLC**	**Bool**
刀具已使用		

<Relocate>.command.in.toolStatus.toolInBuffer	**NC → PLC**	**Bool**
刀具位于周转刀库中		

<Relocate>.command.in.toolStatus.disableStateIgnored	**NC → PLC**	**Bool**
忽略刀具的禁用状态		

<Relocate>.command.in.toolStatus.toBeUnloaded	**NC → PLC**	**Bool**
刀具必须卸下		

<Relocate>.command.in.toolStatus.toBeLoaded	**NC → PLC**	**Bool**
刀具必须装入		

<Relocate>.command.in.toolStatus.masterTool	**NC → PLC**	**Bool**
刀具为主刀具		

<Relocate>.command.in.toolStatus.changeOneToOne	**NC → PLC**	**Bool**
1:1 换刀		

<Relocate>.command.in.toolStatus.manualTool	**NC → PLC**	**Bool**
刀具为手动刀具		

<Relocate>.command.in.toolStatus.toolAtDisabledLocation	**NC → PLC**	**Bool**
刀具位于禁用刀位上		

<Relocate>.command.in.multitoolProperties.distanceType	**NC → PLC**	**UInt**
多刀的距离编码方式		
SINUMERIK ONE Basic Program	LBP_TMLoadMTDistanceCoding	
SINUMERIK 840D sl	DB1071.DBW0	

<Relocate>.command.in.multitoolProperties.distance	**NC → PLC**	**Real**
多刀占据的各个位置的距离采用距离编码方式		
SINUMERIK ONE Basic Program	LBP_TMLoadMTDistance	
SINUMERIK 840D sl	DB1071.DBD4	

<Relocate>.command.in.multitoolProperties.numberOfLocations	**NC → PLC**	**UInt**
多刀占据的位置数量		
SINUMERIK ONE Basic Program	LBP_TMLoadMTPlaces	
SINUMERIK 840D sl	DB1071.DBW2	

<Relocate>.ack.in.busy	**NC → PLC**	**Bool**
正在应答		

<Relocate>.ack.in.done	**NC → PLC**	**Bool**
应答成功		

<Relocate>.ack.in.error	**NC → PLC**	**Bool**
应答被拒，查看“状态”		

<Relocate>.ack.in.status	**NC → PLC**	**Word**
应答失败时的状态信息		

<Relocate>.ack.out.execute	**PLC → NC**	**Bool**
开始应答移刀		

<Relocate>.ack.out.totalComplete	**PLC → NC**	**Bool**
发出应答“操作成功”，状态 1、5 或 6 被使用，刀具位置参数和状态参数被忽略		
SINUMERIK ONE Basic Program	LBP_TMLoadQuit	
SINUMERIK 840D sl	DB71.DBX2.0..3.7	

<Relocate>.ack.out.totalFailed	**PLC → NC**	**Bool**
发出应答“操作失败”，状态 3 被使用，刀具位置参数和状态参数被忽略		
SINUMERIK ONE Basic Program	LBP_TMLoadQuit + PerfStatusError	
SINUMERIK 840D sl	DB71.DBX5.0	

<Relocate>.ack.out.toolMagazine	**PLC → NC**	**Int**
刀具当前所在刀库的编号		

<Relocate>.ack.out.toolLocation	**PLC → NC**	**Int**
刀具在刀库中的当前位置编号		

<Relocate>.ack.out.status	**PLC → NC**	**USInt**
传刀状态		

12.2.8 NCK_typeLoadingPoint

<LoadingPoint>.config		
装刀位的配置数据		

<LoadingPoint>.command		
刀具指令信号		

<LoadingPoint>.ack		
对刀具指令的应答，只有当请求生效时才有效		

<LoadingPoint>.command.in		
从刀具管理收到的信号		

<LoadingPoint>.ack.in	**NC → PLC**	
收到的传刀应答信号		

<LoadingPoint>.ack.out	**PLC → NC**	
发送的传刀应答信号		

<LoadingPoint>.config.registered	**NC → PLC**	**Bool**
装刀位已注册		

<LoadingPoint>.config.loadingPoint	**NC → PLC**	**UInt**
装刀位的编号		

<LoadingPoint>.config.toaUnit	**NC → PLC**	**USInt**
对应 TO 范围的编号		
SINUMERIK ONE Basic Program	LBP_TMLoadTMNo	
SINUMERIK 840D sl	DB71.DBB7	

<LoadingPoint>.config.loadingMagazine	**NC → PLC**	**UInt**
装刀位所在的刀库的编号，通常是装载刀库		

<LoadingPoint>.config.loadingLocation	**NC → PLC**	**UInt**
刀库中装刀位的编号		

<LoadingPoint>.command.in.request	**NC → PLC**	**Bool**
NC 请求执行传刀，指令信号有效		
SINUMERIK ONE Basic Program	LBP_TMLoadIFNo	
SINUMERIK 840D sl	DB71.DBX0.0..1.7	

<LoadingPoint>.command.in.multitoolRequest	**NC → PLC**	**Bool**
已请求多刀的装刀/卸刀，多刀信号有效。只有当"<loadingPoint>.command.in.request"置位时，该信号才有效		
SINUMERIK ONE Basic Program	LBP_TMLoadDataInDB1071	
SINUMERIK 840D sl	DB71.DBX5.7	

<LoadingPoint>.command.in.basic	**NC → PLC**
通用指令信号。只有当" <loadingPoint>.command.in.request"置位时，该信号才生效	

<LoadingPoint>.command.in.options	**NC → PLC**
指令的选项。只有当"<loadingPoint>.command.in.request"置位时，它才有效。	

<LoadingPoint>.command.in.toolProperties	**NC → PLC**
刀具属性。只有当"<loadingPoint>.command.in.request"置位时，该信号才有效。	

<LoadingPoint>.command.in.toolStatus	**NC → PLC**
刀具状态。只有当"<loadingPoint>.command.in.request"置位时，它才有效。	

<LoadingPoint>.command.in.userParameters	**NC → PLC**	**DInt**
使用指令($P_VDITCP)传递的用户自定义参数。只有当"<loadingPoint>.command.in.request"置位时，它才有效。		
SINUMERIK ONE Basic Program	LBP_TMLoadreserved1 - 3	
SINUMERIK 840D sl	DB71.DBD8, .DBD12, .DBD16	

<LoadingPoint>.command.in.multitoolProperties	**NC → PLC**
多刀的属性。只有当" <loadingPoint>.command.in.multitoolRequest"置位时，该信号才有效	

<LoadingPoint>.command.in.basic.ncCommand	**NC → PLC**	**USInt**
内部 NC 指令编号		

<LoadingPoint>.command.in.basic.load		**NC → PLC**	**Bool**
装刀已请求			
SINUMERIK ONE Basic Program	LBP_TMLoadLoading		
SINUMERIK 840D sl	DB71.DBX4.0		

<LoadingPoint>.command.in.basic.unload		**NC → PLC**	**Bool**
卸刀已请求			
SINUMERIK ONE Basic Program	LBP_TMLoadUnloading		
SINUMERIK 840D sl	DB71.DBX4.1		

<LoadingPoint>.command.in.basic.position		**NC → PLC**	**Bool**
已请求“刀库定位”到装刀位上			
SINUMERIK ONE Basic Program	LBP_TMLoadPositioning		
SINUMERIK 840D sl	DB71.DBX4.3		

<LoadingPoint>.command.in.basic.chan		**NC → PLC**	**USInt**
对应的通道号			
SINUMERIK ONE Basic Program	LBP_TMLoadChannel		
SINUMERIK 840D sl	DB71.DBB6		

<LoadingPoint>.command.in.basic.tool		**NC → PLC**	**UInt**
刀具的 T 号			

<LoadingPoint>.command.in.basic.toolSourceMagazine		**NC → PLC**	**Int**
刀具当前刀库的编号；或是需要定位到装刀位上的刀库的编号			
SINUMERIK ONE Basic Program	LBP_TMLoadSMag | LoadIdent		
SINUMERIK 840D sl	DB71.DBW24, .DBW54, .DBW84, .DBW114		

<LoadingPoint>.command.in.basic.toolSourceLocation		**NC → PLC**	**Int**
刀具当前位置的编号；或是需要定位到装刀位上的位置的编号			
SINUMERIK ONE Basic Program	LBP_TMLoadSLoc | LoadNo		
SINUMERIK 840D sl	DB71.DBW26, .DBW56, .DBW86, .DBW116		

<LoadingPoint>.command.in.basic.toolTargetMagazine	**NC → PLC**	**Int**
刀具目标刀库的编号；或是目标装刀位的刀库编号		
SINUMERIK ONE Basic Program	LBP_TMLoadTMag | LoadIdent	
SINUMERIK 840D sl	DB71.DBW28, .DBW58, .DBW88, .DBW118	

<LoadingPoint>.command.in.basic.toolTargetLocation	**NC → PLC**	**Int**
刀具目标位置的编号；或是目标装刀位的位置编号		
SINUMERIK ONE Basic Program	LBP_TMLoadTLoc | LoadNo	
SINUMERIK 840D sl	DB71.DBW30, .DBW60, .DBW90, .DBW120	

<LoadingPoint>.command.in.options.byNcProg	**NC → PLC**	**Bool**
传刀请求由数控程序发出		
SINUMERIK ONE Basic Program	LBP_TMLoadPosbyProgram	
SINUMERIK 840D sl	DB71.DBX4.4	

<LoadingPoint>.command.in.toolProperties.toolLocationType	**NC → PLC**	**UInt**
刀具的位置类型		

<LoadingPoint>.command.in.toolProperties.toolSizeLeft	**NC → PLC**	**UInt**
刀具在左侧占用的半刀位数量		

<LoadingPoint>.command.in.toolProperties.toolSizeRight	**NC → PLC**	**UInt**
刀具在右侧占用的半刀位数量		

<LoadingPoint>.command.in.toolProperties.toolSizeTop	**NC → PLC**	**UInt**
刀具在顶部占用的半刀位数量		

<LoadingPoint>.command.in.toolProperties.toolSizeBottom	**NC → PLC**	**UInt**
刀具在底部占用的半刀位数量		

<LoadingPoint>.command.in.toolStatus.active	**NC → PLC**	**Bool**
刀具生效		

<LoadingPoint>.command.in.toolStatus.enabled	**NC → PLC**	**Bool**
刀具使能		

<LoadingPoint>.command.in.toolStatus.disabled	**NC → PLC**	**Bool**
刀具禁用		

<LoadingPoint>.command.in.toolStatus.measured	**NC → PLC**	**Bool**
刀具已测量		

<LoadingPoint>.command.in.toolStatus.prewarning	**NC → PLC**	**Bool**
已达到刀具预警限值		

<LoadingPoint>.command.in.toolStatus.changeActive	**NC → PLC**	**Bool**
正在换刀		

<LoadingPoint>.command.in.toolStatus.fixedLocation	**NC → PLC**	**Bool**
刀具位置固定		

<LoadingPoint>.command.in.toolStatus.used	**NC → PLC**	**Bool**
刀具已使用		

<LoadingPoint>.command.in.toolStatus.toolInBuffer	**NC → PLC**	**Bool**
刀具位于周转刀库中		

<LoadingPoint>.command.in.toolStatus.disableStateIgnored	**NC → PLC**	**Bool**
忽略刀具的禁用状态		

<LoadingPoint>.command.in.toolStatus.toBeUnloaded	**NC → PLC**	**Bool**
刀具必须卸下		

<LoadingPoint>.command.in.toolStatus.toBeLoaded	**NC → PLC**	**Bool**
刀具必须装入		

<LoadingPoint>.command.in.toolStatus.masterTool	**NC → PLC**	**Bool**
刀具为主刀具		

<LoadingPoint>.command.in.toolStatus.changeOneToOne	**NC → PLC**	**Bool**
1:1 换刀		

<LoadingPoint>.command.in.toolStatus.manualTool	**NC → PLC**	**Bool**
刀具为手动刀具		

<LoadingPoint>.command.in.toolStatus.toolAtDisabledLocation	**NC → PLC**	**Bool**
刀具位于禁用刀位上		

<LoadingPoint>.command.in.multitoolProperties.distanceType	**NC → PLC**	**UInt**
多刀的距离编码方式		
SINUMERIK ONE Basic Program	LBP_TMLoadMTDistanceCoding	
SINUMERIK 840D sl	DB1071.DBW0	

<LoadingPoint>.command.in.multitoolProperties.distance	**NC → PLC**	**Real**
多刀占据的各个位置的距离采用距离编码方式		
SINUMERIK ONE Basic Program	LBP_TMLoadMTDistance	
SINUMERIK 840D sl	DB1071.DBD4	

<LoadingPoint>.command.in.multitoolProperties.numberOfLocations	**NC → PLC**	**UInt**
多刀占据的位置数量		
SINUMERIK ONE Basic Program	LBP_TMLoadMTPlaces	
SINUMERIK 840D sl	DB1071.DBW2	

<LoadingPoint>.ack.in.busy	**NC → PLC**	**Bool**
正在应答		

<LoadingPoint>.ack.in.done	**NC → PLC**	**Bool**
应答成功		

<LoadingPoint>.ack.in.error	**NC → PLC**	**Bool**
应答被拒，查看“状态”		

<LoadingPoint>.ack.in.status	**NC → PLC**	**Word**
应答失败时的状态信息		
<LoadingPoint>.ack.out.execute	**PLC → NC**	**Bool**
开始应答移刀		
<LoadingPoint>.ack.out.totalComplete	**PLC → NC**	**Bool**
发出应答“操作成功”，状态 1、5 或 6 被使用，刀具位置参数和状态参数被忽略		
<LoadingPoint>.ack.out.totalFailed	**PLC → NC**	**Bool**
发出应答“操作失败”，状态 3 被使用，刀具位置参数和状态参数被忽略		
<LoadingPoint>.ack.out.toolMagazine	**PLC → NC**	**Int**
刀具当前所在刀库的编号		
<LoadingPoint>.ack.out.toolLocation	**PLC → NC**	**Int**
刀具在刀库中的当前位置编号		
<LoadingPoint>.ack.out.multitoolLocation	**PLC → NC**	**Int**
<LoadingPoint>.ack.out.status	**PLC → NC**	**USInt**
传刀状态		

12.2.9 NCK_typeToaUnit

<ToaUnit>.config.magazines[0].magazineNumber	**NC → PLC**	**UInt**
刀库编号		
<ToaUnit>.config.magazines[0].magazineType	**NC → PLC**	**USInt**
刀库类型		
<ToaUnit>.config.magazines[0].locationCount	**NC → PLC**	**UInt**
刀库中位置的数量		

<ToaUnit>.config.registered	**NC → PLC**	**Bool**
TO 范围已注册		

<ToaUnit>.config.toaUnitNumber	**NC → PLC**	**USInt**
TO 范围编号		

<ToaUnit>.config.associatedChans	**NC → PLC**	**Bool**
指出，该 TO 范围分配给了哪个通道号。		

<ToaUnit>.config.maxToolholder	**NC → PLC**	**UInt**
最大刀柄编号		

<ToaUnit>.config.bufferMagazine	**NC → PLC**	**UInt**
周转刀库编号		

<ToaUnit>.config.loadingMagazine	**NC → PLC**	**UInt**
装载刀库编号		

<ToaUnit>.config.magazineCount	**NC → PLC**	**USInt**
刀库列表中条目的数量		

<ToaUnit>.config.magazines	**NC → PLC**
刀库列表	

<ToaUnit>.config	**NC → PLC**
TO 范围的配置数据	

12.2.10 NCK_typeHmi

<Hmi>.config
HMI 配置数据

<Hmi>.mappingTable
HMI 映射表

<Hmi>.basic		
HMI 的通用信号		

<Hmi>.progSelection		
HMI 上的程序选择信号		

<Hmi>.plcFunc		
HMI 上的 PLC 功能信号		

<Hmi>.userKeys		
HMI 用户键信号		

<Hmi>.basic.in	**OP → PLC**	
从 HMI 收到的通用信号		

<Hmi>.basic.out	**PLC → OP**	
发送给 HMI 的通用信号		

<Hmi>.basic.inOut	**OP ↔ PLC**	
输入和输出的通用 HMI 信号		

<Hmi>.config.registered	**OP → PLC**	**Bool**
HMI 已注册		

<Hmi>.basic.in.screenDark	**OP → PLC**	**Bool**
屏幕变为黑屏。按键会响应按压操作，但可通过"basic.out.disableKeys"禁止该响应。		
SINUMERIK ONE Basic Program	LBP_HMI.E_SCDark	
SINUMERIK 840D sl	DB19.DBX20.1	

<Hmi>.basic.in.cancelKeyPressed	**OP → PLC**	**Bool**
"Alarm Cancel"键已按下		
SINUMERIK ONE Basic Program	LBP_HMI.E_Cancel	
SINUMERIK 840D sl	DB19.DBX20.2	

<Hmi>.basic.in.simActive		**OP → PLC**	**Bool**
仿真窗口已激活			
SINUMERIK ONE Basic Program	LBP_HMI.E_SimActiv		
SINUMERIK 840D sl	DB19.DBX20.6		

<Hmi>.basic.in.mcsWcsSwitched		**OP → PLC**	**Bool**
出现 0 → 1 上升沿：切换机床坐标系/工件坐标系			
SINUMERIK ONE Basic Program	LBP_HMI.E_ActWCS		
SINUMERIK 840D sl	DB19.DBX20.7		

<Hmi>.basic.in.activeOperatingAreaNumber		**OP → PLC**	**USInt**
当前生效的操作区域编号			
SINUMERIK ONE Basic Program	LBP_HMI.E_ActivWA		
SINUMERIK 840D sl	DB19.DBX21.0		

<Hmi>.basic.in.currentChanNumber		**OP → PLC**	**USInt**
HMI 上显示的当前通道号			
SINUMERIK ONE Basic Program	LBP_HMI.E_Chan		
SINUMERIK 840D sl	DB19.DBX22.0		

<Hmi>.basic.in.inputDeviceActive		**OP → PLC**	**Bool**
输入设备已生效，比如：使用键盘			
SINUMERIK ONE Basic Program	LBP_HMI.E_KeybHandl		
SINUMERIK 840D sl	DB19.DBX23.0		

<Hmi>.basic.in.dataTransferActive		**OP → PLC**	**Bool**
数据传送已生效			
SINUMERIK ONE Basic Program	LBP_HMI.E_DataTransf		
SINUMERIK 840D sl	DB19.DBX23.1		

<Hmi>.basic.in.screenChangeActive		**OP → PLC**	**Bool**
屏幕切换已生效			
SINUMERIK ONE Basic Program	LBP_HMI.E_DisplChange		
SINUMERIK 840D sl	DB19.DBX23.2		

<Hmi>.basic.in.currentScreenNumber		**OP → PLC**	**UInt**
当前选中图片的编号			
SINUMERIK ONE Basic Program	LBP_HMI.E_V24Err		
SINUMERIK 840D sl	DB19.DBW24		

<Hmi>.basic.in.mcpAddress		**OP → PLC**	**USInt**
MCP 地址			
SINUMERIK ONE Basic Program	LBP_HMI.MMC1_MSTT_ADR		
SINUMERIK 840D sl	DB19.DBB123		

<Hmi>.basic.in.directKeyAddress		**OP → PLC**	**USInt**
直接按键的地址			
SINUMERIK ONE Basic Program	LBP_HMI.Tcu1KeyAdr		
SINUMERIK 840D sl	DB19.DBB116		

<Hmi>.basic.in.tcuIndex		**OP → PLC**	**USInt**
TCU 序号			
SINUMERIK ONE Basic Program	LBP_HMI.Tcu1Index		
SINUMERIK 840D sl	DB19.DBB118		

<Hmi>.basic.in.switchoverTcuDisabled		**OP → PLC**	**Bool**
TCU 切换已禁止			
SINUMERIK ONE Basic Program	LBP_HMI.MMC1_TCU_SHIFT_LOCK		
SINUMERIK 840D sl	DB19.DBX126.6		

<Hmi>.basic.in.languageId		**OP → PLC**	**USInt**
操作软件中设置的语言 ID			
SINUMERIK ONE Basic Program	LBP_HMI.E_LanguageID		
SINUMERIK 840D sl	DB10.DBB96		

<Hmi>.basic.out.brightenScreen		**PLC → OP**	**Bool**
忽略“basic.out.darkenScreen”，时间编码器控制仍工作(MD9006 $MM_DISPLAY_SWITCH_OFF_INTERVAL)			
SINUMERIK ONE Basic Program	LBP_HMI.A_SCLight		
SINUMERIK 840D sl	DB19.DBX0.0		

<Hmi>.basic.out.darkenScreen	**PLC → OP**	**Bool**
黑屏控制。屏幕内容不可见，屏幕通过按键操作再次激活被禁止。		
SINUMERIK ONE Basic Program	LBP_HMI.A_SCDark	
SINUMERIK 840D sl	DB19.DBX0.1	

<Hmi>.basic.out.disableKeys	**PLC → OP**	**Bool**
禁止用户的键盘输入		
SINUMERIK ONE Basic Program	LBP_HMI.A_KeyDisable	
SINUMERIK 840D sl	DB19.DBX0.2	

<Hmi>.basic.out.clearCancelAlarms	**PLC → OP**	**Bool**
请求应答待处理的 Cancel 类报警		
SINUMERIK ONE Basic Program	LBP_HMI.A_ClearCanAl	
SINUMERIK 840D sl	DB19.DBX0.3	

<Hmi>.basic.out.clearRecallAlarms	**PLC → OP**	**Bool**
请求应答待处理的 Recall 类报警		
SINUMERIK ONE Basic Program	LBP_HMI.A_ClearRecAl	
SINUMERIK 840D sl	DB19.DBX0.4	

<Hmi>.basic.out.saveActionLog	**PLC → OP**	**Bool**
出现 0 → 1 上升沿：ACTION.COM 保存到 CRASH.COM		
SINUMERIK ONE Basic Program	LBP_HMI.A_SafeActionLog	
SINUMERIK 840D sl	DB19.DBX0.6	

<Hmi>.basic.out.displayWcsValues	**PLC → OP**	**Bool**
以工件坐标系显示实际值(0=机床坐标系, 1=工件坐标系)		
SINUMERIK ONE Basic Program	LBP_HMI.A_ActWCS	
SINUMERIK 840D sl	DB19.DBX0.7	

<Hmi>.basic.out.rejectRemoteAccess	**PLC → OP**	**Bool**
外部 Viewer：没有远程访问		
SINUMERIK ONE Basic Program	LBP_HMI.A_PLCExtViewerReject	
SINUMERIK 840D sl	DB19.DBX1.0	

<Hmi>.basic.out.permitRemoteMonitoringOnly	**PLC → OP**	**Bool**
外部 Viewer：远程访问时只读		
SINUMERIK ONE Basic Program	LBP_HMI.A_PLCExtViewerMode	
SINUMERIK 840D sl	DB19.DBX1.1	

<Hmi>.basic.out.analogSpindle1Percentage	**PLC → OP**	**USInt**
模拟量主轴 1：负载率，单位 %		
SINUMERIK ONE Basic Program	LBP_HMI.A_ASpi1percent	
SINUMERIK 840D sl	DB19.DBB6	

<Hmi>.basic.out.analogSpindle2Percentage	**PLC → OP**	**USInt**
模拟量主轴 2：负载率，单位 %		
SINUMERIK ONE Basic Program	LBP_HMI.A_ASpi2percent	
SINUMERIK 840D sl	DB19.DBB7	

<Hmi>.basic.out.disableTeachInTransfer	**PLC → OP**	**Bool**
禁止 TEACH IN 传送		
SINUMERIK ONE Basic Program	LBP_HMI.A_TeachDisable	
SINUMERIK 840D sl	DB19.DBX13.0	

<Hmi>.basic.inOut.cancelAlarmsCleared	**OP → PLC**	**Bool**
Cancel 类报警已删除（必须通过用户程序复位）		
SINUMERIK ONE Basic Program	LBP_HMI.E_CanCleared	
SINUMERIK 840D sl	DB19.DBX20.3	

<Hmi>.basic.inOut.recallAlarmsCleared	**OP → PLC**	**Bool**
Recall 类报警已删除（必须通过用户程序复位）		
SINUMERIK ONE Basic Program	LBP_HMI.E_RecCleared	
SINUMERIK 840D sl	DB19.DBX20.4	

<Hmi>.basic.inOut.plcHardkeySelection	**PLC → OP**	**USInt**
为 PLC 硬键选择按键值 1..255		
SINUMERIK ONE Basic Program	LBP_HMI.A_Hardkey	
SINUMERIK 840D sl	DB19.DBB10	

<Hmi>.progSelection.in	**OP → PLC**
从 HMI 发出的程序选择信号	

<Hmi>.progSelection.out	**PLC → OP**
发送给 HMI 的程序选择信号	

<Hmi>.progSelection.in.identified	**OP → PLC**	**Bool**
检测出通道选择		
SINUMERIK ONE Basic Program	LBP_HMI.E_PPsel	
SINUMERIK 840D sl	DB19.DBX26.7	

<Hmi>.progSelection.in.selected	**OP → PLC**	**Bool**
通道已选择（任务还没有结束）		
SINUMERIK ONE Basic Program	LBP_HMI.E_PPAct	
SINUMERIK 840D sl	DB19.DBX26.3	

<Hmi>.progSelection.in.completed	**OP → PLC**	**Bool**
任务结束		
SINUMERIK ONE Basic Program	LBP_HMI.E_PPOK	
SINUMERIK 840D sl	DB19.DBX26.1	

<Hmi>.progSelection.in.error	**OP → PLC**	**Bool**
程序选择出错，见"errorCode"中的故障代码		
SINUMERIK ONE Basic Program	LBP_HMI.E_PPErr	
SINUMERIK 840D sl	DB19.DBX26.2	

<Hmi>.progSelection.in.errorCode	**OP → PLC**	**USInt**
故障代码		
SINUMERIK ONE Basic Program	LBP_HMI.E_PPError	
SINUMERIK 840D sl	DB19.DBB27	

<Hmi>.progSelection.out.request	**PLC → OP**	**Bool**
选择指定 NC 程序（开始任务）		
SINUMERIK ONE Basic Program	LBP_HMI.A_PPsel	
SINUMERIK 840D sl	DB19.DBX13.7	

<Hmi>.progSelection.out.progList		**PLC → OP**	**USInt**
程序列表名称			
SINUMERIK ONE Basic Program	LBP_HMI.A_V24NoFile1		
SINUMERIK 840D sl	DB19.DBB16		

<Hmi>.progSelection.out.progNumber		**PLC → OP**	**USInt**
程序号选择针对的是程序列表中包含的程序。			
SINUMERIK ONE Basic Program	LBP_HMI.A_V24NoFile2		
SINUMERIK 840D sl	DB19.DBB17		

<Hmi>.plcFunc.in		**OP → PLC**	
从 HMI 发出的 PLC 功能信号			

<Hmi>.plcFunc.out		**PLC → OP**	
发送给 HMI 的 PLC 功能信号			

<Hmi>.plcFunc.in.busy		**OP → PLC**	**Bool**
在该功能执行期间生效			
SINUMERIK 840D sl	DB19.DBX32.7		

<Hmi>.plcFunc.in.errorCode		**OP → PLC**	**USInt**
表示功能选择出错的故障代码			
SINUMERIK ONE Basic Program	LBP_HMI.E_FuncErr		
SINUMERIK 840D sl	DB19.DBB36		

<Hmi>.plcFunc.out.requestStrobe		**PLC → OP**	**Bool**
功能执行请求 PLC → HMI: 0 → 1; 功能执行反馈 HMI → PLC: 1 → 0			
SINUMERIK 840D sl	DB19.DBX32.6		

<Hmi>.plcFunc.out.parameter1		**PLC → OP**	**USInt**
参数 1：通道编号			
SINUMERIK ONE Basic Program	LBP_HMI.A_Par1		
SINUMERIK 840D sl	DB19.DBB33		

<Hmi>.userKeys.in		**OP → PLC**	
从 HMI 发出的用户按键信号			

<Hmi>.userKeys.out	**PLC → OP**	
发送给 HMI 的用户按键信号		

<Hmi>.userKeys[1].in.pressed	**OP → PLC**	**Bool**
HMI Sidescreen 用户键 已按下		
SINUMERIK ONE Basic Program	LBP_HMI.E_HmiUserKey[1]	
SINUMERIK 840D sl	DB19.DBX404.0	

<Hmi>.userKeys[1].out.activateLed	**PLC → OP**	**Bool**
激活 HMI Sidescreen 用户键 的 LED		
SINUMERIK ONE Basic Program	LBP_HMI.A_HmiUserLed[1]	
SINUMERIK 840D sl	DB19.DBX412.0	

<Hmi>.userKeys[1].out.disable	**PLC → OP**	**Bool**
禁用 HMI Sidescreen 用户键		
SINUMERIK ONE Basic Program	LBP_HMI.A_DisableHmiUserKey[1]	
SINUMERIK 840D sl	DB19.DBX420.0	

<Hmi>.mappingTable.in	**OP → PLC**
收到的信号映射表	

<Hmi>.mappingTable.out
收到的数据映射表

<Hmi>.mappingTable.in.ready	**OP → PLC**	**Bool**
HMI 通讯已就绪		

<Hmi>.mappingTable.in.error	**OP → PLC**	**Bool**
和 HMI 的通讯出错		

<Hmi>.mappingTable.in.status	**OP → PLC**	**Word**
HMI 通讯的状态		

<Hmi>.mappingTable.out.tableLines	**PLC → OP**	**WString[16382]**
JSON 字符串		

<Hmi>.mappingTable.out.version	**PLC → OP**	**USInt**
映射表版本		

<Hmi>.mappingTable.out.numberOfTableLines	**PLC → OP**	**USInt**
使用的行数		

<Hmi>.mappingTable.out.status	**PLC → OP**	**Word**
映射表状态		

12.2.11 NCK_typeHmiAxis

<HmiAxis>.config
“HMI-轴”实例的配置数据

<HmiAxis>.basic
从 HMI 收到的轴的通用信号

<HmiAxis>.basic.in
从 HMI 收到的轴的通用信号

<HmiAxis>.basic.in.progTestSuppressionRequested		**OP → PLC**	**Bool**
HMI: 程序测试已禁止			
SINUMERIK ONE Basic Program	LBP_Axis*.E_MMC_ProgtestSuppress		
SINUMERIK 840D sl	DB31, ...DBX128.0		

<HmiAxis>.basic.in.progTestRequested		**OP → PLC**	**Bool**
HMI: 程序测试已请求			
SINUMERIK ONE Basic Program	LBP_Axis*.E_MMC_ProgtestActivate		
SINUMERIK 840D sl	DB31, ...DBX128.1		

<HmiAxis>.basic.in.handwheelSelected	**OP → PLC**	**USInt**	**起始版本 V11**
HMI：已选择手轮 1..6（0 表示无选择）			

12.2.12 NCK_typeHmiChan

<HmiChan>.config
“HMI-通道”实例的配置数据

<HmiChan>.basic
从 HMI 收到的通道的通用信号

<HmiChan>.axisSelection
通过 HMI 选择工作轴的信号

<HmiChan>.geoAxis1	**起始版本 V11**
HMI：几何轴 1 的信号	

<HmiChan>.geoAxis2	**起始版本 V11**
HMI：几何轴 2 的信号	

<HmiChan>.geoAxis3	**起始版本 V11**
HMI：几何轴 3 的信号	

<HmiChan>.orientationAxis1	**起始版本 V11**
HMI：摆动轴 1 的信号	

<HmiChan>.orientationAxis2	**起始版本 V11**
HMI：摆动轴 2 的信号	

<HmiChan>.orientationAxis3	**起始版本 V11**
HMI：摆动轴 3 的信号	

<HmiChan>.basic.in
从 HMI 收到的通道的通用信号

<HmiChan>.axisSelection.in
通过 HMI 选择工作轴的输入信号

<HmiChan>.axisSelection.out
通过 HMI 选择工作轴的输出信号

<HmiChan>.geoAxis.in			**起始版本 V11**
HMI：几何轴的输入信号			

<HmiChan>.orientationAxis.in			**起始版本 V11**
HMI：摆动轴的输入信号			

<HmiChan>.basic.in.configuredStopFuncSelected	**OP → PLC**	**Bool**
HMI：通过操作软件激活自定义暂停 (CST)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_CfgStopFunc	
SINUMERIK 840D sl	DB21, ...DBX24.1	

<HmiChan>.basic.in.atStopSetOvrZeroSelected	**OP → PLC**	**Bool**
HMI：停止时将倍率设为 0%		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_AtStopSetOvrZero	
SINUMERIK 840D sl	DB21, ...DBX24.2	

<HmiChan>.basic.in.handwheelOffsetDrfSelected	**OP → PLC**	**Bool**
HMI: 手轮偏移(DRF)已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_DRF	
SINUMERIK 840D sl	DB21, ...DBX24.3	

<HmiChan>.basic.in.contourHandwheelSelected	**OP → PLC**	**USInt**	**起始版本 V11**
HMI：手轮 1..6 已选择作为轮廓手轮（0 表示无选择）			

<HmiChan>.basic.in.m01AssociatedAuxFuncSelected	**OP → PLC**	**Bool**
HMI: 程序控制 M01“对应的辅助功能”已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_M01AssocNC	
SINUMERIK 840D sl	DB21, ...DBX24.4	

<HmiChan>.basic.in.m01Selected	**OP → PLC**	**Bool**
HMI: 程序控制 M01“有条件停止”已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_M01	
SINUMERIK 840D sl	DB21, ...DBX24.5	

<HmiChan>.basic.in.dryRunSelected	**OP → PLC**	**Bool**
HMI: 试运行进给速度(DRY)已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_DRY	
SINUMERIK 840D sl	DB21, ...DBX24.6	

<HmiChan>.basic.in.rapidTraversOvrSelected	**OP → PLC**	**Bool**
HMI: 为快速移动选中了进给倍率		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_FD_OR4RT_OR	
SINUMERIK 840D sl	DB21, ...DBX25.3	

<HmiChan>.basic.in.reposModeSelectedNumber	**OP → PLC**	**USInt**
HMI: 选中的 REPOS 重新定位： 0 - 无 REPOS 重新定位生效; 1 - 移动到程序段起点(RMBBL); 2 - 移动到程序断点(RMIBL); 3 - 移动到程序段结尾(RMEBL); 4 - 移动到下一个轨迹点 (RMNBL)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_REPOSPM_0	
SINUMERIK 840D sl	DB21, ...DBX25.0..2	

<HmiChan>.basic.in.reposModeChanged	**OP → PLC**	**Bool**
HMI: REPOS 运行方式切换		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_REPOSMode	
SINUMERIK 840D sl	DB21, ...DBX25.4	

<HmiChan>.basic.in.progTestSelected	**OP → PLC**	**Bool**
HMI: 程序测试(PRT)已请求		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_ProgTest	
SINUMERIK 840D sl	DB21, ...DBX25.7	

<HmiChan>.basic.in.skipBlockLevel0	**OP → PLC**	**Bool**
选择了跳过跳转级为 0 的零件程序段 (Skip, SKP)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_SKP0	
SINUMERIK 840D sl	DB21, ...DBX26.0	

<HmiChan>.basic.in.skipBlockLevel1	**OP → PLC**	**Bool**
选择了跳过跳转级为 1 的零件程序段 (Skip, SKP)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_SKP1	
SINUMERIK 840D sl	DB21, ...DBX26.1	

<HmiChan>.basic.in.skipBlockLevel2	**OP → PLC**	**Bool**
选择了跳过跳转级为 2 的零件程序段 (Skip, SKP)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_SKP2	
SINUMERIK 840D sl	DB21, ...DBX26.2	

<HmiChan>.basic.in.skipBlockLevel3	**OP → PLC**	**Bool**
选择了跳过跳转级为 3 的零件程序段 (Skip, SKP)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_SKP3	
SINUMERIK 840D sl	DB21, ...DBX26.3	

<HmiChan>.basic.in.skipBlockLevel4	**OP → PLC**	**Bool**
选择了跳过跳转级为 4 的零件程序段 (Skip, SKP)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_SKP4	
SINUMERIK 840D sl	DB21, ...DBX26.4	

<HmiChan>.basic.in.skipBlockLevel5	**OP → PLC**	**Bool**
选择了跳过跳转级为 5 的零件程序段 (Skip, SKP)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_SKP5	
SINUMERIK 840D sl	DB21, ...DBX26.5	

<HmiChan>.basic.in.skipBlockLevel6	**OP → PLC**	**Bool**
选择了跳过跳转级为 6 的零件程序段 (Skip, SKP)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_SKP6	
SINUMERIK 840D sl	DB21, ...DBX26.6	

<HmiChan>.basic.in.skipBlockLevel7	**OP → PLC**	**Bool**
选择了跳过跳转级为 7 的零件程序段 (Skip, SKP)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_SKP7	
SINUMERIK 840D sl	DB21, ...DBX26.7	

<HmiChan>.basic.in.skipBlockLevel8	**OP → PLC**	**Bool**
选择了跳过跳转级为 8 的零件程序段 (Skip, SKP)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_SKP8	
SINUMERIK 840D sl	DB21, ...DBX27.0	

<HmiChan>.basic.in.skipBlockLevel9	**OP → PLC**	**Bool**
选择了跳过跳转级为 9 的零件程序段 (Skip, SKP)		
SINUMERIK ONE Basic Program	LBP_Chan*.E_MMC_SKP9	
SINUMERIK 840D sl	DB21, ...DBX27.1	

<HmiChan>.axisSelection.in.selectedType	**OP → PLC**	**USInt**
轴类型选择：0：机床轴，1：几何轴，2：方向轴		
SINUMERIK ONE Basic Program	LBP_Chan*.E_AxisSelType	
SINUMERIK 840D sl	DB21, ...DBB408	

<HmiChan>.axisSelection.in.selectedNumber	**OP → PLC**	**USInt**
轴编号：机床轴/几何轴/方向轴		
SINUMERIK ONE Basic Program	LBP_Chan*.E_AxisSelAxis	
SINUMERIK 840D sl	DB21, ...DBB409	

<HmiChan>.axisSelection.in.selectedKeyActuated	**OP → PLC**	**Bool**
出现 0 → 1 上升沿：“按键”轴选择（触控）		
SINUMERIK ONE Basic Program	LBP_Chan*.E_AxisSelRequest	
SINUMERIK 840D sl	DB21, ...DBX410.0	

<HmiChan>.axisSelection.out.activeType	**PLC → OP**	**USInt**
生效的轴选择（反馈给 HMI）：0：机床轴，1：几何轴，2：方向轴		
SINUMERIK ONE Basic Program	LBP_Chan*.A_AxisSelTypeActive	
SINUMERIK 840D sl	DB21, ...DBB412	

<HmiChan>.axisSelection.out.activeNumber	**PLC → OP**	**USInt**
生效的轴编号（反馈给 HMI）：机床轴/几何轴/方向轴		
SINUMERIK ONE Basic Program	LBP_Chan*.A_AxisSelAxisActive	
SINUMERIK 840D sl	DB21, ...DBB413	

<HmiChan>.geoAxis.in.handwheelSelected	**OP → PLC**	**USInt**	**起始版本 V11**
HMI：手轮 1..6 已选择作为几何轴（0 表示无选择）			

<HmiChan>.geoAxis1.in.handwheelSelected	**OP → PLC**	**USInt**	**起始版本 V11**
HMI：手轮 1..6 已选择作为几何轴（0 表示无选择）			

<HmiChan>.geoAxis2.in.handwheelSelected	**OP → PLC**	**USInt**	**起始版本 V11**
HMI：手轮 1..6 已选择作为几何轴（0 表示无选择）			

<HmiChan>.geoAxis3.in.handwheelSelected	**OP → PLC**	**USInt**	**起始版本 V11**
HMI：手轮 1..6 已选择作为几何轴（0 表示无选择）			

<HmiChan>.orientationAxis.in.handwheelSelected	**OP → PLC**	**USInt**	**起始版本 V11**
HMI：手轮 1..6 已选择作为摆动轴（0 表示无选择）			

<HmiChan>.orientationAxis1.in.handwheelSelected	**OP → PLC**	**USInt**	**起始版本 V11**
HMI：手轮 1..6 已选择作为摆动轴（0 表示无选择）			

<HmiChan>.orientationAxis2.in.handwheelSelected	**OP → PLC**	**USInt**	**起始版本 V11**
HMI：手轮 1..6 已选择作为摆动轴（0 表示无选择）			

<HmiChan>.orientationAxis3.in.handwheelSelected	**OP → PLC**	**USInt**	**起始版本 V11**
HMI：手轮 1..6 已选择作为摆动轴（0 表示无选择）			

12.2.13 NCK_typeHmiModeGroup

<HmiModeGroup>.config
“HMI-运行方式组”实例的配置数据

<HmiModeGroup>.basic
从 HMI 收到的运行方式组的通用信号

<HmiModeGroup>.basic.in
从 HMI 收到的运行方式组的通用信号

<HmiModeGroup>.basic.in.autoModeRequested	**OP → PLC**	**Bool**
HMI: 已请求切换到自动方式		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_MMC_AUTO	
SINUMERIK 840D sl	DB11.DBX4.0	

<HmiModeGroup>.basic.in.mdaModeRequested	**OP → PLC**	**Bool**
HMI: 已请求切换到 MDI 方式		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_MMC_MDA	
SINUMERIK 840D sl	DB11.DBX4.1	

<HmiModeGroup>.basic.in.jogModeRequested	**OP → PLC**	**Bool**
HMI: 已请求切换到 JOG 方式		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_MMC_JOG	
SINUMERIK 840D sl	DB11.DBX4.2	

<HmiModeGroup>.basic.in.teachInRequested	**OP → PLC**	**Bool**
HMI: 已请求执行机床功能 TEACH IN		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_MMC_TEACHIN	
SINUMERIK 840D sl	DB11.DBX5.0	

<HmiModeGroup>.basic.in.reposRequested	**OP → PLC**	**Bool**
HMI: 已请求执行机床功能 REPOS		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_MMC_REPOS	
SINUMERIK 840D sl	DB11.DBX5.1	

<HmiModeGroup>.basic.in.refRequested	**OP → PLC**	**Bool**
HMI: 已请求执行机床功能 REF		
SINUMERIK ONE Basic Program	LBP_ModeGroup.E_MMC_REF	
SINUMERIK 840D sl	DB11.DBX5.2	

<HmiModeGroup>.basic.in.inc1Requested	**OP → PLC**	**Bool**
HMI: 已请求移动 1 个增量		
SINUMERIK 840D sl	DB11.DBX10.0	

<HmiModeGroup>.basic.in.inc10Requested	**OP → PLC**	**Bool**
HMI: 已请求移动 10 个增量		
SINUMERIK 840D sl	DB11.DBX10.1	

<HmiModeGroup>.basic.in.inc100Requested	**OP → PLC**	**Bool**
HMI: 已请求移动 100 个增量		
SINUMERIK 840D sl	DB11.DBX10.2	

<HmiModeGroup>.basic.in.inc1000Requested	**OP → PLC**	**Bool**
HMI: 已请求移动 1 000 个增量		
SINUMERIK 840D sl	DB11.DBX10.3	

<HmiModeGroup>.basic.in.inc10000Requested	**OP → PLC**	**Bool**
HMI: 已请求移动 10 000 个增量		
SINUMERIK 840D sl	DB11.DBX10.4	

<HmiModeGroup>.basic.in.incVarRequested	**OP → PLC**	**Bool**
HMI: 已请求移动用户自定义数量的增量		
SINUMERIK 840D sl	DB11.DBX10.5	

<HmiModeGroup>.basic.in.contTraversRequested	**OP → PLC**	**Bool**
HMI: 连续移动已请求		
SINUMERIK 840D sl	DB11.DBX10.6	

12.2.14 NCK_typeHmiNc

<HmiNc>.config
" HMI-NC"实例的配置数据

<HmiNc>.collision
HMI 上的碰撞信号，发给 NC

<HmiNc>.config.registered	**OP → PLC**	**Bool**
NC 已注册用于 HMI		

<HmiNc>.collision.in
从 HMI 收到的 NC 碰撞信号

<HmiNc>.collision.in.machineAreasAutoDeactivated	**OP → PLC**	**Bool**
HMI: 已请求在选中的自动方式中关闭“机床”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.E_CA_DisableAutoMach	
SINUMERIK 840D sl	DB10.DBX93.0	

<HmiNc>.collision.in.toolAreasAutoDeactivated	**OP → PLC**	**Bool**
HMI: 已请求在选中的自动方式中关闭“刀具”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.E_CA_DisableAutoTools	
SINUMERIK 840D sl	DB10.DBX93.1	

<HmiNc>.collision.in.fixtureAreasAutoDeactivated	**OP → PLC**	**Bool**
HMI: 已请求在选中的自动方式中关闭“夹具”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.E_CA_DisableAutoWorkh	
SINUMERIK 840D sl	DB10.DBX93.2	

<HmiNc>.collision.in.workpieceAreasAutoDeactivated	**OP → PLC**	**Bool**
HMI: 已请求在选中的自动方式中关闭“工件”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.E_CA_DisableAutoWorkp	
SINUMERIK 840D sl	DB10.DBX93.3	

<HmiNc>.collision.in.machineAreasJogDeactivated	**OP → PLC**	**Bool**
HMI: 已请求在选中的 JOG 方式中关闭“机床”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.E_CA_DisableJogMach	
SINUMERIK 840D sl	DB10.DBX93.4	

<HmiNc>.collision.in.toolAreasJogDeactivated	**OP → PLC**	**Bool**
HMI: 已请求在选中的 JOG 方式中关闭“刀具”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.E_CA_DisableJogTools	
SINUMERIK 840D sl	DB10.DBX93.5	

<HmiNc>.collision.in.fixtureAreasJogDeactivated	**OP → PLC**	**Bool**
HMI: 已请求在选中的 JOG 方式中关闭“夹具”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.E_CA_DisableJogWorkh	
SINUMERIK 840D sl	DB10.DBX93.6	

<HmiNc>.collision.in.workpieceAreasJogDeactivated	**OP → PLC**	**Bool**
HMI: 已请求在选中的 JOG 方式中关闭“工件”类型的保护区		
SINUMERIK ONE Basic Program	LBP_NC.E_CA_DisableJogWorkp	
SINUMERIK 840D sl	DB10.DBX93.7	

12.3 NCK 相关的数据类型（详细说明）

12.3.1 概览

下表主要介绍基本数据类型。

数据类型	说明
NCK_typeAxis (页 440)	用户程序和轴/主轴之间的接口 包含来自/发送至轴 x/主轴 x 的信号
NCK_typeChan (页 673)	用户程序和通道之间的接口 包含来自/发送至通道 x 的信号
NCK_typeModeGroup (页 874)	用户程序和运行方式组之间的接口 包含来自/发送至 NC 和操作软件的信号
NCK_typeNc (页 912)	用户程序和 NC 之间的接口 包含来自/发送至 NC、PLC 和操作软件的信号
NCK_typeToolholder (页 1055)	用户程序和刀具管理之间的接口 包含刀夹信号
NCK_typeRelocate (页 1040)	用户程序和刀具管理之间的接口 包含移刀信号
NCK_typeLoadingPoint (页 1023)	用户程序和刀具管理之间的接口 包含装载位信号
NCK_typeToaUnit (页 1022)	用户程序和刀具管理之间的接口 包含 TO 区域信号
NCK_typeHmi (页 958)	用户程序和 HMI 之间的接口 包含来自/发送至 HMI 的一般信号
NCK_typeHmiAxis (页 988)	用户程序和 HMI 之间的接口 包含来自 HMI，用于轴的信号
NCK_typeHmiChan (页 991)	用户程序和 HMI 之间的接口 包含来自 HMI，用于通道的信号
NCK_typeHmiModeGroup (页 1008)	用户程序和 HMI 之间的接口 包含来自 HMI，用于运行方式组的信号
NCK_typeHmiNc (页 1019)	用户程序和 HMI 之间的接口 包含来自 HMI，用于 NC 的信号

与老款 NCK 接口数据块相比，当前 NCK 相关数据类型的结构有所不同 -从功能性和信号方向两方面着眼进行结构化。

上述每种数据类型都具有一个“config”结构，该结构包含与数据类型有关的组态数据，具体含义见下文。

只要合理，所有其他结构都包含两个子结构“in”和“out”。这两个子结构包含各自的专用信号。通过这种方式，立刻就能识别出从 PLC 角度来看，信号是输入信号 (in) 还是输出信号 (out)。

功能性结构将用途一致的信号结合在一起。

说明

创建“NCK_typeHmi*”数据类型的实例时，必须设置“可从 HMI/OPC UA/Web API 到达”和“可从 HMI/OPC UA/Web API 写入”两种属性。如果不设置，HMI Operate 无法写入信号。

12.3.2 NCK_typeAxis

下表说明了数据类型 NCK_typeAxis 的含义：

结构	说明
.config	轴的组态数据
.registered	轴已注册
.axisActive	轴已生效
.axisNumber	轴编号
.basic.in (页 441)	来自轴的通用信号
.basic.out (页 491)	发送至轴的通用信号
.drive.in (页 540)	来自驱动系统的信号
.drive.out (页 561)	发送至驱动系统的信号
interchange.in (页 568)	针对轴交换的输入信号
interchange.out (页 572)	针对轴交换的输出信号
.spindle.in (页 577)	来自主轴的信号
.spindle.out (页 599)	发送至主轴的信号
.coupling.in (页 641)	来自轴的特定连接信号
.coupling.out (页 658)	发送至轴的特定连接信号

结构	说明
.plcCtrl.in (页 665)	PLC 控制的轴的输入状态信号
.plcCtrl.out (页 668)	发送至 PLC 控制的轴的控制信号
.oscillation.in (页 616)	来自轴的特定摆动信号
.oscillation.out (页 620)	发送至轴的特定摆动信号
.safety.in (页 623)	来自轴的安全专用输入信号
.safety.out (页 623)	发送至轴的安全专用输出信号
.safetyInfoChan.in (页 624)	从轴发出的 SIC 信号
.safetyCtrlChan.out (页 637)	发送至轴的 SCC 信号

12.3.3 <Axis>.basic.in

12.3.3.1 actToolBasePos

NC → PLC	<Axis>.basic.in.actToolBasePos
轴的当前 MCS 设定值	

描述

确定相关轴在机床坐标系 (MCS) 中的当前设定值（对应于 $AA_IM 的内容）。

MCS 值包含所有与轴相关的叠加部分（DRF、$AA_OFF、外部零点偏移等）。

说明

直到使用 <Nc>.basic.out.transferActToolBasePos 激活后才会传输这些值。

数据类型和更新

LReal，周期性

值

轴的当前 MCS 设定值。对于线性轴，值始终以测量单位传输；对于旋转轴，值始终以度数传输。

对应于

- <Nc>.basic.out.transferActToolBasePos
  传输轴的当前 MCS 设定值
- 系统变量：$AA_IM
  轴的当前 MCS 设定值
- OPI 变量：actToolBasePos
  MCS 中的轴位置的当前设定值

更多信息

功能手册之基本功能，部分“坐标系中的位置”

12.3.3.2 alarmActive

NC → PLC	<Axis>.basic.in.alarmActive
轴上没有未经确认的报警	

说明

响应：

- 轴沿其加速度特性曲线制动至静止状态。
- 轴的状态转换至“单轴存在报警”：$AA_SNGLAX_STAT == 5

数据类型和更新

BOOL，循环

值 TRUE

发出轴专用报警。

值 FALSE

无轴专用报警

关联

- <Axis>.plcCtrl.out.requestPlcCtrl
  请求通过适用于自主单轴操作的 PLC 进行控制
- 系统变量：$AA_SNGLAX_STAT
  单个轴的状态
- OPI 变量：aaSnglAxStat

更多信息

- 功能手册之 SINUMERIK ONE 轴和主轴；章节“定位轴”

12.3.3.3 clampingToleranceExceeded

NC → PLC	<Axis>.basic.in.clampingToleranceExceeded
已超过夹紧公差 (MD36050 $MA_CLAMP_POS_TOL)	

数据类型与更新

BOOL

值 TRUE

已超过夹紧公差。

关联

- MD36050 $MA_CLAMP_POS_TOL
  夹紧公差

12.3.3.4 contTraversActive

NC → PLC	<Axis>.basic.in.contTraversActive
连续运行处于活动状态	

说明

操作移动键或手轮时，轴持续移动。

在 JOG 运行方式下手动运行机床轴的信号：<Axis>.basic.in.contTraversActive

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型与更新

BOOL，循环

值 TRUE

用于手动运行机床轴的机床功能生效。

值 FALSE

用于手动运行机床轴的机床功能未生效。

关联

- <Axis>.basic.out.inc1
  激活“根据请求将轴移动 1 个增量”
- <Axis>.basic.out.inc10
  激活“根据请求将轴移动 10 个增量”
- <Axis>.basic.out.inc100
  激活“根据请求将轴移动 100 个增量”
- <Axis>.basic.out.inc1000
  激活“根据请求将轴移动 1 000 个增量”
- <Axis>.basic.out.inc10000
  激活“根据请求将轴移动 10 000 个增量”
- <Axis>.basic.out.incVar
  激活“根据请求移动轴，移动幅度为用户自定义增量数”
- <Axis>.basic.out.contTravers 连续运行处于活动状态

  请求机床功能

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.5 disabled

NC → PLC	<Axis>.basic.in.disabled
轴锁定，无法运行	

数据类型与更新

BOOL

值 TRUE

轴锁定，无法运行

12.3.3.6 dynBacklashCompActive

NC → PLC	<Axis>.basic.in.dynBacklashCompActive
动态换向间隙补偿处于活动状态	

数据类型与更新

BOOL

值 TRUE

动态换向间隙补偿处于活动状态。

值 FALSE

动态换向间隙补偿未激活。

12.3.3.7 emergencyRetractionActive

NC → PLC	<Axis>.basic.in.emergencyRetractionActive
紧急回退生效	

数据类型与更新

BOOL

值 TRUE

ESR 响应触发

值 FALSE

未触发 ESR 响应

12.3.3.8 enc1Active

NC → PLC	<Axis>.basic.in.enc1Active
位置测量系统 1 处于活动状态	

说明

位置测量系统 1 已接通

数据类型与更新

BOOL，循环

值 TRUE

位置测量系统 1 处于“主动”或者“被动”状态。已激活对位置测量系统的监控和更新。

值 FALSE

位置测量系统 1 处于“驻留”状态。位置测量系统的监控和更新关闭。

关联

- <Axis>.basic.out.enc1Activation
  激活位置测量系统 1

更多信息

SINUMERIK ONE 监控与补偿功能手册：

- “轴监控” > “驻留机床轴”章节
- “轴监控” > “被动位置测量系统驻留”章节

12.3.3.9 enc1FreqLimitExceeded

NC → PLC	<Axis>.basic.in.enc1FreqLimitExceeded
已超过测量系统 1 的编码器频率极限	

说明

待监控编码器极限频率通过以下机床数据进行设置：MD36300 $MA_ENC_FREQ_LIMIT

编码器极限频率监控针对的是在 NC/PLC 接口中选择的主动测量系统：

<Axis>.basic.out.enc1Activation 和 <Axis>.basic.out.enc2Activation。

数据类型与更新

BOOL，循环

值 TRUE

一旦超过测量系统 1 的编码器极限频率，适用以下规则：

- <Axis>.basic.in.enc1Synchronized = FALSE
  机床轴位置测量系统 1 已回参考点/同步
- <Axis>.basic.in.posCtrlActive = FALSE
  位置控制器处于活动状态

- <Axis>.basic.in.posCtrlActive = TRUE
  只在一个主轴（位置控制器处于活动状态）
- 轴通过快速停止进入静止状态

值 FALSE

如未超过测量系统 1 的编码器极限频率，或者一度超过，后来重新低于用于编码器重新同步的编码器极限频率：

- MD36302 $MA_ENC_FREQ_LIMIT_LOW

关联

- <Axis>.basic.out.enc1Activation
  激活位置测量系统 1
- <Axis>.basic.out.enc2Activation
  激活位置测量系统 2
- MD36300 $MA_ENC_FREQ_LIMIT
- MD36302 $MA_ENC_FREQ_LIMIT_LOW

更多信息

SINUMERIK ONE 监控与补偿功能手册：

- “轴监控” > “测量系统监控” > “编码器极限频率监控”章节

12.3.3.10 enc1Restored

NC → PLC	<Axis>.basic.in.enc1Restored
机床轴位置测量系统 1 的位置已恢复	

说明

MD34210 $MA_ENC_REFP_STATE == 3

在采用距离编码增量测量系统时，控制系统启动后会恢复关闭前缓存中最后的轴位置。此时不会自动回参考点。位置测量系统处于“位置已恢复”状态。

数据类型与更新

BOOL，循环

值 TRUE

机床轴的测量系统 1 的位置已恢复。

值 FALSE

机床轴的测量系统 1 的位置未恢复。

关联

- MD34102 $MA_REFP_SYNC_ENCS
  测量系统校准
- MD34210 $MA_ENC_REFP_STATE
  绝对值编码器校准状态

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“回参考点”章节

12.3.3.11 enc1Synchronized

NC → PLC	<Axis>.basic.in.enc1Synchronized
机床轴位置测量系统 1 已回参考点/同步	

说明

- 轴：
  在成功执行回参考点或同步后，此接口信号置位。
- 主轴：
  最迟在主轴旋转一周（360 度）后，在越过零脉冲，或者非接触式接近开关作出响应的情况下，此接口信号置位。

数据类型与更新

BOOL，循环

值 TRUE

机床轴的位置测量系统 1 已回参考点/已同步。

值 FALSE

机床轴的位置测量系统 1 未回参考点/未同步。

关联

- <Axis>.basic.out.enc1Activation
  激活位置测量系统 1
- <Axis>.basic.out.refPointSelection
  选择参考点 (1 - 4)：最多可以使用四个不同的参考点 (MD34100 $MA_REFP_SET_POS)

- <Axis>.basic.out.refPointApproachCam
  到达参考凸轮，低转速接近参考点
- <Axis>.basic.in.enc1Restored
  机床轴位置测量系统 1 的位置已恢复
- MD34102 $MA_REFP_SYNC_ENCS
  测量系统校准

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“回参考点”章节

12.3.3.12 enc2Active

NC → PLC	<Axis>.basic.in.enc2Active
位置测量系统 2 处于活动状态	

说明

位置测量系统 2 已接通

数据类型与更新

BOOL，循环

值 TRUE

位置测量系统 2 处于“主动”或者“被动”状态。已激活对位置测量系统的监控和更新。

值 FALSE

位置测量系统 2 处于“驻留”状态。位置测量系统的监控和更新关闭。

关联

- <Axis>.basic.out.enc1Activation
  激活位置测量系统 1

更多信息

SINUMERIK ONE 监控与补偿功能手册：

- “轴监控”>“驻留机床轴”章节
- “轴监控”>“被动位置测量系统驻留”章节

12.3.3.13 enc2FreqLimitExceeded

NC → PLC	<Axis>.basic.in.enc2FreqLimitExceeded
已超过测量系统 2 的编码器频率极限	

说明

待监控编码器极限频率通过以下机床数据进行设置：MD36300 $MA_ENC_FREQ_LIMIT

编码器极限频率监控针对的是在 NC/PLC 接口中选择的主动测量系统：

<Axis>.basic.out.enc1Activation 和 <Axis>.basic.out.enc2Activation。

数据类型与更新

BOOL，循环

值 TRUE

一旦超过测量系统 2 的编码器极限频率，适用以下规则：

- <Axis>.basic.in.enc2Synchronized = FALSE
  机床轴位置测量系统 2 已回参考点/同步
- <Axis>.basic.in.posCtrlActive = FALSE
  位置控制器处于活动状态
- <Axis>.basic.in.posCtrlActive = TRUE
  只在一个主轴（位置控制器处于活动状态）
- 轴通过快速停止进入静止状态

值 FALSE

如未超过测量系统 2 的编码器极限频率，或者一度超过，后来重新低于用于编码器重新同步的编码器极限频率：

- MD36302 $MA_ENC_FREQ_LIMIT_LOW

关联

- <Axis>.basic.out.enc1Activation
  激活位置测量系统 1
- <Axis>.basic.out.enc2Activation
  激活位置测量系统 2
- MD36300 $MA_ENC_FREQ_LIMIT
- MD36302 $MA_ENC_FREQ_LIMIT_LOW

更多信息

SINUMERIK ONE 监控与补偿功能手册

- “轴监控” > “测量系统监控” > “编码器极限频率监控”章节

12.3.3.14 enc2Restored

NC → PLC	<Axis>.basic.in.enc2Restored
机床轴位置测量系统 2 的位置已恢复	

说明

MD34210 $MA_ENC_REFP_STATE == 3

在采用距离编码增量测量系统时，控制系统启动后会恢复关闭前缓存中最后的轴位置。此时不会自动回参考点。位置测量系统处于“位置已恢复”状态。

数据类型与更新

BOOL，循环

值 TRUE

机床轴的测量系统 2 的位置已恢复。

值 FALSE

机床轴的测量系统 2 的位置未恢复。

关联

- MD34102 $MA_REFP_SYNC_ENCS
  测量系统校准
- MD34210 $MA_ENC_REFP_STATE
  绝对值编码器校准状态

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“回参考点”章节

12.3.3.15 enc2Synchronized

NC → PLC	<Axis>.basic.in.enc2Synchronized
机床轴位置测量系统 2 已回参考点/同步	

说明

- 轴
  在成功执行回参考点或同步后，此接口信号置位。
- 主轴
  最迟在主轴旋转一周（360 度）后，在越过零脉冲，或者非接触式接近开关作出响应的情况下，此接口信号置位。

数据类型与更新

BOOL，循环

值 TRUE

机床轴的位置测量系统 2 已回参考点/已同步。

值 FALSE

机床轴的位置测量系统 2 未回参考点/未同步。

关联

- <Axis>.basic.out.enc2Activation
  激活位置测量系统 2
- <Axis>.basic.out.refPointSelection
  选择参考点 (1 - 4)：最多可以使用四个不同的参考点 (MD34100 $MA_REFP_SET_POS)
- <Axis>.basic.out.refPointApproachCam
  到达参考凸轮，低转速接近参考点
- <Axis>.basic.in.enc2Restored
  机床轴位置测量系统 2 的位置已恢复
- MD34102 $MA_REFP_SYNC_ENCS
  测量系统校准

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“回参考点”章节

12.3.3.16 fixedStopForceLimited

NC → PLC	<Axis>.basic.in.fixedStopForceLimited
固定挡块力受限	

数据类型与更新

BOOL

值 TRUE

力限制处于活动状态

12.3.3.17 fixedStopReached

NC → PLC	<Axis>.basic.in.fixedStopReached
到达固定挡块	

应用示例

为设定可编程的夹紧转矩，在模拟驱动中将此信号用于：将执行器从转速闭环控制切换至电流或转矩闭环控制。

数据类型与更新

BOOL，循环

值 TRUE

已到达固定挡块。

值 FALSE

未到达固定挡块。

关联

- <Axis>.basic.out.fixedStopAckReached
  应答：“到达固定挡块”，继续处理程序
- <Axis>.basic.out.fixedStopSensor
  到达固定挡块 (MD37060 $MA_FIXED_STOP_ACKN_MASK)
- <Axis>.basic.out.fixedStopEnable
  确认固定挡块激活请求
- <Axis>.basic.in.fixedStopRequest
  请求使能“运行到固定挡块”功能
- <Axis>.basic.in.fixedStopReached
  到达固定挡块

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“运行到固定挡块”章节

12.3.3.18 fixedStopRequest

NC → PLC	<Axis>.basic.in.fixedStopRequest
请求使能“运行到固定挡块”功能	

说明

通过 <Axis>.basic.out.fixedStopEnable（确认固定挡块激活请求）进行反馈。

数据类型与更新

BOOL，循环

值 TRUE

请求对运行至固定挡块的使能

值 FALSE

无请求

关联

- <Axis>.basic.out.fixedStopEnable
  确认固定挡块激活请求

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“运行到固定挡块”章节

12.3.3.19 followUpModeActive

NC → PLC	<Axis>.basic.in.followUpModeActive
跟踪运行处于活动状态	

说明

在“停止”状态下，此信号不置位。

注意

删除剩余行程

在以下状态过渡中，控制器内部会执行剩余行程删除：

- 从“跟踪”状态进入“停止”状态 (<Axis>.basic.out.followUpMode) = 1 → 0（跟踪运行）
- 从“跟踪”状态进入“位置闭环控制”状态 (<Axis>.basic.out.enableCtrl) = TRUE（控制器使能）

数据类型与更新

BOOL，循环

值 TRUE

轴/主轴的跟踪运行生效。

可能的原因：

- 驱动的伺服使能取消：
  - <Axis>.basic.out.enableCtrl == FALSE
    请求控制器使能
  - 控制系统内部存在故障；参见下文的“更多参考”
- 跟踪运行已选择：
  - <Axis>.basic.out.followUpMode == TRUE
    激活跟踪运行
  - 控制器内部，例如在取消正在运行的轴的控制器使能时

在跟踪运行期间：位置设定值=位置实际值

说明

静态监控和夹紧监控不生效。

值 FALSE

轴/主轴的跟踪运行未生效。

静态监控和夹紧监控生效。

关联

- <Axis>.basic.out.enableCtrl
  请求控制器使能
- <Axis>.basic.out.followUpMode
  激活跟踪运行

更多信息

- SINUMERIK ONE SINAMICS S120 报警诊断手册

12.3.3.20 handwheelAssignedNumber

NC → PLC	<Axis>.basic.in.handwheelAssignedNumber
当前所分配手轮的编号：1..6 或 0 未分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）	

说明

当前所分配手轮的编号。

该接口使用了二进制编码。通过机床数据 MD11324=1 进行定义。

若指定关系生效，则可通过手轮在 JOG 运行方式下运行机床轴，或者在 AUTO 或 MDI 方式下产生 DRF 偏移。

数据类型和更新

USInt，循环

值

0：未分配手轮

1...6：当前所分配手轮的编号

关联

- MD11324 $MN_HANDWH_VDI_REPRESENTATION
  VDI 接口中的手轮编号显示
- <Axis>.basic.out.handwheelSelection
  手轮 1..6 已分配或 0 未分配手轮（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.21 handwheelDirInverted

NC → PLC	<Axis>.basic.in.handwheelDirInverted
手轮旋转方向反转	

数据类型与更新

BOOL，循环

值 TRUE

与机床轴对应的手轮的旋转方向取反生效。

值 FALSE

与机床轴对应的手轮的旋转方向取反未生效。

关联

- <Axis>.basic.out.handwheelInvertDir
  反转手轮旋转方向

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.22 handwheelOvrActive

NC → PLC	<Axis>.basic.in.handwheelOvrActive
手轮超驰生效：手轮可用于移动轴	

说明

可用手轮移动轴。

对于定位轴而言，手轮脉冲或是作为行程设定（在 FDA[<轴>]=0 时），或是作为速度叠加（在 FDA[<轴>]>0 时）对编程的轴进给率生效。

在下面的条件下该叠加失效：

- 定位轴到达了编程的目标位置。
- 删除了剩余行程。
- 触发了复位。

说明

在通过 "SINU_TraversePosAxis" 为并行定位轴激活了“自动模式下的手轮超驰”的情况下，此接口信号也会置位。

数据类型和更新

BOOL，循环

值 TRUE

“AUTO 模式下的手轮叠加”功能对编程的定位轴（FDA[<轴>]）生效。

值 FALSE

"AUTO 模式下的手轮叠加"功能不对编程的定位轴生效。

关联：

- <Axis>.basic.out.resetMovement
  脉冲沿切换 0 → 1：已请求轴"删除剩余行程"；已请求主轴"复位"

详细信息

- 功能手册之 SINUMERIK ONE 轴和主轴；章节"手动移动"

12.3.3.23 inc1Active

NC → PLC	<Axis>.basic.in.inc1Active
"根据请求将轴移动 1 个增量"处于活动状态	

说明

操作移动键或手轮时，轴移动 1 个增量。

在 JOG 运行方式下手动运行机床轴的信号：<Axis>.basic.in.inc1Active

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求用于手动运行机床轴的机床功能。

值 FALSE

未请求用于手动运行机床轴的机床功能。

关联

- <Axis>.basic.out.inc1
  激活"根据请求将轴移动 1 个增量"
- <Axis>.basic.out.inc10
  激活"根据请求将轴移动 10 个增量"
- <Axis>.basic.out.inc100
  激活"根据请求将轴移动 100 个增量"

- <Axis>.basic.out.inc1000
  激活“根据请求将轴移动 1000 个增量”
- <Axis>.basic.out.inc10000
  激活“根据请求将轴移动 10000 个增量”
- <Axis>.basic.out.incVar
  激活“根据请求移动轴，移动幅度为用户自定义增量数”
- <Axis>.basic.out.contTravers
  连续运行处于活动状态

  请求机床功能

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.24 inc10Active

NC → PLC	<Axis>.basic.in.inc10Active
“根据请求将轴移动 10 个增量”处于活动状态	

说明

操作移动键或手轮时，轴移动 10 个增量。

在 JOG 运行方式下手动运行机床轴的信号：<Axis>.basic.in.inc10Active

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求用于手动运行机床轴的机床功能。

值 FALSE

未请求用于手动运行机床轴的机床功能。

关联

- <Axis>.basic.out.inc1
  激活“根据请求将轴移动 1 个增量”
- <Axis>.basic.out.inc10
  激活“根据请求将轴移动 10 个增量”
- <Axis>.basic.out.inc100
  激活“根据请求将轴移动 100 个增量”
- <Axis>.basic.out.inc1000
  激活“根据请求将轴移动 1000 个增量”
- <Axis>.basic.out.inc10000
  激活“根据请求将轴移动 10000 个增量”
- <Axis>.basic.out.incVar
  激活“根据请求移动轴，移动幅度为用户自定义增量数”
- <Axis>.basic.out.contTravers
  连续运行处于活动状态

  请求机床功能

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.25 inc100Active

NC → PLC	<Axis>.basic.in.inc100Active
“根据请求将轴移动 100 个增量”处于活动状态	

说明

操作移动键或手轮时，轴移动 100 个增量。

在 JOG 运行方式下手动运行机床轴的信号：<Axis>.basic.in.inc100Active

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求用于手动运行机床轴的机床功能。

值 FALSE

未请求用于手动运行机床轴的机床功能。

关联

- <Axis>.basic.out.inc1
  激活“根据请求将轴移动 1 个增量”
- <Axis>.basic.out.inc10
  激活“根据请求将轴移动 10 个增量”
- <Axis>.basic.out.inc100
  激活“根据请求将轴移动 100 个增量”
- <Axis>.basic.out.inc1000
  激活“根据请求将轴移动 1000 个增量”
- <Axis>.basic.out.inc10000
  激活“根据请求将轴移动 10000 个增量”
- <Axis>.basic.out.incVar
  激活“根据请求移动轴，移动幅度为用户自定义增量数”
- <Axis>.basic.out.contTravers
  连续运行处于活动状态

  请求机床功能

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.26 inc1000Active

NC → PLC	<Axis>.basic.in.inc1000Active
“根据请求将轴移动 1000 个增量”处于活动状态	

说明

操作移动键或手轮时，轴移动 1000 个增量。

在 JOG 运行方式下手动运行机床轴的信号：<Axis>.basic.in.inc1000Active

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求用于手动运行机床轴的机床功能。

值 FALSE

未请求用于手动运行机床轴的机床功能。

关联

- <Axis>.basic.out.inc1
  激活“根据请求将轴移动 1 个增量”
- <Axis>.basic.out.inc10
  激活“根据请求将轴移动 10 个增量”
- <Axis>.basic.out.inc100
  激活“根据请求将轴移动 100 个增量”
- <Axis>.basic.out.inc1000
  激活“根据请求将轴移动 1000 个增量”
- <Axis>.basic.out.inc10000
  激活“根据请求将轴移动 10000 个增量”
- <Axis>.basic.out.incVar
  激活“根据请求移动轴，移动幅度为用户自定义增量数”
- <Axis>.basic.out.contTravers
  连续运行处于活动状态

  请求机床功能

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.27 inc10000Active

NC → PLC	<Axis>.basic.in.inc10000Active
“根据请求将轴移动 10000 个增量”处于活动状态	

说明

操作移动键或手轮时，轴移动 10000 个增量。

在 JOG 运行方式下手动运行机床轴的信号：<Axis>.basic.in.inc10000Active

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求用于手动运行机床轴的机床功能。

值 FALSE

未请求用于手动运行机床轴的机床功能。

关联

- <Axis>.basic.out.inc1
  激活“根据请求将轴移动 1 个增量”
- <Axis>.basic.out.inc10
  激活“根据请求将轴移动 10 个增量”
- <Axis>.basic.out.inc100
  激活“根据请求将轴移动 100 个增量”
- <Axis>.basic.out.inc1000
  激活“根据请求将轴移动 1000 个增量”
- <Axis>.basic.out.inc10000
  激活“根据请求将轴移动 10000 个增量”

- <Axis>.basic.out.incVar
  激活“根据请求移动轴，移动幅度为用户自定义增量数”
- <Axis>.basic.out.contTravers
  连续运行处于活动状态

  请求机床功能

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.28 incVarActive

NC → PLC	<Axis>.basic.in.incVarActive
“根据请求移动轴，移动幅度为用户自定义增量数”处于活动状态	

说明

操作移动键或手轮时，轴以数量可变的增量移动。

在 JOG 运行方式下手动运行机床轴的信号：<Axis>.basic.in.incVarActive

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求用于手动运行机床轴的机床功能。

值 FALSE

未请求用于手动运行机床轴的机床功能。

关联

- <Axis>.basic.out.inc1
  激活“根据请求将轴移动 1 个增量”
- <Axis>.basic.out.inc10
  激活“根据请求将轴移动 10 个增量”

- <Axis>.basic.out.inc100
  激活“根据请求将轴移动 100 个增量”
- <Axis>.basic.out.inc1000
  激活“根据请求将轴移动 1000 个增量”
- <Axis>.basic.out.inc10000
  激活“根据请求将轴移动 10000 个增量”
- <Axis>.basic.out.incVar
  激活“根据请求移动轴，移动幅度为用户自定义增量数”
- <Axis>.basic.out.contTravers
  连续运行处于活动状态

  请求机床功能

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.29 indexAxisInPos

NC → PLC	<Axis>.basic.in.indexAxisInPos
分度轴就位	

说明

信号只与分度轴有关。

应用示例

刀库：一旦分度轴就位，从刀库中取刀的抓刀器就激活。该过程由 PLC 用户程序控制。

特殊情况或错误

分度位置表中为各分度记录的轴位置可能因零点偏移（以及 DRF）而改变。

如果分度轴的实际位置为分度位置表中记录的连同补偿值在内的位置值，则接口信号“分度轴就位”置位。

如果自动运行时在分度轴上执行 DRF 偏移，这种情况下即使轴不再位于分度位置上，接口信号“分度轴就位”也继续存在。

数据类型与更新

BOOL，循环

值 TRUE

下列情况下为 TRUE 信号：

- 分度轴以“精准停”到达一个分度位置。
- 分度轴位于一个在 AUTO 运行方式下通过 CAC、CACP、CACN、CDC 或 CIC 定位至的分度位置上。

值 FALSE

下列情况下为 FALSE 信号：

- 该轴不是分度轴。
- 存在运行指令，且分度轴移动。
- 分度轴位于某个非分度位置的位置上。
- 在 AUTO 运行方式下，分度轴不通过 CAC、CACP、CACN、CDC 或 CIC 定位，而是例如通过 AC 或 DC 移动至任意位置。
- 分度轴的“伺服使能”取消。

关联

- <Axis>.basic.in.traversMinusCommandActive
  负方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)
- <Axis>.basic.in.traversPlusCommandActive
  正方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)
- <Axis>.basic.out.enableCtrl
  请求控制器使能
- MD30500 $MA_INDEX_AX_ASSIGN_POS_TAB
  轴为分度轴

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“分度轴”章节

12.3.3.30 isPathAxis

NC → PLC	<Axis>.basic.in.isPathAxis
轴为轨迹轴	

说明

与程序段搜索类型 5（SERUPRO）配合使用时，在“已找到目标程序段”状态下，该接口信号基于轴在目标程序段中的特性。

数据类型与更新

BOOL，循环

值 TRUE

轴为轨迹轴。亦即，该轴与其他轴一起在轨迹上运行（轨迹轴）。

值 FALSE

轴不是轨迹轴。

更多信息

SINUMERIK ONE 基础功能手册：

- “运行方式组、通道、程序运行、复位行为”>“程序段搜索类型 5 (SERUPRO)”>“再次返回轮廓 (REPOS)”章节

12.3.3.31 isPosAxis

NC → PLC	<Axis>.basic.in.isPosAxis
轴为定位轴	

数据类型与更新

BOOL，循环

值 TRUE

轴为定位轴。

值 FALSE

轴不是定位轴。

关联

- <Axis>.basic.in.posAxisFeedrateValue
  作为定位轴移动的轴的进给值

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“定位轴”章节

12.3.3.32 isStationary

NC → PLC	<Axis>.basic.in.isStationary
轴达到静止转速范围 (n<n_min)	

说明

在存在针对轴/主轴的运行指令 (<Axis>.basic.in.traversMinusCommandActive, <Axis>.basic.in.traversPlusCommandActive) 的情况下，即便轴的当前速度或主轴的实际转速小于等于参数设置的限值，此信号也总是为 0。

数据类型与更新

BOOL，循环

值 TRUE

进给轴/主轴停止。亦即，轴的当前速度或主轴的实际转速小于等于参数设置的限值：

MD36060 $MA_STANDSTILL_VELO_TOL

值 FALSE

进给轴/主轴不处于静止状态。亦即，轴的当前速度或主轴的实际转速大于等于参数设置的限值：

MD36060 $MA_STANDSTILL_VELO_TOL

若存在运行指令（例如主轴上），那么该信号始终 = 0，即便当前转速低于 MD36060 也同样如此。

如果报告".isStationary"接口信号且主轴上位置闭环控制未生效，则在操作界面上实际转速显示为零并且系统变量 $AA_S[n] 读取零值。

关联

- MD36060 $MA_STANDSTILL_VELO_TOL
  "轴/主轴停止"信号的最大速度/转速

12.3.3.33 jogFixedPointReachedNumber

NC → PLC	<Axis>.basic.in.jogFixedPointReachedNumber
通过"精准停"到达待接近的设置模式定点 (1..4)	

说明

若轴在“手动运行至固定点”中以“精准停”到达待定位至的固定点位置，则系统通过 <Axis>.basic.in.jogFixedPointReachedNumber 将定位至的固定点的编号反馈给 PLC。当轴通过例如 NC 程序、SINU_TraversePosAxi”、“SINU_TraverseIndexingAxis”、“SINU_PositionSpindle”或同步动作等其他方法到达机床坐标系中的目标固定点位置，且实际位置位于“精准停”公差窗口内时，也进行此反馈。

数据类型和更新

USInt，循环

值 0

未接近定点

值 1 ... 4

定位至的固定点的编号

关联

- <Axis>.basic.out.jogFixedPointSelection
  选择点动定点 (1 - 4)（MD30600 $MA_FIX_POINT_POS)
- <Axis>.basic.in.jogFixedPointSelectedNumber
  接近设置模式定点 (1..4)
- MD30600 $MA_FIX_POINT_POS[<n>]
  轴的固定值位置
- MD36010 $MA_STOP_LIMIT_FINE
  精准停

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.34 jogFixedPointSelectedNumber

NC → PLC	<Axis>.basic.in.jogFixedPointSelectedNumber
接近点动定点 (1..4)	

说明

“点动接近定点”功能处于活动状态时，通过 <Axis>.basic.in.jogFixedPointSelectedNumber 将待接近定点的编号反馈给 PLC。现在可以用移动键或手轮将选择的机床轴移动到相应的固定点。定点通过机床数据 MD30600 定义。

数据类型和更新

USInt，循环

值 0

不接近任何定点

值 1 ... 4

将运行到的固定点的编号

关联

- <Axis>.basic.out.jogFixedPointSelection
  选择点动定点 (1 - 4)（MD30600 $MA_FIX_POINT_POS)
- <Axis>.basic.in.jogFixedPointReachedNumber
  通过“精准停”到达要接近的点动定点 (1 - 4)
- MD30600 $MA_FIX_POINT_POS[<n>]
  轴的固定值位置

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.35 jogPosReached

NC → PLC	<Axis>.basic.in.jogPosReached
在“点动运行至位置”功能下，轴已到达通过 SD43320“精准停”设定的位置	

说明

通过移动键或手轮启动运行至目标位置。轴运行至位置，到达后自动停止。

数据类型与更新

BOOL，循环

值 TRUE

在“手动运行至位置”中，轴已以“精准停”到达通过 SD43320 设定的位置。

值 FALSE

在“手动运行至位置”中，轴尚未到达通过 SD43320 设定的位置。

关联

- <Axis>.basic.out.jogToPos
  已请求“点动运行至位置”功能 (SD43320 $SA_JOG_POSITION)
- <Axis>.basic.in.jogToPosActive
  “点动运行至位置”功能处于活动状态
- SD43320 $SA_JOG_POSITION
  点动位置

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.36 jogToPosActive

NC → PLC	<Axis>.basic.in.jogToPosActive
“点动运行至位置”功能处于活动状态	

说明

现在可以用移动键或手轮将选择的机床轴移动至通过设定数据 SD43320 设定的位置。

数据类型与更新

BOOL，循环

值 TRUE

“手动运行至位置”功能生效。

值 FALSE

“手动运行至位置”功能未生效。

关联

- <Axis>.basic.out.jogToPos
  已请求“点动运行至位置”功能 (SD43320 $SA_JOG_POSITION)
- <Axis>.basic.in.jogPosReached
  在“点动运行至位置”功能下，轴已到达通过 SD43320“精准停”设定的位置
- SD43320 $SA_JOG_POSITION
  点动位置

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.37 lubricationPulseActive

NC → PLC	<Axis>.basic.in.lubricationPulseActive
已移动在机床数据 (MD33050 $MA_LUBRICATION_DIST) 中编程设置的距离	

说明

一旦轴/主轴经过在机床数据中设置的行程，该接口信号便会取反。

在每次控制系统启动后，行程测量会重新开始。

数据类型与更新

BOOL，循环

脉冲沿 0 → 1

已经过在机床数据（MD33050 $MA_LUBRICATION_DIST）中设置的行程。

脉冲沿 1 → 0

已经过在机床数据（MD33050 $MA_LUBRICATION_DIST）中设置的行程。

关联

- MD33050 $MA_LUBRICATION_DIST
  润滑脉冲距离

12.3.3.38 measurementActive

NC → PLC	<Axis>.basic.in.measurementActive
“测量”功能处于活动状态（工件或刀具等）	

说明

接口信号显示轴的当前测量状态，可在所有测量功能下进行处理。

数据类型与更新

BOOL，循环

值 TRUE

测量功能生效。

值 FALSE

测量功能未生效。

更多信息

- SINUMERIK ONE 基础功能手册；“测量”章节

12.3.3.39 moduloLimitSwitchActive

NC → PLC	<Axis>.basic.in.moduloLimitSwitchActive
模数回转轴的行进范围限制处于活动状态（软件限位开关、工作区域限制）	

说明

应用示例：周边磨砂轮，带有监控

说明

在直线轴/回转轴无模数功能的情况下，此信号不相关。

数据类型与更新

BOOL，循环

值 TRUE

对于模数回转轴，运行范围限制（软件限位开关、工作区域限制）生效。

值 FALSE

对于模数回转轴，运行范围限制未生效。

关联

- <Axis>.basic.out.enableModuloLimitSwitch
  激活模数回转轴的行进范围限制（软件限位开关、工作区域限制等）

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“回转轴”章节

12.3.3.40 posAxisFeedrateValue

NC → PLC	<Axis>.basic.in.posAxisFeedrateValue
作为定位轴移动的轴的进给值	

说明

若轴作为定位轴运行，可通过此接口信号读取其进给值。

输出时间可通过以下机床数据定义：

MD22240 $MC_AUXFU_F_SYNC_TYPE

说明

- 缺省设置下输出被抑制，因为在连续路径运行中输出进给率（辅助功能输出）可能会造成速度跃变。
- 最后的进给值一直保持生效，直至其被新的进给值覆盖。

数据类型与更新

REAL，循环

值

轴的进给值

关联

- <Axis>.basic.in.isPosAxis
  轴为定位轴
- MD22240 $MC_AUXFU_F_SYNC_TYPE
  F 功能的输出时间
- MD32060 $MA_POS_AX_VELO
  定位轴速度的复位位置

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“分度轴”章节

12.3.3.41 posCoarseReached

NC → PLC	<Axis>.basic.in.posCoarseReached
在“粗准停”极限内到达编程的位置 (MD36000 $MA_STOP_LIMIT_COARSE)	

数据类型与更新

BOOL，循环

值 TRUE

出现以下状态中的至少一个：

- 不存在轴运行，且轴的实际位置处于参数设置的准停极限 MD36000 $MA_STOP_LIMIT_COARSE 内
- 控制系统处于“复位”状态
- 轴最近一次被编写为定位轴或定位主轴
- 轨迹运行已通过 NC 停止结束
- 主轴处于位置闭环控制（SPCON/SPOS）中并且静止
- 轴已从转速闭环控制切换至位置闭环控制

值 FALSE

出现以下状态中的至少一个：

- 存在轴运行
- 轴的实际位置处于参数设置的准停极限 MD36000 $MA_STOP_LIMIT_COARSE 外
- 主轴处于转速闭环控制（SPCOF/SPOSA）中
- 轴处于“跟踪运行”状态
- 轴处于“驻留”状态
- 轴已从位置闭环控制切换至转速闭环控制

关联

- MD36000 $MA_STOP_LIMIT_COARSE
  粗准停

12.3.3.42 posCtrlActive

NC → PLC	<Axis>.basic.in.posCtrlActive
位置控制器处于活动状态	

说明

- 无位置闭环控制的主轴
  就无位置闭环控制的主轴而言，此信号总是为 FALSE。
- 采用位置闭环控制的主轴
  就能够进行位置闭环控制的主轴而言，在通过 SPCON 或 M70 等接通位置闭环控制后，如针对受位置闭环控制的轴那样对信号进行处理。

- 悬挂轴
  一旦位置闭环控制不再处于活动状态，就要在悬挂轴上激活抱闸
  （<Axis>.basic.in.posCtrlActive == FALSE
  位置控制器处于活动状态）。
- 模拟轴
  也可为模拟轴设置信号输出：
  MD30350 $MA_SIMU_AX_VDI_OUTPUT = 1

数据类型与更新

BOOL，循环

值 TRUE

位置控制器处于活动状态。即轴/主轴的位置控制环闭合。

值 FALSE

位置控制器未激活，即轴/主轴的位置控制环未闭合。

在控制器使能 (<Axis>.basic.out.enableCtrl == FALSE) 通过 PLC 用户程序或因内部故障复位的情况下，此信号复位。

关联

- <Axis>.basic.out.enableCtrl
  请求控制器使能
- <Axis>.basic.out.followUpMode
  激活跟踪运行
- <Axis>.basic.out.enc1Activation
  激活位置测量系统 1
- <Axis>.basic.out.enc2Activation
  激活位置测量系统 2

更多信息

- SINUMERIK ONE SINAMICS S120 报警诊断手册

12.3.3.43 posFineReached

NC → PLC	<Axis>.basic.in.posFineReached
在“精准停”极限内到达编程的位置 (MD36150 $MA_STOP_LIMIT_FINE)	

数据类型与更新

BOOL，循环

值 TRUE

出现以下状态中的至少一个：

- 不存在轴运行，且轴的实际位置处于参数设置的准停极限 MD36010 $MA_STOP_LIMIT_FINE 内
- 控制系统处于“复位”状态
- 轴最近一次被编写为定位轴或定位主轴
- 轨迹运行已通过 NC 停止结束
- 主轴处于位置闭环控制（SPCON/SPOS）中并且静止
- 轴已从转速闭环控制切换至位置闭环控制

值 FALSE

出现以下状态中的至少一个：

- 存在轴运行
- 轴的实际位置处于参数设置的准停极限 MD36010 $MA_STOP_LIMIT_FINE 外
- 主轴处于转速闭环控制（SPCOF/SPOSA）中
- 轴处于“跟踪运行”状态
- 轴处于“驻留”状态
- 轴已从位置闭环控制切换至转速闭环控制

关联

- MD36010 $MA_STOP_LIMIT_FINE
  精准停

12.3.3.44 ready

NC → PLC	<Axis>.basic.in.ready
轴运行就绪	

说明

在连接有轴的驱动的 NCU 上处理此信号。若此 NCU 上显示与轴相关的带系统响应“NC ...”或“运行方式组 ...”的报警，或者“通道未就绪”，则轴不处于运行就绪状态。

数据类型与更新

BOOL，循环

值 TRUE

轴运行就绪。

值 FALSE

轴尚未运行就绪。

12.3.3.45 reposDelayAck

NC → PLC	<Axis>.basic.in.reposDelayAck
已应答 REPOS 延迟	

数据类型与更新

BOOL，循环

值 TRUE

对 REPOS 延时进行应答。

说明

对于轴而言存在 REPOS 偏移，且"REPOS 延时"生效：
<Axis>.basic.out.reposDelay == TRUE（REPOS 延迟）
已在运行程序段内编写轴并运行 REPOS 偏移。
接口信号的特性类似：
<Chan>.basic.in.reposModeSelectedNumber（REPOS 接近模式应答）

值 FALSE

尚未对 REPOS 延时进行应答，或者不存在 REPOS 偏移。

值	REPOS 接近运行方式
0	无 REPOS 运行方式
1	接近程序段起点 (RMBBL)
2	接近中断点 (RMIBL)
3	接近程序段终点 (RMEBL)
4	接近下一个轨迹点 (RMNBL)

关联

- <HmiChan>.basic.in.reposModeChanged
  HMI：REPOS 运行方式切换
- <Chan>.basic.out.reposModeSelection
  REPOS 接近运行方式：
  可能的值见上表
- <Chan>.basic.out.reposModeActivationRequest
  请求为当前主运行程序段激活"REPOS"运行方式
- <Chan>.basic.in.reposModeChangeAck
  应答 REPOS 接近运行方式的变更请求
- <Chan>.basic.in.reposModeSelectedNumber
  选择的 REPOS 接近运行方式：
  可能的值见上表
- <Chan>.basic.in.reposDelayActive
  "在 REPOS 接近运行方式下延迟"处于活动状态
- <Axis>.basic.out.reposDelay
  已请求 REPOS 延迟
- <Axis>.basic.in.reposOffsetActive
  REPOS 偏移已应用在轴上
- <Axis>.basic.in.reposOffsetValid
  REPOS 偏移计算为有效
- <Axis>.basic.in.isPathAxis
  轴为轨迹轴
- MD11470 $MN_REPOS_MODE_MASK
  重新定位特性

更多信息

SINUMERIK ONE 基础功能手册：

- "运行方式组、通道、程序运行、复位行为" > "程序段搜索类型 5 (SERUPRO)" > "再次返回轮廓 (REPOS)"章节

12.3.3.46 reposOffsetActive

NC → PLC	<Axis>.basic.in.reposOffsetActive
REPOS 偏移已应用在轴上	

数据类型与更新

BOOL，循环

值 TRUE

必须为轴运行 REPOS 偏移。

值 FALSE

不可以为轴运行 REPOS 偏移。

值	REPOS 接近运行方式
0	无 REPOS 运行方式
1	接近程序段起点 (RMBBL)
2	接近中断点 (RMIBL)
3	接近程序段终点 (RMEBL)
4	接近下一个轨迹点 (RMNBL)

关联

- <HmiChan>.basic.in.reposModeChanged
  HMI：REPOS 运行方式切换
- <Chan>.basic.out.reposModeSelection
  REPOS 接近运行方式：
  可能的值见上表
- <Chan>.basic.out.reposModeActivationRequest
  请求为当前主运行程序段激活“REPOS”运行方式
- <Chan>.basic.in.reposModeChangeAck
  应答 REPOS 接近运行方式的变更请求
- <Chan>.basic.in.reposModeSelectedNumber
  已选 REPOS 接近运行方式：
- 可能的值见上表
- <Chan>.basic.in.reposDelayActive
  “在 REPOS 接近运行方式下延迟”处于活动状态

- <Axis>.basic.out.reposDelay
  已请求 REPOS 延迟
- <Axis>.basic.in.reposOffsetValid
  REPOS 偏移计算为有效
- <Axis>.basic.in.reposDelayAck
  已应答 REPOS 延迟
- <Axis>.basic.in.isPathAxis
  轴为轨迹轴
- MD11470 $MN_REPOS_MODE_MASK
  重新定位特性

更多信息

SINUMERIK ONE 基础功能手册：

- “运行方式组、通道、程序运行、复位行为”>“程序段搜索类型 5 (SERUPRO)”>“再次返回轮廓 (REPOS)”章节

12.3.3.47 reposOffsetValid

NC → PLC	<Axis>.basic.in.reposOffsetValid
REPOS 偏移计算为有效	

数据类型与更新

BOOL，循环

值 TRUE

REPOS 偏移算作有效。

值 FALSE

REPOS 偏移算作无效。

值	REPOS 接近运行方式
0	无 REPOS 运行方式
1	接近程序段起点 (RMBBL)
2	接近中断点 (RMIBL)
3	接近程序段终点 (RMEBL)
4	接近下一个轨迹点 (RMNBL)

关联

- <HmiChan>.basic.in.reposModeChanged
  HMI：REPOS 运行方式切换
- <Chan>.basic.out.reposModeSelection
  REPOS 接近运行方式：
  可能的值见上表
- <Chan>.basic.out.reposModeActivationRequest
  请求为当前主运行程序段激活"REPOS"运行方式
- <Chan>.basic.in.reposModeChangeAck
  应答 REPOS 接近运行方式的变更请求
- <Chan>.basic.in.reposModeSelectedNumber
  已选 REPOS 接近运行方式：
  可能的值见上表
- <Chan>.basic.in.reposDelayActive
  "在 REPOS 接近运行方式下延迟"处于活动状态
- <Axis>.basic.out.reposDelay
  已请求 REPOS 延迟
- <Axis>.basic.in.reposOffsetActive
  REPOS 偏移已应用在轴上
- <Axis>.basic.in.reposDelayAck
  已应答 REPOS 延迟
- <Axis>.basic.in.isPathAxis
  轴为轨迹轴
- MD11470 $MN_REPOS_MODE_MASK
  重新定位属性

更多信息

SINUMERIK ONE 基础功能手册：

- "运行方式组、通道、程序运行、复位行为" > "程序段搜索类型 5 (SERUPRO)" > "再次返回轮廓 (REPOS)"章节

12.3.3.48 revolutionalFeedrateActive

NC → PLC	<Axis>.basic.in.revolutionalFeedrateActive
已选中 G95 功能：旋转进给处于活动状态，设定值以 mm/圈为单位设置	

说明

此接口信号显示，轴在 JOG 或 AUTO 运行方式下作为定位轴以旋转进给率运行。

数据类型与更新

BOOL，循环

值 TRUE

旋转进给率（G95）生效。

值 FALSE

旋转进给率（G95）未生效。

关联

- SD41100 $SN_JOG_REV_IS_ACTIVE
  JOG：旋转/直线进给
- SD41120 $SN_JOG_REV_SET_VELO
  JOG 模式下轴的旋转进给
- SD42600 $SC_JOG_FEED_PER_REV_SOURCE
  JOG 模式下的旋转进给控制
- SD43300 $SA_ASSIGN_FEED_PER_REV_SOURCE
  定位轴/主轴的旋转进给
- MD32040 $MA_JOG_REV_VELO_RAPID
  JOG 模式下的旋转进给，含快速运行叠加
- MD32050 $MA_JOG_REV_VELO
  JOG 模式下的旋转进给

更多信息

SINUMERIK ONE 轴和主轴功能手册：

- “进给” > ”轨迹进给 F” > ”进给类型 G93、G94、G95”章节

12.3.3.49 rotAxisInPos

NC → PLC	<Axis>.basic.in.rotAxisInPos
回转轴就位	

数据类型与更新

BOOL

值 TRUE

轴就位

值 FALSE

轴未就位

12.3.3.50 speedCtrlActive

NC → PLC	<Axis>.basic.in.speedCtrlActive
转速控制器处于活动状态	

说明

- 无位置闭环控制的主轴
  对于不带位置闭环控制的主轴，信号可用作 <Axis>.basic.out.enableCtrl（请求控制器使能）信号的反馈。
- 模拟轴
  也可为模拟轴设置信号输出：
  MD30350 $MA_SIMU_AX_VDI_OUTPUT = 1

数据类型与更新

BOOL，循环

值 TRUE

转速控制器处于活动状态，即轴/主轴的转速控制环闭合。

值 FALSE

转速控制器未激活，即轴的转速控制环未闭合。

转速控制器输出被删除。

关联

- <Axis>.basic.in.posCtrlActive
  位置控制器处于活动状态

12.3.3.51 spindleModeActive

NC → PLC	<Axis>.basic.in.spindleModeActive
主轴运行处于活动状态，作为：控制运行、摆动运行、定位运行、同步运行，或者带补偿卡盘的攻丝	

说明

轴作为主轴或回转轴运行：

- 车床：主轴 ↔ C 轴
- 铣床：主轴 ↔ 用于刚性攻丝的回转轴

数据类型与更新

BOOL，循环

值 TRUE

在下列运行方式中，将机床轴作为主轴运行：

- 控制运行
- 摆动运行
- 定位模式
- 带弹性卡头的攻丝
- 同步模式

值 FALSE

机床轴作为回转轴运行。

关联

- <Axis>.spindle.in.requestedGearStageNumber
  所请求齿轮箱挡位 (1 - 5) 的编号
- <Axis>.spindle.in.sFuncValue
  指定的 S 功能值 (REAL)：主轴转速（转/分钟）或恒定切削速度（每分钟转数），具体取决于选择的运行方式（G97 或 G96）

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“主轴”章节

12.3.3.52 swCamsActive

NC → PLC	<Axis>.basic.in.swCamsActive
软件凸轮应用处于活动状态，到达一个软件凸轮后，PLC 发布相应的信号	

数据类型与更新

BOOL，循环

值 TRUE

轴的正负挡块信号向 PLC 接口的输出生效。

值 FALSE

轴的正负挡块信号向 PLC 接口的输出未生效。

关联

- <Nc>.basic.in.swCams[x].minus
  直线轴通过负凸轮 0 沿负方向移动（模数回转轴行为不同）
- <Nc>.basic.in.swCams[x].plus
  直线轴通过正凸轮 0 沿正方向移动（模数回转轴行为不同）
- <Axis>.basic.out.swCams
  激活“输出软件凸轮正信号和负信号”

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“软件凸轮轴，行程开关信号”章节

12.3.3.53 traversMinusCommandActive

NC → PLC	<Axis>.basic.in.traversMinusCommandActive
负方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)	

说明

运行指令根据 MD17900 $MN_VDI_FUNCTION_MASK, Bit 0 输出。

应用示例：一旦检测到运行指令就松开轴，使轴运行。

说明

对于在检测到运行指令时才松开夹紧装置的轴而言，不可采用连续路径运行（G64）。

数据类型与更新

BOOL，循环

值 TRUE

负方向运行指令处于活动状态

值 FALSE

负方向运行指令未激活

关联

- <Axis>.basic.out.traversKeyMinus
  在 JOG 运行方式下沿负方向移动机床轴 (MD32020 $MA_JOG_VELO)
- <Axis>.basic.out.traversKeyPlus
  在 JOG 运行方式下沿正方向移动机床轴 (MD32020 $MA_JOG_VELO)
- <Axis>.basic.in.traversMinusRequestActive
  已请求沿负方向移动
- <Axis>.basic.in.traversPlusRequestActive
  已请求沿正方向移动
- MD17900 $MN_VDI_FUNCTION_MASK
  VDI 信号设置

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.54 traversMinusRequestActive

NC → PLC	<Axis>.basic.in.traversMinusRequestActive
已请求沿负方向移动	

说明

视运行方式而定，以不同的方式触发运行请求：

- JOG 运行方式
  移动键"-"
- 运行方式 REF
  触发轴朝参考点运行的移动键。
- AUTO 或 MDI 运行方式
  执行一个包含几何轴运行指令的程序段。

<Axis>.basic.in.travers　运行指令"-"，沿负的轴方向的运行请求
MinusRequestActive

数据类型与更新

BOOL，循环

值 TRUE

运行请求生效

值 FALSE

运行请求未生效

关联

- <Axis>.basic.out.traversKeyMinus
  在 JOG 运行方式下沿负方向移动机床轴 (MD32020 $MA_JOG_VELO)
- <Axis>.basic.out.traversKeyPlus
  在 JOG 运行方式下沿正方向移动机床轴 (MD32020 $MA_JOG_VELO)
- <Axis>.basic.in.traversMinusCommandActive
  负方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)
- <Axis>.basic.in.traversPlusCommandActive
  正方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.55 traversPlusCommandActive

NC → PLC	<Axis>.basic.in.traversPlusCommandActive
正方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)	

说明

运行指令根据 MD17900 $MN_VDI_FUNCTION_MASK, Bit 0 输出。

应用示例：一旦检测到运行指令就松开轴，使轴运行。

说明

对于在检测到运行指令时才松开夹紧装置的轴而言，不可采用连续路径运行（G64）。

数据类型与更新

BOOL，循环

值 TRUE

正方向运行指令处于活动状态

值 FALSE

正方向运行指令未激活

关联

- <Axis>.basic.out.traversKeyMinus
  在 JOG 运行方式下沿负方向移动机床轴 (MD32020 $MA_JOG_VELO)
- <Axis>.basic.out.traversKeyPlus
  在 JOG 运行方式下沿正方向移动机床轴 (MD32020 $MA_JOG_VELO)
- <Axis>.basic.in.traversMinusRequestActive
  已请求沿负方向移动
- <Axis>.basic.in.traversPlusRequestActive
  已请求沿正方向移动
- MD17900 $MN_VDI_FUNCTION_MASK
  VDI 信号设置

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.56 traversPlusRequestActive

NC → PLC	<Axis>.basic.in.traversPlusRequestActive
已请求沿正方向移动	

说明

视运行方式而定，以不同的方式触发运行请求：

- JOG 运行方式
  移动键"+"
- 运行方式 REF
  触发轴朝参考点运行的移动键。
- AUTO 或 MDI 运行方式
  执行一个包含几何轴运行指令的程序段。

数据类型与更新

BOOL，循环

值 TRUE

运行请求生效

值 FALSE

运行请求未生效

关联

- <Axis>.basic.out.traversKeyMinus
  在 JOG 运行方式下沿负方向移动机床轴 (MD32020 $MA_JOG_VELO)
- <Axis>.basic.out.traversKeyPlus
  在 JOG 运行方式下沿正方向移动机床轴 (MD32020 $MA_JOG_VELO)
- <Axis>.basic.in.traversMinusCommandActive
  负方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)
- <Axis>.basic.in.traversPlusCommandActive
  正方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.3.57 velocityReduced

NC → PLC	<Axis>.basic.in.velocityReduced
轴的当前运行速度因“防撞保护”而降低	

数据类型与更新

BOOL，循环

值 TRUE

当前运行速度因防撞保护而降低。

值 FALSE

当前运行速度不因防撞保护而降低。

更多信息

- SINUMERIK ONE 监控与补偿功能手册：“防撞保护，内部”章节

12.3.4 <Axis>.basic.out

12.3.4.1 acceptExternalWorkOffset

PLC → NC	<Axis>.basic.out.acceptExternalWorkOffset
已请求接受用于外部零点偏移的值 (MD28082 $MC_MM_SYSTEM_FRAME_MASK)	

说明

借助此接口信号，将系统变量 $AA_ETRANS 中的值作为外部零点偏移接收至外部零点偏移的框架（$P_EXTFRAME、$P_EXTFR）：

<Axis>.basic.out.acceptExternalWorkOffset:0 → 1 ⇒ $P_EXTFRAME[<轴>] = $P_EXTFR[<轴>] = $AA_ETRANS[<轴>]

数据类型与更新

BOOL，循环

值 0 → 1

已请求接收用于外部零点偏移的值。

值 1 → 0

未请求接收用于外部零点偏移的值。

关联

- $AA_ETRANS[<Achse>]
  外部零点偏移
- $P_EXTFRAME[<Achse>]
  外部框架的活动系统框架
- $P_EXTFR[<Achse>]
  用于外部框架的数据管理框架
- MD28082 $MC_MM_SYSTEM_FRAME_MASK,Bit1
  系统框架

更多信息

- SINUMERIK ONE 基础功能手册；“轴、坐标系、框架”章节

12.3.4.2 clampingInProgress

PLC → NC	<Axis>.basic.out.clampingInProgress
轴夹紧运行，激活监控	

数据类型与更新

BOOL，循环

值 TRUE

正在夹紧。

夹紧监控生效。

值 FALSE

夹紧过程结束。

静态监控生效。

关联

- MD36050 $MA_CLAMP_POS_TOL
  夹紧公差

12.3.4.3 contTravers

PLC → NC	<Axis>.basic.out.contTravers
激活持续手动运行（默认）	

参见 inc1 (页 521)。

12.3.4.4 disable

PLC → NC	<Axis>.basic.out.disable
请求进给轴/主轴锁定	

数据类型和更新

BOOL，循环

值 TRUE

已请求进给轴/主轴锁定。

轴

轴静止时的影响：

- 系统不向位置控制器输出设定值，即轴运行锁定。但是轴继续处于位置闭环控制下，且剩余的跟随误差会被修正。
- 若通过 NC 程序或手动运行轴，操作界面上会将设定位置和设定速度作为实际位置和实际速度显示。通道复位或程序结束 (M30 / M2) 时，实际位置的显示会被设置为机床轴的实际值。
- 运行指令会被输出至 NC/PLC 接口。

轴运行时的影响：

- 轴遵循当前生效的制动特性曲线减速停止，并显示报警。轴的运行随后被禁止。位置闭环控制保持生效，轴停止在当前位置。

主轴

控制系统运行且主轴静止时的影响：

- 系统不会向转速控制器输出设定值，即主轴的运行被禁止。
- 若通过 NC 程序或手动运行主轴，操作界面上会将转速设定值作为转速实际值显示。

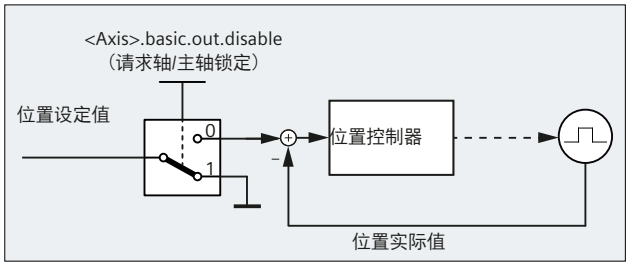
控制系统运行且主轴旋转时的影响：

- 系统立即向转速控制器输出转速设定值零。
- 主轴遵循当前生效的制动特性曲线减速停止，并显示报警。主轴的运行随后被禁止。位置闭环控制保持生效，轴停止在当前位置。

定位运行时的影响：

- 参见“静止的轴/运行的轴”，轴/主轴锁定结构图

轴/主轴锁定结构图



提示

- 如果在存在以下其中一个接口信号的情况下对运行的进给轴/主轴设置了信号“进给轴/主轴禁止”，进给轴/主轴**不**会停止：
  - <Chan>.geoAxis*.out.disableFeed
    几何轴 1、2、3 进给锁定
  - <Chan>.orientationAxis*.out.disableFeed
    定向轴 1、2、3 进给锁定
  - <Axis>.basic.out.feedStopSpindleStop
    进给停止/主轴停止处于活动状态
- 轴/主轴可继续通过 <Axis>.basic.out.followUpMode（跟踪运行）切换至“停止”或“跟踪”状态。
- 轴无法以通道专用方式通过 <Chan>.basic.out.ref 从 NC 程序 (G74) 回参考点，也无法以轴专用方式（机床功能 REF）回参考点。

- 如果对运行的轴只是暂时设置了信号“进给轴/主轴禁止”，轴会停止且不输出报警。收到下一个运行请求后，轴会运行到新位置。
  示例：
  ```
  N10 G0 X0 Y0
  N20 G1 F1000 X100
  N30 Y100
  N40 X200
  ```
  对于 N20：在 20 mm 位置为 X 轴短暂置位“进给轴/主轴禁止” ⇒ X 轴停止
  对于 N40：X 轴从上一位置（20 mm + 制动行程）运行至 200 mm 位置。
- 主轴耦合或进给轴耦合中“进给轴/主轴禁止”的影响：

<Axis>.coupling.in.leadingSpindleActive [1)]	**<Axis>.coupling.in.followingSpindleActive [2)]**	**耦合 [3)]**	**影响**
0	0	关闭	输出设定值
0	1	关闭	不为跟随主轴/跟随轴输出设定值
1	0	关闭	不为引导主轴/引导轴输出设定值
1	1	关闭	不为引导和跟随主轴/引导和跟随轴输出设定值
0	0	打开	输出设定值
0	1	打开	进给轴/主轴禁止对跟随主轴/跟随轴无效
1	0	打开	进给轴/主轴禁止也对跟随主轴/跟随轴有效
1	1	打开	不输出设定值

1) 主动主轴/主动轴处于活动状态

2) 从动主轴/从动轴处于活动状态

3) 主动轴和从动主轴/从动轴之间的连接状态

值 FALSE

不存在对“进给轴/主轴禁止”的请求。

说明

在进给轴/主轴停止时，该信号的复位才生效。

- **轴**
  复位信号后会立即重新向位置控制器输出设定值。若存在运行请求，则系统会立即执行此请求。
- **主轴**
  信号复位后，必须通过轴专用复位、通道复位或程序结束 (M30 / M2) 取消内部仍然生效的主轴锁定。之后会再次向转速控制器输出设定值。

关联

- <Axis>.basic.out.progTestRequest
  请求激活“程序测试”功能（转速设定值未生效）
- <Axis>.basic.in.disabled
  轴锁定，无法运行
- <Chan>.basic.out.progTest
  请求激活程序测试 (PRT)
- <Chan>.basic.in.progTestSelected
  已选程序测试 (PRT)

更多信息

- 同步运行中的行为：
  SINUMERIK ONE 轴和主轴功能手册；“同步主轴”章节

12.3.4.5 disableParameterSetSwitchover

PLC → NC	<Axis>.basic.out.disableParameterSetSwitchover
锁定参数组切换	

数据类型和更新

BOOL，任务控制

值 TRUE

参数组切换被禁用。

值 FALSE

参数组切换被使能。

关联

- <Axis>.basic.out.posCtrlParameterSetSelection
  为位置控制器选择参数组 (1 - 8)

12.3.4.6 disableTraversKeys

PLC → NC	<Axis>.basic.out.disableTraversKeys
锁定“正”、“负”移动键	

说明

在运行期间若激活移动键锁定，则轴静止。

数据类型与更新

BOOL，循环

值 TRUE

移动键“＋”和“-”被禁用。

值 FALSE

移动键“＋”和“-”被使能。

关联

- <Axis>.basic.out.traversKeyMinus
  在 JOG 运行方式下沿负方向移动机床轴 (MD32020 $MA_JOG_VELO)
- <Axis>.basic.out.traversKeyPlus
  在 JOG 运行方式下沿正方向移动机床轴 (MD32020 $MA_JOG_VELO)

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.4.7 dynBacklashComp

PLC → NC	<Axis>.basic.out.dynBacklashComp
激活动态间隙补偿	

数据类型与更新

BOOL

值 TRUE

激活动态间隙补偿

值 FALSE

不激活

12.3.4.8 enableCtrl

PLC → NC	<Axis>.basic.out.enableCtrl
请求控制器使能	

说明

对轴的机械夹紧

若轴在夹紧位置上定位，则夹紧会闭合。之后伺服使能会被复位。否则，若在夹紧过程中以机械方式将轴移离了设定位置，那么位置控制器会始终相对夹紧运作。

取消夹紧时，伺服使能首先会被复位，之后机械夹紧会松开。

说明

- 无伺服使能时对轴/主轴的运行请求：
  - 轴/主轴不会被运行
  - 运行指令会输出至接口
  - 运行请求存在的情形下，设置伺服使能时轴/主轴会立即运行。
- 针对运行中的几何轴的伺服使能复位总是会导致轮廓被破坏。

数据类型与更新

BOOL，循环

值 TRUE

进行伺服使能。

位置闭环构成，且轴/主轴处于闭环控制下。

反馈：
<Axis>.basic.in.posCtrlActive = 1（位置控制器处于活动状态）

若轴/主轴是在接口信号复位前进行的回参考点，那么在该接口信号重新置位后必须对轴/主轴重新执行回参考点。前提条件：在此期间不得超出生效测量系统的极限频率。

说明

如果在控制器使能未置位时将轴/主轴移离位置，则控制器使能置位时的特性取决于“跟踪运行”接口信号：

- <Axis>.basic.out.followUpMode == 1（激活跟踪运行）
  位置闭环控制在当前位置生效
- <Axis>.basic.out.followUpMode == 0（激活跟踪运行）
  位置闭环控制在控制器使能复位前的最后一个位置生效

值 FALSE

未进行伺服使能。

取消“伺服使能”时的特性取决于轴/主轴在该时间点为停止还是运行状态：

- 进给轴/主轴停止：
  - 轴的位置闭环控制回路断开。
  - <Axis>.basic.out.followUpMode == 1（激活跟踪运行） ⇒ 位置设定值 = 位置实际值
  - 驱动上的伺服使能会被复位
  - 以下接口信号会被复位：
    <Axis>.basic.in.posCtrlActive = 0（位置控制器处于活动状态）
    <Axis>.basic.in.speedCtrlActive = 0（转速控制器处于活动状态）
    <Axis>.drive.in.currentCtrlActive = 0（电流控制器处于活动状态）
- 轴/主轴运行
  - 轴会通过快速停止制动，直至停止。
  - 报警 21612“伺服使能 VDI 信号在运行期间复位”
  - 进给轴/主轴的位置控制回路断开。
  - 不受接口信号 <Axis>.basic.out.followUpMode（激活跟踪运行）影响，制动进程结束时系统对位置设定值进行跟踪（位置设定值 = 位置实际值），并设置反馈信号 <Axis>.basic.in.followUpModeActive = 1（跟踪运行处于活动状态）。
  - 以下接口信号会被复位：
    <Axis>.basic.in.posCtrlActive
    <Axis>.basic.in.speedCtrlActive
    <Axis>.drive.in.currentCtrlActive

说明

如果复位了“控制器使能”且未设置“跟踪运行”(<Axis>.basic.out.followUpMode == 0)，控制器内部会保留当前额定位置。如果在该状态下手动运行轴，额定位置与实际位置之间的偏差（跟随误差）会越来越大。设置“控制器使能”后，跟随误差会瞬间降至零（转速设定值跳动）。

关联

- <Axis>.basic.out.followUpMode
  激活跟踪运行
- <Axis>.basic.in.followUpModeActive
  跟踪运行处于活动状态
- <Axis>.basic.in.posCtrlActive
  位置控制器处于活动状态
- <Axis>.basic.in.speedCtrlActive
  转速控制器处于活动状态
- <Axis>.drive.in.currentCtrlActive
  电流控制器处于活动状态
- MD36620 $MA_SERVO_DISABLE_DELAY_TIME
  控制器使能关闭延迟
- MD36610 $MA_AX_EMERGENCY_STOP_TIME
  故障状态下制动斜坡的持续时间

12.3.4.9 enableModuloLimitSwitch

PLC → NC	<Axis>.basic.out.enableModuloLimitSwitch
激活模数回转轴的行进范围限制（软件限位开关、工作区域限制等）	

说明

在直线轴/回转轴无模数功能的情况下，此信号不相关。

应用示例

周边磨砂轮，带有监控

数据类型与更新

BOOL，循环

值 TRUE

针对模数回转轴激活运行范围限制（软件限位开关、工作区域限制）。

值 FALSE

针对模数回转轴取消运行范围限制。

关联

- <Axis>.basic.in.moduloLimitSwitchActive
  模数回转轴的行进范围限制处于活动状态（软件限位开关、工作区域限制）

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“回转轴”章节

12.3.4.10 enableOvr

PLC → NC	<Axis>.basic.out.enableOvr
激活进给/主轴倍率	

说明

借助此接口信号激活通过 NC/PLC 接口设定的进给率修调系数 <Axis>.basic.out.feedrateOvrFactor。

若进给/主轴倍率未生效，则控制系统内部以独立于当前开关位置的方式将值 1.0 ≙ 100 % 用作补偿值。特例是第一开关位置。就第一开关位置而言，根据所选择的编码，以下系数生效：

- 二进制编码：补偿值 = 0
- 格雷码编码：校正值 = MD12030 $MN_OVR_FACTOR_FEEDRATE[0]

数据类型与更新

BOOL，循环

值 TRUE

进给/主轴倍率生效。

值 FALSE

进给/主轴倍率未生效。

关联

- MD12000 $MN_OVR_AX_IS_GRAY_CODE（轴专用校正开关，格雷码编码）
- MD12030 $MN_OVR_FACTOR_FEEDRATE（轨迹进给校正开关评估）
- MD12100 $MN_OVR_FACTOR_LIMIT_BIN（二进制编码校正开关极限值）

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“进给”>“进给控制”>“通过机床控制面板设置进给倍率”章节

12.3.4.11 enc1Activation

PLC → NC	<Axis>.basic.out.enc1Activation
激活位置测量系统 1	

表格主要介绍两种位置测量系统的行为：

enc2Activation	enc1Activation	作用
FALSE	TRUE	位置测量系统 1 生效： • 通过位置测量系统 1 实现对机床轴的位置闭环控制。 • 通过位置测量系统 1 监控机床轴（测量系统监控、静态监控、夹紧监控、轮廓偏差等）。 • 若存在位置测量系统 2（MD30200 $MA_NUM_ENCS == 2），采集其的位置实际值，但不执行相关监控。
TRUE	FALSE	位置测量系统 2 生效： • 通过位置测量系统 2 实现对机床轴的位置闭环控制。 • 通过位置测量系统 2 监控机床轴（测量系统监控、静态监控、夹紧监控、轮廓偏差等）。 • 若存在位置测量系统 1（MD30200 $MA_NUM_ENCS == 2），采集其的位置实际值，但不执行相关监控。

enc2Activation	enc1Activation	作用
TRUE	TRUE	• 通过位置测量系统 1 实现对机床轴的位置闭环控制。 • 若存在位置测量系统 2（MD30200 $MA_NUM_ENCS == 2），其位置实际值同样会被采集。
FALSE	FALSE	位置测量系统 1 和 2 无效（机床轴“驻留”）： • 系统不会采集实际值。 • 位置测量系统的监控被取消 • 以下接口信号会被复位： – <Axis>.basic.in.enc1Synchronized / <Axis>.basic.in.enc2Synchronized == 0（已回参考点/已同步，编码器 1/2） – <Axis>.basic.in.posCtrlActive（位置控制器处于活动状态） – <Axis>.basic.in.speedCtrlActive（转速控制器处于活动状态） – <Axis>.drive.in.currentCtrlActive（电流控制器处于活动状态）

说明

- 若在运行的轴上为生效的位置测量系统复位了该接口信号，那么该轴会通过斜坡停止制动至静止状态，控制系统内部不会取消伺服使能。
- 如果采用转速闭环控制的主轴无位置测量系统，则“控制器使能”接口信号必须置位：<Axis>.basic.out.enableCtrl == 1（控制器使能）
- 取消“驻留”状态后，必须为增量位置测量系统重新执行回参考点，以达到“已回参考点”状态。
- 如果“驻留”状态活跃，则 NC 启动时会为该轴忽略以下接口信号：<Axis>.basic.in.enc1Synchronized / <Axis>.basic.in.enc2Synchronized（已回参考点/已同步 1/2）。

应用示例

1. 从位置测量系统 1 切换至位置测量系统 2（及反之）：
   如果轴在两个位置测量系统中均执行了回参考点，且在此期间未超过所使用测量编码器的极限频率，即
   <Axis>.basic.in.enc1Synchronized 和 <Axis>.basic.in.enc2Synchronized == 1（已回参考点/已同步 1/2），则切换后无需重新执行回参考点。
   切换时，位置测量系统 1 和 2 之间的当前偏差会立即运行。
   通过以下机床数据可设定一个公差带，用于规定切换时两个实际值之间所允许的偏差：
   MD36500 $MA_ENC_CHANGE_TOL（位置实际值切换时的最大公差）
   若实际值差值大于公差，则不会进行切换，并显示报警 25100“无法进行测量系统切换”。
2. 机床轴驻留：
   为了移除测量编码器，位置测量系统的监控会被取消。

3. 关闭位置测量系统：
   位置测量系统 1 或 2 关闭时，其对应的接口信号会被复位：
   <Axis>.basic.in.enc1Synchronized / <Axis>.basic.in.enc2Synchronized（已回参考点/已同步 1/2）。
4. 回参考点运行：
   轴的回参考点运行由所选的位置测量系统执行。每个位置测量系统必须单独回参考点。

数据类型与更新

BOOL，循环

值 TRUE

位置测量系统已生效。

值 FALSE

位置测量系统未激活。

关联

- <Axis>.basic.out.enableCtrl
  请求控制器使能
- <Axis>.basic.in.enc1Synchronized
  机床轴位置测量系统 1 已回参考点/同步
- <Axis>.basic.in.enc2Synchronized
  机床轴位置测量系统 2 已回参考点/同步
- <Axis>.basic.in.speedCtrlActive
  转速控制器处于活动状态
- MD36500 $MA_ENC_CHANGE_TOL（位置实际值切换时的最大公差）
- MD30200 $MA_NUM_ENCS（编码器数量）

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“速度、设定值/额定值系统、闭环控制”章节

12.3.4.12 enc2Activation

PLC → NC	<Axis>.basic.out.enc2Activation
激活位置测量系统 2	

表格主要介绍两种位置测量系统的行为：

enc2Activation	enc1Activation	作用
FALSE	TRUE	位置测量系统 1 生效： • 通过位置测量系统 1 实现对机床轴的位置闭环控制。 • 通过位置测量系统 1 监控机床轴（测量系统监控、静态监控、夹紧监控、轮廓偏差等）。 • 若存在位置测量系统 2（MD30200 $MA_NUM_ENCS == 2），采集其的位置实际值，但不执行相关监控。
TRUE	FALSE	位置测量系统 2 生效： • 通过位置测量系统 2 实现对机床轴的位置闭环控制。 • 通过位置测量系统 2 监控机床轴（测量系统监控、静态监控、夹紧监控、轮廓偏差等）。 • 若存在位置测量系统 1（MD30200 $MA_NUM_ENCS == 2），采集其的位置实际值，但不执行相关监控。
TRUE	TRUE	• 通过位置测量系统 1 实现对机床轴的位置闭环控制。 • 若存在位置测量系统 2（MD30200 $MA_NUM_ENCS == 2），其位置实际值同样会被采集。
FALSE	FALSE	位置测量系统 1 和 2 无效（机床轴“驻留”）： • 系统不会采集实际值。 • 位置测量系统的监控被取消 • 以下接口信号会被复位： – <Axis>.basic.in.enc1Synchronized / <Axis>.basic.in.enc2Synchronized == 0（已回参考点/已同步，编码器 1/2） – <Axis>.basic.in.posCtrlActive（位置控制器处于活动状态） – <Axis>.basic.in.speedCtrlActive（转速控制器处于活动状态） – <Axis>.drive.in.currentCtrlActive（电流控制器处于活动状态）

说明

- 若在运行的轴上为生效的位置测量系统复位了该接口信号，那么该轴会通过斜坡停止制动至静止状态，控制系统内部不会取消伺服使能。
- 如果采用转速闭环控制的主轴无位置测量系统，则“控制器使能”接口信号必须置位：<Axis>.basic.out.enableCtrl == 1（控制器使能）
- 取消“驻留”状态后，必须为增量位置测量系统重新执行回参考点，以达到“已回参考点”状态。
- 如果“驻留”状态活跃，则 NC 启动时会为该轴忽略以下接口信号：<Axis>.basic.in.enc1Synchronized / <Axis>.basic.in.enc2Synchronized（已回参考点/已同步 1/2）。

应用示例

1. 从位置测量系统 1 切换至位置测量系统 2（及反之）：
   如果轴在两个位置测量系统中均执行了回参考点，且在此期间未超过所使用测量编码器的极限频率，即
   <Axis>.basic.in.enc1Synchronized 和 <Axis>.basic.in.enc2Synchronized == 1（已回参考点/已同步 1/2），则切换后无需重新执行回参考点。
   切换时，位置测量系统 1 和 2 之间的当前偏差会立即运行。
   通过以下机床数据可设定一个公差带，用于规定切换时两个实际值之间所允许的偏差：
   MD36500 $MA_ENC_CHANGE_TOL（位置实际值切换时的最大公差）
   若实际值差值大于公差，则不会进行切换，并显示报警 25100“无法进行测量系统切换”。
2. 机床轴驻留：
   为了移除测量编码器，位置测量系统的监控会被取消。
3. 关闭位置测量系统：
   位置测量系统 1 或 2 关闭时，其对应的接口信号会被复位：
   <Axis>.basic.in.enc1Synchronized / <Axis>.basic.in.enc2Synchronized（已回参考点/已同步 1/2）。
4. 回参考点运行：
   轴的回参考点运行由所选的位置测量系统执行。每个位置测量系统必须单独回参考点。

数据类型与更新

BOOL，循环

值 TRUE

位置测量系统已生效。

值 FALSE

位置测量系统未激活。

关联

- <Axis>.basic.out.enableCtrlReglerfreigabe
  请求控制器使能
- <Axis>.basic.in.enc1Synchronized
  机床轴位置测量系统 1 已回参考点/同步
- <Axis>.basic.in.enc2Synchronized
  机床轴位置测量系统 2 已回参考点/同步
- <Axis>.basic.in.speedCtrlActive
  转速控制器处于活动状态
- MD36500 $MA_ENC_CHANGE_TOL（位置实际值切换时的最大公差）
- MD30200 $MA_NUM_ENCS（编码器数量）

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“速度、设定值/额定值系统、闭环控制”章节

12.3.4.13 feedrateOvrFactor

PLC → NC	<Axis>.basic.out.feedrateOvrFactor
进给倍率的标准化因数可以是二进制数或格雷码 (MD12000 $MN_OVR_AX_IS_GRAY_CODE)	

说明

二进制/格雷码编码

倍率系数可以二进制码或格雷码设定。使用的格式必须通过以下机床数据告知控制系统：

MD12000 $MN_OVR_AX_IS_GRAY_CODE = <编码方式>

数据类型和更新

USInt，循环

值

二进制编码 **采用二进制码时，接口中的值对应倍率系数。**		
二进制编码	**十进制**	**倍率系数**
0000 0000	0	0.00
0000 0001	1	0.01

二进制编码
采用二进制码时，接口中的值对应倍率系数。

二进制编码	十进制	倍率系数
0000 0010	2	0.02
0000 0011	3	0.03
000 0100	4	0.04
…	…	…
0110 0100	100	1.00
…	…	…
1100 1000	200	2.00

格雷码编码
以下机床数据可以指定接口格雷码形式的数值对应的倍率系数：
MD12010 $MN_OVR_FACTOR_AX_SPEED[<开关位置> - 1] = <校正系数>

二进制编码	格雷码	倍率系数 [1)]
1	00001	0.00
2	00011	0.01
3	00010	0.02
4	00110	0.04
5	00111	0.06
6	00101	0.08
7	00100	0.10
8	01100	0.20
9	01101	0.30
10	01111	0.40
11	01110	0.50
12	01010	0.60
13	01011	0.70
14	01001	0.75
15	01000	0.80
16	11000	0.85
17	11001	0.90

格雷码编码
以下机床数据可以指定接口格雷码形式的数值对应的倍率系数：
MD12010 $MN_OVR_FACTOR_AX_SPEED[<开关位置> - 1] = <校正系数>

二进制编码	格雷码	倍率系数 [1)]
18	11011	0.95
19	11010	1.00
20	11110	1.05
21	11111	1.10
22	11101	1.15
23	11100	1.20
24	10100	1.20
25	10101	1.20
26	10111	1.20
27	10110	1.20
28	10010	1.20
29	10011	1.20
30	10001	1.20
31	10000	1.20

限制

- 控制系统内部将生效的倍率系数限制在 2.00 或 200% 内。
- 可通过以下机床数据将最大倍率系数限制为一个小于 200% 的值：
  MD12100 $MN_OVR_FACTOR_LIMIT_BIN

说明

在通过以下功能进行螺纹切削期间，轴专用的进给倍率**不生效**：G33、G331、G332、G63

关联

- <Chan>.basic.out.pathFeedrateOvrFactor
  可以通过二进制或格雷码编码形式规定轨迹进给倍率系数 (MD12020 $MN_OVR_FEED_IS_GRAY_CODE)
- <Axis>.spindle.out.ovrFactor
  控制主轴转速的倍率系数通过 MCP 主轴倍率旋钮等确定
- <Axis>.basic.out.enableOvr
  激活进给/主轴倍率

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“进给”>“进给控制”>“通过机床控制面板设置进给倍率”章节

12.3.4.14 feedStopSpindleStop

PLC → NC	进给停止/主轴停止处于活动状态
进给停止/主轴停止处于活动状态	

说明

轴

- 若在轴运行期间将此接口信号置位，则轴通过其制动特性曲线制动至静止状态。如该轴与其它轴具有插补关联，那么这些轴也会制动。
- 在此接口信号复位后，系统会继续执行停止的运行。
- 位置闭环控制保持生效，跟随误差会减小。
- 若此接口置位，并且请求运行，则轴不运行。但运行请求仍然保留。在此接口信号复位后，系统会继续执行运行请求，即运行轴。
- 该接口信号在所有运行方式下生效。

螺纹加工	生效方式
G33、G34、G35	生效（产生轮廓偏差）
G331、G332	生效
G63	生效

主轴

- 若在主轴运行期间将此接口信号置位，则主轴通过其制动特性曲线制动至静止状态。如该主轴与其它轴/主轴具有插补关联，那么这些轴也会制动。
- 在此接口信号复位后，系统会继续执行停止的运行。
- 在定位模式期间保留位置闭环控制，跟随误差会减小。
- 若此接口置位，并且在定位模式期间请求运行，则主轴不运行。但运行请求仍然保留。在此接口信号复位后，系统会继续执行运行请求，即运行主轴。
- 该接口信号在所有运行方式下生效。
- 在攻丝（G331、G332）期间，此接口信号无效。

螺纹加工	生效方式
G33、G34、G35	生效（根据参数设置的动态特性，可能产生轮廓偏差）
G331、G332	不生效
G63	生效

说明

“进给轴/主轴锁定”处于活动状态时 (<Axis>.basic.out.disable == 1)，此接口信号不生效。

数据类型和更新

BOOL，循环

值 TRUE

进给/主轴停止生效。

值 FALSE

进给/主轴停止未生效。

关联

- <Axis>.basic.out.disable
  请求轴/主轴锁定

更多信息

- SINUMERIK ONE 轴和主轴功能手册； “进给”>“进给控制”>“进给锁定和进给/主轴停止”章节

12.3.4.15 fixedFeedrateSelection

PLC → NC	<Axis>.basic.out.fixedFeedrateSelection
JOG 运行方式下的固定进给率选择 (1..4，0 代表不选择）(MD12202 $MN_PERMANENT_FEED / MD12204 $MN_PERMANENT_ROT_AX_FEED)	

说明

固定进给率 1 ..4 激活，加工轴

借助这些接口信号在 JOG 运行方式下激活通过机床数据设置的固定进给率。

以选择的固定进给率运行轴，而非编程设置的 JOG 速度/JOG 快速运行速度 (MD... ... JOG_VELO_...)。运行方向通过接口信号设定：

<Axis>.basic.out.traversKeyMinus, <Axis>.basic.out.traversKeyPlus（正/负移动键）

通过以下机床数据对固定进给率进行参数设置：

- 直线轴：MD12202 $MN_PERMANENT_FEED
- 回转轴：MD12204 $MN_PERMANENT_ROT_AX_FEED

说明

- 在以下情形下，无法以固定进给率运行：
  - 主轴
  - 定位轴
  - 攻丝
- 固定进给率始终被视作线性进给。即便在旋转进给率生效的情况下，控制系统内部也会转换为线性进给。

数据类型和更新

BOOL，循环

值

0：固定进给率未选择

1 - 4：已选固定进给率 1 至 4

关联

- MD12200 $MN_RUN_OVERRIDE_0（倍率 0 时的运行特性）
- MD12202 $MN_PERMANENT_FEED（用于线性轴的固定进给率）
- MD12204 $MN_PERMANENT_ROT_AX_FEED（用于回转轴的固定进给率）
- MD21150 $MC_JOG_VELO_RAPID_ORI（定向轴的 JOG 快速移动速度）
- MD21155 $MC_JOG_VELO_ORI（定向轴的基准 JOG 速度）
- MD21160 $MC_JOG_VELO_RAPID_GEO（几何轴的 JOG 快速移动速度）
- MD21165 $MC_JOG_VELO_GEO（几何轴的基准 JOG 速度）
- MD32010 $MA_JOG_VELO_RAPID（基准 JOG 快进速度）
- MD32020 $MA_JOG_VELO（基准 JOG 轴速度）

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“进给” > “进给控制” > “固定进给值”章节

12.3.4.16 fixedStopAckReached

PLC → NC	<Axis>.basic.out.fixedStopAckReached
应答：“到达固定挡块”，继续处理程序	

说明

在 MD37060 $MA_FIXED_STOP_ACKN_MASK == 0 或 2 时不相关

数据类型与更新

BOOL，循环

值 TRUE

到达固定点停止后的含义:<Axis>.basic.in.fixedStopReached（到达固定挡块）== 1

- 轴以夹紧扭矩抵住固定挡块
- 固定挡块监控窗口激活。
- 执行了程序段转换。

值 FALSE

到达固定点停止后的含义:<Axis>.basic.in.fixedStopReached（到达固定挡块）== 1

- 轴以夹紧扭矩抵住固定挡块
- 固定挡块监控窗口生效
- 不执行程序段切换，并显示通道消息“等待：帮助功能应答缺失”。
- 功能终止，显示报警“20094 轴 %1 功能已终止”

在零件程序中通过 `FXS=0` 取消功能后的含义：

- 扭矩极限和固定挡块监控窗口都将被取消。

关联

- <Axis>.basic.out.fixedStopSensor
  到达固定挡块 (MD37060 $MA_FIXED_STOP_ACKN_MASK)
- <Axis>.basic.out.fixedStopEnable
  确认固定挡块激活请求
- <Axis>.basic.in.fixedStopRequest
  请求使能“运行到固定挡块”功能

- <Axis>.basic.in.fixedStopReached
  到达固定挡块
- MD37060 $MA_FIXED_STOP_ACKN_MASK
  考虑通过 PLC 应答“运行到固定挡块”

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“运行到固定挡块”章节

12.3.4.17 fixedStopEnable

PLC → NC	<Axis>.basic.out.fixedStopEnable
确认固定挡块激活请求	

说明

在 MD37060 $MA_FIXED_STOP_ACKN_MASK == 0 或 2 时不相关

数据类型与更新

BOOL，循环

值 TRUE

对通过 <Axis>.basic.in.fixedStopRequest = 1 使能运行到固定挡块的请求的**正反馈消息**。

值 FALSE

对通过 <Axis>.basic.in.fixedStopRequest = 1 使能运行到固定挡块的请求的**负反馈消息**。

在 NC 程序中激活“运行到固定挡块”功能（FXS）**后**的含义：

- 运行到固定挡块功能被禁用。
- 轴以降低的转矩停止在起始位置。
- 显示通道消息“等待：帮助功能应答缺失"。

到达固定挡块**前**的含义 <Axis>.basic.in.fixedStopReached == 0（到达固定挡块）：

- 运行到固定挡块被终止。
- 显示报警“20094：轴 %1 功能已终止”。

到达固定挡块**后**的含义 <Axis>.basic.in.fixedStopReached == 1（到达固定挡块）：

- 扭矩极限值和固定点停止监控窗口都将被取消。

关联

- <Axis>.basic.out.fixedStopAckReached
  应答：“到达固定挡块”，继续处理程序
- <Axis>.basic.out.fixedStopSensor
  到达固定挡块 (MD37060 $MA_FIXED_STOP_ACKN_MASK)
- <Axis>.basic.out.disable
  请求轴/主轴锁定
- <Axis>.basic.in.fixedStopRequest
  请求使能“运行到固定挡块”功能
- <Axis>.basic.in.fixedStopReached
  到达固定挡块
- MD37060 $MA_FIXED_STOP_ACKN_MASK
  考虑通过 PLC 应答“运行到固定挡块”

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“运行到固定挡块”章节

12.3.4.18 fixedStopSensor

PLC → NC	<Axis>.basic.out.fixedStopSensor
到达固定挡块 (MD37060 $MA_FIXED_STOP_ACKN_MASK)	

说明

仅在以下情形下，信号才生效：MD37040 $MA_FIXED_STOP_BY_SENSOR == 1

数据类型与更新

BOOL，循环

值 TRUE

已到达固定挡块。

值 FALSE

未到达固定挡块。

关联

- <Axis>.basic.out.fixedStopAckReached
  应答：“到达固定挡块”，继续处理程序
- <Axis>.basic.out.fixedStopEnable
  确认固定挡块激活请求
- <Axis>.basic.in.fixedStopRequest
  请求使能“运行到固定挡块”功能
- <Axis>.basic.in.fixedStopReached
  到达固定挡块
- MD37040 $MA_FIXED_STOP_BY_SENSOR
  通过传感器识别固定挡块

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“运行到固定挡块”章节

12.3.4.19 followUpMode

PLC → NC	<Axis>.basic.out.followUpMode
激活跟踪运行	

说明

如果因故障导致控制器内部取消控制器使能，那么 NC 启动前必须在成功删除存在的报警并由控制器随之设置控制器使能后为轴激活“停止”状态：<Axis>.basic.out.followUpMode = FALSE（激活跟踪运行）。

否则，在 NC 启动且跟踪运行生效时，内部的剩余行程删除将使得无法执行之前的 NC 程序段的运行。

注意
过早结束运行程序段 在从“跟踪”状态切换至“停止”状态，或者在位置闭环控制中进行伺服使能时，控制系统内部会激活剩余行程删除。这会导致只运行该轴的运行程序段直接停止。

数据类型与更新

BOOL，循环

值 TRUE

跟踪运行生效：

- 持续跟踪位置设定值：位置设定值=位置实际值
- 反馈信号：<Axis>.basic.in.followUpModeActive = TRUE（跟踪运行处于活动状态）
- 静态监控或夹紧监控未生效。
- 重新激活闭环控制时，若有一个 NC 程序生效，那么控制系统内部会重新定位（REPOSA：通过所有轴沿一条直线定位）至最后编写的位置。

说明

只有同时复位控制器使能时，跟踪运行才会生效：<Axis>.basic.out.enableCtrl = FALSE

值 FALSE

跟踪运行未生效：

- 不跟踪位置设定值：位置设定值=编程的设定位置
- 反馈信号：<Axis>.basic.in.followUpModeActive = FALSE（跟踪运行未激活）
- 静态监控或夹紧监控生效。

说明

复位控制器使能后 (<Axis>.basic.out.enableCtrl == FALSE)，轴处于“停止”状态。此时，轴的设定位置不会跟踪实际位置。如果在该状态下手动运行轴，额定位置与实际位置之间的偏差（跟随误差）会越来越大。设置“控制器使能”后，跟随误差会瞬间降至零（转速设定值跳动）。

关联

- <Axis>.basic.out.enableCtrl
  请求控制器使能
- <Axis>.basic.out.clampingInProgress
  轴夹紧运行，激活监控
- <Axis>.basic.in.followUpModeActive
  跟踪运行处于活动状
  态

12.3.4.20 handwheelInvertDir

PLC → NC	<Axis>.basic.out.handwheelInvertDir
反转手轮旋转方向	

说明

只允许在机床轴处于静止时修改该取反信号。

应用示例

- 手轮旋转方向和预期的轴移动方向不一致。
- 手轮指定给了多根定向轴。

数据类型与更新

BOOL，循环

值 TRUE

已请求将与机床轴对应的手轮的旋转方向取反。

值 FALSE

未请求将与机床轴对应的手轮的旋转方向取反。

关联

- <Axis>.basic.in.handwheelDirInverted
  手轮旋转方向反转

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.4.21 handwheelSelection

PLC → NC	<Axis>.basic.out.handwheelSelection
手轮 1..6 已分配或 0 未分配手轮（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）	

说明

为轴所分配的手轮的编号。

该接口使用了二进制编码。通过机床数据 MD11324=1 进行定义。

数据类型和更新

USInt，循环

值

0：未分配手轮

1 - 6：手轮的编号

关联

- MD11324 $MN_HANDWH_VDI_REPRESENTATION
  VDI 接口中的手轮编号显示
- <Axis>.basic.in.handwheelAssignedNumber
  当前所分配手轮的编号：1..6 或 0 未分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.4.22 hwLimitSwitchMinus

PLC → NC	<Axis>.basic.out.hwLimitSwitchMinus
已操作负向硬件限位开关	

说明

显示报警 21614“硬件限位开关 -”，轴进入静止状态。

数据类型与更新

BOOL，循环

值 TRUE

已触发硬件限位开关 -。

值 FALSE

未触发硬件限位开关 -。

关联

- MD36600 $MA_BRAKE_MODE_CHOICE
  启用硬件限位开关时的制动行为

12.3.4.23 hwLimitSwitchPlus

PLC → NC	<Axis>.basic.out.hwLimitSwitchPlus
已操作正向硬件限位开关	

说明

显示报警 21614“硬件限位开关 +”，轴进入静止状态。

数据类型与更新

BOOL，循环

值 TRUE

已触发硬件限位开关 +。

值 FALSE

未触发硬件限位开关 +。

关联

- MD36600 $MA_BRAKE_MODE_CHOICE
  启用硬件限位开关时的制动行为

12.3.4.24 inc1

PLC → NC	<Axis>.basic.out.inc1
激活“根据请求将轴移动 1 个增量”	

说明

对于每个用于在 JOG 运行方式下手动运行机床轴的机床功能，均有一个请求信号。

增量手动运行 (<Axis>.basic.out.incVar)

除了五个固定步数（MD11330 中的缺省设置：INC1、INC10、INC100、INC1000 和 INC10000），还有可由设定数据 SD41010 设置的可变步数（INCvar）。固定步数和可变步数的每一步的长度通过轴专用机床数据 MD31090 设置。

通过按下移动键“+”或“-”，或通过旋转电子手轮，轴开始沿相应的方向移动与生效的机床功能对应的步数。

连续手动运行 (<Axis>.basic.out.contTravers)

连续手动运行时，借助移动键“+”或“-”使得轴沿相应的方向连续移动。

说明

- 若同时置位多个请求，将无机床功能生效。
- 如果在轴当前正在执行某机床功能时取消该功能或者切换到另一个功能，则几何轴终止移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求用于手动运行机床轴的机床功能。

值 FALSE

未请求用于手动运行机床轴的机床功能。

关联

- <Axis>.basic.in.inc1Active
  激活“根据请求将轴移动 1 个增量”
- <Axis>.basic.in.inc10Active
  激活“根据请求将轴移动 10 个增量”
- <Axis>.basic.in.inc100Active
  激活“根据请求将轴移动 100 个增量”
- <Axis>.basic.in.inc1000Active
  激活“根据请求将轴移动 1 000 个增量”
- <Axis>.basic.in.inc10000Active
  激活“根据请求将轴移动 10 000 个增量”
- <Axis>.basic.in.incVarActive
  “根据请求移动轴，移动幅度为用户自定义增量数”处于活动状态
- <Axis>.basic.in.contTraversActive
  连续运行处于活动状态
- MD11320 $MN_HANDWH_IMP_PER_LATCH
  手轮每个卡槽位置的脉冲数
- MD11330 $MN_JOG_INCR_SIZE_TAB
  INC/手轮运行时的增量大小
- MD31090 $MA_JOG_INCR_WEIGHT
  INC/手轮运行时的增量加权
  SD41010 $SN_JOG_VAR_INCR_SIZE
  JOG 模式下的可变增量大小

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.4.25 inc10

PLC → NC	<Axis>.basic.out.inc10
激活“根据请求将轴移动 10 个增量”	

参见 inc1 (页 521)。

12.3.4.26 inc100

PLC → NC	<Axis>.basic.out.inc100
激活“根据请求将轴移动 100 个增量”	

参见 inc1 (页 521)。

12.3.4.27 inc1000

PLC → NC	<Axis>.basic.out.inc1000
激活“根据请求将轴移动 1 000 个增量”	

参见 inc1 (页 521)。

12.3.4.28 inc10000

PLC → NC	<Axis>.basic.out.inc10000
激活“根据请求将轴移动 10 000 个增量”	

参见 inc1 (页 521)。

12.3.4.29 incVar

PLC → NC	<Axis>.basic.out.incVar
激活“根据请求移动轴，移动幅度为用户自定义增量数”	

参见 inc1 (页 521)。

12.3.4.30 jogFixedPointSelection

PLC → NC	<Axis>.basic.out.jogFixedPointSelection
选择点动定点 (1 - 4) (MD30600 $MA_FIX_POINT_POS)	

说明

选择“点动运行至定点”功能后，PLC 通过 <Axis>.basic.out.jogFixedPointSelection 将目标定点的编号输出至 NC。

功能激活后（见 <Axis>.basic.in.jogFixedPointSelectedNumber），可以操作移动键或手轮将选中的机床轴移动到指定的定点。定点通过机床数据 MD30600 定义。

数据类型和更新

USInt，循环

值

1..4：将运行到的固定点的编号

关联

- <Axis>.basic.in.jogFixedPointSelectedNumber
  接近点动定点 (1..4)
- <Axis>.basic.in.jogFixedPointReachedNumber
  通过“精准停”到达要接近的点动定点 (1..4)
- MD30600 $MA_FIX_POINT_POS[<n>]
  轴的固定值位置

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.4.31 jogToPos

PLC → NC	<Axis>.basic.out.jogToPos
已请求“点动运行至位置”功能 (SD43320 $SA_JOG_POSITION)	

说明

功能激活后（参见 <Axis>.basic.in.jogToPosActive），可以操作移动键或手轮将选择的机床轴移动至通过设定数据 SD43320 设定的位置。

应用

此功能适用于只能手动运行的机床，比如运行至钻孔位置。

说明

对于“手动运行至位置”功能而言，以下前提条件适用：

- 此功能只能在 JOG 运行方式下激活。
  机床功能 JOG-REPOS 或 JOG-REF 生效时以及在 AUTO 模式下的 JOG 中无法激活此功能。
- 待运行的轴必须已执行回参考点。
- 不允许激活运动转换。
- 待运行的轴不允许与从动轴之间存在有效耦合。

数据类型与更新

BOOL，循环

值 TRUE

已请求“手动运行至位置”功能。

值 FALSE

未请求“手动运行至位置”功能。

关联

- <Axis>.basic.in.jogToPosActive
  “点动运行至位置”功能处于活动状态
- <Axis>.basic.in.jogPosReached
  在“点动运行至位置”功能下，轴已到达通过 SD43320“精准停”设定的位置
- SD43320 $SA_JOG_POSITION
  点动位置

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.4.32 limitSpeed

PLC → NC	<Axis>.basic.out.limitSpeed
限制主轴速度/转速 (MD35160 $MA_SPIND_EXTERN_VELO_LIMIT)	

说明

控制系统将速度/主轴转速限制为参数设置的限值：

MD35160 $MA_SPIND_EXTERN_VELO_LIMIT

数据类型与更新

BOOL，循环

值 TRUE

速度/主轴转速限制生效。

值 FALSE

速度/主轴转速限制未生效。

关联

- MD35100 $MA_SPIND_VELO_LIMIT
  最大主轴转速
- MD35160 $MA_SPIND_EXTERN_VELO_LIMIT
  PLC 主轴转速限制
- SD43220 $SA_SPIND_MAX_VELO_G26
  G26 的可编程主轴转速上限
- SD43230 $SA_SPIND_MAX_VELO_LIMS
  G96/G961/G97 的主轴转速限制

更多信息

- SINUMERIK ONE 基础功能手册；“轴、坐标系、框架”章节

12.3.4.33 posCtrlParameterSetSelection

PLC → NC	<Axis>.basic.out.posCtrlParameterSetSelection
为位置控制器选择参数组 (1 - 8)	

说明

下列情况下忽略位置控制器参数组选择：
MD35590 $MA_PARAMSET_CHANGE_ENABLE == 0

数据类型和更新

USInt，任务控制

值

1 - 8：所选参数组的编号

关联

- <Axis>.drive.in.posCtrlParameterSetNumber
  位置控制器当前有效参数组的编号

12.3.4.34 progTestEnableCtrl

PLC → NC	<Axis>.basic.out.progTestEnableCtrl
程序测试：请求控制器使能	

数据类型与更新

BOOL，循环

12.3.4.35 progTestRequest

PLC → NC	<Axis>.basic.out.progTestRequest
请求激活“程序测试”功能（转速设定值未生效）	

说明

若“程序测试”功能生效，则轴在内部轴禁用下运行。系统会生成设定值，但其不输出至机床轴。实际值 = 设定值。

数据类型与更新

BOOL，循环

值 TRUE

已请求抑制程序测试（禁止设定值输出）。

值 FALSE

未请求抑制程序测试（禁止设定值输出）。

关联

- <Axis>.basic.out.progTestSuppression
  请求抑制“程序测试”功能（转速设定值生效）
- <HmiAxis>.basic.in.progTestSuppressionRequested
  HMI：已抑制程序测试
- <HmiAxis>.basic.in.progTestRequested
  HMI：已请求进行程序测试

更多信息

- SINUMERIK ONE 基础功能手册；“跨通道程序协调”>“逐通道试运行”章节

12.3.4.36 progTestSuppression

PLC → NC	<Axis>.basic.out.progTestSuppression
请求抑制“程序测试”功能（转速设定值生效）	

说明

在轴当前归属于的通道中，若“程序测试”（PRT）生效，则轴在内部轴禁用下运行。系统会生成设定值，但其不输出至机床轴。实际值 = 设定值。

若为轴抑制程序测试，则即便在“程序测试”功能在轴当前归属于的通道中生效的情况下，系统也将设定值输出至机床轴。

数据类型与更新

BOOL，循环

值 TRUE

已请求抑制程序测试（禁止设定值输出）。

值 FALSE

未请求抑制程序测试（禁止设定值输出）。

关联

- <Axis>.basic.out.progTestRequest
  请求激活“程序测试”功能（转速设定值未生效）
- <HmiAxis>.basic.in.progTestSuppressionRequested
  HMI：已抑制程序测试
- <HmiAxis>.basic.in.progTestRequested
  HMI：已请求进行程序测试

更多信息

- SINUMERIK ONE 基础功能手册；“跨通道程序协调”>“逐通道试运行”章节

12.3.4.37 rapidTraversOvr

PLC → NC	<Axis>.basic.out.rapidTraversOvr
激活设置模式下的快进速度 (MD32010 $MA_JOG_VELO_RAPID)	

说明

在 JOG 运行方式下连续或断续手动运行时，此信号才生效。

信号与以下情形不相关：

- 在回参考点运行中（JOG 运行方式）
- 在 AUTO 和 MDI 运行方式下

快进速度可通过快进倍率开关调整。

数据类型和更新

BOOL，循环

值 TRUE

在借助移动键手动运行机床轴时，此接口信号的置位使快进速度（MD32010）生效。

值 FALSE

在借助移动键手动运行机床轴时，设定的 JOG 速度（SD41110 或 MD32020）生效。

关联：

- <Axis>.basic.out.traversKeyMinus
  以设置模式朝负向方向移动加工轴 (MD32020 $MA_JOG_VELO)
- <Axis>.basic.out.traversKeyPlus
  以设置模式朝正向方向移动加工轴 (MD32020 $MA_JOG_VELO)
- SD41110 $SN_JOG_SET_VELO
  设置模式下的轴速度
- MD32010 $MA_JOG_VELO_RAPID
  基准快速运行
- MD32020 $MA_JOG_VELO
  基准轴速度

详细信息

- 功能手册之 SINUMERIK ONE 轴和主轴；章节“手动移动”

12.3.4.38 refPointApproachCam

PLC → NC	<Axis>.basic.out.refPointApproachCam
到达参考凸轮，低转速接近参考点	

说明

建议采用适宜的从参考点延伸至运行范围末端的参考凸轮，以此确保机床轴在回参考点运行前处于该参考凸轮后的区域内。

数据类型与更新

BOOL，循环

值 TRUE

机床轴位于参考凸轮上。

值 FALSE

机床轴位于参考凸轮前。

关联

- <Axis>.basic.out.refPointSelection
  选择参考点（1 至 4）：最多可以使用四个不同的参考点 (MD34100 $MA_REFP_SET_POS)

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“回参考点”章节

12.3.4.39 refPointSelection

PLC → NC	<Axis>.basic.out.refPointSelection
选择参考点 (1 - 4)：最多可以使用四个不同的参考点 (MD34100 $MA_REFP_SET_POS)	

说明

到达参考凸轮后，NC 通过此接口报告已定位至哪个编码的参考凸轮，以及需要将哪个参考点值作为轴实际值置位：

Istwert = MD34100 $MA_REFP_SET_POS [<参考点值 x>]

说明

- 此接口信号必须一致保持置位，直至到达参考点或运行到新编码的参考凸轮。
- 若机床轴到达参考点，并且未选择参考点值，则使用参考点值 1。
- 就带有距离编码的参考标记的长度测量系统而言，此接口信号不相关。

应用示例

在运行行程较大的机床上，可通过分布在轴的运行行程上的至多四个编码的参考凸轮定位至至多四个不同的参考点。由此可以减小回参考点过程的运行时间。

数据类型和更新

USInt，循环

值

0：参考点值 1

1 - 4：参考点值 1 至 4

关联

- MD34100 $MA_REFP_SET_POS
  参考点值

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“回参考点”章节

12.3.4.40 reposDelay

PLC → NC	<Axis>.basic.out.reposDelay
已请求 REPOS 延迟	

说明

如果机床轴参与轨迹运行（<Axis>.basic.in.isPathAxis == 1 (轨迹轴)），则此接口信号不生效。

数据类型与更新

BOOL，循环

值 TRUE

已请求 REPOS 延时。

程序段搜索后不通过定位程序段为该轴运行 REPOS 偏移，而是通过编写了该轴的下一个运行程序段运行。

值 FALSE

未请求 REPOS 延时。

执行程序段搜索后，通过接近程序段为该轴运行一次 REPOS 偏移。

值	REPOS 接近运行方式
0	无 REPOS 运行方式
1	接近程序段起点 (RMBBL)
2	接近中断点 (RMIBL)
3	接近程序段终点 (RMEBL)
4	接近下一个轨迹点 (RMNBL)

关联

- <HmiChan>.basic.in.reposModeChanged
  HMI：REPOS 运行方式切换
- <Chan>.basic.out.reposModeSelection
  REPOS 接近运行方式：
  可能的值见上表
- <Chan>.basic.out.reposModeActivationRequest
  请求为当前主运行程序段激活“REPOS”运行方式
- <Chan>.basic.in.reposModeChangeAck
  应答 REPOS 接近运行方式的变更请求
- <Chan>.basic.in.reposModeSelectedNumber
  选择的 REPOS 接近运行方式：
  可能的值见上表
- <Chan>.basic.in.reposDelayActive
  “在 REPOS 接近运行方式下延迟”处于活动状态
- <Axis>.basic.in.reposOffsetActive
  REPOS 偏移已应用在轴上
- <Axis>.basic.in.reposOffsetValid
  REPOS 偏移计算为有效
- <Axis>.basic.in.reposDelayAck
  已应答 REPOS 延迟

- <Axis>.basic.in.isPathAxis
  轴为轨迹轴
- MD11470 $MN_REPOS_MODE_MASK
  重新定位属性

更多信息

- SINUMERIK ONE 基础功能手册；“运行方式组、通道、程序运行、复位行为” > “程序段搜索类型 5 (SERUPRO)” > “再次返回轮廓 (REPOS)”章节

12.3.4.41 reposPtp

PLC → NC	<Axis>.basic.out.reposPtp
REPOS 以进行“点对点”(PTP) 坐标转换	

数据类型与更新

BOOL，循环

12.3.4.42 resetMovement

PLC → NC	<Axis>.basic.out.resetMovement
脉冲沿切换 0 → 1：已请求“轴 - 删除剩余行程”；已请求“主轴 - 复位”	

说明

轴：删除剩余行程

- 运行方式 AUTOMATIC 和 MDA
  仅当将轴作为定位轴运行时，此接口信号才生效。定位轴此时会通过其制动特性曲线制动至静止状态。还未运行的轴剩余行程会被删除。
- JOG 运行方式
  轴通过其加速度特性曲线制动至静止状态，随后将可能的剩余行程删除。

说明

在成功实施剩余行程删除后，在 NC 程序中以轴的当前位置重新处理后续程序段。这样便运行另一不同于 NC 程序中的编程的轮廓。通过在“删除剩余行程”后的程序段中运行到绝对位置 G90，至少能够定位至编写的位置。在借助增量尺寸 G91 运行时，不会定位至编写的位置。

主轴：复位

- 控制模式：
  - 停止主轴
  - 继续执行 NC 程序
  - 主轴以下一个在 NC 程序中编写的 M 值及 S 值继续旋转。
- 往复模式：
  - 往复终止
  - 轴继续运行
  - 以当前的齿轮档继续执行 NC 程序。
  - 主轴以下一个在 NC 程序中编写的 M 值及 S 值继续旋转。由此产生的转速（当前 S 值和最后生效的齿轮档）可能会过大。在此情形下，信号接口不置位：
    <Axis>.spindle.in.speedSetpointAboveMax
- 定位模式：停止主轴
- 轴模式：停止主轴

说明

主轴复位时的特性与以下机床数据中的设置无关：
MD35040 $MA_SPIND_ACTIVE_AFTER_RESET

暂停时间（G4）：删除剩余行程

在程序段包含暂停时间（G4）的情况下，此接口信号对运行时间无影响。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

轴：已请求“删除剩余行程”。

主轴：已请求复位

脉冲沿切换 1 → 0

无作用

关联

- <Chan>.basic.out.deleteDistanceToGo
  脉冲沿切换 0 → 1：已请求删除剩余行程（通道专用剩余行程）
- <Axis>.spindle.in.speedSetpointAboveMax
  额定转速超过通过 G26 编程的最大转速极限值 (SD43220 $SA_SPIND_MAX_VELO_G26)
- MD35040 $MA_SPIND_ACTIVE_AFTER_RESET（独立主轴复位）

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“定位轴”章节

12.3.4.43 swCams

PLC → NC	<Axis>.basic.out.swCams
激活“输出软件凸轮正信号和负信号”	

数据类型与更新

BOOL，循环

值 TRUE

请求激活将轴的正负挡块信号输出至 PLC 接口。

值 FALSE

未请求激活将轴的正负挡块信号输出至 PLC 接口。

关联

- <Nc>.basic.in.swCams[x].minus
  直线轴通过负凸轮 0 沿负方向移动（模数回转轴行为不同）
- <Nc>.basic.in.swCams[x].plus
  直线轴通过正凸轮 0 沿正方向移动（模数回转轴行为不同）
- <Axis>.basic.in.swCamsActive
  软件凸轮应用处于活动状态。到达一个软件凸轮后，PLC 发布相应的信号。

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“软件凸轮轴，行程开关信号”章节

12.3.4.44 swLimitSwitch2Minus

PLC → NC	<Axis>.basic.out.swLimitSwitch2Minus
激活第二个负软件限位开关	

数据类型与更新

BOOL，循环

值 TRUE

已触发第二软件限位开关 -。

值 FALSE

未触发第二软件限位开关 -。

关联

- MD36100 $MA_POS_LIMIT_MINUS
  第 1 负向软件限位开关
- MD36120 $MA_POS_LIMIT_MINUS2
  第 2 负向软件限位开关
- MD36110 $MA_POS_LIMIT_PLUS
  第 1 正向软件限位开关
- MD36130 $MA_POS_LIMIT_PLUS2
  第 2 正向软件限位开关

12.3.4.45 swLimitSwitch2Plus

PLC → NC	<Axis>.basic.out.swLimitSwitch2Plus
激活第二个正软件限位开关	

数据类型与更新

BOOL，循环

值 TRUE

已触发第二软件限位开关 +。

值 FALSE

未触发第二软件限位开关 +。

关联

- MD36100 $MA_POS_LIMIT_MINUS
  第 1 负向软件限位开关
- MD36120 $MA_POS_LIMIT_MINUS2
  第 2 负向软件限位开关
- MD36110 $MA_POS_LIMIT_PLUS
  第 1 正向软件限位开关
- MD36130 $MA_POS_LIMIT_PLUS2
  第 2 正向软件限位开关

12.3.4.46 traversKeyMinus

PLC → NC	<Axis>.basic.out.traversKeyMinus
在 JOG 运行方式下沿负方向移动机床轴 (MD32020 $MA_JOG_VELO)	

说明

根据生效的机床功能以及点按及长按模式（SD41050 和 MD11300）的设置，在信号切换时触发不同的响应：

1. 在点按模式下连续手动运行
   机床轴一直沿相应方向移动至生效的限位开关，如同接口信号具有信号状态 1 的情形。
2. 在长按模式下连续手动运行
   在该信号发出首个上升沿 0 → 1 后，机床轴开始朝对应方向移动。即使之后信号输出下降沿 1 → 0，轴也继续移动。在该信号再次给出上升沿 0 → 1 后（同一个移动方向！），轴终止移动。
3. 在点按模式下断续手动运行
   该信号置 1 后，机床轴移动指定的步数。若在移动完步数前，信号切换至状态 0，则移动中断。信号状态恢复为 1 后，轴会继续移动。在运行完指定增量行程前，轴的移动可以如上文所述多次停止和继续。
4. 在长按模式下断续手动运行
   在该信号输出第一个上升沿 0 → 1 后，机床轴开始移动指定的步数。如果在几何轴移动完步数前，同一个方向信号再次输出上升沿 0 → 1，则终止移动。将不会把步数移动完。

说明

此接口信号与以下运行方式不相关：

- AUTOMATIC
- MDA

若两个移动信号（"+"和"-"）同时置位，则不移动或终止当前移动。

通过 <Axis>.basic.out.disableTraversKeys（移动键锁定），可以借助移动键锁定运行。

数据类型与更新

BOOL，循环

值 TRUE

已请求借助移动键沿负轴方向移动机床轴。

值 FALSE

未请求借助移动键沿负轴方向移动机床轴。

关联

- <Axis>.basic.out.disableTraversKeys
  锁定“正”、“负”移动键
- <Axis>.basic.in.traversMinusCommandActive
  负方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)
- <Axis>.basic.in.traversPlusCommandActive
  正方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)
- MD11300 $MN_JOG_INC_MODE_LEVELTRIGGRD
  INC 和 REF 在点动运行模式下
- SD41050 $SN_JOG_CONT_MODE_LEVELTRIGGRD
  持续点动模式下的点动/连续运行

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.4.47 traversKeyPlus

PLC → NC	<Axis>.basic.out.traversKeyPlus
在 JOG 运行方式下沿正方向移动机床轴 (MD32020 $MA_JOG_VELO)	

说明

根据生效的机床功能以及点按及长按模式（SD41050 和 MD11300）的设置，在信号切换时触发不同的响应：

1. 在点按模式下连续手动运行
   机床轴一直沿相应方向移动至生效的限位开关，如同接口信号具有信号状态 1 的情形。
2. 在长按模式下连续手动运行
   在该信号发出首个上升沿 0 → 1 后，机床轴开始朝对应方向移动。即使之后信号输出下降沿 1 → 0，轴也继续移动。在该信号再次给出上升沿 0 → 1 后（同一个移动方向！），轴终止移动。
3. 在点按模式下断续手动运行
   该信号置 1 后，机床轴移动指定的步数。若在移动完步数前，信号切换至状态 0，则移动中断。信号状态恢复为 1 后，轴会继续移动。在运行完指定增量行程前，轴的移动可以如上文所述多次停止和继续。
4. 在长按模式下断续手动运行
   在该信号输出第一个上升沿 0 → 1 后，机床轴开始移动指定的步数。如果在几何轴移动完步数前，同一个方向信号再次输出上升沿 0 → 1，则终止移动。将不会把步数移动完。

说明

此接口信号与以下运行方式不相关：

- AUTOMATIC
- MDA

若两个移动信号（"+"和"-"）同时置位，则不移动或终止当前移动。

通过 <Axis>.basic.out.disableTraversKeys（移动键锁定），可以借助移动键锁定运行。

数据类型与更新

BOOL，循环

值 TRUE

已请求借助移动键沿正轴方向移动机床轴。

值 FALSE

未请求借助移动键沿正轴方向移动机床轴。

关联

- <Axis>.basic.out.disableTraversKeys
  锁定“正”、“负”移动键
- <Axis>.basic.in.traversMinusCommandActive
  负方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)
- <Axis>.basic.in.traversPlusCommandActive
  正方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)

- MD11300 $MN_JOG_INC_MODE_LEVELTRIGGRD
  INC 和 REF 处于点动运行模式下
- SD41050 $SN_JOG_CONT_MODE_LEVELTRIGGRD
  持续点动模式下的点动/连续运行

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.5 <Axis>.drive.in

12.3.5.1 autonomousMotionActive

NC → PLC	<Axis>.drive.in.autonomousMotionActive
驱动自动控制运动处于活动状态	

说明

采用 611U 报文类型时，SINAMICS 适用 NC 62.07 及以上版本：

当 MELDW.11 == TRUE（控制器使能）且 ZSW1.2 == FALSE（运行使能）时，变量 = TRUE

说明

- SINAMICS S120：信息字(MELDW)
  消息字 (MELDW) 仅包含在 SIMODRIVE 611u 兼容的 PROFIDrive 报文中，例如报文 102、103、105、106、110、111、116、118、125、126、136、138、139
  文档：SINAMICS 参数手册，功能图 2419 和 2420
- SINAMICS S120：状态字 1 / 2 (ZSW1/2)
  状态字 ZSW1 或 ZSW2 仅基于 SIMODRIVE 611u 兼容的 PROFIDrive 报文（SIMODRIVE 611u 接口模式，p2038 = 1）

应用示例

驱动内部功能：

- 转子或磁极位置识别
- 函数发生器

数据类型与更新

BOOL，循环

值 TRUE

驱动自控运动生效。

轴基于驱动内部功能生成的设定值运行。驱动还对 NC 的控制信号（例如控制器使能）进行响应。NC 给定的设定值会被忽略。

值 FALSE

驱动自控运动未生效。

更多信息

- SINAMICS S120 功能手册；
- 磁极位置识别：章节“伺服控制”>“磁极位置识别”
- 功能发生器：章节“伺服控制”>“电流控制器和转速控制器优化”

12.3.5.2 classCAlarmActive

NC → PLC	<Axis>.drive.in.classCAlarmActive
驱动报告 C 级警告/报警	

说明

报警是针对驱动识别出或预期的故障状态的响应，其不会引起驱动关闭，且无需应答。

数据类型与更新

BOOL，循环

值 TRUE

驱动报告出现 C 级报警。

值 FALSE

驱动报告无 C 级报警。

更多信息

- SINAMICS S120 参数手册，章节“故障和报警”

12.3.5.3 currentCtrlActive

NC → PLC	<Axis>.drive.in.currentCtrlActive
电流控制器处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

电流控制器处于活动状态，即轴/主轴的电流控制环闭合。

值 FALSE

电流控制器未激活，即轴的电流控制环未闭合。

电流控制器输出（包含对调节电压的前馈量）会被删除。

关联

- <Axis>.basic.in.posCtrlActive
  位置控制器处于活动状态
- <Axis>.basic.in.speedCtrlActive
  转速控制器处于活动状态

12.3.5.4 driveCtrlActive

NC → PLC	<Axis>.drive.in.driveCtrlActive
额定转速切换：轴已应用驱动控制	

说明

仅当轴具有对驱动的控制权时，才能进行控制器使能 (<Axis>.basic.out.enableCtrl)：

<Axis>.drive.in.driveCtrlActive == TRUE

说明

所有参与设定值切换，且当前不具有驱动控制权的轴由控制系统切换至跟踪运行。亦即，其不处于位置闭环控制下。因此建议为悬挂轴配备制动控制系统。

数据类型与更新

BOOL，循环

值 TRUE

轴已接收对驱动的控制。

值 FALSE

轴未接收对驱动的控制。

关联

- <Axis>.basic.out.enableCtrl
  请求控制器使能
- <Axis>.drive.out.requestDriveCtrl
  请求应用驱动控制

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“额定值转换”章节

12.3.5.5 driveDataSetNumber

NC → PLC	<Axis>.drive.in.driveDataSetNumber
当前活动的驱动数据组 (DDS) 的显示接口	

当前有效的驱动数据组 (DDS) 的显示接口。

格式

显示接口的格式，即哪些比特位用于驱动数据组 (DDS) 寻址，通过格式接口 (<Axis>.drive.in.numberOfDriveDataSets) 进行设置。

数据类型和更新

USInt，循环

值

当前有效驱动数据组 (DDS)

关联

- <Axis>.drive.out.driveDataSetSelection
  选择驱动数据组 (1 - maxNumDataSet)
- <Axis>.drive.in.numberOfDriveDataSets
  当前活动的电机/驱动数据组 (MDS/DDS) 的格式

更多信息

- SINUMERIK ONE 基础功能手册，“若干 NC/PLC 接口信号与功能”>“功能”>“电机/驱动数据组切换”章节

12.3.5.6 driveTestTraversRequested

NC → PLC	<Axis>.drive.in.driveTestTraversRequested
已为驱动测试请求运行使能，因为满足轴的运行前提条件	

数据类型和更新

BOOL，循环

值 TRUE

已为驱动测试请求运行使能，因为满足轴的运行条件：

- 相关轴的机械制动已提前松开
- 请求进给轴/主轴锁定：<Axis>.basic.out.disable== FALSE

通过 <Axis>.drive.out.enableDriveTestTravers == TRUE 进行反馈

值 FALSE

未请求用于驱动测试的运行使能。

关联

- <Axis>.drive.out.enableDriveTestTravers
  存在用于驱动测试的运行使能

12.3.5.7 enabled

NC → PLC	<Axis>.drive.in.enabled
驱动模块运行已使能	

数据类型和更新

BOOL, 循环

值 TRUE

驱动运行已使能。

值 FALSE

驱动运行未使能。

12.3.5.8 esrDcVoltageAlarmActive

NC → PLC	<Axis>.drive.in.esrDcVoltageAlarmActive
延时停止和回退 (ESR)：直流母线电压低于规定的最小值 (p1248)	

数据类型与更新

BOOL，循环

值 TRUE

直流母线电压低于最小值 (p1248)

值 FALSE

直流母线电压不低于最小值 (p1248)

12.3.5.9 esrGeneratorOperationActive

NC → PLC	<Axis>.drive.in.esrGeneratorOperationActive
延时停止和回退 (ESR)：发电功能处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

发电功能处于活动状态 (r0887.12)。

值 FALSE

发电功能未激活

12.3.5.10 esrSpeedBelowMin

NC → PLC	<Axis>.drive.in.esrSpeedBelowMin
延时停止和回退 (ESR)：实际转速低于规定的最小转速 (p2161) - 轴为发电轴	

数据类型与更新

BOOL，循环

值 TRUE

低于发电功能下当前轴的最小转速 (p2161)

值 FALSE

不低于发电功能下当前轴的最小转速 (p2161)

12.3.5.11 heatSinkTemperatureWarning

NC → PLC	<Axis>.drive.in.heatSinkTemperatureWarning
电力半导体的散热器温度已超过编程的警告阈值 (p0294)	

说明

接口信号 <Axis>.drive.in.motorTemperaturWarning 和
<Axis>.drive.in.heatSinkTemperatureWarning 通过循环驱动报文的以下信号推导得出：

- 情形 1：信息字中的温度报警
  - <Axis>.drive.in.motorTemperaturWarning" ≙ MELDW, Bit 6（无电机过热警告）
  - <Axis>.drive.in.heatSinkTemperatureWarning ≙ MELDW, Bit 7（无功率部件热过载报警）
- 情形 2：B 级警告（“SIMODRIVE 611u”接口模式，p2038=TRUE）
  <Axis>.drive.in.motorTemperaturWarning == TRUE 且
  <Axis>.drive.in.heatSinkTemperatureWarning == TRUE，前提条件是：
  循环驱动报文，ZSW1：Bit 11 == FALSE 且 Bit 12 == TRUE（B 级警告）

当没有来自信息字的特殊信息时，便从 B 级报警中引导出接口信号。

显示报警：报警号 = 200.000 + 报警值 (r2124)

说明

- SINAMICS S120：信息字(MELDW)
  消息字 (MELDW) 仅包含在 SIMODRIVE 611u 兼容的 PROFIDrive 报文中，例如报文 102、103、105、106、110、111、116、118、125、126、136、138、139
  文档：SINAMICS 参数手册，功能图 2419 和 2420
- SINAMICS S120：状态字 1 / 2 (ZSW1/2)
  状态字 ZSW1 或 ZSW2 仅基于 SIMODRIVE 611u 兼容的 PROFIDrive 报文（SIMODRIVE 611u 接口模式，p2038 = TRUE）

数据类型与更新

BOOL，循环

值 TRUE

功率半导体的散热器温度超出了设置的报警阈值（p0294）。

说明

驱动中进行设置的响应（p0290）。若超温状态持续，系统会在约 20 s 后输出故障，驱动会制动停止，脉冲使能会被取消。

值 FALSE

散热器温度低于报警阈值。

更多信息

- S120 调试手册，章节“调试” > “SINAMICS 组件上的温度传感器”
- S120 功能手册，章节“监控功能和保护功能”
- S120 参数手册
  - MELDW，位 6 ≙ BO:r2135.14 → 功能图：2548，8016
  - MELDW，位 7 ≙ BO:r2135.15 → 功能图：2548，2452，2456，8016
- SINUMERIK 诊断手册

12.3.5.12 holdingBrakeReleased

NC → PLC	<Axis>.drive.in.holdingBrakeReleased
抱闸已释放	

数据类型与更新

BOOL

值 TRUE

抱闸打开，电机可转动。

值 FALSE

抱闸未打开。

12.3.5.13 motorDataSetNumber

NC → PLC	<Axis>.drive.in.motorDataSetNumber
当前活动的电机数据组 (MDS) 的显示接口	

说明

当前有效电机数据组 (DDS) 的显示接口。

格式

显示接口的格式，即哪些比特位用于电机数据组 (MDS) 寻址，通过格式接口 (<Axis>.drive.in.numberOfDriveDataSets) 进行设置。

数据类型和更新

USInt，循环

值

当前有效电机数据组 (DDS)

关联

- <Axis>.drive.out.motorDataSetSelection
  选择电机数据组 (1 - maxNumDataSet)
- <Axis>.drive.out.motorChangeoverDone
  已完成所有必要的电气和（或）机械改装操作（更换刀盘等）
- <Axis>.drive.in.numberOfDriveDataSets
  当前活动的电机/驱动数据组 (MDS/DDS) 的格式

更多信息

- SINUMERIK ONE 基础功能手册，“若干 NC/PLC 接口信号与功能”>“功能”>“电机/驱动数据组切换”章节

12.3.5.14 motorTemperatureWarning

NC → PLC	<Axis>.drive.in.motorTemperatureWarning
电机温度已超过驱动中配置的警告阈值 (p0604)。	

说明

当前电机温度会在操作界面上的以下区域显示：

操作区“诊断”>“维护显示：轴/主轴”

数据类型与更新

BOOL，循环

值 TRUE

电机温度超出了驱动中配置的报警阈值（p0604）。

说明

若电机温度过高时间长于设置的时间级（p0606），系统会输出故障消息，驱动会制动停止，脉冲使能会被取消。

若电机温度在时间级（p0606）结束前重新降至报警阈值（p0604）以下，那么该接口信号会被重新复位。

值 FALSE

电机温度处于报警阈值以下（p0604）。

关联

- <Axis>.drive.in.heatSinkTemperatureWarning
  电力半导体的散热器温度已超过编程的警告阈值 (p0294)

12.3.5.15 numberOfDriveDataSets

NC → PLC	<Axis>.drive.in.numberOfDriveDataSets
当前活动的电机/驱动数据组 (MDS/DDS) 的格式	

数据类型和更新

USInt，循环

关联

- <Axis>.drive.out.numberOfDriveDataSetsValid
  当前活动的电机/驱动数据组 (MDS/DDS) 的格式有效
- <Axis>.drive.out.driveDataSetSelection
  选择驱动数据程序段 (1 - maxNumDataSet)
- <Axis>.drive.out.motorDataSetSelection
  选择电机数据组 (1 - maxNumDataSet)
- <Axis>.drive.out.motorChangeoverDone
  已完成所有必要的电气和/或机械改装操作（更换刀盘等）。

- <Axis>.drive.in.driveDataSetNumber
  当前活动的驱动数据组 (DDS) 的显示接口
- <Axis>.drive.in.motorDataSetNumber
  当前活动的电机数据组 (MDS) 的显示接口

更多信息

- 基础功能功能手册，“不同的 NC/PLC 接口信号与功能章节” > “电机/驱动数据组的切换”
- 调试手册 IBN CNC：NC、PLC、驱动

12.3.5.16 numberOfDriveDataSetsValid

NC → PLC	<Axis>.drive.in.numberOfDriveDataSetsValid
当前活动的电机/驱动数据组 (MDS/DDS) 的格式有效	

数据类型和更新

BOOL，循环

关联

- <Axis>.drive.out.numberOfDriveDataSets
  当前活动的电机/驱动数据组 (MDS/DDS) 的格式
- <Axis>.drive.out.driveDataSetSelection
  选择驱动数据程序段 (1 - maxNumDataSet)
- <Axis>.drive.out.motorDataSetSelection
  选择电机数据组 (1 - maxNumDataSet)
- <Axis>.drive.out.motorChangeoverDone
  已完成所有必要的电气和/或机械改装操作（更换刀盘等）。
- <Axis>.drive.in.driveDataSetNumber
  当前活动的驱动数据组 (DDS) 的显示接口
- <Axis>.drive.in.motorDataSetNumber
  当前活动的电机数据组 (MDS) 的显示接口

更多信息

- 基础功能功能手册，“不同的 NC/PLC 接口信号与功能章节” > “电机/驱动数据组的切换”
- 调试手册 IBN CNC：NC、PLC、驱动

12.3.5.17 posCtrlParameterSetNumber

NC → PLC	<Axis>.drive.in.posCtrlParameterSetNumber
位置控制器当前有效参数组的编号	

说明

在切换关闭的情况下，此接口不相关：

MD35590 $MA_PARAMSET_CHANGE_ENABLE == FALSE

在此情形下，参数组 1 始终生效。

数据类型和更新

INT，循环

值

当前活动的位置控制器参数组编号

关联

- <Axis>.basic.out.posCtrlParameterSetSelection
  为位置控制器选择参数组 (1 - 8)

12.3.5.18 pulsesEnabled

NC → PLC	<Axis>.drive.in.pulsesEnabled
驱动脉冲已使能	

说明

<Axis>.drive.in.pulsesEnabled = MELDW.13

说明

- SINAMICS S120：信息字(MELDW)
  消息字 (MELDW) 仅包含在 SIMODRIVE 611u 兼容的 PROFIDrive 报文中，例如报文 102、103、105、106、110、111、116、118、125、126、136、138、139
  文档：SINAMICS 参数手册，功能图 2419 和 2420
- SINAMICS S120：状态字 1 / 2 (ZSW1/2)
  状态字 ZSW1 或 ZSW2 仅基于 SIMODRIVE 611u 兼容的 PROFIDrive 报文（SIMODRIVE 611u 接口模式，p2038 = TRUE）

数据类型与更新

BOOL，循环

值 TRUE

驱动中脉冲已使能。

值 FALSE

驱动中脉冲未使能 ⇒

- <Axis>.drive.in.currentCtrlActive = FALSE（电流控制器处于活动状态）
- <Axis>.basic.in.speedCtrlActive = FALSE（转速控制器处于活动状态）
- <Axis>.basic.in.posCtrlActive = FALSE（位置控制器激活）

关联

- <Axis>.drive.out.enablePulses
  用于驱动的控制脉冲使能

12.3.5.19 rapidStopActive

NC → PLC	<Axis>.drive.in.rapidStopActive
斜坡函数发生器锁定（快速停止，转速设定值为 0）	

数据类型与更新

BOOL，循环

值 TRUE

对于轴的驱动，斜坡函数发生器快速停止生效。

值 FALSE

对于轴的驱动，斜坡函数发生器快速停止未生效。

关联

- <Axis>.drive.out.rapidStop
  无斜坡函数发生器停止（快速停止，额定转速为 0）

12.3.5.20 ready

NC → PLC	<Axis>.drive.in.ready
驱动就绪	

说明

<Axis>.drive.in.ready = 驱动：MELDW.12

说明

- SINAMICS S120：信息字(MELDW)
  消息字 (MELDW) 仅包含在 SIMODRIVE 611u 兼容的 PROFIDrive 报文中，例如报文 102、103、105、106、110、111、116、118、125、126、136、138、139
  文档：SINAMICS 参数手册，功能图 2419 和 2420
- SINAMICS S120：状态字 1 / 2 (ZSW1/2)
  状态字 ZSW1 或 ZSW2 仅基于 SIMODRIVE 611u 兼容的 PROFIDrive 报文（SIMODRIVE 611u 接口模式，p2038 = TRUE）

数据类型与更新

BOOL，循环

值 TRUE

驱动已经准备好运行。

值 FALSE

驱动未就绪。

若在正在执行的运行中复位了该信号，那么驱动会被制动停止（脉冲禁止或快速停止）。启动中会保持脉冲禁止。另外还会复位以下接口信号：

- <Nc>.basic.in.drivesReady = FALSE（NC 所有机床轴的相关驱动都已准备就绪）
- <Axis>.drive.in.currentCtrlActive = FALSE（电流控制器处于活动状态）
- <Axis>.basic.in.speedCtrlActive = FALSE（转速控制器处于活动状态）

关联

- <Nc>.basic.in.drivesReady
  NC 所有机床轴的相关驱动都已准备就绪
- <Axis>.drive.in.currentCtrlActive
  电流控制器处于活动状态
- <Axis>.basic.in.speedCtrlActive
  转速控制器处于活动状态

更多信息

- SINUMERIK ONE 基础功能手册，“若干 NC/PLC 接口信号与功能”>“功能”>“电机/驱动数据组切换”章节
- SINAMICS S120/S150 参数手册

12.3.5.21 speedAtSetpoint

NC → PLC	<Axis>.drive.in.speedAtSetpoint
n_实际 = n_设定：实际转速（n_实际）与设定值（n_设定）相同，在指定时段 (p2167) 未偏离公差范围 (p2163)	

数据类型与更新

BOOL，循环

值 TRUE

转速实际值至少在设置的时间（$n_{实际} = n_{额定}$ 接通延迟，p2167）内处于转速设定值的公差带（转速阈值 4，p2163）内。

值 FALSE

转速实际值位于转速设定值的公差带（转速阈值 4，p2163）外。

12.3.5.22 speedBelowMin

NC → PLC	<Axis>.drive.in.speedBelowMin
n_实际 < n_最小：实际转速（n_实际）低于规定的最小转速（n_最小，p2161）	

说明

$|n_{实际}| < n_{最小}$

<Axis>.drive.in.speedBelowMin = MELDW.2

说明

- SINAMICS S120：信息字(MELDW)
  消息字 (MELDW) 仅包含在兼容的 PROFIDrive 报文中，例如报文 102、103、105、106、110、111、116、118、125、126、136、138、139
  文档：SINAMICS 参数手册，功能图 2419 和 2420
- SINAMICS S120：状态字 1 / 2 (ZSW1/2)
  状态字 ZSW1 或 ZSW2 仅指兼容的 PROFIDrive 报文

数据类型和更新

BOOL，循环

值 TRUE

转速实际值 n_{act} 小于 n_{min}（p2161 转速阈值 3）。

值 FALSE

转速实际值超出了最小转速阈值 n_{min}。

12.3.5.23 speedBelowThreshold

NC → PLC	<Axis>.drive.in.speedBelowThreshold
n_实际 < n_x：实际转速（n_实际）低于规定的阈值 (n_x, p2155)	

说明

$|n_{实际}| < n_x$

<Axis>.drive.in.speedBelowThreshold = MELDW.3

说明

- SINAMICS S120：信息字(MELDW)
  消息字 (MELDW) 仅包含在 SIMODRIVE 611u 兼容的 PROFIDrive 报文中，例如报文 102、103、105、106、110、111、116、118、125、126、136、138、139
  文档：SINAMICS 参数手册，功能图 2419 和 2420
- SINAMICS S120：状态字 1 / 2 (ZSW1/2)
  状态字 ZSW1 或 ZSW2 仅基于 SIMODRIVE 611u 兼容的 PROFIDrive 报文（SIMODRIVE 611u 接口模式，p2038 = 1）

数据类型和更新

BOOL，循环

值 TRUE

转速实际值 n_{act} 小于 n_x（p2155 转速阈值 2）。

值 FALSE

转速实际值 $n_{实际}$ 超过转速阈值 n_x。

12.3.5.24 speedCtrlIntegratorDisabled

NC → PLC	<Axis>.drive.in.speedCtrlIntegratorDisabled
驱动转速控制器的积分器锁定	

数据类型与更新

BOOL，循环

值 TRUE

驱动中的转速控制器的积分器被禁用。

转速控制器作为 P 控制器生效。

值 FALSE

驱动中的转速控制器的积分器未被禁用。

转速控制器作为比例积分控制器工作。

关联

- <Axis>.drive.out.disableSpeedCtrlIntegrator
  转速控制器：锁定积分器组件 (I)，只比例控制器 (P) 进入活动状态 (PI → P)

12.3.5.25 speedSetpointReached

NC → PLC	<Axis>.drive.in.speedSetpointReached
如已达到编程设置的额定转速（n_设定），且在指定时段 (p2166) 内未偏离公差范围 (p2164)，这种情况下信号会激活，活动状态一直持续到编程设置新的额定转速为止	

数据类型与更新

BOOL，循环

值 TRUE

在给出新的转速设定值 (n_set) 后，转速实际值达到通过驱动参数 p2164 定义的转速公差带，并且在持续时间 p2166 内处于公差带内。

后续因负载变化而导致的转速变化（例如：超出公差带）对该接口信号无影响。信号保持处于活动状态，直至规定新的转速设定值为止。

值 FALSE

修改转速设定值后，驱动仍进行加速。

关联

- <Axis>.drive.in.speedAtSetpoint
  n_实际 = n_设定：实际转速（n_实际）与设定值（n_设定）相同，在指定时段 (p2167) 未偏离公差范围 (p2163)
- <Axis>.drive.in.torqueBelowThreshold
  M_d < M_dx：实际转矩利用率 (M_d) 低于指定阈值 (M_dx, p2194)

更多信息

- SINUMERIK 调试手册 IBN CNC：NCK, PLC, 驱动
- SIMATIC S120 参数手册

12.3.5.26 torqueBelowThreshold

NC → PLC	<Axis>.drive.in.torqueBelowThreshold
M_d < M_dx：实际转矩利用率 (M_d) 低于指定阈值 (M_dx, p2194)	

说明

$|M_d| < M_{dx}$

<Axis>.drive.in.torqueBelowThreshold = MELDW.1

说明

- SINAMICS S120：信息字(MELDW)
  消息字 (MELDW) 仅包含在 SIMODRIVE 611u 兼容的 PROFIDrive 报文中，例如报文 102、103、105、106、110、111、116、118、125、126、136、138、139
  文档：SINAMICS 参数手册，功能图 2419 和 2420
- SINAMICS S120：状态字 1 / 2 (ZSW1/2)
  状态字 ZSW1 或 ZSW2 仅基于 SIMODRIVE 611u 兼容的 PROFIDrive 报文（SIMODRIVE 611u 接口模式，p2038 = 1）

数据类型与更新

BOOL，循环

值 TRUE

当前转矩利用率低于转矩利用率阈值（转矩阈值 2，p2194）。

加速已完成，驱动处于静止状态，且驱动转矩设定值 $|M_d|$ 未超出转矩阈值 M_{dx}。

转矩阈值的特性曲线取决于转速。

在引导启动过程中，<Axis>.drive.in.torqueBelowThreshold == TRUE。在加速过程结束(<Axis>.drive.in.speedSetpointReached) == TRUE)，且转矩阈值的消息闭锁时间结束后，该接口信号才会更新。

值 FALSE

转矩设定值|M_d|超过了转矩阈值 M_{dx}。

借助该接口信号可判断电机是否过载。之后可在 PLC 用户程序中进行对应的响应。

更多信息

- SIMATIC S120 参数手册

12.3.5.27 varSignalingFunc1Active

NC → PLC	<Axis>.drive.in.varSignalingFunc1Active
信号功能变量 1 (r3294.0)，编程的驱动变量超过指定阈值，包括回差	

说明

SINAMICS 适用 SW2.6 及以上版本：

使用“变量报告功能”可对驱动中属性为“可跟踪（traceable）”的 BICO 互联及参数进行监控。

<Axis>.drive.in.varSignalingFunc1Active = MELDW.5

说明

- SINAMICS S120：信息字(MELDW)
  消息字 (MELDW) 仅包含在 SIMODRIVE 611u 兼容的 PROFIDrive 报文中，例如报文 102、103、105、106、110、111、116、118、125、126、136、138、139
  文档：SINAMICS 参数手册，功能图 2419 和 2420
- SINAMICS S120：状态字 1 / 2 (ZSW1/2)
  状态字 ZSW1 或 ZSW2 仅基于 SIMODRIVE 611u 兼容的 PROFIDrive 报文（SIMODRIVE 611u 接口模式，p2038 = 1）

数据类型与更新

BOOL，循环

值 TRUE

参数设置的驱动值超出了设定的阈值（包含回差）。

值 FALSE

参数设置的驱动值低于设定的阈值（包含回差）。

更多信息

- SINAMICS S120 功能手册，章节“伺服控制”>“变量报告功能”

12.3.5.28 varSignalingFunc2Active

NC → PLC	<Axis>.drive.in.varSignalingFunc2Active
信号功能变量 2 (r3294.1)，编程的驱动变量超过指定阈值，包括回差	

说明

SINAMICS 适用 SW2.6 及以上版本：

使用“变量报告功能”可对驱动中属性为“可跟踪（traceable）”的 BICO 互联及参数进行监控。

<Axis>.drive.in.varSignalingFunc2Active = MELDW.5

说明

- SINAMICS S120：信息字(MELDW)
  消息字 (MELDW) 仅包含在 SIMODRIVE 611u 兼容的 PROFIDrive 报文中，例如报文 102、103、105、106、110、111、116、118、125、126、136、138、139
  文档：SINAMICS 参数手册，功能图 2419 和 2420
- SINAMICS S120：状态字 1 / 2 (ZSW1/2)
  状态字 ZSW1 或 ZSW2 仅基于 SIMODRIVE 611u 兼容的 PROFIDrive 报文（SIMODRIVE 611u 接口模式，p2038 = 1）

数据类型与更新

BOOL，循环

值 TRUE

参数设置的驱动值超出了设定的阈值（包含回差）。

值 FALSE

参数设置的驱动值低于设定的阈值（包含回差）。

更多信息

- SINAMICS S120 功能手册，章节“伺服控制”>“变量报告功能”

12.3.5.29 varSignalingFunc3Active

NC → PLC	<Axis>.drive.in.varSignalingFunc3Active
信号功能变量 3 (r3294.2)，编程的驱动变量超过指定阈值，包括回差	

说明

SINAMICS 适用 SW2.6 及以上版本：

使用“变量报告功能”可对驱动中属性为“可跟踪（traceable）”的 BICO 互联及参数进行监控。

<Axis>.drive.in.varSignalingFunc3Active = MELDW.5

说明

- SINAMICS S120：信息字(MELDW)
  消息字 (MELDW) 仅包含在 SIMODRIVE 611u 兼容的 PROFIDrive 报文中，例如报文 102、103、105、106、110、111、116、118、125、126、136、138、139
  文档：SINAMICS 参数手册，功能图 2419 和 2420
- SINAMICS S120：状态字 1 / 2 (ZSW1/2)
  状态字 ZSW1 或 ZSW2 仅基于 SIMODRIVE 611u 兼容的 PROFIDrive 报文（SIMODRIVE 611u 接口模式，p2038 = 1）

数据类型和更新

BOOL，循环

值 TRUE

参数设置的驱动值超出了设定的阈值（包含回差）。

值 FALSE

参数设置的驱动值低于设定的阈值（包含回差）。

更多信息

- SINAMICS S120 功能手册，章节“伺服控制”>“变量报告功能”

12.3.6 <Axis>.drive.out

12.3.6.1 disableSpeedCtrlIntegrator

PLC → NC	<Axis>.drive.out.disableSpeedCtrlIntegrator
转速控制器：锁定积分器组件 (I)，只比例控制器 (P) 进入活动状态 (PI → P)	

说明

转速控制器的积分器禁用被激活时，可能会出现补偿过程，例如：积分器在此前稳定负载时。

驱动通过以下信号就转速控制器的禁用进行反馈：

<Axis>.drive.in.speedCtrlIntegratorDisabled（驱动转速控制器的积分器锁定）

数据类型和更新

BOOL，循环

值 TRUE

转速控制器的积分器（I 分量）已被禁止或需要禁止（P 代替 PI 特性）。

说明

激活转速控制器的积分器锁定时，根据不同的应用情况可能会出现补偿过程（例如：积分器在此前稳定负载时）。

通过接口信号从驱动 <Axis>.drive.in.speedCtrlIntegratorDisabled（驱动转速控制器的积分器锁定）向 PLC 应答完成积分器锁定。

值 FALSE

转速控制器的积分器（I 分量）已使能（PI 特性）。

关联

- <Axis>.drive.in.speedCtrlIntegratorDisabled
  驱动转速控制器的积分器锁定

12.3.6.2 driveDataSetSelection

PLC → NC	<Axis>.drive.out.driveDataSetSelection
选择驱动数据组 (1 - maxNumDataSet)	

说明

通过接口请求切换到新的驱动数据组 (DDS)。

格式

显示接口的格式，即哪些比特位用于驱动数据组 (DDS) 寻址，通过格式接口 (<Axis>.drive.in.numberOfDriveDataSets) 进行设置。

主轴驱动

就主主轴驱动而言，以下划分适用：

- MDS[0] → 星形接线方式
- MDS[1] → 三角形接线方式

切换时间点

原则上驱动参数组的切换可在任意时间执行。但是在切换转速控制器参数和电机转速定标时可能会出现转矩跃变，因此建议只在静止状态下执行切换，特别是轴处于静止状态。

只要在驱动中识别到切换至另一个电机数据组的请求，脉冲使能就会被复位。

应用示例

- 主轴驱动
  比如，通过电机数据组切换，可在主主轴驱动 (HSD) 上在运行方式 1（星形）和运行方式 2（三角形）之间进行切换。
- 驱动参数组
  比如，以下情况会用到驱动参数组切换：
  – 齿轮档切换
  – 测量回路切换

数据类型与更新

BOOL，循环

值

所请求驱动数据组 (DDS) 的编号

关联

- <Axis>.drive.in.driveDataSetNumber
  当前活动的驱动数据组 (DDS) 的显示接口
- <Axis>.drive.in.motorDataSetNumber
  当前活动的电机数据组 (MDS) 的显示接口

- <Axis>.drive.out.motorChangeoverDone
  已完成所有必要的电气和（或）机械改装操作（更换刀盘等）
- <Axis>.drive.in.numberOfDriveDataSets
  当前活动的电机/驱动数据组 (MDS/DDS) 的格式

更多信息

- 基础功能手册，“若干 NC/PLC 接口信号与功能”>“功能”>“电机/驱动数据组切换”章节

12.3.6.3 enableDriveTestTravers

PLC → NC	<Axis>.drive.out.enableDriveTestTravers
存在用于驱动测试的运行使能	

说明

此信号是对以下请求的反馈：

<Axis>.drive.in.driveTestTraversRequested == 1 （驱动测试运行请求）

对轴的运行使能的决定权只取决于 PLC。

数据类型与更新

BOOL，循环

值 TRUE

存在用于驱动测试的运行使能，即已为驱动测试使能轴运行。

值 FALSE

不存在用于驱动测试的运行使能，即已为驱动测试锁定轴运行。

关联

- <Axis>.drive.in.driveTestTraversRequested
  已为驱动测试请求运行使能，因为满足轴的运行前提条件

12.3.6.4 enablePulses

PLC → NC	<Axis>.drive.out.enablePulses
用于驱动的控制脉冲使能	

说明

仅当驱动发出就绪信号时，驱动中才进行脉冲使能：
<Axis>.drive.in.ready == TRUE（反馈：驱动就绪）

在运动中取消脉冲使能时，轴/主轴不再受控制动。此时轴/主轴会惰转制动。

数据类型和更新

BOOL，循环

值 TRUE

已为驱动使能脉冲。

值 FALSE

已为驱动禁止脉冲。

关联

- <Axis>.drive.in.ready
  驱动就绪
- <Axis>.drive.in.pulsesEnabled
  驱动脉冲已使能

12.3.6.5 motorChangeoverDone

PLC → NC	<Axis>.drive.out.motorChangeoverDone
已完成所有必要的电气和（或）机械改装操作（更换刀盘等）	

说明

在请求切换到新电机数据组 (MDS) 和（或）驱动数据组 (DDS) (<Axis>.drive.out.driveDataSetSelection, <Axis>.drive.out.motorDataSetSelection) 之前，必须将信号复位。

数据类型与更新

BOOL，循环

值 TRUE

必要的电子和/或机械切换（例如星形/三角形切换时的接触器切换）已完成：

- 轴不允许重新运行
- 驱动会使能脉冲

值 FALSE

必要的电气（例如星-三角切换中的接触器切换）和（或）机械切换尚未完成：

- 轴不允许运行。
- 驱动不会使能脉冲

关联

- <Axis>.drive.out.driveDataSetSelection
  选择驱动数据组 (1 - maxNumDataSet)
- <Axis>.drive.out.motorDataSetSelection
  选择电机数据组 (1 - maxNumDataSet)
- <Axis>.drive.in.driveDataSetNumber
  当前活动的驱动数据组 (DDS) 的显示接口
- <Axis>.drive.in.motorDataSetNumber
  当前活动的电机数据组 (MDS) 的显示接口
- <Axis>.drive.in.numberOfDriveDataSets
  当前活动的电机/驱动数据组 (MDS/DDS) 的格式

更多信息

- 基础功能手册，“若干 NC/PLC 接口信号与功能”>“功能”>“电机/驱动数据组切换”章节

12.3.6.6 motorDataSetSelection

PLC → NC	<Axis>.drive.out.motorDataSetSelection
选择电机数据组 (1 - maxNumDataSet)	

说明

通过接口请求切换到新的电机数据组 (MDS)。

格式

请求接口的格式，即哪些比特位用于电机数据组 (MDS) 寻址，通过格式接口 (<Axis>.drive.in.numberOfDriveDataSets) 进行设置。

主轴驱动

就主主轴驱动而言，以下划分适用：

- MDS[0] → 星形接线方式
- MDS[1] → 三角形接线方式

切换时间点

原则上驱动参数组的切换可在任意时间执行。但是在切换转速控制器参数和电机转速定标时可能会出现转矩跃变，因此建议只在静止状态下执行切换，特别是轴处于静止状态。

只要在驱动中识别到切换至另一个电机数据组的请求，脉冲使能就会被复位。

应用示例

- 主轴驱动
  比如，通过电机数据组切换，可在主主轴驱动 (HSD) 上在运行方式 1（星形）和运行方式 2（三角形）之间进行切换。
- 驱动参数组
  比如，以下情况会用到驱动参数组切换：
  – 齿轮档切换
  – 测量回路切换

数据类型和更新

USInt，循环

值

所请求电机数据组 (MDS) 的编号

关联

- <Axis>.drive.in.driveDataSetNumber
  当前活动的驱动数据组 (DDS) 的显示接口
- <Axis>.drive.in.motorDataSetNumber
  当前活动的电机数据组 (MDS) 的显示接口
- <Axis>.drive.out.motorChangeoverDone
  已完成所有必要的电气和（或）机械改装操作（更换刀盘等）
- <Axis>.drive.in.numberOfDriveDataSets
  当前活动的电机/驱动数据组 (MDS/DDS) 的格式

更多信息

- 基础功能手册，“若干 NC/PLC 接口信号与功能”>“功能”>“电机/驱动数据组切换”章节

12.3.6.7 rapidStop

PLC → NC	<Axis>.drive.out.rapidStop
无斜坡函数发生器停止（快速停止，额定转速为 0）	

说明

仅限 NC 和驱动之间的循环接口在"611U 兼容模式"下运行时。

在驱动中，必须通过驱动参数 p0922 和 p2038 设置一个与接口模式"SIMODRIVE 611 universal"相兼容的 PROFIDrive 报文。

数据类型和更新

BOOL，循环

值 TRUE

为驱动请求了转速设定值 0 的快速停止。此时驱动会在无斜坡函数发生器斜坡的状态下制动至停止（再生制动）。

针对驱动中触发的快速停止，通过以下变量进行反馈：

<Axis>.drive.in.rapidStopActive == TRUE （斜坡函数发生器锁定）

值 FALSE

未为驱动请求转速设定值 0 的快速停止。并由 NC 控制。

关联

- <Axis>.drive.in.rapidStopActive
  斜坡函数发生器锁定（快速停止，转速设定值为 0）
- 驱动参数 p0922
  IF1 PROFIdrive PZD 报文选择
- 驱动参数 p2038
  IF1 PROFIdrive STW/ZSW 接口模式

更多信息

- 驱动：SINAMICS S120/S150 参数手册

12.3.6.8 releaseHoldingBrake

PLC → NC	<Axis>.drive.out.releaseHoldingBrake
松开电机的抱闸	

数据类型和更新

BOOL, 循环

值 TRUE

打开电机抱闸。

值 FALSE

不打开电机抱闸。

12.3.6.9 requestDriveCtrl

PLC → NC	<Axis>.drive.out.requestDriveCtrl
请求进行驱动监测	

说明

- 若在多个轴中同时存在接收驱动控制的请求，则不发生切换。在当前具有驱动控制的轴中保留驱动控制。
- 在控制系统启动中缺少接收请求的情况下，控制系统会将驱动控制分配给机床数据（MD30110 $MA_CTRLOUT_MODULE_NR[<轴>]）找到的第一个具有相同逻辑驱动地址的轴。

数据类型与更新

BOOL，循环

值 TRUE

已请求接收驱动控制。

值 FALSE

未请求接收驱动控制。

关联

- <Axis>.drive.in.driveCtrlActive
  额定转速切换：轴已应用驱动控制

更多信息

- 轴和主轴功能手册；“额定值转换”章节

12.3.7 <Axis>.interchange.in

12.3.7.1 assignedChanNumber

NC → PLC	<Axis>.interchange.in.assignedChanNumber
已分配给 NC 轴的通道编号（通道编号或 0：中性轴）	

数据类型和更新

USInt，循环

值

1 ... 10：与轴/主轴对应的通道的编号。

关联

- <Axis>.spindle.in.sFuncValue
  指定的 S 功能值 (REAL)：以每分钟转数 (RPM) 为单位的主轴转速，或以每分钟转数为单位的恒定切削速度，取决于所选的操作模式（G97 或 G96）
- MD20070 $MC_AXCONF_MACHAX_USED
  通道中生效的机床轴编号
- MD30550 $MA_AXCONF_ASSIGN_MASTER_CHAN
  用于轴交换的通道清除设置

更多信息

- 功能手册之基本功能；章节“跨通道的轴交换”

12.3.7.2 assignedToPlc

NC → PLC	<Axis>.interchange.in.assignedToPlc
PLC 轴/主轴	

说明

PLC 轴/主轴

可借助 PLC 用户程序通过该变量读取轴的有关轴/主轴交换的当前状态。

数据类型和更新

BOOL，循环

值 TRUE

轴是“PLC 轴/主轴”

值 FALSE

轴不是“PLC 轴/主轴”

关联

- <Axis>.interchange.out.assignToPlc
  请求 PLC 轴
- <Axis>.interchange.out.requireSwitchOver
  脉冲沿切换 0 → 1：激活配置
- <Axis>.plcCtrl.out.assignToChanSelection
  为 NC 轴分配通道（通道编号，或 0：中性轴）
- MD20070 $MC_AXCONF_MACHAX_USED
  通道内有效的机床轴号
- MD30550 $MA_AXCONF_ASSIGN_MASTER_CHAN
  用于轴交换的通道清除设置

更多信息

- 功能手册之基本功能；章节“跨通道的轴交换”

12.3.7.3 neutralAxis

NC → PLC	<Axis>.interchange.in.neutralAxis
中性轴/主轴	

数据类型和更新

BOOL，循环

值 TRUE

轴是“中立轴/主轴”

值 FALSE

轴不是“中立轴/主轴”

关联：

- <Axis>.spindle.in.sFuncValue
  指定的 S 功能值 (REAL)：以每分钟转数 (RPM) 为单位的主轴转速，或以每分钟转数为单位的恒定切削速度，取决于所选的操作模式（G97 或 G96）
- MD20070 $MC_AXCONF_MACHAX_USED
  通道中生效的机床轴编号
- MD30550 $MA_AXCONF_ASSIGN_MASTER_CHAN
  用于轴交换的通道清除设置

详细信息

- 功能手册之基本功能；章节“跨通道的轴交换”

12.3.7.4 possible

NC → PLC	<Axis>.interchange.in.possible
可进行轴/主轴交换	

数据类型和更新

BOOL，循环

值 TRUE

可进行轴交换。

值 FALSE

无法进行轴交换。

关联：

- <Axis>.spindle.in.sFuncValue
  指定的 S 功能值 (REAL)：以每分钟转数 (RPM) 为单位的主轴转速，或以每分钟转数为单位的恒定切削速度，取决于所选的操作模式（G97 或 G96）
- MD20070 $MC_AXCONF_MACHAX_USED
  通道中生效的机床轴编号
- MD30550 $MA_AXCONF_ASSIGN_MASTER_CHAN
  用于轴交换的通道清除设置

详细信息

- 功能手册之基本功能；章节“跨通道的轴交换”

12.3.7.5 requested

NC → PLC	<Axis>.interchange.in.requested
已请求轴/主轴交换	

数据类型和更新

BOOL，循环

值 TRUE

PLC 请求了新的轴类型。

值 FALSE

未请求新的轴类型。

关联：

- <Axis>.spindle.in.sFuncValue
  指定的 S 功能值 (REAL)：以每分钟转数 (RPM) 为单位的主轴转速，或以每分钟转数为单位的恒定切削速度，取决于所选的操作模式（G97 或 G96）
- MD20070 $MC_AXCONF_MACHAX_USED
  通道中生效的机床轴编号
- MD30550 $MA_AXCONF_ASSIGN_MASTER_CHAN
  用于轴交换的通道清除设置

详细信息

- 功能手册之基本功能；章节“跨通道的轴交换”

12.3.8 <Axis>.interchange.out

12.3.8.1 assignToChanSelection

PLC → NC	<Axis>.interchange.out.assignToChanSelection
为 NC 轴分配通道（通道编号，或 0：中性轴）	

说明

请求跨通道取轴/主轴

可由 PLC 用户程序通过 DBB8 请求跨通道取轴/主轴。

- <Axis>.interchange.out.assignToPlc
  从一个 NC 通道取至 PLC
- <Axis>.interchange.out.assignToPlc
  从 PLC 取至 NC 通道
- <Axis>.plcCtrl.out.assignToChanSelection
  从一个 NC 通道取至另一个 NC 通道

数据类型和更新

USInt，循环

值

轴/主轴需要分配至的通道的编号（二进制编码）。

示例：应将轴分配给通道 B。

表格 12-1　通道分配

ChanD	ChanC	ChanB	ChanA
位 3	位 2	位 1	位 0
0	0	1	0

关联

- <Axis>.interchange.in.assignedChanNumber
  已分配给 NC 轴的通道编号（通道编号或 0：中性轴）
- <Axis>.interchange.in.requested
  已请求轴/主轴交换
- <Axis>.interchange.in.possible
  可进行轴/主轴交换
- <Axis>.interchange.in.neutralAxis
  中性轴/主轴
- <Axis>.plcCtrl.in.spindleCrtlActive
  轴是"PLC 轴/主轴"
- MD20070 $MC_AXCONF_MACHAX_USED
  Maschinenachsnummer gültig im Kanal•
- MD30550 $MA_AXCONF_ASSIGN_MASTER_CHAN
  用于轴交换的通道清除设置

更多信息

- 功能手册之基本功能；章节"跨通道的轴交换"

12.3.8.2 assignToPlc

PLC → NC	<Axis>.interchange.out.assignToPlc
请求 PLC 轴	

说明

请求跨通道取轴/主轴

可由 PLC 用户程序请求轴/主轴交换。

数据类型和更新

BOOL，循环

值 TRUE

请求轴类型"PLC 轴"

值 FALSE

不请求"PLC 轴"轴类型

关联

- <Axis>.interchange.in.assignedChanNumber
  已分配给 NC 轴的通道编号（通道编号或 0：中性轴）
- <Axis>.interchange.in.requested
  已请求轴/主轴交换
- <Axis>.interchange.in.possible
  可进行轴/主轴交换
- <Axis>.interchange.in.neutralAxis
  中性轴/主轴
- <Axis>.interchange.in.assignedToPlc
  PLC 轴/主轴
- MD20070 $MC_AXCONF_MACHAX_USED
  通道内有效的机床轴号
- MD30550 $MA_AXCONF_ASSIGN_MASTER_CHAN
  用于轴交换的通道清除设置

更多信息

- 功能手册之基本功能；章节"跨通道的轴交换"

12.3.8.3 requireSwitchOver

PLC → NC	<Axis>.interchange.out.requireSwitchOver
脉冲沿切换 0 → 1：激活配置	

数据类型和更新

BOOL，循环

脉冲沿切换 0 → 1

PLC 的请求

脉冲沿切换 1 → 0

通过 NC 应答

关联

- <Axis>.interchange.in.requested
  已请求轴/主轴交换
- <Axis>.interchange.in.possible
  可进行轴/主轴交换
- <Axis>.interchange.in.neutralAxis
  中性轴/主轴
- <Axis>.plcCtrl.in.spindleCrtlActive
  轴是“PLC 轴/主轴”
- <Axis>.interchange.in.assignedChanNumber
  已分配给 NC 轴的通道编号（通道编号或 0：中性轴）
- MD20070 $MC_AXCONF_MACHAX_USED
  通道中生效的机床轴编号
- MD30550 $MA_AXCONF_ASSIGN_MASTER_CHAN
  用于轴交换的通道清除设置

更多信息

- 功能手册之基本功能；章节“跨通道的轴交换”

12.3.8.4 reset

PLC → NC	<Axis>.plcCtrl.out.requestPlcCtrl
请求通过适用于自主单轴操作的 PLC 进行控制	

说明

NC 的响应：

- 轴转换至“单轴复位中”状态：
  $AA_SNGLAX_STAT == TRUE
- 停止的顺序运行终止。
- 轴的内部状态复位。
- 复位生效的轴专用机床数据生效。
- 确认执行：
  - <Axis>.plcCtrl.in.resetDone（由 PLC 控制的轴：已执行轴复位）== TRUE
  - <Axis>.plcCtrl.in.stopped（由 PLC 控制的轴：轴已停止）== FALSE
- 结果：轴处于复位状态。

说明

采用通道复位时，对于由 PLC 控制的轴而言，轴专用机床数据不生效。

数据类型和更新

BOOL，循环

脉冲沿切换 0 → 1

PLC 控制的轴：已请求复位。

脉冲沿切换 1 → 0

无作用。

关联

- <Axis>.plcCtrl.out.requestPlcCtrl
  请求通过适用于自主单轴操作的 PLC 进行控制
- <Axis>.plcCtrl.in.plcCtrlActive
  由 PLC 控制的轴：轴由 PLC 控制，适用于自主单轴操作
- <Axis>.plcCtrl.in.stopped
  由 PLC 控制的轴：轴已停止
- 系统变量：$AA_SNGLAX_STAT
  单个轴的状态
- OPI 变量：aaSnglAxStat

更多信息

- 功能手册之轴和主轴；章节“定位轴”

12.3.9 <Axis>.spindle.in

12.3.9.1 constantCuttingSpeedActive

NC → PLC	<Axis>.spindle.in.constantCuttingSpeedActive
恒定切削速度：主轴转速受控，用以补偿工件直径变化	

说明

工件直径变化在切削速度考虑范围内。

数据类型与更新

BOOL，循环

值 TRUE

恒定切削速度处于活动状态

值 FALSE

恒定切削速度未激活

12.3.9.2 ctrlModeActive

NC → PLC	<Axis>.spindle.in.ctrlModeActive
主轴处于转速控制模式下（通过 M03/M04/M05 等初始化）	

说明

执行下列功能时，主轴处于控制模式下：

- 主轴旋转方向设定 M3/M4 或主轴停止 M5
- M41 - M45，或者齿轮箱自动换挡 M40

数据类型与更新

BOOL，循环

值 TRUE

控制模式生效。

值 FALSE

控制模式未生效。

关联

- <Axis>.spindle.in.posModeActive
  主轴在定位运行模式下
- <Axis>.spindle.in.oscillModeActive
  主轴在摆动运行模式下

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.9.3 cwRotActive

NC → PLC	<Axis>.spindle.in.cwRotActive
实际旋转方向： 0 - 逆时针 1 - 顺时针	

说明

实际旋转方向通过主轴的位置测量编码器推导得出。

说明

在以下情形下，此接口信号不相关：

- <Axis>.basic.in.isStationary == TRUE（轴/主轴停止）
- 主轴无位置测量编码器

数据类型与更新

BOOL，循环

值 TRUE

当前旋转方向为向右转

值 FALSE

当前旋转方向为向左转

关联

- <Axis>.basic.in.isStationary
  轴达到静止转速范围 (n<n_min)

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.9.4 gearChangeoverRequested

NC → PLC	<Axis>.spindle.in.gearChangeoverRequested
已请求齿轮箱换挡	

说明

仅在满足以下条件的情况下，才切换至新的设定齿轮档：设定齿轮档 <> 实际齿轮档

数据类型与更新

BOOL，循环

值 0 → 1

已请求将齿轮档切换至设定齿轮档。

值 1 → 0

无作用

关联

- <Axis>.spindle.out.gearStageSelection
  选择齿轮箱实际挡位 (1 - 5)
- <Axis>.spindle.in.requestedGearStageNumber
  所请求齿轮箱挡位 (1 - 5) 的编号

更多信息

- 轴和主轴功能手册；“主轴”>“可组态的齿轮箱调整”>“主轴齿轮挡位和齿轮档切换”章节

12.3.9.5 inPos

NC → PLC	<Axis>.spindle.in.inPos
主轴已就位	

说明

输出接口信号的前提条件：

- <Axis>.basic.in.posFineReached == TRUE（精准停） **AND**
- 到达编写的设定位置

说明

- 仅在定位运行 (<Axis>.spindle.in.posModeActive == TRUE) 下处理此接口信号，例如：
  - NC 程序：SPOS、SPOSA 和 M19
  - 同步动作：SPOS 和 M19
  - PLC 用户程序：通过"SINU_PositionSpindle"或 <Axis>.spindle.out.startPosOperation 定位
- 若主轴已位于编写的位置上，则此接口信号保持置位。

数据类型和更新

BOOL，循环

值 TRUE

主轴就位。

值 FALSE

主轴未就位。

关联：

- <Axis>.spindle.out.startPosOperation
  主轴启动定位
- <Axis>.basic.in.posFineReached
  已在“精准停”限值内到达编程位置 (MD36150 $MA_STOP_LIMIT_FINE)
- <Axis>.spindle.in.posModeActive
  主轴处于定位运行模式
- SPOS
- SPOSA
- M19

详细信息

功能手册之轴和主轴；章节“主轴”>

- “运行方式”>“定位模式”>“换刀信号“主轴就位””
- “编程”>“通过 PLC 接口进行主轴运动”

12.3.9.6 mFuncNumber

NC → PLC	<Axis>.spindle.in.mFuncNumber
指定的 M 功能 (INT)：考虑的功能： 3/4 - 主轴左/右转动 5 - 主轴停止 19 - 主轴在指定角度停止 70 - 主轴作为轴	

说明

主轴的最后一个有效 M 功能来自 NC 程序。

数据类型和更新

UInt，任务控制

值

3：M3（主轴顺时针旋转）

4：M4（主轴逆时针旋转）

5：M5（主轴停止）

19：M19（主轴定位至 SD43240 中记录的位置）

70：M70（切换至进给轴模式）

关联

- <Chan>.auxMFuncs[x].in.hasChanged
  M 功能发生变化；信号显示相应的值有效，可用于一个 PLC 节拍
- <Chan>.auxMFuncs[x].in.notDecoded
  M 功能未包含在解码列表中，可用于一个 PLC 节拍
- <Chan>.auxMFuncs[x].in.extendedAddress
  M 功能扩展地址

12.3.9.7 oscillModeActive

NC → PLC	<Axis>.spindle.in.oscillModeActive
主轴在摆动运行模式下	

说明

在齿轮档切换时，主轴自动切换至往复模式。

数据类型与更新

BOOL，循环

值 TRUE

往复模式生效。

值 FALSE

往复模式未生效。

关联

- <Axis>.spindle.in.gearChangeoverRequested
  已请求齿轮箱换挡
- <Axis>.spindle.in.posModeActive
  主轴在定位运行模式下
- <Axis>.spindle.in.ctrlModeActive
  主轴处于转速控制模式下（通过 M03/M04/M05 等初始化）

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.9.8 posModeActive

NC → PLC	<Axis>.spindle.in.posModeActive
主轴在定位运行模式下	

数据类型与更新

BOOL，循环

值 TRUE

定位模式（SPOS 或 SPOSA）生效。

值 FALSE

定位模式（SPOS 或 SPOSA）未生效。

关联

- <Axis>.spindle.in.oscillModeActive
  主轴在摆动运行模式下
- <Axis>.spindle.in.ctrlModeActive
  主轴处于转速控制模式下（通过 M03/M04/M05 等初始化）

更多信息

- 轴和主轴功能手册；“主轴”>“可组态的齿轮箱调整”>“在固定位置进行齿轮箱换挡”章节

12.3.9.9 requestedGearStageNumber

NC → PLC	<Axis>.spindle.in.requestedGearStageNumber
所请求齿轮箱挡位 (1 - 5) 的编号	

说明

由 NC 请求的、需要在机床上切换至的齿轮档。

数据类型和更新

UInt，循环

值

所请求齿轮箱挡位 (1 - 5) 的编号

关联

- <Axis>.spindle.out.gearStageSelection
  选择齿轮箱实际挡位 (1 - 5)
- <Axis>.spindle.out.gearChangeoverDone
  完成齿轮箱换挡

- <Axis>.spindle.out.oscillEnable
  使能主轴摆动运动
- <Axis>.spindle.in.gearChangeoverRequested
  已请求齿轮箱换挡

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.9.10 rigidTappingActive

NC → PLC	<Axis>.spindle.in.rigidTappingActive
刚性攻丝生效。	

说明

如果“无补偿卡盘攻丝”(G331/G332) 处于活动状态，则主轴在系统内部切换至位置闭环控制下的轴运行方式。

注意
无补偿卡盘导致螺纹损毁 刚性攻丝期间置位以下接口信号会导致螺纹损毁： • <ModeGroup>.basic.out.reset（请求复位运行方式组）= TRUE • <Chan>.basic.out.reset（通道复位）= TRUE • <Axis>.basic.out.enableCtrl（控制器使能）= FALSE • <Axis>.basic.out.feedStopSpindleStop （进给停止）= TRUE

数据类型与更新

BOOL，循环

值 TRUE

刚性攻丝生效。

值 FALSE

刚性攻丝未生效。

关联

- <ModeGroup>.basic.out.reset
  请求复位运行方式组
- <Chan>.basic.out.reset
  已请求通道复位
- <Axis>.basic.out.enableCtrl
  请求控制器使能
- <Axis>.basic.out.feedStopSpindleStop
  进给停止/主轴停止处于活动状态

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.9.11 sFuncValue

NC → PLC	<Axis>.spindle.in.sFuncValue
指定的 S 功能值 (REAL)：以每分钟转数 (RPM) 为单位的主轴转速，或以每分钟转数为单位的恒定切削速度，取决于所选的操作模式（G97 或 G96）	

说明

主轴的最后一个有效 S 功能来自 NC 程序。

数据类型和更新

REAL，任务控制

值

下列 S 功能根据具体模式（G97 或 G96）进行输出：

- S...作为主轴转速，单位 1/min（编程的值）
- S...作为恒定切削速度，单位 m/min 或 ft/min

下列 S 功能不输出：

- S...作为编程设置的 G25 主轴转速限制
- S...作为编程设置的 G26 主轴转速限制
- 控制器中未定义主轴时，S...作为主轴转速，单位 1/min
- S...作为停留时间，单位为主轴转数

关联

- <Chan>.auxSFuncs[x].in.hasChanged
  S 功能发生变化；信号显示相应的值有效
- <Chan>.auxSFuncs[x].in.fastAckActive
  S 功能快速应答处于活动状态
- <Chan>.auxSFuncs[x].in.extendedAddress
  S 功能扩展地址

12.3.9.12 speedLimitationExceeded

NC → PLC	<Axis>.spindle.in.speedLimitationExceeded
实际转速超过规定的极限值 (MD35100 $MA_SPIND_VELO_LIMIT) 和公差 (MD35150 $MA_SPIND_DES_VELO_TOL)	

说明

若符合以下条件，便超出转速极限值

实际转速 >（MD35100 $MA_SPIND_VELO_ LIMIT + MD35150 $MA_SPIND_DES_VELO_TOL）

数据类型与更新

BOOL，循环

值 TRUE

超出转速极限值。

值 FALSE

未超出转速极限值。

关联

- MD35150 $MA_SPIND_DES_VELO_TOL
  主轴转速公差
- MD35100 $MA_SPIND_VELO_LIMIT
  最大主轴转速

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.9.13 speedSetpointAboveMax

NC → PLC	<Axis>.spindle.in.speedSetpointAboveMax
额定转速超过通过 G26 编程的最大转速极限值 (SD43220 $SA_SPIND_MAX_VELO_G26)	

说明

设定转速被 NC 自动限制，因为其超出生效的最大限值：

- MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT
- MD35100 $MA_SPIND_VELO_LIMIT
- <Axis>.basic.out.limitSpeed
- `G26`
- `LIMS`
- `VELOLIM`

说明

- PLC 用户程序中可能的响应为锁定轨迹进给：
  <Chan>.basic.out.disableFeed = TRUE（激活通道中的进给锁定）
- 更新接口信号 <Axis>.spindle.in.speedSetpointReached（达到额定转速公差范围）。

Safety Integrated

根据当前活动的安全速度挡位 SG1 - SGn，考虑下列机床数据：

- MD36933 $MA_SAFE_DES_VELO_LIMIT

示例：

所有缺省限值均大于 1500 rpm。

- SG1 生效
- MD36933 $MA_SAFE_DES_VELO_LIMIT[<SG1>] = 90%

编程：M3 S1500

转速设定值限定为 1000 rpm (MD36932 * MD36933) ⇒
<Axis>.spindle.in.speedSetpointAboveMax = TRUE

数据类型和更新

BOOL，循环

值 TRUE

额定转速超过主轴转速上限。

检查 SD43220 $SA_SPIND_MAX_VELO_G26。

值 FALSE

额定转速未超过主轴转速上限。

关联

- <Chan>.basic.out.disableFeed
  激活通道中的进给锁定
- <Axis>.basic.out.feedStopSpindleStop
  进给停止/主轴停止处于活动状态
- <Axis>.spindle.in.speedSetpointReached
  达到额定转速公差范围 (MD35150 $MA_SPIND_DES_VELO_TOL)
- <Axis>.basic.out.limitSpeed
  限制主轴速度/转速 (MD35160 $MA_SPIND_EXTERN_VELO_LIMIT)
- SD43220 $SA_SPIND_MAX_VELO_G26
- MD35100 $MA_SPIND_VELO_LIMIT
  最大主轴转速
- MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT
  齿轮箱挡位最大转速
- MD36933 $MA_SAFE_DES_VELO_LIMIT
  SG 额定速度限制
- `G26`
  主轴转速上限
- `LIMS`
  G96/G961/G97 处于活动状态时对主动主轴的转速限制
- `VELOLIM`
  转速控制运行下编程设置的主轴转速限制

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.9.14 speedSetpointBelowMin

NC → PLC	<Axis>.spindle.in.speedSetpointBelowMin
额定转速低于通过 G25 编程的最小转速极限值 (SD43210 $SA_SPIND_MIN_VELO_G25) 或为各齿轮箱挡位设置的最小转速极限值 (MD35140 $MA_GEAR_STEP_MIN_VELO_LIMIT)	

说明

- PLC 用户程序中可能的响应为锁定轨迹进给：
  <Chan>.basic.out.disableFeed = TRUE（激活通道中的进给锁定）
- 更新接口信号 <Axis>.spindle.in.speedSetpointReached（达到额定转速公差范围）。

数据类型与更新

BOOL，循环

值 TRUE

额定转速低于主轴转速下限。

检查 SD43210 $SA_SPIND_MIN_VELO_G25 或更改变速箱挡位的最小转速。

值 FALSE

额定转速不低于主轴转速下限。

关联

- 轴和主轴功能手册；“主轴”章节

更多信息

- <Chan>.basic.out.disableFeed
  激活通道中的进给锁定
- <Axis>.basic.out.feedStopSpindleStop
  进给停止/主轴停止处于活动状态
- MD35120 $MA_GEAR_STEP_MIN_VELO
  自动选择齿轮箱挡位 M40 的最小转速
- MD35140 $MA_GEAR_STEP_MIN_VELO_LIMIT
  齿轮箱挡位最小转速
- SD43210 $SA_SPIND_MIN_VELO_G25
- G25
  主轴转速下限

12.3.9.15 speedSetpointReached

NC → PLC	<Axis>.spindle.in.speedSetpointReached
达到额定转速公差范围 (MD35150 $MA_SPIND_DES_VELO_TOL)	

说明

若满足以下条件，则主轴转速位于设定范围内：

| 实际转速 - 设定转速 | < MD35150 $MA_SPIND_DES_VELO_TOL

说明

在主轴的加速或制动阶段，主轴转速不位于设定范围内是正常情况。

数据类型与更新

BOOL，循环

值 TRUE

主轴转速位于设定范围内。

值 FALSE

主轴转速不位于设定范围内。

关联

- <Chan>.basic.out.disableFeed
  激活通道中的进给锁定
- <Axis>.basic.out.feedStopSpindleStop
  进给停止/主轴停止处于活动状态
- <Axis>.spindle.in.speedSetpointAboveMax
  额定转速超过通过 G26 编程的最大转速极限值 (SD43220 $SA_SPIND_MAX_VELO_G26)
- <Axis>.spindle.in.speedSetpointBelowMin
  额定转速低于通过 G25 编程的最小转速极限值 (SD43210 $SA_SPIND_MIN_VELO_G25)
  或为各齿轮箱挡位设置的最小转速极限值 (MD35140 $MA_GEAR_STEP_MIN_VELO_LIMIT)

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.9.16 syncModeActive

NC → PLC	<Axis>.spindle.in.syncModeActive
主轴在同步运行模式下	

说明

在同步模式中，副主轴根据传动比跟随主主轴的运动，且系统会对粗同步及精同步进行监控。

在关闭耦合（撤销同步模式）后，副主轴进入“控制模式”。

数据类型与更新

BOOL，循环

值 TRUE

主轴处于“同步运行”方式中。

说明

此信号仅针对作为副主轴生效的机床轴置位：
<Axis>.coupling.in.followingSpindleActive（通用耦合：主轴是从动主轴）== TRUE

值 FALSE

主轴不作为“同步运行”中的副主轴运行。

关联

- <Axis>.coupling.in.syncFineReached
  通用耦合：（位置或转速）实际值和设定值之差达到“精公差范围”(MD37210 $MA_COUPLE_POS_TOL_FINE, MD37230 $MA_COUPLE_VELO_TOL_FINE)
- <Axis>.coupling.in.syncCoarseReached
  通用耦合：（位置或转速）实际值和设定值之差在粗公差范围内 (MD37200 $MA_COUPLE_POS_TOL_COARSE, MD37220 $MA_COUPLE_VELO_TOL_COARSE)
- <Axis>.coupling.in.followingSpindleActive
  通用耦合：主轴是从动主轴

更多信息

- 轴和主轴功能手册；“同步主轴”章节

12.3.9.17 toolWithDynLimitationActive

NC → PLC	<Axis>.spindle.in.toolWithDynLimitationActive
当前夹紧的刀具具有动态极限值	

数据类型与更新

BOOL，循环

值 TRUE

当前刀具具有动态限制。

值 FALSE

当前刀具不具有动态限制。

12.3.10 <Axis>.spindle.in.smi24

12.3.10.1 sensorS1Present

NC → PLC	<Axis>.spindle.in.smi24.sensorS1Present
SMI24：存在传感器 S1（夹紧系统模拟测量）	

说明

仅与配备 SMI 24 的主轴（Weiss 主轴）相关

数据类型和更新

BOOL，循环

值 TRUE

传感器 S1 存在。

值 FALSE

传感器 S1 不存在。

关联

- <Axis>.spindle.in.smi24.sensorS1StateNumber
  SMI24：传感器 S1（夹紧系统模拟测量）量化状态 (0 - 11)
- <Axis>.spindle.in.smi24.sensorS1Value
  SMI24：传感器 S1（夹紧系统模拟测量）原始值

更多信息

- 轴和主轴功能手册；“主轴”章节
- Weiss 操作说明

12.3.10.2 sensorS1StateNumber

NC → PLC	<Axis>.spindle.in.smi24.sensorS1StateNumber
SMI24：传感器 S1（夹紧系统模拟测量）量化状态 (0 - 11)	

说明

仅与配备 SMI 24 的主轴（Weiss 主轴）相关

说明

传感器 S1 根据夹紧装置的位置提供一个模拟电压值。为了简化对夹紧状态的分析，传感器模块 SMI 24 会将该模拟电压值转换为一个状态值。

该状态值对应特定的电压范围。电压范围可通过以下参数设置：驱动参数 p5041[0 - 5]。

数据类型和更新

UInt，循环

值

下表介绍状态值决定的夹紧状态。

状态值	夹紧状态
0	传感器 S1 不存在，或状态值无效
1	状态初始化中
2	已松开，发出消息（故障状态）
3	已松开
4	夹紧，带刀具

状态值	夹紧状态
5	松开，带刀具
6	松开，无刀具
7	已夹紧，带刀具，且 S4 == FALSE
8	已夹紧，带刀具，且 S4 == TRUE
9	夹紧，无刀具
10	已夹紧，无刀具

关联

- <Axis>.spindle.in.smi24.sensorS1Value
  SMI24：传感器 S1（夹紧系统模拟测量）原始值

更多信息

- 轴和主轴功能手册；“主轴”章节
- Weiss 操作说明

12.3.10.3 sensorS1Value

NC → PLC	<Axis>.spindle.in.smi24.sensorS1Value
SMI24：传感器 S1（夹紧系统模拟测量）原始值	

说明

仅与配备 SMI 24 的主轴（Weiss 主轴）相关

说明

传感器 S1 提供一个模拟电压值：0 - 10 V。夹紧状态模拟值映射至：0 - 10 000 个增量，分辨率 1 mV

说明

SIMATIC S7 输入模块：0 - 27 648 个增量，分辨率 0.36 mV
转换至配备 SMI 24 主轴时的匹配系数：2.7648

数据类型和更新

UInt，循环

值

夹紧系统的模拟测量值

关联

- <Axis>.spindle.in.smi24.sensorS1StateNumber
  SMI24：传感器 S1（夹紧系统模拟测量）量化状态 (0 - 11)
- 驱动参数：p5041[0 - 5], p5043[0 - 6]

更多信息

- 轴和主轴功能手册；“主轴”章节
- Weiss 操作说明

12.3.10.4 sensorS4Actuated

NC → PLC	<Axis>.spindle.in.smi24.sensorS4Actuated
SMI24：传感器 S4（活塞在终位）处于活动状态	

说明

仅与配备 SMI 24 的主轴（Weiss 主轴）相关

数据类型和更新

BOOL，循环

值 TRUE

活塞位于终端位置，即活塞未占用

值 FALSE

活塞未处于终端位置

关联

- <Axis>.spindle.in.smi24.sensorS4Present
  SMI24：存在传感器 S4（活塞在终位）

更多信息

- 轴和主轴功能手册；“主轴”章节
- Weiss 操作说明

12.3.10.5 sensorS4Present

NC → PLC	<Axis>.spindle.in.smi24.sensorS4Present
SMI24：存在传感器 S4（活塞在终位）	

说明

仅与配备 SMI 24 的主轴（Weiss 主轴）相关

数据类型和更新

BOOL，循环

值 TRUE

传感器 S4 存在。

值 FALSE

传感器 S4 不存在。

关联

- <Axis>.spindle.in.smi24.sensorS4Actuated
  SMI24：传感器 S4（活塞在终位）处于活动状态

更多信息

- 轴和主轴功能手册；“主轴”章节
- Weiss 操作说明

12.3.10.6 sensorS5Actuated

NC → PLC	<Axis>.spindle.in.smi24.sensorS5Actuated
SMI24：传感器 S5（电机轴就位）处于活动状态	

说明

仅与配备 SMI 24 的主轴（Weiss 主轴）相关

数据类型和更新

BOOL，循环

值 TRUE

电机轴已就位（前提条件：主轴停止）

值 FALSE

电机轴未就位

关联

- <Axis>.spindle.in.smi24.sensorS5Present
  SMI24：存在传感器 S5（电机轴就位）

更多信息

- 轴和主轴功能手册；“主轴”章节
- Weiss 操作说明

12.3.10.7 sensorS5Present

NC → PLC	<Axis>.spindle.in.smi24.sensorS5Present
SMI24：存在传感器 S5（电机轴就位）	

说明

仅与配备 SMI 24 的主轴（Weiss 主轴）相关

数据类型和更新

BOOL，循环

值 TRUE

传感器 S5 存在。

值 FALSE

传感器 S5 不存在。

关联

- <Axis>.spindle.in.smi24.sensorS5Actuated
  SMI24：传感器 S5（电机轴就位）处于活动状态

更多信息

- 轴和主轴功能手册；“主轴”章节
- Weiss 操作说明

12.3.10.8 sensorS6Present

NC → PLC	<Axis>.spindle.in.smi24.sensorS6Present
SMI24：存在传感器 S6（温度）	

数据类型与更新

BOOL

值 TRUE

传感器 S1 存在。

值 FALSE

传感器 S1 不存在。

12.3.10.9 sensorsPresent

NC → PLC	<Axis>.spindle.in.smi24.sensorsPresent
SMI24：存在传感器	

说明

仅与配备 SMI 24 的主轴（Weiss 主轴）相关

数据类型和更新

BOOL，循环

值 TRUE

配备 SMI 24 的主轴所需的传感器系统存在。

值 FALSE

配备 SMI 24 的主轴所需的传感器系统不存在。

关联

- <Axis>.spindle.in.smi24.sensorS1Present
  SMI24：存在传感器 S1（夹紧系统模拟测量）
- <Axis>.spindle.in.smi24.sensorS4Present
  SMI24：存在传感器 S4（活塞在终位）

- <Axis>.spindle.in.smi24.sensorS5Present
  SMI24：存在传感器 S5（电机轴就位）
- <Axis>.spindle.in.smi24.sensorS6Present
  SMI24：存在传感器 S6（温度）

更多信息

- 轴和主轴功能手册；“主轴”章节
- Weiss 操作说明

12.3.10.10 speedLimited

NC → PLC	<Axis>.spindle.in.smi24.speedLimited
SMI24：模拟测量量化 (r5001)，转速受限 (p5043)	

说明

SMI24：模拟测量量化 (r5001)，转速受限 (p5043)

数据类型与更新

BOOL，循环

值 TRUE

模拟测量量化 (r5001)，速度受限 (p5043)

值 FALSE

模拟测量不量化，速度不受限

12.3.11 <Axis>.spindle.out

12.3.11.1 autoGearStageChange

PLC → NC	<Axis>.spindle.out.autoGearStageChange
激活齿轮箱自动换挡	

说明

激活自动选择齿轮箱挡位

技术 - 通过 PLC 接口进行主轴移动：选择齿轮档

数据类型与更新

BOOL，循环

值 TRUE

激活自动选择齿轮箱挡位

值 FALSE

不激活自动选择齿轮箱挡位

12.3.11.2 deleteSValue

PLC → NC	<Axis>.spindle.out.deleteSValue
删除 S 值：主轴按照规定的制动斜率制动至静止状态	

说明

- **控制模式：**
  主轴通过为其设置的制动斜坡制动至静止状态。
- **往复模式、轴模式、定位模式：**
  删除当前 S 值对当前功能无影响。若需重新切换回控制模式，必须编写一个新的 S 值。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

已请求删除 S 值。

脉冲沿切换 1 → 0

无作用

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.11.3 disableSpeedMonitor

PLC → NC	<Axis>.spindle.out.disableSpeedMonitor
在齿轮箱换挡期间锁定转速监控	

说明

齿轮箱换挡时禁用转速监控

数据类型与更新

BOOL

值 TRUE

齿轮箱换挡时无转速监控

值 FALSE

齿轮箱换挡时进行转速监控

12.3.11.4 gearChangeoverDone

PLC → NC	<Axis>.spindle.out.gearChangeoverDone
完成齿轮箱换挡	

说明

若在机床上啮入新的齿轮档，则由 PLC 用户程序置位下列接口信号，作为发送至 NC 的反馈信号：

- <Axis>.spindle.out.gearStageSelection（选择齿轮箱实际挡位 (1 - 5)）
- <Axis>.spindle.out.gearChangeoverDone = 0 → 1（完成齿轮箱换挡）

NC 向 PLC 反馈：<Axis>.spindle.in.gearChangeoverRequested = 0（已请求齿轮箱换挡）

PLC 向 NC 反馈：<Axis>.spindle.out.gearChangeoverDone = 0（已完成齿轮箱换挡）

这样一来，控制系统内部的齿轮档切换便完成。

说明

- 在除往复模式以外的所有主轴运行方式下，此接口信号不相关。
- 若由 PLC 用户程序反馈给 NC 的实际齿轮档不同于 NC 从 PLC 请求的设定齿轮档，则仍将齿轮档切换视作成功完成，且反馈的实际齿轮档在 NC 中激活。

数据类型与更新

BOOL，循环

值 TRUE

完成齿轮箱换挡。

值 FALSE

无作用

关联

- <Axis>.spindle.out.gearStageSelection
  选择齿轮箱实际挡位 (1 - 5)
- <Axis>.spindle.in.requestedGearStageNumber
  所请求齿轮箱挡位 (1 - 5) 的编号
- <Axis>.spindle.in.gearChangeoverRequested
  已请求齿轮箱换挡
- <Axis>.spindle.out.oscillEnable
  使能主轴摆动运动

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.11.5 gearStageSelection

PLC → NC	<Axis>.spindle.out.gearStageSelection
选择齿轮箱实际挡位 (1 - 5)	

说明

若在机床上啮入新的齿轮档，则由 PLC 用户程序置位下列接口信号，作为发送至 NC 的反馈信号：

- <Axis>.spindle.out.gearStageSelection 选择齿轮箱实际挡位 (1 - 5)
- <Axis>.spindle.out.gearChangeoverDone 完成齿轮箱换挡

NC 由此获知已成功啮入新的齿轮档。对应的参数组随即在 NC 中生效。

说明

若由 PLC 用户程序反馈给 NC 的实际齿轮档不同于 NC 从 PLC 请求的设定齿轮档，则仍将齿轮档切换视作成功完成，且反馈的实际齿轮档在 NC 中激活。

数据类型和更新

USInt，循环

值

0：进给轴模式

1 - 5：齿轮级 1 到 5

关联

- <Axis>.spindle.out.gearChangeoverDone
  完成齿轮箱换挡
- <Axis>.spindle.out.oscillEnable
  使能主轴摆动运动
- <Axis>.spindle.in.requestedGearStageNumber
  所请求齿轮箱挡位 (1 - 5) 的编号
- <Axis>.spindle.in.gearChangeoverRequested
  已请求齿轮箱换挡

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.11.6 invertM3M4

PLC → NC	<Axis>.spindle.out.invertM3M4
反转 M03/M04：电机反方向旋转	

说明

M3/M4 取反

在以下功能中主轴电机改变旋转方向:

- M3
- M4
- M5
- 移动时执行 SPOS/M19/SPOSA；静止时执行 SPOS/M19/SPOSA 无作用。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

已请求将通过 M3 / M4 编写的旋转方向取反。

脉冲沿切换 1 → 0

无作用

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.11.7 oscillEnable

PLC → NC	<Axis>.spindle.out.oscillEnable
针对主轴使能摆动运动	

说明

对于“以通过 PLC 用户程序控制的方式进行机床轴往复”功能而言，必须置位或设置下列系统数据：

- <Axis>.spindle.out.oscillRequest = 1（请求通过 PLC 进行摆动控制）
- <Axis>.spindle.out.oscillEnable = 1（使能主轴摆动运动）
- MD35400 $MA_SPIND_OSCILL_DES_VELO = <往复转速>
- MD35410 $MA_SPIND_OSCILL_ACCEL = <往复加速度>
- <Axis>.spindle.out.oscillSetDirCw = ... → 0 → 1 → 0 ...（将摆动方向设为“顺时针”）
- <Axis>.spindle.out.oscillSetDirCcw = ... → 1 → 0 → 1 ...（将摆动方向设为“逆时针”）

必须通过 PLC 用户程序设定起始旋转方向和每个旋转方向的运行持续时间。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

已请求“通过 PLC 用户程序控制的方式进行机床轴往复”功能。

脉冲沿切换 1 → 0

未请求“以通过 PLC 用户程序控制的方式进行机床轴往复”功能。

关联

- <Axis>.spindle.out.oscillSetDirCw
  将摆动方向设为“顺时针”
- <Axis>.spindle.out.oscillSetDirCcw
  将摆动方向设为“逆时针”

- MD35400 $MA_SPIND_OSCILL_DES_VELO
  摆动转速
- MD35410 $MA_SPIND_OSCILL_ACCEL
  摆动时的加速度

12.3.11.8 oscillRequest

PLC → NC	<Axis>.spindle.out.oscillRequest
请求通过 PLC 进行摆动控制	

说明

通过 PLC 往复

对于“以通过 PLC 用户程序控制的方式进行机床轴往复”功能而言，必须置位或设置下列系统数据：

- <Axis>.spindle.out.oscillRequest = 1（请求通过 PLC 进行摆动控制）
- <Axis>.spindle.out.oscillEnable = 1（使能主轴摆动运动）
- MD35400 $MA_SPIND_OSCILL_DES_VELO = <往复转速>
- MD35410 $MA_SPIND_OSCILL_ACCEL = <往复加速度>
- <Axis>.spindle.out.oscillSetDirCw = ... → 0 → 1 → 0 ...（将摆动方向设为“顺时针”）
- <Axis>.spindle.out.oscillSetDirCcw = ... → 1 → 0 → 1 ...（将摆动方向设为“逆时针”）

必须通过 PLC 用户程序设定起始旋转方向和每个旋转方向的运行持续时间。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

已请求“通过 PLC 用户程序控制的方式进行机床轴往复”功能。

脉冲沿切换 1 → 0

未请求“以通过 PLC 用户程序控制的方式进行机床轴往复”功能。

关联

- <Axis>.spindle.out.oscillEnable
  使能主轴摆动运动
- <Axis>.spindle.out.oscillSetDirCw
  将摆动方向设为“顺时针”
- <Axis>.spindle.out.oscillSetDirCcw
  将摆动方向设为“逆时针”
- MD35400 $MA_SPIND_OSCILL_DES_VELO
  摆动转速
- MD35410 $MA_SPIND_OSCILL_ACCEL
  摆动时的加速度

12.3.11.9 oscillSetDirCcw

PLC → NC	<Axis>.spindle.out.oscillSetDirCcw
将摆动方向设为“逆时针”	

说明

在除往复模式以外的所有主轴运行方式下，此接口信号不相关。

数据类型与更新

BOOL，循环

值 TRUE

已请求逆时针往复旋转方向。

值 FALSE

未请求逆时针往复旋转方向。

关联

- <Axis>.spindle.out.gearStageSelection
  选择齿轮箱实际挡位 (1 - 5)
- <Axis>.spindle.out.gearChangeoverDone
  完成齿轮箱换挡
- <Axis>.spindle.out.oscillRequest
  请求通过 PLC 进行摆动控制

- <Axis>.spindle.out.oscillEnable
  使能主轴摆动运动
- <Axis>.spindle.out.oscillSetDirCw
  将摆动方向设为“顺时针”
- <Axis>.basic.in.isStationary
  xis>.basic.in.isStationary
- <Axis>.spindle.in.requestedGearStageNumber
  所请求齿轮箱挡位 (1 - 5) 的编号
- <Axis>.spindle.in.gearChangeoverRequested
  已请求齿轮箱换挡
- <Axis>.spindle.in.speedSetpointReached
  达到额定转速公差范围 (MD35150 $MA_SPIND_DES_VELO_TOL)
- <Axis>.spindle.in.oscillModeActive
  主轴在摆动运行模式下

更多信息

- 轴和主轴功能手册；“主轴”>“可组态的齿轮箱调整”>“在摆动运行模式下进行齿轮箱换挡”章节

12.3.11.10 oscillSetDirCw

PLC → NC	<Axis>.spindle.out.oscillSetDirCw
将摆动方向设为“顺时针”	

说明

在除往复模式以外的所有主轴运行方式下，此接口信号不相关。

数据类型与更新

BOOL，循环

值 TRUE

已请求顺时针往复旋转方向。

值 FALSE

未请求顺时针往复旋转方向。

关联

- <Axis>.spindle.out.gearStageSelection
  选择齿轮箱实际挡位 (1 - 5)
- <Axis>.spindle.out.gearChangeoverDone
  完成齿轮箱换挡
- <Axis>.spindle.out.oscillRequest
  请求通过 PLC 进行摆动控制
- <Axis>.spindle.out.oscillEnable
  使能主轴摆动运动
- <Axis>.spindle.out.oscillSetDirCcw
  将摆动方向设为“顺时针”
- <Axis>.basic.in.isStationary
  轴达到静止转速范围 (n<n_min)
- <Axis>.spindle.in.requestedGearStageNumber
  所请求齿轮箱挡位 (1 - 5) 的编号
- <Axis>.spindle.in.gearChangeoverRequested
  已请求齿轮箱换挡
- <Axis>.spindle.in.speedSetpointReached
  达到额定转速公差范围 (MD35150 $MA_SPIND_DES_VELO_TOL)
- <Axis>.spindle.in.oscillModeActive
  主轴在摆动运行模式下

更多信息

- 轴和主轴功能手册；“主轴”>“可组态的齿轮箱调整”>“在摆动运行模式下进行齿轮箱换挡”章节

12.3.11.11 ovrFactor

PLC → NC	<Axis>.spindle.out.ovrFactor
控制主轴转速的倍率系数通过 MCP 主轴倍率旋钮等确定	

说明

二进制/格雷码编码

倍率系数可以二进制码或格雷码设定。使用的格式必须通过以下机床数据告知控制系统：

MD12060 $MN_OVR_SPIN_IS_GRAY_CODE = <编码方式>

限制

- 控制系统内部将生效的倍率系数限制在 2.00 或 200% 内。
- 在主轴运行方式”摆动运行”中，生效的倍率系数始终被设定为 100 %。
- 可通过以下机床数据将最大倍率系数限制为一个小于 200% 的值：
  MD12100 $MN_OVR_FACTOR_LIMIT_BIN

说明

- 在通过以下功能进行螺纹切削期间，主轴专用的转速倍率不生效：G63
- 主轴专用转速倍率在其他限制（例如 G26、LIMS...）之前，作用于编程的值。

数据类型和更新

USInt，循环

二进制编码时的值

采用二进制码时，接口中的值对应倍率系数。

二进制编码	十进制	倍率系数
0000 0000	0	0.00
0000 0001	1	0.01
0000 0010	2	0.02
0000 0011	3	0.03
000 0100	4	0.04
...	...	...
0110 0100	100	1.00
...	...	...
1100 1000	200	2.00

格雷码编码时的值

以下机床数据可以指定接口格雷码形式的数值对应的倍率系数：

MD12070 $MN_OVR_FACTOR_SPIND_SPEED[<开关位置> - 1] = <倍率系数>

开关位置	格雷码	倍率系数 [1)]
1	00001	0.50
2	00011	0.55

开关位置	格雷码	倍率系数 1)
3	00010	0.60
4	00110	0.65
5	00111	0.70
6	00101	0.75
7	00100	0.80
8	01100	0.85
9	01101	0.90
10	01111	0.95
11	01110	1.00
12	01010	1.05
13	01011	1.10
14	01001	1.15
15	01000	1.20
16	11000	1.20
17	11001	1.20
18	11011	1.20
19	11010	1.20
20	11110	1.20
21	11111	1.20
22	11101	1.20
23	11100	1.20
24	10100	1.20
25	10101	1.20
26	10111	1.20
27	10110	1.20
28	10010	1.20
29	10011	1.20
30	10001	1.20
31	10000	1.20

1) 缺省值

关联

- <Axis>.basic.out.feedrateOvrFactor
  进给倍率的标准化因数可以是二进制数或格雷码 (MD12000 $MN_OVR_AX_IS_GRAY_CODE)
- <Axis>.basic.out.enableOvr
  激活进给/主轴倍率
- MD12080 $MN_OVR_REFERENCE_IS_PROG_FEED
  倍率基准速度

更多信息

- 轴和主轴功能手册；“进给”>“进给控制”>“通过机床控制面板设置进给倍率”章节

12.3.11.12 resyncEnc1

PLC → NC	<Axis>.spindle.out.resyncEnc1
请求重新同步主轴的行程系统 1	

说明

在除控制模式以外的所有主轴运行方式下，此接口信号不相关。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

已请求主轴的测量系统 1 的同步。

脉冲沿切换 1 → 0

无作用

关联

- <Axis>.basic.in.enc1Synchronized
  机床轴位置测量系统 1 已回参考点/同步

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.11.13 resyncEnc1ForPos

PLC → NC	<Axis>.spindle.out.resyncEnc1ForPos
请求在定位主轴前，重新同步主轴的行程系统 1	

说明

在除定位模式以外的所有主轴运行方式下，此接口信号不相关。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

已请求在主轴定位前重新同步测量系统 1。

脉冲沿切换 1 → 0

无作用

关联

- <Axis>.basic.in.enc1Synchronized
  机床轴位置测量系统 1 已回参考点/同步

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.11.14 resyncEnc2

PLC → NC	<Axis>.spindle.out.resyncEnc2
请求重新同步主轴的行程系统 2	

说明

重新同步主轴，测量系统 2

说明

在除控制模式以外的所有主轴运行方式下，此接口信号不相关。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

已请求主轴的测量系统 2 的同步。

脉冲沿切换 1 → 0

无作用

关联

- <Axis>.basic.in.enc2Synchronized
  机床轴位置测量系统 2 已回参考点/同步

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.11.15 resyncEnc2ForPos

PLC → NC	<Axis>.spindle.out.resyncEnc2ForPos
请求在定位主轴前，重新同步主轴的行程系统 2	

说明

在除定位模式以外的所有主轴运行方式下，此接口信号不相关。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

已请求在主轴定位前重新同步测量系统 2。

脉冲沿切换 1 → 0

无作用

关联

- <Axis>.basic.in.enc2Synchronized
  机床轴位置测量系统 2 已回参考点/同步

更多信息

- 轴和主轴功能手册；“主轴”章节

12.3.11.16 startCcw

PLC → NC	<Axis>.spindle.out.startCcw
启动主轴逆时针旋转	

说明

启动主轴逆时针移动

技术 - 通过 PLC 接口进行主轴移动：主轴逆转

数据类型与更新

BOOL

值 TRUE

启动主轴逆时针移动

值 FALSE

不启动主轴移动

12.3.11.17 startCw

PLC → NC	<Axis>.spindle.out.startCw
启动主轴顺时针旋转	

说明

启动主轴顺时针移动

技术 - 通过 PLC 接口进行主轴移动：主轴顺转

数据类型与更新

BOOL

值 TRUE

启动主轴顺时针移动

值 FALSE

不启动主轴移动

12.3.11.18 startPosOperation

PLC → NC	<Axis>.spindle.out.startPosOperation
主轴开始定位	

说明

定位主轴启动

技术 - 通过 PLC 接口进行主轴移动：定位主轴启动

数据类型与更新

BOOL

值 TRUE

定位主轴启动

值 FALSE

不定位主轴启动

12.3.11.19 stop

PLC → NC	<Axis>.spindle.out.stop
停止主轴运动	

说明

停止主轴移动

技术 - 通过 PLC 接口进行主轴移动：主轴停止

数据类型与更新

BOOL

值 TRUE

停止主轴移动

值 FALSE

不停止主轴移动

12.3.11.20 useFeedrateOvr

PLC → NC	<Axis>.spindle.out.useFeedrateOvr
为主轴转速控制使用进给倍率	

数据类型与更新

BOOL

值 TRUE

使用主轴速度控制进给修调

值 FALSE

不使用主轴速度控制进给修调

12.3.12 <Axis>.oscillation.in

12.3.12.1 active

NC → PLC	<Axis>.oscillation.in.active
摆动运行处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

轴目前作为往复轴运行。

值 FALSE

轴为定位轴。

关联

- <Axis>.oscillation.in.sparkingOutActive
  修光处于活动状态
- <Axis>.oscillation.in.motionActive
  摆动运动处于活动状态

更多信息

- 轴和主轴功能手册；“摆动”章节

12.3.12.2 disabled

NC → PLC	<Axis>.oscillation.in.disabled
“切换至摆动运行”锁定	

说明

例如在编程有误的情况下，此信号置位。

说明

在已经运行的情况下，也可能出现此状态。

数据类型与更新

BOOL，循环

值 TRUE

无法启动往复轴。

值 FALSE

可以启动往复运动。

更多信息

- 轴和主轴功能手册；“摆动”章节

12.3.12.3 error

NC → PLC	<Axis>.oscillation.in.error
摆动运行中出错	

数据类型与更新

BOOL

值 TRUE

往复运动期间出错，故往复运动已终止。

值 FALSE

往复运动正常进行。

更多信息

- 轴和主轴功能手册；“摆动”章节

12.3.12.4 motionActive

NC → PLC	<Axis>.oscillation.in.motionActive
摆动运动处于活动状态	

说明

在以下情形下，此信号不相关：

<Axis>.oscillation.in.active == FALSE

数据类型与更新

BOOL，循环

值 TRUE

轴在两个换向点之间执行往复运动。

值 FALSE

轴当前不作往复运动。

关联

- <Axis>.oscillation.in.active
  摆动运行处于活动状态

更多信息

- 轴和主轴功能手册；“摆动”章节

12.3.12.5 externalReversalActive

NC → PLC	<Axis>.oscillation.in.externalReversalActive
“从外部触发摆动换向”处于活动状态	

说明

从外部触发往复运动换向有效

数据类型与更新

BOOL，循环

值 TRUE

“从外部触发往复运动换向”后，制动阶段生效。

值 FALSE

“从外部触发往复运动换向”后，制动阶段未生效。

关联

- <Axis>.oscillation.out.externalReversal
  停止摆动运动，沿相反的方向运行摆动轴

更多信息

- 轴和主轴功能手册；“摆动”章节

12.3.12.6 sparkingOutActive

NC → PLC	<Axis>.oscillation.in.sparkingOutActive
修光处于活动状态	

说明

光磨有效

数据类型与更新

BOOL，循环

值 TRUE

轴正在进行光磨。

值 FALSE

轴当前未执行光磨。

关联

- <Axis>.oscillation.in.active
  摆动运行处于活动状态

更多信息

- 轴和主轴功能手册；“摆动”章节

12.3.13 <Axis>.oscillation.out

12.3.13.1 activateReversalPoint

PLC → NC	<Axis>.oscillation.out.activateReversalPoint
激活选中的换向点	

说明

如需将制动到达的位置作为新的换向位置，可使用信号 <Axis>.oscillation.out.activateReversalPoint（激活选择的换向点）实现。信号 <Axis>.oscillation.out.selectReversalPoint（选择换向点）不在考虑范围内，变更基于最后“从外部触发的摆动换向”生效。

说明

对于轴而言，不允许通过移动键或手轮来修改换向点。在此情形下，系统会发出显示报警 20081“无法将制动位置接收作为换向位置 – 手轮生效”。

数据类型与更新

BOOL，循环

值 TRUE

可通过手动运行轴来变更换向点的位置。

与 <Axis>.oscillation.out.externalReversal（停止摆动运动，沿相反的方向运行摆动轴）组合使用：

在“从外部触发往复轴换向”后，制动后的位置应成为新的换向点。

值 FALSE

不可通过轴的手动运行来修改换向点的位置。

与 <Axis>.oscillation.out.externalReversal（停止摆动运动，沿相反的方向运行摆动轴）组合使用：

换向点的位置不变。

关联

- <Axis>.oscillation.out.externalReversal
  停止摆动运动，沿相反的方向运行摆动轴
- <Axis>.oscillation.out.selectReversalPoint
  选择换向点：
  0 - 第一个换向点
  1 - 第二个换向点

更多信息

- 轴和主轴功能手册；“摆动”章节

12.3.13.2 externalReversal

PLC → NC	<Axis>.oscillation.out.externalReversal
停止摆动运动，沿相反的方向运行摆动轴	

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

中断往复运动，然后使往复轴在相反方向上继续运行。

脉冲沿切换 1 → 0

不中断往复运动。

更多信息

- 轴和主轴功能手册；“摆动”章节

12.3.13.3 selectReversalPoint

PLC → NC	<Axis>.oscillation.out.selectReversalPoint
选择换向点： 0 - 第一个换向点 1 - 第二个换向点	

说明

Schleifen pendeln:设置换向点

数据类型与更新

BOOL，循环

值 TRUE

换向点 2

值 FALSE

换向点 1

更多信息

- 轴和主轴功能手册；“摆动”章节

12.3.13.4 stopAtNextReversalPoint

PLC → NC	<Axis>.oscillation.out.stopAtNextReversalPoint
在下一个换向点停止	

数据类型和更新

BOOL，循环

值 TRUE

在下一个换向点上中断 PLC 控制的轴的往复运动。

值 FALSE

未请求在下一个换向点上中断 PLC 控制的轴的往复运动。

关联

- <Axis>.plcCtrl.out.stop
  PLC 控制轴：沿着制动斜坡停止轴
- <Axis>.plcCtrl.out.requestPlcCtrl
  请求通过适用于自主单轴操作的 PLC 进行控制

更多信息

- 功能手册之轴和主轴；章节“摆动运行”

12.3.14 <Axis>.safety.in

12.3.14.1 brakeTestModeEnabled

NC → PLC	<Axis>.safety.in.brakeTestModeEnabled
轴已针对制动测试使能	

数据类型和更新

BOOL

值 TRUE

Safety Integrated 制动测试已激活。

值 FALSE

Safety Integrated 制动测试未激活。

12.3.14.2 infoCtrlChanActive

NC → PLC	<Axis>.safety.in.infoCtrlChanActive
已在 NCK 中激活“通过 SIC 连接”（MD37950 $MA_SAFE_INFO_ENABLE，位 0）	

数据类型与更新

BOOL

12.3.15 <Axis>.safety.out

12.3.15.1 activateBrakeRampForSs2e

PLC → NC	<Axis>.safety.out.activateBrakeRampForSs2e
选择 SS2E 后沿 NCK 斜坡制动 (MD36959 MA_SAFE_BRAKE_RAMP_TIME)	

数据类型和更新

BOOL

TRUE 值

„activateBrakeRampForSs2e" = 1：
选择 SS2E 后，如果在 MD 36959 MA_SAFE_BRAKE_RAMP_TIME 中参数化的值 > 0，则轴将沿 NCK 斜坡制动。

FALSE 值

„activateBrakeRampForSs2e" = 0：
选择 SS2E 后，轴沿路径制动（插补制动）

12.3.15.2 enableBrakeTestMode

PLC → NC	<Axis>.safety.out.enableBrakeTestMode
请求使能轴以进行制动测试	

数据类型与更新

BOOL

12.3.15.3 setpointLimitSelection

PLC → NC	<Axis>.safety.out.setpointLimitSelection
选择设定点速度限制的评估因子 (0..3) (MD36933 $MA_SAFE_DES_VELO_LIMIT)	

数据类型和更新

USInt

12.3.16 <Axis>.safetyInfoChan.in

12.3.16.1 acceptanceTestModeSelected

NC → PLC	<Axis>.safetyInfoChan.in.acceptanceTestModeSelected
已选验收测试模式	

数据类型与更新

BOOL

值 TRUE

已选验收测试模式

值 FALSE

未选验收测试模式

12.3.16.2 acceptanceTestSlpSelected

NC → PLC	<Axis>.safetyInfoChan.in.acceptanceTestSlpSelected
已选 SLP 验收测试	

说明

已请求延时停止和回退

数据类型与更新

BOOL

值 TRUE

已选 SLP 验收测试

值 FALSE

未选 SLP 验收测试

12.3.16.3 activeBrake

NC → PLC	<Axis>.safetyInfoChan.in.activeBrake
TRUE:制动测试 2 处于活动状态 FALSE：制动测试 1 处于活动状态	

数据类型和更新

BOOL

值 TRUE

制动测试 2 生效

值 FALSE

制动测试 1 处于活动状态

12.3.16.4 brakeTest

NC → PLC	<Axis>.safetyInfoChan.in.brakeTest
已选制动测试	

数据类型与更新

BOOL

值 TRUE

已选制动测试

值 FALSE

未选制动测试

12.3.16.5 brakeTestActive

NC → PLC	<Axis>.safetyInfoChan.in.brakeTestActive
制动测试处于活动状态	

数据类型和更新

BOOL

值 TRUE

制动测试处于活动状态

值 FALSE

制动测试未激活

12.3.16.6 brakeTestCompleted

NC → PLC	<Axis>.safetyInfoChan.in.brakeTestCompleted
TRUE：完成制动测试 FALSE：未完整完成制动测试	

数据类型与更新

BOOL

值 TRUE

完成制动测试

值 FALSE

未完整完成制动测试

12.3.16.7 brakeTestResult

NC → PLC	<Axis>.safetyInfoChan.in.brakeTestResult
TRUE:成功完成制动测试 FALSE：制动测试出错	

数据类型和更新

BOOL

值 TRUE

成功完成制动测试

值 FALSE

制动测试出错

12.3.16.8 currentLoadSign

NC → PLC	<Axis>.safetyInfoChan.in.currentLoadSign
当前负载符号： TRUE：负 FALSE：正	

数据类型与更新

BOOL

值 TRUE

当前负载符号为负

值 FALSE

当前负载符号为正

12.3.16.9 esrRetractRequested

NC → PLC	<Axis>.safetyInfoChan.in.esrRetractRequested
已请求 ESR 退回	

数据类型与更新

BOOL

值 TRUE

已请求延时停止和回退。

值 FALSE

未请求延时停止和回退。

12.3.16.10 externalBrakeRequested

NC → PLC	<Axis>.safetyInfoChan.in.externalBrakeRequested
TRUE：制动器闭合 FALSE：制动器打开	

数据类型与更新

BOOL

值 TRUE

制动器闭合

值 FALSE

制动器打开

12.3.16.11 safetyFaultActive

NC → PLC	<Axis>.safetyInfoChan.in.safetyFaultActive
出现内部事件（SINAMICS 安全故障）	

数据类型和更新

BOOL

值 TRUE

待处理的内部事件

值 FALSE

无内部事件

12.3.16.12 safetyMessageActive

NC → PLC	<Axis>.safetyInfoChan.in.safetyMessageActive
有安全消息	

数据类型与更新

BOOL

值 TRUE

安全消息处于活动状态。

值 FALSE

安全消息未处于活动状态。

12.3.16.13 sdiNegativeSelected

NC → PLC	<Axis>.safetyInfoChan.in.sdiNegativeSelected
已选择安全功能 Safe Direction（负方向）（尚未激活）	

数据类型与更新

BOOL

值 TRUE

已选“安全运动方向（负向）”安全功能

值 FALSE

未选“安全运动方向（负向）”安全功能

12.3.16.14 sdiPositiveSelected

NC → PLC	<Axis>.safetyInfoChan.in.sdiPositiveSelected
已选择安全功能 Safe Direction（正方向）（尚未激活）	

数据类型与更新

BOOL

值 TRUE

已选“安全运动方向（正向）”安全功能

值 FALSE

未选“安全运动方向（正向）”安全功能

12.3.16.15 setpointInput

NC → PLC	<Axis>.safetyInfoChan.in.setpointInput
驱动设定值规定：SBT 功能或外部源（控制器）规定	

数据类型与更新

BOOL

值 TRUE

有针对驱动的设定值规定：SBT 功能或外部源（控制器）规定

值 FALSE

没有针对驱动的设定值规定：SBT 功能或外部源（控制器）规定

12.3.16.16 slaActive

NC → PLC	<Axis>.safetyInfoChan.in.slaActive
安全功能 Safely Limited Acceleration 处于活动状态	

数据类型与更新

BOOL

值 TRUE

安全限制加速处于活动状态。

值 FALSE

安全限制加速未激活。

12.3.16.17 slpRangeSelected

NC → PLC	<Axis>.safetyInfoChan.in.slpRangeSelected
TRUE：已选 SLP 位置范围 2 FALSE：已选 SLP 位置范围 1	

说明

安全限制位置：已选限制 Bit 0

数据类型与更新

BOOL

值 TRUE

已选 SLP 位置范围 2

值 FALSE

已选 SLP 位置范围 1

12.3.16.18 slpSelectedAndUserAgreement

NC → PLC	<Axis>.safetyInfoChan.in.slpSelectedAndUserAgreement
已选择 SLP 并执行用户确认	

数据类型与更新

BOOL

值 TRUE

已选择 SLP 并执行用户确认

值 FALSE

未选择 SLP 或未进行用户确认

12.3.16.19 slsActive

NC → PLC	<Axis>.safetyInfoChan.in.slsActive
安全功能 Safely Limited Speed 处于活动状态	

数据类型与更新

BOOL

值 TRUE

安全限制速度 (SLS) 处于活动状态。

值 FALSE

安全限制速度 (SLS) 未激活。

12.3.16.20 slsBit0Active

NC → PLC	<Axis>.safetyInfoChan.in.slsBit0Active
一个 SLS 限制（共四个）处于活动状态；与 Bit 0 组合使用	

数据类型和更新

BOOL

值 TRUE

生效

值 FALSE

未生效

关联

- <Axis>.safetyInfoChan.in.slsBit1Active

12.3.16.21 slsBit1Active

NC → PLC	<Axis>.safetyInfoChan.in.slsBit1Active
一个 SLS 限制（共四个）处于活动状态；与 Bit 1 组合使用	

数据类型和更新

BOOL

值 TRUE

生效

值 FALSE

未生效

关联

- <Axis>.safetyInfoChan.in.slsBit0Active

12.3.16.22 slsLimit

NC → PLC	<Axis>.safetyInfoChan.in.slsLimit
SLS 转速限制具有 32 位分辨率和符号位	

数据类型与更新

DInt

值

SLS 转速限制具有 32 位分辨率和符号位

12.3.16.23 slsSelected

NC → PLC	<Axis>.safetyInfoChan.in.slsSelected
已选 Safely Limited Speed 安全功能（尚未激活）	

数据类型与更新

BOOL

值 TRUE

已选“安全限制速度”安全功能

值 FALSE

未选“安全限制速度”安全功能

12.3.16.24 sosActive

NC → PLC	<Axis>.safetyInfoChan.in.sosActive
安全功能 Safe Operating Stop 处于活动状态	

数据类型与更新

BOOL

值 TRUE

安全操作停止 (SOS) 处于活动状态。

值 FALSE

安全操作停止 (SOS) 未激活。

12.3.16.25 sosSelected

NC → PLC	<Axis>.safetyInfoChan.in.sosSelected
已选安全功能 Safe Operating Stop（尚未激活）	

数据类型与更新

BOOL

值 TRUE

已选“安全操作停止”安全功能

值 FALSE

未选“安全操作停止”安全功能

12.3.16.26 ss1Active

NC → PLC	<Axis>.safetyInfoChan.in.ss1Active
安全功能 Safe Stop 1 处于活动状态	

数据类型与更新

BOOL

值 TRUE

安全停止 1 (SS1) 处于活动状态。

值 FALSE

安全停止 1 (SS1) 未激活。

12.3.16.27 ss2Active

NC → PLC	<Axis>.safetyInfoChan.in.ss2Active
安全功能 Safe Stop 2 处于活动状态	

数据类型与更新

BOOL

值 TRUE

安全停止 2 (SS2) 处于活动状态。

值 FALSE

安全停止 2 (SS2) 未激活。

12.3.16.28 ss2eActive

NC → PLC	<Axis>.safetyInfoChan.in.ss2eActive
安全功能 SS2E 处于活动状态	

说明

安全限制速度已选限制 Bit 0

数据类型与更新

BOOL

值 TRUE

安全功能 SS2E 处于活动状态

值 FALSE

安全功能 SS2E 未激活

12.3.16.29 ss2esrActive

NC → PLC	<Axis>.safetyInfoChan.in.ss2esrActive
Safe Stop 2 延时停止和回退 (SS2ESR) 处于活动状态	

说明

安全限制速度已选限制 Bit1

数据类型与更新

BOOL

值 TRUE

Safe Stop 2 延时停止和回退 (SS2ESR) 处于活动状态

值 FALSE

Safe Stop 2 延时停止和回退 (SS2ESR) 未激活

12.3.16.30 stoActive

NC → PLC	<Axis>.safetyInfoChan.in.stoActive
安全功能 Safe Torque Off 处于活动状态	

数据类型和更新

BOOL

值 TRUE

Safe Torque Off 已激活。

值 FALSE

Safe Torque Off 未激活。

12.3.16.31 testStopActive

NC → PLC	<Axis>.safetyInfoChan.in.testStopActive
强制潜在故障检查生效	

数据类型与更新

BOOL

值 TRUE

测试停止处于活动状态。

值 FALSE

测试停止未激活。

12.3.16.32 testStopRequired

NC → PLC	<Axis>.safetyInfoChan.in.testStopRequired
已请求强制非激活故障检查	

数据类型与更新

BOOL

值 TRUE

需要测试停止。

值 FALSE

不需要测试停止。

12.3.17 <Axis>.safetyCtrlChan.out

12.3.17.1 brakeSelection

PLC → NC	<Axis>.safetyCtrlChan.out.brakeSelection
TRUE：已选制动测试 2 FALSE：已选制动测试 1	

数据类型与更新

BOOL

值 TRUE

已选制动测试 2

值 FALSE

已选制动测试 1

12.3.17.2 closeBrake

PLC → NC	<Axis>.safetyCtrlChan.out.closeBrake
关闭内部制动器	

数据类型与更新

BOOL

值 TRUE

关闭内部制动器

值 FALSE

不关闭内部制动器

12.3.17.3 externalBrakeClosed

PLC → NC	<Axis>.safetyCtrlChan.out.externalBrakeClosed
TRUE：外部制动器闭合 FALSE：外部制动器打开	

数据类型与更新

BOOL

值 TRUE

外部制动器闭合

值 FALSE

外部制动器打开

12.3.17.4 selectBrakeTest

PLC → NC	<Axis>.safetyCtrlChan.out.selectBrakeTest
制动测试已选	

数据类型与更新

BOOL

值 TRUE

已选制动测试

值 FALSE

未选制动测试

12.3.17.5 selectDirectionOfRotation

PLC → NC	<Axis>.safetyCtrlChan.out.selectDirectionOfRotation
TRUE：已选负方向 FALSE：已选正方向	

说明

旋转方向

数据类型与更新

BOOL

值 TRUE

已选负方向

值 FALSE

已选正方向

12.3.17.6 selectTestSequence

PLC → NC	<Axis>.safetyCtrlChan.out.selectTestSequence
TRUE：已选测试序列 2 FALSE：已选测试序列 1	

数据类型与更新

BOOL

值 TRUE

已选测试序列 2

值 FALSE

已选测试序列 1

12.3.17.7 startBrakeTest

PLC → NC	<Axis>.safetyCtrlChan.out.startBrakeTest
已请求启动制动测试	

数据类型与更新

BOOL

值 TRUE

已请求启动制动测试

值 FALSE

未请求启动制动测试

12.3.17.8 testStop

PLC → NC	<Axis>.safetyCtrlChan.out.testStop
已选强制非激活故障检查	

说明

用于扩展功能的强制潜在故障检查

数据类型与更新

BOOL

值 TRUE

已选强制潜在故障检查

值 FALSE

未选强制潜在故障检查

12.3.18 <Axis>.coupling.in

12.3.18.1 AccelReached

NC → PLC	<Axis>.coupling.in.AccelReached
通用耦合：从动轴达到或超过针对“轴加速”显示的最小值 (MD37560 $MA_EG_ACC_TOL)	

仅当耦合功能“电子齿轮（ELG）”生效时，此信号才相关。当从动轴的加速度达到或超出通过 MD37560 和 MD32300 定义的加速度值时，此信号置位。

数据类型与更新

BOOL，循环

值 TRUE

通用耦合：从动轴的加速度已达到或超出参数设置的加速度值，其中显示轴的加速度。

值 FALSE

通用耦合：从动轴的加速度低于参数设置的加速度值，其中显示轴的加速度。

关联

- MD37560 $MA_EG_ACC_TOL
  “轴加速”阈值
- MD32300 $MA_MAX_AX_ACCEL
  最大轴加速度

更多信息

- 轴和主轴功能手册；“轴耦合”章节

12.3.18.2 AccelThresholdReached

NC → PLC	<Axis>.coupling.in.AccelThresholdReached
通用耦合：从动主轴达到或超过加速警告阈值	

说明

仅当耦合功能“电子齿轮（ELG）”生效时，此信号才相关。当从动轴的加速度达到或超出通过 MD37550 和 MD32300 定义的加速度报警阈值（= 最大轴加速度的百分比值）时，此信号置位。

数据类型与更新

BOOL，循环

值 TRUE

通用耦合：从动轴的加速度已达到或超出参数设置的加速度报警阈值。

值 FALSE

通用耦合：从动轴的加速度低于参数设置的加速度报警阈值。

关联

- MD37550 $MA_EG_VEL_WARNING
  速度警告阈值
- MD32300 $MA_MAX_AX_ACCEL
  最大轴加速度

更多信息

- 轴和主轴功能手册；“轴耦合”章节

12.3.18.3 correctionActive

NC → PLC	<Axis>.coupling.in.correctionActive
通用耦合：同步运行补偿处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

通用耦合：同步运行补偿处于活动状态。

值 FALSE

通用耦合：同步运行补偿未激活。

12.3.18.4 correctionDone

NC → PLC	<Axis>.coupling.in.correctionDone
通用耦合：跟随完成运行补偿 ($AA_COUP_CORR)	

数据类型与更新

BOOL，循环

值 TRUE

通用耦合：跟随完成运行补偿

值 FALSE

通用耦合：未跟随完成运行补偿

关联

- $AA_COUP_CORR

12.3.18.5 followingSpindleActive

NC → PLC	<Axis>.coupling.in.followingSpindleActive
通用耦合：主轴是从动主轴	

说明

该信号只用于同步运行。

说明

<Axis>.coupling.in.followingSpindleActive == TRUE 时，从动主轴以相应的传动比跟随主动主轴移动。

- **特殊情况或错误**
  如果在副主轴上出现的故障导致副主轴的“伺服使能”被取消，在一定前提条件下，控制系统内部会将副主轴与主主轴的耦合关系更换并切换至实际值耦合。

数据类型与更新

BOOL，循环

值 TRUE

通用耦合：机床轴当前作为副主轴工作。

值 FALSE

通用耦合：机床轴当前不作为副主轴工作。

关联

- <Axis>.spindle.in.syncModeActive
  主轴在同步运行模式下
- <Axis>.coupling.in.leadingSpindleActive
  通用耦合：主轴是主动主轴

更多信息

- 轴和主轴功能手册；“同步主轴”章节

12.3.18.6 gantryActive

NC → PLC	<Axis>.coupling.in.gantryActive
龙门：轴是龙门系统的组成部分	

数据类型与更新

BOOL

值 TRUE

龙门架轴是龙门组合的构成部分。

值 FALSE

龙门架轴不是龙门组合的构成部分。

12.3.18.7 gantryCutOffLimitExceeded

NC → PLC	<Axis>.coupling.in.gantryCutOffLimitExceeded
龙门：已超过关闭极限值	

数据类型与更新

BOOL

值 TRUE

已超过龙门关闭极限

值 FALSE

未超过龙门关闭极限

12.3.18.8 gantryLeadingActive

NC → PLC	<Axis>.coupling.in.gantryLeadingActive
龙门：轴为主动轴	

数据类型与更新

BOOL

值 TRUE

轴为龙门主动轴。

值 FALSE

轴不是龙门主动轴。

12.3.18.9 gantrySyncOk

NC → PLC	<Axis>.coupling.in.gantrySyncOk
龙门：系统的所有轴都已同步	

数据类型与更新

BOOL

值 TRUE

龙门组合的所有轴都已同步

值 FALSE

并非龙门组合的所有轴都已同步

12.3.18.10 gantrySyncRunReady

NC → PLC	<Axis>.coupling.in.gantrySyncRunReady
龙门：可以开始同步运行	

数据类型与更新

BOOL

值 TRUE

龙门同步运行进入启动准备就绪状态

值 FALSE

龙门同步运行未进入启动准备就绪状态

12.3.18.11 gantryWarningLimitExceeded

NC → PLC	<Axis>.coupling.in.gantryWarningLimitExceeded
龙门：已超过警告阈值	

数据类型与更新

BOOL

值 TRUE

已超过龙门警告极限

值 FALSE

未超过龙门警告极限

12.3.18.12 leadingSpindleActive

NC → PLC	<Axis>.coupling.in.leadingSpindleActive
通用耦合：主轴是主动主轴	

说明

该信号只用于同步运行。

说明

- **特殊情况或错误**
  如果在副主轴上出现的故障导致副主轴的“伺服使能”被取消，在一定前提条件下，控制系统内部会将副主轴与主主轴的耦合关系更换并切换至实际值耦合。在此情形下，原先的主主轴变为新的生效的副主轴（接口信号“副主轴有效”）。

数据类型与更新

BOOL，循环

值 TRUE

通用耦合：机床轴当前作为主主轴工作。

值 FALSE

通用耦合：机床轴当前不作为主主轴工作。

关联

- <Axis>.spindle.in.syncModeActive
  主轴在同步运行模式下
- <Axis>.coupling.in.followingSpindleActive
  通用耦合：主轴是从动主轴

更多信息

- 轴和主轴功能手册；“同步主轴”章节

12.3.18.13 maxAccelReached

NC → PLC	<Axis>.coupling.in.maxAccelReached
通用耦合：已达到最大加速度	

数据类型与更新

BOOL，循环

值 TRUE

通用耦合：已达到最大加速度。

值 FALSE

通用耦合：未达到最大加速度。

12.3.18.14 maxSpeedReached

NC → PLC	<Axis>.coupling.in.maxSpeedReached
通用耦合：已达到最大转速	

数据类型与更新

BOOL，循环

值 TRUE

通用耦合：已达到最大转速。

值 FALSE

通用耦合：未达到最大转速。

12.3.18.15 mcsCouplingActive

NC → PLC	<Axis>.coupling.in.mcsCouplingActive
MCS 耦合：耦合处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

耦合处于活动状态

值 FALSE

耦合未处于活动状态

12.3.18.16 mcsFollowingAxisModeActive

NC → PLC	<Axis>.coupling.in.mcsFollowingAxisModeActive
MCS 耦合：轴为从动轴	

数据类型和更新

BOOL，循环

值 TRUE

轴为跟随轴。

值 FALSE

轴不是跟随轴。

12.3.18.17 mcsMirroringActive

NC → PLC	<Axis>.coupling.in.mcsMirroringActive
MCS 耦合：镜像处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

镜像处于活动状态

值 FALSE

镜像未激活

12.3.18.18 mcsOffsetChanged

NC → PLC	<Axis>.coupling.in.mcsOffsetChanged
MCS 耦合：偏移已变化	

数据类型与更新

BOOL，循环

值 TRUE

偏移已变化。

值 FALSE

偏移未变化。

12.3.18.19 motionOverlaid

NC → PLC	<Axis>.coupling.in.motionOverlaid
通用耦合：从动主轴的附加叠加处于活动状态（比如主动主轴和从动主轴之间的起始角偏移）	

说明

该信号只用于同步运行中的副主轴。

说明

从动主轴叠加运动应用实例：

- 启用同步模式，副主轴与主主轴之间存在定义的角度偏移
- 在主主轴旋转的情况下启用同步模式

- 在同步模式生效时修改传动比
- 在同步模式生效时设定新的角度偏差
- 在同步模式生效时，在 JOG 运行方式下通过移动键"+"/"-"或手轮运行从动轴。

数据类型与更新

BOOL，循环

值 TRUE

通用耦合：副主轴执行一个额外的运动分量，叠加在与主主轴的耦合运动上。

值 FALSE

通用耦合：副主轴不执行额外的运动分量或者已经结束。

关联

- <Axis>.spindle.in.syncModeActive
  主轴在同步运行模式下

更多信息

- 轴和主轴功能手册；“同步主轴”章节

12.3.18.20 msActive

NC → PLC	<Axis>.coupling.in.msActive
主/从：耦合处于活动状态	

说明

在主从耦合中，不允许为从动轴的制动控制逻辑处理接口信号 <Axis>.basic.in.posCtrlActive（位置控制器激活），因为在主从耦合处于活动状态时，该信号不再置位。使用本接口信号作为替代。

数据类型与更新

BOOL，循环

值 TRUE

耦合生效。

值 FALSE

耦合未生效。

更多信息

- 功能手册之工艺

12.3.18.21 msCompCtrlActive

NC → PLC	<Axis>.coupling.in.msCompCtrlActive
主/从：转矩补偿控制器处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

转矩补偿控制器处于活动状态。

值 FALSE

转矩补偿控制器未激活。

关联

- <Axis>.coupling.out.msTorqueComp
  主/从：请求接通转矩补偿控制器
- MD37254 $MA_MS_TORQUE_CTRL_MODE
  转矩补偿控制器互联

更多信息

- 功能手册之工艺

12.3.18.22 msSpeedDifferenceCoarseReached

NC → PLC	<Axis>.coupling.in.msSpeedDifferenceCoarseReached
主/从：转速差在粗公差范围内 (MD37270 $MS_VELO_TOL_COARSE)	

数据类型和更新

BOOL，循环

值 TRUE

主动轴与从动轴之间的转速差处于通过 MD37270 设定的公差范围内。

值 FALSE

主动轴与从动轴之间的转速差处于通过 MD37270 设定的公差范围外。

关联

- <Axis>.coupling.out.msActivate
  主/从：激活耦合
- <Axis>.coupling.in.msSpeedDifferenceFineReached
  主/从：转速差在精公差范围内 (MD37272 $MS_VELO_TOL_FINE)
- <Axis>.coupling.in.msActive
  主/从：耦合处于活动状态
- MD37270 $MA_MS_VELO_TOL_COARSE
  主/从：粗速度公差

更多信息

- 功能手册之工艺

12.3.18.23 msSpeedDifferenceFineReached

NC → PLC	<Axis>.coupling.in.msSpeedDifferenceFineReached
主/从：转速差在精公差范围内 (MD37272 $MS_VELO_TOL_FINE)	

数据类型与更新

BOOL，循环

值 TRUE

主动轴与从动轴之间的转速差处于通过 MD37272 设定的公差范围内。

值 FALSE

主动轴与从动轴之间的转速差处于通过 MD37272 设定的公差范围外。

关联

- <Axis>.coupling.out.msTorqueComp
  主/从：请求接通转矩补偿控制器
- <Axis>.coupling.out.msActivate
  主/从：激活耦合
- <Axis>.coupling.in.msSpeedDifferenceCoarseReached
  主/从：转速差在粗公差范围内 (MD37270 $MS_VELO_TOL_COARSE)

- <Axis>.coupling.in.msActive
  主/从：耦合处于活动状态
- MD37272 $MA_MS_VELO_TOL_FINE
  主/从：精速度公差

更多信息

- 功能手册之工艺

12.3.18.24 processActive

NC → PLC	<Axis>.coupling.in.processActive
通用耦合：同步处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

通用耦合：同步处于活动状态。

值 FALSE

通用耦合：同步未激活。

12.3.18.25 SpeedThresholdReached

NC → PLC	<Axis>.coupling.in.SpeedThresholdReached
通用耦合：从动主轴达到或超过转速警告阈值	

说明

仅当耦合功能“电子齿轮（ELG）”生效时，此信号才相关。当从动轴的速度达到或超出通过MD37550和MD32000定义的速度报警阈值（=最大轴速度的百分比值）时，此信号置位。

数据类型与更新

BOOL，循环

值 TRUE

通用耦合：从动轴的速度已达到或超出参数设置的速度报警阈值。

值 FALSE

通用耦合：从动轴的速度低于参数设置的速度报警阈值。

关联

- MD37550 $MA_EG_VEL_WARNING
  速度警告阈值
- MD32000 $MA_MAX_AX_VELO
  最大轴速度

更多信息

- 轴和主轴功能手册；“轴耦合”章节

12.3.18.26 syncCoarse2Reached

NC → PLC	<Axis>.coupling.in.syncCoarse2Reached
通用耦合：实际位置闭环控制的同步运行达到“粗公差范围”(MD37202 $MA_COUPLE_POS_TOL_COARSE_2)	

数据类型与更新

BOOL，循环

值 TRUE

通用耦合：从动主轴与主动主轴之间的位置差或速度差处于“粗同步运行 2”公差带内。

值 FALSE

通用耦合：从动主轴与主动主轴之间的位置差或速度差处于“粗同步运行 2”公差带外。

关联

- <Axis>.spindle.in.syncModeActive
  主轴在同步运行模式下
- MD37202 $MA_COUPLE_POS_TOL_COARSE_2
  “粗同步运行 2”的阈值

更多信息

- 轴和主轴功能手册；“同步主轴”章节

12.3.18.27 syncCoarseReached

NC → PLC	<Axis>.coupling.in.syncCoarseReached
通用耦合：（位置或转速）实际值和设定值之差达到“粗公差范围”(MD37200 $MA_COUPLE_POS_TOL_COARSE, MD37220 $MA_COUPLE_VELO_TOL_COARSE)	

说明

该信号只用于同步运行中的副主轴。

说明

- **应用示例**
  在将工件从主主轴传送到副主轴时夹紧工件。只有在两个主轴达到充分同步时，PLC 用户程序才会触发工件的夹紧动作。

数据类型与更新

BOOL，循环

值 TRUE

通用耦合：副主轴与主主轴之间的位置差或速度差处于“粗同步”公差带内。

值 FALSE

通用耦合：副主轴与主主轴之间的位置差或速度差不处于“粗同步”公差带内。

关联

- <Axis>.spindle.in.syncModeActive
  主轴在同步运行模式下
- MD37200 $MA_COUPLE_POS_TOL_COARSE
  “粗同步运行”的阈值
- MD37220 $MA_COUPLE_VELO_TOL_COARSE
  “粗”速度公差

更多信息

- 轴和主轴功能手册；“同步主轴”章节

12.3.18.28 syncFine2Reached

NC → PLC	<Axis>.coupling.in.syncFine2Reached
通用耦合：实际位置闭环控制的同步运行达到“精公差范围”(MD37212 $MA_COUPLE_POS_TOL_FINE_2)	

数据类型与更新

BOOL，循环

值 TRUE

通用耦合：从动主轴与主动主轴之间的位置差或速度差处于“精同步运行 2”公差带内。

值 FALSE

通用耦合：从动主轴与主动主轴之间的位置差或速度差处于“精同步运行 2”公差带外。

关联

- <Axis>.spindle.in.syncModeActive
  主轴在同步运行模式下
- MD37212 $MA_COUPLE_POS_TOL_FINE_2
  “精同步运行 2”的阈值

更多信息

- 轴和主轴功能手册；“同步主轴”章节

12.3.18.29 syncFineReached

NC → PLC	<Axis>.coupling.in.syncFineReached
通用耦合：（位置或转速）实际值和设定值之差达到“精公差范围”(MD37210 $MA_COUPLE_POS_TOL_FINE, MD37230 $MA_COUPLE_VELO_TOL_FINE)	

说明

该信号只用于同步运行中的副主轴。

说明

- **应用示例**
  在将工件从主主轴传送到副主轴时夹紧工件。只有在两个主轴达到充分同步时，PLC 用户程序才会触发工件的夹紧动作。

数据类型与更新

BOOL，循环

值 TRUE

通用耦合：副主轴与主主轴之间的位置差或速度差处于“精同步”公差带内。

值 FALSE

通用耦合：副主轴与主主轴之间的位置差或速度差不处于“精同步”公差带内。

关联

- <Axis>.spindle.in.syncModeActive
  主轴在同步运行模式下
- MD37210 $MA_COUPLE_POS_TOL_FINE
  “精同步运行”的阈值
- MD37230 $MA_COUPLE_VELO_TOL_FINE
  “精”速度公差

更多信息

- 轴和主轴功能手册；“同步主轴”章节

12.3.18.30 syncSpindleCtrlMode

NC → PLC	<Axis>.coupling.in.syncSpindleCtrlMode
同步主轴耦合：根据主动主轴控制从动主轴： 0 - 设定值 1 - 实际值 (MD21310 $MC_COUPLING_MODE_1)	

说明

此信号仅对同步模式中生效的从动轴有意义。

说明

- **特殊情况或错误**
  如果在副主轴上出现的故障导致副主轴的“伺服使能”被取消，在一定前提条件下，控制系统内部会将副主轴与主主轴的耦合关系切换至实际值耦合。

数据类型与更新

BOOL，循环

值 TRUE

同步主轴耦合：实际值耦合生效

值 FALSE

同步主轴耦合：设定值耦合生效

关联

- <Axis>.spindle.in.syncModeActive
  主轴在同步运行模式下
- MD21310 $MC_COUPLING_MODE_1
  主轴同步运行时的耦合方式

更多信息

- 轴和主轴功能手册；“同步主轴”章节

12.3.19 <Axis>.coupling.out

12.3.19.1 enableFollowOverlay

PLC → NC	<Axis>.coupling.out.enableFollowOverlay
通用耦合：从动轴使能	

说明

引导轴与从动轴的被动同步需要此信号。

只要 <Axis>.coupling.out.enableFollowOverlay 信号赋值为 TRUE，通过 EGONSYN 选择的从动轴就会在扩展设备耦合组中同步运行。EG 耦合组中的模数轴会减小其模数位置值。这样便尽可能确保同步。

若未对从动轴作“从动轴叠加使能”，则不会运行至同步。取而代之地，程序在 EGONSYN 程序段处停止，并发出自清除的报警 16771。

数据类型与更新

BOOL，循环

值 TRUE

可为从动轴叠加一个额外的运行。

值 FALSE

不可叠加运行从动轴。

更多信息

- 轴和主轴功能手册；“轴耦合”章节

12.3.19.2 gantryDisableSync

PLC → NC	<Axis>.coupling.out.gantryDisableSync
龙门：锁定自动同步	

数据类型与更新

BOOL

值 TRUE

自动同步锁定龙门

值 FALSE

无作用

12.3.19.3 gantryStartSyncRun

PLC → NC	<Axis>.coupling.out.gantryStartSyncRun
龙门：启动同步运行	

数据类型与更新

BOOL

值 TRUE

启动龙门同步

值 FALSE

无作用

12.3.19.4 mcsCollisionProtection

PLC → NC	<Axis>.coupling.out.mcsCollisionProtection
MCS 耦合：启动轴对的碰撞防护	

数据类型与更新

BOOL，循环

值 TRUE

启用轴对碰撞防护

值 FALSE

未启用轴对碰撞防护

12.3.19.5 mcsDecouple

PLC → NC	<Axis>.coupling.out.mcsDecouple
MCS 耦合：请求禁用耦合	

数据类型与更新

BOOL，循环

值 TRUE

关闭耦合

值 FALSE

无作用

12.3.19.6 msActivate

PLC → NC	<Axis>.coupling.out.msActivate
主/从：激活耦合	

说明

可通过此接口信号，针对特定轴通/断主从耦合激活耦合。

通/断主从耦合时，必须满足以下条件：

- MD37262 = 0：主从耦合非永久生效。
- <Axis>.basic.in.posCtrlActive == TRUE：主动轴和从动轴必须处于位置闭环控制中。
- <Axis>.basic.in.isStationary == 1：主动轴和从动轴必须处于静止状态。
- <Chan>.basic.in.stateReset == TRUE：主动轴和从动轴的通道必须处于“复位”状态。

如果未满足上述条件中的任一项，则耦合不会被启用或关闭。不出现报警，保持原有耦合状态。如果后来满足了所有条件，则会根据信号状态启用或关闭耦合。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

已请求接通主从耦合。

脉冲沿切换 1 → 0

已请求关闭主从耦合。

关联

- <Chan>.basic.in.stateReset
  通道状态“复位”
- <Axis>.basic.in.isStationary
  轴达到静止转速范围 (n<n_min)
- <Axis>.basic.in.posCtrlActive
  位置控制器处于活动状态
- <Axis>.coupling.in.msActive
  主/从：耦合处于活动状态
- MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE
  永久性主从耦合

更多信息

- 功能手册之工艺

12.3.19.7 msTorqueComp

PLC → NC	<Axis>.coupling.out.msTorqueComp
主/从：请求接通转矩补偿控制器	

说明

可通过此变量，针对特定轴开/关转矩补偿控制器。

接通的前提条件：

- 附加转速设定值的接通已使能：
  MD37254 $MA_MS_TORQUE_CTRL_MODE[<从动轴>] = TRUE
- 已达到“精细”转速差
  <Axis>.coupling.in.msSpeedDifferenceFineReached == TRUE

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

已请求接通转矩补偿控制器。

脉冲沿切换 1 → 0

已请求关闭转矩补偿控制器。

关联

- <Axis>.coupling.in.msSpeedDifferenceFineReached
  主/从：转速差在精公差范围内 (MD37272 $MS_VELO_TOL_FINE)
- <Axis>.coupling.in.msCompCtrlActive
  主/从：转矩补偿控制器处于活动状态
- MD37254 $MA_MS_TORQUE_CTRL_MODE
  转矩补偿控制器互联

更多信息

- 功能手册之工艺

12.3.19.8 syncCorrection

PLC → NC	<Axis>.coupling.out.syncCorrection
通用耦合：请求同步补偿	

数据类型与更新

BOOL

值 TRUE

跟踪同步运行

值 FALSE

无作用

12.3.19.9 syncResetCorrection

PLC → NC	<Axis>.coupling.out.syncResetCorrection
通用耦合：删除同步补偿	

数据类型与更新

BOOL，循环

值 TRUE

删除同步运行补偿

值 FALSE

无作用

12.3.19.10 syncSpindleResync

PLC → NC	<Axis>.coupling.out.syncSpindleResync
同步主轴耦合：重新同步：恢复编程的偏移 ($P_COUP_OFFS)	

数据类型与更新

BOOL

值 TRUE

重新同步

值 FALSE

不重新同步

关联

- $P_COUP_OFFS

12.3.19.11 syncSpindleSyncDisable

PLC → NC	<Axis>.coupling.out.syncSpindleSyncDisable
同步主轴耦合：PLC 通过偏移编程锁定从动主轴的指定同步移动	

说明

当包含零件程序指令 COUPON(<FS>,<LS>,<Offset>) 的程序段切入主运行时，为从动轴处理接口信号 <Axis>.coupling.out.syncSpindleSyncDisable（锁定同步）：

- <Axis>.coupling.out.syncSpindleSyncDisable == TRUE 时，只创建速度恒定的同步运行。副主轴不会有额外的运动。随后的主轴同步与编写指令 COUPON(<副主轴>,<主主轴>) 类似。
- <Axis>.coupling.out.syncSpindleSyncDisable == FALSE 时，执行位置偏移。

数据类型与更新

BOOL，循环

值 TRUE

同步主轴耦合：由 PLC 禁用通过偏移编程设定的副主轴同步运动。

值 FALSE

同步主轴耦合：不由 PLC 禁用通过偏移编程设定的副主轴同步运动。

关联

- <Axis>.spindle.in.syncModeActive
  主轴在同步运行模式下
- <Axis>.coupling.in.syncCoarseReached
  通用耦合：（位置或转速）实际值和设定值之差达到“粗公差范围”(MD37200 $MA_COUPLE_POS_TOL_COARSE, MD37220 $MA_COUPLE_VELO_TOL_COARSE)
- <Axis>.coupling.in.syncFineReached
  通用耦合：（位置或转速）实际值和设定值之差达到“精公差范围”(MD37210 $MA_COUPLE_POS_TOL_FINE, MD37230 $MA_COUPLE_VELO_TOL_FINE)

更多信息

- 轴和主轴功能手册；“同步主轴”章节

12.3.20 <Axis>.plcCtrl.in

12.3.20.1 plcCtrlActive

NC → PLC	<Axis>.plcCtrl.in.plcCtrlActive
由 PLC 控制的轴：轴由 PLC 控制，适用于自主单轴操作	

数据类型和更新

BOOL，循环

值 TRUE

轴的控制已传递给 PLC。

值 FALSE

轴由 NC 控制。

关联

- <Axis>.plcCtrl.out.requestPlcCtrl
  请求通过适用于自主单轴操作的 PLC 进行控制
- 系统变量：$AA_SNGLAX_STAT
  单个轴的状态

更多信息

- 功能手册之轴和主轴；章节“定位轴”

12.3.20.2 plcCtrlPermanent

NC → PLC	<Axis>.plcCtrl.in.plcCtrlPermanent
轴始终由 PLC 控制 $MA_BASE_FUNCTION_MASK = 0x20 Bit5=1	

数据类型和更新

BOOL，循环

值 TRUE

该轴已固定分配为 PLC 轴。

值 FALSE

该轴未固定分配为 PLC 轴。

12.3.20.3 resetDone

NC → PLC	<Axis>.plcCtrl.in.resetDone
由 PLC 控制的轴：已执行轴复位	

说明

复位状态：

- 轴的机床数据经重新载入。
- 轴状态处于“单轴复位中”：
  $AA_SNGLAX_STAT == TRUE
- <Axis>.plcCtrl.in.stopped（由 PLC 控制的轴：轴已停止）== FALSE

数据类型和更新

BOOL，循环

值 TRUE

PLC 控制的轴处于复位状态。

值 FALSE

PLC 控制的轴不处于复位状态。

关联

- <Axis>.plcCtrl.out.reset
  PLC 控制轴：请求复位
- <Axis>.plcCtrl.in.stopped
  由 PLC 控制的轴：轴已停止
- 系统变量：$AA_SNGLAX_STAT
  单个轴的状态
- OPI 变量：aaSnglAxStat

更多信息

- 功能手册之轴和主轴；章节“定位轴”

12.3.20.4 stopped

NC → PLC	<Axis>.plcCtrl.in.stopped
由 PLC 控制的轴：轴已停止	

说明

轴停止已激活。在停止后，轴处于“单轴中断”状态：

$AA_SNGLAX_STAT == 3

数据类型和更新

BOOL，循环

值 TRUE

PLC 控制的轴已由于<Axis>.plcCtrl.out.stop = TRUE 而停止。

值 FALSE

PLC 控制的轴未停止。

关联

- <Axis>.plcCtrl.out.stop
  PLC 控制轴：沿着制动斜坡停止轴
- <Axis>.plcCtrl.out.requestPlcCtrl
  请求通过适用于自主单轴操作的 PLC 进行控制
- <Axis>.basic.in.posCoarseReached
  已在“粗准停”限值内到达编程位置 (MD36000 $MA_STOP_LIMIT_COARSE)
- <Axis>.basic.in.posFineReached
  已在“精准停”限值内到达编程位置 (MD36150 $MA_STOP_LIMIT_FINE)
- 系统变量：$AA_SNGLAX_STAT
  单个轴的状态
- OPI 变量：aaSnglAxStat

更多信息

- 功能手册之轴和主轴；章节“定位轴”

12.3.21 <Axis>.plcCtrl.out

12.3.21.1 chanAssignmentSelection

PLC → NC	<Axis>.plcCtrl.out.assignToChanSelection
Assign NC axis to channel (channel number or 0:neutral axis)	

说明

轴/主轴需要分配至的通道的编号

数据类型和更新

INT, 循环

值

轴/主轴需要分配至的通道的编号

关联

- <Axis>.interchange.in.requested
  已请求轴/主轴交换
- <Axis>.interchange.in.possible
  可进行轴/主轴交换
- <Axis>.interchange.in.neutralAxis
  中性轴/主轴
- <Axis>.plcCtrl.in.spindleCrtlActive
  轴是"PLC 轴/主轴"
- <Axis>.interchange.in.assignedChanNumber
  已分配给 NC 轴的通道编号（通道编号或 0：中性轴）
- MD20070 $MC_AXCONF_MACHAX_USED
  通道中生效的机床轴编号
- MD30550 $MA_AXCONF_ASSIGN_MASTER_CHAN
  用于轴交换的通道清除设置

更多信息

- 功能手册之基本功能；章节"跨通道的轴交换"

12.3.21.2 requestPlcCtrl

PLC → NC	<Axis>.plcCtrl.out.requestPlcCtrl
请求通过适用于自主单轴操作的 PLC 进行控制	

说明

NC 的响应：

- 检查轴是否为主处理轴或中立轴。
- 检查是否允许由 PLC 控制另一根轴。
- 确认传递：
  - <Axis>.plcCtrl.in.plcCtrlActive（由 PLC 控制的轴：轴由 PLC 控制，适用于自主单轴操作）== TRUE
  - $AA_SNGLAX_STAT == TRUE
- 结果：PLC 控制轴。

数据类型和更新

BOOL，循环

值 TRUE

将轴的控制传递给 PLC。

值 FALSE

未请求将轴的控制传递给 PLC。

关联

- <Axis>.plcCtrl.in.plcCtrlActive
  由 PLC 控制的轴：轴由 PLC 控制，适用于自主单轴操作
- 系统变量：$AA_SNGLAX_STAT
  单个轴的状态
- OPI 变量：aaSnglAxStat

更多信息

- 功能手册之轴和主轴；章节“定位轴”

12.3.21.3 resume

PLC → NC	<Axis>.plcCtrl.out.resume
PLC 控制轴：请求继续执行已中断的运动	

说明

借助此信号，可继续执行处于“单轴中断”状态（$AA_SNGLAX_STAT = 3）的由 PLC 控制的轴的运动。

NC 的响应：

- 检查是否存在带"CANCELCLEAR"或"NCSTARTCLEAR"清除条件的轴专用报警。如存在，则清除此报警。
- 检查是否能够继续执行轴的运动？如果是，则轴将转换至“单轴生效”状态：$AA_SNGLAX_STAT == 4
- 确认执行：
  - <Axis>.basic.in.posCoarseReached（已在“粗准停”限值内到达编程位置） == FALSE
  - <Axis>.basic.in.posFineReached（已在“精准停”限值内到达编程位置） == FALSE
  - <Axis>.plcCtrl.in.stopped（由 PLC 控制的轴：轴已停止）== FALSE
  - <Axis>.basic.in.traversMinusCommandActive（负向的移动指令生效） == TRUE
  - <Axis>.basic.in.traversPlusCommandActive（正向的移动指令生效） == TRUE
- 结果：继续执行轴的运动。

说明

在以下情形下，此信号会被忽略：
- 轴不由 PLC 控制。
- 轴不处于“单轴中断”状态。
- 由于存在报警，不能继续执行轴的运动。

数据类型和更新

BOOL，循环

脉冲沿切换 0 → 1

PLC 控制的轴：请求继续执行中断的运动。

脉冲沿切换 1 → 0

无作用。

关联

- <Axis>.plcCtrl.out.stop
  PLC 控制轴：沿着制动斜坡停止轴
- <Axis>.plcCtrl.out.requestPlcCtrl
  请求通过适用于自主单轴操作的 PLC 进行控制
- <Axis>.plcCtrl.in.plcCtrlActive
  由 PLC 控制的轴：轴由 PLC 控制，适用于自主单轴操作
- <Axis>.basic.in.posCoarseReached
  已在“粗准停”限值内到达编程位置 (MD36000 $MA_STOP_LIMIT_COARSE)
- <Axis>.basic.in.posFineReached
  已在“精准停”限值内到达编程位置 (MD36150 $MA_STOP_LIMIT_FINE)
- <Axis>.plcCtrl.in.stopped
  由 PLC 控制的轴：轴已停止
- <Axis>.basic.in.traversMinusCommandActive
  负向的移动指令生效 (MD17900 $MN_VDI_FUNCTION_MASK)
- <Axis>.basic.in.traversPlusCommandActive
  正向的移动指令生效 (MD17900 $MN_VDI_FUNCTION_MASK)
- 系统变量：$AA_SNGLAX_STAT
  单个轴的状态
- OPI 变量：aaSnglAxStat

更多信息

- 功能手册之轴和主轴；章节“定位轴”

12.3.21.4 stop

PLC → NC	<Axis>.plcCtrl.out.stop
PLC 控制轴：沿着制动斜坡停止轴	

说明

借助此信号可将由 PLC 控制的轴的运动停止。

NC 的响应：

- 轴沿其加速度特性曲线制动至静止状态。
- 轴转换至“单轴中断”状态：
  $AA_SNGLAX_STAT == 3
- 确认执行：
  - <Axis>.basic.in.posCoarseReached（已在“粗准停”限值内到达编程位置） == TRUE
  - <Axis>.basic.in.posFineReached（已在“精准停”限值内到达编程位置） == TRUE
  - <Axis>.plcCtrl.in.stopped（由 PLC 控制的轴：轴已停止）== FALSE
  - <Axis>.basic.in.traversMinusCommandActive（负向的移动指令生效） == FALSE
  - <Axis>.basic.in.traversPlusCommandActive（正向的移动指令生效） == FALSE
- 结果：轴停止。

说明

从动轴的运动只能通过停止主动轴来中断。
通过“扩展停止和退回”功能触发的回退运动无法停止。

数据类型和更新

BOOL，循环

值 TRUE

PLC 控制的轴：已请求通过制动斜坡对轴进行制动。

值 FALSE

PLC 控制的轴：未请求通过制动斜坡对轴进行制动。

关联

- <Axis>.oscillation.out.stopAtNextReversalPoint
  在下一个换向点停止
- <Axis>.plcCtrl.out.requestPlcCtrl
  请求通过适用于自主单轴操作的 PLC 进行控制
- <Axis>.basic.in.posCoarseReached
  已在“粗准停”限值内到达编程位置 (MD36000 $MA_STOP_LIMIT_COARSE)
- <Axis>.basic.in.posFineReached
  已在“精准停”限值内到达编程位置 (MD36150 $MA_STOP_LIMIT_FINE)
- <Axis>.plcCtrl.in.stopped
  由 PLC 控制的轴：轴已停止

- <Axis>.basic.in.traversMinusCommandActive
  负向的移动指令生效 (MD17900 $MN_VDI_FUNCTION_MASK)
- <Axis>.basic.in.traversPlusCommandActive
  正向的移动指令生效 (MD17900 $MN_VDI_FUNCTION_MASK)
- 系统变量：$AA_SNGLAX_STAT
  单个轴的状态
- OPI 变量：aaSnglAxStat

更多信息

- 功能手册之轴和主轴；章节“定位轴”

12.3.22 NCK_typeChan

下表说明了数据类型 NCK_typeChan 的含义：

结构	说明
.config	通道的组态数据
.registered	通道已注册
.chanActive	通道已生效
.chanNumber	通道编号
.basic.in (页 675)	来自通道的通用信号
.basic.out (页 726)	发送至通道的通用信号
.gCommandGroup.in (页 765)	输入的 G 指令组信号
.geoAxis1.in ... .geoAxis3.in (页 766)	几何轴的输入信号
.geoAxis1.out ... .geoAxis3.out (页 779)	几何轴的输出信号
.grinding.in (页 865)	来自通道的特定磨削信号
.grinding.out (页 866)	发送至通道的特定磨削信号
.orientationAxis1.in ... .orientationAxis3.in (页 796)	定向轴的输入信号

结构	说明
.orientationAxis1.out ... .orientationAxis3.out (页 808)	定向轴的输出信号
.toolMgmt.in (页 825)	刀具管理的输入信号
.toolMgmt.out (页 830)	针对刀具管理的输出信号
.auxMDynFuncs.in (页 834)	动态辅助功能解码的输入信号
.auxMFuncs.in (页 835)	辅助功能的输入信号 [1..5]
.auxSFuncs.in (页 838)	S 功能的输入信号 [1..3]
.auxTFuncs.in (页 841)	T 功能的输入信号 [1..3]
.auxDFuncs.in (页 844)	D 功能的输入信号 [1..3]
.auxDLFuncs.in (页 846)	DL 功能的输入信号 [1..1]
.auxHFuncs.in (页 848)	帮助功能的输入信号 [1..3]
.auxFFuncs.in (页 852)	F 功能的输入信号 [1..6]
.protAreas.in (页 855)	保护范围的输入信号 [1..10]
.protAreas.out (页 859)	保护范围的输出信号 [1..10]
.nibbelingAndPunching.in (页 868)	步冲和冲压操作的输入信号
.nibbelingAndPunching.out (页 869)	步冲和冲压操作的输出信号
.cycles.in (页 873)	输入的循环信号
.cycles.out (页 874)	输出的循环信号

结构	说明
.progEvent.in (页 861)	输入的 PROG_EVENT 信号
.synActs.in (页 864)	同步动作的输入信号
.synActs.out (页 865)	同步动作的输出信号

12.3.23 <Chan>.basic.in

12.3.23.1 allAxesReferenced

NC → PLC	<Chan>.basic.in.allAxesReferenced
通道的所有轴（直线轴和回转轴）都已回参考点	

说明

通道中的主轴对该接口信号没有影响。

数据类型与更新

BOOL，循环

值 TRUE

通道中所有需要回参考点的轴（直线轴和回转轴）均已回参考点。

机床数据：
MD20700 $MC_REFP_NC_START_LOCK（无参考点的 NC 启动禁止）
为零。

若一根轴上连接了两个位置测量系统来避免 NC 启动，那么生效的测量系统已执行回参考点时该轴才作为已回参考点。
只有该信号存在时，NC 启动才能用于零件程序的执行。

当 MD34110 $MA_REFP_CYCLE_NR _ = -1，
且轴不处于驻留位置时（位置测量系统未生效且伺服使能取消），
需要为轴执行回参考点。

值 FALSE

通道中一个或多个应回参考点的进给轴未回参考点。

关联

- <Axis>.basic.in.enc1Synchronized
  机床轴位置测量系统 1 已回参考点/同步
- <Axis>.basic.in.enc2Synchronized
  机床轴位置测量系统 2 已回参考点/同步

更多信息

- 功能手册之进给轴和主轴；章节“回参考点”

12.3.23.2 allAxesStationary

NC → PLC	<Chan>.basic.in.allAxesStationary
通道的所有轴都停止，且无继续运行指令。	

说明

“轴停止”的标准插补结束

数据类型与更新

BOOL，循环

值 TRUE

通道的所有停止，并且不存在继续运行。

值 FALSE

通道的轴并非全部停止，或者存在继续运行。

更多信息

- 基础功能手册；“连续轨迹控制运行、准停、预读”>“准停模式”章节

12.3.23.3 asupActive

NC → PLC	<Chan>.basic.in.asupActive
异步子程序 (ASUB) 处于活动状态	

说明

激活通道中的异步子程序 (ASUP)。

该信号表示在 NC 通道中有一个异步子程序处于激活状态。通过指令“SINU_RequestAsup”在通道中启动 ASUP 后，该信号也将进行输出。

数据类型和更新

BOOL，循环

值 TRUE

有 ASUB 生效。

值 FALSE

无 ASUB 生效。

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”>”异步子程序 (ASUB)”章节

12.3.23.4 asupSilentActive

NC → PLC	<Chan>.basic.in.asupSilentActive
“静止”异步子程序处于活动状态：程序段由于 DISPLOF 不显示在屏幕上	

说明

“静止”ASUB 生效

数据类型与更新

BOOL，循环

值 TRUE

有“静止”ASUB 生效。

值 FALSE

无“静止”ASUB 生效。

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”>”异步子程序 (ASUB)”章节

12.3.23.5 asupStopStateActive

NC → PLC	<Chan>.basic.in.asupStopStateActive
在程序完全运行完毕前，控制器停止运行异步子程序	

说明

此接口信号仅在以下状态下处理：

- AUTO 或 MDI 运行方式
- 通道状态“中断”
- 程序状态“停止”

数据类型与更新

BOOL，循环

值 TRUE

ASUB 处于“停止”状态下。

说明

ASUB 已由控制系统在完全执行前停止。

值 FALSE

ASUB 不处于“停止”状态下。

说明

出现以下事件时，此信号总是被置 0：
- NC 启动
- 通道复位

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”>“异步子程序 (ASUB)”>“功能”>“包含 REPOSA 的异步子程序”章节

12.3.23.6 blockSearchActionActive

NC → PLC	<Chan>.basic.in.blockSearchActionActive
程序段搜索：动作程序段处于活动状态/正在运行	

数据类型与更新

BOOL，循环

值 TRUE

动作程序段生效/正在执行。

值 FALSE

无动作程序段生效。

关联

- <Chan>.basic.in.blockSearchLastActionActive
  程序段搜索：最后一个动作程序段处于活动状态/正在运行
- <Chan>.basic.in.blockSearchActive
  程序段搜索：数据采集处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”>“程序段搜索类型 1、2 和 4”章节

12.3.23.7 blockSearchActive

NC → PLC	<Chan>.basic.in.blockSearchActive
程序段搜索：数据采集处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

“程序段搜索”功能处于活动状态。
该功能已通过操作界面选择，并通过接口信号 <Chan>.basic.out.ncStart（NC 启动）启动。

值 FALSE

“程序段搜索”功能未生效，或者已找到搜索目标。

关联

- <Chan>.basic.out.ncStart
  脉冲沿切换 0 → 1：请求开始运行在通道中选择的程序

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”>“程序段搜索类型 1、2 和 4”或“程序段搜索类型 5 (SERUPRO)”章节

12.3.23.8 blockSearchApproachActive

NC → PLC	<Chan>.basic.in.blockSearchApproachActive
程序段搜索：“在轮廓处计算的程序段搜索”接近程序段处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

在程序段搜索类型 2“在轮廓处计算的程序段搜索”中，用于继续执行 NC 程序的定位程序段生效。

值 FALSE

无定位程序段生效。

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.9 blockSearchLastActionActive

NC → PLC	<Chan>.basic.in.blockSearchLastActionActive
程序段搜索：最后一个动作程序段处于活动状态/正在运行	

说明

如果最后一个动作程序段处于活动状态，意味着 NC 方面的所有动作程序段均已执行，并可由 PLC 或操作人员执行动作，比如溢出转存、将运行方式切换至 JOG/REPOS 等。例如，PLC 在运行开始之前还可以执行换刀。

动作程序段包含“进行计算的程序段搜索”期间收集的动作，例如：

- 辅助功能输出：H、M00、M01、M..
- 刀具编程：T、D、DL
- 主轴编程：S 值、M3/M4/M5/M19、SPOS
- 进给率编程 F

数据类型和更新

BOOL，循环

值 TRUE

最后的动作程序段生效/正在执行。

值 FALSE

最后的动作程序段未生效。

关联

- <Chan>.basic.in.blockSearchActive
  程序段搜索：数据采集处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.10 blockSearchSeruproActive

NC → PLC	<Chan>.basic.in.blockSearchSeruproActive
通过程序测试进行程序段搜索 (SERUPRO)：在查找到搜索的程序段前，信号处于活动状态，“停止”后程序状态切换	

说明

程序段搜索 SERUPRO 只能在 AUTO 运行方式下在程序“取消”状态下激活。

数据类型与更新

BOOL，循环

值 TRUE

通过程序测试进行的程序段搜索生效

在程序段搜索范围内执行程序段期间（内部通道状态“程序测试”），直至目标程序段切换至主处理（程序状态“停止”）为止，此接口信号置位。

值 FALSE

通过程序测试进行的程序段搜索未生效

在目标程序段切换至主处理时（内部通道状态“程序测试”已取消；显示停止条件：“找到搜索目标”），此接口信号复位。

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.11 circularJogTraversActive

NC → PLC	<Chan>.basic.in.circularJogTraversActive
“点动圆弧运行”处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

“点动圆弧运行”处于活动状态。

值 FALSE

“点动圆弧运行”未激活。

关联

- <Chan>.basic.out.circularJogTravers
  点动圆弧运行

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.23.12 clcActive

NC → PLC	<Chan>.basic.in.clcActive
间距控制 (CLC) 处于活动状态（编译循环）	

数据类型与更新

BOOL，循环

值 TRUE

间距调节（CLC）生效。

值 FALSE

间距调节（CLC）未生效

关联

- <Chan>.basic.out.deactivateClc
  Abstandsregelung (CLC):请求禁用（类似程序指令 CLC_GAIN=0.0）
- <Chan>.basic.out.clcOvr
  间距控制 (CLC)：请求使用通道专用进给倍率
- <Chan>.basic.in.clcStopAtLowerLimitActive
  间距控制 (CLC)：轴运行达到运动下限，已停止 (MD62505 $MC_CLC_SENSOR_LOWER_LIMIT)
- <Chan>.basic.in.clcStopAtUpperLimitActive
  间距控制 (CLC)：轴运行达到运动上限，已停止 (MD62506 $MC_CLC_SENSOR_UPPER_LIMIT)
- <Chan>.basic.in.clcStopAtLowerLimitActive
  间距控制 (CLC)：轴运行达到运动下限，已停止 (MD62505 $MC_CLC_SENSOR_LOWER_LIMIT)
- <Chan>.basic.in.clcStopAtUpperLimitActive
  间距控制 (CLC)：轴运行达到运动上限，已停止 (MD62506 $MC_CLC_SENSOR_UPPER_LIMIT)

更多信息

- 监控和补偿功能手册；“间距控制”章节

12.3.23.13 clcStopAtLowerLimitActive

NC → PLC	<Chan>.basic.in.clcStopAtLowerLimitActive
间距控制 (CLC)：轴运行达到运动下限，已停止 (MD62505 $MC_CLC_SENSOR_LOWER_LIMIT)	

说明

间距调节的运动下限在以下机床数据中设置：

MD62505 $MC_CLC_SENSOR_LOWER_LIMIT

可通过在零件程序中编写 `CLC_LIM(...)` 针对程序段调整设置的限值。

提示

如果除了 <Chan>.basic.in.clcStopAtLowerLimitActive，还同时设置了 <Chan>.basic.in.clcStopAtUpperLimitActive，这种情况下信号如下生效：“间距控制 (CLC)：运动处于静止状态”。

数据类型与更新

BOOL，循环

值 TRUE

受间距调节的轴的因间距调节而产生的运行达到运动下限，并且停止。

值 FALSE

受间距调节的轴的因间距调节而产生的运行未达到运动下限。

关联

- `CLC_LIM(...)`：限制控制范围
- <Chan>.basic.in.clcStopAtUpperLimitActive
  间距控制 (CLC)：轴运行达到运动上限，已停止 (MD62506 $MC_CLC_SENSOR_UPPER_LIMIT)
- <Chan>.basic.in.clcStopAtLowerLimitActive
  间距控制 (CLC)：轴运行达到运动下限，已停止 (MD62505 $MC_CLC_SENSOR_LOWER_LIMIT)
- MD62505 $MC_CLC_SENSOR_LOWER_LIMIT
  间距控制的运动下限

更多信息

- 监控和补偿功能手册；“间距控制”章节

12.3.23.14 clcStopAtUpperLimitActive

NC → PLC	<Chan>.basic.in.clcStopAtUpperLimitActive
间距控制 (CLC)：轴运行达到运动上限，已停止 (MD62506 $MC_CLC_SENSOR_UPPER_LIMIT)	

说明

间距调节的运动上限在以下机床数据中设置：

MD62506 $MC_CLC_SENSOR_UPPER_LIMIT

可通过在零件程序中编写 `CLC_LIM(...)` 针对程序段调整设置的限值。

提示

如果除了 <Chan>.basic.in.clcStopAtLowerLimitActive，还同时设置了 <Chan>.basic.in.clcStopAtUpperLimitActive，这种情况下信号如下生效：“间距控制 (CLC)：运动处于静止状态”。

数据类型与更新

BOOL，循环

值 TRUE

受间距调节的轴的因间距调节而产生的运行达到运动上限，并且停止。

值 FALSE

受间距调节的轴的因间距调节而产生的运行未达到运动上限。

关联

- `CLC_LIM(...)`；限制控制范围
- <Chan>.basic.in.clcStopAtLowerLimitActive
  间距控制 (CLC)：轴运行达到运动下限，已停止 (MD62505 $MC_CLC_SENSOR_LOWER_LIMIT)
- <Chan>.basic.in.clcStopAtUpperLimitActive
  间距控制 (CLC)：轴运行达到运动上限，已停止 (MD62506 $MC_CLC_SENSOR_UPPER_LIMIT)
- MD62506 $MC_CLC_SENSOR_UPPER_LIMIT
  间距控制的运动上限

更多信息

- 监控和补偿功能手册；“间距控制”章节

12.3.23.15 collisionStopActive

NC → PLC	<Chan>.basic.in.collisionStopActive
防撞保护已触发“停止通道中的运行”	

数据类型与更新

BOOL，循环

值 TRUE

碰撞监测触发了通道中的运行停止。

值 FALSE

碰撞监测未触发通道中的运行停止。

更多信息

- 监控和补偿功能手册；“防撞保护”章节

12.3.23.16 configuredStopAtBlockEnd

NC → PLC	<Chan>.basic.in.configuredStopAtBlockEnd
在程序段结尾由于组态的停止而停止	

数据类型与更新

BOOL

值 TRUE

在程序段结尾由于组态的停止而停止

12.3.23.17 configuredStopFuncActive

NC → PLC	<Chan>.basic.in.configuredStopFuncActive
组态的停止激活	

说明

已组态的停止 (CST) 已激活。

已在 NC 中激活“已组态的停止”功能。
调试期间通过设定数据确定的停止状态适用。

数据类型和更新

BOOL，循环

值 TRUE

组态的停止激活

值 FALSE

组态的停止未激活

关联数据：

- 设置选项“已组态的停止”（需要许可证）
- 通过 MD52210 Bit18 激活 SINUMERIK Operate 中的显示
- <Chan>.basic.out.configuredStopFunc
  激活已组态的停止

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.18 contourHandwheelAssignedNumber

NC → PLC	<Chan>.basic.in.contourHandwheelAssignedNumber
当前所分配手轮的编号：1..6 或 0 未分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）	

说明

当前所分配轮廓手轮的编号。

该接口使用了二进制编码。通过机床数据 MD11324 进行定义。

数据类型和更新

USInt，循环

值

0：未分配手轮

1...6：当前所分配手轮的编号

关联

- MD11324 $MN_HANDWH_VDI_REPRESENTATION
  VDI 接口中的手轮编号显示
- <Chan>.basic.out.contourHandwheelSelection
  手轮 1..6 已分配或 0 未分配手轮（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.23.19 contourHandwheelDirInverted

NC → PLC	<Chan>.basic.in.contourHandwheelDirInverted
轮廓手轮旋转方向反转处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

轮廓手轮旋转方向的取反生效。

值 FALSE

轮廓手轮旋转方向的取反未生效。

关联

- <Chan>.basic.out.contourHandwheelInvertDir
  请求反转所选轮廓手轮的旋转方向

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.23.20 disableReadInIgnored

NC → PLC	<Chan>.basic.in.disableReadInIgnored
读取禁止被忽略	

说明

下列情况下，忽略读入锁定 <Chan>.basic.out.disableReadIn：

<Chan>.basic.out.disableReadIn == TRUE 且当前程序段具有“读入锁定无效”状态

下列情况下，不忽略读入锁定 <Chan>.basic.out.disableReadIn：

<Chan>.basic.out.disableReadIn == FALSE 或

(<Chan>.basic.out.disableReadIn) == TRUE 且当前程序段具有“读入锁定有效”状态

忽略读取禁止的程序段具有“读取禁止无效”状态。

通过以下机床数据设置忽略读入锁定 <Chan>.basic.out.disableReadIn：

- MD11602 $MN_ASUP_START_MASK，位 2
- MD20116 $MC_IGNORE_INHIBIT_ASUP
- MD20107 $MC_ PROG_EVENT_IGN_INHIBIT

数据类型与更新

BOOL，循环

值 TRUE

忽略读入锁定 <Chan>.basic.out.disableReadIn。

值 FALSE

不忽略读入锁定 <Chan>.basic.out.disableReadIn。

关联

- <Chan>.basic.in.singleBlockStopIgnored
  单段 (SBL) 时忽略程序段结尾停止
- MD11602 $MN_ASUP_START_MASK, Bit 2
  读入锁定处于活动状态时也允许启动
- MD20116 $MC_IGNORE_INHIBIT_ASUP
  虽然读入锁定有效，但仍执行中断程序
- MD20107 $MC_ PROG_EVENT_IGN_INHIBIT
  程序事件忽略读入锁定

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.21 dryRunFeedrateActive

NC → PLC	<Chan>.basic.in.dryRunFeedrateActive
试运行进给率处于活动状态	

数据类型与更新

BOOL

值 TRUE

试运行进给率处于活动状态

值 FALSE

试运行进给率未激活

12.3.23.22 executionFromExternalActive

NC → PLC	<Chan>.basic.in.executionFromExternalActive
外部执行处于活动状态：（子）零件程序直接由闪存驱动器等运行	

说明

直接在闪存驱动器等外部驱动器上运行零件程序（及子程序）。

数据类型与更新

BOOL

值 TRUE

从外部数据源运行处于活动状态。

值 FALSE

从外部数据源运行未激活。

12.3.23.23 externalProgLanguageModeActive

NC → PLC	<Chan>.basic.in.externalProgLanguageModeActive
外部编程语言运行方式处于活动状态	

数据类型与更新

BOOL

值 TRUE

外部编程语言模式处于活动状态

值 FALSE

外部编程语言模式未激活

12.3.23.24 handwheelOvrActive

NC → PLC	<Chan>.basic.in.handwheelOvrActive
“手轮超驰自动模式”功能处于活动状态：针对编程的轨迹轴的附加进给倍率	

说明

第 1 几何轴的手轮脉冲作为速度叠加对编程的轨迹进给率起作用。

在下面的条件下该叠加失效：

- 轨迹轴到达程序编写的目标位置。
- 删除了剩余行程。
- 触发了复位。

数据类型与更新

BOOL，循环

值 TRUE

功能“自动方式下的手轮叠加”对所编程的轨迹轴生效。

值 FALSE

功能“自动方式下的手轮叠加”未对所编程的轨迹轴生效。

关联

- <Chan>.basic.out.deleteDistanceToGo
  脉冲沿切换 0 → 1：已请求删除剩余行程（通道专用剩余行程）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.23.25 interruptProcessingActive

NC → PLC	<Chan>.basic.in.interruptProcessingActive
中断加工处于活动状态	

说明

在程序运行方式（AUTO 或 MDI）下执行中断处理时，此接口信号不置位。

数据类型与更新

BOOL，循环

值 TRUE

中断处理生效。

生效的中断程序使得一个或多个运行方式组通道不处于所需的运行方式下。

值 FALSE

中断处理未生效。

所有通道均处于请求的运行方式下。

关联

- MD11600 $MN_BAG_MASK
  定义运行方式组行为

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.26 jogRetractActive

NC → PLC	<Chan>.basic.in.jogRetractActive
点动退刀：已选，处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

手动退刀被选中且生效。

值 FALSE

手动退刀未被选中。

关联

- <Chan>.basic.in.jogRetractDataAvailable
  点动退刀：退刀数据用于沿刀具方向的运行

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.23.27 jogRetractDataAvailable

NC → PLC	<Chan>.basic.in.jogRetractDataAvailable
点动退刀：退刀数据用于沿刀具方向的运行	

说明

在信号状态生效时，可以选择手动退刀（操作界面或 PI 服务"RETRAC"）。

数据类型与更新

BOOL，循环

值 TRUE

存在沿刀具中心轴方向退刀的数据。

值 FALSE

不存在沿刀具中心轴方向退刀的数据。手动退刀无法被选中。

关联

- <Chan>.basic.in.jogRetractActive
  点动退刀：已选，处于活动状态

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.23.28 m00m01Active

NC → PLC	<Chan>.basic.in.m00m01Active
M00 / M01：“编程的停止/有条件停止”处于活动状态	

数据类型和更新

BOOL，循环

值 TRUE

辅助功能 M00 或 M01（程序停止/有条件停止）处于活动状态。

编写了辅助功能的程序段已执行，辅助功能已输出。

程序状态“停止”。

值 FALSE

辅助功能 M00 或 M01 未生效。

关联

- <Chan>.basic.out.m01
  请求激活程序控制“有条件停止”(M01)
- 已选 <HmiChan>.basic.in.m01SelectedM01
  HMI：已请求程序控制 M01“有条件停止”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.29 m00m01AssociatedAuxFuncActive

NC → PLC	<Chan>.basic.in.m00m01AssociatedAuxFuncActive
M00 / M01：相应的辅助功能“编程的停止/有条件停止”处于活动状态	

说明

M00 / M01：相应的辅助功能“编程的停止/有条件停止”处于激活状态。

在 SINUMERIK Operate 操作界面的“自动”>“程序控制”操作区域将 HMI/PLC 接口信号 <HmiChan>.basic.in.m01AssociatedAuxFuncSelected 置位，可以选择“关联辅助功能”(M-1)。

必须在 PLC 用户程序中进行信号的后续处理。

数据类型和更新

BOOL，循环

值 TRUE

“关联辅助功能”生效。

值 FALSE

关联辅助功能未生效。

关联

- MD22254 $MC_AUXFU_ASSOC_M0_VALUE
  用于程序停止的附加 M 功能
- MD22256 $MC_AUXFU_ASSOC_M1_VALUE
  用于有条件停止的附加 M 功能

- <HmiChan>.basic.in.m01AssociatedAuxFuncSelected
  HMI：已请求程序控制 M01"相关辅助功能"
- <Chan>.basic.out.m01AssociatedAuxFunc
  请求激活 M0/M1"关联辅助功能"

更多信息

- 基础功能手册；"辅助功能输出至 PLC"章节

12.3.23.30 m02m30Active

NC → PLC	<Chan>.basic.in.m02m30Active
M02 / M30："程序结束"处于活动状态	

说明

- 在 NC 程序的最后的零件程序段中，不允许编写以下功能：
  - 触发读入停止的辅助功能
  - 编写的越过程序结束（M02 / M30）生效的主轴转速（S 值）
- 完成程序结束复位 (M02 / M30) 后，将信号设为 TRUE 值。在执行后续 PROG_EVENT 程序期间，信号设置为 FALSE，PROG_EVENT 程序结束后重新设为 TRUE。

说明

此信号不适合作为用于触发自动执行功能（如工件计数、棒料进给等）的触发器。为识别之前的执行（NC 程序、ASUB、PROG_EVENT 等），必须在独立的零件程序段中写入程序结束指令（M02 / M30）。可将指令（M02 / M30）或 M 功能的解码信号用作触发器。

数据类型与更新

BOOL，循环

值 TRUE

程序结束 M02 / M30 生效。

接口信号在以下情形后置位：

- 程序复位（M02、M30 或 M17）结束
  提示：若在含 M02、M30 或 M17 的程序段中编写了运行，则在到达所有轴的目标位置后，信号才置位。
- 在以下情形后执行 PROG_EVENT 程序（PROG_EVENT.SPF）：
  – 程序结束复位（M02 / M30）
  – 通道复位
  – 热启动（上电）
  – 程序段搜索后输出最后的动作程序段

值 FALSE

程序结束 M02 / M30 未生效。

接口信号复位或保持复位：

- 在启动后，以及在 NC 程序、异步子程序（ASUB）或 PROG_EVENT 程序执行期间。
  提示：
  启动在“程序测试”模式计算的程序段搜索 (SERUPRO) 后，保留信号的最后状态。
- 在程序因报警而终止后。
- 在不执行 PROG_EVENT 程序的通道复位期间及之后。
- 在不执行 PROG_EVENT 程序的控制系统启动期间及之后。

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”>“程序运行”>“事件控制的程序调用 (PROG_EVENT)”章节

12.3.23.31 ncAlarmActive

NC → PLC	<Chan>.basic.in.ncAlarmActive
通道内至少有一个 NC 报警处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

通道中至少存在一个 NC 报警。

值 FALSE

通道中不存在 NC 报警。

关联

- <Chan>.basic.in.ncAlarmWithStopActive
  通道内至少有一个会触发加工停止的 NC 报警处于活动状态
- <Nc>.basic.in.ncAlarmActive
  至少有一个 NC 报警处于活动状态

更多信息

- 诊断说明

12.3.23.32 ncAlarmWithProgStopActive

NC → PLC	<Chan>.basic.in.ncAlarmWithProgStopActive
通道内至少有一个触发程序停止的 NC 报警处于活动状态（程序进程被锁定）	

数据类型与更新

BOOL，循环

值 TRUE

当程序因出现报警而中止执行时，会输出接口信号。

这包括所有会产生停止响应的报警：

- 在轨迹上停止并删除就绪信号
- 轴停止
- 直接在轨迹上停止
- 编译器停止，插补缓冲器仍然执行
- 在程序段结束处停止

此外还应注意具有以下报警响应的报警：

- 在此通道中禁止 NC 启动
- 在此通道中禁止 NC 启动，可能允许异步子程序启动。

对于带有这些响应的报警，只有当报警仍未应答并触发了 NC 启动时，接口信号才会被置位。在此情形下，执行 NC 启动时接口信号才会被置位。

当在该通道中因为 NC 启动禁止而拒绝启动 ASUP 时，接口信号会被置位。

只要无激活信号的报警响应生效时，接口信号就会被删除。这取决于报警的删除条件。

值 FALSE

通道中不存在导致程序停止的 NC 报警。

关联

- <Nc>.basic.in.ncAlarmActive
  至少有一个 NC 报警处于活动状态
- <Chan>.basic.in.ncAlarmActive
  通道内至少有一个 NC 报警处于活动状态
- <Chan>.basic.in.ncAlarmWithStopActive
  通道内至少有一个会触发加工停止的 NC 报警处于活动状态

更多信息

- 诊断手册

12.3.23.33 ncAlarmWithStopActive

NC → PLC	<Chan>.basic.in.ncAlarmWithStopActive
通道内至少有一个会触发加工停止的 NC 报警处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

通道中至少存在一个导致加工停止的 NC 报警。

值 FALSE

通道中不存在导致加工停止的 NC 报警。

关联

- <Chan>.basic.in.ncAlarmActive
  存在通道专用 NC 报警
- <Nc>.basic.in.ncAlarmActive
  至少有一个 NC 报警处于活动状态

更多信息

- 诊断说明

12.3.23.34 orientableToolholderActive

NC → PLC	<Chan>.basic.in.orientableToolholderActive
可定向刀夹处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

可定向刀夹处于活动状态。

值 FALSE

可定向刀夹未激活。

12.3.23.35 overStoreActive

NC → PLC	<Chan>.basic.in.overStoreActive
溢出转存处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

溢出转存处于活动状态

值 FALSE

溢出转存未激活

12.3.23.36 productionTargetReached

NC → PLC	<Chan>.basic.in.productionTargetReached
已达到待加工工件数量 ($AC_ACTUAL_PARTS == $AC_REQUIRED_PARTS)	

数据类型与更新

BOOL，循环

值 TRUE

加工的工件数（实际工件数）等于待加工的工件数（设定工件数）：

$AC_ACTUAL_PARTS == $AC_REQUIRED_PARTS

值 FALSE

加工的工件数（实际工件数）不等于待加工的工件数（设定工件数）：

$AC_ACTUAL_PARTS <> $AC_REQUIRED_PARTS

关联

- MD27880 $MC_PART_COUNTER
  激活工件计数器

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”>“程序运行时间/工件计数器”>“工件计数器”章节

12.3.23.37 progStateAborted

NC → PLC	<Chan>.basic.in.progStateAborted
程序状态“取消”	

说明

此信号在以下情形下置位：

- 已在通道中选择当前 NC 程序，但未启动。
- 已在通道中执行当前 NC 程序，并通过 <Chan>.basic.out.reset = TRUE（通道复位）取消。

程序状态：

- <Chan>.basic.in.progStateAborted
  程序状态“取消”
- <Chan>.basic.in.progStateInterrupted
  程序状态“中断”
- <Chan>.basic.in.progStateStopped
  程序状态“停止”
- <Chan>.basic.in.progStateWaiting
  程序状态“等待”
- <Chan>.basic.in.progStateRunning
  程序状态“运行”

数据类型与更新

BOOL，循环

值 TRUE

程序状态 == "取消"

值 FALSE

程序状态 ≠ "取消"

关联

- <Chan>.basic.out.reset
  复位
- <Chan>.basic.in.progStateInterrupted
  程序状态“中断”
- <Chan>.basic.in.progStateStopped
  程序状态“停止”
- <Chan>.basic.in.progStateWaiting
  程序状态“等待”
- <Chan>.basic.in.progStateRunning
  程序状态“运行”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.38 progStateInterrupted

NC → PLC	<Chan>.basic.in.progStateInterrupted
程序状态“中断”	

说明

此信号显示，可通过 NC 启动 (<Chan>.basic.out.ncStart) = TRUE) 继续执行中断的 NC 程序。

例如，如果在“停止”状态 (<Chan>.basic.in.progStateStopped) == TRUE) 下从自动模式或 MDA 运行方式切换至 JOG 运行方式，便进入“中断”状态。在随后从 JOG 运行方式切换至自动模式或 MDA 运行方式后，可通过 NC 启动 (<Chan>.basic.out.ncStart) = TRUE) 从中断点起继续执行 NC 程序。

程序状态：

- <Chan>.basic.in.progStateAborted
  程序状态“取消”
- <Chan>.basic.in.progStateInterrupted
  程序状态“中断”
- <Chan>.basic.in.progStateStopped
  程序状态“停止”
- <Chan>.basic.in.progStateWaiting
  程序状态“等待”
- <Chan>.basic.in.progStateRunning
  程序状态“运行”

数据类型与更新

BOOL，循环

值 TRUE

程序状态 == “中断”

值 FALSE

程序状态 ≠ “中断”

关联

- <Chan>.basic.in.progStateAborted
  程序状态“取消”
- <Chan>.basic.in.progStateInterrupted
  程序状态“中断”
- <Chan>.basic.in.progStateStopped
  程序状态“停止”
- <Chan>.basic.in.progStateWaiting
  程序状态“等待”
- <Chan>.basic.in.progStateRunning
  程序状态“运行”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.39 progStateRunning

NC → PLC	<Chan>.basic.in.progStateRunning
程序状态“运行”	

说明

通过 <Chan>.basic.out.ncStart = TRUE 启动 NC 程序后，设置此信号。

如果 NC 程序的执行进程因 <Chan>.basic.out.disableReadIn = TRUE 而停止，会保持设置此信号。

在 NC 程序的执行因以下事件停止的情况下，此信号不复位：

- 进给禁止或主轴禁用生效
- <Chan>.basic.out.disableReadIn（读入锁定）
- 进给率修调：0 %
- 主轴监控和轴监控的响应
- 在 NC 程序中为“跟踪运行”下的轴、无“伺服使能”的轴或“驻留轴”设定位置设定值

程序状态：

- <Chan>.basic.in.progStateAborted
  程序状态“取消”
- <Chan>.basic.in.progStateInterrupted
  程序状态“中断”
- <Chan>.basic.in.progStateStopped
  程序状态“停止”
- <Chan>.basic.in.progStateWaiting
  程序状态“等待”
- <Chan>.basic.in.progStateRunning
  程序状态“运行”

数据类型与更新

BOOL，循环

值 TRUE

程序状态 == “运行”

值 FALSE

程序状态 ≠ "运行"

关联

- <Chan>.basic.out.disableReadIn = TRUE
  请求锁定读入
- <Chan>.basic.out.ncStart = TRUE
  脉冲沿切换 0 → 1：请求开始运行在通道中选择的程序
- <Chan>.basic.in.progStateAborted
  程序状态“取消”
- <Chan>.basic.in.progStateInterrupted
  程序状态“中断”
- <Chan>.basic.in.progStateStopped
  程序状态“停止”
- <Chan>.basic.in.progStateWaiting
  程序状态“等待”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.40 progStateStopped

NC → PLC	<Chan>.basic.in.progStateStopped
程序状态“停止”	

说明

导致 NC 程序停止的事件：

- <Chan>.basic.out.ncStop
  请求 NC 停止
- <Chan>.basic.out.ncStopAxesAndSpindles
  轴和主轴的 NC 停止请求
- <Chan>.basic.out.ncStopBlockEnd
  请求在当前程序段结尾执行 NC 停止

- 编写的指令 M00 或 M01
- 单程序段模式

程序状态：

- <Chan>.basic.in.progStateAborted
  程序状态“取消”
- <Chan>.basic.in.progStateInterrupted
  程序状态“中断”
- <Chan>.basic.in.progStateStopped
  程序状态“停止”
- <Chan>.basic.in.progStateWaiting
  程序状态“等待”
- <Chan>.basic.in.progStateRunning
  程序状态“运行”

数据类型与更新

BOOL，循环

值 TRUE

程序状态 == “停止”

值 FALSE

程序状态 ≠ “停止”

关联

- <Chan>.basic.out.ncStopBlockEnd
  请求在当前程序段结尾执行 NC 停止
- <Chan>.basic.out.ncStop
  请求 NC 停止
- <Chan>.basic.out.ncStopAxesAndSpindles
  轴和主轴的 NC 停止请求
- <Chan>.basic.in.progStateAborted
  程序状态“取消”
- <Chan>.basic.in.progStateInterrupted
  程序状态“中断”

- <Chan>.basic.in.progStateWaiting
  程序状态“等待”
- <Chan>.basic.in.progStateRunning
  程序状态“运行”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.41 progStateWaiting

NC → PLC	<Chan>.basic.in.progStateWaiting
程序状态“等待”	

说明

在 NC 程序中通过 WAIT_M 或 WAIT_E 等待与另一通道的 NC 程序的同步的情况下，进入程序状态“等待”。

程序状态：

- <Chan>.basic.in.progStateAborted
  程序状态“取消”
- <Chan>.basic.in.progStateInterrupted
  程序状态“中断”
- <Chan>.basic.in.progStateStopped
  程序状态“停止”
- <Chan>.basic.in.progStateWaiting
  程序状态“等待”
- <Chan>.basic.in.progStateRunning
  程序状态“运行”

数据类型与更新

BOOL，循环

值 TRUE

程序状态 == “等待”

值 FALSE

程序状态 ≠ “等待”

关联

- <Chan>.basic.in.progStateAborted
  程序状态“取消”
- <Chan>.basic.in.progStateInterrupted
  程序状态“中断”
- <Chan>.basic.in.progStateStopped
  程序状态“停止”
- <Chan>.basic.in.progStateRunning
  程序状态“运行”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.42 progTestSelected

NC → PLC	<Chan>.basic.in.progTestSelected
已选程序测试 (PRT)	

说明

如果程序测试（PRT）生效，则在执行程序段或 NC 程序时加工轴不运动。但在操作界面上通过变化的设定位置显示轴运动。

提示

程序测试（PRT）生效期间，所有轴（非主轴）的运行都在“轴禁止”下执行。

提示

在 SINUMERIK Operate 操作界面的“自动”>“程序控制”操作区域将 HMI/PLC 接口信号 <HmiChan>.basic.in.progTestSelected 置位，可以选择程序测试 (PRT)。

注意
刀库一致性 程序测试时轴被禁用，因此刀库配置保持不变。用户/机床制造商必须通过相适应的 PLC 用户程序确保 NC 内部的刀具管理与实际的刀库布局一致。

数据类型和更新

BOOL，循环

值 TRUE

程序测试 (PRT) 处于活动状态。

值 FALSE

程序测试 (PRT) 未激活。

关联

- <Chan>.basic.out.progTest
  请求激活程序测试 (PRT)
- <HmiChan>.basic.in.progTestSelected
  HMI：已请求程序测试 (PRT)

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.43 protAreasNotGuaranteed

NC → PLC	<Chan>.basic.in.protAreasNotGuaranteed
无法保证保护区域监控，因为监控在运动启动后激活（不是防撞保护 CA 的组成部分）	

数据类型与更新

BOOL

值 TRUE

无法保证保护区域监控

12.3.23.44 ptpTraversActive

NC → PLC	<Chan>.basic.in.ptpTraversActive
PTP（点对点）运行处于活动状态	

说明

只有转换处于活动状态时，变量才与 JOG 运行方式有关。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

PTP (Point-To-Point) 运动生效。

脉冲沿切换 1 → 0

CP (Continuous Path) 运动生效。

关联

- <Chan>.basic.out.ptpTravers
  激活点对点运行 (PTP)

更多信息

- 功能手册之工艺

12.3.23.45 ready

NC → PLC	<Chan>.basic.in.ready
通道运行就绪	

说明

通道就绪，以供执行 NC 程序，或者运行指定给通道的轴和主轴。

数据类型与更新

BOOL，循环

值 TRUE

通道就绪。

值 FALSE

通道未就绪。

关联

- MD11600 $MN_BAG_MASK
  定义运行方式组行为

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.46 reducedDynamicsActive

NC → PLC	<Chan>.basic.in.reducedDynamicsActive
低动态模式已激活	

说明

通道中的函数状态：

- 0：低动态模式未激活
- 1：低动态模式已激活

数据类型和更新

- BOOL

12.3.23.47 refActive

NC → PLC	<Chan>.basic.in.refActive
“回参考点运行”功能处于活动状态	

说明

针对请求激活回参考点的反馈信号：

1. <Chan>.basic.out.refPoint:0 → 1（请求）
2. <Chan>.basic.in.refActive:0 → 1（反馈）

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

回参考点生效。

脉冲沿切换 1 → 0

无作用

关联

- <Chan>.basic.out.refPoint
  脉冲沿切换 0 → 1：请求“回参考点运行”功能

更多信息

- 功能手册之进给轴和主轴；章节“回参考点”

12.3.23.48 reposDelayActive

NC → PLC	<Chan>.basic.in.reposDelayActive
“在 REPOS 接近运行方式下延迟”处于活动状态	

说明

该通道当前控制的所有轴或是无 REPOS 偏移，或是不运行其 REPOS 偏移。

数据类型与更新

BOOL，循环

值 TRUE

REPOS 延时生效。

值 FALSE

REPOS 延时未生效。

值	REPOS 接近运行方式
0	无 REPOS 运行方式
1	接近程序段起点 (RMBBL)
2	接近中断点 (RMIBL)
3	接近程序段终点 (RMEBL)
4	接近下一个轨迹点 (RMNBL)

关联

- <HmiChan>.basic.in.reposModeChanged
  HMI：REPOS 运行方式切换
- <Chan>.basic.out.reposModeSelection
  REPOS 接近运行方式：
  可能的值见上表
- <Chan>.basic.out.reposModeActivationRequest
  请求为当前主运行程序段激活“REPOS”运行方式
- <Axis>.basic.out.reposDelay
  已请求 REPOS 延迟
- <Chan>.basic.in.reposModeSelectedNumber
  已选 REPOS 接近运行方式：
  可能的值见上表

- <Chan>.basic.in.reposDelayActive
  “在 REPOS 接近运行方式下延迟”处于活动状态
- <Chan>.basic.in.reposModeChangeAck
  应答 REPOS 接近运行方式的变更请求
- <Chan>.basic.in.reposModeSelectedNumber
  已选 REPOS 接近运行方式：
  可能的值见上表
- <Axis>.basic.in.reposOffsetActive
  REPOS 偏移已应用在轴上
- <Axis>.basic.in.reposOffsetValid
  REPOS 偏移计算为有效
- <Axis>.basic.in.reposDelayAck
  已应答 REPOS 延迟
- MD11470 $MN_REPOS_MODE_MASK
  重新定位特性

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.49 reposModeChangeAck

NC → PLC	<Chan>.basic.in.reposModeChangeAck
应答 REPOS 接近运行方式的变更请求	

数据类型与更新

BOOL，循环

值 TRUE

存在对 REPOS 模式变更的应答。

在 NC 中应用请求的 REPOS 模式 <Chan>.basic.out.reposModeSelection 和延迟信号 <Axis>.basic.out.reposDelay 后，应答 NC 识别到的接口信号 <Chan>.basic.out.reposModeActivationRequest（请求为当前主运行程序段激活“REPOS”运行方式）。
信号状态基于当前的主处理程序段。

值 FALSE

不存在对 REPOS 模式变更的应答。

SERUPRO 异步子程序在 REPOS 和 <Chan>.basic.out.reposModeActivationRequest（请求为当前主运行程序段激活"REPOS"运行方式）前自动停止，不作用于 SERUPRO 接近。

值	**REPOS 接近运行方式**
0	无 REPOS 运行方式
1	接近程序段起点 (RMBBL)
2	接近中断点 (RMIBL)
3	接近程序段终点 (RMEBL)
4	接近下一个轨迹点 (RMNBL)

关联

- <HmiChan>.basic.in.reposModeChanged
  HMI：REPOS 运行方式切换
- <Chan>.basic.out.reposModeSelection
  REPOS 接近运行方式：
  可能的值见上表
- <Chan>.basic.out.reposModeActivationRequest
  请求为当前主运行程序段激活"REPOS"运行方式
- <Axis>.basic.out.reposDelay
  已请求 REPOS 延迟
- <Chan>.basic.in.reposModeSelectedNumber
  已选 REPOS 接近运行方式：
  可能的值见上表
- <Chan>.basic.in.reposDelayActive
  "在 REPOS 接近运行方式下延迟"处于活动状态
- <Axis>.basic.in.reposOffsetActive
  REPOS 偏移已应用在轴上
- <Axis>.basic.in.reposOffsetValid
  REPOS 偏移计算为有效

- <Axis>.basic.in.reposDelayAck
  已应答 REPOS 延迟
- MD11470 $MN_REPOS_MODE_MASK
  重新定位特性

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.50 reposModeSelectedNumber

NC → PLC	<Chan>.basic.in.reposModeSelectedNumber
已选 REPOS 接近运行方式：可能的值见下表	

数据类型与更新

USInt，循环

值	REPOS 接近运行方式
0	无 REPOS 运行方式
1	接近程序段起点 (RMBBL)
2	接近中断点 (RMIBL)
3	接近程序段终点 (RMEBL)
4	接近下一个轨迹点 (RMNBL)

关联

- <HmiChan>.basic.in.reposModeChanged
  HMI：REPOS 运行方式切换
- <Chan>.basic.out.reposModeSelection
  REPOS 接近运行方式：
  可能的值见上表
- <Chan>.basic.out.reposModeActivationRequest
  请求为当前主运行程序段激活“REPOS”运行方式
- <Axis>.basic.out.reposDelay
  已请求 REPOS 延迟
- <Chan>.basic.in.reposModeSelectedNumber
  已选 REPOS 接近运行方式：

- 可能的值见上表
- <Chan>.basic.in.reposDelayActive
  “在 REPOS 接近运行方式下延迟”处于活动状态
- <Axis>.basic.in.reposOffsetActive
  REPOS 偏移已应用在轴上
- <Axis>.basic.in.reposOffsetValid
  REPOS 偏移计算为有效
- <Axis>.basic.in.reposDelayAck
  已应答 REPOS 延迟
- <Chan>.basic.in.reposModeChangeAck
  应答 REPOS 接近运行方式的变更请求
- MD11470 $MN_REPOS_MODE_MASK
  重新定位特性

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为” > “程序段搜索类型 5 (SERUPRO)” > “再次返回轮廓 (REPOS)”章节

12.3.23.51 retraceSupportActive

NC → PLC	<Chan>.basic.in.retraceSupportActive
重新返回轮廓 (RESU)：机床返回中断点，从中断点继续运行	

说明

识别到 (<Chan>.basic.out.retraceSupportStart) == TRUE 后，设置此接口信号。

当最后的动作程序段结束时，此接口信号复位。

数据类型与更新

BOOL，循环

值 TRUE

重置生效。

值 FALSE

重置未生效。

关联

- <Chan>.basic.out.retraceSupportTraversBackwards
  重新返回轮廓 (RESU)：继续加工轮廓；请求“向后运行”
- <Chan>.basic.out.retraceSupportStart
  重新返回轮廓 (RESU)：脉冲沿切换 0 → 1：继续加工轮廓；请求“开始重新返回轮廓”
- <Chan>.basic.in.retraceSupportModeActive
  重新返回轮廓 (RESU)：支持中断后继续在轮廓上加工
- <Chan>.basic.in.retraceSupportActive
  重新返回轮廓 (RESU)：机床返回中断点，从中断点继续运行

更多信息

- 技术功能手册；“继续运行 (Retrace Support)”章节

12.3.23.52 retraceSupportModeActive

NC → PLC	<Chan>.basic.in.retraceSupportModeActive
重新返回轮廓 (RESU)：支持中断后继续在轮廓上加工	

说明

可在中断后继续运行。

当 (<Chan>.basic.out.retraceSupportTraversBackwards) == TRUE 时，设置此接口信号

当 (<Chan>.basic.out.retraceSupportStart) == TRUE 时，复位此接口信号

数据类型与更新

BOOL，循环

值 TRUE

返回模式生效。

值 FALSE

返回模式未生效

关联

- <Chan>.basic.out.retraceSupportTraversBackwards
  重新返回轮廓 (RESU)：继续加工轮廓；请求“向后运行”
- <Chan>.basic.out.retraceSupportStart
  重新返回轮廓 (RESU)：脉冲沿切换 0 → 1：继续加工轮廓；请求“开始重新返回轮廓”

- <Chan>.basic.in.retraceSupportModeActive
  重新返回轮廓 (RESU)：支持中断后继续在轮廓上加工
- <Chan>.basic.in.retraceSupportActive
  重新返回轮廓 (RESU)：机床返回中断点，从中断点继续运行

更多信息

- 技术功能手册；“继续运行 (Retrace Support)”章节

12.3.23.53 revolutionalFeedrateActive

NC → PLC	<Chan>.basic.in.revolutionalFeedrateActive
旋转进给 (G95) 处于活动状态	

说明

此接口信号显示，在 AUTO 运行方式下以旋转进给率运行轨迹轴或同步轴。

数据类型和更新

BOOL，循环

值 TRUE

旋转进给率（G95）生效。

值 FALSE

旋转进给率（G95）未生效。

更多信息

- 轴和主轴功能手册；“进给”>“轨迹进给 F”>“进给类型 G93、G94、G95”章节

12.3.23.54 singleBlockStopAtBlockEnd

NC → PLC	<Chan>.basic.in.singleBlockStopAtBlockEnd
在程序段结尾由于单段而停止	

数据类型与更新

BOOL

值 TRUE

在程序段结尾由于单段而停止

12.3.23.55 singleBlockStopIgnored

NC → PLC	<Chan>.basic.in.singleBlockStopIgnored
单段 (SBL) 时忽略程序段结尾停止	

说明

在以下情形下，忽略单程序段模式（SBL）下程序段末尾的停止：

<Chan>.basic.out.singleBlock == TRUE（单程序段）且当前程序段具有“单段无效”状态

在以下情形下，不忽略单程序段模式（SBL）下程序段末尾的停止：

<Chan>.basic.out.singleBlock == FALSE（单程序段）或

(<Chan>.basic.out.singleBlock) == TRUE（单程序段）且当前程序段具有“单段有效”状态

通过以下机床数据和指令设置单段模式 <Chan>.basic.out.singleBlock 下忽略程序段结尾停止：

- MD10702 $MN_IGNORE_SINGLEBLOCK_MASK（忽略单程序段停止）
- MD20117 $MC_IGNORE_SINGLEBLOCK_ASUP（单程序段模式下仍完整执行中断程序）
- MD20106 $MC_PROG_EVENT_IGN_SINGLEBLOCK（程序事件忽略单程序段）
- SBLOF（抑制单程序段）、SBLON（取消单程序段抑制）

忽略单程序段模式下的程序段末尾停止的程序段具有“单程序段无效”状态。

数据类型与更新

BOOL，循环

值 TRUE

单程序段模式（SBL）下程序段末尾的停止被忽略。

值 FALSE

单程序段模式（SBL）下程序段末尾的停止未被忽略。

关联

- <Chan>.basic.out.singleBlock
  请求激活“单段处理”(SBL) 功能
- <Chan>.basic.in.disableReadInIgnored
  忽略读入锁定

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.56 stateActive

NC → PLC	<Chan>.basic.in.stateActive
通道处于“活动”状态	

说明

此信号在以下情形下置位：

- 在 AUTO 或 MDI 运行方式下执行 NC 程序。
- 在 JOG 运行方式下至少有一根轴在运行。

通道状态：

- <Chan>.basic.in.stateActive
  通道处于“活动”状态
- <Chan>.basic.in.stateInterrupted
  通道状态“中断”
- <Chan>.basic.in.stateReset
  通道状态“复位”

数据类型与更新

BOOL，循环

值 TRUE

通道状态 == “生效”

值 FALSE

通道状态 ≠ “生效”

关联

- <Chan>.basic.in.stateInterrupted
  通道状态“中断”
- <Chan>.basic.in.stateReset
  通道状态“复位”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.57 stateInterrupted

NC → PLC	<Chan>.basic.in.stateInterrupted
通道状态“中断”	

说明

在 AUTO 或 MDI 运行方式下或在 JOG 运行方式下运行轴时，可能因以下事件之一而触发中断：

- <Chan>.basic.out.ncStop
  请求 NC 停止
- <Chan>.basic.out.ncStopAxesAndSpindles
  轴和主轴的 NC 停止请求
- <Chan>.basic.out.ncStopBlockEnd
  请求在当前程序段结尾执行 NC 停止
- 编程的停止 M00 或 M01
- 单程序段模式

通道状态：

- <Chan>.basic.in.stateActive
  通道处于“活动”状态
- <Chan>.basic.in.stateInterrupted
  通道状态“中断”
- <Chan>.basic.in.stateReset
  通道状态“复位”

数据类型与更新

BOOL，循环

值 TRUE

通道状态 == “中断”

值 FALSE

通道状态 ≠ “中断”

关联

- <Chan>.basic.out.ncStop
  请求 NC 停止
- <Chan>.basic.out.ncStopAxesAndSpindles
  轴和主轴的 NC 停止请求
- <Chan>.basic.out.ncStopBlockEnd
  请求在当前程序段结尾执行 NC 停止
- <Chan>.basic.in.stateActive
  通道处于“活动”状态
- <Chan>.basic.in.stateReset
  通道状态“复位”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.58 stateReset

NC → PLC	<Chan>.basic.in.stateReset
通道状态“复位”	

说明

此信号在以下情形下置位：

- 程序结束复位（M02 / M30）
- 通道复位
- 热启动（上电）

通道状态：

- <Chan>.basic.in.stateActive
  通道处于“活动”状态
- <Chan>.basic.in.stateInterrupted
  通道状态“中断”
- <Chan>.basic.in.stateReset
  通道状态“复位”

数据类型与更新

BOOL，循环

值 TRUE

通道状态 == “复位”

值 FALSE

通道状态 ≠ “复位”

关联

- <Chan>.basic.in.stateActive
  通道处于“活动”状态
- <Chan>.basic.in.stateInterrupted
  通道状态“中断”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.23.59 stopConditionActive

NC → PLC	<Chan>.basic.in.stopConditionActive
程序运行时的停止条件处于活动状态	

数据类型与更新

BOOL

值 TRUE

运行程序期间，停止条件处于活动状态。

值 FALSE

停止条件未激活。

12.3.23.60 toffonActive

NC → PLC	<Chan>.basic.in.toffonActive
在线刀具长度补偿 (TOFFON)：功能处于活动状态 (MD19320 TECHNO_FUNCTION_MASK Bit 29)	

说明

借助在线刀具长度补偿，能够实时修改有效的刀具长度，使得系统即便在刀具定向变化时也将此长度变化考虑在内。

应用范围：

- 定向转换（TRAORI）
- 可定向刀夹 (TCARR)

数据类型与更新

BOOL，循环

值 TRUE

在线刀具长度补偿（TOFF）生效。

值 FALSE

在线刀具长度补偿（TOFF）未生效

关联

- <Chan>.basic.in.toffonMotionActive
  在线刀具长度补偿 (TOFFON)：补偿运动处于活动状态 (MD21194 TOFF_VELO)
- MD19320 TECHNO_FUNCTION_MASK Bit29

更多信息

- 坐标转换功能手册；“多轴转换”章节

12.3.23.61 toffonMotionActive

NC → PLC	<Chan>.basic.in.toffonMotionActive
在线刀具长度补偿 (TOFFON)：补偿运动处于活动状态 (MD21194 TOFF_VELO)	

说明

如果在线刀具长度补偿处于活动状态 (<Chan>.basic.in.toffonActive == TRUE) 时激活补偿运动，会将接口信号设为 TRUE：

<Chan>.basic.in.toffonMotionActive == TRUE

数据类型与更新

BOOL，循环

值 TRUE

补偿运动生效。

值 FALSE

补偿运动未生效。

关联

- <Chan>.basic.in.toffonActive
  在线刀具长度补偿 (TOFFON)：功能处于活动状态 (MD19320 TECHNO_FUNCTION_MASK Bit 29)
- MD21194 TOFF_VELO

更多信息

- 坐标转换功能手册；“多轴转换”章节

12.3.23.62 transformationActive

NC → PLC	<Chan>.basic.in.transformationActive
坐标转换处于活动状态	

说明

在通道中，在生效的 NC 程序中编写了指令 TRANSMIT、TRACYL、TRAANG 或 TRAORI 中的一个。相应的程序段已执行，且相应的转换生效。

数据类型与更新

BOOL，循环

值 TRUE

转换生效。

值 FALSE

无转换生效。

关联

- <Chan>.basic.in.transformationActiveNumber
  当前活动转换的编号

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节
- 功能手册之工艺

12.3.23.63 transformationActiveNumber

NC → PLC	<Chan>.basic.in.transformationActiveNumber
当前活动转换的编号	

说明

转换处于活动状态时，接口信号显示活动转换的转换索引。

示例 - 编程代码：

N24100 $MC_TRAFO_TYP_1=24

N24200 $MC_TRAFO_TYP_2=513

N24300 $MC_TRAFO_TYP_3=24

接口信号中的显示：

TRAORI → E_TransformNo=1

TRAORI(2) → E_TransformNo=3

数据类型与更新

BYTE，循环

值

通道中活动转换的索引编号

关联

- <Chan>.basic.in.transformationActive
  坐标转换处于活动状态

12.3.24 <Chan>.basic.out

12.3.24.1 abortProgLevel

PLC → NC	<Chan>.basic.out.abortProgLevel
脉冲沿切换 0 → 1：请求取消运行当前程序级（子程序级、ASUB 级、数据存储器例行程序）的程序。	

说明

在程序级终止后，自调用点起，继续执行下一程序级的调用程序。

主程序级无法通过程序级终止，而是借助通道复位终止。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

请求当前程序级（子程序级、ASUB 级、例行程序）的程序运行终止。

脉冲沿切换 1 → 0

将最后的请求复位。

关联

- <Chan>.basic.out.reset
  已请求通道复位

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.24.2 blockSearchAckPlcActionDone

PLC → NC	<Chan>.basic.out.blockSearchAckPlcActionDone
程序段搜索：完成 PLC 动作反馈	

说明

在程序段搜索末尾运行动作程序段：
<Chan>.basic.in.blockSearchActionActive（程序段搜索：动作程序段处于活动状态/正在运行）== TRUE 且
<Chan>.basic.in.blockSearchLastActionActive（程序段搜索：最后一个动作程序段处于活动状态/正在运行）== TRUE

系统通过报警"10208 通道 <*通道编号*> 触发 NC 启动以继续执行程序"提示：从目标程序段起需要重新触发 NC 启动。

若在通过 PLC 用户程序进行 NC 启动前还需执行动作（例如换刀），可通过设置搜索模式延迟报警输出，直至现有信号重新置位：

MD11450 $MN_SEARCH_RUN_MODE = 1

数据类型与更新

BOOL，循环

值 TRUE

PLC 动作已结束。

值 FALSE

PLC 动作尚未结束。

关联

- <Chan>.basic.out.ncStart
  脉冲沿切换 0 → 1：请求开始运行在通道中选择的程序
- <Chan>.basic.in.blockSearchActionActive
  程序段搜索：动作程序段处于活动状态/正在运行
- <Chan>.basic.in.blockSearchLastActionActive
  程序段搜索：最后一个动作程序段处于活动状态/正在运行
- <Chan>.basic.in.blockSearchActive
  程序段搜索：数据采集处于活动状态

更多信息

- 基础功能手册；"运行方式组、通道、程序运行、复位行为"章节；
  - 章节"程序段搜索类型 1、2 和 4"
  - 章节"程序段搜索类型 5（SERUPRO）"

12.3.24.3 circularJogTravers

PLC → NC	<Chan>.basic.out.circularJogTravers
请求“点动圆弧运行”（SD42690 至 42694 $SC_JOG_CIRCLE_...）	

说明

一旦此功能激活（参见 <Chan>.basic.in.circularJogTraversSelected），机床操作人员可以操作移动键或手轮使活动平面的两根几何轴同时沿圆弧运行。

应用

此功能适用于只能手动运行的机床。

说明

对于“圆弧手动运行”而言，以下前提条件适用：

- “圆弧手动运行”功能只能在 JOG 模式下激活。
  机床功能 JOG-REPOS 和 JOG-REF 生效时无法激活此功能。
- 参与运行的轴必须回参考点。
- 生效平面不可斜置。

数据类型与更新

BOOL，循环

值 TRUE

已请求“圆弧手动运行”功能。

值 FALSE

未请求“圆弧手动运行”功能。

关联

- <Chan>.basic.in.circularJogTraversSelected
  圆弧手动运行处于活动状态
- SD42690 to 42694 $SC_JOG_CIRCLE_...

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.24.4 clcOvr

PLC → NC	<Chan>.basic.out.clcOvr
间距控制 (CLC)：请求使用通道专用进给倍率	

说明

在信号状态 1 下，通道专用进给倍率对间距控制运动的最大速度（MD62516）起作用：

- 倍率设置 < 100%
  MD62516 中设置的针对距离调节运动的速度限制相应减小。
- 倍率设置 > 100%
  MD62516 中的限制值生效。

在信号状态 0 下，间距控制运动的最大速度与倍率设置无关。

数据类型与更新

BOOL，循环

值 TRUE

通道专用进给倍率 <Chan>.basic.out.pathFeedrateOvrFactor 对间距控制 (CLC) 有效。

值 FALSE

通道专用进给倍率 <Chan>.basic.out.pathFeedrateOvrFactor 对间距控制 (CLC) 无效。

关联

- <Chan>.basic.out.pathFeedrateOvrFactor
  可以通过二进制或格雷码编码形式规定轨迹进给倍率系数 (MD12020 $MN_OVR_FEED_IS_GRAY_CODE)
- <Chan>.basic.out.pathFeedrateOvr
  轨迹进给倍率处于活动状态
- <Chan>.basic.in.clcActive
  间距控制 (CLC) 处于活动状态（编译循环）
- MD62516 $MC_CLC_SENSOR_VELO_LIMIT
  间距控制运动的速度

更多信息

- 监控和补偿功能手册；“间距控制”章节

12.3.24.5 configuredStopFunc

PLC → NC	<Chan>.basic.out.configuredStopFunc
激活已组态的停止	

说明

激活已组态的停止 (CST)。

应在 NC 中激活“已组态的停止”功能。

激活在调试期间通过设置数据确定停止状态。

数据类型和更新

BOOL，循环

值 TRUE

激活已组态的停止

值 FALSE

已组态的停止未激活

关联

- 设置选项“已组态的停止”（需要许可证）
- 通过 MD52210 Bit18 激活 SINUMERIK Operate 中的显示
- <Chan>.basic.in.configuredStopFuncActive
  已组态的停止已激活

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.24.6 contourHandwheelInvertDir

PLC → NC	<Chan>.basic.out.contourHandwheelInvertDir
请求反转所选轮廓手轮的旋转方向	

说明

只允许在轴处于静止状态时修改该接口信号。

数据类型与更新

BOOL，循环

值 TRUE

已请求取反轮廓手轮的旋转方向。

值 FALSE

未请求取反轮廓手轮的旋转方向。

关联

- <Chan>.basic.in.contourHandwheelDirInverted
  轮廓手轮旋转方向反转处于活动状态

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.24.7 contourHandwheelSelection

PLC → NC	<Chan>.basic.out.contourHandwheelSelection
手轮 1..6 已分配或 0 未分配手轮（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）	

说明

所分配轮廓手轮的编号。

该接口使用了二进制编码。通过机床数据 MD11324=1 进行定义。

数据类型和更新

USInt，循环

值

0：无手轮分配

1...6：所分配手轮的编号

关联

- MD11324 $MN_HANDWH_VDI_REPRESENTATION
  VDI 接口中的手轮编号显示
- <Chan>.basic.in.contourHandwheelActiveNumber
  轮廓手轮处于活动状态，接口可解释为比特或二进制编码

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.24.8 contourHandwheelSim

PLC → NC	<Chan>.basic.out.contourHandwheelSim
激活轮廓手轮模拟	

说明

在仿真中，进给率不再由轮廓手轮给出，而是来自程序。在仿真关闭时，当前的运行通过制动斜坡制动。

说明

仿真只在 AUTO 运行方式下有效，只能在轮廓手轮激活后激活。

数据类型与更新

BOOL，循环

值 TRUE

接通轮廓手轮仿真。

值 FALSE

关闭轮廓手轮仿真。

关联

- <Chan>.basic.out.contourHandwheelSimInvertDir
  模拟轮廓手轮时，它沿轮廓朝向编程方向的反方向移动

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.24.9 contourHandwheelSimInvertDir

PLC → NC	<Chan>.basic.out.contourHandwheelSimInvertDir
模拟轮廓手轮时，它沿轮廓朝向编程方向的反方向移动	

数据类型与更新

BOOL

值 TRUE

模拟轮廓手轮时，它沿轮廓朝向编程方向的反方向移动

12.3.24.10 deactivateClc

PLC → NC	<Chan>.basic.out.deactivateClc
间距控制 (CLC)：请求禁用（类似程序指令 CLC_GAIN=0.0）	

数据类型与更新

BOOL，循环

值 TRUE

已请求关闭间距控制 (CLC)（类似于程序指令 CLC_GAIN=0.0）。

值 FALSE

未请求关闭间距控制 (CLC)。

关联

- CLC_GAIN=0.0 ;
  关闭控制环增益
- <Chan>.basic.in.clcActive
  间距控制 (CLC) 处于活动状态（编译循环）

更多信息

- 监控和补偿功能手册；“间距控制”章节

12.3.24.11 deleteDistanceToGo

PLC → NC	<Chan>.basic.out.deleteDistanceToGo
脉冲沿切换 0 → 1：已请求删除剩余行程（通道专用剩余行程）	

说明

信号失效条件：

- 定位轴
- 暂停时间

应用示例：

运行因外部信号（例如测头）而结束

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

请求删除剩余行程（通道专用）。

几何轴：

在信号的上升沿时，程序段中当前运行的几何轴通过制动斜坡停止，随后删除剩余行程。可能存在的跟随误差会被削减。之后切换至下一个运行程序段。

轨迹轴：

“删除剩余行程”只在 AUTO 运行方式下对轨迹轴生效。

说明

通过“删除剩余行程”使轴停止后，系统会以新的位置为后续程序段进行程序段预处理。这样一来，“删除剩余行程”后几何轴运行的轮廓会有别于 NC 程序中的定义。

通过在“删除剩余行程”后的程序段中运行到绝对位置 (G90)，可确保至少精确地逼近程序段末尾位置。增量运行 (G91) 时，不会逼近在 NC 程序中指定的位置。

脉冲沿切换 1 → 0

无作用

关联

- <Axis>.basic.out.resetMovement
  脉冲沿切换 0 → 1：已请求“轴 - 删除剩余行程”；已请求“主轴 - 复位”

12.3.24.12 deleteSubProgRepetition

PLC → NC	<Chan>.basic.out.deleteSubProgRepetition
脉冲沿切换 0 → 1：删除剩余子程序重复次数	

数据类型与更新

BOOL

脉冲沿切换 0 → 1

删除子程序运行的剩余重复

脉冲沿切换 1 → 0

无作用

12.3.24.13 disableFeed

PLC → NC	<Chan>.basic.out.disableFeed
激活通道中的进给锁定	

说明

- 此接口信号对通道中运行的所有几何轴、同步轴及定位轴生效。
- 若在轴运行期间将此接口信号置位，则轴通过其制动特性曲线制动至静止状态。就几何轴而言，以遵循轮廓的方式进行制动。
- 在此接口信号复位后，系统会继续执行停止的运行。
- 位置闭环控制保持生效，跟随误差会减小。
- 若此接口置位，并且请求轴的运行，则轴不运行。但运行请求仍然保留。在此接口信号复位后，系统会继续执行运行请求，即运行轴。
- 该接口信号在所有运行方式下生效。

螺纹加工

此接口信号在各种螺纹加工期间不生效：

螺纹加工	**生效方式**
G33、G34、G35	不生效
G331、G332	生效
G63	生效

数据类型与更新

BOOL，循环

值 TRUE

进给禁止在通道中生效。

值 FALSE

进给禁止在通道中未生效。

更多信息

- 轴和主轴功能手册；“进给”>“进给控制”>“进给锁定和进给/主轴停止”章节

12.3.24.14 disableNcStart

PLC → NC	<Chan>.basic.out.disableNcStart
请求锁定 NC 启动	

说明

如果设置了 NC 启动锁定，会忽略在通道中选择的 NC 程序启动 <Chan>.basic.out.ncStart。

说明

在运行方式组的另一通道中通过零件程序指令 START 启动零件程序的情况下，NC 启动禁止不生效。

应用：

例如因缺少润滑剂而抑制程序的重新运行。

数据类型与更新

BOOL，循环

值 TRUE

已请求激活“NC 启动禁止”功能。

值 FALSE

未请求激活“NC 启动禁止”功能。

关联

- <Chan>.basic.out.ncStart
  脉冲沿切换 0 → 1：请求开始运行在通道中选择的程序

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.24.15 disableReadIn

PLC → NC	<Chan>.basic.out.disableReadIn
请求锁定读入	

说明

若读取禁止置位，则系统禁止下一个程序段的数据传送到插补器。

说明

仅在 AUTO 和 MDI 运行方式下生效。

应用：

如果下个 NC 程序段的执行要求结束辅助功能的执行（例如在换刀时），则必须通过读取禁止功能阻止自动切换程序段。

数据类型与更新

BOOL，循环

值 TRUE

已请求激活“读取禁止”功能。

值 FALSE

未请求激活“读取禁止”功能。

关联

- <Chan>.basic.in.progStateRunning
  程序状态：运行

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.24.16 dryRunFeedrate

PLC → NC	<Chan>.basic.out.dryRunFeedrate
请求激活试运行进给率 (DRY)	

说明

若通道处于“复位”状态，则在 G 功能 G01、G02、G03 生效的情况下，在下一次 NC 启动时以当前有效的进给率运行：

- 脉冲沿切换 0 → 1：试运行进给率
- 脉冲沿切换 1 → 0：编程的进给率
  提示
  若在一 G33 程序段中请求编程的进给率，则该编程的进给率在程序段末尾才会生效。在 G33 程序段内不执行 NC 停止。

提示

在 SINUMERIK Operate 操作界面的“自动”>“程序控制”操作区域将 HMI/PLC 接口信号 <HmiChan>.basic.in.dryRunSelected 置位，可以选择试运行进给 (DRY)。

数据类型和更新

BOOL，循环

脉冲沿切换 0 → 1

已请求激活空运行进给（DRY）。

脉冲沿切换 1 → 0

未请求激活空运行进给（DRY）。

关联

- <HmiChan>.basic.in.dryRunSelected
  HMI：已请求试运行进给率 (DRY)
- SD42100 $SC_DRY_RUN_FEED
  试运行进给

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”>“程序控制”>“试运行进给”章节
- 轴和主轴功能手册；“进给”>“进给控制”>“试运行进给”章节

12.3.24.17 enableGoToStart

PLC → NC	<Chan>.basic.out.enableGoToStart
请求 GOTOS 使能（跳至当前程序开头）	

数据类型与更新

BOOL，循环

值 TRUE

已请求在通道中激活 GOTOS（跳回到程序开始）的使能。

值 FALSE

未请求在通道中激活 GOTOS（跳回到程序开始）的使能。

NC 程序从 GOTOS 后的下一个指令继续执行。

关联

- MD27850 $MC_PROG_NET_TIMER_MODE
  程序运行时间净值计数器控制
- MD27860 $MC_PROCESSTIMER_MODE
  激活程序运行时间测量
- MD27880 $MC_PART_COUNTER
  激活工件计数器

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”>“程序运行”>“跳转到程序开头”章节

12.3.24.18 enableProtAreas

PLC → NC	<Chan>.basic.out.enableProtAreas
保护区域使能（不是防撞保护 CA 的构成部分）	

数据类型与更新

BOOL

值 TRUE

使能保护区域

值 FALSE

不使能保护区域

12.3.24.19 geoAxesTraversBcsWcsSelection

PLC → NC	<Chan>.basic.out.geoAxesTraversBcsWcsSelection
针对几何轴手动运行和手轮超驰 (DRF) 的坐标系选择： 0 - 不选择 1 - 基准坐标系 2 - 工件坐标系 3 - 刀具坐标系	

说明

选择：用于直角手动运行和 AUTO 模式下沿刀具方向的手轮叠加（DRF）的坐标系

选择用于以下的坐标系：

- 几何轴的沿刀具方向的直角手动运行（TCS）
- AUTO 运行方式下几何轴沿沿刀具方向的手轮叠加（DRF）

数据类型与更新

BYTE，循环

值

0：功能未被选择

1：在基本坐标系（BKS）中手动运行几何轴

2：在工件坐标系（WCS）中手动运行几何轴

3：在刀具坐标系（TCS）中进行几何轴的手动运行或手轮叠加（DRF）

更多信息

- 基础功能功能手册；章节：
  - “轴、坐标系、框架”>“坐标系”>“叠加补偿”>“DRF 偏移”
  - PLC 功能手册
- 坐标转换功能手册；章节：
  - “运动坐标转换”

12.3.24.20 handwheelOffsetDrf

PLC → NC	<Chan>.basic.out.contourHandwheelInvertDir
请求反转所选轮廓手轮的旋转方向	

说明

通过手轮偏移(DRF)可以在运行方式 AUTOMATIC 和 MDA 下使用手轮执行轴专用偏移。

提示

在 SINUMERIK Operate 操作界面的“自动”>“程序控制”操作区域将 HMI/PLC 接口信号 <HmiChan>.basic.in.handwheelOffsetDrfSelected 置位，可以选择手轮偏移 (DRF)。

数据类型和更新

BOOL，循环

值 TRUE

已请求激活手轮偏移 (DRF)。

值 FALSE

未请求激活手轮偏移 (DRF)。

关联

- <HmiChan>.basic.in.handwheelOffsetDrfSelected
  HMI：已请求手轮偏移 (DRF)

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.24.21 lockAllSynActs

PLC → NC	<Chan>.basic.out.lockAllSynActs
锁定支持的所有同步动作 (MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)	

数据类型与更新

BOOL

值 TRUE

锁定所有未受保护的同步动作

值 FALSE

无作用

12.3.24.22 m01

PLC → NC	<Chan>.basic.out.m01
请求激活程序控制“有条件停止”(M01)	

说明

通过 SINUMERIK Operate 操作界面选择程序控制“有条件停止” M01：在“自动”>“程序控制”操作区域，将 HMI/PLC 接口信号 <HmiChan>.basic.in.m01Selected 置位。

数据类型和更新

BOOL，循环

值 TRUE

已请求激活程序控制“有条件停止” M01。

值 FALSE

未请求激活程序控制“有条件停止”M01。

关联

- <HmiChan>.basic.in.m01Selected
  HMI：已请求程序控制 M01“有条件停止”
- <Chan>.basic.in.m00m01Active
  M00 / M01：“编程的停止/有条件停止”处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.24.23 m01AssociatedAuxFunc

PLC → NC	<Chan>.basic.out.m01AssociatedAuxFunc
请求激活 M0/M1“关联辅助功能”	

说明

在 SINUMERIK Operate 操作界面的“自动”>“程序控制”操作区域将 HMI/PLC 接口信号 <HmiChan>.basic.in.m01AssociatedAuxFuncSelected 置位，可以选择“关联辅助功能”(M-1)。

数据类型和更新

BOOL，任务控制

值 TRUE

已请求激活“关联辅助功能”。

值 FALSE

未请求激活“关联辅助功能”。

关联

- <HmiChan>.basic.in.m01AssociatedAuxFuncSelected
  HMI：已请求程序控制 M01“相关辅助功能”
- <Chan>.basic.in.m00m01AssociatedAuxFuncActive
  M00 / M01：相应的辅助功能“编程的停止/有条件停止”处于活动状态
- MD22254 $MC_AUXFU_ASSOC_M0_VALUE
  用于程序停止的附加 M 功能
- MD22256 $MC_AUXFU_ASSOC_M1_VALUE
  用于有条件停止的附加 M 功能

更多信息

- 基础功能手册；“辅助功能输出至 PLC” > “关联辅助功能”章节

12.3.24.24 ncStart

PLC → NC	<Chan>.basic.out.ncStart
脉冲沿切换 0 → 1：请求开始运行在通道中选择的程序	

说明

- 运行方式 AUTOMATIC：所选择的 NC 程序会启动或继续执行，或者输出程序中断期间保存的辅助功能。
  在“程序中断”状态下从 PLC 向 NC 传输数据时，NC 启动时这些数据会立即被计算。
- MDI 运行方式：
  执行 MDI 程序段存储器中的程序段。

数据类型与更新

BOOL，循环

信号状态 0 → 1

请求启动通道中选择的程序。

信号状态 1 → 0

将最后的请求复位。

关联

- <Chan>.basic.out.disableNcStart
  请求锁定 NC 启动
- <Chan>.basic.out.ncStop
  请求 NC 停止
- <Chan>.basic.out.reset
  已请求通道复位

更多信息

- 基础功能功能手册
  – “运行方式组、通道、程序运行、复位行为” > “程序运行” > “通道状态”章节
  – “运行方式组、通道、程序运行、复位行为” > “程序运行” > “程序状态”章节

12.3.24.25 ncStop

PLC → NC	<Chan>.basic.out.ncStop
请求 NC 停止	

说明

- 自动模式或 MDA 运行方式
  通道中激活的零件程序将中断执行。
  运行的轴（非主轴）通过参数设置的加速度制动至静止状态。
  – 通道状态：中断
  – 程序状态：中止
- JOG 运行方式
  未完全驶完的增量行程 (INC...)，会在下一次 NC 启动 <Chan>.basic.out.ncStart 时，以 JOG 运行方式继续运行。

说明

如果 NC 停止后将刀具补偿等数据传输至 NC，系统会在下一次 NC 启动时将这些数据考虑在内。

数据类型与更新

BOOL，循环

值 TRUE

已请求 NC 停止。

值 FALSE

未请求 NC 停止。

关联

- <Chan>.basic.out.ncStart
  脉冲沿切换 0 → 1：请求开始运行在通道中选择的程序
- <Chan>.basic.out.ncStopAxesAndSpindles
  轴和主轴的 NC 停止请求
- <Chan>.basic.in.progStateStopped
  程序状态“停止”
- <Chan>.basic.in.stateInterrupted
  通道状态“中断”

更多信息

- 基础功能功能手册
  – “运行方式组、通道、程序运行、复位行为” > “程序运行” > “通道状态”章节
  – “运行方式组、通道、程序运行、复位行为” > “程序运行” > “程序状态”章节

12.3.24.26 ncStopAxesAndSpindles

PLC → NC	<Chan>.basic.out.ncStopAxesAndSpindles
轴和主轴的 NC 停止请求	

说明

- 自动模式或 MDA 运行方式
  通道中激活的零件程序将中断执行。
  运行的轴和主轴通过参数设置的加速度制动至静止状态。
  – 通道状态：中断
  – 程序状态：中止
- JOG 运行方式
  未完全驶完的增量行程 (INC...)，会在下一次 NC 启动 <Chan>.basic.out.ncStart 时，以 JOG 运行方式继续运行。

说明

如果 NC 停止后将刀具补偿等数据传输至 NC，系统会在下一次 NC 启动时将这些数据考虑在内。

数据类型与更新

BOOL，循环

值 TRUE

已请求 NC 停止。

值 FALSE

未请求 NC 停止。

关联

- <Chan>.basic.out.ncStart
  脉冲沿切换 0 → 1：请求开始运行在通道中选择的程序
- <Chan>.basic.out.ncStop
  请求 NC 停止
- <Chan>.basic.in.progStateStopped
  程序状态“停止”
- <Chan>.basic.in.stateInterrupted
  通道状态“中断”
- <Chan>.basic.in.progStateStopped
  程序状态“停止”

更多信息

- 基础功能功能手册
  – “运行方式组、通道、程序运行、复位行为”>“程序运行”>“通道状态”章节
  – “运行方式组、通道、程序运行、复位行为”>“程序运行”>“程序状态”章节

12.3.24.27 ncStopBlockEnd

PLC → NC	<Chan>.basic.out.ncStopBlockEnd
请求在当前程序段结尾执行 NC 停止	

说明

执行当前程序段，直至程序段末尾。随后停止程序运行：

- 通道状态：“中断”
- 程序状态：“停止”

数据类型与更新

BOOL，循环

值 TRUE

已请求当前程序段的末尾处的 NC 停止。

值 FALSE

未请求当前程序段的末尾处的 NC 停止。

关联

- <Chan>.basic.out.ncStart
  脉冲沿切换 0 → 1：请求开始运行在通道中选择的程序
- <Chan>.basic.out.ncStop
  请求 NC 停止
- <Chan>.basic.out.ncStopAxesAndSpindles
  轴和主轴的 NC 停止请求
- <Chan>.basic.in.progStateStopped
  程序状态“停止”
- <Chan>.basic.in.stateInterrupted
  通道状态“中断”

更多信息

- 基础功能功能手册
  – “运行方式组、通道、程序运行、复位行为” > “程序运行” > “通道状态”章节
  – “运行方式组、通道、程序运行、复位行为” > “程序运行” > “程序状态”章节

12.3.24.28 pathFeedrateLimit

PLC → NC	<Chan>.basic.out.pathFeedrateLimit
激活刀具轨迹进给限制 (MD12100 $MN_OVR_FACTOR_LIMIT_BIN)	

数据类型与更新

BOOL

值 TRUE

激活刀具行程进给限制

值 FALSE

未激活刀具行程进给限制

关联

- MD12100

12.3.24.29 pathFeedrateOvr

PLC → NC	<Chan>.basic.out.pathFeedrateOvr
轨迹进给倍率处于活动状态	

说明

若轨迹进给倍率未生效，则控制系统内部以独立于当前开关位置的方式将值 1.0 ≙ 100 % 用作补偿值。特例是第一开关位置。就第一开关位置而言，根据所选择的编码生效：

- 二进制编码：补偿值 = 0
- 格雷编码：补偿值 = MD12030 $MN_OVR_FACTOR_FEEDRATE[0]

数据类型与更新

BOOL，循环

值 TRUE

轨迹进给倍率 <Chan>.basic.out.pathFeedrateOvrFactor 有效。

值 FALSE

轨迹进给倍率不生效。

关联

- <Chan>.basic.out.pathFeedrateOvrFactor
  可以通过二进制或格雷码编码形式规定轨迹进给倍率系数 (MD12020 $MN_OVR_FEED_IS_GRAY_CODE)
- MD12020 $MN_OVR_FEED_IS_GRAY_CODE
  轨迹进给倍率开关，格雷码编码
- MD12030 $MN_OVR_FACTOR_FEEDRATE
  轨迹进给倍率开关的权重系数

更多信息

- 轴和主轴功能手册；“进给”>“进给控制”>“通过机床控制面板设置进给倍率”章节

12.3.24.30 pathFeedrateOvrFactor

PLC → NC	<Chan>.basic.out.pathFeedrateOvrFactor
可以通过二进制或格雷码编码形式规定轨迹进给倍率系数 (MD12020 $MN_OVR_FEED_IS_GRAY_CODE)	

说明

倍率系数可以二进制码或格雷码设定。使用的格式必须通过以下机床数据告知控制系统：

MD12020 $MN_OVR_FEED_IS_GRAY_CODE = <编码方式>

数据类型与更新

BYTE，循环

二进制编码时的值

采用二进制码时，接口中的值对应倍率系数。

二进制编码	十进制	倍率系数
0000 0000	0	0.00
0000 0001	1	0.01
0000 0010	2	0.02
0000 0011	3	0.03
000 0100	4	0.04
...	...	...
0110 0100	100	1.00

二进制编码	十进制	倍率系数
...	...	...
1100 1000	200	2.00

格雷码编码时的值

以下机床数据可以指定接口格雷码形式的数值对应的倍率系数：

MD12030 $MN_OVR_FACTOR_FEEDRATE[<开关位置> - 1] = <倍率系数>

开关位置	格雷码	倍率系数 [1)]
1	00001	0.00
2	00011	0.01
3	00010	0.02
4	00110	0.04
5	00111	0.06
6	00101	0.08
7	00100	0.10
8	01100	0.20
9	01101	0.30
10	01111	0.40
11	01110	0.50
12	01010	0.60
13	01011	0.70
14	01001	0.75
15	01000	0.80
16	11000	0.85
17	11001	0.90
18	11011	0.95
19	11010	1.00
20	11110	1.05
21	11111	1.10
22	11101	1.15
23	11100	1.20
24	10100	1.20

开关位置	格雷码	倍率系数 1)
25	10101	1.20
26	10111	1.20
27	10110	1.20
28	10010	1.20
29	10011	1.20
30	10001	1.20
31	10000	1.20

1) 缺省值

限制

- 控制系统内部将生效的倍率系数限制在 2.00 或 200% 内。
- 可通过以下机床数据将最大倍率系数限制为一个小于 200% 的值：
  MD12100 $MN_OVR_FACTOR_LIMIT_BIN

关联

- <Chan>.basic.out.pathFeedrateOvr
  轨迹进给倍率处于活动状态

更多信息

- 轴和主轴功能手册；“进给”>“进给控制”>“通过机床控制面板设置进给倍率”章节

12.3.24.31 pathFixedFeedrateSelection

PLC → NC	<Chan>.basic.out.pathFixedFeedrateSelection
激活用于轨迹轴或几何轴的固定进给率 1 - 4（0 代表未选择）通过接口信号，在自动模式下激活固定进给率，而非编程设置的进给率和组态的 JOG 速度。	

说明

借助这些接口信号在 AUTO 运行方式下激活通过机床数据设置的固定进给率，用以替代编写的进给率或配置的 JOG 速度。

通过以下机床数据对固定进给率进行参数设置：

- 直线轴：MD12202 $MN_PERMANENT_FEED
- 回转轴：MD12204 $MN_PERMANENT_ROT_AX_FEED

说明

- 在以下情形下，无法以固定进给率运行：
  - 主轴
  - 定位轴
  - 攻丝
- 固定进给率始终被视作线性进给。即便在旋转进给率生效的情况下，控制系统内部也会转换为线性进给。

数据类型与更新

INT，循环

值

0：固定进给率未选择

1 - 4：固定进给率 1 至 4

关联

- MD12200 $MN_RUN_OVERRIDE_0
  倍率为 0 时的运行行为
- MD12202 $MN_PERMANENT_FEED
  用于直线轴的固定进给率
- MD12204 $MN_PERMANENT_ROT_AX_FEED
  用于回转轴的固定进给率

更多信息

- 轴和主轴功能手册；“进给”>“进给控制”>“固定进给值”章节

12.3.24.32 pathRapidTraversOvr

PLC → NC	<Chan>.basic.out.pathRapidTraversOvr
轨迹快速运行叠加处于活动状态	

说明

若轨迹快速移动倍率未生效，则控制系统内部以独立于当前开关位置的方式将值 1.0 ≙ 100 % 用作系数。特例是第一开关位置。就第一开关位置而言，根据所选择的编码，以下系数生效：

- 二进制编码：0
- 格雷编码：MD12050 $MN_OVR_FACTOR_RAPID_TRA[0]

数据类型与更新

BOOL，循环

值 TRUE

轨迹快速运行倍率 <Chan>.basic.out.pathRapidTraversOvrFactor 有效。

值 FALSE

轨迹快速运行倍率无效。

关联

- <Chan>.basic.out.pathRapidTraversOvrFactor
  可以通过二进制或格雷码编码形式规定轨迹快速运行叠加系数 (MD12050 $MN_OVR_FACTOR_RAPID_TRA)
- MD12020 $MN_OVR_FEED_IS_GRAY_CODE
  轨迹进给倍率开关，格雷码编码
- MD12030 $MN_OVR_FACTOR_FEEDRATE
  轨迹进给倍率开关的权重系数

更多信息

- 轴和主轴功能手册；“进给”>“进给控制”>“通过机床控制面板设置进给倍率”章节

12.3.24.33 pathRapidTraversOvrFactor

PLC → NC	<Chan>.basic.out.pathRapidTraversOvrFactor
可以通过二进制或格雷码编码形式规定轨迹快速运行叠加系数 (MD12050 $MN_OVR_FACTOR_RAPID_TRA)	

数据类型与更新

BYTE，循环

二进制编码时的值

采用二进制码时，接口中的值对应倍率系数。

二进制编码	十进制	倍率系数
0000 0000	0	0.00
0000 0001	1	0.01
0000 0010	2	0.02
0000 0011	3	0.03

二进制编码	十进制	倍率系数
000 0100	4	0.04
...	...	...
0110 0100	100	1.00
...	...	...
1100 1000	200	2.00

格雷码编码时的值

以下机床数据可以指定接口格雷码形式的数值对应的倍率系数：

MD12050 $MN_OVR_FACTOR_RAPID_TRA[<开关位置> - 1] = <倍率系数>

开关位置	格雷码	倍率系数 [1)]
1	00001	0.00
2	00011	0.01
3	00010	0.02
4	00110	0.04
5	00111	0.06
6	00101	0.08
7	00100	0.10
8	01100	0.20
9	01101	0.30
10	01111	0.40
11	01110	0.50
12	01010	0.60
13	01011	0.70
14	01001	0.75
15	01000	0.80
16	11000	0.85
17	11001	0.90
18	11011	0.95
19	11010	1.00
20	11110	1.00
21	11111	1.00

开关位置	格雷码	倍率系数 1)
22	11101	1.00
23	11100	1.00
24	10100	1.00
25	10101	1.00
26	10111	1.00
27	10110	1.00
28	10010	1.00
29	10011	1.00
30	10001	1.00
31	10000	1.00

1) 缺省值

限制

- 控制系统内部将生效的倍率系数限制在 2.00 或 200% 内。
- 可通过以下机床数据将最大倍率系数限制为一个小于 200% 的值：
  MD12100 $MN_OVR_FACTOR_LIMIT_BIN

说明

在通过以下功能进行螺纹切削期间，轨迹快速移动倍率不生效：G33、G331、G332、G63

关联

- <Chan>.basic.out.pathRapidTraversOvr
  轨迹快速运行叠加处于活动状态

更多信息

- 轴和主轴功能手册；“进给”>“进给控制”>“通过机床控制面板设置进给倍率”章节

12.3.24.34 progTest

PLC → NC	<Chan>.basic.out.progTest
请求激活程序测试 (PRT)	

说明

如果程序测试（PRT）生效，则在执行程序段或 NC 程序时加工轴不运动。但在操作界面上通过变化的设定位置显示轴运动。

说明

程序测试（PRT）生效期间，所有轴（非主轴）的运行都在“轴禁止”下执行。

说明

在 SINUMERIK Operate 操作界面的“自动”>“程序控制”操作区域将 HMI/PLC 接口信号 <HmiChan>.basic.in.progTestSelected 置位，可以选择 PRT（程序测试）。

注意
NC 内部刀具管理和刀库的一致性 程序测试时轴被禁用，因此刀库配置保持不变。用户/机床制造商必须通过相适应的 PLC 用户程序确保 NC 内部的刀具管理与实际的刀库布局一致。

数据类型和更新

BOOL，循环

值 TRUE

已请求激活程序测试 (PRT)。

值 FALSE

未请求激活程序测试 (PRT)。

关联

- <HmiChan>.basic.in.progTestSelected
  HMI：已请求程序测试 (PRT)
- <Chan>.basic.in.progTestSelecte
  已选程序测试 (PRT)

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节；“程序测试”章节

12.3.24.35 ptpTravers

PLC → NC	<Chan>.basic.out.ptpTravers
激活点对点运行 (PTP)	

说明

借助此信号，也可在 JOG 运行方式下在坐标轨迹运动（CP）与坐标 PTP 运动之间进行切换。

说明

只有转换处于活动状态时，变量才与 JOG 运行方式有关。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

激活 PTP (Point-To-Point) 运动。

脉冲沿切换 1 → 0

激活 CP (Continuous-Path) 运动。

关联

- <Chan>.basic.in.ptpTraversActive
  PTP（点对点）运行处于活动状态

更多信息

- 功能手册之工艺

12.3.24.36 refPoint

PLC → NC	<Chan>.basic.out.refPoint
脉冲沿切换 0 → 1：请求“回参考点运行”功能	

说明

应答信号：<Chan>.basic.in.refActive（“回参考点运行”功能处于活动状态）

说明

可借助以下轴专用机床数据定义，在通道专用回参考点中，机床轴按何种顺序回参考点：
MD34110 $MA_REFP_CYCLE_NR

在通道专用回参考点过程中，机床数据 MD34110 $MA_REFP_CYCLE_NR 中记录的所有轴都到达参考点后，接口信号 <Chan>.basic.in.allAxesStationary（通道的所有轴都停止，且无继续运行指令）置位。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

已请求“回参考点”功能。

脉冲沿切换 1 → 0

无作用

关联

- <Chan>.basic.in.refActive
  “回参考点运行”功能处于活动状态
- <Chan>.basic.in.allAxesStationary
  通道的所有轴都停止，且无继续运行指令。
- MD34110 $MA_REFP_CYCLE_NR
  通道专用回参考点运行中的轴顺序

更多信息

- 功能手册之进给轴和主轴；章节“回参考点”

12.3.24.37 reposModeActivationRequest

PLC → NC	<Chan>.basic.out.reposModeActivationRequest
请求为当前主运行程序段激活“REPOS”运行方式	

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

请求为当前主处理程序段激活"REPOS"功能。

激活：

- <Chan>.basic.out.reposModeSelection
  REPOS 接近运行方式
- <Axis>.basic.out.reposDelay
  已请求 REPOS 延迟

脉冲沿切换 1 → 0

将最后的请求复位。

值	REPOS 接近运行方式
0	无 REPOS 运行方式
1	接近程序段起点 (RMBBL)
2	接近中断点 (RMIBL)
3	接近程序段终点 (RMEBL)
4	接近下一个轨迹点 (RMNBL)

关联

- <HmiChan>.basic.in.reposModeChanged
  HMI：REPOS 运行方式切换
- <Chan>.basic.out.reposModeSelection
  REPOS 接近运行方式：
  可能的值见上表
- <Axis>.basic.out.reposDelay
  已请求 REPOS 延迟
- <Chan>.basic.in.reposModeChangeAck
  应答 REPOS 接近运行方式的变更请求
- <Chan>.basic.in.reposModeSelectedNumber
  已选 REPOS 接近运行方式：
  可能的值见上表
- <Chan>.basic.in.reposDelayActiv
  "在 REPOS 接近运行方式下延迟"处于活动状态
- <Axis>.basic.in.reposOffsetActive
  REPOS 偏移已应用在轴上

- <Axis>.basic.in.reposOffsetValid
  REPOS 偏移计算为有效
- <Axis>.basic.in.reposDelayAck
  已应答 REPOS 延迟
- MD11470 $MN_REPOS_MODE_MASK
  重新定位特性

更多信息

- 基础功能功能手册
  – “连续轨迹控制运行、准停、预读”>“精磨和重新定位 (REPOS)”章节
  – “运行方式组、通道、程序运行、复位行为”>“再次返回轮廓 (REPOS)”章节

12.3.24.38 reposModeSelection

PLC → NC	<Chan>.basic.out.reposModeSelection
REPOS 接近运行方式： 可能的值见下表	

数据类型与更新

INT，循环

值	REPOS 接近运行方式
0	无 REPOS 运行方式
1	接近程序段起点 (RMBBL)
2	接近中断点 (RMIBL)
3	接近程序段终点 (RMEBL)
4	接近下一个轨迹点 (RMNBL)

关联

- <HmiChan>.basic.in.reposModeChanged
  HMI：REPOS 运行方式切换
- <Chan>.basic.out.reposModeActivationRequest
  请求为当前主运行程序段激活“REPOS”运行方式
- <Axis>.basic.out.reposDelay
  已请求 REPOS 延迟

- <Chan>.basic.in.reposModeChangeAck
  应答 REPOS 接近运行方式的变更请求
- <Chan>.basic.in.reposModeSelectedNumber
  已选 REPOS 接近运行方式：
  可能的值见上表
- <Chan>.basic.in.reposDelayActive
  “在 REPOS 接近运行方式下延迟”处于活动状态
- <Axis>.basic.in.reposOffsetActive
  REPOS 偏移已应用在轴上
- <Axis>.basic.in.reposOffsetValid
  REPOS 偏移计算为有效
- <Axis>.basic.in.reposDelayAck
  已应答 REPOS 延迟
- MD11470 $MN_REPOS_MODE_MASK
  重新定位特性

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.24.39 reset

PLC → NC	<Chan>.basic.out.reset
已请求通道复位	

说明

响应：

- 所有运行的轴和主轴均根据其加速度特性曲线，在轮廓范围内制动至静止状态。
- 所有通道专用初始设置生效。
- 所有通道报警（上电报警除外）均被删除。
- 通道状态：“复位”(<Chan>.basic.in.stateReset == TRUE)
- 生效的 NC 程序的程序状态：“取消”(<Chan>.basic.in.progStateAborted) == TRUE)

说明

通道复位后，在通过 PLC 用户程序请求新的动作（例如 NC 启动或轴交换）前必须先等待，直至 NC/PLC 接口上显示“复位”通道状态：

<Chan>.basic.in.stateReset == TRUE

数据类型与更新

BOOL，循环

值 TRUE

已请求通道复位。

值 FALSE

未请求通道复位。

关联

- <ModeGroup>.basic.out.reset
  运行方式组复位
- <Chan>.basic.out.ncStart
  脉冲沿切换 0 → 1：请求开始运行在通道中选择的程序
- <Chan>.basic.out.ncStop
  请求 NC 停止
- <Chan>.basic.in.progStateAborted
  程序状态“取消”
- <Chan>.basic.in.stateReset
  通道状态“复位”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.24.40 retraceSupportStart

PLC → NC	<Chan>.basic.out.retraceSupportStart
重新返回轮廓 (RESU)：脉冲沿切换 0 → 1：继续加工轮廓；请求“开始重新返回轮廓”	

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

RESU：激活返回支持

12.3.24.41 retraceSupportTraversBackwards

PLC → NC	<Chan>.basic.out.retraceSupportTraversBackwards
重新返回轮廓 (RESU)：继续加工轮廓；请求“向后运行”	

数据类型与更新

BOOL

12.3.24.42 singleBlock

PLC → NC	<Chan>.basic.out.singleBlock
请求激活“单段处理”(SBL) 功能	

说明

在 AUTO 和 MDI 运行方式中，操作人员必须通过重新触发 NC 启动为通道中所选零件程序的每个单程序段使能执行。

数据类型与更新

BOOL，循环

值 TRUE

已请求激活“单程序段”功能。

值 FALSE

未请求激活“单程序段”功能。

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”>“程序控制”>“单段”章节

12.3.24.43 skipBlockLevel0..9

PLC → NC	<Chan>.basic.out.skipBlockLevel0..9
请求为 0 - 9 级激活跳转零件程序段的功能 (Skip, SKP)	

说明

在存在多个连续的跳转程序段的情况下，仅当接口信号在程序段序列中的第一个跳转程序段解码前便已存在的情况下，才跳跃这些程序段。

在 SINUMERIK Operate 操作界面的“自动”>“程序控制”操作区域将 HMI/PLC 接口信号 <HmiChan>.basic.in.skipBlockLevel0..9 置位，可以选择“程序段跳跃”(SKP)。

说明

建议在 NC 程序启动前便将此接口信号置位。

数据类型与更新

BOOL，循环

值 TRUE

已请求激活跳转级的“程序段跳跃”(SKP)。

值 FALSE

未请求激活跳转级的“程序段跳跃”(SKP)。

关联

- <HmiChan>.basic.in.skipBlockLevel0..9
  已选择跳转跳转级 0 - 9 的某个零件程序段 (Skip, SKP)

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.24.44 suppressDisableNcStart

PLC → NC	<Chan>.basic.out.suppressDisableNcStart
请求抑制“锁定 NC 启动”	

数据类型与更新

BOOL

值 TRUE

抑制启动锁定

值 FALSE

不抑制启动锁定

12.3.25 <Chan>.gCommandGroup.in

12.3.25.1 activeValues

NC → PLC	<Chan>.gCommandGroup.in.activeValues
G 指令组 1..64 中当前生效的 G 指令的内部编号	

说明

以 BCD 格式显示的 G 指令或其记忆名称在给定的 G 指令组中生效。

- 特殊情况
  在值为 0 时，G 指令组中无 G 指令或记忆名称生效。
- 上电后的初始位置
  上电后，系统会为所有 G 指令组传输值零，即生效的 G 指令组未定义。
- 零件程序结束或终止
  零件程序结束或终止后，保留最后生效的 G 指令。
- NC 启动
  NC 启动后，以下机床数据中设定的 8 个 G 指令组的值会根据通过机床数据定义的初始设置以及零件程序中编写的值改写：
  MD22510 $MC_GCODE_GROUPS_TO_PLC

说明

与辅助功能不同，G 指令在无应答控制的情形下输出至 PLC，即零件程序会在 G 指令输出后立即继续执行。

说明

- 组中有效的 G 功能在编程每个 G 功能或记忆标识符（例如：SPLINE）时会更新。
- G 组中的 G 功能输出形式为双字，从 1 开始。值为 0 的 G 功能表示该 G 组中没有有效的 G 功能。

数据类型和更新

USInt，循环

值

- G 指令组 1：生效的 G 指令的内部编号
- G 功能组 2:生效的 G 指令的内部编号

- ...
- G 指令组 64：生效的 G 指令的内部编号

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节
- G 指令组和 G 指令及其内部编号的完整列表请见：
  编程手册之基本原理；章节“表格” > “G 指令”

12.3.25.2 g00Active

NC → PLC	<Chan>.gCommandGroup.in.g00Active
G00 快速运行处于活动状态	

数据类型与更新

BOOL

值 TRUE

G 功能 G00 处于活动状态

值 FALSE

G 功能 G00 未激活

12.3.26 <Chan>.geoAxis1.in

12.3.26.1 contManualTraversActive

NC → PLC	<Chan>.geoAxis1.in.contManualTraversActive
连续运行几何轴（正常运行）	

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型与更新

BOOL，循环

值 TRUE

用于连续运行几何轴的机床功能处于活动状态。

值 FALSE

用于连续运行几何轴的机床功能未激活。

关联

- <Chan>.geoAxis1.out.inc1
  “根据请求将几何轴移动 1 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc10
  “根据请求将几何轴移动 10 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc100
  “根据请求将几何轴移动 100 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc1000
  “根据请求将几何轴移动 1 000 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc10000
  “根据请求将几何轴移动 10 000 个增量”处于活动状态
- <Chan>.geoAxis1.out.incVar
  激活“根据请求移动几何轴，移动幅度为用户自定义增量数”
- <Chan>.geoAxis1.out.contManualTravers
  激活连续运行几何轴（正常运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.26.2 handwheelAssignedNumber

NC → PLC	<Chan>.geoAxis1.in.handwheelAssignedNumber
当前所分配手轮的编号：1..6 或 0 未分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）	

说明

当前所分配手轮的编号。

该接口使用了二进制编码。通过机床数据 MD11324=1 进行定义。

数据类型和更新

USInt，循环

值

0：未分配手轮

1...6：当前所分配手轮的编号

关联

- MD11324 $MN_HANDWH_VDI_REPRESENTATION
  VDI 接口中的手轮编号显示
- <Chan>.geoAxis1.out.handwheelSelection
  将手轮 1 - 6（0 代表未选择）分配给几何轴，比特或二进制编码 (MD11324 $MN_HANDWH_VDI_REPRESENTATION)

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.26.3 handwheelDirInverted

NC → PLC	<Chan>.geoAxis1.in.handwheelDirInverted
“反转分配给几何轴的手轮的旋转方向”处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

与几何轴对应的手轮的旋转方向取反生效。

值 FALSE

与几何轴对应的手轮的旋转方向取反未生效。

关联

- <Chan>.geoAxis1.out.handwheelInvertDir
  请求反转分配给几何轴的手轮的旋转方向

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.26.4 inc1Active

NC → PLC	<Chan>.geoAxis1.in.inc1Active
“根据请求将几何轴移动 1 个增量”处于活动状态	

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求通过 INC1 进行手动操作。

值 FALSE

针对定向轴，未请求通过 INC1 进行手动操作的机床功能。

关联

- <Chan>.geoAxis1.out.inc1
  “根据请求将几何轴移动 1 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc10
  “根据请求将几何轴移动 10 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc100
  “根据请求将几何轴移动 100 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc1000
  “根据请求将几何轴移动 1000 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc10000
  “根据请求将几何轴移动 10000 个增量”处于活动状态
- <Chan>.geoAxis1.out.incVar
  激活“根据请求移动几何轴，移动幅度为用户自定义增量数”
- <Chan>.geoAxis1.out.contManualTravers
  激活连续运行几何轴（正常运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.26.5 inc10Active

NC → PLC	<Chan>.geoAxis1.in.inc10Active
“根据请求将几何轴移动 10 个增量”处于活动状态	

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求通过 INC10 进行手动操作。

值 FALSE

针对定向轴，未请求通过 INC10 进行手动操作的机床功能。

关联

- <Chan>.geoAxis1.out.inc1
  “根据请求将几何轴移动 1 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc10
  “根据请求将几何轴移动 10 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc100
  “根据请求将几何轴移动 100 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc1000
  “根据请求将几何轴移动 1000 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc10000
  “根据请求将几何轴移动 10000 个增量”处于活动状态
- <Chan>.geoAxis1.out.incVar
  激活“根据请求移动几何轴，移动幅度为用户自定义增量数”
- <Chan>.geoAxis1.out.contManualTravers
  激活连续运行几何轴（正常运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.26.6 inc100Active

NC → PLC	<Chan>.geoAxis1.in.inc100Active
“根据请求将几何轴移动 100 个增量”处于活动状态	

说明

几何轴：当前活动的机床功能

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求通过 INC100 进行手动操作。

值 FALSE

针对定向轴，未请求通过 INC100 进行手动操作的机床功能。

关联

- <Chan>.geoAxis1.out.inc1
  “根据请求将几何轴移动 1 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc10
  “根据请求将几何轴移动 10 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc100
  “根据请求将几何轴移动 100 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc1000
  “根据请求将几何轴移动 1000 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc10000
  “根据请求将几何轴移动 10000 个增量”处于活动状态
- <Chan>.geoAxis1.out.incVar
  激活“根据请求移动几何轴，移动幅度为用户自定义增量数”
- <Chan>.geoAxis1.out.contManualTravers
  激活连续运行几何轴（正常运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.26.7 inc1000Active

NC → PLC	<Chan>.geoAxis1.in.inc1000Active
“根据请求将几何轴移动 1000 个增量”处于活动状态	

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求通过 INC1000 进行手动操作。

值 FALSE

针对定向轴，未请求通过 INC1000 进行手动操作的机床功能。

关联

- <Chan>.geoAxis1.out.inc1
  “根据请求将几何轴移动 1 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc10
  “根据请求将几何轴移动 10 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc100
  “根据请求将几何轴移动 100 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc1000
  “根据请求将几何轴移动 1000 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc10000
  “根据请求将几何轴移动 10000 个增量”处于活动状态
- <Chan>.geoAxis1.out.incVar
  激活“根据请求移动几何轴，移动幅度为用户自定义增量数”
- <Chan>.geoAxis1.out.contManualTravers
  激活连续运行几何轴（正常运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.26.8 inc10000Active

NC → PLC	<Chan>.geoAxis1.in.inc10000Active
“根据请求将几何轴移动 10000 个增量”处于活动状态	

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求通过 INC10000 进行手动操作。

值 FALSE

针对定向轴，未请求通过 INC10000 进行手动操作的机床功能。

关联

- <Chan>.geoAxis1.out.inc1
  “根据请求将几何轴移动 1 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc10
  “根据请求将几何轴移动 10 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc100
  “根据请求将几何轴移动 100 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc1000
  “根据请求将几何轴移动 1000 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc10000
  “根据请求将几何轴移动 10000 个增量”处于活动状态
- <Chan>.geoAxis1.out.incVar
  激活“根据请求移动几何轴，移动幅度为用户自定义增量数”
- <Chan>.geoAxis1.out.contManualTravers
  激活连续运行几何轴（正常运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.26.9 incVarActive

NC → PLC	<Chan>.geoAxis1.in.incVarActive
“根据请求移动几何轴，移动幅度为用户自定义增量数”处于活动状态	

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求用于手动运行几何轴的机床功能。

值 FALSE

未请求用于手动运行几何轴的机床功能。

关联

- <Chan>.geoAxis1.out.inc1
  “根据请求将几何轴移动 1 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc10
  “根据请求将几何轴移动 10 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc100
  “根据请求将几何轴移动 100 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc1000
  “根据请求将几何轴移动 1000 个增量”处于活动状态
- <Chan>.geoAxis1.out.inc10000
  “根据请求将几何轴移动 10000 个增量”处于活动状态
- <Chan>.geoAxis1.out.incVar
  激活“根据请求移动几何轴，移动幅度为用户自定义增量数”
- <Chan>.geoAxis1.out.contManualTravers
  激活连续运行几何轴（正常运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.26.10 traversMinusCommand

NC → PLC	<Chan>.geoAxis1.in.traversMinusCommand
已请求几何轴沿负方向移动	

说明

视 MD17900, Bit 0 的设置而定，当存在“运行请求” (Bit 0 = FALSE) 或轴实际移动时 (Bit 0 = TRUE)，输出运行指令。

应用示例

一旦检测到运行指令就松开轴，使轴运行。

说明

对于在检测到运行指令时才松开夹紧装置的轴而言，不可采用连续路径运行（G64）。

数据类型和更新

BOOL，循环

值 TRUE

针对几何轴存在轴运行请求，或者几何轴沿负轴方向运行。

值 FALSE

对于几何轴而言不存在运行请求，或者几何轴不移动。

关联

- <Chan>.geoAxis1.out.traversKeyMinus
  请求几何轴“负”移动键信号（用于负轴方向运行）
- <Chan>.geoAxis1.out.traversKeyPlus
  请求几何轴“正”移动键信号（用于正轴方向运行）
- <Chan>.geoAxis1.in.traversMinusRequest
  已请求几何轴沿负方向移动
- <Chan>.geoAxis1.in.traversPlusRequest
  已请求几何轴沿正方向移动
- MD17900 $MN_VDI_FUNCTION_MASK
  VDI 信号设置

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.26.11 traversMinusRequest

NC → PLC	<Chan>.geoAxis1.in.traversMinusRequest
已请求几何轴沿负方向移动	

说明

视运行方式而定，以不同的方式触发运行请求：

- JOG 运行方式
  移动键"+"或"-"
- 运行方式 REF
  触发轴朝参考点运行的移动键。
- AUTO 或 MDI 运行方式
  执行一个包含几何轴运行指令的程序段。

数据类型与更新

BOOL，循环

值 TRUE

针对几何轴，存在负轴方向运行请求。

值 FALSE

几何轴当前没有运行请求。

关联

- <Chan>.geoAxis1.out.traversKeyMinus
  请求几何轴“负”移动键信号（用于负轴方向运行）
- <Chan>.geoAxis1.out.traversKeyPlus
  请求几何轴“正”移动键信号（用于正轴方向运行）
- <Chan>.geoAxis1.in.traversMinusCommand
  几何轴负方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)
- <Chan>.geoAxis1.in.traversPlusCommand
  几何轴正方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.26.12 traversPlusCommand

NC → PLC	<Chan>.geoAxis1.in.traversPlusCommand
几何轴正方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)	

说明

视 MD17900, Bit 0 的设置而定，当存在“运行请求” (Bit 0 = FALSE) 或轴实际移动时 (Bit 0 = TRUE)，输出运行指令。

应用示例

一旦检测到运行指令就松开轴，使轴运行。

说明

对于在检测到运行指令时才松开夹紧装置的轴而言，不可采用连续路径运行（G64）。

数据类型与更新

BOOL，循环

值 TRUE

针对几何轴存在轴运行请求，或者几何轴沿正轴方向运行。

值 FALSE

对于几何轴而言不存在运行请求，或者几何轴不移动。

关联

- <Chan>.geoAxis1.out.traversKeyMinus
  请求几何轴“负”移动键信号（用于负轴方向运行）
- <Chan>.geoAxis1.out.traversKeyPlus
  请求几何轴“正”移动键信号（用于正轴方向运行）
- <Chan>.geoAxis1.in.traversMinusRequest
  已请求几何轴沿负方向移动
- <Chan>.geoAxis1.in.traversPlusRequest
  已请求几何轴沿正方向移动
- MD17900 $MN_VDI_FUNCTION_MASK
  VDI 信号设置

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.26.13 traversPlusRequest

NC → PLC	<Chan>.geoAxis1.in.traversPlusRequest
已请求几何轴沿正方向移动	

说明

视运行方式而定，以不同的方式触发运行请求：

- JOG 运行方式
  移动键"+"或"-"
- 运行方式 REF
  触发轴朝参考点运行的移动键。
- AUTO 或 MDI 运行方式
  执行一个包含几何轴运行指令的程序段。

数据类型与更新

BOOL，循环

值 TRUE

针对几何轴，存在正轴方向运行请求。

值 FALSE

几何轴当前没有运行请求。

关联

- <Chan>.geoAxis1.out.traversKeyMinus
  请求几何轴“负”移动键信号（用于负轴方向运行）
- <Chan>.geoAxis1.out.traversKeyPlus
  请求几何轴“正”移动键信号（用于正轴方向运行）
- <Chan>.geoAxis1.in.traversMinusCommand
  几何轴负方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)
- <Chan>.geoAxis1.in.traversPlusCommand
  几何轴正方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.27 <Chan>.geoAxis1.out

12.3.27.1 contManualTravers

PLC → NC	<Chan>.geoAxis1.out.contManualTravers
激活连续运行几何轴（正常运行）	

说明

连续手动运行时，借助移动键"+"或"-"使得几何轴沿相应的方向连续移动。

说明

- 若同时置位多个请求，将无机床功能生效。
- 如果在几何轴当前正在执行某机床功能时取消该功能或者切换到另一个功能，则几何轴终止移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求用于连续运行几何轴的机床功能。

值 FALSE

未请求用于连续运行几何轴的机床功能。

关联

- <Chan>.geoAxis1.in.inc1Active
  "根据请求将几何轴移动 1 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc10Active
  "根据请求将几何轴移动 10 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc100Active
  "根据请求将几何轴移动 100 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc1000Active
  "根据请求将几何轴移动 1 000 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc10000Active
  "根据请求将几何轴移动 10 000 个增量"处于活动状态
- <Chan>.geoAxis1.in.incVarActive
  "根据请求移动几何轴，移动幅度为用户自定义增量数"处于活动状态

- <Chan>.geoAxis1.in.contManualTraversActive
  连续运行几何轴（正常运行）
- MD11320 $MN_HANDWH_IMP_PER_LATCH
  手轮每个卡槽位置的脉冲数
- MD11330 $MN_JOG_INCR_SIZE_TAB
  INC/手轮运行时的增量大小
- MD31090 $MA_JOG_INCR_WEIGHT
  INC/手轮运行时的增量加权
- SD41010 $SN_JOG_VAR_INCR_SIZE
  JOG 模式下的可变增量大小

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.27.2 disableFeed

PLC → NC	<Chan>.geoAxis1.out.disableFeed
锁定几何轴进给	

说明

- 若在几何轴运行期间将此接口信号置位，则几何轴通过其制动特性曲线制动至静止状态。
- 在此接口信号复位后，系统会继续执行停止的运行。
- 位置闭环控制保持生效，跟随误差会减小。
- 若此接口置位，并且请求轴的运行，则轴不运行。但运行请求仍然保留。在此接口信号复位后，系统会继续执行运行请求，即运行轴。
- 此接口信号仅在 JOG 运行方式下生效。

数据类型与更新

BOOL，循环

值 TRUE

几何轴的进给禁止生效。

值 FALSE

几何轴的进给禁止未生效。

12.3.27.3 disableTraversKey

PLC → NC	<Chan>.geoAxis1.out.disableTraversKey
锁定几何轴移动键	

说明

几何轴：移动键锁定

说明

在运行期间激活移动键禁用的情况下，运行会终止。

数据类型与更新

BOOL，循环

值 TRUE

针对几何轴的移动键禁用生效。

值 FALSE

针对几何轴的移动键禁用未生效。

关联

- <Chan>.geoAxis1.out.traversKeyMinus
  请求几何轴“负”移动键信号（用于负轴方向运行）
- <Chan>.geoAxis1.out.traversKeyPlus
  请求几何轴“正”移动键信号（用于正轴方向运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.27.4 handwheelInvertDir

PLC → NC	<Chan>.geoAxis1.out.handwheelInvertDir
请求反转分配给几何轴的手轮的旋转方向	

说明

只允许在几何轴处于静止时修改该取反信号。

说明

应用示例

- 手轮旋转方向和预期的轴移动方向不一致。
- 手轮指定给了多根定向轴。

数据类型与更新

BOOL，循环

值 TRUE

已请求将与几何轴对应的手轮的旋转方向取反。

值 FALSE

未请求将与几何轴对应的手轮的旋转方向取反。

关联

- <Chan>.geoAxis1.in.handwheelDirInverted
  “反转分配给几何轴的手轮的旋转方向”处于活动状态

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.27.5 handwheelSelection

PLC → NC	<Chan>.geoAxis1.out.handwheelSelection
手轮 1..6 已分配或 0 未分配手轮（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）	

说明

所分配手轮的编号。

该接口使用了二进制编码。通过机床数据 MD11324=1 进行定义。

数据类型和更新

USInt，循环

值

0：无手轮分配

1...6：所分配手轮的编号

关联

- MD11324 $MN_HANDWH_VDI_REPRESENTATION
  VDI 接口中的手轮编号显示
- <Chan>.geoAxis1.in.handwheelAssignedNumber
  已分配手轮 1 - 6（0 代表未选择）的编号，比特或二进制编码 (MD11324 $MN_HANDWH_VDI_REPRESENTATION)

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.27.6 inc1

PLC → NC	<Chan>.geoAxis1.out.inc1
激活“根据请求将几何轴移动 1 个增量”	

说明

除了五个固定步数（MD11330 中的缺省设置：INC1、INC10、INC100、INC1000 和 INC10000），还有可由设定数据 SD41010 设置的可变步数（INCvar）。固定步数和可变步数的每一步的长度通过轴专用机床数据 MD31090 设置。

通过按下移动键“+”或“-”，或通过旋转电子手轮，几何轴开始沿相应的方向移动与生效的机床功能对应的步数。

说明

- 若同时置位多个请求，将无机床功能生效。
- 如果在几何轴当前正在执行某机床功能时取消该功能或者切换到另一个功能，则几何轴终止移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求用于手动运行几何轴的机床功能。

值 FALSE

未请求用于手动运行几何轴的机床功能。

关联

- <Chan>.geoAxis1.in.inc1Active
  “根据请求将几何轴移动 1 个增量”处于活动状态
- <Chan>.geoAxis1.in.inc10Active
  “根据请求将几何轴移动 10 个增量”处于活动状态
- <Chan>.geoAxis1.in.inc100Active
  “根据请求将几何轴移动 100 个增量”处于活动状态
- <Chan>.geoAxis1.in.inc1000Active
  “根据请求将几何轴移动 1 000 个增量”处于活动状态
- <Chan>.geoAxis1.in.inc10000Active
  “根据请求将几何轴移动 10 000 个增量”处于活动状态
- <Chan>.geoAxis1.in.incVarActive
  “根据请求移动几何轴，移动幅度为用户自定义增量数”处于活动状态
- <Chan>.geoAxis1.in.contManualTraversActive
  连续运行几何轴（正常运行）
- MD11320 $MN_HANDWH_IMP_PER_LATCH
  手轮每个卡槽位置的脉冲数
- MD11330 $MN_JOG_INCR_SIZE_TAB
  INC/手轮运行时的增量大小
- MD31090 $MA_JOG_INCR_WEIGHT
  INC/手轮运行时的增量加权
- SD41010 $SN_JOG_VAR_INCR_SIZE
  JOG 模式下的可变增量大小

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.27.7 inc10

PLC → NC	<Chan>.geoAxis1.out.inc10
激活“根据请求将几何轴移动 10 个增量”	

说明

除了五个固定步数（MD11330 中的缺省设置：INC1、INC10、INC100、INC1000 和 INC10000），还有可由设定数据 SD41010 设置的可变步数（INCvar）。固定步数和可变步数的每一步的长度通过轴专用机床数据 MD31090 设置。

通过按下移动键"+"或"-"，或通过旋转电子手轮，几何轴开始沿相应的方向移动与生效的机床功能对应的步数。

说明

- 若同时置位多个请求，将无机床功能生效。
- 如果在几何轴当前正在执行某机床功能时取消该功能或者切换到另一个功能，则几何轴终止移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求用于手动运行几何轴的机床功能。

值 FALSE

未请求用于手动运行几何轴的机床功能。

关联

- <Chan>.geoAxis1.in.inc1Active
  "根据请求将几何轴移动 1 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc10Active
  "根据请求将几何轴移动 10 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc100Active
  "根据请求将几何轴移动 100 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc1000Active
  "根据请求将几何轴移动 1 000 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc10000Active
  "根据请求将几何轴移动 10 000 个增量"处于活动状态
- <Chan>.geoAxis1.in.incVarActive
  "根据请求移动几何轴，移动幅度为用户自定义增量数"处于活动状态
- <Chan>.geoAxis1.in.contManualTraversActive
  连续运行几何轴（正常运行）

- MD11320 $MN_HANDWH_IMP_PER_LATCH
  手轮每个卡槽位置的脉冲数
- MD11330 $MN_JOG_INCR_SIZE_TAB
  INC/手轮运行时的增量大小
- MD31090 $MA_JOG_INCR_WEIGHT
  INC/手轮运行时的增量加权
- SD41010 $SN_JOG_VAR_INCR_SIZE
  JOG 模式下的可变增量大小

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.27.8 inc100

PLC → NC	<Chan>.geoAxis1.out.inc10
激活“根据请求将几何轴移动 10 个增量”	

说明

除了五个固定步数（MD11330 中的缺省设置：INC1、INC10、INC100、INC1000 和 INC10000），还有可由设定数据 SD41010 设置的可变步数（INCvar）。固定步数和可变步数的每一步的长度通过轴专用机床数据 MD31090 设置。

通过按下移动键“+”或“-”，或通过旋转电子手轮，几何轴开始沿相应的方向移动与生效的机床功能对应的步数。

说明

- 若同时置位多个请求，将无机床功能生效。
- 如果在几何轴当前正在执行某机床功能时取消该功能或者切换到另一个功能，则几何轴终止移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求用于手动运行几何轴的机床功能。

值 FALSE

未请求用于手动运行几何轴的机床功能。

关联

- <Chan>.geoAxis1.in.inc1Active
  “根据请求将几何轴移动 1 个增量”处于活动状态
- <Chan>.geoAxis1.in.inc10Active
  “根据请求将几何轴移动 10 个增量”处于活动状态
- <Chan>.geoAxis1.in.inc100Active
  “根据请求将几何轴移动 100 个增量”处于活动状态
- <Chan>.geoAxis1.in.inc1000Active
  “根据请求将几何轴移动 1 000 个增量”处于活动状态
- <Chan>.geoAxis1.in.inc10000Active
  “根据请求将几何轴移动 10 000 个增量”处于活动状态
- <Chan>.geoAxis1.in.incVarActive
  “根据请求移动几何轴，移动幅度为用户自定义增量数”处于活动状态
- <Chan>.geoAxis1.in.contManualTraversActive
  连续运行几何轴（正常运行）
- MD11320 $MN_HANDWH_IMP_PER_LATCH
  手轮每个卡槽位置的脉冲数
- MD11330 $MN_JOG_INCR_SIZE_TAB
  INC/手轮运行时的增量大小
- MD31090 $MA_JOG_INCR_WEIGHT
  INC/手轮运行时的增量加权
- SD41010 $SN_JOG_VAR_INCR_SIZE
  JOG 模式下的可变增量大小

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.27.9 inc1000

PLC → NC	<Chan>.geoAxis1.out.inc1000
激活“根据请求将几何轴移动 1 000 个增量”	

说明

除了五个固定步数（MD11330 中的缺省设置：INC1、INC10、INC100、INC1000 和 INC10000），还有可由设定数据 SD41010 设置的可变步数（INCvar）。固定步数和可变步数的每一步的长度通过轴专用机床数据 MD31090 设置。

通过按下移动键"+"或"-"，或通过旋转电子手轮，几何轴开始沿相应的方向移动与生效的机床功能对应的步数。

说明

- 若同时置位多个请求，将无机床功能生效。
- 如果在几何轴当前正在执行某机床功能时取消该功能或者切换到另一个功能，则几何轴终止移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求用于手动运行几何轴的机床功能。

值 FALSE

未请求用于手动运行几何轴的机床功能。

关联

- <Chan>.geoAxis1.in.inc1Active
  "根据请求将几何轴移动 1 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc10Active
  "根据请求将几何轴移动 10 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc100Active
  "根据请求将几何轴移动 100 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc1000Active
  "根据请求将几何轴移动 1 000 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc10000Active
  "根据请求将几何轴移动 10 000 个增量"处于活动状态
- <Chan>.geoAxis1.in.incVarActive
  "根据请求移动几何轴，移动幅度为用户自定义增量数"处于活动状态
- <Chan>.geoAxis1.in.contManualTraversActive
  连续运行几何轴（正常运行）

- MD11320 $MN_HANDWH_IMP_PER_LATCH
  手轮每个卡槽位置的脉冲数
- MD11330 $MN_JOG_INCR_SIZE_TAB
  INC/手轮运行时的增量大小
- MD31090 $MA_JOG_INCR_WEIGHT
  INC/手轮运行时的增量加权
- SD41010 $SN_JOG_VAR_INCR_SIZE
  JOG 模式下的可变增量大小

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.27.10 inc10000

PLC → NC	<Chan>.geoAxis1.out.inc10000
激活“根据请求将几何轴移动 10 000 个增量”	

说明

除了五个固定步数（MD11330 中的缺省设置：INC1、INC10、INC100、INC1000 和 INC10000），还有可由设定数据 SD41010 设置的可变步数（INCvar）。固定步数和可变步数的每一步的长度通过轴专用机床数据 MD31090 设置。

通过按下移动键“+”或“-”，或通过旋转电子手轮，几何轴开始沿相应的方向移动与生效的机床功能对应的步数。

说明

- 若同时置位多个请求，将无机床功能生效。
- 如果在几何轴当前正在执行某机床功能时取消该功能或者切换到另一个功能，则几何轴终止移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求用于手动运行几何轴的机床功能。

值 FALSE

未请求用于手动运行几何轴的机床功能。

关联

- <Chan>.geoAxis1.in.inc1Active
  “根据请求将几何轴移动 1 个增量”处于活动状态
- <Chan>.geoAxis1.in.inc10Active
  “根据请求将几何轴移动 10 个增量”处于活动状态
- <Chan>.geoAxis1.in.inc100Active
  “根据请求将几何轴移动 100 个增量”处于活动状态
- <Chan>.geoAxis1.in.inc1000Active
  “根据请求将几何轴移动 1 000 个增量”处于活动状态
- <Chan>.geoAxis1.in.inc10000Active
  “根据请求将几何轴移动 10 000 个增量”处于活动状态
- <Chan>.geoAxis1.in.incVarActive
  “根据请求移动几何轴，移动幅度为用户自定义增量数”处于活动状态
- <Chan>.geoAxis1.in.contManualTraversActive
  连续运行几何轴（正常运行）
- MD11320 $MN_HANDWH_IMP_PER_LATCH
  手轮每个卡槽位置的脉冲数
- MD11330 $MN_JOG_INCR_SIZE_TAB
  INC/手轮运行时的增量大小
- MD31090 $MA_JOG_INCR_WEIGHT
  INC/手轮运行时的增量加权
- SD41010 $SN_JOG_VAR_INCR_SIZE
  JOG 模式下的可变增量大小

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.27.11 incVar

PLC → NC	<Chan>.geoAxis1.out.incVar
激活“根据请求移动几何轴，移动幅度为用户自定义增量数”	

说明

除了五个固定步数（MD11330 中的缺省设置：INC1、INC10、INC100、INC1000 和 INC10000），还有可由设定数据 SD41010 设置的可变步数（INCvar）。固定步数和可变步数的每一步的长度通过轴专用机床数据 MD31090 设置。

通过按下移动键"+"或"-"，或通过旋转电子手轮，几何轴开始沿相应的方向移动与生效的机床功能对应的步数。

说明

- 若同时置位多个请求，将无机床功能生效。
- 如果在几何轴当前正在执行某机床功能时取消该功能或者切换到另一个功能，则几何轴终止移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求用于手动运行几何轴的机床功能。

值 FALSE

未请求用于手动运行几何轴的机床功能。

关联

- <Chan>.geoAxis1.in.inc1Active
  "根据请求将几何轴移动 1 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc10Active
  "根据请求将几何轴移动 10 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc100Active
  "根据请求将几何轴移动 100 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc1000Active
  "根据请求将几何轴移动 1 000 个增量"处于活动状态
- <Chan>.geoAxis1.in.inc10000Active
  "根据请求将几何轴移动 10 000 个增量"处于活动状态
- <Chan>.geoAxis1.in.incVarActive
  "根据请求移动几何轴，移动幅度为用户自定义增量数"处于活动状态
- <Chan>.geoAxis1.in.contManualTraversActive
  连续运行几何轴（正常运行）

- MD11320 $MN_HANDWH_IMP_PER_LATCH
  手轮每个卡槽位置的脉冲数
- MD11330 $MN_JOG_INCR_SIZE_TAB
  INC/手轮运行时的增量大小
- MD31090 $MA_JOG_INCR_WEIGHT
  INC/手轮运行时的增量加权
- SD41010 $SN_JOG_VAR_INCR_SIZE
  JOG 模式下的可变增量大小

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.27.12 rapidTraversOvr

PLC → NC	<Chan>.geoAxis1.out.rapidTraversOvr
激活几何轴快速运行速度	

说明

在 JOG 运行方式下连续或断续手动运行时，此信号才生效。

信号与以下情形不相关：

- 在回参考点运行中（JOG 运行方式）
- 在 AUTO 和 MDI 运行方式下

快进速度可通过快进倍率开关调整。

数据类型与更新

BOOL，循环

值 TRUE

在借助移动键手动运行几何轴时，此接口信号的置位使快进速度（MD32010）生效。

值 FALSE

在借助移动键手动运行几何轴时，设定的 JOG 速度（SD41110 或 MD32020）生效。

关联

- <Chan>.geoAxis1.out.traversKeyMinus
  请求几何轴“负”移动键信号（用于负轴方向运行）
- <Chan>.geoAxis1.out.traversKeyPlus
  请求几何轴“正”移动键信号（用于正轴方向运行）
- MD32010 $MA_JOG_VELO_RAPID
  常规快速运行
- MD32020 $MA_JOG_VELO
  常规轴速度
- SD41110 $SN_JOG_SET_VELO
  JOG 模式下的轴速度

更多信息

- 轴和主轴功能手册；“进给”章节
- 轴和主轴功能手册；“手动运行”章节

12.3.27.13 traversKeyMinus

PLC → NC	<Chan>.geoAxis1.out.traversKeyMinus
请求几何轴“负”移动键信号（用于负轴方向运行）	

说明

根据生效的机床功能以及点按及长按模式（SD41050 和 MD11300）的设置，在信号切换时触发不同的响应：

1. 在点按模式下连续手动运行
   几何轴一直沿相应方向移动至活动的限位开关，如同接口信号具有信号状态 TRUE 的情形。
2. 在长按模式下连续手动运行
   在该信号发出首个上升沿 0 → 1 后，几何轴开始朝对应方向移动。即使之后信号输出下降沿 1 → 0，轴也继续移动。在该信号再次给出上升沿 0 → 1 后（同一个移动方向！），轴终止移动。
3. 在点按模式下断续手动运行
   信号状态为 TRUE 时，几何轴开始按照设置的增量运行。如果运行完增量前，信号切换至 FALSE 状态，这种情况下中断运行。信号状态恢复为 TRUE 后，继续运行。在没有移动完指定步数前，轴的移动可以如上文所述暂停和继续。
4. 在长按模式下断续手动运行
   在该信号输出第一个上升沿 0 → 1 后，几何轴开始移动指定的步数。如果在几何轴移动完步数前，同一个方向信号再次输出上升沿 0 → 1，则终止移动。将不会把步数移动完。

说明

- 此接口信号与以下运行方式不相关：
  - AUTOMATIC
  - MDA
- 若两个移动信号（"+"和"-"）同时置位，则不移动或终止当前移动。
- 和机床轴不同的是，在一个时间点上只能通过移动键移动一根几何轴。
- 通过几何轴移动键锁定功能，可以借助移动键将运行锁定。
- 在以下情形下，不可在 JOG 运行方式下移动几何轴：
  - 已通过轴专用接口将几何轴作为机床轴移动。
  - 已通过移动键使另一几何轴移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求借助移动键沿负轴方向移动几何轴。

值 FALSE

未请求借助移动键运行几何轴。

关联

- <Chan>.geoAxis1.out.disableTraversKey
  锁定几何轴移动键
- <Axis>.basic.out.traversKeyPlus
  在 JOG 运行方式下沿正方向移动机床轴 (MD32020 $MA_JOG_VELO)
- MD11300 $MN_JOG_INC_MODE_LEVELTRIGGRD
  INC 和 REF 处于点动运行模式下
- SD41050 $SN_JOG_CONT_MODE_LEVELTRIGGRD
  在 JOG 连续模式下点动或持续运行

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.27.14 traversKeyPlus

PLC → NC	<Chan>.geoAxis1.out.traversKeyPlus
请求几何轴“正”移动键信号（用于正轴方向运行）	

说明

根据生效的机床功能以及点按及长按模式（SD41050 和 MD11300）的设置，在信号切换时触发不同的响应：

1. 在点按模式下连续手动运行
   几何轴一直沿相应方向移动至活动的限位开关，如同接口信号具有信号状态 TRUE 的情形。
2. 在长按模式下连续手动运行
   在该信号发出首个上升沿 0 → 1 后，几何轴开始朝对应方向移动。即使之后信号输出下降沿 1 → 0，轴也继续移动。在该信号再次给出上升沿 0 → 1 后（同一个移动方向！），轴终止移动。
3. 在点按模式下断续手动运行
   信号状态为 TRUE 时，几何轴开始按照设置的增量运行。如果运行完增量前，信号切换至 FALSE 状态，这种情况下中断运行。信号状态恢复为 TRUE 后，继续运行。在没有移动完指定步数前，轴的移动可以如上文所述暂停和继续。
4. 在长按模式下断续手动运行
   在该信号输出第一个上升沿 0 → 1 后，几何轴开始移动指定的步数。如果在几何轴移动完步数前，同一个方向信号再次输出上升沿 0 → 1，则终止移动。将不会把步数移动完。

说明

- 此接口信号与以下运行方式不相关：
  - AUTOMATIC
  - MDA
- 若两个移动信号（“+”和“-”）同时置位，则不移动或终止当前移动。
- 和机床轴不同的是，在一个时间点上只能通过移动键移动一根几何轴。
- 通过几何轴移动键锁定功能，可以借助移动键将运行锁定。
- 在以下情形下，不可在 JOG 运行方式下移动几何轴：
  - 已通过轴专用接口将几何轴作为机床轴移动。
  - 已通过移动键使另一几何轴移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求借助移动键沿正轴方向移动几何轴。

值 FALSE

未请求借助移动键运行几何轴。

关联

- <Chan>.geoAxis1.out.disableTraversKey
  锁定几何轴移动键
- <Axis>.basic.out.traversKeyMinus
  在 JOG 运行方式下沿负方向移动机床轴 (MD32020 $MA_JOG_VELO)
- MD11300 $MN_JOG_INC_MODE_LEVELTRIGGRD
  INC 和 REF 处于点动运行模式下
- SD41050 $SN_JOG_CONT_MODE_LEVELTRIGGRD
  在 JOG 连续模式下点动或持续运行

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.28 <Chan>.orientationAxis1.in

12.3.28.1 contManualTraversActive

NC → PLC	<Chan>.orientationAxis1.in.contManualTraversActive
连续运行定向轴（正常运行）	

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型与更新

BOOL，循环

值 TRUE

用于手动运行定向轴的机床功能生效。

值 FALSE

用于手动运行定向轴的机床功能未生效。

关联

- <Chan>.orientationAxis1.out.inc1
  激活“根据请求将定向轴移动 1 个增量”
- <Chan>.orientationAxis1.out.inc10
  激活“根据请求将定向轴移动 10 个增量”
- <Chan>.orientationAxis1.out.inc100
  激活“根据请求将定向轴移动 100 个增量”
- <Chan>.orientationAxis1.out.inc1000
  激活“根据请求将定向轴移动 1 000 个增量”
- <Chan>.orientationAxis1.out.inc10000
  激活“根据请求将定向轴移动 10 000 个增量”
- <Chan>.orientationAxis1.out.incVar
  激活“根据请求移动定向轴，移动幅度为用户自定义增量数”
- <Chan>.orientationAxis1.out.contManualTravers
  激活定向轴连续运行（正常运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.28.2 handwheelAssignedNumber

NC → PLC	<Chan>.orientationAxis1.in.handwheelAssignedNumber
当前所分配手轮的编号：1..6 或 0 未分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）	

说明

当前所分配手轮的编号。

该接口使用了二进制编码。通过机床数据 MD11324=1 进行定义。

数据类型和更新

USInt，循环

值

0：未分配手轮

1...6：当前所分配手轮的编号

关联

- MD11324 $MN_HANDWH_VDI_REPRESENTATION
  VDI 接口中的手轮编号显示
- <Chan>.orientationAxis1.out.handwheelSelection
  将手轮 1 - 6（0 代表未选择）分配给定向轴，比特或二进制编码 (MD11324 $MN_HANDWH_VDI_REPRESENTATION)

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.28.3 handwheelDirInverted

NC → PLC	<Chan>.orientationAxis1.in.handwheelDirInverted
“反转分配给定向轴的手轮的旋转方向”激活	

数据类型与更新

BOOL

值 TRUE

手轮旋转方向反转处于活动状态

值 FALSE

手轮旋转方向反转未激活

12.3.28.4 inc1Active

NC → PLC	<Chan>.orientationAxis1.in.inc1Active
“根据请求将定向轴移动 1 个增量”处于活动状态	

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求通过 INC1 进行手动操作。

值 FALSE

针对定向轴，未请求通过 INC1 进行手动操作的机床功能。

关联

- <Chan>.orientationAxis1.out.inc1
  激活“根据请求将定向轴移动 1 个增量”
- <Chan>.orientationAxis1.out.inc10
  激活“根据请求将定向轴移动 10 个增量”
- <Chan>.orientationAxis1.out.inc100
  激活“根据请求将定向轴移动 100 个增量”
- <Chan>.orientationAxis1.out.inc1000
  激活“根据请求将定向轴移动 1000 个增量”
- <Chan>.orientationAxis1.out.inc10000
  激活“根据请求将定向轴移动 10000 个增量”
- <Chan>.orientationAxis1.out.incVar
  激活“根据请求移动定向轴，移动幅度为用户自定义增量数”
- <Chan>.orientationAxis1.out.contManualTravers
  激活定向轴连续运行（正常运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.28.5 inc10Active

NC → PLC	<Chan>.orientationAxis1.in.inc10Active
“根据请求将定向轴移动 10 个增量”处于活动状态	

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求通过 INC10 进行手动操作。

值 FALSE

针对定向轴，未请求通过 INC10 进行手动操作的机床功能。

关联

- <Chan>.orientationAxis1.out.inc1
  激活“根据请求将定向轴移动 1 个增量”
- <Chan>.orientationAxis1.out.inc10
  激活“根据请求将定向轴移动 10 个增量”
- <Chan>.orientationAxis1.out.inc100
  激活“根据请求将定向轴移动 100 个增量”
- <Chan>.orientationAxis1.out.inc1000
  激活“根据请求将定向轴移动 1000 个增量”
- <Chan>.orientationAxis1.out.inc10000
  激活“根据请求将定向轴移动 10000 个增量”
- <Chan>.orientationAxis1.out.incVar
  激活“根据请求移动定向轴，移动幅度为用户自定义增量数”
- <Chan>.orientationAxis1.out.contManualTravers
  激活定向轴连续运行（正常运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.28.6 inc100Active

NC → PLC	<Chan>.orientationAxis1.in.inc100Active
“根据请求将定向轴移动 100 个增量”处于活动状态	

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求通过 INC100 进行手动操作。

值 FALSE

针对定向轴，未请求通过 INC100 进行手动操作的机床功能。

关联

- <Chan>.orientationAxis1.out.inc1
  激活“根据请求将定向轴移动 1 个增量”
- <Chan>.orientationAxis1.out.inc10
  激活“根据请求将定向轴移动 10 个增量”
- <Chan>.orientationAxis1.out.inc100
  激活“根据请求将定向轴移动 100 个增量”
- <Chan>.orientationAxis1.out.inc1000
  激活“根据请求将定向轴移动 1000 个增量”
- <Chan>.orientationAxis1.out.inc10000
  激活“根据请求将定向轴移动 10000 个增量”
- <Chan>.orientationAxis1.out.incVar
  激活“根据请求移动定向轴，移动幅度为用户自定义增量数”
- <Chan>.orientationAxis1.out.contManualTravers
  激活定向轴连续运行（正常运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.28.7 inc1000Active

NC → PLC	<Chan>.orientationAxis1.in.inc1000Active
“根据请求将定向轴移动 1000 个增量”处于活动状态	

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求通过 INC1000 进行手动操作。

值 FALSE

针对定向轴，未请求通过 INC1000 进行手动操作的机床功能。

关联

- <Chan>.orientationAxis1.out.inc1
  激活“根据请求将定向轴移动 1 个增量”
- <Chan>.orientationAxis1.out.inc10
  激活“根据请求将定向轴移动 10 个增量”
- <Chan>.orientationAxis1.out.inc100
  激活“根据请求将定向轴移动 100 个增量”
- <Chan>.orientationAxis1.out.inc1000
  激活“根据请求将定向轴移动 1000 个增量”
- <Chan>.orientationAxis1.out.inc10000
  激活“根据请求将定向轴移动 10000 个增量”
- <Chan>.orientationAxis1.out.incVar
  激活“根据请求移动定向轴，移动幅度为用户自定义增量数”
- <Chan>.orientationAxis1.out.contManualTravers
  激活定向轴连续运行（正常运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.28.8 inc10000Active

NC → PLC	<Chan>.orientationAxis1.in.inc10000Active
“根据请求将定向轴移动 10000 个增量”处于活动状态	

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求通过 INC10000 进行手动操作。

值 FALSE

针对定向轴，未请求通过 INC10000 进行手动操作的机床功能。

关联

- <Chan>.orientationAxis1.out.inc1
  激活“根据请求将定向轴移动 1 个增量”
- <Chan>.orientationAxis1.out.inc10
  激活“根据请求将定向轴移动 10 个增量”
- <Chan>.orientationAxis1.out.inc100
  激活“根据请求将定向轴移动 100 个增量”
- <Chan>.orientationAxis1.out.inc1000
  激活“根据请求将定向轴移动 1000 个增量”
- <Chan>.orientationAxis1.out.inc10000
  激活“根据请求将定向轴移动 10000 个增量”
- <Chan>.orientationAxis1.out.incVar
  激活“根据请求移动定向轴，移动幅度为用户自定义增量数”
- <Chan>.orientationAxis1.out.contManualTravers
  激活定向轴连续运行（正常运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.28.9 incVarActive

NC → PLC	<Chan>.orientationAxis1.in.inc10000Active
“根据请求将定向轴移动 10000 个增量”处于活动状态	

说明

根据机床功能，在操作移动键或手轮时的响应各不相同。

数据类型和更新

BOOL，循环

值 TRUE

已请求用于手动运行定向轴的机床功能。

值 FALSE

未请求用于手动运行定向轴的机床功能。

关联

- <Chan>.orientationAxis1.out.inc1
  激活“根据请求将定向轴移动 1 个增量”
- <Chan>.orientationAxis1.out.inc10
  激活“根据请求将定向轴移动 10 个增量”
- <Chan>.orientationAxis1.out.inc100
  激活“根据请求将定向轴移动 100 个增量”
- <Chan>.orientationAxis1.out.inc1000
  激活“根据请求将定向轴移动 1000 个增量”
- <Chan>.orientationAxis1.out.inc10000
  激活“根据请求将定向轴移动 10000 个增量”
- <Chan>.orientationAxis1.out.incVar
  激活“根据请求移动定向轴，移动幅度为用户自定义增量数”
- <Chan>.orientationAxis1.out.contManualTravers
  激活定向轴连续运行（正常运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.28.10 traversMinusRequest

NC → PLC	<Chan>.orientationAxis1.in.traversMinusRequest
已请求定向轴沿负方向移动	

说明

视运行方式而定，以不同的方式触发运行请求：

- JOG 运行方式
  移动键“+”或“-”
- 运行方式 REF
  触发轴朝参考点运行的移动键。
- AUTO 或 MDI 运行方式
  执行一个包含定向轴运行指令的程序段。

数据类型与更新

BOOL，循环

值 TRUE

针对定向轴，存在负轴方向运行请求。

值 FALSE

对于定向轴而言不存在运行请求。

关联

- <Chan>.orientationAxis1.out.traversKeyMinus
  请求定向轴“负”移动键信号（用于负轴方向运行）
- <Chan>.orientationAxis1.in.traversMinusCommand
  定向轴负方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.28.11 traversPlusRequest

NC → PLC	<Chan>.orientationAxis1.in.traversPlusRequest
已请求定向轴沿正方向移动	

说明

视运行方式而定，以不同的方式触发运行请求：

- JOG 运行方式
  移动键“+”或“-”
- 运行方式 REF
  触发轴朝参考点运行的移动键。
- AUTO 或 MDI 运行方式
  执行一个包含定向轴运行指令的程序段。

数据类型与更新

BOOL，循环

值 TRUE

针对定向轴，存在正轴方向运行请求。

值 FALSE

对于定向轴而言不存在运行请求。

关联

- <Chan>.orientationAxis1.out.traversKeyPlus
  请求定向轴“正”移动键信号（用于正轴方向运行）
- <Chan>.orientationAxis1.in.traversPlusCommand
  定向轴正方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.28.12 traversMinusCommand

NC → PLC	<Chan>.orientationAxis1.in.traversMinusCommand
定向轴负方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)	

说明

视 MD17900, Bit 0 的设置而定，当存在“运行请求” (Bit 0 = FALSE) 或轴实际移动时 (Bit 0 = TRUE)，输出运行指令。

应用示例

一旦检测到运行指令就松开轴，使轴运行。

说明

对于在检测到运行指令时才松开夹紧装置的轴而言，不可采用连续路径运行（G64）。

数据类型与更新

BOOL，循环

值 TRUE

对于定向轴而言存在运行请求，或者定向轴移动。

值 FALSE

对于定向轴而言不存在运行请求，或者定向轴不移动。

关联

- <Chan>.orientationAxis1.out.traversKeyMinus
  请求定向轴“负”移动键信号（用于负轴方向运行）
- <Chan>.orientationAxis1.in.traversMinusRequest
  已请求定向轴沿负方向移动
- MD17900 $MN_VDI_FUNCTION_MASK
  VDI 信号设置

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.28.13 traversPlusCommand

NC → PLC	<Chan>.orientationAxis1.in.traversPlusCommand
定向轴正方向运行指令处于活动状态 (MD17900 $MN_VDI_FUNCTION_MASK)	

说明

视 MD17900, Bit 0 的设置而定，当存在“运行请求” (Bit 0 = FALSE) 或轴实际移动时 (Bit 0 = TRUE)，输出运行指令。

应用示例

一旦检测到运行指令就松开轴，使轴运行。

说明

对于在检测到运行指令时才松开夹紧装置的轴而言，不可采用连续路径运行（G64）。

数据类型与更新

BOOL，循环

值 TRUE

对于定向轴而言存在运行请求，或者定向轴移动。

值 FALSE

对于定向轴而言不存在运行请求，或者定向轴不移动。

关联

- <Chan>.orientationAxis1.out.traversKeyPlus
  请求定向轴“正”移动键信号（用于正轴方向运行）
- <Chan>.orientationAxis1.in.traversPlusRequest
  已请求定向轴沿正方向移动
- MD17900 $MN_VDI_FUNCTION_MASK
  VDI 信号设置

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.29 <Chan>.orientationAxis1.out

12.3.29.1 contManualTravers

PLC → NC	<Chan>.orientationAxis1.out.contManualTravers
激活定向轴连续运行（正常运行）	

说明

连续手动运行时，借助移动键"+"或"-"使得定向轴沿相应的方向连续移动。

说明

- 若同时置位多个请求，将无机床功能生效。
- 如果在定向轴当前正在执行某机床功能时取消该功能或者切换到另一个功能，则定向轴终止移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求用于手动运行定向轴的机床功能。

值 FALSE

未请求用于手动运行定向轴的机床功能。

关联

- <Chan>.orientationAxis1.in.inc1Active
  “根据请求将定向轴移动 1 个增量”处于活动状态
- <Chan>.orientationAxis1.in.inc10Active
  “根据请求将定向轴移动 10 个增量”处于活动状态
- <Chan>.orientationAxis1.in.inc100Active
  “根据请求将定向轴移动 100 个增量”处于活动状态
- <Chan>.orientationAxis1.in.inc1000Active
  “根据请求将定向轴移动 1 000 个增量”处于活动状态
- <Chan>.orientationAxis1.in.inc10000Active
  “根据请求将定向轴移动 10 000 个增量”处于活动状态
- <Chan>.orientationAxis1.in.incVarActive
  “根据请求移动定向轴，移动幅度为用户自定义增量数”处于活动状态
- <Chan>.orientationAxis1.in.contManualTraversActive
  连续运行定向轴（正常运行）
- MD11320 $MN_HANDWH_IMP_PER_LATCH
  手轮每个卡槽位置的脉冲数
- MD11330 $MN_JOG_INCR_SIZE_TAB
  INC/手轮运行时的增量大小
- MD31090 $MA_JOG_INCR_WEIGHT
  INC/手轮运行时的增量加权
- SD41010 $SN_JOG_VAR_INCR_SIZE
  JOG 模式下的可变增量大小

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.29.2 disableFeed

PLC → NC	<Chan>.orientationAxis1.out.disableFeed
锁定定向轴进给	

数据类型与更新

BOOL，循环

值 TRUE

激活进给停止

值 FALSE

不激活进给停止

12.3.29.3 disableTraversKey

PLC → NC	<Chan>.orientationAxis1.out.disableTraversKey
锁定定向轴移动键	

说明

在运行期间激活移动键禁用的情况下，运行会终止。

数据类型与更新

BOOL，循环

值 TRUE

针对定向轴的移动键禁用生效。

值 FALSE

针对定向轴的移动键禁用未生效。

关联

- <Chan>.orientationAxis1.out.traversKeyMinus
  请求定向轴“负”移动键信号（用于负轴方向运行）
- <Chan>.orientationAxis1.out.traversKeyPlus
  请求定向轴“正”移动键信号（用于正轴方向运行）

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.29.4 handwheelSelection

PLC → NC	<Chan>.orientationAxis1.out.handwheelSelection
手轮 1..6 已分配或 0 未分配手轮（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）	

说明

所分配手轮的编号。

该接口使用了二进制编码。通过机床数据 MD11324=1 进行定义。

数据类型和更新

USInt，循环

值

0：无手轮分配

1...6：所分配手轮的编号

关联

- MD11324 $MN_HANDWH_VDI_REPRESENTATION
  VDI 接口中的手轮编号显示
- <Chan>.orientationAxis1.in.handwheelAssignedNumber
  定向轴已分配手轮 1 - 6（0 代表未选择）的编号，比特或二进制编码 (MD11324 $MN_HANDWH_VDI_REPRESENTATION)

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.29.5 handwheelInvertDir

PLC → NC	<Chan>.orientationAxis1.out.handwheelInvertDir
请求反转分配给定向轴的手轮的旋转方向	

说明

只允许在定向轴处于静止时修改该接口信号。

数据类型和更新

BOOL，循环

值 TRUE

已请求取反对应的手轮的旋转方向。

值 FALSE

未请求取反对应的手轮的旋转方向。

关联

- <Chan>.orientationAxis1.in.handwheelDirInverted
  “反转分配给定向轴的手轮的旋转方向”激活

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.29.6 inc1

PLC → NC	<Chan>.orientationAxis1.out.inc1
激活“根据请求将定向轴移动 1 个增量”	

说明

除了五个固定步数（MD11330 中的缺省设置：INC1、INC10、INC100、INC1000 和 INC10000），还有可由设定数据 SD41010 设置的可变步数（INCvar）。固定步数和可变步数的每一步的长度通过轴专用机床数据 MD31090 设置。

通过按下移动键“+”或“-”，或通过旋转电子手轮，定向轴开始沿相应的方向移动与生效的机床功能对应的步数。

说明

- 若同时置位多个请求，将无机床功能生效。
- 如果在定向轴当前正在执行某机床功能时取消该功能或者切换到另一个功能，则定向轴终止移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求用于手动运行定向轴的机床功能。

值 FALSE

未请求用于手动运行定向轴的机床功能。

关联

- <Chan>.orientationAxis1.in.inc1Active
  “根据请求将定向轴移动 1 个增量”处于活动状态
- <Chan>.orientationAxis1.in.inc10Active
  “根据请求将定向轴移动 10 个增量”处于活动状态

- <Chan>.orientationAxis1.in.inc100Active
  “根据请求将定向轴移动 100 个增量”处于活动状态
- <Chan>.orientationAxis1.in.inc1000Active
  “根据请求将定向轴移动 1 000 个增量”处于活动状态
- <Chan>.orientationAxis1.in.inc10000Active
  “根据请求将定向轴移动 10 000 个增量”处于活动状态
- <Chan>.orientationAxis1.in.incVarActive
  “根据请求移动定向轴，移动幅度为用户自定义增量数”处于活动状态
- <Chan>.orientationAxis1.in.contManualTraversActive
  连续运行定向轴（正常运行）
- MD11320 $MN_HANDWH_IMP_PER_LATCH
  手轮每个卡槽位置的脉冲数
- MD11330 $MN_JOG_INCR_SIZE_TAB
  INC/手轮运行时的增量大小
- MD31090 $MA_JOG_INCR_WEIGHT
  INC/手轮运行时的增量加权
- SD41010 $SN_JOG_VAR_INCR_SIZE
  JOG 模式下的可变增量大小

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.29.7 inc10

PLC → NC	<Chan>.orientationAxis1.out.inc10
激活“根据请求将定向轴移动 10 个增量”	

说明

除了五个固定步数（MD11330 中的缺省设置：INC1、INC10、INC100、INC1000 和 INC10000），还有可由设定数据 SD41010 设置的可变步数（INCvar）。固定步数和可变步数的每一步的长度通过轴专用机床数据 MD31090 设置。

通过按下移动键"+"或"-"，或通过旋转电子手轮，定向轴开始沿相应的方向移动与生效的机床功能对应的步数。

说明

- 若同时置位多个请求，将无机床功能生效。
- 如果在定向轴当前正在执行某机床功能时取消该功能或者切换到另一个功能，则定向轴终止移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求用于手动运行定向轴的机床功能。

值 FALSE

未请求用于手动运行定向轴的机床功能。

关联

- <Chan>.orientationAxis1.in.inc1Active
  "根据请求将定向轴移动 1 个增量"处于活动状态
- <Chan>.orientationAxis1.in.inc10Active
  "根据请求将定向轴移动 10 个增量"处于活动状态
- <Chan>.orientationAxis1.in.inc100Active
  "根据请求将定向轴移动 100 个增量"处于活动状态
- <Chan>.orientationAxis1.in.inc1000Active
  "根据请求将定向轴移动 1 000 个增量"处于活动状态
- <Chan>.orientationAxis1.in.inc10000Active
  "根据请求将定向轴移动 10 000 个增量"处于活动状态
- <Chan>.orientationAxis1.in.incVarActive
  "根据请求移动定向轴，移动幅度为用户自定义增量数"处于活动状态
- <Chan>.orientationAxis1.in.contManualTraversActive
  连续运行定向轴（正常运行）
- MD11320 $MN_HANDWH_IMP_PER_LATCH
  手轮每个卡槽位置的脉冲数
- MD11330 $MN_JOG_INCR_SIZE_TAB
  INC/手轮运行时的增量大小

- MD31090 $MA_JOG_INCR_WEIGHT
  INC/手轮运行时的增量加权
- SD41010 $SN_JOG_VAR_INCR_SIZE
  JOG 模式下的可变增量大小

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.29.8 inc100

PLC → NC	<Chan>.orientationAxis1.out.inc100
激活“根据请求将定向轴移动 100 个增量”	

说明

除了五个固定步数（MD11330 中的缺省设置：INC1、INC10、INC100、INC1000 和 INC10000），还有可由设定数据 SD41010 设置的可变步数（INCvar）。固定步数和可变步数的每一步的长度通过轴专用机床数据 MD31090 设置。

通过按下移动键“+”或“-”，或通过旋转电子手轮，定向轴开始沿相应的方向移动与生效的机床功能对应的步数。

说明

- 若同时置位多个请求，将无机床功能生效。
- 如果在定向轴当前正在执行某机床功能时取消该功能或者切换到另一个功能，则定向轴终止移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求用于手动运行定向轴的机床功能。

值 FALSE

未请求用于手动运行定向轴的机床功能。

关联

- <Chan>.orientationAxis1.in.inc1Active
  “根据请求将定向轴移动 1 个增量”处于活动状态
- <Chan>.orientationAxis1.in.inc10Active
  “根据请求将定向轴移动 10 个增量”处于活动状态
- <Chan>.orientationAxis1.in.inc100Active
  “根据请求将定向轴移动 100 个增量”处于活动状态
- <Chan>.orientationAxis1.in.inc1000Active
  “根据请求将定向轴移动 1 000 个增量”处于活动状态
- <Chan>.orientationAxis1.in.inc10000Active
  “根据请求将定向轴移动 10 000 个增量”处于活动状态
- <Chan>.orientationAxis1.in.incVarActive
  “根据请求移动定向轴，移动幅度为用户自定义增量数”处于活动状态
- <Chan>.orientationAxis1.in.contManualTraversActive
  连续运行定向轴（正常运行）
- MD11320 $MN_HANDWH_IMP_PER_LATCH
  手轮每个卡槽位置的脉冲数
- MD11330 $MN_JOG_INCR_SIZE_TAB
  INC/手轮运行时的增量大小
- MD31090 $MA_JOG_INCR_WEIGHT
  INC/手轮运行时的增量加权
- SD41010 $SN_JOG_VAR_INCR_SIZE
  JOG 模式下的可变增量大小

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.29.9 inc1000

PLC → NC	<Chan>.orientationAxis1.out.inc1000
激活“根据请求将定向轴移动 1 000 个增量”	

说明

除了五个固定步数（MD11330 中的缺省设置：INC1、INC10、INC100、INC1000 和 INC10000），还有可由设定数据 SD41010 设置的可变步数（INCvar）。固定步数和可变步数的每一步的长度通过轴专用机床数据 MD31090 设置。

通过按下移动键"+"或"-"，或通过旋转电子手轮，定向轴开始沿相应的方向移动与生效的机床功能对应的步数。

说明

- 若同时置位多个请求，将无机床功能生效。
- 如果在定向轴当前正在执行某机床功能时取消该功能或者切换到另一个功能，则定向轴终止移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求用于手动运行定向轴的机床功能。

值 FALSE

未请求用于手动运行定向轴的机床功能。

关联

- <Chan>.orientationAxis1.in.inc1Active
  "根据请求将定向轴移动 1 个增量"处于活动状态
- <Chan>.orientationAxis1.in.inc10Active
  "根据请求将定向轴移动 10 个增量"处于活动状态
- <Chan>.orientationAxis1.in.inc100Active
  "根据请求将定向轴移动 100 个增量"处于活动状态
- <Chan>.orientationAxis1.in.inc1 000Active
  "根据请求将定向轴移动 1 000 个增量"处于活动状态
- <Chan>.orientationAxis1.in.inc10 000Active
  "根据请求将定向轴移动 10 000 个增量"处于活动状态
- <Chan>.orientationAxis1.in.incVarActive
  "根据请求移动定向轴，移动幅度为用户自定义增量数"处于活动状态
- <Chan>.orientationAxis1.in.contManualTraversActive
  连续运行定向轴（正常运行）

- MD11320 $MN_HANDWH_IMP_PER_LATCH
  手轮每个卡槽位置的脉冲数
- MD11330 $MN_JOG_INCR_SIZE_TAB
  INC/手轮运行时的增量大小
- MD31090 $MA_JOG_INCR_WEIGHT
  INC/手轮运行时的增量加权
- SD41010 $SN_JOG_VAR_INCR_SIZE
  JOG 模式下的可变增量大小

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.29.10 inc10000

PLC → NC	<Chan>.orientationAxis1.out.inc10000
激活“根据请求将定向轴移动 10 000 个增量”	

说明

除了五个固定步数（MD11330 中的缺省设置：INC1、INC10、INC100、INC1000 和 INC10000），还有可由设定数据 SD41010 设置的可变步数（INCvar）。固定步数和可变步数的每一步的长度通过轴专用机床数据 MD31090 设置。

通过按下移动键“+”或“-”，或通过旋转电子手轮，定向轴开始沿相应的方向移动与生效的机床功能对应的步数。

说明

- 若同时置位多个请求，将无机床功能生效。
- 如果在定向轴当前正在执行某机床功能时取消该功能或者切换到另一个功能，则定向轴终止移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求用于手动运行定向轴的机床功能。

值 FALSE

未请求用于手动运行定向轴的机床功能。

关联

- <Chan>.orientationAxis1.in.inc1Active
  “根据请求将定向轴移动 1 个增量”处于活动状态
- <Chan>.orientationAxis1.in.inc10Active
  “根据请求将定向轴移动 10 个增量”处于活动状态
- <Chan>.orientationAxis1.in.inc100Active
  “根据请求将定向轴移动 100 个增量”处于活动状态
- <Chan>.orientationAxis1.in.inc1000Active
  “根据请求将定向轴移动 1 000 个增量”处于活动状态
- <Chan>.orientationAxis1.in.inc10000Active
  “根据请求将定向轴移动 10 000 个增量”处于活动状态
- <Chan>.orientationAxis1.in.incVarActive
  “根据请求移动定向轴，移动幅度为用户自定义增量数”处于活动状态
- <Chan>.orientationAxis1.in.contManualTraversActive
  连续运行定向轴（正常运行）
- MD11320 $MN_HANDWH_IMP_PER_LATCH
  手轮每个卡槽位置的脉冲数
- MD11330 $MN_JOG_INCR_SIZE_TAB
  INC/手轮运行时的增量大小
- MD31090 $MA_JOG_INCR_WEIGHT
  INC/手轮运行时的增量加权
- SD41010 $SN_JOG_VAR_INCR_SIZE
  JOG 模式下的可变增量大小

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.29.11 incVar

PLC → NC	<Chan>.orientationAxis1.out.incVar
激活“根据请求移动定向轴，移动幅度为用户自定义增量数”	

说明

除了五个固定步数（MD11330 中的缺省设置：INC1、INC10、INC100、INC1000 和 INC10000），还有可由设定数据 SD41010 设置的可变步数（INCvar）。固定步数和可变步数的每一步的长度通过轴专用机床数据 MD31090 设置。

通过按下移动键"+"或"-"，或通过旋转电子手轮，定向轴开始沿相应的方向移动与生效的机床功能对应的步数。

说明

- 若同时置位多个请求，将无机床功能生效。
- 如果在定向轴当前正在执行某机床功能时取消该功能或者切换到另一个功能，则定向轴终止移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求用于手动运行定向轴的机床功能。

值 FALSE

未请求用于手动运行定向轴的机床功能。

关联

- <Chan>.orientationAxis1.in.inc1Active
  "根据请求将定向轴移动 1 个增量"处于活动状态
- <Chan>.orientationAxis1.in.inc10Active
  "根据请求将定向轴移动 10 个增量"处于活动状态
- <Chan>.orientationAxis1.in.inc100Active
  "根据请求将定向轴移动 100 个增量"处于活动状态
- <Chan>.orientationAxis1.in.inc1000Active
  "根据请求将定向轴移动 1 000 个增量"处于活动状态
- <Chan>.orientationAxis1.in.inc10000Active
  "根据请求将定向轴移动 10 000 个增量"处于活动状态
- <Chan>.orientationAxis1.in.incVarActive
  "根据请求移动定向轴，移动幅度为用户自定义增量数"处于活动状态
- <Chan>.orientationAxis1.in.contManualTraversActive
  连续运行定向轴（正常运行）

- MD11320 $MN_HANDWH_IMP_PER_LATCH
  手轮每个卡槽位置的脉冲数
- MD11330 $MN_JOG_INCR_SIZE_TAB
  INC/手轮运行时的增量大小
- MD31090 $MA_JOG_INCR_WEIGHT
  INC/手轮运行时的增量加权
- SD41010 $SN_JOG_VAR_INCR_SIZE
  JOG 模式下的可变增量大小

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.29.12 rapidTraversOvr

PLC → NC	<Chan>.orientationAxis1.out.rapidTraversOvr
激活定向轴快速运行速度 (MD32010)	

说明

在 JOG 运行方式下连续或断续手动运行时，此信号才生效。

快进速度可通过快进倍率开关调整。

数据类型与更新

BOOL，循环

值 TRUE

在借助移动键手动运行定向轴时，此接口信号的置位使快进速度（MD32010）生效。

值 FALSE

在借助移动键手动运行定向轴时，设定的 JOG 速度（SD41110 或 MD32020）生效。

关联

- <Chan>.orientationAxis1.out.traversKeyMinus
  请求定向轴“负”移动键信号（用于负轴方向运行）
- <Chan>.orientationAxis1.out.traversKeyPlus
  请求定向轴“正”移动键信号（用于正轴方向运行）
- MD32010 $MA_JOG_VELO_RAPID
  常规快速运行

- MD32020 $MA_JOG_VELO
  常规轴速度
- SD41110 $SN_JOG_SET_VELO
  JOG 模式下的轴速度

更多信息

- 轴和主轴功能手册；“进给”章节
- 轴和主轴功能手册；“手动运行”章节

12.3.29.13 traversKeyMinus

PLC → NC	<Chan>.orientationAxis1.out.traversKeyMinus
请求定向轴“负”移动键信号（用于负轴方向运行）	

说明

根据生效的机床功能以及点按及长按模式（SD41050 和 MD11300）的设置，在信号切换时触发不同的响应：

1. 在点按模式下连续手动运行
   定向轴一直沿相应方向移动至活动的限位开关，如同接口信号具有信号状态 TRUE 的情形。
2. 在长按模式下连续手动运行
   在首次脉冲沿切换 0 → 1 时，定向轴开始沿相应的方向移动。即使之后信号输出下降沿 1 → 0，轴也继续移动。在该信号再次给出上升沿 0 → 1 后（同一个移动方向！），轴终止移动。
3. 在点按模式下断续手动运行
   信号状态为 TRUE 时，定向轴开始按照设置的增量运行。如果运行完增量前，信号切换至 FALSE 状态，这种情况下中断运行。信号状态恢复为 TRUE 后，继续运行。在没有移动完指定步数前，定向轴的移动可以如上文所述暂停和继续。
4. 在长按模式下断续手动运行
   在首次脉冲沿切换 0 → 1 时，定向轴开始移动指定的步数。如果在定向轴移动完步数前，同一个方向信号再次输出上升沿 0 → 1，则终止移动。将不会把步数移动完。

说明

- 此接口信号与以下运行方式不相关：
  - AUTOMATIC
  - MDA
- 若两个移动信号（"+"和"-"）同时置位，则不移动或终止当前移动。
- 和机床轴不同的是，在一个时间点上只能通过移动键移动一根定向轴。
- 通过 <Chan>.orientationAxis1.out.disableTraversKey，可以借助移动键锁定运行。
- 在以下情形下，不可在 JOG 运行方式下移动定向轴：
  - 已通过轴专用接口将定向轴作为机床轴移动。
  - 已通过移动键使另一定向轴移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求借助移动键沿负轴方向移动机床轴。

值 FALSE

未请求借助移动键运行定向轴。

关联

- <Chan>.orientationAxis1.out.disableTraversKey
  锁定定向轴移动键
- <Axis>.basic.out.traversKeyMinus
  在 JOG 运行方式下沿负方向移动机床轴 (MD32020 $MA_JOG_VELO)
- MD11300 $MN_JOG_INC_MODE_LEVELTRIGGRD
  INC 和 REF 处于点动运行模式下
- SD41050 $SN_JOG_CONT_MODE_LEVELTRIGGRD
  在 JOG 连续模式下点动或持续运行

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.29.14 traversKeyPlus

PLC → NC	<Chan>.orientationAxis1.out.traversKeyPlus
请求定向轴“正”移动键信号（用于正轴方向运行）	

说明

根据生效的机床功能以及点按及长按模式（SD41050 和 MD11300）的设置，在信号切换时触发不同的响应：

1. 在点按模式下连续手动运行
   定向轴一直沿相应方向移动至活动的限位开关，如同接口信号具有信号状态 TRUE 的情形。
2. 在长按模式下连续手动运行
   在首次脉冲沿切换 0 → 1 时，定向轴开始沿相应的方向移动。即使之后信号输出下降沿 1 → 0，轴也继续移动。在该信号再次给出上升沿 0 → 1 后（同一个移动方向！），轴终止移动。
3. 在点按模式下断续手动运行
   信号状态为 TRUE 时，定向轴开始按照设置的增量运行。如果运行完增量前，信号切换至 FALSE 状态，这种情况下中断运行。信号状态恢复为 TRUE 后，继续运行。在没有移动完指定步数前，定向轴的移动可以如上文所述暂停和继续。
4. 在长按模式下断续手动运行
   在首次脉冲沿切换 0 → 1 时，定向轴开始移动指定的步数。如果在定向轴移动完步数前，同一个方向信号再次输出上升沿 0 → 1，则终止移动。将不会把步数移动完。

说明

- 此接口信号与以下运行方式不相关：
  - AUTOMATIC
  - MDA
- 若两个移动信号（“+”和“-”）同时置位，则不移动或终止当前移动。
- 和机床轴不同的是，在一个时间点上只能通过移动键移动一根定向轴。
- 通过 <Chan>.orientationAxis1.out.disableTraversKey，可以借助移动键锁定运行。
- 在以下情形下，不可在 JOG 运行方式下移动定向轴：
  - 已通过轴专用接口将定向轴作为机床轴移动。
  - 已通过移动键使另一定向轴移动。

数据类型与更新

BOOL，循环

值 TRUE

已请求借助移动键沿正轴方向移动定向轴。

值 FALSE

未请求借助移动键运行定向轴。

关联

- <Chan>.orientationAxis1.out.disableTraversKey
  锁定定向轴移动键
- <Axis>.basic.out.traversKeyPlus
  在 JOG 运行方式下沿正方向移动机床轴 (MD32020 $MA_JOG_VELO)

- MD11300 $MN_JOG_INC_MODE_LEVELTRIGGRD
  INC 和 REF 处于点动运行模式下
- SD41050 $SN_JOG_CONT_MODE_LEVELTRIGGRD
  在 JOG 连续模式下点动或持续运行

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.30 <Chan>.toolMgmt.in

12.3.30.1 lastReplacementToolReached

NC → PLC	<Chan>.toolMgmt.in.lastReplacementToolReached
已到达刀具组的最后一个备用刀具 (MD20310)。刀具组最后一个备用刀具的刀具号位于：lastReplacementToolTNumber	

说明

刀具组中最后一个备用刀具的刀具号位于：<Chan>.toolMgmt.in.lastTNumber

更改信号的持续时间为一个“Main”循环。

说明

必须在 MD20310 $MC_TOOL_MANAGEMENT_MASK，位 18 中激活对最后的备用刀具的监控。

数据类型与更新

BOOL，任务控制

值 TRUE

达到刀具组中的最后一把备用刀具

值 FALSE

不相关

关联

- <Chan>.toolMgmt.in.toolPrewarnLimitReached
  达到刀具预警极限值。已达到预警极限值的刀具，其刀具号位于：
  toolPrewarnLimitTNumber
- <Chan>.toolMgmt.in.toolLimitValueReached
  达到刀具极限值。已达到极限值的刀具，其刀具号位于：toolLimitTNumber
- <Chan>.toolMgmt.in.newReplacementToolActive
  完成向新备用刀具的过渡。新备用刀具的刀具号位于：newReplacementToolTNumber
- <Chan>.toolMgmt.in.lastTNumber
  最后一个备用刀具的刀具号

更多信息

- 功能手册之刀具管理

12.3.30.2 lastTNumber

NC → PLC	<Chan>.toolMgmt.in.lastTNumber
最后一个备用刀具的刀具号	

数据类型与更新

DINT，循环

值

已传输的刀具管理函数：最后一个备用刀具的刀具号

12.3.30.3 newReplacementToolActive

NC → PLC	<Chan>.toolMgmt.in.newReplacementToolActive
完成向新备用刀具的过渡。新备用刀具的刀具号位于：newReplacementToolTNumber	

说明

新备用刀具的刀具号位于：<Chan>.toolMgmt.in.newTNumber

更改信号的持续时间为一个“Main”循环。

数据类型与更新

BOOL，任务控制

值 TRUE

过渡至新的备用刀具。

值 FALSE

不相关

关联

- <Chan>.toolMgmt.in.toolPrewarnLimitReached
  达到刀具预警极限值。已达到预警极限值的刀具，其刀具号位于：toolPrewarnLimitTNumber
- <Chan>.toolMgmt.in.toolLimitValueReached
  达到刀具极限值。已达到极限值的刀具，其刀具号位于：toolLimitTNumber
- <Chan>.toolMgmt.in.lastReplacementToolReached
  已到达刀具组的最后一个备用刀具 (MD20310).刀具组最后一个备用刀具的刀具号位于：lastReplacementToolTNumber
- <Chan>.toolMgmt.in.newTNumber
  新备用刀具的刀具号

更多信息

- 功能手册之刀具管理

12.3.30.4 newTNumber

NC → PLC	<Chan>.toolMgmt.in.newTNumber
新备用刀具的刀具号	

数据类型与更新

DINT，循环

值

已传输的刀具管理函数：新备用刀具的刀具号

12.3.30.5 toolLimitTNumber

NC → PLC	<Chan>.toolMgmt.in.toolLimitTNumber
刀具限制刀具号	

数据类型与更新

DINT，循环

值

已传输的刀具管理函数：已达到极限值的刀具的刀具号

12.3.30.6 toolLimitValueReached

NC → PLC	<Chan>.toolMgmt.in.toolLimitValueReached
达到刀具极限值。已达到极限值的刀具，其刀具号位于：toolLimitTNumber	

说明

达到极限值的刀具的刀具号位于：<Chan>.toolMgmt.in.toolLimitTNumber

更改信号的持续时间为一个“Main”循环。

数据类型与更新

BOOL，任务控制

值 TRUE

达到刀具极限值。

值 FALSE

不相关

关联

- <Chan>.toolMgmt.in.toolPrewarnLimitReached
  达到刀具预警极限值。已达到预警极限值的刀具，其刀具号位于：toolPrewarnLimitTNumber
- <Chan>.toolMgmt.in.newReplacementToolActive
  完成向新备用刀具的过渡。新备用刀具的刀具号位于：newReplacementToolTNumber
- <Chan>.toolMgmt.in.lastReplacementToolReached
  已到达刀具组的最后一个备用刀具 (MD20310).刀具组最后一个备用刀具的刀具号位于：lastReplacementToolTNumber
- <Chan>.toolMgmt.in.toolLimitTNumber
  刀具限制刀具号

更多信息

- 功能手册之刀具管理

12.3.30.7 toolMissing

NC → PLC	<Chan>.toolMgmt.in.toolMissing
编程的刀具缺失或者无法使用	

说明

刀具管理：缺少刀具

刀具不存在或者无法使用。

数据类型与更新

BOOL，任务控制

值 TRUE

编写的刀具缺失。

值 FALSE

不相关

12.3.30.8 toolPrewarnLimitReached

NC → PLC	<Chan>.toolMgmt.in.toolPrewarnLimitReached
达到刀具预警极限值。已达到预警极限值的刀具，其刀具号位于：toolPrewarnLimitTNumber	

说明

达到预警极限的刀具的刀具号位于：<Chan>.toolMgmt.in.toolPrewarnLimitTNumber

更改信号的持续时间为一个 "Main" 循环。

数据类型与更新

BOOL，任务控制

值 TRUE

达到刀具预警极限

值 FALSE

不相关

关联

- <Chan>.toolMgmt.in.toolLimitValueReached
  达到刀具极限值。已达到极限值的刀具，其刀具号位于：toolLimitTNumber
- <Chan>.toolMgmt.in.newReplacementToolActive
  完成向新备用刀具的过渡。新备用刀具的刀具号位于：newReplacementToolTNumber
- <Chan>.toolMgmt.in.lastReplacementToolReached
  已到达刀具组的最后一个备用刀具 (MD20310)。刀具组最后一个备用刀具的刀具号位于：lastReplacementToolTNumber
- <Chan>.toolMgmt.in.toolPrewarnLimitTNumber
  刀具预警极限刀具号

更多信息

- 功能手册之刀具管理

12.3.30.9 toolPrewarnLimitTNumber

NC → PLC	<Chan>.toolMgmt.in.toolPrewarnLimitTNumber
刀具预警极限刀具号	

数据类型与更新

DINT

值

已传输的刀具管理函数：已达到预警极限的刀具的刀具号

12.3.31 <Chan>.toolMgmt.out

12.3.31.1 deactivateWearMonitoring

PLC → NC	<Chan>.toolMgmt.out.deactivateWorkpieceCounter
刀具管理：已请求禁用工件计数器监控	

数据类型与更新

BOOL，任务控制

值 TRUE

已请求关闭磨损监控。

值 FALSE

未请求关闭磨损监控。

关联

- $TC_MOP15
  磨损设定值或总补偿设定值
- $TC_MOP5
  磨损预警极限或总补偿预警极限
- $TC_MOP6
  磨损实际值或总补偿实际值
- $TC_TP9
  刀具监控类型
- MD18080 $MN_MM_TOOL_MANAGEMENT_MASK
  刀具管理的存储器预留
- MD20310 TOOL_MANAGEMENT_MASK, Bit 17
  激活刀具管理功能

更多信息

- 功能手册之刀具管理；
  - 章节“功能说明”>“刀具监控（件数、使用寿命、磨损）”>“磨损监控”
  - 章节“功能说明”>“无刀具管理的刀具监控”>“磨损监控”

12.3.31.2 deactivateWorkpieceCounter

PLC → NC	<Chan>.toolMgmt.out.deactivateWorkpieceCounter
刀具管理：已请求禁用工件计数器监控	

说明

工件计数监控实现与过程、工件材料或其他影响相关的工件计数。可通过此接口信号关闭工件计数监控。

数据类型与更新

BOOL，任务控制

值 TRUE

已请求关闭工件计数监控。

值 FALSE

未请求关闭工件计数监控。

关联

- SETPIECE
  功能
- TMPCIT
  程序实例服务
- $TC_MOP4
  剩余件数
- $TC_TP9
  刀具监控类型

更多信息

- 功能手册之刀具管理；
  - 章节“功能说明”>“刀具监控（件数、使用寿命、磨损）”>“工件计数监控”
  - 章节“功能说明”>“无刀具管理的刀具监控”>“工件计数监控”

12.3.31.3 ignoreDisabledSate

PLC → NC	<Chan>.toolMgmt.out.ignoreDisabledSate
刀具管理：刀具锁定未生效	

说明

在生效监控功能（件数、使用寿命或磨损）的实际值达到零值的情况下，刀具进入“禁用”状态。若刀具此时仍处于加工中，则将其留在加工中，直至下一次换刀。之后该刀具不再可使用。

通过此接口信号可设置：在搜索可使用刀具时，NC 不将“禁用”状态考虑在内。

在通过初始化程序段选择刀具的情况下（Reset- 和 Start_Mode_Mask），此接口信号不生效。

数据类型与更新

BOOL，任务控制

值 TRUE

刀具锁定无效

值 FALSE

刀具锁定生效

关联

- MD22562 $MC_TOOL_CHANGE_ERROR_MODE
  换刀过程中出错时的行为

更多信息

- 功能手册之刀具管理；章节“功能说明”>“刀具监控（件数、使用寿命、磨损）”>“磨损监控”>“发送至 PLC 和来自 PLC 的信号”

12.3.31.4 toolLifeMonitoring

PLC → NC	<Chan>.toolMgmt.out.toolLifeMonitoring
采集刀具使用寿命数据（刀具管理）(MD20310 Bit 17)	

说明

在不以快进（G0）运行几何轴的情况下，总是默认监测使用寿命。

借助此接口信号取消及重新激活对使用寿命的监测。

前提条件

通过 MD20310 TOOL_MANAGEMENT_MASK, Bit 17 = TRUE 将功能使能

数据类型与更新

BOOL，循环

值 TRUE

监测使用寿命。

值 FALSE

不监测使用寿命。

关联

- $A_MONIFACT
  读取使用寿命监控的系数
- $TC_MOP1
  使用寿命预警极限
- MD18080 $MN_MM_TOOL_MANAGEMENT_MASK
  刀具管理的存储器预留
- MD20124 $MC_TOOL_MANAGEMENT_TOOLHOLDER
  刀夹编号
- MD20310 TOOL_MANAGEMENT_MASK, Bit 17
  激活刀具管理功能
- MD20320 $MC_TOOL_TIME_MONITOR_MASK
  针对刀夹中刀具的时间监控

更多信息

- 功能手册之刀具管理；
  – 章节“功能说明”>“刀具监控（件数、使用寿命、磨损）”>“使用寿命监控”
  – 章节“功能说明”>“无刀具管理的刀具监控”>“使用寿命监控”

12.3.32 <Chan>.auxMDynFuncs.in

12.3.32.1 m00Decoded - m99Decoded

NC → PLC	<Chan>.auxMDynFuncs.in.m00..99Decoded
脉冲沿切换 0 → 1：M 功能 00 - 99 解码	

说明

动态 M 功能 M0 至 M99 解码

说明

- 如在通道中编程设置一个主轴，此处不显示 M 功能。这种情况下 M 功能作为高级 M 功能显示在 <Chan>.auxMFuncs[1].in.extendedAddress 下，同时轴向显示在 <Axis>.spindle.in.mFuncNumber 下。
- **动态** M 功能（M00 至 M99）由 PLC 基本程序解码。
  静态 M 功能必须在 PLC 用户程序中由动态 M 功能构成。

数据类型和更新

BOOL，任务控制

值 TRUE

对应的 M 功能已被解码。

值 FALSE

对应的 M 功能未被解码，或者接口信号被 PLC 基本程序复位以作应答。

针对常规辅助功能输出，此接口信号由 PLC 基本程序在一个完整的 "Main" 循环后复位。

针对快速辅助功能输出，此接口信号由 PLC 基本程序在同一个 "Hardware interrupt" 循环中复位。

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.33 <Chan>.auxMFuncs[x].in

12.3.33.1 extendedAddress

NC → PLC	<Chan>.auxMFuncs[1].in.extendedAddress
M 功能扩展地址	

说明

如果存在一个 M 功能更改信号 (<Chan>.auxMFuncs[x].in.hasChanged)，这里会提供最多五个可在 NC 程序段中编程设置，具有扩展地址的 M 功能。

数据类型与更新

INT，任务控制

值

扩展地址

关联

- <Chan>.auxMFuncs[x].in.hasChanged
  M 功能发生变化；信号显示相应的值有效，可用于一个 PLC 节拍
- MD10715 $MN_M_NO_FCT_CYCLE
- MD10716 $MN_M_NO_FCT_CYCLE_NAME

- MD10718 $MN_M_NO_FCT_CYCLE_PAR
- $C_M...
  系统变量

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.33.2 fastAckActive

NC → PLC	<Chan>.auxMFuncs[1].in.fastAckActive
M 功能快速应答处于活动状态	

数据类型和更新

BOOL，任务控制

值 TRUE

在 "Main" 循环开头，一条包含新值的 M 信息已与相应的更改信号一起输出至接口。

值 FALSE

更改信号在下一个 "Main" 循环开头由 PLC 基本程序复位。更改的值不再有效。

12.3.33.3 hasChanged

NC → PLC	<Chan>.auxMFuncs[1].in.hasChanged
M 功能发生变化；信号显示相应的值有效，可用于一个 PLC 节拍	

数据类型与更新

BOOL，任务控制

值 TRUE

在 "Main" 循环开头，一条包含新值的 M 信息已与相应的更改信号一起输出至接口。此时更改信号表示相应的值生效。

值 FALSE

更改信号在下一个 "Main" 循环开头由 PLC 基本程序复位。更改的值不再有效。

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.33.4 notDecoded

NC → PLC	<Chan>.auxMFuncs[1].in.notDecoded
M 功能未包含在解码列表中，可用于一个 PLC 节拍	

数据类型与更新

BOOL，任务控制

值 TRUE

M 功能大于 99（扩展地址 = 0）或扩展地址 > 0 时不包含在解码列表中。

此信号与相应的 M 更改信号一起，存在于一个 "Main" 循环中。

原因：

- 编写了错误的 M 功能
- PLC 解码列表中的 M 功能未配置

解决方法例如为：

- PLC 设置读取禁止
- 输出 PLC 报警

值 FALSE

M 功能小于 99（扩展地址 = 0）或扩展地址 > 0 时包含在解码列表中。

12.3.33.5 value

NC → PLC	<Chan>.auxMFuncs[1].in.value
M 功能值	

说明

如果存在一个 M 功能更改信号 (<Chan>.auxMFuncs[x].in.hasChanged)，这里会提供最多五个可在 NC 程序段中编程设置，具有 M 功能编号的 M 功能。

M 功能值一直保持，直至被新的 M 功能覆盖。

以下事件会导致 M 功能值被清除：

- PLC 启动。
- 输入新的 M 功能。

说明

零件程序中，M 功能以 INTEGER 的格式编程（8 位十进制加符号）。

数据类型与更新

DINT，任务控制

值

M 功能值

关联

- <Chan>.auxMFuncs[x].in.hasChanged
  M 功能发生变化；信号显示相应的值有效，可用于一个 PLC 节拍
- MD10715 $MN_M_NO_FCT_CYCLE
- MD10716 $MN_M_NO_FCT_CYCLE_NAME
- MD10718 $MN_M_NO_FCT_CYCLE_PAR
- $C_M...
  系统变量

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.34 <Chan>.auxSFuncs[x].in

12.3.34.1 extendedAddress

NC → PLC	<Chan>.auxSFuncs[1].in.extendedAddress
S 功能扩展地址	

说明

如果存在一个 S 功能更改信号 (<Chan>.auxSFuncs[x].in.hasChanged)，这里会提供最多三个可在 NC 程序段中编程设置，具有扩展地址的 S 功能。

数据类型与更新

INT，任务控制

值

扩展地址

关联

- <Chan>.auxSFuncs[x].in.hasChanged
  S 功能发生变化；信号显示相应的值有效

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.34.2 fastAckActive

NC → PLC	<Chan>.auxSFuncs[1].in.fastAckActive
S 功能快速应答处于活动状态	

说明

附加信息“Quick”（快速应答）处于活动状态

数据类型和更新

BOOL，任务控制

值 TRUE

在 "Main" 循环开头，一条包含新值的 S 信息已与相应的更改信号一起输出至接口。

值 FALSE

更改信号在下一个 "Main" 循环开头由 PLC 基本程序复位。更改的值不再有效。

12.3.34.3 hasChanged

NC → PLC	<Chan>.auxSFuncs[1].in.hasChanged
S 功能发生变化；信号显示相应的值有效	

数据类型与更新

BOOL，任务控制

值 TRUE

在 "Main" 循环开头，一条包含新值的 S 信息已与相应的更改信号一起输出至接口。此时更改信号表示相应的值生效。

值 FALSE

更改信号在下一个 "Main" 循环开头由 PLC 基本程序复位。更改的值不再有效。

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.34.4 value

NC → PLC	<Chan>.auxSFuncs[1].in.value
S 功能值	

说明

如果存在一个 S 功能更改信号 (<Chan>.auxSFuncs[x].in.hasChanged)，这里会提供最多三个可在 NC 程序段中编程设置，具有 S 功能编号的 S 功能。

S 功能值一直保持，直至被新的 S 功能覆盖。

以下事件会导致 S 功能值被清除：

- PLC 启动。
- 输入新的 S 功能。

数据类型与更新

REAL，任务控制

值

S 功能值

关联

- <Chan>.auxSFuncs[x].in.hasChanged
  S 功能发生变化；信号显示相应的值有效

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.35 <Chan>.auxTFuncs[x].in

12.3.35.1 extendedAddress

NC → PLC	<Chan>.auxTFuncs[1].in.extendedAddress
T 功能扩展地址	

说明

- 已编程的 T 功能在刀具管理激活时不能输出至 PLC。
- 8 位十进制 T 编号只能在 <Chan>.auxTFuncs[x].in.value“T 功能 1”下使用。
- 数据格式 REAL 符合 STEP 7 中的浮点表示（24 位尾数和 8 位指数）。此浮点格式最多可有 7 个有效位。

数据类型与更新

INT

值

扩展地址

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.35.2 fastAckActive

NC → PLC	<Chan>.auxTFuncs[1].in.fastAckActive
T 功能快速应答处于活动状态	

说明

- 10 位十进制 T 编号只能使用 <Chan>.auxTFuncs[1].in.hasChanged“T 功能 1 更改”信号。

数据类型和更新

BOOL，任务控制

值 TRUE

在 "Main" 循环开头，一条包含新值的 T 信息已与相应的更改信号一起输出至接口。

值 FALSE

更改信号在下一个 "Main" 循环开头由 PLC 基本程序复位。更改的值不再有效。

12.3.35.3 hasChanged

NC → PLC	<Chan>.auxTFuncs[1].in.hasChanged
T 功能发生变化；信号显示相应的值有效	

数据类型与更新

BOOL，任务控制

值 TRUE

在 "Main" 循环开头，一条包含新值的 T 信息已与相应的更改信号一起输出至接口。此时更改信号表示相应的值生效。

值 FALSE

更改信号在下一个 "Main" 循环开头由 PLC 基本程序复位。更改的值不再有效。

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.35.4 value

NC → PLC	<Chan>.auxTFuncs[1].in.value
T 功能值	

说明

如果存在一个 T 功能更改信号，这里会提供在 NC 程序段中编程设置的 T 功能。

T 功能值一直保持，直至被新的 T 功能覆盖。

以下事件会导致 T 功能值被清除：

- PLC 启动。
- 输入新的 T 功能。

说明

- 已编程的 T 功能在刀具管理激活时不能输出至 PLC。
- 8 位十进制 T 编号只能在 <Chan>.auxTFuncs[x].in.value“T 功能 1”下使用。
- 数据格式 REAL 符合 STEP 7 中的浮点表示（24 位尾数和 8 位指数）。此浮点格式最多可有 7 个有效位。

数据类型与更新

INT，任务控制

值

T 功能值

关联

- <Chan>.auxTFuncs[x].in.hasChanged
  T 功能发生变化；信号显示相应的值有效
- MD10717 $MN_T_NO_FCT_CYCLE_NAME
  用于替换 T 功能的换刀循环的名称
- MD10719 $MN_T_NO_FCT_CYCLE_MODE
  T 功能替换的编程设置
- MD22220 $MC_AUXFU_T_SYNC_TYPE
  T 功能的输出时间点
- MD22550 $MC_TOOL_CHANGE_MODE
  M 功能下的新刀具补偿
- $C_T...
  系统变量

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.36 <Chan>.auxDFuncs[x].in

12.3.36.1 extendedAddress

NC → PLC	<Chan>.auxDFuncs[1].in.extendedAddress
D 功能扩展地址	

说明

- 已编程的带有名称的 D 功能（例如：D=SCHNEIDE_1）不能以 ASCII 格式输出至 PLC。
- 5 位十进制 D 编号只能作为 <Chan>.auxDFuncs[x].in.extendedAddress"D 功能 1"下使用。
- 数据格式 REAL 符合 STEP 7 中的浮点表示（24 位尾数和 8 位指数）。此浮点格式最多可有 7 个有效位。

数据类型与更新

BYTE

值

扩展地址

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.36.2 fastAckActive

NC → PLC	<Chan>.auxDFuncs[1].in.fastAckActive
D 功能快速应答处于活动状态	

说明

- 5 位刀沿号只能使用 <Chan>.auxDFuncs[1].in.hasChanged"D 功能 1 更改”信号。

数据类型和更新

BOOL，任务控制

值 TRUE

在 "Main" 循环开头，一条包含新值的 D 信息已与相应的更改信号一起输出至接口。

值 FALSE

更改信号在下一个 "Main" 循环开头由 PLC 基本程序复位。更改的值不再有效。

12.3.36.3 hasChanged

NC → PLC	<Chan>.auxDFuncs[1].in.hasChanged
D 功能发生变化；信号显示相应的值有效	

数据类型与更新

BOOL，任务控制

值 TRUE

在 "Main" 循环开头，一条包含新值的 D 信息已与相应的更改信号一起输出至接口。此时更改信号表示相应的值生效。

值 FALSE

更改信号在下一个 "Main" 循环开头由 PLC 基本程序复位。更改的值不再有效。

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.36.4 value

NC → PLC	<Chan>.auxDFuncs[1].in.value
D 功能值	

说明

如果存在一个 D 功能更改信号，这里会提供在 NC 程序段中编程设置的 D 功能。

D 功能值一直保持，直至被新的 D 功能覆盖。

以下事件会导致 D 功能值被清除：

- PLC 启动。
- 输入新的 D 功能。

说明

- 已编程的带有名称的 D 功能（例如：D=SCHNEIDE_1）不能以 ASCII 格式输出至 PLC。
- 5 位十进制 D 编号只能作为 <Chan>.auxDFuncs[x].in.extendedAddress“D 功能 1”下使用。
- 数据格式 REAL 符合 STEP 7 中的浮点表示（24 位尾数和 8 位指数）。此浮点格式最多可有 7 个有效位。

数据类型与更新

BYTE，任务控制

值

D 功能值

关联

- <Chan>.auxDFuncs[x].in.hasChanged
  D 功能发生变化；信号显示相应的值有效
- MD22250 $MC_AUXFU_D_SYNC_TYPE
  D 功能的输出时间点
- MD22252 $MC_AUXFU_DL_SYNC_TYPE
  DL 功能的输出时间点

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.37 <Chan>.auxDLFuncs[x].in

12.3.37.1 extendedAddress

NC → PLC	<Chan>.auxDLFuncs[1].in.extendedAddress
扩展的 DL 功能地址	

数据类型与更新

INT

值

扩展地址

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.37.2 fastAckActive

NC → PLC	<Chan>.auxDLFuncs[1].in.fastAckActive
DL 功能快速应答处于活动状态	

数据类型和更新

BOOL，任务控制

值 TRUE

在 "Main" 循环开头，一条包含新值的 DL 信息已与相应的更改信号一起输出至接口。

值 FALSE

更改信号在下一个 "Main" 循环开头由 PLC 基本程序复位。更改的值不再有效。

12.3.37.3 hasChanged

NC → PLC	<Chan>.auxDLFuncs[1].in.hasChanged
DL 功能发生变化；信号显示相应的值有效	

数据类型与更新

BOOL，任务控制

值 TRUE

在 "Main" 循环开头，一条包含新值的 DL 信息已与相应的更改信号一起输出至接口。此时更改信号表示相应的值生效。

值 FALSE

更改信号在下一个 "Main" 循环开头由 PLC 基本程序复位。更改的值不再有效。

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.37.4 value

NC → PLC	<Chan>.auxDLFuncs[1].in.value
DL 功能值	

说明

- 数据格式 REAL 符合 STEP 7 中的浮点表示（24 位尾数和 8 位指数）。此浮点格式最多可有 7 个有效位。

数据类型与更新

REAL，任务控制

值

DL 功能值

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.38 <Chan>.auxHFuncs[x].in

12.3.38.1 extendedAddress

NC → PLC	<Chan>.auxHFuncs[1].in.extendedAddress
H 功能扩展地址	

说明

如果存在一个 H 功能更改信号，这里会提供最多三个可在 NC 程序段中编程设置，具有扩展地址的 H 功能。

数据类型与更新

INT，任务控制

值

扩展地址

关联

- <Chan>.auxHFuncs[x].in.hasChanged
  H 功能发生变化；信号显示相应的值有效
- MD22110 $MC_AUXFU_H_TYPE_INT
  H 辅助功能的类型为整数
- MD22230 $MC_AUXFU_H_SYNC_TYPE
  H 功能的输出时间点

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.38.2 fastAckActive

NC → PLC	<Chan>.auxHFuncs[1].in.fastAckActive
H 功能快速应答处于活动状态	

数据类型和更新

BOOL，任务控制

值 TRUE

在 "Main" 循环开头，一条包含新值的 H 信息已与相应的更改信号一起输出至接口。

值 FALSE

更改信号在下一个 "Main" 循环开头由 PLC 基本程序复位。更改的值不再有效。

12.3.38.3 hasChanged

NC → PLC	<Chan>.auxHFuncs[1].in.hasChanged
H 功能发生变化；信号显示相应的值有效	

数据类型与更新

BOOL，任务控制

值 TRUE

在 "Main" 循环开头，一条包含新值的 H 信息已与相应的更改信号一起输出至接口。此时更改信号表示相应的值生效。

值 FALSE

更改信号在下一个 "Main" 循环开头由 PLC 基本程序复位。更改的值不再有效。

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.38.4 value

NC → PLC	<Chan>.auxHFuncs[1].in.value
H 功能值（类型 REAL，MD22110 $MC_AUXFU_H_TYPE_INT=0）	

说明

如果存在一个 H 功能更改信号，这里会提供最多三个可在 NC 程序段中编程设置，具有 H 功能编号的 H 功能。

H 功能值一直保持，直至被新的 H 功能覆盖。

以下事件会导致 H 功能值被清除：

- PLC 启动。
- 输入新的 H 功能。

数据类型与更新

REAL，任务控制

值

H 功能值为浮点数

关联

- <Chan>.auxHFuncs[x].in.hasChanged
  H 功能发生变化；信号显示相应的值有效
- MD22110 $MC_AUXFU_H_TYPE_INT
  H 辅助功能的类型为整数
- MD22230 $MC_AUXFU_H_SYNC_TYPE
  H 功能的输出时间点

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.38.5 valueTypeDint

NC → PLC	<Chan>.auxHFuncs[1].in.valueTypeDint
H 功能值（类型 DINT，MD22110 $MC_AUXFU_H_TYPE_INT=1）	

说明

如果存在一个 H 功能更改信号，这里会提供最多三个可在 NC 程序段中编程设置，具有 H 功能编号的 H 功能。

H 功能值一直保持，直至被新的 H 功能覆盖。

以下事件会导致 H 功能值被清除：

- PLC 启动。
- 输入新的 H 功能。

数据类型与更新

DINT，任务控制

值

H 功能值为定点数

关联

- <Chan>.auxHFuncs[x].in.hasChanged
  H 功能发生变化；信号显示相应的值有效
- MD22110 $MC_AUXFU_H_TYPE_INT
  H 辅助功能的类型为整数
- MD22230 $MC_AUXFU_H_SYNC_TYPE
  H 功能的输出时间点

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.39 <Chan>.auxFFuncs[x].in

12.3.39.1 extendedAddress

NC → PLC	<Chan>.auxFFuncs[1].in.extendedAddress
F 功能扩展地址	

说明

如果存在一个 F 功能更改信号 (<Chan>.auxFFuncs[x].in.hasChanged)，这里会提供最多五个可在 NC 程序段中编程设置，具有扩展地址的 F 功能。

如果存在一个 F 功能更改信号，这里会提供最多六个可在 NC 程序段中编程设置，具有扩展地址的 F 功能。

F 功能的扩展地址由进给方式（轨迹进给率或轴专用进给率）和轴名称构成。

- 轨迹进给率：0
- 轴专用进给率：定位轴（1, 2, 3, ... 最大机床轴数）的机床轴编号

说明

- 零件程序中，F 功能以 REAL 数据格式编程。
- F 功能的扩展地址包括一个标识，标识含义如下：
  - 0：轨迹进给率
  - 1 - 31：定位轴上用于进给的机床轴号
- H 功能的数据类型取决于机床数据：MD22110 $MC_AUXFU_H_TYPE_INT

数据类型与更新

INT，任务控制

值

扩展地址

关联

- <Chan>.auxFFuncs[x].in.hasChanged
  F 功能发生变化；信号显示相应的值有效
- MD22240 $MC_AUXFU_F_SYNC_TYPE
  F 功能的输出时间

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.39.2 fastAckActive

NC → PLC	<Chan>.auxFFuncs[1].in.fastAckActive
F 功能快速应答处于活动状态	

数据类型和更新

BOOL，任务控制

值 TRUE

在 "Main" 循环开头，一条包含新值的 F 信息已与相应的更改信号一起输出至接口。

值 FALSE

更改信号在下一个 "Main" 循环开头由 PLC 基本程序复位。更改的值不再有效。

12.3.39.3 hasChanged

NC → PLC	<Chan>.auxFFuncs[1].in.hasChanged
F 功能发生变化；信号显示相应的值有效	

数据类型与更新

BOOL，任务控制

值 TRUE

在 "Main" 循环开头，一条包含新值的 F 信息已与相应的更改信号一起输出至接口。此时更改信号表示相应的值生效。

值 FALSE

更改信号在下一个 "Main" 循环开头由 PLC 基本程序复位。更改的值不再有效。

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.39.4 value

NC → PLC	<Chan>.auxFFuncs[1].in.value
F 功能值	

说明

如果存在一个 F 功能更改信号，这里会提供最多三个可在 NC 程序段中编程设置，具有 F 功能编号的 F 功能。

F 功能值一直保持，直至被新的 F 功能覆盖。

以下事件会导致 F 功能值被清除：

- PLC 启动。
- 输入新的 F 功能。

说明

- 零件程序中，F 功能以 REAL 数据格式编程。
- F 功能的扩展地址包括一个标识，标识含义如下：
  - 0：轨迹进给率
  - 1 - 31：定位轴上用于进给的机床轴号
- H 功能的数据类型取决于机床数据：MD22110 $MC_AUXFU_H_TYPE_INT

数据类型与更新

DINT，任务控制

值

F 功能值

关联

- <Chan>.auxFFuncs[x].in.hasChanged
  F 功能发生变化；信号显示相应的值有效
- MD22240 $MC_AUXFU_F_SYNC_TYPE
  F 功能的输出时间

更多信息

- 基础功能手册；“辅助功能输出至 PLC”章节

12.3.40 <Chan>.protAreas[x].in

12.3.40.1 chanPreactivated

NC → PLC	<Chan>.protAreas[1].in.chanPreactivated
通道专用保护区域预激活	

说明

仅经预激活的通道专用保护区才能通过相应的接口信号激活：
<Chan>.protAreas[x].out.chan（激活通道专用保护区域 1 至 10）

数据类型和更新

BOOL，循环

值 TRUE

通道相关保护区域 1 (...10) 预激活。在 NC 程序中通过指令 NPROT 进行预激活。

值 FALSE

通道相关保护区域 1 (...10) 未预激活。

关联

- <Chan>.basic.out.enableProtAreas
  保护区域使能（不是防撞保护 CA 的构成部分）
- <Chan>.protAreas[x].out.nc
  激活机床专用保护区域
- <Chan>.protAreas[x].out.chan
  激活通道专用保护区域
- <Chan>.protAreas[x].in.ncPreactivated
  机床专用保护区域预激活
- <Chan>.protAreas[x].in.ncViolated
  机床专用保护区域受到干扰
- <Chan>.protAreas[x].in.chanViolated
  通道专用保护区域受到干扰

更多信息

- 基础功能手册；“监控和补偿”>“保护区域”章节

12.3.40.2 chanViolated

NC → PLC	<Chan>.protAreas[1].in.chanViolated
通道专用保护区域受到干扰	

数据类型与更新

BOOL，循环

值 TRUE

在当前程序段或当前 JOG 运行中，激活的通道相关保护区域 1 - 10 受到干扰。

如果预激活的通道相关保护区域 1 - 10 处于活动状态，会在当前程序段中受到干扰。

值 FALSE

在当前程序段或当前 JOG 运行中，激活的通道相关保护区域 1 - 10 不受干扰。

如果预激活的通道相关保护区域 1 - 10 处于活动状态，在当前程序段中不受干扰。

关联

- <Chan>.basic.out.enableProtAreas
  保护区域使能（不是防撞保护 CA 的构成部分）
- <Chan>.protAreas[x].out.nc
  激活机床专用保护区域
- <Chan>.protAreas[x].out.chan
  激活通道专用保护区域
- <Chan>.protAreas[x].in.chanPreactivated
  通道专用保护区域预激活
- <Chan>.protAreas[x].in.ncPreactivated
  机床专用保护区域预激活
- <Chan>.protAreas[1].in.chanPreactivated
  通道专用保护区域预激活
- <Chan>.protAreas[1].in.ncViolated
  机床专用保护区域受到干扰

更多信息

- 基础功能手册；“监控和补偿”>“保护区域”章节

12.3.40.3 ncPreactivated

NC → PLC	<Chan>.protAreas[1].in.ncPreactivated
机床专用保护区域预激活	

说明

仅经预激活的机床相关保护区才能通过相应的接口信号激活或取消激活：
<Chan>.protAreas[x].out.nc（激活机床相关保护区域 1 至 10）

数据类型与更新

BOOL，循环

值 TRUE

机床相关保护区域 1 (...10) 预激活。
在 NC 程序中通过指令 NPROT 进行预激活。

值 FALSE

机床相关保护区域 1 (...10) 未预激活。

关联

- <Chan>.basic.out.enableProtAreas
  保护区域使能（不是防撞保护 CA 的构成部分）
- <Chan>.protAreas[x].out.nc
  激活机床专用保护区域
- <Chan>.protAreas[x].out.chan
  激活通道专用保护区域
- <Chan>.protAreas[x].in.chanPreactivated
  通道专用保护区域预激活
- <Chan>.protAreas[x].in.ncViolated
  机床专用保护区域受到干扰
- <Chan>.protAreas[x].in.chanViolated
  通道专用保护区域受到干扰

更多信息

- 基础功能手册；“监控和补偿”>“保护区域”章节

12.3.40.4 ncViolated

NC → PLC	<Chan>.protAreas[1].in.ncViolated
机床专用保护区域受到干扰	

数据类型与更新

BOOL，循环

值 TRUE

在当前程序段或当前 JOG 运行中，激活的机床相关保护区域 1 - 10 受到干扰。

如果预激活的机床相关保护区域 1 - 10 处于活动状态，会在当前程序段中受到干扰。

值 FALSE

在当前程序段或当前 JOG 运行中，激活的机床相关保护区域 1 - 10 不受干扰。

如果预激活的机床相关保护区域 1 - 10 处于活动状态，在当前程序段中不受干扰。

关联

- <Chan>.basic.out.enableProtAreas
  保护区域使能（不是防撞保护 CA 的构成部分）
- <Chan>.protAreas[x].out.nc
  激活机床专用保护区域
- <Chan>.protAreas[x].out.chan
  激活通道专用保护区域
- <Chan>.protAreas[x].in.chanPreactivated
  通道专用保护区域预激活
- <Chan>.protAreas[x].in.ncPreactivated
  机床专用保护区域预激活
- <Chan>.protAreas[x].in.chanViolated
  通道专用保护区域受到干扰

更多信息

- 基础功能手册；“监控和补偿”>“保护区域”章节

12.3.41 <Chan>.protAreas[x].out

12.3.41.1 chan

PLC → NC	<Chan>.protAreas[1].out.chan
激活通道专用的保护区域	

数据类型与更新

BOOL，循环

值 TRUE

请求激活通道专用保护区。

保护区立即生效。

说明

仅在已通过 NC 程序进行预激活的情况下，保护区的激活才生效。

值 FALSE

请求取消生效的通道专用保护区。

保护区立即失效。

说明

仅在通过 NC 程序预激活并且通过接口信号激活的情况下，保护区的取消才生效。

关联

- <Chan>.basic.out.enableProtAreas
  保护区域使能（不是防撞保护 CA 的构成部分）
- <Chan>.protAreas[x].out.nc
  激活机床专用保护区域
- <Chan>.protAreas[x].out.chan
  激活通道专用保护区域
- <Chan>.protAreas[x].in.ncPreactivated
  机床专用保护区域预激活
- <Chan>.protAreas[x].in.chanPreactivated
  通道专用保护区域预激活

- <Chan>.protAreas[x].in.ncViolated
  机床专用保护区域受到干扰
- <Chan>.protAreas[x].in.chanViolated
  通道专用保护区域受到干扰

更多信息

- 基础功能手册；“监控和补偿”>“保护区域”章节

12.3.41.2 nc

PLC → NC	<Chan>.protAreas[1].out.nc
激活机床专用保护区域	

数据类型与更新

BOOL，循环

值 TRUE

请求激活机床相关保护区。

保护区立即生效。

说明

仅在已通过 NC 程序进行预激活的情况下，保护区的激活才生效。

值 FALSE

请求取消生效的机床相关保护区。

保护区立即失效。

说明

仅在通过 NC 程序预激活并且通过接口信号激活的情况下，保护区的取消才生效。

关联

- <Chan>.basic.out.enableProtAreas
  保护区域使能（不是防撞保护 CA 的构成部分）
- <Chan>.protAreas[x].out.nc
  激活机床专用保护区域

- <Chan>.protAreas[x].out.chan
  激活通道专用保护区域
- <Chan>.protAreas[x].in.ncPreactivated
  机床专用保护区域预激活
- <Chan>.protAreas[x].in.chanPreactivated
  通道专用保护区域预激活
- <Chan>.protAreas[x].in.ncViolated
  机床专用保护区域受到干扰
- <Chan>.protAreas[x].in.chanViolated
  通道专用保护区域受到干扰

更多信息

- 基础功能手册；“监控和补偿”>“保护区域”章节

12.3.42 <Chan>.progEvent.in

12.3.42.1 blockSearchActive

NC → PLC	<Chan>.progEvent.in.blockSearchActive
PROG_EVENT：第 1 次 NC 启动	

说明

接口信号至少在一个 PLC 循环内出现。

数据类型和更新

BOOL，循环

值 TRUE

通过搜索运行后的第 1 次 NC 启动触发 PROG_EVENT

值 FALSE

不通过搜索运行后的第 1 次 NC 启动触发 PROG_EVENT

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.42.2 ncStartActive

NC → PLC	<Chan>.progEvent.in.ncStartActive
PROG_EVENT：从“复位”通道状态 NC 启动	

说明

接口信号至少在一个 PLC 循环内出现。

数据类型和更新

BOOL，循环

值 TRUE

通过从“复位”通道状态 NC 启动触发 PROG_EVENT

值 FALSE

不通过从“复位”通道状态 NC 启动触发 PROG_EVENT

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.42.3 partProgramEndActive

NC → PLC	<Chan>.progEvent.in.partProgramEndActive
PROG_EVENT：程序结束复位（比如 M30）	

说明

接口信号至少在一个 PLC 循环内出现。

数据类型和更新

BOOL，循环

值 TRUE

通过程序结束复位（比如 M30）触发 PROG_EVENT

值 FALSE

不通过程序结束复位（比如 M30）触发 PROG_EVENT

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.42.4 powerOnActive

NC → PLC	<Chan>.progEvent.in.powerOnActive
PROG_EVENT：上电	

说明

接口信号至少在一个 PLC 循环内出现。

数据类型和更新

BOOL，循环

值 TRUE

通过暖启动（通电）触发 PROG_EVENT

值 FALSE

不通过暖启动（通电）触发 PROG_EVENT

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.42.5 resetActive

NC → PLC	<Chan>.progEvent.in.resetActive
PROG_EVENT：通道复位	

说明

接口信号至少在一个 PLC 循环内出现。

数据类型和更新

BOOL，循环

值 TRUE

通过通道复位触发 PROG_EVENT

值 FALSE

不通过通道复位触发 PROG_EVENT

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.43 <Chan>.synActs[x].in

12.3.43.1 lockable

NC → PLC	<Chan>.synActs[1].in.lockable
同步运行动作 ID/IDS 可锁定，不通过 NCK 进行保护 (MD11500 $MN_PREVENT_SYNACT_LOCK; MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN)	

说明

可以禁用同步动作识别码

数据类型与更新

BOOL

值 TRUE

可以禁用同步动作识别码，其不受 NCK 保护。

值 FALSE

无作用

关联

- MD11500 $MN_PREVENT_SYNACT_LOCK
- MD21240 $MN_PREVENT_SYNACT_LOCK_CHAN

12.3.44 <Chan>.synActs[x].out

12.3.44.1 lock

PLC → NC	<Chan>.synActs[1].out.lock
锁定同步动作 ID/IDS	

说明

锁定同步动作 ID/IDS

数据类型和更新

BOOL

值 TRUE

锁定同步动作 ID/IDS

值 FALSE

同步动作 ID/IDS 未锁定

更多信息

- 功能手册：基础功能；章节“同步”

12.3.45 <Chan>.grinding.in

12.3.45.1 requested

NC → PLC	<Chan>.grinding.in.requested
已通过 NCK 初始化动作请求磨削信号使能 ($AC_IN_KEY_G_RUN_OUT[])	

数据类型与更新

BOOL

值 TRUE

已请求通过一个 NCK 动作使能输入信号。

值 FALSE

无作用

关联

- $AC_IN_KEY_G_RUN_OUT[]

12.3.45.2 signalsEnabled

NC → PLC	<Chan>.grinding.in.signalsEnabled
磨削工艺：已使能 PLC 和 NCK 的输入信号 ($AC_IN_KEY_G_ISENABLE[])	

数据类型和更新

BOOL

值 TRUE

输入信号已通过 PLC 和 NCK 使能

值 FALSE

无作用

关联：

- $AC_IN_KEY_G_ISENABLE[]

12.3.46 <Chan>.grinding.out

12.3.46.1 ackFuncsExecution

PLC → NC	<Chan>.grinding.out.ackFuncsExecution
确认已执行磨削功能 ($AC_IN_KEY_G_RUN_IN[])	

数据类型和更新

BOOL

值 TRUE

确认已执行磨削功能

值 FALSE

未确认已执行磨削功能

关联：

- $AC_IN_KEY_G_RUN_IN[]

12.3.46.2 disableSignals

PLC → NC	<Chan>.grinding.out.disableSignals
禁用磨削信号	

数据类型和更新

BOOL

值 TRUE

禁用输入信号

值 FALSE

不禁用输入信号

12.3.46.3 set

PLC → NC	<Chan>.grinding.out.set
PLC 输入信号，如已使能 ($AC_IN_KEY_G[])	

数据类型与更新

BOOL

值 TRUE

设置输入信号

值 FALSE

不设置输入信号

关联

- $AC_IN_KEY_G[]

12.3.47 <Chan>.nibbelingAndPunching.in

12.3.47.1 strokeManualInitiationAck

NC → PLC	<Chan>.nibbelingAndPunching.in.strokeManualInitiationAck
已触发手动冲程	

数据类型与更新

BOOL，循环

值 TRUE

冲程已手动释放。

值 FALSE

冲程未手动释放。

关联

- <Chan>.nibbelingAndPunching.out.enableStroke
  冲程执行使能
- <Chan>.nibbelingAndPunching.out.manualStrokeInterface1
  在接口 1 执行手动冲程
- <Chan>.nibbelingAndPunching.out.manualStrokeInterface2
  在接口 2 执行手动冲程
- <Chan>.nibbelingAndPunching.in.strokeInitiationActive
  冲程触发处于活动状态

更多信息

- 功能手册之工艺

12.3.47.2 strokeInitiationActive

NC → PLC	<Chan>.nibbelingAndPunching.in.strokeInitiationActive
冲程触发处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

冲程释放生效。

值 FALSE

冲程释放未生效。

关联

- <Chan>.nibbelingAndPunching.out.enableStroke
  冲程执行使能

更多信息

- 功能手册之工艺

12.3.48 <Chan>.nibbelingAndPunching.out

12.3.48.1 delayStroke

PLC → NC	<Chan>.nibbelingAndPunching.out.delayStroke
激活“延迟的冲程”选项 (MD26018 $MC_NIBPUNCH_PRE_START_TIME; SD42402 $SC_NIB_PRE_START_TIME)	

说明

通过此信号能够激活经延迟的冲程。该信号和程序指令 PDELAYON 的功能类似。

数据类型与更新

BOOL

值 TRUE

激活选件“冲程延时”。

值 FALSE

无对激活选件“冲程延时”的请求。

关联

- MD26018
- <Chan>.nibbelingAndPunching.out.enableStroke
  冲程执行使能

更多信息

- 功能手册之工艺

12.3.48.2 enableStroke

PLC → NC	<Chan>.nibbelingAndPunching.out.enableStroke
冲程执行使能	

说明

允许由 PLC 使能冲程。若此信号未置位，则不允许 NC 触发冲程。NC 等待冲程使能，之后才继续零件程序。

数据类型与更新

BOOL

值 TRUE

冲程的执行被使能。

值 FALSE

冲程的执行被禁止。

更多信息

- 功能手册之工艺

12.3.48.3 manualStrokeInterface1

PLC → NC	<Chan>.nibbelingAndPunching.out.manualStrokeInterface1
在接口 1 执行手动冲程	

说明

借助此信号能够在手动模式中在第 1 冲裁接口上触发单独一个冲程。原则上在任何运行方式下均可进行此操作，前提是轴不运动。若轴在手动冲程释放期间发生运动，则忽略信号。在冲程的持续时间内，轴被禁用，即在信号“冲程释放有效”取消后，轴才能重新运动。

数据类型与更新

BOOL

值 TRUE

在第 1 冲裁接口上执行手动冲程。

值 FALSE

无在第 1 冲裁接口上执行手动冲程的请求。

关联

- <Chan>.nibbelingAndPunching.out.enableStroke
  冲程执行使能
- <Chan>.nibbelingAndPunching.out.manualStrokeInterface2
  在接口 2 执行手动冲程
- <Chan>.nibbelingAndPunching.in.strokeInitiationActive
  冲程触发处于活动状态
- <Chan>.nibbelingAndPunching.in.strokeManualInitiationAck
  已触发手动冲程

更多信息

- 功能手册之工艺

12.3.48.4 manualStrokeInterface2

PLC → NC	<Chan>.nibbelingAndPunching.out.manualStrokeInterface2
在接口 2 执行手动冲程	

说明

借助此信号能够在手动模式中在第 2 冲裁接口上触发单独一个冲程。原则上在任何运行方式下均可进行此操作，前提是轴不运动。若轴在手动冲程释放期间发生运动，则忽略信号。在冲程的持续时间内，轴被禁用，即在信号“冲程释放有效”取消后，轴才能重新运动。

数据类型与更新

BOOL

值 TRUE

在第 2 冲裁接口上执行手动冲程。

值 FALSE

无在第 2 冲裁接口上执行手动冲程的请求。

关联

- <Chan>.nibbelingAndPunching.out.enableStroke
  冲程执行使能
- <Chan>.nibbelingAndPunching.out.manualStrokeInterface1
  在接口 1 执行手动冲程
- <Chan>.nibbelingAndPunching.in.strokeInitiationActive
  冲程触发处于活动状态
- <Chan>.nibbelingAndPunching.in.strokeManualInitiationAck
  已触发手动冲程

更多信息

- 功能手册之工艺

12.3.48.5 noStroke

PLC → NC	<Chan>.nibbelingAndPunching.out.noStroke
冲程未生效	

说明

NC 立即停止运动来响应该接口信号。如果该信号会中断某一运动或者其他动作，系统会报警。

对于 NC，该接口信号就物理而言与快速信号“冲程生效”相同，即系统可以如此布线，使得通过“与”链接将两个信号引入到同一个 NC 输入端。

数据类型与更新

BOOL

值 TRUE

冲程未运行。

值 FALSE

冲程运行。

关联

- <Chan>.nibbelingAndPunching.out.enableStroke
  冲程执行使能

更多信息

- 功能手册之工艺

12.3.48.6 suppressStroke

PLC → NC	<Chan>.nibbelingAndPunching.out.suppressStroke
激活冲程抑制	

说明

此接口信号抑制快速信号“冲程释放”。其实现不释放冲程的零件程序执行（空运行）。在行程划分生效时，轴以“Stop and Go”方式运行。

数据类型与更新

BOOL

值 TRUE

激活冲程抑制。

值 FALSE

无对激活冲程抑制的请求。

关联

- <Chan>.nibbelingAndPunching.out.enableStroke
  冲程执行使能

更多信息

- 功能手册之工艺

12.3.49 <Chan>.cycles.in

12.3.49.1 signals

NC → PLC	<Chan>.cycles.in.signals
循环输入信号	

数据类型与更新

BOOL

值 TRUE

循环输入信号处于活动状态

值 FALSE

循环输入信号未激活

12.3.50 <Chan>.cycles.out

12.3.50.1 signals

PLC → NC	<Chan>.cycles.out.signals
循环输出信号	

数据类型与更新

BOOL

值 TRUE

循环输出信号处于活动状态

值 FALSE

循环输出信号未激活

12.3.51 NCK_typeModeGroup

下表主要介绍数据类型 NCK_typeModeGroup：

结构		说明
.config		运行方式组的组态数据
	.registered	运行方式组已注册
	.modeGroupActive	运行方式组处于活动状态
	.modeGroupNumber	运行方式组编号
.basic.in (页 875)		来自运行方式组的一般信号
.basic.out (页 891)		发送至运行方式组的一般信号

12.3.52 <ModeGroup>.basic.in

12.3.52.1 autoModeActive

NC → PLC	<ModeGroup>.basic.in.autoModeActive
自动运行方式处于活动状态	

说明

将该运行方式组内的运行方式改为 AUTOMATIC 模式。

数据类型和更新

BOOL，循环

值 TRUE

AUTO 运行方式生效。

值 FALSE

AUTO 运行方式未生效。

关联

- <ModeGroup>.basic.out.autoMode
  请求切换至自动运行方式
- <ModeGroup>.basic.out.disableModeChangeover
  锁定该运行方式组内的运行方式切换

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.2 contTraversActive

NC → PLC	<ModeGroup>.basic.in.contTraversActive
连续运行（默认）处于活动状态	

说明

接口必须显式使能。

<Nc>.basic.out.selectIncFromModeGroup := TRUE

数据类型与更新

BOOL，循环

值 TRUE

连续运行处于活动状态

值 FALSE

连续运行未激活

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.3 inc1Active

NC → PLC	<ModeGroup>.basic.in.inc1Active
“根据请求移动 1 个增量”处于活动状态	

说明

机床功能 INC1（根据请求移动 1 个增量）已激活

说明

接口的功能性必须显式使能。
<Nc>.basic.out.selectIncFromModeGroup := TRUE
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，循环

值 TRUE

已请求机床功能 INC1。

值 FALSE

未请求机床功能 INC1。

关联

- <ModeGroup>.basic.in.inc10Active
  “根据请求移动 10 个增量”处于活动状态
- <ModeGroup>.basic.in.inc100Active
  “根据请求移动 100 个增量”处于活动状态
- <ModeGroup>.basic.in.inc1000Active
  “根据请求移动 1000 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10000Active
  “根据请求移动 10000 个增量”处于活动状态
- <ModeGroup>.basic.in.incVarActive
  “根据请求移动用户自定义增量数”处于活动状态
- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.contTravers
  激活连续运行（默认）
- <ModeGroup>.basic.in.contTraversActive
  连续运行（默认）已激活

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.4 inc10Active

NC → PLC	<ModeGroup>.basic.in.inc10Active
“根据请求移动 10 个增量”处于活动状态	

说明

机床功能 INC10（根据请求移动 10 个增量）已激活

说明

接口的功能性必须显式使能。
<Nc>.basic.out.selectIncFromModeGroup := TRUE
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，循环

值 TRUE

已请求机床功能 INC10。

值 FALSE

未请求机床功能 INC10。

关联

- <ModeGroup>.basic.in.inc1Active
  “根据请求移动 1 个增量”处于活动状态
- <ModeGroup>.basic.in.inc100Active
  “根据请求移动 100 个增量”处于活动状态
- <ModeGroup>.basic.in.inc1000Active
  “根据请求移动 1000 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10000Active
  “根据请求移动 10000 个增量”处于活动状态
- <ModeGroup>.basic.in.incVarActive
  “根据请求移动用户自定义增量数”处于活动状态
- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”

- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.contTravers
  激活连续运行（默认）
- <ModeGroup>.basic.in.contTraversActive
  连续运行（默认）已激活

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.5 inc100Active

NC → PLC	<ModeGroup>.basic.in.inc100Active
“根据请求移动 100 个增量”处于活动状态	

说明

机床功能 INC100（根据请求移动 100 个增量）已激活

说明

接口的功能性必须显式使能。
<Nc>.basic.out.selectIncFromModeGroup := TRUE
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，循环

值 TRUE

已请求机床功能 INC100。

值 FALSE

未请求机床功能 INC100。

关联

- <ModeGroup>.basic.in.inc1Active
  “根据请求移动 1 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10Active
  “根据请求移动 10 个增量”处于活动状态

- <ModeGroup>.basic.in.inc1000Active
  “根据请求移动 1000 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10000Active
  “根据请求移动 10000 个增量”处于活动状态
- <ModeGroup>.basic.in.incVarActive
  “根据请求移动用户自定义增量数”处于活动状态
- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.contTravers
  激活连续运行（默认）
- <ModeGroup>.basic.in.contTraversActive
  连续运行（默认）已激活

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.6 inc1000Active

NC → PLC	<ModeGroup>.basic.in.inc1000Active
“根据请求移动 1000 个增量”处于活动状态	

说明

机床功能 INC1000（根据请求移动 1000 个增量）已激活

说明

接口的功能性必须显式使能。

<Nc>.basic.out.selectIncFromModeGroup := TRUE

可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，循环

值 TRUE

已请求机床功能 INC1000。

值 FALSE

未请求机床功能 INC1000。

关联

- <ModeGroup>.basic.in.inc1Active
  “根据请求移动 1 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10Active
  “根据请求移动 10 个增量”处于活动状态
- <ModeGroup>.basic.in.inc100Active
  “根据请求移动 100 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10000Active
  “根据请求移动 10000 个增量”处于活动状态
- <ModeGroup>.basic.in.incVarActive
  “根据请求移动用户自定义增量数”处于活动状态
- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”

- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.contTravers
  激活连续运行（默认）
- <ModeGroup>.basic.in.contTraversActive
  连续运行（默认）已激活

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.7 inc10000Active

NC → PLC	<ModeGroup>.basic.in.inc10000Active
“根据请求移动 10000 个增量”处于活动状态	

说明

机床功能 INC10000（根据请求移动 10000 个增量）已激活

说明

接口的功能性必须显式使能。
<Nc>.basic.out.selectIncFromModeGroup := TRUE
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，循环

值 TRUE

已请求机床功能 INC10000。

值 FALSE

未请求机床功能 INC10000。

关联

- <ModeGroup>.basic.in.inc1Active
  “根据请求移动 1 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10Active
  “根据请求移动 10 个增量”处于活动状态

- <ModeGroup>.basic.in.inc100Active
  “根据请求移动 100 个增量”处于活动状态
- <ModeGroup>.basic.in.inc1000Active
  “根据请求移动 1000 个增量”处于活动状态
- <ModeGroup>.basic.in.incVarActive
  “根据请求移动用户自定义增量数”处于活动状态
- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.contTravers
  激活连续运行（默认）
- <ModeGroup>.basic.in.contTraversActive
  连续运行（默认）已激活

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.8 incVarActive

NC → PLC	<ModeGroup>.basic.in.incVarActive
“根据请求移动用户自定义增量数”处于活动状态	

说明

机床功能 INC10（根据请求以用户自定义的数量进行移动）已激活

说明

接口的功能性必须显式使能。
<Nc>.basic.out.selectIncFromModeGroup := TRUE
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，循环

值 TRUE

已请求机床功能 INCVar。

值 FALSE

未请求机床功能 INCVar。

关联

- <ModeGroup>.basic.in.inc1Active
  “根据请求移动 1 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10Active
  “根据请求移动 10 个增量”处于活动状态
- <ModeGroup>.basic.in.inc100Active
  “根据请求移动 100 个增量”处于活动状态
- <ModeGroup>.basic.in.inc1000Active
  “根据请求移动 1000 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10000Active
  “根据请求移动 10000 个增量”处于活动状态
- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”

- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.contTravers
  激活连续运行（默认）
- <ModeGroup>.basic.in.contTraversActive
  连续运行（默认）已激活

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.9 jogModeActive

NC → PLC	<ModeGroup>.basic.in.jogModeActive
JOG 运行方式生效	

说明

将该运行方式组内的运行方式改为 JOG 模式。

数据类型和更新

BOOL，循环

值 TRUE

JOG 运行方式生效。

值 FALSE

JOG 运行方式未生效。

关联

- <ModeGroup>.basic.out.jogMode
  请求切换至 JOG 运行方式
- <ModeGroup>.basic.out.disableModeChangeover
  锁定该运行方式组内的运行方式切换

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.10 mdaModeActive

NC → PLC	<ModeGroup>.basic.in.mdaModeActive
MDA 运行方式处于活动状态	

说明

将该运行方式组内的运行方式改为 MDA 模式。

数据类型和更新

BOOL，循环

值 TRUE

MDI 运行方式生效。

值 FALSE

MDI 运行方式未生效。

关联

- <ModeGroup>.basic.out.mdaMode
  请求切换至 MDA 运行方式
- <ModeGroup>.basic.out.disableModeChangeover
  锁定该运行方式组内的运行方式切换

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.11 nckInternalJogActive

NC → PLC	<ModeGroup>.basic.in.nckInternalJogActive
NCK 内部运行方式 JOG 处于活动状态	

说明

此 NC/PLC 接口显示，“JOG in AUTOMATIC”模式下 NC 已于内部切换至“内部 JOG”模式。

数据类型与更新

BOOL，循环

值 TRUE

NC 内部 JOG 生效

值 FALSE

NC 内部 JOG 未生效

关联

- MD10735 $MN_JOG_MODE_MASK

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.12 ready

NC → PLC	<ModeGroup>.basic.in.ready
运行方式组准备就绪	

说明

信号切换 0 → 1 时的响应：

- 以设定值 0 或最大制动电流将运行的轴和主轴制动至静止状态。
- 从 PLC 到 NC 的接口信号被置入未生效状态（初始设置）。

说明

如果出现会取消“运行方式组就绪”状态的报警，切换运行方式前必须先触发运行方式组复位 <ModeGroup>.basic.out.reset。

数据类型与更新

BOOL，循环

值 TRUE

运行方式组运行就绪。

值 FALSE

运行方式组未运行就绪。

关联

- <ModeGroup>.basic.out.reset
  请求复位运行方式组

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.13 refActive

NC → PLC	<ModeGroup>.basic.in.refActive
机床功能 REF 处于活动状态	

说明

在该运行方式组中，机床功能 REF（参考点）已激活。

数据类型和更新

BOOL，循环

值 TRUE

机床功能 REF 生效。

值 FALSE

机床功能 REF 未生效。

关联

- <ModeGroup>.basic.out.refPoint
  请求机床功能 REF

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.14 reposActive

NC → PLC	<ModeGroup>.basic.in.reposActive
机床功能 REPOS 已激活	

说明

在该运行方式组中，机床功能 REPOS 已激活。

数据类型和更新

BOOL，循环

值 TRUE

机床功能 REPOS 生效。

值 FALSE

机床功能 REPOS 未生效。

关联

- <ModeGroup>.basic.out.repos
  请求机床功能 REPOS

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.15 resetAllChanDone

NC → PLC	<ModeGroup>.basic.in.resetAllChanDone
运行方式组内的所有通道都处于复位状态	

说明

该信号表示属于该运行方式组的所有通道均已经进行了复位，并且处于“复位”状态。

数据类型和更新

BOOL，循环

值 TRUE

运行方式组的所有通道均处于“复位”状态下。

值 FALSE

运行方式组的至少一个通道不处于“复位”状态下。

关联

- <Chan>.basic.out.reset
  请求复位运行方式组

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.16 resetDone

NC → PLC	<ModeGroup>.basic.in.resetDone
已执行运行方式组复位	

说明

此 NC/PLC 接口显示，“JOG in AUTOMATIC”模式下运行方式组处于“运行方式组复位”状态。

数据类型与更新

BOOL，循环

值 TRUE

运行方式组已复位并处于复位状态下。

值 FALSE

运行方式组未复位。

关联

- <ModeGroup>.basic.out.reset
  请求复位运行方式组

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.52.17 teachInActive

NC → PLC	<ModeGroup>.basic.in.teachInActive
机床功能 TEACH IN 处于活动状态	

说明

在该运行方式组中，机床功能 TEACH IN 已激活。

数据类型和更新

BOOL，循环

值 TRUE

机床功能 TEACH IN 生效。

值 FALSE

机床功能 TEACH IN 未生效。

关联

- <ModeGroup>.basic.out.teachIn
  请求机床功能 TEACH IN

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53 <ModeGroup>.basic.out

12.3.53.1 autoMode

PLC → NC	<ModeGroup>.basic.out.autoMode
请求切换至自动运行方式	

数据类型和更新

BOOL，循环

值 TRUE

存在切换至 AUTO 运行方式的请求。

值 FALSE

不存在切换至 AUTO 运行方式的请求。

关联

- <ModeGroup>.basic.out.mdaMode
  请求切换至 MDA 运行方式
- <ModeGroup>.basic.out.jogMode
  请求切换至 JOG 运行方式
- <ModeGroup>.basic.out.disableModeChangeover
  切换运行方式锁定
- <ModeGroup>.basic.in.autoModeActive
  自动运行方式处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.2 contTravers

PLC → NC	<ModeGroup>.basic.out.contTravers
激活连续运行（默认）	

说明

激活连续运行（默认）
运行方式组内的有效增量预选已重置。

数据类型和更新

BOOL，循环

值 TRUE

已请求机床功能 ContManTravel。

值 FALSE

未请求机床功能 ContManTravel。

关联

- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 10000 个增量”
- <ModeGroup>.basic.out.incVar
  激活“根据请求移动用户自定义增量数”
- <ModeGroup>.basic.in.inc1Active
  “根据请求移动 1 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10Active
  “根据请求移动 10 个增量”处于活动状态
- <ModeGroup>.basic.in.inc100Active
  “根据请求移动 100 个增量”处于活动状态
- <ModeGroup>.basic.in.inc1000Active
  “根据请求移动 1000 个增量”处于活动状态

- <ModeGroup>.basic.in.inc10000Active
  “根据请求移动 10000 个增量”处于活动状态
- <ModeGroup>.basic.in.incVarActive
  “根据请求移动用户自定义增量数”处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.3 disableModeChangeover

PLC → NC	<ModeGroup>.basic.out.disableModeChangeover
切换运行方式锁定	

说明

锁定该运行方式组内的运行方式切换

数据类型和更新

BOOL，循环

值 TRUE

运行方式组的当前生效的运行方式无法切换。

说明

可在运行方式内选择的机床功能可进行切换。

值 FALSE

运行方式组的运行方式可切换。

关联

- <ModeGroup>.basic.out.autoMode
  请求切换至自动运行方式
- <ModeGroup>.basic.out.mdaMode
  请求切换至 MDA 运行方式
- <ModeGroup>.basic.out.jogMode
  请求切换至 JOG 运行方式

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.4 inc1

PLC → NC	<ModeGroup>.basic.out.inc1
激活“根据请求移动 1 个增量”	

说明

激活机床功能 INC1（根据请求移动 1 个增量）

说明

接口的功能性必须显式使能。
(<Nc>.basic.out.selectIncFromModeGroup := TRUE)
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，循环

值 TRUE

已请求机床功能 INC1。

值 FALSE

未请求机床功能 INC1。

关联

- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 10000 个增量”
- <ModeGroup>.basic.out.incVar
  激活“根据请求移动用户自定义增量数”
- <ModeGroup>.basic.in.inc1Active
  “根据请求移动 1 个增量”处于活动状态

- <ModeGroup>.basic.in.inc10Active
  “根据请求移动 10 个增量”处于活动状态
- <ModeGroup>.basic.in.inc100Active
  “根据请求移动 100 个增量”处于活动状态
- <ModeGroup>.basic.in.inc1000Active
  “根据请求移动 1000 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10000Active
  “根据请求移动 10000 个增量”处于活动状态
- <ModeGroup>.basic.in.incVarActive
  “根据请求移动用户自定义增量数”处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.5 inc10

PLC → NC	<ModeGroup>.basic.out.inc10
激活“根据请求移动 10 个增量”	

说明

激活机床功能 INC10（根据请求移动 10 个增量）

说明

接口的功能性必须显式使能。
(<Nc>.basic.out.selectIncFromModeGroup := TRUE)
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，循环

值 TRUE

已请求机床功能 INC10。

值 FALSE

未请求机床功能 INC10。

关联

- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 10000 个增量”
- <ModeGroup>.basic.out.incVar
  激活“根据请求移动用户自定义增量数”
- <ModeGroup>.basic.in.inc1Active
  “根据请求移动 1 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10Active
  “根据请求移动 10 个增量”处于活动状态
- <ModeGroup>.basic.in.inc100Active
  “根据请求移动 100 个增量”处于活动状态
- <ModeGroup>.basic.in.inc1000Active
  “根据请求移动 1000 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10000Active
  “根据请求移动 10000 个增量”处于活动状态
- <ModeGroup>.basic.in.incVarActive
  “根据请求移动用户自定义增量数”处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.6 inc100

PLC → NC	<ModeGroup>.basic.out.inc100
激活“根据请求移动 100 个增量”	

说明

激活机床功能 INC100（根据请求移动 100 个增量）

说明

接口的功能性必须显式使能。
(<Nc>.basic.out.selectIncFromModeGroup := TRUE)
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，循环

值 TRUE

已请求机床功能 INC100。

值 FALSE

未请求机床功能 INC100。

关联

- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 10000 个增量”
- <ModeGroup>.basic.out.incVar
  激活“根据请求移动用户自定义增量数”
- <ModeGroup>.basic.in.inc1Active
  “根据请求移动 1 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10Active
  “根据请求移动 10 个增量”处于活动状态
- <ModeGroup>.basic.in.inc100Active
  “根据请求移动 100 个增量”处于活动状态
- <ModeGroup>.basic.in.inc1000Active
  “根据请求移动 1000 个增量”处于活动状态

- <ModeGroup>.basic.in.inc10000Active
  “根据请求移动 10000 个增量”处于活动状态
- <ModeGroup>.basic.in.incVarActive
  “根据请求移动用户自定义增量数”处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.7 inc1000

PLC → NC	<ModeGroup>.basic.out.inc1000
激活“根据请求移动 100 个增量”	

说明

激活机床功能 INC1000（根据请求移动 1000 个增量）

说明

接口的功能性必须显式使能。
(<Nc>.basic.out.selectIncFromModeGroup := TRUE)
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，循环

值 TRUE

已请求机床功能 INC1000。

值 FALSE

未请求机床功能 INC1000。

关联

- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”

- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 10000 个增量”
- <ModeGroup>.basic.out.incVar
  激活“根据请求移动用户自定义增量数”
- <ModeGroup>.basic.in.inc1Active
  “根据请求移动 1 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10Active
  “根据请求移动 10 个增量”处于活动状态
- <ModeGroup>.basic.in.inc100Active
  “根据请求移动 100 个增量”处于活动状态
- <ModeGroup>.basic.in.inc1000Active
  “根据请求移动 1000 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10000Active
  “根据请求移动 10000 个增量”处于活动状态
- <ModeGroup>.basic.in.incVarActive
  “根据请求移动用户自定义增量数”处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.8 inc10000

PLC → NC	<ModeGroup>.basic.out.inc10000
激活“根据请求移动 1000 个增量”	

说明

激活机床功能 INC10000（根据请求移动 10000 个增量）

说明

接口的功能性必须显式使能。
(<Nc>.basic.out.selectIncFromModeGroup := TRUE)
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，循环

值 TRUE

已请求机床功能 INC10000。

值 FALSE

未请求机床功能 INC10000。

关联

- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 10000 个增量”
- <ModeGroup>.basic.out.incVar
  激活“根据请求移动用户自定义增量数”
- <ModeGroup>.basic.in.inc1Active
  “根据请求移动 1 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10Active
  “根据请求移动 10 个增量”处于活动状态
- <ModeGroup>.basic.in.inc100Active
  “根据请求移动 100 个增量”处于活动状态
- <ModeGroup>.basic.in.inc1000Active
  “根据请求移动 1000 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10000Active
  “根据请求移动 10000 个增量”处于活动状态
- <ModeGroup>.basic.in.incVarActive
  “根据请求移动用户自定义增量数”处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.9 incVar

PLC → NC	<ModeGroup>.basic.out.incVar
激活“根据请求移动用户自定义增量数”	

说明

激活机床功能 incVar（根据请求以用户自定义的数量进行移动）

说明

接口的功能性必须显式使能。
(<Nc>.basic.out.selectIncFromModeGroup := TRUE)
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，循环

值 TRUE

已请求机床功能 INCVar。

值 FALSE

未请求机床功能 INCVar。

关联

- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 10000 个增量”
- <ModeGroup>.basic.in.inc1Active
  “根据请求移动 1 个增量”处于活动状态

- <ModeGroup>.basic.in.inc10Active
  “根据请求移动 10 个增量”处于活动状态
- <ModeGroup>.basic.in.inc100Active
  “根据请求移动 100 个增量”处于活动状态
- <ModeGroup>.basic.in.inc1000Active
  “根据请求移动 1000 个增量”处于活动状态
- <ModeGroup>.basic.in.inc10000Active
  “根据请求移动 10000 个增量”处于活动状态
- <ModeGroup>.basic.in.incVarActive
  “根据请求移动用户自定义增量数”处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.10 jogMode

PLC → NC	<ModeGroup>.basic.out.jogMode
请求切换至 JOG 运行方式	

数据类型和更新

BOOL，循环

值 TRUE

存在切换至 JOG 运行方式的请求。

值 FALSE

不存在切换至 JOG 运行方式的请求。

关联

- <ModeGroup>.basic.out.autoMode
  请求切换至自动运行方式
- <ModeGroup>.basic.out.mdaMode
  请求切换至 MDA 运行方式

- <ModeGroup>.basic.out.disableModeChangeover
  切换运行方式锁定
- <ModeGroup>.basic.in.jogModeActive
  JOG 运行方式处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.11 mdaMode

PLC → NC	<ModeGroup>.basic.out.mdaMode
请求切换至 MDA 运行方式	

说明

<ModeGroup>.basic.out.disableModeChangeover（运行方式切换锁定）== 1 时不相关

数据类型与更新

BOOL，循环

值 TRUE

存在切换至 MDI 运行方式的请求。

值 FALSE

不存在切换至 MDI 运行方式的请求。

关联

- <ModeGroup>.basic.out.autoMode
  请求切换至自动运行方式
- <ModeGroup>.basic.out.jogMode
  请求切换至 JOG 运行方式
- <ModeGroup>.basic.out.disableModeChangeover
  切换运行方式锁定
- <ModeGroup>.basic.in.mdaModeActive
  MDA 运行方式处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.12 refPoint

PLC → NC	<ModeGroup>.basic.out.refPoint
请求机床功能 REF	

说明

仅在 JOG 运行方式下生效。

数据类型与更新

BOOL，循环

值 TRUE

存在激活机床功能 REF 的请求。

值 FALSE

不存在激活机床功能 REF 的请求。

关联

- <ModeGroup>.basic.in.jogModeActive
  JOG 运行方式处于活动状态
- <ModeGroup>.basic.in.refActive
  机床功能 REF 处于活动状态

更多信息

- 功能手册之进给轴和主轴；章节“回参考点”

12.3.53.13 repos

PLC → NC	<ModeGroup>.basic.out.repos
请求机床功能 REPOS	

说明

仅在 JOG 运行方式下生效。

数据类型与更新

BOOL，循环

值 TRUE

存在激活机床功能 REPOS 的请求。

值 FALSE

不存在激活机床功能 REPOS 的请求。

关联

- <ModeGroup>.basic.in.jogModeActive
  odeGroup>.basic.in.jogModeActive
- <ModeGroup>.basic.in.reposActive
  机床功能 REPOS 活

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.14 reset

PLC → NC	<ModeGroup>.basic.out.reset
请求复位运行方式组	

说明

- **特殊情况**
  如果通过报警复位接口信号 <ModeGroup>.basic.in.ready（运行方式组就绪），这种情况下运行方式组的所有通道都不再处于“复位”通道状态。如需切换运行方式组的运行方式，必须先通过 <ModeGroup>.basic.out.reset 触发“运行方式组复位”。

说明

PLC 用户程序发出运行方式组复位请求后，必须先等待通道进入“运行方式组的所有通道都位于复位状态”(<ModeGroup>.basic.in.resetAllChanDone) == 1) 这一运行方式组状态，或者进入“复位”(<Chan>.basic.in.stateReset == 1) 这一通道状态，才能请求在运行方式组的相关通道中执行新动作（例如 NC 启动或轴交换）。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

已触发对“运行方式组复位”的请求。响应：

- 在运行方式组的所有通道中，控制器内部根据 <Chan>.basic.out.reset = 1 触发“通道复位”。
- 随后所有通道都处于“复位”状态下
  <Chan>.basic.in.stateReset == 1
- 所有处于活动状态的 NC 程序随后都处于“已取消”程序状态下
  <Chan>.basic.in.progStateAborted == 1
- 所有运行的轴和主轴均根据其加速度特性曲线，在轮廓范围内制动至静止状态。
- 通道专用初始设置生效。

脉冲沿切换 1 → 0

最后的对“运行方式组复位”的请求复位。从此刻起可以重新请求“运行方式组复位”。

关联

- <Chan>.basic.out.reset
  已请求通道复位
- <ModeGroup>.basic.in.resetAllChanDone
  运行方式组内的所有通道都处于复位状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.15 singleBlockTypeA

PLC → NC	<ModeGroup>.basic.out.singleBlockTypeA
单段处理运行方式，选择类型 A	

说明

激活“单程序段类型 A”时的跨运行方式组特性：

- 所有通道停止
- 所有通道获取启动指令
- KS 通道在程序段末尾停止。
- KA 通道接收 STOPATEND 指令，与 <Chan>.basic.out.ncStopBlockEnd（程序段交界处 NC 停止）类似

所有通道（随时）在程序段交界处停止。

说明

- <ModeGroup>.basic.out.singleBlockTypeB == 1 且
  <ModeGroup>.basic.out.singleBlockTypeA == 0
  ⇒ B 型单段
- <ModeGroup>.basic.out.singleBlockTypeB == 0 且
  <ModeGroup>.basic.out.singleBlockTypeA == 1
  ⇒ A 型单段
- <ModeGroup>.basic.out.singleBlockTypeB == 1 且
  <ModeGroup>.basic.out.singleBlockTypeA == 1 **或**
  <ModeGroup>.basic.out.singleBlockTypeB == 0 且
  <ModeGroup>.basic.out.singleBlockTypeA == 0
  ⇒ 未选择运行方式组专用单段

数据类型与更新

BOOL，循环

值 TRUE

存在激活“单程序段类型 A”的请求。

值 FALSE

不存在激活“单程序段类型 A”的请求。

关联

- <ModeGroup>.basic.out.singleBlockTypeB
  单段处理运行方式，选择类型 B

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.16 singleBlockTypeB

PLC → NC	<ModeGroup>.basic.out.singleBlockTypeB
单段处理运行方式，选择类型 B	

说明

激活“单程序段类型 B”时的跨运行方式组特性：

- 所有通道停止
- 所有通道获取启动指令

- KS 通道在程序段末尾停止。
- KA 通道接收 STOPATEND 指令，与 <Chan>.basic.out.ncStopBlockEnd（程序段交界处 NC 停止）类似

所有通道（随时）在程序段交界处停止。

说明

- <ModeGroup>.basic.out.singleBlockTypeB == 1 且
  <ModeGroup>.basic.out.singleBlockTypeA
  ⇒ B 型单段
- <ModeGroup>.basic.out.singleBlockTypeB == 0 且
  <ModeGroup>.basic.out.singleBlockTypeA == 1
  ⇒ A 型单段
- <ModeGroup>.basic.out.singleBlockTypeB == 1 且
  <ModeGroup>.basic.out.singleBlockTypeA == 1 **或**
  <ModeGroup>.basic.out.singleBlockTypeB == 0 且
  <ModeGroup>.basic.out.singleBlockTypeA == 0
  ⇒ 未选择运行方式组专用单段

数据类型与更新

BOOL，循环

值 TRUE

存在激活“单程序段类型 B”的请求。

值 FALSE

不存在激活“单程序段类型 B”的请求。

关联

- <ModeGroup>.basic.out.singleBlockTypeA
  单段处理运行方式，选择类型 A

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.17 stopAll

PLC → NC	<ModeGroup>.basic.out.stopAll
请求停止运行方式组（轴和主轴）	

说明

请求停止所有已分配给运行方式组的通道（NC 停止），包括轴和主轴

数据类型和更新

BOOL，循环

值 TRUE

存在对“运行方式组停止，进给轴和主轴”的请求。

响应：

- 在运行方式组的所有通道中触发“NC 停止”。
- 因运行方式组停止而引起的通道状态切换：
  – “生效”→“中断”
  – “复位”→“复位”
- 因运行方式组停止而引起的程序状态切换：
  – “运行”→“停止”
- 运行方式组的所有运行的轴和主轴均根据其加速度特性曲线，在轮廓范围内制动至静止状态。

停止的程序可通过“NC 启动”继续执行。运行方式组的主轴不受影响。

值 FALSE

不存在对“运行方式组停止，进给轴和主轴”的请求。

说明

停止的程序可通过“NC 启动”继续执行。运行方式组的主轴不受影响。即主轴停止时程序继续运行。

关联

- <Chan>.basic.out.ncStopBlockEnd
  请求在当前程序段结尾执行 NC 停止
- <Chan>.basic.out.ncStop
  请求 NC 停止
- <Chan>.basic.in.stateActive
  通道状态“激活”
- <Chan>.basic.in.stateInterrupted
  通道状态“已中断”

- <Chan>.basic.progStateRunning
  程序状态“正在运行”
- <Chan>.basic.progStateStopped
  程序状态“停止”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.18 stopAxesOnly

PLC → NC	<ModeGroup>.basic.out.stopAxesOnly
请求停止运行方式组（仅轴）	

说明

请求停止所有已分配给运行方式组的通道（NC 停止），包括轴

数据类型和更新

BOOL，循环

值 TRUE

存在对“运行方式组停止”的请求。

响应：

- 在运行方式组的所有通道中触发“NC 停止”。
- 因运行方式组停止而引起的通道状态切换：
  – “生效”→“中断”
  – “复位”→“复位”
- 因运行方式组停止而引起的程序状态切换：
  – “运行”→“停止”
- 运行方式组的所有运行的轴均根据其加速度特性曲线，在轮廓范围内制动至静止状态。

停止的程序可通过“NC 启动”继续执行。运行方式组的主轴不受影响。

值 FALSE

不存在对“运行方式组停止”的请求。

关联

- <Chan>.basic.out.ncStopBlockEnd
  请求在当前程序段结尾执行 NC 停止
- <Chan>.basic.out.ncStop
  请求 NC 停止
- <Chan>.basic.in.stateActive
  通道状态“激活”
- <Chan>.basic.in.stateInterrupted
  通道状态“已中断”
- <Chan>.basic.progStateRunning
  程序状态“正在运行”
- <Chan>.basic.progStateStopped
  程序状态“停止”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.53.19 teachIn

PLC → NC	<ModeGroup>.basic.out.teachIn
请求机床功能 TEACH IN	

说明

仅在 JOG 运行方式下生效。

数据类型与更新

BOOL，循环

值 TRUE

存在激活机床功能 TEACH IN 的请求。

值 FALSE

不存在激活机床功能 TEACH IN 的请求。

关联

- <ModeGroup>.basic.in.jogModeActive
  JOG 运行方式处于活动状态
- <ModeGroup>.basic.in.teachInActive
  机床功能 TEACH IN 处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.54 NCK_typeNc

下表说明了数据类型 NCK_typeNc 的含义：

结构		说明
.config		NC 的组态数据
	.activeAxes	生效轴：
	.activeChans	生效通道
	.activeModeGroups	生效的运行方式组
	.isVirtual	NC 为虚拟（数字孪生）
	.maxAxis	最高生效轴编号
	.maxChan	最高生效通道编号
	.maxModeGroup	生效的运行方式组数量
	.ncCycleTimeout	NC 循环使用寿命监控时间
	.ncRunupTimeout	NC 引导启动监控时间
	.registered	NC 已注册
.basic.in (页 913)		来自 NC 的通用信号
.basic.out (页 922)		发送至 NC 的通用信号
.handwheels.in (页 928)		来自 NC 的手轮信号
.digitalInputs.in (页 929)		来自 NC 的数字输入信号
.digitalInputs.out (页 930)		发送至 NC 的数字输入信号

结构	说明
.digitalOutputs.in (页 932)	来自 NC 的数字输出信号
.digitalOutputs.out (页 933)	发送至 NC 的数字输出信号
.analogInputs.in (页 937)	来自 NC 的模拟量输入信号
.analogInputs.out (页 938)	发送至 NC 的模拟量输入信号
.analogOutputs.in (页 941)	来自 NC 的模拟量输出信号
.analogOutputs.out (页 942)	发送至 NC 的模拟量输出信号
.robot.in	来自 NC 的机器人信号
.robot.out	发送至 NC 的机器人信号
.collisionProtAreas.in (页 946)	来自 NC 的干涉保护范围信号
.collisionProtAreas.out (页 948)	发送至 NC 的干涉保护范围信号
.collision.out (页 949)	发送至 NC 的干涉保护信号

12.3.55 <Nc>.basic.in

12.3.55.1 collisionMonitoringActive

NC → PLC	<Nc>.basic.in.collisionMonitoringActive
碰撞监测：保护区域监控处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

防撞保护 - 保护区域监控处于活动状态。

值 FALSE

防撞保护 - 保护区域监控未激活。

12.3.55.2 drivesCyclicOperation

NC → PLC	<Nc>.basic.in.drivesCyclicOperation
NC 所有机床轴的相关驱动都处于循环运行中，即与 NC 循环交换 PROFIdrive 报文	

数据类型与更新

BOOL，循环

值 TRUE

NC 所有机床轴的相关驱动都处于循环运行中，即与 NC 循环交换 PROFIdrive 报文。

值 FALSE

NC 中至少有一个机床轴，其相应的驱动未处于循环运行中，即不与 NC 循环交换 PROFIdrive 报文。

12.3.55.3 drivesReady

NC → PLC	<Nc>.basic.in.drivesReady
NC 所有机床轴的相关驱动都已准备就绪	

数据类型与更新

BOOL，循环

值 TRUE

NC 的所有机床轴所对应的驱动均处于运行就绪状态：

<Axis>.drive.in.ready == 1（驱动准备就绪）

值 FALSE

NC 的至少一根机床轴所对应的驱动不处于运行就绪状态：

<Axis>.drive.in.ready == 0（驱动准备就绪）

关联

- <Axis>.drive.in.ready
  驱动就绪

12.3.55.4 emergencyStopActive

NC → PLC	<Nc>.basic.in.emergencyStopActive
急停处于活动状态	

说明

为了应答 NC 的“急停”状态，以下接口信号必须一直保持置位，直至接口信号

<Nc>.basic.in.emergencyStopActive (LBP_NC.E_EMERGENCY) 复位：

- <Nc>.basic.out.emergencyStopAck (LBP_NC.A_EMERGENCY_Ackn) = 1（应答急停）
- <ModeGroup>.basic.out.reset (LBP_ModeGroup.A_MGReset) = 1（手持终端复位），适用于 NC 的所有手持终端

数据类型和更新

BOOL，循环

值 TRUE

急停处于活动状态。

值 FALSE

急停未激活。

关联

- <Nc>.basic.out.emergencyStop
  已请求急停
- <Nc>.basic.out.emergencyStopAck
  已请求应答“急停”状态
- <ModeGroup>.basic.out.reset
  请求复位运行方式组

更多信息

- 基础功能功能手册；“急停”章节

12.3.55.5 imperialMetricChangeCounter

NC → PLC	<Nc>.basic.in.imperialMetricChangeCounter
更改计数器英制（英寸）/公制单位制	

数据类型和更新

USInt

值

英制/公制单位系统更改计数器

12.3.55.6 imperialUnitsActive

NC → PLC	<Nc>.basic.in.imperialUnitsActive
所有设定值和实际值都以英制单位（英寸）解释/显示	

数据类型与更新

BOOL，循环

值 TRUE

设定值和实际值以英寸为单位进行解释和显示。

值 FALSE

设定值和实际值不以英寸为单位进行解释和显示。

12.3.55.7 measuringProbe1Actuated

NC → PLC	<Nc>.basic.in.measuringProbe1Actuated
已操作探针 1	

数据类型与更新

BOOL，循环

值 TRUE

测头偏转。

值 FALSE

测头未偏转。

更多信息

- 技术功能手册；“测量”章节

12.3.55.8 measuringProbe2Actuated

NC → PLC	<Nc>.basic.in.measuringProbe2Actuated
已操作探针 2	

数据类型与更新

BOOL，循环

值 TRUE

测头偏转。

值 FALSE

测头未偏转。

更多信息

- 技术功能手册；“测量”章节

12.3.55.9 ncAlarmActive

NC → PLC	<Nc>.basic.in.ncAlarmActive
至少有一个 NC 报警处于活动状态	

说明

此接口信号是所有现有通道的接口信号的概要：

<Chan>.basic.in.ncAlarmActive（存在通道专用 NC 报警）

数据类型与更新

BOOL，循环

值 TRUE

存在至少一个 NC 报警。

值 FALSE

没有待处理的 NC 报警。

关联

- <Chan>.basic.in.ncAlarmActive
  通道内至少有一个 NC 报警处于活动状态
- <Chan>.basic.in.ncAlarmWithStopActive
  通道内至少有一个会触发加工停止的 NC 报警处于活动状态

更多信息

- 诊断手册

12.3.55.10 ncReady

NC → PLC	<Nc>.basic.in.ncReady
NC 准备就绪：控制器处于循环运行状态	

数据类型和更新

BOOL，循环

值 TRUE

控制系统就绪。

接口信号为继电器触点"NC 就绪"的映射。

以下情形下，信号会置位：

- "NC Ready"继电器触点闭合
- 所有控制器内部电压都已建立
- 控制器处于"循环运行"状态下

值 FALSE

控制系统未就绪。

当"NC Ready"继电器触点打开时，信号复位

可能的原因：

- 欠压或过压监控已响应
- 某些组件未就绪（NC-CPU 就绪）
- NC-CPU 的监视器已响应

在尚且可能的情况下，控制系统内部作出以下响应：

- NC：取消伺服使能 ⇒ 驱动停止
- PLC 基本程序：
  - 清除从 NC 发送至 PLC（用户接口）的状态信号
  - 清除辅助功能的修改信号
  - 结束对 PLC-NC 用户接口的循环处理

解决方法：

至少触发一次暖启动，才能退出故障状态。

关联

- "SINU_Update"、 输出信号：ncCpuReady

12.3.55.11 ncuAirTemperatureAlarmActive

NC → PLC	<Nc>.basic.in.ncuAirTemperatureAlarmActive
环境温度 NC 监控和（或）风扇监控已响应	

数据类型与更新

BOOL，循环

值 TRUE

环境温度监控和/或风扇监控已响应。

可能的原因：

- 过高的环境温度
- 用于模块冷却的 24V 直流风扇的转速监控作出响应。

可行解决方法：

更换风扇，或提供额外的通风。

值 FALSE

环境温度监控和风扇监控未响应。

关联

- 馈电/反馈单元的继电器触点：端子 5.1、5.2 或 5.1、5.3

更多信息

- 诊断手册

12.3.55.12 ncuHeatSinkAlarmActive

NC → PLC	<Nc>.basic.in.ncuHeatSinkAlarmActive
已超过 NCU 散热器温度极限值。无法继续保证 NCU 连续运行。	

数据类型与更新

BOOL，循环

值 TRUE

超出了 NCU 的散热器温度限值。无法继续确保 NCU 持续运行。

值 FALSE

未超出 NCU 的散热器温度限值。

12.3.55.13 sinuOperateReady

NC → PLC	<Nc>.basic.in.sinuOperateReady
SINUMERIK Operate 运行准备就绪，周期性向 NC 报告	

数据类型与更新

BOOL，循环

值 TRUE

SINUMERIK Operate 已运行就绪并循环向 NC 报告。

值 FALSE

SINUMERIK Operate 未就绪。

12.3.55.14 swCams[1..32].minus

NC → PLC	<Nc>.basic.in.swCams[1..32].minus
直线轴通过负凸轮沿负方向移动（模数回转轴行为不同）	

说明

负挡块信号 1 至 32 的脉冲沿和轴（回转轴）的运行方向相关，按插补周期发送给 PLC 接口。

数据类型与更新

ARRAY[1..32] of BOOL，循环

值 TRUE

- 线性轴
  当轴朝负向移动越过负挡块时，负挡块信号会从 0 变 1。
- 模数回转轴
  每次正挡块信号输出上升沿时，负挡块信号便切换电平。

值 FALSE

- 线性轴
  当轴朝正向移动越过负挡块时，负挡块信号会从 1 变 0。
- 模数回转轴
  每次正挡块信号输出上升沿时，负挡块信号便切换电平。

关联

- <Nc>.basic.in.swCams[1..32].plus
  直线轴通过正凸轮沿正方向移动（模数回转轴行为不同）
- <Axis>.basic.out.swCams
  激活“输出软件凸轮正信号和负信号”

更多信息

- 轴和主轴功能手册；“软件凸轮，行程开关信号”章节

12.3.55.15 swCams[1..32].plus

NC → PLC	<Nc>.basic.in.swCams[1..32].plus
直线轴通过正凸轮沿正方向移动（模数回转轴行为不同）	

说明

正挡块信号 1 至 32 的脉冲沿和轴（回转轴）的运行方向相关，按插补周期发送给 PLC 接口。

说明

描述的采用模数回转轴时的正挡块特性适用于以下条件：

正挡块 – 负挡块 < 180 度

如果不满足此条件或者所选的负挡块大于正挡块，则正挡块信号的响应正好相反。负挡块信号的响应保持不变。

数据类型与更新

ARRAY[1..32] of BOOL

值 TRUE

- 线性轴
  当轴朝正向移动越过正挡块时，正挡块信号会从 0 变 1。
- 模数回转轴
  当轴正转越过负挡块时，正挡块信号会从 0 变 1。

值 FALSE

- 线性轴
  当轴朝负向移动越过正挡块时，正挡块信号会从 1 变 0。
- 模数回转轴
  当轴正转越过正挡块时，正挡块信号会从 1 变 0。

关联

- <Nc>.basic.in.swCams[1..32].minus
  直线轴通过负凸轮沿负方向移动（模数回转轴行为不同）
- <Axis>.basic.out.swCams
  激活“输出软件凸轮正信号和负信号”

更多信息

- 轴和主轴功能手册；“软件凸轮，行程开关信号”章节

12.3.56 <Nc>.basic.out

12.3.56.1 emergencyStop

PLC → NC	<Nc>.basic.out.emergencyStop
已请求急停	

说明

所有机床轴按照为各轴单独编程的时间进行制动：MD36610 $MA_AX_EMERGENCY_STOP_TIME

数据类型与更新

BOOL，循环

值 TRUE

已请求急停。

值 FALSE

未请求急停。

关联

- <Nc>.basic.out.emergencyStopAck
  已请求应答“急停”状态
- <Nc>.basic.in.emergencyStopActive
  急停处于活动状态
- MD36610 $MA_AX_EMERGENCY_STOP_TIME
  出错时的制动斜坡时间

更多信息

- 基础功能功能手册；“急停”章节

12.3.56.2 emergencyStopAck

PLC → NC	<Nc>.basic.out.emergencyStopAck
已请求应答“急停”状态	

说明

为了应答 NC 的“急停”状态，以下接口信号必须一直保持置位，直至变量 <Nc>.basic.out.emergencyStop（急停处于活动状态）复位为止：

- <Nc>.basic.out.emergencyStopAck = TRUE（应答急停）
- <ModeGroup>.basic.out.reset = TRUE（运行方式组复位），适用于 NC 的所有运行方式组

数据类型与更新

BOOL，循环

值 TRUE

已请求对“急停”状态的应答。

值 FALSE

未请求对“急停”状态的应答。

关联

- <Nc>.basic.out.emergencyStop
  已请求急停
- <Nc>.basic.in.emergencyStopActive
  急停处于活动状态
- <ModeGroup>.basic.out.reset
  请求复位运行方式组

更多信息

- 基础功能功能手册；“急停”章节

12.3.56.3 enableReducedDynamics

PLC → NC	<Nc>.basic.out.enableReducedDynamics
接通请求“低动态模式”	

说明

接通/关闭该功能：

- 0：关闭请求“低动态模式”
- 1：接通请求“低动态模式”

数据类型和更新

BOOL

12.3.56.4 keyOperatedSwitchPos0

PLC → NC	<Nc>.basic.out.keyOperatedSwitchPos0
根据钥匙开关位置 0，可使能或锁定对 NC 中特定元件的访问	

说明

根据钥匙开关的位置，可禁止或使能对 NC 中特定单元的访问：

- 钥匙开关位置 0 具有最低访问权限。
- 钥匙开关位置 3 具有最高访问权限。

钥匙开关位置 1 - 3 的接口信号可直接通过机床控制面板的钥匙开关或 PLC 用户程序设定。

这其中只可设置一个位。若同时有多个位置位，那么系统内部会激活开关位置 3。

数据类型与更新

BOOL，循环

值 TRUE

钥匙开关位置 0

值 FALSE

钥匙开关位置不为 0

关联

- 用于存取级别的机床数据：MD11612, MD51044 - MD51064, MD51070 - MD51073, MD51199 - MD51211, MD51215 - MD51225, MD51235
- 通过口令加密

12.3.56.5 keyOperatedSwitchPos1

PLC → NC	<Nc>.basic.out.keyOperatedSwitchPos1
根据钥匙开关位置 1，可使能或锁定对 NC 中特定元件的访问	

说明

根据钥匙开关的位置，可禁止或使能对 NC 中特定单元的访问：

- 钥匙开关位置 0 具有最低访问权限。
- 钥匙开关位置 3 具有最高访问权限。

钥匙开关位置 1 - 3 的接口信号可直接通过机床控制面板的钥匙开关或 PLC 用户程序设定。

这其中只可设置一个位。若同时有多个位置位，那么系统内部会激活开关位置 3。

数据类型与更新

BOOL，循环

值 TRUE

钥匙开关位置 1

值 FALSE

钥匙开关位置不为 1

关联

- 用于存取级别的机床数据：MD11612, MD51044 - MD51064, MD51070 - MD51073, MD51199 - MD51211, MD51215 - MD51225, MD51235
- 通过口令加密

12.3.56.6 keyOperatedSwitchPos2

PLC → NC	<Nc>.basic.out.keyOperatedSwitchPos2
根据钥匙开关位置 2，可使能或锁定对 NC 中特定元件的访问	

说明

根据钥匙开关的位置，可禁止或使能对 NC 中特定单元的访问：

- 钥匙开关位置 0 具有最低访问权限。
- 钥匙开关位置 3 具有最高访问权限。

钥匙开关位置 1 - 3 的接口信号可直接通过机床控制面板的钥匙开关或 PLC 用户程序设定。

这其中只可设置一个位。若同时有多个位置位，那么系统内部会激活开关位置 3。

数据类型与更新

BOOL，循环

值 TRUE

钥匙开关位置 2

值 FALSE

钥匙开关位置不为 2

关联

- 用于存取级别的机床数据：MD11612, MD51044 - MD51064, MD51070 - MD51073, MD51199 - MD51211, MD51215 - MD51225, MD51235
- 通过口令加密

12.3.56.7 keyOperatedSwitchPos3

PLC → NC	<Nc>.basic.out.keyOperatedSwitchPos3
根据钥匙开关位置 3，可使能或锁定对 NC 中特定元件的访问	

说明

根据钥匙开关的位置，可禁止或使能对 NC 中特定单元的访问：

- 钥匙开关位置 0 具有最低访问权限。
- 钥匙开关位置 3 具有最高访问权限。

钥匙开关位置 1 - 3 的接口信号可直接通过机床控制面板的钥匙开关或 PLC 用户程序设定。

这其中只可设置一个位。若同时有多个位置位，那么系统内部会激活开关位置 3。

数据类型与更新

BOOL，循环

值 TRUE

钥匙开关位置 3

值 FALSE

钥匙开关位置不为 3

关联

- 用于存取级别的机床数据：MD11612, MD51044 - MD51064, MD51070 - MD51073, MD51199 - MD51211, MD51215 - MD51225, MD51235
- 通过口令加密

12.3.56.8 selectIncFromModeGroup

PLC → NC	<Nc>.basic.out.selectIncFromModeGroup
可在运行方式组内选择按增量运行	

说明

INC 选择被传输至运行方式组接口。

数据类型与更新

BOOL

值 TRUE

运行方式组接口中的 INC 选择处于活动状态

值 FALSE

轴和几何轴接口中的 INC 选择处于活动状态

12.3.56.9 transferActToolBasePos

PLC → NC	<Nc>.basic.out.transferActToolBasePos
传输轴的当前 MCS 设定值	

数据类型和更新

BOOL，周期性

TRUE 值

轴的当前 MCS 设定值输出到界面上。

FALSE 值

轴的当前 MCS 设定值未输出到界面上。

对应于

- <Axis>.basic.in.actToolBasePos
  轴的当前 MCS 设定值

更多信息

- 功能手册之基本功能，部分“坐标系中的位置”

12.3.57 <Nc>.handwheels[x].in

12.3.57.1 movedCounter

NC → PLC	<Nc>.handwheels[1].in.movedCounter
手轮无方向、不按比例计数。可用于故障查找	

说明

手轮运动计数器可用于调试

数据类型和更新

USInt，循环

值

手轮运动计数器，不受方向影响

12.3.57.2 stationary

NC → PLC	<Nc>.handwheels[1].in.stationary
以太网手轮静止	

说明

静态辨识通过连接手轮的以太网模块进行。若一个手轮在定义的时间内未传输手轮脉冲，以太网模块会将该状态辨识为手轮静止并将该状态传输至 NC/PLC 接口。

数据类型与更新

BOOL，循环

值 TRUE

以太网手轮静止

值 FALSE

以太网手轮移动

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.58 <Nc>.digitalInputs[x].in

12.3.58.1 active

NC → PLC	<Nc>.digitalInputs[1].in.active
NC 硬件数字输入端的实际状态	

说明

通过实际值可在 PLC 用户程序中读取当前 NC 输出值。

说明

“实际值”接口中的值可能因后续不同的影响因素 - “锁定”和“设置”，而与 NC 输出端的值有所不同。

数据类型与更新

BOOL，循环

值 TRUE

实际值的值为 1。

值 FALSE

实际值的值为 0。

关联

- <Nc>.digitalInputs[x].out.lock
  如果 digitalInputs[].out.set 未生效，则将软件数字输入端"$A_IN[]""锁定"为定义值 0
- <Nc>.digitalInputs[x].out.set
  将定义值 1 赋给 NC 软件数字输入端"$A_IN[]"
- <Nc>.digitalInputs[x].in.active
  NC 硬件数字输入端的实际状态
- MD10350 $MN_FASTIO_DIG_NUM_INPUTS

更多信息

- 基础功能功能手册；"数字和模拟 NC 外围设备"章节

12.3.59 <Nc>.digitalInputs[x].out

12.3.59.1 lock

PLC → NC	<Nc>.digitalInputs[1].out.lock
如果 digitalInputs[].out.set 未生效，则将软件数字输入端"$A_IN[]""锁定"为定义值 0	

说明

在读取时，被禁止的输入提供值 0。

数据类型与更新

BOOL，循环

值 TRUE

NC 数字量输入被禁止。

值 FALSE

NC 数字量输入被使能。

关联

- <Nc>.digitalInputs[1].out.set
  将定义值 1 赋给 NC 软件数字输入端"$A_IN[]"
- <Nc>.digitalInputs[1].in.active
  NC 硬件数字输入端的实际状态
- MD10350 $MN_FASTIO_DIG_NUM_INPUTS

更多信息

- 基础功能功能手册；"数字和模拟 NC 外围设备"＞"通过 PLC 进行间接外围设备访问"＞"NC 数字输入/输出端"＞"NC 输入端"章节

12.3.59.2 set

PLC → NC	<Nc>.digitalInputs[1].out.set
将定义值 1 赋给 NC 软件数字输入端"$A_IN[]"	

说明

如果 PLC 用户程序为 NC 输入端赋值 TRUE，NCU 板载输入端的信号状态和 NC 输入端的锁定会失效。

数据类型与更新

BOOL，循环

值 TRUE

将 NC 输入值既定地设置为值 1。

值 FALSE

不影响 NC 输入值。

关联

- <Nc>.digitalInputs[1].out.lock
  如果 digitalInputs[].out.set 未生效，则将软件数字输入端"$A_IN[]""锁定"为定义值 0
- <Nc>.digitalInputs[1].in.active
  NC 硬件数字输入端的实际状态
- MD10350 $MN_FASTIO_DIG_NUM_INPUTS

更多信息

- 基础功能功能手册；“数字和模拟 NC 外围设备”>“通过 PLC 进行间接外围设备访问”>“NC 数字输入/输出端”>“NC 输入端”章节

12.3.60 <Nc>.digitalOutputs[x].in

12.3.60.1 setpointState

NC → PLC	<Nc>.digitalOutputs[1].in.setpointState
如果 digitalOutputs[].out.overwrite 未生效，则通过 NC 设置 NC 软件数字输出端“$A_OUT[]”的状态	

说明

通过设定值可在 PLC 用户程序中读取当前 NC 输出值。

说明

接口“设定值”中的值可能因随后不同的影响方法“设定”和“禁止”而不同于 NC 输出上的值。

数据类型与更新

BOOL，循环

值 TRUE

设定值的值为 1。

值 FALSE

设定值的值为 0。

关联

- <Nc>.digitalOutputs[1].out.lock
  将 NC 硬件数字输出端“锁定”在定义状态 0
- <Nc>.digitalOutputs[1].out.overwrite
  脉冲沿切换 0 → 1：将“digitalOutpute[].out.settingState”传递到硬件数字输出端，为 NC 数字输出状态“$A_OUT[]”赋值 0

- <Nc>.digitalOutputs[1].out.settingState
  如果“digitalOutputs[].out.overwrite”或“digitalOutputs[].out.target”处于活动状态，则将值传输至 NC 硬件数字输出端
- MD10360 $MN_FASTIO_DIG_NUM_OUTPUTS

更多信息

- 基础功能功能手册；“数字和模拟 NC 外围设备”章节

12.3.61 <Nc>.digitalOutputs[x].out

12.3.61.1 lock

PLC → NC	<Nc>.digitalOutputs[1].out.lock
将 NC 硬件数字输出端“锁定”在定义状态 0	

说明

若 NC 数字量输出被禁止，则在硬件输出上既定地输出 0 V。

若 NC 数字量输出未被禁止，在则硬件输出上输出在 NC 程序中或由 PLC 用户程序设定的值。

数据类型与更新

BOOL，循环

值 TRUE

NC 数字量输出被禁止。

值 FALSE

NC 数字量输出被使能。

关联

- <Nc>.digitalOutputs[x].out.overwrite
  脉冲沿切换 0 → 1：将“digitalOutpute[].out.settingState”传递到硬件数字输出端，为 NC 数字输出状态“$A_OUT[]”赋值 0
- <Nc>.digitalOutputs[x].out.settingState
  如果“digitalOutputs[].out.overwrite”或“digitalOutputs[].out.target”处于活动状态，则将值传输至 NC 硬件数字输出端

- <Nc>.digitalOutputs[x].out.target
  将“digitalOutputs[].out.settingState”传递到硬件数字输出端；NC 软件数字输出端“$A_OUT[]”的初始状态保持不变
- MD10360 $MN_FASTIO_DIG_NUM_OUTPUTS

更多信息

- 基础功能功能手册；“数字和模拟 NC 外围设备”>“通过 PLC 进行间接外围设备访问”>“NC 数字输入/输出端”>“NC 输出端”章节

12.3.61.2 overwrite

PLC → NC	<Nc>.digitalOutputs[1].out.overwrite
脉冲沿切换 0 → 1：将“digitalOutpute[].out.settingState”传递到硬件数字输出端，为 NC 数字输出状态“$A_OUT[]”赋值 0	

说明

在上升沿切换 0 → 1 中，相关输出端不使用通过系统变量 $A_OUT 写入的值，而是使用由 PLC 用户程序指定的设置值。此时，通过系统变量 $A_OUT 写入的值会丢失。

在下降沿切换 1 → 0 时，则为相应输出保留硬件输出上的当前值。

说明

变量 <Nc>.digitalOutputs[x].out.settingState（NC 数字输出端：设置值）由以下变量共用：

- <Nc>.digitalOutputs[x].out.overwrite
  （覆盖），脉冲沿切换 0 → 1 时
- <Nc>.digitalOutputs[x].out.target
  （规定），信号状态 TRUE 下

必须避免同时激活这两个接口。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

激活对应的“设置值”。

脉冲沿切换 1 → 0

无作用。

关联

- <Nc>.digitalOutputs[x].out.lock
  将 NC 硬件数字输出端“锁定”在定义状态 0
- <Nc>.digitalOutputs[x].out.settingState
  如果“digitalOutputs[].out.overwrite”或“digitalOutputs[].out.target”处于活动状态，则将值传输至 NC 硬件数字输出端
- MD10360 $MN_FASTIO_DIG_NUM_OUTPUTS

更多信息

- 基础功能功能手册；“数字和模拟 NC 外围设备”章节

12.3.61.3 settingState

PLC → NC	<Nc>.digitalOutputs[1].out.settingState
如果“digitalOutputs[].out.overwrite”或“digitalOutputs[].out.target”处于活动状态，则将值传输至 NC 硬件数字输出端	

说明

通过设置值可由 PLC 用户程序指定一个输出值。为了使设置值生效，必须通过改值位或写值位将其激活。

说明

变量 <Nc>.digitalOutputs[x].out.settingState（NC 数字输出端：设置值）由以下变量共用：

- <Nc>.digitalOutputs[x].out.overwrite（覆盖），脉冲沿切换 0 → 1 时
- <Nc>.digitalOutputs[x].out.target（规定），信号状态 TRUE 下

必须避免同时激活这两个接口。

数据类型与更新

BOOL，循环

值 TRUE

设置值的值为 1。

值 FALSE

设置值的值为 0。

关联

- <Nc>.digitalOutputs[x].out.lock
  将 NC 硬件数字输出端“锁定”在定义状态 0
- <Nc>.digitalOutputs[x].out.overwrite
  脉冲沿切换 0 → 1：将“digitalOutpute[].out.settingState”传递到硬件数字输出端，为 NC 数字输出状态“$A_OUT[]”赋值 0
- MD10360 $MN_FASTIO_DIG_NUM_OUTPUTS

更多信息

- 基础功能功能手册；“数字和模拟 NC 外围设备”>“通过 PLC 进行间接外围设备访问”>“NC 数字输入/输出端”>“NC 输出端”章节

12.3.61.4 target

PLC → NC	<Nc>.digitalOutputs[1].out.target
将“digitalOutputs[].out.settingState”传递到硬件数字输出端；NC 软件数字输出端“$A_OUT[]”的初始状态保持不变	

说明

设置比特位后，相关输出端不使用 NC 输出值，而是使用由 PLC 用户程序指定的设置值。此时保留当前 NC 输出值。

将比特位复位后，对于相关输出端来说，最后的 NC 输出值重新生效。

说明

变量<Nc>.digitalOutputs[x].out.settingState（NC 数字输出端：设置值）由以下变量共用：

- <Nc>.digitalOutputs[x].out.overwrite （覆盖），脉冲沿切换 0 → 1 时
- <Nc>.digitalOutputs[x].out.target（规定），信号状态 TRUE 下

必须避免同时激活这两个接口。

数据类型与更新

BOOL，循环

值 TRUE

设定生效。

值 FALSE

设定未生效。

关联

- <Nc>.digitalOutputs[x].out.lock
  将 NC 硬件数字输出端“锁定”在定义状态 0
- <Nc>.digitalOutputs[x].out.overwrite
  脉冲沿切换 0 → 1：将“digitalOutpute[].out.settingState”传递到硬件数字输出端，为 NC 数字输出状态“$A_OUT[]”赋值 0
- <Nc>.digitalOutputs[x].out.settingState
  如果“digitalOutputs[].out.overwrite”或“digitalOutputs[].out.target”处于活动状态，则将值传输至 NC 硬件数字输出端
- MD10360 $MN_FASTIO_DIG_NUM_OUTPUTS

更多信息

- 基础功能功能手册；“数字和模拟 NC 外围设备”>“通过 PLC 进行间接外围设备访问”>“NC 数字输入/输出端”>“NC 输出端”章节

12.3.62 <Nc>.analogInputs[x].in

12.3.62.1 actualValue

NC → PLC	<Nc>.analogInputs[1].in.actualValue
NC 硬件模拟输入端实际值可在 NC 软件模拟输入端“$A_INA[]”使用	

数据类型和更新

INT，循环

关联

- <Nc>.analogInputs[x].out.lock
  如果“analogInputs[].out.target”未生效，则将 NC 软件模拟输入端 $A_INA[]“锁定”为定义值 0
- <Nc>.analogInputs[x].out.target
  将“analogInputs[].out.settingValue”传递到 NC 软件模拟输入端 $A_INA[]
- <Nc>.analogInputs[x].out.settingValue
  如果“analogInput[].out.target”处于活动状态，“analogInputs[].out.settingValue”会传递到 NC 软件模拟输入端 $A_INA[]
- MD10300 $MN_FASTIO_ANA_NUM_INPUTS

更多信息

- 基础功能功能手册；“数字和模拟 NC 外围设备”>“通过 PLC 进行间接外围设备访问”>“NC 模拟输入/输出端”>“NC 输入端”章节

12.3.63 <Nc>.analogInputs[x].out

12.3.63.1 lock

PLC → NC	<Nc>.analogInputs[1].out.lock
如果“analogInputs[].out.target”未生效，则将 NC 软件模拟输入端 $A_INA[]“锁定”为定义值 0	

说明

在读取时，被禁止的输入提供值 0。

数据类型与更新

BOOL，循环

值 TRUE

NC 模拟量输入被禁止。

值 FALSE

NC 模拟量输入被使能。

关联

- <Nc>.analogInputs[x].out.target
  将“analogInputs[].out.settingValue”传递到 NC 软件模拟输入端 $A_INA[]
- <Nc>.analogInputs[x].out.settingValue
  如果“analogInput[].out.target”处于活动状态，“analogInputs[].out.settingValue”会传递到 NC 软件模拟输入端 $A_INA[]
- <Nc>.analogInputs[x].in.actualValue
  NC 硬件模拟输入端实际值可在 NC 软件模拟输入端“$A_INA[]”使用
- MD10300 $MN_FASTIO_ANA_NUM_INPUTS

更多信息

- 基础功能功能手册；“数字和模拟 NC 外围设备”>“通过 PLC 进行间接外围设备访问”>“NC 模拟输入/输出端”>“NC 输入端”章节

12.3.63.2 target

PLC → NC	<Nc>.analogInputs[1].out.target
将“analogInputs[].out.settingValue”传递到 NC 软件模拟输入端 $A_INA[]	

说明

- 若设置值设定被启用，则设置值作为 NC 输入值生效。
- 若设置值设定被关闭，则 NC 输入上的模拟值作为 NC 输入值生效，或者在禁用时，值 0 生效。

数据类型与更新

BOOL，循环

值 TRUE

启用设置值设定。

值 FALSE

关闭设置值设定。

关联

- <Nc>.analogInputs[x].out.lock
  如果“analogInputs[].out.target”未生效，则将 NC 软件模拟输入端 $A_INA[]“锁定”为定义值 0
- <Nc>.analogInputs[x].out.target
  将“analogInputs[].out.settingValue”传递到 NC 软件模拟输入端 $A_INA[]
- <Nc>.analogInputs[x].out.settingValue
  如果“analogInput[].out.target”处于活动状态，“analogInputs[].out.settingValue”会传递到 NC 软件模拟输入端 $A_INA[]
- <Nc>.analogInputs[x].in.actualValue
  NC 硬件模拟输入端实际值可在 NC 软件模拟输入端“$A_INA[]”使用
- MD10300 $MN_FASTIO_ANA_NUM_INPUTS

更多信息

- 基础功能功能手册；“数字和模拟 NC 外围设备”>“通过 PLC 进行间接外围设备访问”>“NC 模拟输入/输出端”>“NC 输入端”章节

12.3.63.3 settingValue

PLC → NC	<Nc>.analogInputs[1].out.settingValue
如果“analogInput[].out.target”处于活动状态，“analogInputs[].out.settingValue”会传递到 NC 软件模拟输入端 $A_INA[]	

说明

- 当为输入启用“规定”<Nc>.analogInputs[x].out.target 时，设置值作为 NC 输入值进行传递。
- 设置值必须作为定点数（16 位，含符号）以补码设定。

数据类型与更新

BOOL，循环

值 TRUE

如果“analogInput[].out.target”处于活动状态，“analogInputs[].out.settingValue”会传递到 NC 软件模拟输入端 $A_INA[]

关联

- <Nc>.analogInputs[x].out.lock
  如果“analogInputs[].out.target”未生效，则将 NC 软件模拟输入端 $A_INA[]“锁定”为定义值 0
- <Nc>.analogInputs[x].out.target
  将“analogInputs[].out.settingValue”传递到 NC 软件模拟输入端 $A_INA[]
- <Nc>.analogInputs[x].out.settingValue
  如果“analogInput[].out.target”处于活动状态，“analogInputs[].out.settingValue”会传递到 NC 软件模拟输入端 $A_INA[]
- <Nc>.analogInputs[x].in.actualValue
  NC 硬件模拟输入端实际值可在 NC 软件模拟输入端“$A_INA[]”使用
- MD10300 $MN_FASTIO_ANA_NUM_INPUTS

更多信息

- 基础功能功能手册；“数字和模拟 NC 外围设备”>“通过 PLC 进行间接外围设备访问”>“NC 模拟输入/输出端”>“NC 输入端”章节

12.3.64 <Nc>.analogOutputs[x].in

12.3.64.1 setpointValue

NC → PLC	<Nc>.analogOutputs[1].in.setpointValue
如果“覆盖”未激活，则通过 NC 为 NC 软件模拟输出端“$A_OUTA[]”赋值	

说明

通过设定值可在 PLC 用户程序中读取当前 NC 输出值。

设定值作为定点数（16 位，含符号）以补码表示。

数据类型和更新

INT

值

如果“覆盖”未激活，则通过 NC 为 NC 软件模拟输出端“$A_OUTA[]”赋值

关联

- <Nc>.analogOutputs[1].out.overwrite
  脉冲沿切换 0 → 1：将“analogOutputs[].out.settingValue”传递到硬件模拟输出端 1，实际 NC 软件值 $A_OUTA[1] 丢失
- <Nc>.analogOutputs[2].out.target
  NC 模拟输出端 1 - 8：规定
- <Nc>.analogOutputs[3].out.lock
  NC 模拟输出端 1 - 8：禁止
- <Nc>.analogOutputs[4].out.settingValue
  NC 模拟输出端 1 - 8：设置值
- MD10310 $MN_FASTIO_ANA_NUM_OUTPUTS

更多信息

- 基础功能功能手册；“数字和模拟 NC 外围设备”>“通过 PLC 进行间接外围设备访问”>“NC 模拟输入/输出端”>“NC 输出端”章节

12.3.65 <Nc>.analogOutputs[x].out

12.3.65.1 settingValue

PLC → NC	<Nc>.analogOutputs[1].out.settingValue
如果“analogOutputs[].out.overwrite”或“analogOutputs[].out.target”处于活动状态，则将“analogOutputs[].out.settingValue”传递到 NC 硬件模拟输出端 1。	

说明

通过设置值可由 PLC 用户程序指定一个输出值。为了使设置值生效，必须通过针对“覆盖”或“设定”的接口将其激活。

说明

变量 <Nc>.analogOutputs[x].out.settingValue（NC 模拟输出端 1 - 8：设置值）由以下变量共用：

- <Nc>.analogOutputs[x].out.overwrite
  （覆盖），脉冲沿切换 0 → 1 时
- <Nc>.analogOutputs[x].out.target
  （规定），信号状态 TRUE 下

必须避免同时激活这两个接口。

数据类型和更新

Int，循环

关联

- <Nc>.analogOutputs[x].out.overwrite
  脉冲沿切换 0 → 1：将“analogOutputs[].out.settingValue”传递到硬件模拟输出端 1，实际 NC 软件值 $A_OUTA[1] 丢失
- <Nc>.analogOutputs[x].out.target
  如果“analogOutputs[].out.lock”未激活，则将“analogOutputs[].out.settingValue”传递到硬件模拟输出端 1，实际 NC 软件模拟输出值 $A_OUTA[1] 保留
- <Nc>.analogOutputs[x].out.lock
  如果“analogOutputs[].out.overwrite”或“analogOutpus[].out.target”处于活动状态，则“锁定”NC 硬件模拟输出端 1。
- <Nc>.analogOutputs[x].in.setpointValue
  如果“覆盖”未激活，则通过 NC 为 NC 软件模拟输出端“$A_OUTA[]”赋值
- MD10310 $MN_FASTIO_ANA_NUM_OUTPUTS

更多信息

- 基础功能功能手册；“数字和模拟 NC 外围设备”>“通过 PLC 进行间接外围设备访问”>“NC 模拟输入/输出端”>“NC 输出端”章节

12.3.65.2 lock

PLC → NC	<Nc>.analogOutputs[1].out.lock
如果“analogOutputs[].out.overwrite”或“analogOutpus[].out.target”处于活动状态，则“锁定”NC 硬件模拟输出端 1。	

说明

- 如果设置了一个位，相应的输出不会使用 NC 输出值，而是使用由 PLC 用户程序指定的设置值。此时保留当前 NC 输出值。
- 如果将一个位复位，则对于相应的输出而言，最后的 NC 输出值重新生效。

数据类型与更新

BOOL，循环

值 TRUE

设定生效。

值 FALSE

设定未生效。

关联

- <Nc>.analogOutputs[x].out.overwrite
  脉冲沿切换 0 → 1：将“analogOutputs[].out.settingValue”传递到硬件模拟输出端 1，实际 NC 软件值 $A_OUTA[1] 丢失
- <Nc>.analogOutputs[x].out.target
  如果“analogOutputs[].out.lock”未激活，则将“analogOutputs[].out.settingValue”传递到硬件模拟输出端 1，实际 NC 软件模拟输出值 $A_OUTA[1] 保留
- <Nc>.analogOutputs[x].out.settingValue
  如果“analogOutputs[].out.overwrite”或“analogOutputs[].out.target”处于活动状态，则将“analogOutputs[].out.settingValue”传递到 NC 硬件模拟输出端 1。
- <Nc>.analogOutputs[x].in.setpointValue
  如果“覆盖”未激活，则通过 NC 为 NC 软件模拟输出端“$A_OUTA[]”赋值
- MD10310 $MN_FASTIO_ANA_NUM_OUTPUTS

更多信息

- 基础功能功能手册；“数字和模拟 NC 外围设备”>“通过 PLC 进行间接外围设备访问”>“NC 模拟输入/输出端”>“NC 输出端”章节

12.3.65.3 overwrite

PLC → NC	<Nc>.analogOutputs[1].out.overwrite
脉冲沿切换 0 → 1：将“analogOutputs[].out.settingValue”传递到硬件模拟输出端 1，实际 NC 软件值 $A_OUTA[1] 丢失	

说明

在上升沿切换 0 → 1 中，相关输出端不使用通过系统变量 $A_OUTA 写入的值，而是使用由 PLC 用户程序指定的设置值。此时，通过系统变量 $A_OUTA 写入的值会丢失。

在下降沿切换 1 → 0 时，则为相应输出保留硬件输出上的当前值。

说明

变量 <Nc>.analogOutputs[x].out.settingValue（NC 模拟输出端 1 - 8：设置值）由以下变量共用：

- <Nc>.analogOutputs[x].out.overwrite（覆盖），脉冲沿切换 0 → 1 时
- <Nc>.analogOutputs[x].out.target（规定），信号状态 TRUE 下

必须避免同时激活这两个接口。

数据类型与更新

BOOL，循环

脉冲沿切换 0 → 1

激活对应的“设置值”。

脉冲沿切换 1 → 0

无作用。

关联

- <Nc>.analogOutputs[x].out.target
  如果"analogOutputs[].out.lock"未激活，则将"analogOutputs[].out.settingValue"传递到硬件模拟输出端 1，实际 NC 软件模拟输出值 $A_OUTA[1] 保留。NC 模拟输出端 1 - 8：规定
- <Nc>.analogOutputs[x].out.lock
  如果"analogOutputs[].out.overwrite"或"analogOutpus[].out.target"处于活动状态，则"锁定"NC 硬件模拟输出端 1。
- <Nc>.analogOutputs[x].out.settingValue
  如果"analogOutputs[].out.overwrite"或"analogOutputs[].out.target"处于活动状态，则将"analogOutputs[].out.settingValue"传递到 NC 硬件模拟输出端 1。
- <Nc>.analogOutputs[1].in.setpointValue
  如果"覆盖"未激活，则通过 NC 为 NC 软件模拟输出端"$A_OUTA[]"赋值
- MD10310 $MN_FASTIO_ANA_NUM_OUTPUTS

更多信息

- 基础功能功能手册；"数字和模拟 NC 外围设备" > "通过 PLC 进行间接外围设备访问" > "NC 模拟输入/输出端" > "NC 输出端"章节

12.3.65.4 target

PLC → NC	<Nc>.analogOutputs[1].out.target
如果"analogOutputs[].out.lock"未激活，则将"analogOutputs[].out.settingValue"传递到硬件模拟输出端 1，实际 NC 软件模拟输出值 $A_OUTA[1] 保留	

说明

- 设置比特位后，相关输出端不使用 NC 输出值，而是使用由 PLC 用户程序指定的设置值。此时保留当前 NC 输出值。
- 将比特位复位后，对于相关输出端来说，最后的 NC 输出值重新生效。

说明

变量 <Nc>.analogOutputs[x].out.settingValue（NC 模拟输出端 1 - 8：设置值）由以下变量共用：

- <Nc>.digitalOutputs[x].out.overwrite（覆盖），脉冲沿切换 0 → 1 时
- <Nc>.digitalOutputs[x].out.target（规定），信号状态 TRUE 下

必须避免同时激活这两个接口。

数据类型与更新

BOOL，循环

值 TRUE

设定生效。

值 FALSE

设定未生效。

关联

- <Nc>.analogOutputs[x].out.overwrite
  脉冲沿切换 0 → 1：将"analogOutputs[].out.settingValue"传递到硬件模拟输出端 1，实际 NC 软件值 $A_OUTA[1] 丢失
- <Nc>.analogOutputs[x].out.lock
  如果"analogOutputs[].out.overwrite"或"analogOutpus[].out.target"处于活动状态，则"锁定"NC 硬件模拟输出端 1。
- <Nc>.analogOutputs[x].out.settingValue
  如果"analogOutputs[].out.overwrite"或"analogOutputs[].out.target"处于活动状态，则将"analogOutputs[].out.settingValue"传递到 NC 硬件模拟输出端 1。
- <Nc>.analogOutputs[x].in.setpointValue
  如果"覆盖"未激活，则通过 NC 为 NC 软件模拟输出端"$A_OUTA[]"赋值
- MD10310 $MN_FASTIO_ANA_NUM_OUTPUTS

更多信息

- 基础功能功能手册；"数字和模拟 NC 外围设备" > "通过 PLC 进行间接外围设备访问" > "NC 模拟输入/输出端" > "NC 输出端"章节

12.3.66 <Nc>.collisionProtAreas.in

12.3.66.1 active

NC → PLC	<Nc>.collisionProtAreas.in.active
所有通过接口信号连接在一起的保护区域都处于活动状态	

说明

通过编程设置系统变量 $NP_BIT_NO，完成保护区域 ↔ 接口信号 (<索引>.<位>) 分配：

$NP_BIT_NO[<保护区域>] = <位编号>

下表介绍保护区域的分配关系：

Index	位							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	7	6	5	4	3	2	1	0
1	15	14	13	12	11	10	9	8
2	23	22	21	20	19	18	17	16
3	31	30	29	28	27	26	25	24
4	39	38	37	36	35	34	33	32
5	47	46	45	44	43	42	41	40
6	55	54	53	52	51	50	49	48
7	63	62	61	60	59	58	57	56

数据类型与更新

ARRAY[0..7] of BYTE，循环

值 TRUE

与接口信号关联的保护区生效。

在有多个保护区与接口信号关联的情况下：所有保护区均生效。

值 FALSE

与接口信号关联的保护区未生效。

在有多个保护区与接口信号关联的情况下：至少一个保护区未生效。

关联

- <Nc>.collisionProtAreas.out.activate
  激活与接口信号关联的保护区域
- 系统变量 $NP_BIT_NO，用于切换的接口位的编号激活

更多信息

- 监控和补偿功能手册；章节：“机床几何建模”>“调试”>“系统变量：保护区域”>“$NP_BIT_NO”
- 监控和补偿功能手册；“防撞保护”章节

12.3.67 <Nc>.collisionProtAreas.out

12.3.67.1 activate

PLC → NC	<Nc>.collisionProtAreas.out.activate
激活与接口信号关联的保护区域	

说明

通过编程设置系统变量 $NP_BIT_NO，完成保护区域 ↔ 接口信号 (<索引>.<位>) 分配：

$NP_BIT_NO[<保护区域>] = <位编号>

下表介绍保护区域的分配关系：

Index	位							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	7	6	5	4	3	2	1	0
1	15	14	13	12	11	10	9	8
2	23	22	21	20	19	18	17	16
3	31	30	29	28	27	26	25	24
4	39	38	37	36	35	34	33	32
5	47	46	45	44	43	42	41	40
6	55	54	53	52	51	50	49	48
7	63	62	61	60	59	58	57	56

数据类型与更新

ARRAY[0..7] of BYTE，循环

值 TRUE

请求激活与接口信号关联的保护区。

值 FALSE

请求取消与接口信号关联的保护区。

关联

- <Nc>.collisionProtAreas.in.active
  所有通过接口信号连接在一起的保护区域都处于活动状态
- 系统变量 $NP_BIT_NO，用于切换的接口位的编号激活

更多信息

- 监控和补偿功能手册；章节：“机床几何建模”>“调试”>“系统变量：保护区域”>“$NP_BIT_NO”
- 监控和补偿功能手册；“防撞保护”章节

12.3.68 <Nc>.collision.out

12.3.68.1 deactivateFixtureAreasAuto

PLC → NC	<Nc>.collision.out.deactivateFixtureAreasAuto
请求在“AUTO”运行方式下操作时，禁用夹具的所有保护区域	

说明

- 运行方式：自动方式
- 保护区域类型 ($NP_PROT_TYPE)：夹具 (FIXTURE)

说明

在 SINUMERIK Operate 操作界面的“自动”、“JOG”或“MDA”>“ETC 键（“>”）”>“设置”>“防撞保护”>“启用和关闭防撞保护”操作区域，将保护区域组的 HMI/PLC 接口信号 <HmiNc>.collision.in.machineAreasAutoDeactivated 置位，可以禁用该保护区域组。

说明

若在 AUTO 运行方式中使能了手动运行（MD10735 位 0 == 1：可在 AUTO 中点动），则在 AUTO 运行方式中手动运行期间，针对 JOG 运行方式的设置亦适用。

数据类型与更新

BOOL，循环

值 TRUE

已请求在选择的运行方式中将该保护区类型的所有保护区取消。

值 FALSE

未请求在选择的运行方式中将该保护区类型的所有保护区取消。

关联

- <HmiNc>.collision.in.machineAreasAutoDeactivated
  HMI：已请求在"AUTO"运行方式下，禁用所有保护区域类型为“机床”的保护区域
- $NP_PROT_TYPE 保护区域类型
- MD10735 $MN_JOG_MODE_MASK JOG 运行方式设置

更多信息

- 操作手册；“防撞保护”章节
- 监控和补偿功能手册；章节：“机床几何建模”>“调试”>“系统变量：保护区”>“$NP_PROT_TYPE”
- 监控和补偿功能手册；“防撞保护”章节

12.3.68.2 deactivateFixtureAreasJog

PLC → NC	<Nc>.collision.out.deactivateFixtureAreasJog
请求在"JOG"运行方式下操作时，禁用所有“夹具”类型的保护区域	

说明

- 运行方式：Jog
- 保护区域类型 ($NP_PROT_TYPE)：夹具 (FIXTURE)

说明

在 SINUMERIK Operate 操作界面的“自动”、"JOG"或"MDA">"ETC 键（">"）">“设置”>“防撞保护”>“启用和关闭防撞保护”操作区域，将保护区域组的 HMI/PLC 接口信号 <HmiNc>.collision.in.machineAreasAutoDeactivated 置位，可以禁用该保护区域组。

说明

若在 AUTO 运行方式中使能了手动运行（MD10735 位 0 == 1：可在 AUTO 中点动），则在 AUTO 运行方式中手动运行期间，针对 JOG 运行方式的设置亦适用。

数据类型与更新

BOOL，循环

值 TRUE

已请求在选择的运行方式中将该保护区类型的所有保护区取消。

值 FALSE

未请求在选择的运行方式中将该保护区类型的所有保护区取消。

关联

- <HmiNc>.collision.in.machineAreasAutoDeactivated
  HMI：已请求在"AUTO"运行方式下，禁用所有保护区域类型为“机床”的保护区域
- $NP_PROT_TYPE 保护区域类型
- MD10735 $MN_JOG_MODE_MASK JOG 运行方式设置

更多信息

- 操作手册；“防撞保护”章节
- 监控和补偿功能手册；章节：“机床几何建模”>“调试”>“系统变量：保护区”>“$NP_PROT_TYPE”
- 监控和补偿功能手册；“防撞保护”章节

12.3.68.3 deactivateMachineAreasAuto

PLC → NC	<Nc>.collision.out.deactivateMachineAreasAuto
请求在"AUTO"运行方式下操作时，禁用所有“机床”类型的保护区域	

说明

- 运行方式：自动方式
- 保护区域类型 ($NP_PROT_TYPE)：机床 (MACHINE)

说明

在 SINUMERIK Operate 操作界面的“自动”、"JOG"或"MDA">"ETC 键（">"）">"设置">"防撞保护">"启用和关闭防撞保护”操作区域，将保护区域组的下列 HMI/PLC 接口信号置位，可以禁用该保护区域组：

- <HmiNc>.collision.in.machineAreasAutoDeactivated
- <HmiNc>.collision.in.toolAreasAutoDeactivated
- <HmiNc>.collision.in.fixtureAreasAutoDeactivated
- <HmiNc>.collision.in.workpieceAreasAutoDeactivated
- <HmiNc>.collision.in.machineAreasJogDeactivated
- <HmiNc>.collision.in.toolAreasJogDeactivated
- <HmiNc>.collision.in.fixtureAreasJogDeactivated
- <HmiNc>.collision.in.workpieceAreasJogDeactivated

说明

若在 AUTO 运行方式中使能了手动运行（MD10735 位 0 == 1：可在 AUTO 中点动），则在 AUTO 运行方式中手动运行期间，针对 JOG 运行方式的设置亦适用。

数据类型与更新

BOOL，循环

值 TRUE

已请求在选择的运行方式中将该保护区类型的所有保护区取消。

值 FALSE

未请求在选择的运行方式中将该保护区类型的所有保护区取消。

关联

- <HmiNc>.collision.in.machineAreasAutoDeactivated
  HMI：已请求在"AUTO"运行方式下，禁用所有保护区域类型为“机床”的保护区域
- <HmiNc>.collision.in.toolAreasAutoDeactivated
  HMI：已请求在"AUTO"运行方式下，禁用所有保护区域类型为“刀具”的保护区域
- <HmiNc>.collision.in.fixtureAreasAutoDeactivated
  HMI：已请求在"AUTO"运行方式下，禁用所有保护区域类型为“夹具”的保护区域
- <HmiNc>.collision.in.workpieceAreasAutoDeactivated
  HMI：已请求在"AUTO"运行方式下，禁用所有保护区域类型为“工件”的保护区域
- <HmiNc>.collision.in.machineAreasJogDeactivated
  HMI：已请求在"JOG"运行方式下，禁用所有保护区域类型为“机床”的保护区域
- <HmiNc>.collision.in.toolAreasJogDeactivated
  HMI：已请求在"JOG"运行方式下，禁用所有保护区域类型为“刀具”的保护区域
- <HmiNc>.collision.in.fixtureAreasJogDeactivated
  HMI：已请求在"JOG"运行方式下，禁用所有保护区域类型为“夹具”的保护区域
- <HmiNc>.collision.in.workpieceAreasJogDeactivated
  HMI：已请求在"JOG"运行方式下，禁用所有保护区域类型为“工件”的保护区域
- $NP_PROT_TYPE 保护区域类型
- MD10735 $MN_JOG_MODE_MASK JOG 运行方式设置

更多信息

- 操作手册；“防撞保护”章节
- 监控和补偿功能手册；章节：“机床几何建模”>“调试”>“系统变量：保护区”>“$NP_PROT_TYPE”
- 监控和补偿功能手册；“防撞保护”章节

12.3.68.4 deactivateMachineAreasJog

PLC → NC	<Nc>.collision.out.deactivateMachineAreasJog
请求在“JOG”运行方式下操作时，禁用所有“机床”类型的保护区域	

说明

碰撞监测：取消保护区组

- 运行方式：Jog
- 保护区域类型 ($NP_PROT_TYPE)：机床 (MACHINE)

说明

在 SINUMERIK Operate 操作界面的“自动”、“JOG”或“MDA”>“ETC 键（“>”）”>“设置”>“防撞保护”>“启用和关闭防撞保护”操作区域，将保护区域组的 HMI/PLC 接口信号 <HmiNc>.collision.in.machineAreasAutoDeactivated 置位，可以禁用该保护区域组。

说明

若在 AUTO 运行方式中使能了手动运行（MD10735 位 0 == 1：可在 AUTO 中点动），则在 AUTO 运行方式中手动运行期间，针对 JOG 运行方式的设置亦适用。

数据类型与更新

BOOL，循环

值 TRUE

已请求在选择的运行方式中将该保护区类型的所有保护区取消。

值 FALSE

未请求在选择的运行方式中将该保护区类型的所有保护区取消。

关联

- <HmiNc>.collision.in.machineAreasAutoDeactivated
  HMI：已请求在"AUTO"运行方式下，禁用所有保护区域类型为“机床”的保护区域
- $NP_PROT_TYPE 保护区域类型
- MD10735 $MN_JOG_MODE_MASK JOG 运行方式设置

更多信息

- 操作手册；“防撞保护”章节
- 监控和补偿功能手册；章节：“机床几何建模”>“调试”>“系统变量：保护区”>“$NP_PROT_TYPE”
- 监控和补偿功能手册；“防撞保护”章节

12.3.68.5 deactivateToolAreasAuto

PLC → NC	<Nc>.collision.out.deactivateToolAreasAuto
请求在"AUTO"运行方式下操作时，禁用所有“刀具”类型的保护区域	

说明

- 运行方式：自动方式
- 保护区域类型 ($NP_PROT_TYPE)：刀具 (TOOL)

说明

在 SINUMERIK Operate 操作界面的“自动”、"JOG"或"MDA">"ETC 键（">"）">“设置”>“防撞保护”>“启用和关闭防撞保护”操作区域，将保护区域组的 HMI/PLC 接口信号 <HmiNc>.collision.in.machineAreasAutoDeactivated 置位，可以禁用该保护区域组。

说明

若在 AUTO 运行方式中使能了手动运行（MD10735 位 0 == 1：可在 AUTO 中点动），则在 AUTO 运行方式中手动运行期间，针对 JOG 运行方式的设置亦适用。

数据类型与更新

BOOL，循环

值 TRUE

已请求在选择的运行方式中将该保护区类型的所有保护区取消。

值 FALSE

未请求在选择的运行方式中将该保护区类型的所有保护区取消。

关联

- <HmiNc>.collision.in.machineAreasAutoDeactivated
  HMI：已请求在"AUTO"运行方式下，禁用所有保护区域类型为“机床”的保护区域
- $NP_PROT_TYPE 保护区域类型
- MD10735 $MN_JOG_MODE_MASK JOG 运行方式设置

更多信息

- 操作手册；“防撞保护”章节
- 监控和补偿功能手册；章节：“机床几何建模”>“调试”>“系统变量：保护区”>“$NP_PROT_TYPE”
- 监控和补偿功能手册；“防撞保护”章节

12.3.68.6 deactivateToolAreasJog

PLC → NC	<Nc>.collision.out.deactivateToolAreasJog
请求在"JOG"运行方式下操作时，禁用所有“刀具”类型的保护区域	

说明

- 运行方式：Jog
- 保护区域类型 ($NP_PROT_TYPE)：刀具 (TOOL)

说明

在 SINUMERIK Operate 操作界面的“自动”、"JOG"或"MDA">"ETC 键（">"）">“设置”>"防撞保护">“启用和关闭防撞保护”操作区域，将保护区域组的 HMI/PLC 接口信号 <HmiNc>.collision.in.machineAreasAutoDeactivated 置位，可以禁用该保护区域组。

说明

若在 AUTO 运行方式中使能了手动运行（MD10735 位 0 == 1：可在 AUTO 中点动），则在 AUTO 运行方式中手动运行期间，针对 JOG 运行方式的设置亦适用。

数据类型与更新

BOOL，循环

值 TRUE

已请求在选择的运行方式中将该保护区类型的所有保护区取消。

值 FALSE

未请求在选择的运行方式中将该保护区类型的所有保护区取消。

关联

- <HmiNc>.collision.in.machineAreasAutoDeactivated
  HMI：已请求在"AUTO"运行方式下，禁用所有保护区域类型为“机床”的保护区域
- $NP_PROT_TYPE 保护区域类型
- MD10735 $MN_JOG_MODE_MASK JOG 运行方式设置

更多信息

- 操作手册；“防撞保护”章节
- 监控和补偿功能手册；章节：“机床几何建模”>“调试”>“系统变量：保护区”>“$NP_PROT_TYPE”
- 监控和补偿功能手册；“防撞保护”章节

12.3.68.7 deactivateWorkpieceAreasAuto

PLC → NC	<Nc>.collision.out.deactivateWorkpieceAreasAuto
请求在"AUTO"运行方式下操作时，禁用所有“工件”类型的保护区域	

说明

- 运行方式：自动方式
- 保护区域类型 ($NP_PROT_TYPE)：工件 (WORKPIECE)

说明

在 SINUMERIK Operate 操作界面的“自动”、"JOG"或"MDA">"ETC 键（">"）">"设置">"防撞保护">"启用和关闭防撞保护”操作区域，将保护区域组的 HMI/PLC 接口信号 <HmiNc>.collision.in.machineAreasAutoDeactivated 置位，可以禁用该保护区域组。

说明

若在 AUTO 运行方式中使能了手动运行（MD10735 位 0 == 1：可在 AUTO 中点动），则在 AUTO 运行方式中手动运行期间，针对 JOG 运行方式的设置亦适用。

数据类型与更新

BOOL，循环

值 TRUE

已请求在选择的运行方式中将该保护区类型的所有保护区取消。

值 FALSE

未请求在选择的运行方式中将该保护区类型的所有保护区取消。

关联

- <HmiNc>.collision.in.machineAreasAutoDeactivated
  HMI：已请求在"AUTO"运行方式下，禁用所有保护区域类型为“机床”的保护区域
- $NP_PROT_TYPE 保护区域类型
- MD10735 $MN_JOG_MODE_MASK JOG 运行方式设置

更多信息

- 操作手册；“防撞保护”章节
- 监控和补偿功能手册；章节：“机床几何建模”>“调试”>“系统变量：保护区”>“$NP_PROT_TYPE”
- 监控和补偿功能手册；“防撞保护”章节

12.3.68.8 deactivateWorkpieceAreasJog

PLC → NC	<Nc>.collision.out.deactivateWorkpieceAreasJog
请求在"JOG"运行方式下操作时，禁用所有“工件”类型的保护区域	

说明

- 运行方式：Jog
- 保护区域类型 ($NP_PROT_TYPE)：工件 (WORKPIECE)

说明

在 SINUMERIK Operate 操作界面的“自动”、"JOG"或"MDA">"ETC 键（">"）">"设置">"防撞保护">“启用和关闭防撞保护”操作区域，将保护区域组的 HMI/PLC 接口信号 <HmiNc>.collision.in.machineAreasAutoDeactivated 置位，可以禁用该保护区域组。

说明

若在 AUTO 运行方式中使能了手动运行（MD10735 位 0 == 1：可在 AUTO 中点动），则在 AUTO 运行方式中手动运行期间，针对 JOG 运行方式的设置亦适用。

数据类型与更新

BOOL，循环

值 TRUE

已请求在选择的运行方式中将该保护区类型的所有保护区取消。

值 FALSE

未请求在选择的运行方式中将该保护区类型的所有保护区取消。

关联

- <HmiNc>.collision.in.machineAreasAutoDeactivated
  HMI：已请求在"AUTO"运行方式下，禁用所有保护区域类型为“机床”的保护区域
- $NP_PROT_TYPE 保护区域类型
- MD10735 $MN_JOG_MODE_MASK JOG 运行方式设置

更多信息

- 操作手册；“防撞保护”章节
- 监控和补偿功能手册；章节：“机床几何建模”>“调试”>“系统变量：保护区”>“$NP_PROT_TYPE”
- 监控和补偿功能手册；“防撞保护”章节

12.3.69 NCK_typeHmi

下表说明了数据类型 NCK_typeHmi 的含义：

说明

创建"NCK_typeHmi"数据类型的"LSINU_Hmi"数据块时，必须设置“可从 HMI/OPC UA/Web API 到达”和“可从 HMI/OPC UA/Web API 写入”两种属性。

结构	说明
.config	HMI 的组态数据
.registered	HMI 已注册
.basic.in (页 959)	来自 HMI 的通用信号
.basic.inOut (页 965)	来自和发送至 HMI 的一般信号
.basic.out (页 966)	发送至 HMI 的通用信号
.plcFunc.in (页 973)	来自 HMI 的 PLC 功能输入信号
.plcFunc.out (页 974)	发送至 HMI 的 PLC 功能输出信号
.progSelection.in (页 976)	来自 HMI 的程序选择信号
.progSelection.out (页 982)	程序选择发送至 HMI 的信号
.mappingTable.in (页 985)	映射表输入信号
.mappingTable.out (页 986)	映射表输出数据
.userKeys.in (页 987)	来自 HMI 的用户自定义键信号
.userKeys.out (页 987)	用户自定义键发送至 HMI 的信号

12.3.70 <Hmi>.basic.in

12.3.70.1 activeOperatingAreaNumber

OP → PLC	<Hmi>.basic.in.activeOperatingAreaNumber
活动操作区域的编号	

数据类型与更新

USInt，循环

值

当前操作区域（201 - 206）

关联

- MD9032 $MM_HMI_MONITOR = "String"，其中 "String" = "DB<DB-Nummer>.DBB<Byte-Adresse>"

12.3.70.2 cancelKeyPressed

OP → PLC	<Hmi>.basic.in.cancelKeyPressed
已按下“取消警报”按键（必须通过用户程序复位）	

数据类型与更新

BOOL，循环

值 TRUE

按下取消键

值 FALSE

接口信号必须在 PLC 用户程序中复位。

12.3.70.3 currentChanNumber

OP → PLC	<Hmi>.basic.in.currentChanNumber
HMI 中显示的当前通道编号	

数据类型与更新

USInt

值

有效通道编号：1 - NC 的最大通道数

12.3.70.4 currentScreenNumber

OP → PLC	<Hmi>.basic.in.currentScreenNumber
当前所选画面编号	

说明

显示当前画面编号

HMI 监视器处于活动状态时，当前画面编号不再显示在此变量中，

而是显示在 HMI 监视器的用户自定义配置区域内。

数据类型与更新

UInt

值 > 0

画面编号

更多信息

- PLC 功能手册

12.3.70.5 dataTransferActive

OP → PLC	<Hmi>.basic.in.dataTransferActive
数据传输处于活动状态	

数据类型与更新

BOOL

值 TRUE

数据传输处于活动状态

值 FALSE

数据传输未激活

12.3.70.6 directKeyAddress

OP → PLC	<Hmi>.basic.in.directKeyAddress
直接键地址	

数据类型与更新

USInt

12.3.70.7 inputDeviceActive

OP → PLC	<Hmi>.basic.in.inputDeviceActive
输入设备处于活动状态，比如使用键盘等	

数据类型与更新

BOOL

值 TRUE

输入设备处于活动状态，比如使用键盘等

值 FALSE

输入设备未激活

12.3.70.8 mcpAddress

OP → PLC	<hmi>.basic.in.mcpAddress
MCP 地址	

数据类型和更新

USInt

值

MCP 地址

12.3.70.9 mcsWcsSwitched

OP → PLC	<Hmi>.basic.in.mcsWcsSwitched
脉冲沿切换 0 → 1：切换机床坐标系和工件坐标系	

数据类型和更新

BOOL，循环

值 TRUE

请求将操作界面上显示轴专用位置和剩余行程的坐标系，

从工件坐标系 (WCS) 切换至机床坐标系 (MCS)，

或从 MCS 切换至 WCS。

信号只存在一个 PLC 循环的时间。

值 FALSE

无作用

关联

- <Hmi>.basic.out.displayWcsValues
  在工件坐标系中显示实际值（0=机床坐标系，1=工件坐标系）

12.3.70.10 screenChangeActive

OP → PLC	<Hmi>.basic.in.screenChangeActive
画面切换处于活动状态	

数据类型与更新

BOOL

值 TRUE

画面切换处于活动状态

值 FALSE

画面切换未激活

12.3.70.11 screenDark

OP → PLC	<Hmi>.basic.in.screenDark
屏幕已切换至待机模式	

数据类型和更新

BOOL，循环

值 TRUE

屏幕已切换至待机模式。

值 FALSE

屏幕未切换至待机模式。

关联

- <Hmi>.basic.out.darkenScreen
  屏幕变暗
- MD9006 $MM_DISPLAY_SWITCH_OFF_INTERVAL
  屏幕变暗时间

12.3.70.12 simActive

OP → PLC	<Hmi>.basic.in.simActive
模拟窗口处于活动状态	

数据类型与更新

BOOL，循环

值 TRUE

在操作界面上选中一个 NC 程序并为其激活“模拟”功能。

值 FALSE

“模拟”功能未激活。

更多信息

- 操作手册；章节“模拟加工”

12.3.70.13 switchoverTcuDisabled

OP → PLC	<Hmi>.basic.in.switchoverTcuDisabled
TCU 切换锁定	

数据类型与更新

BOOL

值 TRUE

TCU 切换锁定

值 FALSE

TCU 切换未锁定

12.3.70.14 tcuIndex

OP → PLC	<Hmi>.basic.in.tcuIndex
TCU 索引	

数据类型与更新

USInt

12.3.71 <hmi>.basic.inOut

12.3.71.1 cancelAlarmsCleared

OP → PLC	<Hmi>.basic.inOut.cancelAlarmsCleared
已删除取消型警报（必须通过用户程序复位）	

数据类型与更新

BOOL，循环

值 TRUE

反馈信息：Cancel 型报警已删除

值 FALSE

提示

接口信号必须在 PLC 用户程序中复位。

关联

- <Hmi>.basic.out.clearCancelAlarms
  删除取消型警报

12.3.71.2 plcHardkeySelection

PLC → OP	<Hmi>.basic.inOut.plcHardkeySelection
为 PLC 硬键选择键值 1 - 255（PLC 选择，HMI 应答）	

数据类型与更新

USInt

值 0

不选择

值 1 - 255

选择相应的程序范围

12.3.71.3 recallAlarmsCleared

OP → PLC	<Hmi>.basic.inOut.recallAlarmsCleared
已删除暂停结束型警报（必须通过用户程序复位）	

数据类型与更新

BOOL，循环

值 TRUE

反馈信息：Recall 型报警已删除

值 FALSE

提示

接口信号必须在 PLC 用户程序中复位。

关联

- <Hmi>.basic.out.clearRecallAlarms
  请求应答待处理的暂停结束型警报

12.3.72 <Hmi>.basic.out

12.3.72.1 analogSpindle1Percentage

PLC → OP	<Hmi>.basic.out.analogSpindle1Percentage
模拟主轴 1：利用率 (%)	

数据类型与更新

USInt

值

模拟主轴 1：利用率 (%)

12.3.72.2 analogSpindle2Percentage

PLC → OP	<Hmi>.basic.out.analogSpindle2Percentage
模拟主轴 2：利用率 (%)	

数据类型与更新

USInt

值

模拟主轴 2：利用率 (%)

12.3.72.3 brightenScreen

PLC → OP	<Hmi>.basic.out.brightenScreen
忽略"basic.out.darkenScreen"，定时器控制仍有效 (MD9006 $MM_DISPLAY_SWITCH_OFF_INTERVAL)	

说明

屏幕变亮

数据类型与更新

BOOL

值 TRUE

显示屏幕内容。屏幕变暗设置被忽略。

值 FALSE

屏幕变暗设置生效。

关联

- <Hmi>.basic.out.darkenScreen
  画面变暗。内容不可见，无法操作按键重新激活.

12.3.72.4 clearCancelAlarms

PLC → OP	<Hmi>.basic.out.clearCancelAlarms
请求应答待处理的取消型警报	

数据类型与更新

BOOL

值 TRUE

已在操作面板上操作故障删除键 ⇒应答 NC 和操作面板的所有取消型警报。

值 FALSE

未按下操作面板上的故障删除键。

关联

- <Hmi>.basic.in.cancelAlarmsCleared
  已删除取消型警报（必须通过用户程序复位）

12.3.72.5 clearRecallAlarms

PLC → OP	<Hmi>.basic.out.clearRecallAlarms
请求应答待处理的暂停结束型警报	

数据类型与更新

BOOL

值 TRUE

已按下操作面板上的故障删除键。

对 NC 和操作面板的所有 Recall 型报警请求进行应答。

值 FALSE

未按下操作面板上的故障删除键。

关联

- <Hmi>.basic.in.recallAlarmsCleared
  已删除暂停结束型警报（必须通过用户程序复位）

12.3.72.6 darkenScreen

PLC → OP	<Hmi>.basic.out.darkenScreen
画面变暗。内容不可见，无法操作按键重新激活	

说明

屏幕变暗

数据类型与更新

BOOL

值 TRUE

屏幕变暗设置启用。

屏幕的自动明/暗控制因此失效。在按下键盘时，屏幕不会自动变亮。

提示

在接口信号置位且屏幕变暗设置启用时，前置操作面板的键盘仍保持生效。因此还建议将前置操作面板的键盘禁用：

<Hmi>.basic.out.disableKeys = TRUE（按键锁定）

值 FALSE

屏幕变亮设置启用。

在此状态下，由控制系统自动进行屏幕明/暗控制：

- 如果在以下机床数据设定的时间内未操作键盘，那么屏幕就会变暗：
  MD9006 $MM_DISPLAY_SWITCH_OFF_INTERVAL（屏幕变暗时间）
- 只要在前置操作面板上操作按键，屏幕就会变亮。

关联

- <Hmi>.basic.out.brightenScreen
  忽略 basic.out.darkenScreen，定时器控制仍有效 (MD9006 $MM_DISPLAY_SWITCH_OFF_INTERVAL)
- <Hmi>.basic.out.disableKeys
  锁定用户按键输入
- MD9006 $MM_DISPLAY_SWITCH_OFF_INTERVAL
  屏幕变暗时间

12.3.72.7 disableKeys

PLC → OP	<Hmi>.basic.out.disableKeys
锁定用户按键输入	

提示

屏幕变暗 <Hmi>.basic.out.darkenScreen = TRUE 时，前置操作面板的键盘仍保持生效。因此还建议将前置操作面板的键盘禁用：

<Hmi>.basic.out.disableKeys = TRUE（按键锁定）

数据类型与更新

BOOL

值 TRUE

键盘禁用，操作人员无法操作键盘。

值 FALSE

键盘使能，操作人员可操作键盘。

关联

- <Hmi>.basic.out.brightenScreen
  屏幕变亮
- <Hmi>.basic.out.darkenScreen
  屏幕变暗

12.3.72.8 disableTeachInTransfer

PLC → OP	<Hmi>.basic.out.disableTeachInTransfer
TEACH IN 传输锁定	

数据类型与更新

BOOL

值 TRUE

TEACH IN 传输锁定

值 FALSE

TEACH IN 传输使能

12.3.72.9 displayWcsValues

PLC → OP	<Hmi>.basic.out.displayWcsValues
在工件坐标系中显示实际值 0=机床坐标系 1=工件坐标系	

数据类型和更新

BOOL

值 TRUE

如果选择了“加工”操作区，系统会在工件坐标系 (WCS) 中显示轴位置和剩余行程。

值 FALSE

如果选择了“加工”操作区，系统会在之前生效的坐标系中显示轴位置和剩余行程。

关联

- <Hmi>.basic.in.mcsWcsSwitched
  脉冲沿切换 0 → 1：切换机床坐标系和工件坐标系

12.3.72.10 permitRemoteMonitoringOnly

PLC → OP	<Hmi>.basic.out.permitRemoteMonitoringOnly
外部查看器：远程访问时只能查看	

说明

外部浏览器的权限

数据类型与更新

BOOL

值 TRUE

远程访问时只能查看

12.3.72.11 rejectRemoteAccess

PLC → OP	<Hmi>.basic.out.rejectRemoteAccess
外部查看器：无远程访问	

数据类型与更新

BOOL

值 TRUE

远程访问

值 FALSE

无远程访问

12.3.72.12 saveActionLog

PLC → OP	<Hmi>.basic.out.saveActionLog
脉冲沿切换 0 → 1：在 CRASH.COM 之后保存 ACTION.COM	

数据类型和更新

BOOL

脉冲沿切换 0 → 1

将 ACTION.LOG 保存到 CRASH.LOG

激活运行记录仪后，即激活记录后，会生成文件 ACTION.COM。文件 ACTION.COM 是二进制文件，作为环形缓冲区不断被覆盖。最新结果位于文件上方开头部分。

存在空的二进制文件 CRASH.COM。ACTION.COM 日志的中间状态保存在 CRASH.COM 文件中。置位接口信号 <Hmi>.basic.out.saveActionLog 或通过报警触发写入触发器，可以触发写入。内容保持至下一次触发。

12.3.73 <Hmi>.plcFunc.in

12.3.73.1 busy

OP → PLC	<Hmi>.plcFunc.in.busy
执行功能时处于活动状态	

数据类型与更新

BOOL

值 TRUE

有效

值 FALSE

未激活

12.3.73.2 errorCode

OP → PLC	<Hmi>.plcFunc.in.errorCode
功能选择错误代码	

说明

功能选择号的错误代码

针对在功能选择下从“LBP_HMI.A_FuncNo”(DB19 DBB32, 82) 请求的功能，

显示错误识别码。功能执行完毕后，通过 NC 输出以下

错误代码：

- 0：无错误
- 1：功能编号无效 <Hmi>.plcFunc.out.selection
- 2:参数无效 <Hmi>.plcFunc.out.parameter1
- 3：HMI 内部变量的写入异常中断
- 10:通道不存在 <Hmi>.plcFunc.out.parameter1

数据类型和更新

USInt

关联

- <Hmi>.plcFunc.out.selection
  功能通道选择：1
- <Hmi>.plcFunc.out.requestStrobe
  功能请求 PLC → HMI：0 → 1；功能已执行 HMI → PLC：1 → 0
- <Hmi>.plcFunc.out.parameter1 (Parameter 1 - 3)
  参数 1：通道编号

12.3.74 <Hmi>.plcFunc.out

12.3.74.1 parameter1

PLC → OP	<Hmi>.plcFunc.out.parameter1
参数 1：通道编号	

说明

功能选择号的参数 1 至 3

针对在功能选择下从 <Hmi>.plcFunc.out.selection 请求的功能，可以规定参数：

参数 1：<Hmi>.plcFunc.out.parameter1

参数的数量和值域取决于各个功能。

数据类型与更新

USInt

关联

- <Hmi>.plcFunc.out.selection
  功能通道选择：1
- <Hmi>.plcFunc.in.errorCode
  功能选择错误代码

更多信息

- PLC 功能手册

12.3.74.2 selection

PLC → OP	<Hmi>.plcFunc.out.selection
功能通道选择：1	

说明

通过 PLC 选择功能

- 功能编号（Bit 0 至 Bit 5）
  在 Bit 0 至 Bit 5，指定在 Bit 6 设置请求选通脉冲后执行的功能编号。
  - 功能编号 0：
    无功能
  - 功能编号 1：
    通道选择
- 请求脉冲（位 6）
  - 值 FALSE：
    功能已由 NC 执行或未请求执行功能
  - 值 TRUE：
    已请求由 NC 执行设置的功能
- 占用功能（位 7）
  - 值 FALSE：
    请求的功能执行完成。
  - 值 TRUE：
    请求的功能当前由 NC 执行。

数据类型和更新

USInt

关联

- <Hmi>.plcFunc.out.parameter1
  参数 1：通道编号
- <Hmi>.plcFunc.in.errorCode
  功能选择错误代码

更多信息

- PLC 功能手册

12.3.74.3 requestStrobe

PLC → OP	<Hmi>.plcFunc.out.requestStrobe
功能请求 PLC → HMI：0 → 1 执行功能 HMI → PLC：1 → 0	

数据类型和更新

BOOL，循环

12.3.75 <Hmi>.progSelection.in

12.3.75.1 completed

OP → PLC	<Hmi>.progSelection.in.Hmi
任务结束	

说明

通过 PLC 选择程序：任务结束

数据类型与更新

BOOL

值 TRUE

任务结束

值 FALSE

任务未结束

关联

- <Hmi>.progSelection.out.request
  选择规定的 NC 程序（开始任务）
- <Hmi>.progSelection.out.progList
  程序列表编号

- <Hmi>.progSelection.out.progNumber
  程序编号选择基于所选程序列表中包含的程序
- <Hmi>.progSelection.in.completed
  任务结束
- <Hmi>.progSelection.in.error
  程序选择出错，参见"errorCode"中的错误识别码
- <Hmi>.progSelection.in.selected
  已选程序（任务尚未完成）
- <Hmi>.progSelection.in.identified
  识别到程序选择
- <Hmi>.progSelection.in.errorCode
  错误识别码
- MD9106 $MM_SERVE_EXTCALL_PROGRAMSEXTCALL-Aufrufe bearbeiten
- MD51041 $MN_ENABLE_PROGLIST_USERAktivierung PLC-Programmliste Bereich USER
- MD51043 $MN_ENABLE_PROGLIST_MANUFACTAktivierung PLC-Programmliste Bereich MANUFACTURER

更多信息

- PLC 功能手册

12.3.75.2 identified

OP → PLC	<Hmi>.progSelection.in.identified
识别到程序选择	

说明

通过 PLC 选择程序：选择

数据类型与更新

BOOL

值 TRUE

选择生效

值 FALSE

选择未生效

关联

- <Hmi>.progSelection.out
  选择规定的 NC 程序（开始任务）
- <Hmi>.progSelection.out.progList
  程序列表编号
  <Hmi>.progSelection.out.progNumber
  程序编号选择基于所选程序列表中包含的程序
- <Hmi>.progSelection.in.completed
  任务结束
- <Hmi>.progSelection.in.error
  程序选择出错，参见“errorCode”中的错误识别码
- <Hmi>.progSelection.in.selected
  已选程序（任务尚未完成）
- <Hmi>.progSelection.in.identified
  识别到程序选择
- <Hmi>.progSelection.in.errorCode
  错误识别码
- MD9106 $MM_SERVE_EXTCALL_PROGRAMSEXTCALL-Aufrufe bearbeiten•
- MD51041 $MN_ENABLE_PROGLIST_USERAktivierung PLC-Programmliste Bereich USER•
- MD51043 $MN_ENABLE_PROGLIST_MANUFACTAktivierung PLC-Programmliste Bereich MANUFACTURER

更多信息

- PLC 功能手册

12.3.75.3 error

OP → PLC	<Hmi>.progSelection.in.error
程序选择出错，参见“errorCode”中的错误识别码	

说明

通过 PLC 选择程序：错误

数据类型与更新

BOOL

值 TRUE

任务错误结束

值 FALSE

任务正确结束

关联

- <Hmi>.progSelection.out.request
  选择规定的 NC 程序（开始任务）
- <Hmi>.progSelection.out.progList
  程序列表编号
- <Hmi>.progSelection.out.progNumber
  程序编号选择基于所选程序列表中包含的程序
- <Hmi>.progSelection.in.completed
  任务结束
- <Hmi>.progSelection.in.error
  程序选择出错，参见“errorCode”中的错误识别码
- <Hmi>.progSelection.in.selected
  已选程序（任务尚未完成）
- <Hmi>.progSelection.in.identified
  识别到程序选择
- <Hmi>.progSelection.in.errorCode
  错误识别码
- MD9106 $MM_SERVE_EXTCALL_PROGRAMSEXTCALL-Aufrufe bearbeiten
- MD51041 $MN_ENABLE_PROGLIST_USERAktivierung PLC-Programmliste Bereich USER
- MD51043 $MN_ENABLE_PROGLIST_MANUFACTAktivierung PLC-Programmliste Bereich MANUFACTURER

更多信息

- PLC 功能手册

12.3.75.4 errorCode

OP → PLC	<Hmi>.progSelection.in.errorCode
错误识别码	

说明

通过 PLC 选择程序：错误识别码

输出字节，指出文件传输的错误值

数据类型与更新

USInt

值

0 - 无错误
1 - 无效的程序列表编号 <Hmi>.progSelection.out.progList
3 - 未找到用户自定义程序列表 plc_proglist_main.ppl（仅限 <Hmi>.progSelection.out.progList ≠ 129、131）
4 - 无效的程序编号 <Hmi>.progSelection.out.progNumber
5 - 未能打开所选工件的任务表。
6 - 任务表中出错（任务表解释器报错）
7 - 任务表解释器报告空任务表

关联

- <Hmi>.progSelection.out.request
  选择规定的 NC 程序（开始任务）
- <Hmi>.progSelection.out.progList
  程序列表编号
- <Hmi>.progSelection.out.progNumber
  程序编号选择基于所选程序列表中包含的程序
- <Hmi>.progSelection.in.completed
  任务结束
- <Hmi>.progSelection.in.error
  程序选择出错，参见"errorCode"中的错误识别码
- <Hmi>.progSelection.in.selected
  已选程序（任务尚未完成）
- <Hmi>.progSelection.in.identified
  识别到程序选择

- MD9106 $MM_SERVE_EXTCALL_PROGRAMSEXTCALL-Aufrufe bearbeiten
- MD51041 $MN_ENABLE_PROGLIST_USERAktivierung PLC-Programmliste Bereich USER
- MD51043 $MN_ENABLE_PROGLIST_MANUFACTAktivierung PLC-Programmliste Bereich MANUFACTURER

12.3.75.5 selected

OP → PLC	<Hmi>.progSelection.in.selected
已选程序（任务尚未完成）	

数据类型与更新

BOOL

值 TRUE

通过 PLC 选择程序：生效

值 FALSE

通过 PLC 选择程序：未生效

关联

- <Hmi>.progSelection.out.request
  选择规定的 NC 程序（开始任务）
- <Hmi>.progSelection.out.progList
  程序列表编号
- <Hmi>.progSelection.out.progNumber
  程序编号选择基于所选程序列表中包含的程序
- <Hmi>.progSelection.in.completed
  任务结束
- <Hmi>.progSelection.in.error
  程序选择出错，参见“errorCode”中的错误识别码
- <Hmi>.progSelection.in.selected
  已选程序（任务尚未完成）
- <Hmi>.progSelection.in.identified
  识别到程序选择

- <Hmi>.progSelection.in.errorCode
  错误识别码
- MD9106 $MM_SERVE_EXTCALL_PROGRAMSEXTCALL-Aufrufe bearbeiten
- MD51041 $MN_ENABLE_PROGLIST_USERAktivierung PLC-Programmliste Bereich USER
- MD51043 $MN_ENABLE_PROGLIST_MANUFACTAktivierung PLC-Programmliste Bereich MANUFACTURER

更多信息

- PLC 功能手册

12.3.76 <Hmi>.progSelection.out

12.3.76.1 progList

PLC → OP	<Hmi>.progSelection.out.progList
程序列表编号	

说明

通过 PLC 选择程序：程序列表的下标

借助 PLC/HMI 接口可选择 PLC 程序列表中预设的“程序/工件”，以通过 NC 执行。

为了设定程序列表，将其编号（下标）通过此控制字节以二进制编码方式输出：位 7 在字节中始终为 1。

<Hmi>.progSelection.out.progList = <程序列表编号>

编号 （二进制编码）	编号 （十进制）	程序列表
1000 0001	129	用户程序列表 (/user/sinumerik/hmi/plc/programlist/plc_proglist_user.ppl)
1000 0011	131	制造商程序列表 (/oem/sinumerik/hmi/plc/programlist/plc_proglist_manufacturer.ppl)

数据类型与更新

USInt

关联

- <Hmi>.progSelection.out.request
  选择规定的 NC 程序（开始任务）
- <Hmi>.progSelection.out.progNumber
  程序编号选择基于所选程序列表中包含的程序
- <Hmi>.progSelection.in.completed
  任务结束
- <Hmi>.progSelection.in.error
  程序选择出错，参见"errorCode"中的错误识别码
- <Hmi>.progSelection.in.selected
  已选程序（任务尚未完成）
- <Hmi>.progSelection.in.identified
  识别到程序选择
- <Hmi>.progSelection.in.errorCode
  错误识别码
- MD9106 $MM_SERVE_EXTCALL_PROGRAMS
  处理 EXTCALL 调用
- MD51041 $MN_ENABLE_PROGLIST_USER
  激活 PLC 程序列表 USER（用户）部分
- MD51043 $MN_ENABLE_PROGLIST_MANUFACT
  激活 PLC 程序列表 MANUFACTURER（制造商）部分

更多信息

- PLC 功能手册

12.3.76.2 progNumber

PLC → OP	<Hmi>.progSelection.out.progNumber
程序编号选择基于所选程序列表中包含的程序	

说明

通过 PLC 选择程序：程序列表内部的程序下标

借助 PLC/HMI 接口可选择 PLC 程序列表中预设的“程序/工件”，以通过 NC 执行。

为了对通过 <Hmi>.progSelection.out.progList 选择的程序列表内的程序进行规定，通过此控制字节以二进制编码形式输出程序编号。

范围	程序编号
user	1 - 100
oem	201 - 255

数据类型与更新

USInt

关联

- <Hmi>.progSelection.out.request
  选择规定的 NC 程序（开始任务）
- <Hmi>.progSelection.out.progList
  程序列表编号
- <Hmi>.progSelection.in.completed
  任务结束
- <Hmi>.progSelection.in.error
  程序选择出错，参见"errorCode"中的错误识别码
- <Hmi>.progSelection.in.selected
  已选程序（任务尚未完成）
- <Hmi>.progSelection.in.identified
  识别到程序选择
- <Hmi>.progSelection.in.errorCode
  错误识别码
- MD9106 $MM_SERVE_EXTCALL_PROGRAMSEXTCALL-Aufrufe bearbeiten
- MD51041 $MN_ENABLE_PROGLIST_USERAktivierung PLC-Programmliste Bereich USER
- MD51043 $MN_ENABLE_PROGLIST_MANUFACTAktivierung PLC-Programmliste Bereich MANUFACTURER

更多信息

- PLC 功能手册

12.3.76.3 request

PLC → OP	<Hmi>.progSelection.out.request
选择规定的 NC 程序（开始任务）	

数据类型与更新

BOOL

值 TRUE

选择生效

值 FALSE

选择未生效

12.3.77 <Hmi>.mappingTable.in

12.3.77.1 error

OP → PLC	<hmi>.mappingTable.in.error
建立 HMI 通信时出错	

数据类型和更新

BOOL

12.3.77.2 ready

OP → PLC	<hmi>.mappingTable.in.ready
HMI 通信准备就绪	

数据类型和更新

BOOL

12.3.77.3 status

OP → PLC	<hmi>.mappingTable.in.status
HMI 通信状态	

数据类型和更新

Word

12.3.78 <Hmi>.mappingTable.out

12.3.78.1 numberOfTableLines

PLC → OP	<hmi>.mappingTable.out.numberOfTableLines
所用行数	

数据类型和更新

USInt

12.3.78.2 status

PLC → OP	<hmi>.mappingTable.out.status
映射结构状态	

数据类型和更新

Word

12.3.78.3 tableLines

PLC → OP	<hmi>.mappingTable.out.tableLines
JSON 字符串	

数据类型和更新

WString

12.3.78.4 version

PLC → OP	<hmi>.mappingTable.out.version
映射结构版本	

数据类型和更新

USInt

12.3.79 <Hmi>.userKeys.in

12.3.79.1 pressed

OP → PLC	<Hmi>.userKeys.in.pressed
已按下 HMI Sidescreen 用户自定义键 1	

数据类型与更新

BOOL

12.3.80 <Hmi>.userKeys.out

12.3.80.1 activateLed

PLC → OP	<Hmi>.userKeys.out.activateLed
激活 HMI Sidescreen 用户自定义键 1 的 LED	

数据类型与更新

BOOL

12.3.80.2 disable

PLC → OP	<Hmi>.userKeys.out.disable
锁定 HMI Sidescreen 用户自定义键 1	

数据类型与更新

BOOL

12.3.81 NCK_typeHmiAxis

下表说明了数据类型 NCK_typeHmiAxis 的含义：

说明

创建"NCK_typeHmiAxis"数据类型的所有实例时，必须设置“可从 HMI/OPC UA/Web API 到达”和“可从 HMI/OPC UA/Web API 写入”两种属性。

结构	说明
.config	HMI 轴实例的组态数据
.basic.in (页 988)	来自 HMI 的轴通用信号

12.3.82 <HmiAxis>.basic.in

12.3.82.1 handwheelSelected

OP → PLC	<hmiAxis>.basic.in.handwheelSelected
HMI：手轮 1..6 已选择（0 未选择）	

说明

操作组件中所选手轮的编号。

所选加工轴的手轮编号可在 PLC-Logik 中进行进一步处理。

数据类型和更新

USInt，在操作面板上更改数值

值

0：未选择手轮

1...6：所选手轮的编号

关联

- MD11324 $MN_HANDWH_VDI_REPRESENTATION
  VDI 接口中手轮编号的显示
- <Axis>.basic.out.handwheelSelection
  手轮 1..6 已分配或 0 未分配手轮（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）
- <Axis>.basic.in.handwheelAssignedNumber
  当前所分配手轮的编号：1..6 或 0 未分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.82.2 progTestRequested

OP → PLC	<HmiAxis>.basic.in.progTestRequested
HMI：已请求进行程序测试	

说明

激活程序测试

若“程序测试”功能生效，则轴在内部轴禁用下运行。系统会生成设定值，但其不输出至机床轴。实际值 = 设定值。

数据类型和更新

BOOL

值 TRUE

HMI 请求激活“程序测试”（禁止设定值输出）功能。

值 FALSE

HMI 未请求激活“程序测试”（锁定设定值输出）

功能。

关联

- <Axis>.basic.out.progTestSuppression
  请求抑制“程序测试”功能（转速设定值生效）
- <Axis>.basic.out.progTestRequest
  请求激活“程序测试”功能（转速设定值未生效）
- <HmiAxis>.basic.in.progTestSuppressionRequested
  HMI：已抑制程序测试

更多信息

- 基础功能手册；“跨通道程序协调
  和逐通道试运行”>“逐通道试运行”章节

12.3.82.3 progTestSuppressionRequested

OP → PLC	<HmiAxis>.basic.in.progTestSuppressionRequested
HMI：已抑制程序测试	

说明

抑制程序测试

在轴当前归属于的通道中，若“程序测试”（PRT）生效，则轴在内部轴禁用下运行。系统会生成设定值，但其不输出至机床轴。实际值 = 设定值。

若为轴抑制程序测试，则即便在“程序测试”功能在轴当前归属于的通道中生效的情况下，系统也将设定值输出至机床轴。

数据类型与更新

BOOL

值 TRUE

HMI 请求抑制程序测试（禁止设定值输出）。

值 FALSE

HMI 未请求抑制程序测试（禁止设定值输出）。

关联

- <Axis>.basic.out.progTestSuppression
  请求抑制“程序测试”功能（转速设定值生效）
- <Axis>.basic.out.progTestRequest
  请求激活“程序测试”功能（转速设定值未生效）
- <HmiAxis>.basic.in.progTestRequested
  HMI：已请求进行程序测试

更多信息

- 基础功能手册；“跨通道程序协调
  和逐通道试运行”>“逐通道试运行”章节

12.3.83 NCK_typeHmiChan

下表说明了数据类型 NCK_typeHmiChan 的含义：

说明

创建“NCK_typeHmiChan”数据类型的所有实例时，必须设置“可从 HMI/OPC UA/Web API 到达”和“可从 HMI/OPC UA/Web API 写入”两种属性。

结构	说明
.config	HMI 通道实例的组态数据
.basic.in (页 992)	来自 HMI 的通道通用信号
.axisSelection.in (页 1001)	通过 HMI 选择工作轴的输入信号
.axisSelection.out (页 1004)	通过 HMI 选择工作轴的输出信号
.geoAxis.in (页 1006)	HMI：几何轴输入信号
.orientationAxis.in (页 1007)	HMI：定向轴输入信号

12.3.84 <HmiChan>.basic.in

12.3.84.1 atStopSetOvrZeroSelected

OP → PLC	<HmiChan>.basic.in.atStopSetOvrZeroSelected
HMI：停止时将倍率设为 0%	

说明

在停止时将倍率设置为 0%
当机床功能“已组态的停止 (CST)”为停止状态时，将倍率设置为 0%。
PLC-Logik 负责影响倍率。

说明

仅在 Powerride 时使用。

数据类型和更新

BOOL，在操作面板上更改数值

12.3.84.2 configuredStopFuncSelected

OP → PLC	<HmiChan>.basic.in.configuredStopFuncSelected
HMI：请求通过操作软件激活已组态的停止 (CST)	

说明

通过操作软件激活已组态的停止 (CST)
应在 NC 中激活“已组态的停止”功能。
该信号必须在 PLC-Logik 中进行处理，并将其传输给 NC (<Chan>.basic.out.configuredStopFunc)。

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

HMI：已请求“已组态的停止”(CST)

值 FALSE

HMI：未请求“已组态的停止”(CST)

关联

- 设置选项“已组态的停止”（需要许可证）
- 通过 MD52210 Bit18 激活 SINUMERIK Operate 中的显示
- <Chan>.basic.in.configuredStopFuncActive
  已组态的停止已激活
- <Chan>.basic.out.configuredStopFunc
  激活已组态的停止

更多信息

基础功能手册；“运行方式组、通道、程序运行、Reset 行为”章节

12.3.84.3 contourHandwheelSelected

OP → PLC	<hmiChan>.basic.in.contourHandwheelSelected
HMI：选择手轮 1..6 作为轮廓手轮（0 未选择）	

说明

操作组件中作为轮廓手轮所选择的手轮的编号

轮廓手轮的手轮编号可在 PLC-Logik 中进行进一步处理。

数据类型和更新

USInt，在操作面板上更改数值

值

0：未选择手轮

1...6：所选轮廓手轮的编号

关联

- MD11324 $MN_HANDWH_VDI_REPRESENTATION
  VDI 接口中手轮编号的显示
- <Chan>.basic.out.contourHandwheelSelection
  手轮 1..6 已分配或 0 未分配手轮（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）
- <Chan>.basic.in.contourHandwheelAssignedNumber
  当前所分配手轮的编号：1..6 或 0 未分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.84.4 dryRunSelected

OP → PLC	<HmiChan>.basic.in.dryRunSelected
HMI：已请求试运行进给率 (DRY)	

说明

空运行进给（DRY）已选择

提示

在 SINUMERIK Operate 操作界面的“自动”>“程序控制”操作区域将 HMI/PLC 接口信号 <HmiChan>.basic.in.dryRunSelected 置位，可以选择试运行进给 (DRY)。

提示

空运行进给率的值通过以下数据设置：

SD42100 $SC_DRY_RUN_FEED

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

空运行进给（DRY）已选择。

值 FALSE

空运行进给（DRY）未选择。

关联

- <Chan>.basic.out.dryRunFeedrate
  请求激活试运行进给率 (DRY)
- SD42100 $SC_DRY_RUN_FEED
  试运行进给率
- SD42101 $SC_DRY_RUN_FEED_MODE
  测试运行速度模式

更多信息

- 功能手册之基本功能
  - 章节：“运行方式组、通道、程序运行、复位行为”>“程序测试”>“以试运行进给率执行程序”
  - 章节：“进给”>“进给控制”>“试运行进给”
- 操作手册
  - 章节“工件加工”>“程序运行控制”>“程序控制”

12.3.84.5 handwheelOffsetDrfSelected

OP → PLC	<HmiChan>.basic.in.handwheelOffsetDrfSelected
HMI：已请求手轮偏移 (DRF)	

说明

已选手轮偏移 (DRF)

只要手轮偏移 (DRF) 处于活动状态，就能在自动或 MDA 运行方式下，通过分配给轴的手轮更改手轮偏移。

提示

通过确认“手轮偏移”复选框，在 SINUMERIK Operate 操作界面的“自动”>“程序控制”操作区域中可以选择手轮偏移 (DRF)。

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

已选手轮偏移 (DRF)。

值 FALSE

未选手轮偏移 (DRF)。

关联

- <Chan>.basic.out.handwheelOffsetDrf
  已请求激活手轮偏移 (DRF)

更多信息

- 轴和主轴功能手册；“手动运行”章节

12.3.84.6 m01AssociatedAuxFuncSelected

OP → PLC	<HmiChan>.basic.in.m01AssociatedAuxFuncSelected
HMI：已请求程序控制 M01“相关辅助功能”	

说明

关联辅助功能已选择

在 SINUMERIK Operate 操作界面的“自动”>“程序控制”操作区域将 HMI/PLC 接口信号 <HmiChan>.basic.in.m01AssociatedAuxFuncSelected 置位，可以选择“关联辅助功能”(M-1)。

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

关联辅助功能已选择。

值 FALSE

关联辅助功能未选择。

关联

- <Chan>.basic.out.m01AssociatedAuxFunc
  请求激活 M0/M1“关联辅助功能”
- <Chan>.basic.in.m00m01AssociatedAuxFuncActive
  M00 / M01：相应的辅助功能“编程的停止/有条件停止”处于活动状态
- • MD22254 $MC_AUXFU_ASSOC_M0_VALUE
  用于程序停止的附加 M 功能
- • MD22256 $MC_AUXFU_ASSOC_M1_VALUE

更多信息

- 基础功能手册；“辅助功能输出至 PLC”>“关联辅助功能”章节

12.3.84.7 m01Selected

OP → PLC	<HmiChan>.basic.in.m01Selected
HMI：已请求程序控制 M01“有条件停止”	

说明

M01 已选择

通过 SINUMERIK Operate 操作界面选择程序控制“有条件停止” M01：在“自动”>“程序控制”操作区域，将 HMI/PLC 接口信号 <HmiChan>.basic.in.m01Selected 置位。

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

已选择程序控制“有条件停止”M01。

值 FALSE

未选择程序控制“有条件停止”M01。

关联

- <Chan>.basic.out.m01
  请求激活程序控制“有条件停止”(M01)
- <Chan>.basic.in.m00m01Active
  M00 / M01：“编程的停止/有条件停止”处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”>“程序控制”>“通过操作界面或 PLC 用户程序选择功能”章节
- HMI Advanced 操作手册之“加工操作区”

12.3.84.8 progTestSelected

OP → PLC	<HmiChan>.basic.in.progTestSelected
HMI：已请求程序测试 (PRT)	

说明

已请求通过操作软件进行程序测试 (PTR)

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

HMI：已请求程序测试 (PRT)

值 FALSE

HMI：未请求程序测试 (PRT)

关联

- <Chan>.basic.out.progTest
  请求激活程序测试 (PRT)
- <Chan>.basic.in.progTestSelected
  已选程序测试 (PRT)

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、Reset 行为”章节

12.3.84.9 rapidTraversOvrSelected

OP → PLC	<HmiChan>.basic.in.rapidTraversOvrSelected
HMI：已选快速运行的进给倍率	

说明

快进进给率修调已选择

若在机床控制面板上无独立的快速移动倍率开关，可在进给率修调与快速移动倍率之间进行切换。可通过操作界面选择快速移动进给率修调，从而切换至快速移动倍率。此时接口信号置位：

<HmiChan>.basic.in.rapidTraversOvrSelected = TRUE

通过 PLC 基本程序设置下列接口信号：

- 快速运行修调生效 = 已选快速运行进给修调
  <Chan>.basic.out.pathRapidTraversOvr = <HmiChan>.basic.in.rapidTraversOvrSelected
- 快速运行修调 = 进给修调
  <Chan>.basic.out.pathRapidTraversOvrFactor =
  <Chan>.basic.out.pathFeedrateOvrFactor

> 100% 的倍率值会被限制为 100%。

提示

也可通过 PLC 用户程序，在快速运行修调与进给修调之间

进行切换。为此需要设置下列接口信号：

- 快速运行修调生效
  <Chan>.basic.out.pathRapidTraversOvr = TRUE
- 快速运行修调 = 进给修调
  <Chan>.basic.out.pathRapidTraversOvrFactor =
  <Chan>.basic.out.pathFeedrateOvrFactor

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

空运行进给（DRY）已选择。

值 FALSE

空运行进给（DRY）未选择。

关联

- <Chan>.basic.out.pathFeedrateOvrFactor
  可以通过二进制或格雷码编码形式规定轨迹进给倍率系数 (MD12020 $MN_OVR_FEED_IS_GRAY_CODE)
- <Chan>.basic.out.pathRapidTraversOvrFactor
  可以通过二进制或格雷码编码形式规定轨迹快速运行叠加系数 (MD12050 $MN_OVR_FACTOR_RAPID_TRA)
- <Chan>.basic.out.pathRapidTraversOvr
  轨迹快速运行叠加处于活动状态

更多信息

- 轴和主轴功能手册；“进给”>“进给控制”>“通过机床控制面板设置进给倍率”章节

12.3.84.10 reposModeSelectedNumber

OP → PLC	<HmiChan>.basic.in.reposModeSelectedNumber
HMI：已选 REPOS 接近运行方式： 可能的值见下表	

说明

请求 REPOS 模式
再次返回轮廓可以使加工从中断位置开始继续进行。
为此，有多种方式可供选择。下表列出了可行的方式和相应的数值。

数据类型和更新

USInt，在操作面板上更改数值

值	REPOS 接近运行方式
0	无 REPOS 运行方式
1	接近程序段起点 (RMBBL)
2	接近中断点 (RMIBL)
3	接近程序段终点 (RMEBL)
4	接近下一个轨迹点 (RMNBL)

关联

- MD11470 $MN_REPOS_MODE_MASK
  重新定位属性
- <Chan>.basic.out.reposModeSelection
  REPOS 接近运行方式：
  可能的值见上表
- <Chan>.basic.out.reposModeActivationRequest
  请求为当前主运行程序段激活“REPOS”运行方式
- <Chan>.basic.in.reposModeSelectedNumber
  REPOS 接近运行方式：
  可能的值见上表

更多信息

基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.84.11 reposModeChanged

OP → PLC	<HmiChan>.basic.in.reposModeChanged
HMI：REPOS 运行方式切换	

说明

REPOS 模式变更

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

HMI：REPOS 运行方式切换

值 FALSE

HMI：无 REPOS 运行方式切换

12.3.84.12 skipBlockLevel0...9

OP → PLC	<HmiChan>.basic.in.skipBlockLevel0 ... <HmiChan>.basic.in.skipBlockLevel9
已选择跳转跳转级 0 - 9 的某个零件程序段 (Skip, SKP)	

数据类型与更新

BOOL

值 TRUE

已请求跳转程序段，跳转级 0 - 9

12.3.85 <HmiChan>.axisSelection.in

12.3.85.1 selectedKeyActuated

OP → PLC	<HmiChan>.axisSelection.in.selectedKeyActuated
脉冲沿切换 0 → 1：轴选择“按键”（触摸键）	

说明

操作软件 (SINUMERIK Operate) 中针对相应轴的复选框被激活。

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

按下轴选择“按键”（触摸键）

值 FALSE

未按下轴选择“按键”（触摸键）

更多信息

- 功能手册：SINUMERIK ONE 轴和主轴
- 选择加工轴/几何轴（通过 HMI 选择 JOG 轴）

12.3.85.2 selectedNumber

OP → PLC	<HmiChan>.axisSelection.in.selectedNumber
轴编号：机床轴/几何轴/定向轴	

说明

操作软件 (SINUMERIK Operate) 中所选轴的编号

数据类型和更新

USInt，在操作面板上更改数值

值

轴编号：机床轴/几何轴/定向轴

使用 SINUMERIK Operate 时，将传输在特定通道的机床数据中被定义的编号。

关联

- <Axis>.config.in.axisNumber
  轴编号
- MD20050[0..2] $MC_AXCONF_GEOAX_ASSIGN_TAB
  将几何轴分配给通道轴
  MD20070[0..19] $MC_AXCONF_MACHAX_USED
  通道中的加工轴编号有效

更多信息

- 功能手册：SINUMERIK ONE 轴和主轴

12.3.85.3 selectedType

OP → PLC	<HmiChan>.axisSelection.in.selectedType
轴选择类型： 0：机床轴， 1：几何轴， 2：定向轴	

说明

操作软件中所选轴的类型
该信号可由 PLC-Logik 使用，例如用于区分哪个接口必须用于运行请求。

数据类型和更新

USInt，在操作面板上更改数值

值

轴选择类型：
0：机床轴
1：几何轴
2：定向轴

关联

- <Axis>.basic.out.traverseKeyMinus
  在 JOG 运行方式下沿负方向移动加工轴 (MD32020 $MA_JOG_VELO)
- <Axis>.basic.out.traverseKeyPlus
  在 JOG 运行方式下沿正方向移动加工轴 (MD32020 $MA_JOG_VELO)
- <Chan>.geoAxis1.out.traversKeyMinus
- <Chan>.geoAxis2.out.traversKeyMinus
- <Chan>.geoAxis3.out.traversKeyMinus
  请求几何轴“负”移动键信号（用于负轴方向运行）
- <Chan>.geoAxis1.out.traversKeyPlus
- <Chan>.geoAxis2.out.traversKeyPlus
- <Chan>.geoAxis3.out.traversKeyPlus
  请求几何轴“正”移动键信号（用于正轴方向运行）
- <Chan>.orientationAxis1.out.traversKeyMinus
- <Chan>.orientationAxis2.out.traversKeyMinus

- <Chan>.orientationAxis3.out.traversKeyMinus
  请求定向轴“负”移动键信号（用于负轴方向运行）
- <Chan>.orientationAxis1.out.traversKeyPlus
- <Chan>.orientationAxis2.out.traversKeyPlus
- <Chan>.orientationAxis3.out.traversKeyPlus
  请求定向轴“正”移动键信号（用于正轴方向运行）

更多信息

功能手册：SINUMERIK ONE 轴和主轴

12.3.86 <HmiChan>.axisSelection.out

12.3.86.1 activeNumber

PLC → OP	<HmiChan>.axisSelection.out.activeNumber
活动的轴编号（反馈至 HMI）：机床轴/几何轴/定向轴	

说明

PLC-Logik 中已选择/已使用的轴的编号
可作为对操作软件的反馈，激活轴名称旁边的复选框 (SINUMERIK Operate)。

数据类型和更新

USInt，在操作面板上更改数值

值

活动的轴编号（反馈至 HMI）：机床轴/几何轴/定向轴

含义取决于 <HmiChan>.axisSelection.out.activeType

关联

- <HmiChan>.axisSelection.in.axisNumber
  轴编号：机床轴/几何轴/定向轴
- <HmiChan>.axisSelection.out.activeType
  轴选择处于活动状态

- MD20050[0..2] $MC_AXCONF_GEOAX_ASSIGN_TAB
  将几何轴分配给通道轴
- MD20070[0..19] $MC_AXCONF_MACHAX_USED
  通道内有效的机床轴号

更多信息

- 功能手册：SINUMERIK ONE 轴和主轴
- 选择加工轴/几何轴（通过 HMI 选择 JOG 轴）

12.3.86.2 activeType

PLC → OP	<HmiChan>.axisSelection.out.activeType
轴选择处于活动状态（反馈至 HMI）： 0：机床轴 1：几何轴 2：定向轴	

说明

PLC-Logik 中所选轴的类型

数据类型和更新

USInt，在操作面板上更改数值

值

轴选择处于活动状态（反馈至 HMI）：
0：机床轴
1：几何轴
2：定向轴

关联

- <HmiChan>.axisSelection.in.selectedType
  轴选择类型：
  0：机床轴
  1：几何轴
  2：定向轴
- <HmiChan>.axisSelection.out.activeNumber
  活动的轴编号（反馈至 HMI）：机床轴/几何轴/定向轴

- MD20050[0..2] $MC_AXCONF_GEOAX_ASSIGN_TAB
  将几何轴分配给通道轴
- MD20070[0..19] $MC_AXCONF_MACHAX_USED
  通道内有效的机床轴号

更多信息

- 功能手册：SINUMERIK ONE 轴和主轴

12.3.87 <HmiChan>.geoAxis.in

12.3.87.1 handwheelSelected

OP → PLC	<hmiChan>.geoAxis1.in.handwheelSelected
HMI：已针对几何轴选择手轮 1..6（0 未选择）	

说明

操作组件中所选手轮的编号

所选几何轴的手轮编号可在 PLC-Logik 中进行进一步处理。

数据类型和更新

USInt，在操作面板上更改数值

值

0：未选择手轮

1...6：所选手轮（适用于几何轴）的编号

关联

- MD11324 $MN_HANDWH_VDI_REPRESENTATION
  VDI 接口中手轮编号的显示
- <Chan>.geoAxis1.out.handwheelSelection
  手轮 1..6 已分配或 0 未分配手轮（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）
- <Chan>.geoAxis1.in.handwheelAssignedNumber
  当前所分配手轮的编号：1..6 或 0 未分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.88 <HmiChan>.orientationAxis.in

12.3.88.1 handwheelSelected

OP → PLC	<hmiChan>.orientationAxis1.in
HMI：已针对定向轴选择手轮 1..6（0 未选择）	

说明

操作组件中所选手轮的编号

所选定向轴的手轮编号可在 PLC-Logik 中进行进一步处理。

数据类型和更新

USInt，在操作面板上更改数值

值

1...6：所选手轮（适用于定向轴）的编号

关联

- MD11324 $MN_HANDWH_VDI_REPRESENTATION
  VDI 接口中手轮编号的显示
- <Chan>.orientationAxis1.out.handwheelSelection
  手轮 1..6 已分配或 0 未分配手轮（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）
- <Chan>.orientationAxis1.in.handwheelAssignedNumber
  当前所分配手轮的编号：1..6 或 0 未分配（MD11324 $MN_HANDWH_VDI_REPRESENTATION = 1，已进行二进制编码）

更多信息

- SINUMERIK ONE 轴和主轴功能手册；“手动运行”章节

12.3.89 NCK_typeHmiModeGroup

下表说明了数据类型 NCK_typeHmiModeGroup 的含义：

说明

创建"NCK_typeHmiModeGroup"数据类型的所有实例时，必须设置"可从 HMI/OPC UA/Web API 到达"和"可从 HMI/OPC UA/Web API 写入"两种属性。

结构	说明
.config	HMI 运行方式组实例的组态数据
.basic.in (页 1008)	来自 HMI 的运行方式组通用信号

12.3.90 <HmiModeGroup>.basic.in

12.3.90.1 autoModeRequested

OP → PLC	<HmiModeGroup>.basic.in.autoModeRequested
HMI：已请求切换至自动运行方式	

说明

通过操作软件 (HMI) 请求采用 AUTOMATIC 运行方式

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

存在选择 AUTO 运行方式的 HMI 请求。

值 FALSE

不存在选择 AUTO 运行方式的 HMI 请求。

关联

- <ModeGroup>.basic.out.autoMode
  请求切换至自动运行方式
- <ModeGroup>.basic.in.autoModeActive
  自动运行方式处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.90.2 contTraversRequested

OP → PLC	<HmiModeGroup>.basic.in.contTraversRequested
HMI：已请求连续运行	

数据类型和更新

BOOL

值 TRUE

HMI：已请求连续运行

值 FALSE

HMI：未请求连续运行

12.3.90.3 mdaModeRequested

OP → PLC	<HmiModeGroup>.basic.in.mdaModeRequested
HMI：已请求切换至 MDA 运行方式	

说明

通过操作软件 (HMI) 请求采用 MDA 运行方式

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

存在选择 MDI 运行方式的 HMI 请求。

值 FALSE

不存在选择 MDI 运行方式的 HMI 请求。

关联

- <ModeGroup>.basic.out.mdaMode
  请求切换至 MDA 运行方式
- <ModeGroup>.basic.in.mdaModeActive
  MDA 运行方式处于活动状态

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.90.4 inc1Requested

OP → PLC	<HmiModeGroup>.basic.in.inc1Requested
HMI：已请求移动 1 个增量	

说明

通过操作软件请求机床功能 INC1（根据请求移动 1 个增量）。

说明

运行方式组内的增量预选功能必须已显式使能。
(<Nc>.basic.out.selectIncFromModeGroup := TRUE)
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

已请求机床功能 INC1（“移动 1 个增量”）

值 FALSE

未请求机床功能 INC1（“移动 1 个增量”）

关联

- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”

- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 10000 个增量”
- <ModeGroup>.basic.out.incVar
  激活“根据请求移动用户自定义增量数”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.90.5 inc10Requested

OP → PLC	<HmiModeGroup>.basic.in.inc10Requested
HMI：已请求移动 10 个增量	

说明

通过操作软件请求机床功能 INC10（根据请求移动 10 个增量）。

说明

运行方式组内的增量预选功能必须已显式使能。
(<Nc>.basic.out.selectIncFromModeGroup := TRUE)
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

已请求机床功能 INC10（“移动 10 个增量”）

值 FALSE

未请求机床功能 INC10（“移动 10 个增量”）

关联

- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”

- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 10000 个增量”
- <ModeGroup>.basic.out.incVar
  激活“根据请求移动用户自定义增量数”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.90.6 inc100Requested

OP → PLC	<HmiModeGroup>.basic.in.inc100Requested
HMI：已请求移动 100 个增量	

说明

通过操作软件请求机床功能 INC100（根据请求移动 100 个增量）。

说明

运行方式组内的增量预选功能必须已显式使能。
(<Nc>.basic.out.selectIncFromModeGroup := TRUE)
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

已请求机床功能 INC100（“移动 100 个增量”）

值 FALSE

未请求机床功能 INC100（“移动 100 个增量”）

关联

- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”

- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 10000 个增量”
- <ModeGroup>.basic.out.incVar
  激活“根据请求移动用户自定义增量数”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.90.7 inc1000Requested

OP → PLC	<HmiModeGroup>.basic.in.inc1000Requested
HMI：已请求移动 1000 个增量	

说明

通过操作软件请求机床功能 INC1000（根据请求移动 1000 个增量）。

说明

运行方式组内的增量预选功能必须已显式使能。
(<Nc>.basic.out.selectIncFromModeGroup := TRUE)
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

已请求机床功能 INC1000（“移动 1000 个增量”）

值 FALSE

未请求机床功能 INC1000（“移动 1000 个增量”）

关联

- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”

- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 10000 个增量”
- <ModeGroup>.basic.out.incVar
  激活“根据请求移动用户自定义增量数”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.90.8 inc10000Requested

OP → PLC	<HmiModeGroup>.basic.in.inc10000Requested
HMI：已请求移动 10000 个增量	

说明

通过操作软件请求机床功能 INC10000（根据请求移动 10000 个增量）。

说明

运行方式组内的增量预选功能必须已显式使能。
(<Nc>.basic.out.selectIncFromModeGroup := TRUE)
可在运行方式组内选择按增量运行。

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

已请求机床功能 INC10000（“移动 10000 个增量”）

值 FALSE

未请求机床功能 INC10000（“移动 10000 个增量”）

关联

- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”

- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”
- <ModeGroup>.basic.out.incVar
  激活“根据请求移动用户自定义增量数”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.90.9 incVarRequested

OP → PLC	<HmiModeGroup>.basic.in.incVarRequested
HMI：已请求移动用户自定义增量数	

说明

请求连续运行（默认）
请求重置运行方式组内已激活的增量预选。

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

HMI：已请求移动用户自定义增量数

值 FALSE

HMI：未请求连续运行

关联

- <ModeGroup>.basic.out.inc1
  激活“根据请求移动 1 个增量”
- <ModeGroup>.basic.out.inc10
  激活“根据请求移动 10 个增量”
- <ModeGroup>.basic.out.inc100
  激活“根据请求移动 100 个增量”
- <ModeGroup>.basic.out.inc1000
  激活“根据请求移动 1000 个增量”

- <ModeGroup>.basic.out.inc10000
  激活“根据请求移动 10000 个增量”
- <ModeGroup>.basic.out.incVar
  激活“根据请求移动用户自定义增量数”

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.90.10 jogModeRequested

OP → PLC	<HmiModeGroup>.basic.in.jogModeRequested
HMI：已请求切换至 JOG 运行方式	

说明

通过操作软件 (HMI) 请求采用 JOG 运行方式

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

存在选择机床功能 TEACH IN 的 HMI 请求。

值 FALSE

不存在选择机床功能 TEACH IN 的 HMI 请求。

关联

- <ModeGroup>.basic.out.jogMode
  请求切换至 JOG 运行方式
- <ModeGroup>.basic.in.jogModeActive
  JOG 运行方式生效

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.90.11 reposRequested

OP → PLC	<HmiModeGroup>.basic.in.reposRequested
HMI：已请求执行机床功能 REPOS	

说明

通过操作软件 (HMI) 请求采用 REPOS 运行方式

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

存在选择机床功能 REPOS 的 HMI 请求。

值 FALSE

不存在选择机床功能 REPOS 的 HMI 请求。

关联

- <ModeGroup>.basic.in.reposActive
  机床功能 REPOS 已激活
- <ModeGroup>.basic.out.repos
  请求机床功能 REPOS

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.90.12 refRequested

OP → PLC	<HmiModeGroup>.basic.in.refRequested
HMI：已请求执行机床功能 REF	

说明

通过操作软件 (HMI) 请求采用 REF 运行方式

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

存在选择机床功能 REF 的 HMI 请求。

值 FALSE

不存在选择机床功能 REF 的 HMI 请求。

关联

- <ModeGroup>.basic.in.refActive
  机床功能 REF 处于活动状态
- <ModeGroup>.basic.out.refPoint
  请求机床功能 REF

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.90.13 teachInRequested

OP → PLC	<HmiModeGroup>.basic.in.teachInRequested
HMI：已请求执行机床功能 TEACH IN	

说明

通过操作软件 (HMI) 请求采用 TEACH IN 运行方式

数据类型和更新

BOOL，在操作面板上更改数值

值 TRUE

存在选择机床功能 TEACH IN 的 HMI 请求。

值 FALSE

不存在选择机床功能 TEACH IN 的 HMI 请求。

关联

- <ModeGroup>.basic.in.teachInActive
  机床功能 TEACH IN 处于活动状态
- <ModeGroup>.basic.out.teachIn
  请求机床功能 TEACH IN

更多信息

- 基础功能手册；“运行方式组、通道、程序运行、复位行为”章节

12.3.91 <HmiModeGroup>basic.out

12.3.91.1 disableModeChangeover

PLC → NC	<HmiModeGroup>.basic.out.disableModeChangeover
锁定该运行方式组内的运行方式切换	

数据类型和更新

BOOL

12.3.92 NCK_typeHmiNC

下表说明了数据类型 NCK_typeHmiNc 的含义：

说明

创建"NCK_typeHmiNC"数据类型的实例时，必须设置"可从 HMI/OPC UA/Web API 到达"和"可从 HMI/OPC UA/Web API 写入"两种属性。

结构	说明
.config	HMI NC 实例的组态数据
.registered	NC 已进行了 HMI 注册
.collision.in (页 1019)	来自 HMI 的 NC 碰撞信号

12.3.93 <HmiNc>.collision.in

12.3.93.1 collision.in

OP → PLC	<HmiNc>.collision.in.machineAreasAutoDeactivated
HMI：已请求在"AUTO"运行方式下，禁用所有保护区域类型为"机床"的保护区域	

OP → PLC	<HmiNc>.collision.in.toolAreasAutoDeactivated
HMI：已请求在"AUTO"运行方式下，禁用所有保护区域类型为“刀具”的保护区域	

OP → PLC	<HmiNc>.collision.in.fixtureAreasAutoDeactivated
HMI：已请求在"AUTO"运行方式下，禁用所有保护区域类型为“夹具”的保护区域	

OP → PLC	<HmiNc>.collision.in.workpieceAreasAutoDeactivated
HMI：已请求在"AUTO"运行方式下，禁用所有保护区域类型为“工件”的保护区域	

OP → PLC	<HmiNc>.collision.in.machineAreasJogDeactivated
HMI：已请求在"JOG"运行方式下，禁用所有保护区域类型为“机床”的保护区域	

OP → PLC	<HmiNc>.collision.in.toolAreasJogDeactivated
HMI：已请求在"JOG"运行方式下，禁用所有保护区域类型为“刀具”的保护区域	

OP → PLC	<HmiNc>.collision.in.fixtureAreasJogDeactivated
HMI：已请求在"JOG"运行方式下，禁用所有保护区域类型为“夹具”的保护区域	

OP → PLC	<HmiNc>.collision.in.workpieceAreasJogDeactivated
HMI：已请求在"JOG"运行方式下，禁用所有保护区域类型为“工件”的保护区域	

说明

碰撞监测：取消保护区组

变量	运行方式	保护区类型 [1)]
<HmiNc>.collision.in.machineAreasAutoDeactivated	自动方式	机床 (MACHINE)
<HmiNc>.collision.in.toolAreasAutoDeactivated		刀具 (TOOL)
<HmiNc>.collision.in.fixtureAreasAutoDeactivated		夹具 (FIXTURE)
<HmiNc>.collision.in.workpieceAreasAutoDeactivated		工件 (WORKPIECE)
<HmiNc>.collision.in.machineAreasJogDeactivated	Jog	机床 (MACHINE)
<HmiNc>.collision.in.toolAreasJogDeactivated		刀具 (TOOL)
<HmiNc>.collision.in.fixtureAreasJogDeactivated		夹具 (FIXTURE)
<HmiNc>.collision.in.workpieceAreasJogDeactivated		工件 (WORKPIECE)

[1)] 保护区的类型（$NP_PROT_TYPE）

说明

在 SINUMERIK Operate 操作界面的“自动”、“JOG”或“MDA”>“ETC 键（“>”）”>“设置”>“防撞保护”>“启用和关闭防撞保护”操作区域，将保护区域组的 HMI/PLC 接口信号 <HmiNc>.collision.in... 置位，可以禁用该保护区域组。

说明

如果在自动运行方式下使能手动运行（MD10735，Bit 0 == 1：可在自动模式下点动运行），则即便在自动运行方式下，也可在手动运行期间采用 JOG 运行方式的设置。

数据类型和更新

BOOL

值 TRUE

HMI 已请求在选择的运行方式中将该保护范围类型的所有保护范围取消。

值 FALSE

未请求在选择的运行方式中将该保护区类型的所有保护区取消。

关联

- <Nc>.collision.out...
  Kollisionsvermeidung: Schutzbereichsgruppe deaktivieren
- $NP_PROT_TYPETyp des Schutzbereichs
- MD10735 $MN_JOG_MODE_MASKEinstellungen für Betriebsart JOG

更多信息

- 操作手册；“干涉保护”章节
- 监控和补偿功能手册；章节：“机床几何建模”>“调试”>“系统变量：保护范围”>“$NP_PROT_TYPE”
- 监控和补偿功能手册；“干涉保护”章节

12.3.94 NCK_typeToaUnit

下表说明了数据类型 NCK_typeToaUnit 的含义：

结构	说明
.config	工艺目标范围的组态数据
.registered	工艺目标范围已注册
.toaUnitNumber	工艺目标范围的编号
.associatedChans	表示哪些通道编号已分配给该工艺目标范围
.maxToolholder	最高刀架号
.bufferMagazine	暂时存储刀库的编号
.loadingMagazine	上料库的编号
.magazineCount	刀库列表中条目的数量
.magazines[0].magazineNumber	刀库编号 (UINT)
.magazines[0].magazineType	刀库类型 (UINT)
.magazines[0].locationCount	刀库中的刀位数量 (UINT)

12.3.95 NCK_typeLoadingPoint

下表说明了数据类型 NCK_typeLoadingPoint 的含义：

结构	说明
.config	装载位的组态数据
.registered	装载位已注册
.loadingPoint	装载位的编号
.toaUnit	对应的工艺目标范围的编号
.loadingMagazine (页 1024)	装载位所在的刀库的编号，通常是上料库
.loadingLocation (页 1024)	刀库中装载位的位置编号
.ack.in (页 1025)	用于应答刀具传输的输入信号
.ack.out (页 1026)	用于应答刀具传输的输出信号
.command.in (页 1028)	来自刀具管理的信号
.command.in.basic (页 1029)	通用指令信号。仅当已设置 <loadingPoint>.command.in.request 时有效
.command.in.options (页 1032)	指令的选项。仅当已设置 <loadingPoint>.command.in.request 时有效
.command.in.toolProperties (页 1032)	刀具属性，仅有效。仅当已设置 <loadingPoint>.command.in.request 时有效
.command.in.toolStatus (页 1033)	刀具的状态。仅当已设置 <loadingPoint>.command.in.request 时有效
.command.in.multitoolProperties (页 1039)	多刀的属性。仅当已设置 <loadingPoint>.command.in.multitoolRequest 时有效

12.3.96 <LoadingPoint>.config

12.3.96.1 loadingLocation

NC → PLC	<LoadingPoint>.config.loadingLocation
装载刀库中装载位置的刀位编号	

说明

在注册了 LoadingPoint 实例后，此处提供了装载刀库中装载位置的刀位编号。在 <LoadingPoint>.config.loadingMagazine 中可提供装载刀库的刀库编号。

数据类型和更新

UInt，一次

值

刀位号

关联

- <LoadingPoint>.config.loadingMagazine
  装载位置所处的刀库编号（装载刀库）

更多信息

- 功能手册：SINUMERIK ONE 刀具管理

12.3.96.2 loadingMagazine

NC → PLC	<LoadingPoint>.config.loadingMagazine
装载位置所处的刀库编号（装载刀库）	

说明

在注册了 LoadingPoint 实例后，此处提供了装载刀库的刀库编号。在 <LoadingPoint>.config.loadingLocation 中可提供装载刀库中装载位置的刀位编号。

数据类型和更新

UInt，一次

值

刀库编号

关联

- <LoadingPoint>.config.loadingLocation
  装载刀库中装载位置的刀位编号

更多信息

- 功能手册：SINUMERIK ONE 刀具管理

12.3.97 <LoadingPoint>.ack.in

12.3.97.1 busy

NC → PLC	<LoadingPoint>.ack.in.busy
正在应答	

数据类型和更新

BOOL

12.3.97.2 done

NC → PLC	<LoadingPoint>.ack.in.done
Excel_表_中的_描述	

数据类型和更新

BOOL

12.3.97.3 error

NC → PLC	<LoadingPoint>.ack.in.error
应答被拒，见状态	

数据类型和更新

BOOL

12.3.97.4 status

NC → PLC	<LoadingPoint>.ack.in.status
应答失败后的状态信息	

数据类型和更新

Word

12.3.98 <LoadingPoint>.ack.out

12.3.98.1 execute

PLC → NC	<LoadingPoint>.ack.out.execute
开始应答移刀	

数据类型和更新

BOOL

12.3.98.2 multitoolLocation

PLC → NC	<LoadingPoint>.ack.out.multitoolLocation
刀具多刀内的当前刀位编号	

数据类型和更新

Int

12.3.98.3 status

PLC → NC	<LoadingPoint>.ack.out.status
刀具传输状态	

状态信息

包含应答状态 1–10 的信息结束指令，包含应答状态 105 的信息

使指令继续保持活动状态。

说明

1：刀具管理任务已完成。

3：出错。

5：完成定位或新刀具到达指定位置。

10:同 1，但刀库位置不变

105:刀具在指定位置。

数据类型和更新

USInt

12.3.98.4 toolLocation

PLC → NC	<LoadingPoint>.ack.out.toolLocation
刀具当前所处刀库刀位编号	

数据类型和更新

Int

12.3.98.5 toolMagazine

PLC → NC	<LoadingPoint>.ack.out.toolMagazine
刀具当前所处刀库编号	

数据类型和更新

Int

12.3.98.6 totalComplete

PLC → NC	<LoadingPoint>.ack.out.totalComplete
执行正面应答，使用状态 1 或 5 或 6，忽略刀具位置参数和状态参数	

数据类型和更新

BOOL

12.3.98.7 totalFailed

PLC → NC	<LoadingPoint>.ack.out.totalFailed
执行负面应答，使用状态 3，忽略刀具位置参数和状态参数	

数据类型和更新

BOOL

12.3.99 <LoadingPoint>.command.in

12.3.99.1 multitoolRequest

NC → PLC	<LoadingPoint>.command.in.multitoolRequest
已请求使用 Multitool 装载/卸载，用于 Multitool 的信号有效。只有为 <LoadingPoint>.command.in.request 赋值后才有效	

数据类型与更新

BOOL

值 TRUE

已请求使用 Multitool 装载/卸载，用于 Multitool 的信号有效。只有为 <LoadingPoint>.command.in.request 赋值后才有效

12.3.99.2 request

NC → PLC	<LoadingPoint>.command.in.request
NC 请求传输刀具，指令信号有效	

数据类型与更新

BOOL

值 TRUE

NC 请求传输刀具，指令信号有效

12.3.99.3 userParameters

NC → PLC	<LoadingPoint>.command.in.userParameters
使用指令 ($P_VDITCP) 传递的用户自定义参数。只有为 <LoadingPoint>.command.in.request 赋值后才有效	

数据类型与更新

DInt

值

使用指令 ($P_VDITCP) 传递的用户自定义参数。只有为 <LoadingPoint>.command.in.request 赋值后才有效

12.3.100 <LoadingPoint>.command.in.basic

12.3.100.1 chan

NC → PLC	<LoadingPoint>.command.in.basic.chan
相关通道编号	

数据类型与更新

USInt

值

相关通道编号

12.3.100.2 load

NC → PLC	<LoadingPoint>.command.in.basic.load
已请求装载一个刀具	

数据类型与更新

BOOL

值 TRUE

已请求装载一个刀具

值 FALSE

未请求装载刀具

12.3.100.3 position

NC → PLC	<LoadingPoint>.command.in.basic.position
已请求在装载位“定位刀库”	

数据类型与更新

BOOL

值 TRUE

已请求在装载位“定位刀库”

12.3.100.4 toolSourceLocation

NC → PLC	<LoadingPoint>.command.in.basic.toolSourceLocation
刀具所处刀位编号，或者必须定位在装载位的刀位编号	

数据类型与更新

Int

值

刀具所处刀位编号，或者必须定位在装载位的刀位编号。

12.3.100.5 toolSourceMagazine

NC → PLC	<LoadingPoint>.command.in.basic.toolSourceMagazine
刀具所处刀库编号，或者必须定位在装载位的刀库编号	

数据类型和更新

Int

值

刀具所处刀库编号，或者必须定位在装载位的刀库编号。

12.3.100.6 toolTargetLocation

NC → PLC	<LoadingPoint>.command.in.basic.toolTargetLocation
刀具目标刀位编号，或者要定位的装载位刀位编号	

数据类型与更新

Int

值 TRUE

刀具目标刀位编号，或者要定位的装载位刀位编号

12.3.100.7 toolTargetMagazine

NC → PLC	<LoadingPoint>.command.in.basic.toolTargetMagazine
刀具目标刀库编号，或者要定位的装载位刀库编号	

数据类型与更新

Int

值

刀具目标刀库编号，或者要定位的装载位刀库编号。

12.3.100.8 unload

NC → PLC	<LoadingPoint>.command.in.basic.unload
已请求卸载一个刀具	

数据类型与更新

BOOL

值 TRUE

已请求卸载一个刀具

值 FALSE

未请求卸载刀具

12.3.101 <LoadingPoint>.command.in.options

12.3.101.1 byNcProg

NC → PLC	<loadingPoint>.command.in.options.byNcProg
刀具传输请求已通过 NC 程序初始化	

数据类型和更新

BOOL

12.3.102 <LoadingPoint>.command.in.toolProperties

12.3.102.1 toolLocationType

NC → PLC	<loadingPoint>.command.in.toolProperties.toolLocationType
刀具的刀位类型	

数据类型和更新

UInt

12.3.102.2 toolSizeBottom

NC → PLC	<loadingPoint>.command.in.toolProperties.toolSizeBottom
刀具向下使用的半刀位个数	

数据类型和更新

UInt

12.3.102.3 toolSizeLeft

NC → PLC	<loadingPoint>.command.in.toolProperties.toolSizeLeft
刀具向左使用的半刀位个数	

数据类型和更新

UInt

12.3.102.4 toolSizeRight

NC → PLC	<loadingPoint>.command.in.toolProperties.toolSizeRight
刀具向右使用的半刀位个数	

数据类型和更新

UInt

12.3.102.5 toolSizeTop

NC → PLC	<loadingPoint>.command.in.toolProperties.toolSizeTop
刀具向上使用的半刀位个数	

数据类型和更新

UInt

12.3.103 <LoadingPoint>.command.in.toolStatus

12.3.103.1 active

NC → PLC	<LoadingPoint>.command.in.toolStatus.active
刀具处于活动状态	

数据类型和更新

BOOL

值 TRUE

刀具处于活动状态

值 FALSE

刀具未激活

12.3.103.2 changeActive

NC → PLC	<LoadingPoint>.command.in.toolStatus.changeActive
更换刀具	

数据类型和更新

BOOL

值 TRUE

更换刀具

值 FALSE

不更换刀具

12.3.103.3 changeOneToOne

NC → PLC	<LoadingPoint>.command.in.toolStatus.changeOneToOne
1:1 更换刀具	

数据类型和更新

BOOL

12.3.103.4 disabled

NC → PLC	<LoadingPoint>.command.in.toolStatus.disabled
刀具已禁用	

数据类型和更新

BOOL

值 TRUE

刀具已禁用

值 FALSE

刀具未锁定

12.3.103.5 disableStateIgnored

NC → PLC	<LoadingPoint>.command.in.toolStatus.disableStateIgnored
忽略刀具锁定状态	

数据类型和更新

BOOL

值 TRUE

忽略刀具锁定状态

值 FALSE

不忽略刀具锁定状态

12.3.103.6 enabled

NC → PLC	<LoadingPoint>.command.in.toolStatus.enabled
刀具正在被监控	

数据类型和更新

BOOL

值 TRUE

刀具正在被监控

值 FALSE

刀具未使能

12.3.103.7 fixedLocation

NC → PLC	<LoadingPoint>.command.in.toolStatus.fixedLocation
刀具已进行固定位置编码	

数据类型和更新

BOOL

值 TRUE

刀具已进行固定位置编码

值 FALSE

刀具未进行固定位置编码

12.3.103.8 manualTool

NC → PLC	<LoadingPoint>.command.in.toolStatus.manualTool
刀具为手动更换刀具	

数据类型和更新

BOOL

12.3.103.9 masterTool

NC → PLC	<LoadingPoint>.command.in.toolStatus.masterTool
刀具为主刀具	

数据类型和更新

BOOL

值 TRUE

刀具为主刀具

值 FALSE

刀具不是主刀具

12.3.103.10 measured

NC → PLC	<LoadingPoint>.command.in.toolStatus.measured
刀具已测量	

数据类型和更新

BOOL

值 TRUE

刀具已测量

值 FALSE

刀具未测量

12.3.103.11 prewarning

NC → PLC	<LoadingPoint>.command.in.toolStatus.prewarning
达到刀具预警极限	

数据类型和更新

BOOL

值 TRUE

达到刀具预警极限

值 FALSE

未达到刀具预警极限

12.3.103.12 toBeLoaded

NC → PLC	<LoadingPoint>.command.in.toolStatus.toBeLoaded
必须装载刀具	

数据类型和更新

BOOL

值 TRUE

必须装载刀具

值 FALSE

无需装载刀具

12.3.103.13 toBeUnloaded

NC → PLC	<LoadingPoint>.command.in.toolStatus.toBeUnloaded
必须卸载刀具	

数据类型和更新

BOOL

值 TRUE

必须卸载刀具

值 FALSE

无需卸载刀具

12.3.103.14 toolAtDisabledLocation

NC → PLC	<LoadingPoint>.command.in.toolStatus.toolAtDisabledLocation
刀具处于锁定的刀库刀位	

数据类型和更新

BOOL

12.3.103.15 toolInBuffer

NC → PLC	<LoadingPoint>.command.in.toolStatus.toolInBuffer
刀具位于缓存刀库中	

数据类型和更新

BOOL

值 TRUE

刀具位于缓存刀库中

值 FALSE

刀具不在缓存刀库中

12.3.103.16 used

NC → PLC	<LoadingPoint>.command.in.toolStatus.used
刀具已使用	

数据类型和更新

BOOL

值 TRUE

刀具已使用

值 FALSE

刀具未使用

12.3.104 <LoadingPoint>.command.in.multitoolProperties

12.3.104.1 distance

NC → PLC	<loadingPoint>.command.in.multitoolProperties.distance
多刀的状态。仅当已设置 <loadingPoint>.command.in.multitoolRequest 时有效	

数据类型和更新

Real

12.3.104.2 distanceType

NC → PLC	<loadingPoint>.command.in.multitoolProperties.distanceType
多刀的属性。仅当已设置 <loadingPoint>.command.in.multitoolRequest 时有效	

数据类型和更新

UInt

12.3.104.3 numberOfLocations

NC → PLC	<loadingPoint>.command.in.multitoolProperties.numberOfLocations
NC 指令的内部编号	

数据类型和更新

UInt

12.3.105 NCK_typeRelocate

下表说明了数据类型 NCK_typeRelocate 的含义：

结构	说明
.config	用于切换的接口的组态数据
.registered	用于切换的接口已注册
.toaUnit	对应的工艺目标范围的编号
.ack.in (页 1040)	用于应答刀具传输的输入信号
.ack.out (页 1041)	用于应答刀具传输的输出信号
.command.in (页 1043)	来自刀具管理的信号
.command.in.basic (页 1044)	通用指令信号。仅在已设置<relocate>.command.in.request的情况下有效
.command.in.options (页 1046)	指令的选项。仅在已设置<relocate>.command.in.request的情况下有效
.command.in.toolProperties (页 1047)	刀具的属性。仅在已设置<relocate>.command.in.request的情况下有效
.command.in.toolStatus (页 1048)	刀具的状态。仅在已设置<relocate>.command.in.request的情况下有效
.command.in.multitoolProperties (页 1054)	多刀的属性，仅在已设置<relocate>.command.in.multitoolRequestt 时有效

12.3.106 <Relocate>.ack.in

12.3.106.1 busy

NC → PLC	<Relocate>.ack.in.busy
正在应答	

数据类型和更新

BOOL

12.3.106.2 完成

NC → PLC	<Relocate>.ack.in.done
应答成功	

数据类型和更新

BOOL

12.3.106.3 error

NC → PLC	<Relocate>.ack.in.error
应答被拒，见状态	

数据类型和更新

BOOL

12.3.106.4 status

NC → PLC	<Relocate>.ack.in.status
应答失败后的状态信息	

数据类型和更新

BOOL

12.3.107 <Relocate>.ack.out

12.3.107.1 execute

PLC → NC	<Relocate>.ack.out.execute
开始应答移刀	

数据类型和更新

BOOL

12.3.107.2 status

PLC → NC	<Relocate>.ack.out.status
刀具传输状态	

状态信息

包含应答状态 1–10 的信息结束指令，包含应答状态 105 的信息

使指令继续保持活动状态。

说明

1 刀具管理任务已完成。

3 出错。

5 完成定位或新刀具到达指定位置。

6 同 1，但预留初始刀位。

10 同 1，但刀库位置不变

105 刀具在指定位置。

数据类型和更新

USInt

12.3.107.3 toolLocation

PLC → NC	<Relocate>.ack.out.toolLocation
刀具当前所处刀库刀位编号	

数据类型和更新

Int

12.3.107.4 toolMagazine

PLC → NC	<Relocate>.ack.out.toolMagazine
刀具当前所处刀库编号	

数据类型和更新

Int

12.3.107.5 totalComplete

PLC → NC	<Relocate>.ack.out.totalComplete
执行正面应答，使用状态 1 或 5 或 6，忽略刀具位置参数和状态参数	

数据类型与更新

BOOL

12.3.107.6 totalFailed

PLC → NC	<Relocate>.ack.out.totalFailed
执行负面应答，使用状态 3，忽略刀具位置参数和状态参数	

数据类型与更新

BOOL

12.3.108 <Relocate>.command.in

12.3.108.1 multitoolRequest

NC → PLC	<Relocate>.command.in.multitoolRequest
已请求使用 Multitool 移刀，用于 Multitool 的指令信号有效。只有为 <Relocate>.command.in.request 赋值后才有效	

数据类型与更新

BOOL

值 TRUE

已请求使用 Multitool 移刀，用于 Multitool 的指令信号有效。只有为 <Relocate>.command.in.request 赋值后才有效

12.3.108.2 request

NC → PLC	<Relocate>.command.in.request
NC 请求移刀，指令信号有效	

数据类型与更新

BOOL

值 TRUE

NC 请求移刀，指令信号有效

12.3.108.3 userParameters

NC → PLC	<Relocate>.command.in.userParameters
使用指令 ($P_VDITCP) 传递的用户自定义参数。只有为 <Relocate>.command.in.request 赋值后才有效	

数据类型与更新

DInt

12.3.109 <Relocate>.command.in.basic

12.3.109.1 chan

NC → PLC	<Relocate>.command.in.basic.chan
相关通道编号	

数据类型与更新

USInt

值

相关通道编号

12.3.109.2 relocate

NC → PLC	<Relocate>.command.in.basic.relocate
已请求移刀	

数据类型与更新

BOOL

值 TRUE

已请求移刀

12.3.109.3 tool

NC → PLC	<Relocate>.command.in.basic.tool
刀具的刀具号	

数据类型和更新

UInt

12.3.109.4 toolSourceLocation

NC → PLC	<Relocate>.command.in.basic.toolSourceLocation
刀具所处刀位编号	

数据类型与更新

Int

值

刀具所处刀位编号

12.3.109.5 toolSourceMagazine

NC → PLC	<Relocate>.command.in.basic.toolSourceMagazine
刀具所处刀库编号	

数据类型与更新

Int

值

刀具所处刀库编号

12.3.109.6 toolTargetLocation

NC → PLC	<Relocate>.command.in.basic.toolTargetLocation
刀具目标刀位编号	

数据类型与更新

Int

值 TRUE

刀具目标刀位编号

12.3.109.7 toolTargetMagazine

NC → PLC	<Relocate>.command.in.basic.toolTargetMagazine
刀具目标刀库编号	

数据类型与更新

Int

值

刀具目标刀库编号

12.3.110 <Relocate>.command.in.options

12.3.110.1 byNcProg

NC → PLC	<Relocate>.command.in.options.byNcProg
刀具传输请求已通过 NC 程序初始化	

数据类型与更新

BOOL

值 TRUE

刀具传输请求已通过 NC 程序初始化

值 FALSE

刀具传输请求未通过 NC 程序初始化

12.3.111 <Relocate>.command.in.toolProperties

12.3.111.1 toolLocationType

NC → PLC	<Relocate>.command.in.toolProperties.toolLocationType
刀具的刀位类型	

数据类型和更新

UInt

12.3.111.2 toolSizeBottom

NC → PLC	<Relocate>.command.in.toolProperties.toolSizeBottom
刀具向下使用的半刀位个数	

数据类型和更新

UInt

12.3.111.3 toolSizeLeft

NC → PLC	<Relocate>.command.in.toolProperties.toolSizeLeft
刀具向左使用的半刀位个数	

数据类型和更新

UInt

12.3.111.4 toolSizeRight

NC → PLC	<Relocate>.command.in.toolProperties.toolSizeRight
刀具向右使用的半刀位个数	

数据类型和更新

UInt

12.3.111.5 toolSizeTop

NC → PLC	<Relocate>.command.in.toolProperties.toolSizeTop
刀具向上使用的半刀位个数	

数据类型和更新

UInt

12.3.112 <Relocate>.command.in.toolStatus

12.3.112.1 active

NC → PLC	<Relocate>.command.in.toolStatus.active
刀具处于活动状态	

数据类型和更新

BOOL

值 TRUE

刀具处于活动状态

值 FALSE

刀具未激活

12.3.112.2 changeActive

NC → PLC	<Relocate>.command.in.toolStatus.changeActive
更换刀具	

数据类型和更新

BOOL

值 TRUE

更换刀具

值 FALSE

不更换刀具

12.3.112.3 changeOneToOne

NC → PLC	<Relocate>.command.in.toolStatus.changeOneToOne
1:1 更换刀具	

数据类型和更新

BOOL

12.3.112.4 disabled

NC → PLC	<Relocate>.command.in.toolStatus.disabled
刀具已禁用	

数据类型和更新

BOOL

值 TRUE

刀具已禁用

值 FALSE

刀具未锁定

12.3.112.5 disableStateIgnored

NC → PLC	<Relocate>.command.in.toolStatus.disableStateIgnored
忽略刀具锁定状态	

数据类型和更新

BOOL

值 TRUE

忽略刀具锁定状态

值 FALSE

不忽略刀具锁定状态

12.3.112.6 enabled

NC → PLC	<Relocate>.command.in.toolStatus.enabled
刀具正在被监控	

数据类型和更新

BOOL

值 TRUE

刀具正在被监控

值 FALSE

刀具未使能

12.3.112.7 fixedLocation

NC → PLC	<Relocate>.command.in.toolStatus.fixedLocation
刀具已进行固定位置编码	

数据类型和更新

BOOL

值 TRUE

刀具已进行固定位置编码

值 FALSE

刀具未进行固定位置编码

12.3.112.8 manualTool

NC → PLC	<Relocate>.command.in.toolStatus.manualTool
刀具为手动更换刀具	

数据类型和更新

BOOL

12.3.112.9 masterTool

NC → PLC	<Relocate>.command.in.toolStatus.masterTool
刀具为主刀具	

数据类型和更新

BOOL

值 TRUE

刀具为主刀具

值 FALSE

刀具不是主刀具

12.3.112.10 measured

NC → PLC	<Relocate>.command.in.toolStatus.measured
刀具已测量	

数据类型和更新

BOOL

值 TRUE

刀具已测量

值 FALSE

刀具未测量

12.3.112.11 prewarning

NC → PLC	<Relocate>.command.in.toolStatus.prewarning
达到刀具预警极限	

数据类型和更新

BOOL

值 TRUE

达到刀具预警极限

值 FALSE

未达到刀具预警极限

12.3.112.12 toBeLoaded

NC → PLC	<Relocate>.command.in.toolStatus.toBeLoaded
必须装载刀具	

数据类型和更新

BOOL

值 TRUE

必须装载刀具

值 FALSE

无需装载刀具

12.3.112.13 toBeUnloaded

NC → PLC	<Relocate>.command.in.toolStatus.toBeUnloaded
必须卸载刀具	

数据类型和更新

BOOL

值 TRUE

必须卸载刀具

值 FALSE

无需卸载刀具

12.3.112.14 toolAtDisabledLocation

NC → PLC	<Relocate>.command.in.toolStatus.toolAtDisabledLocation
刀具处于锁定的刀库刀位	

数据类型和更新

BOOL

12.3.112.15 toolInBuffer

NC → PLC	<Relocate>.command.in.toolStatus.toolInBuffer
刀具位于缓存刀库中	

数据类型和更新

BOOL

值 TRUE

刀具位于缓存刀库中

值 FALSE

刀具不在缓存刀库中

12.3.112.16 used

NC → PLC	<Relocate>.command.in.toolStatus.used
刀具已使用	

数据类型和更新

BOOL

值 TRUE

刀具已使用

值 FALSE

刀具未使用

12.3.113 <Relocate>.command.in.multitoolProperties

12.3.113.1 distance

NC → PLC	<Relocate>.command.in.multitoolProperties.distance
根据间距编码方式，Multitool 内的刀位间距	

数据类型与更新

Real

值

根据间距编码方式，Multitool 内的刀位间距

12.3.113.2 distanceType

NC → PLC	<Relocate>.command.in.multitoolProperties.distanceType
Multitool 的间距编码方式	

数据类型与更新

UInt

12.3.113.3 numberOfLocations

NC → PLC	<Relocate>.command.in.multitoolProperties.numberOfLocations
多刀的位置数量	

数据类型与更新

UInt

值

多刀的位置数量

12.3.114 NCK_typeToolholder

下表说明了数据类型 NCK_typeToolholder 的含义：

结构	说明
.config	刀架的组态数据
.registered	刀架已注册
.toolholder	刀夹编号
.toaUnit	对应的工艺目标范围的编号
.bufferMagazine (页 1056)	刀架所在的刀库的编号，通常是暂时存储刀库
.bufferLocation (页 1056)	刀库中刀套的位置编号
.command.in (页 1061)	来自刀具管理的信号
.ack.in (页 1057)	用于应答刀具更换的输入信号
.ack.out (页 1058)	用于应答刀具更换的输出信号
.command.in.basic (页 1062)	通用指令信号。仅在已设置 <toolholder>.command.in.request 的情况下有效
.command.in.options (页 1065)	指令的选项。仅在已设置 <toolholder>.command.in.request 的情况下有效
.command.in.toolProperties (页 1066)	刀架中新刀具的属性。仅在已设置 <toolholder>.command.in.request 的情况下有效
.command.in.toolStatus (页 1068)	刀架中新刀具的状态。仅在已设置 <toolholder>.command.in.request 的情况下有效
.command.in.multitoolCommand (页 1074)	多刀的通用指令信号。仅当已设置 <toolholder>.command.in.multitoolRequest 时有效

结构	说明
.command.in.multitoolProperties (页 1075)	刀架中新多刀的属性。仅当已设置 <toolholder>.command.in.multitoolRequest 时有效
.command.in.multitoolStatus (页 1078)	刀架中新多刀的状态。仅当已设置 <toolholder>.command.in.multitoolRequest 时有效

12.3.115 <Toolholder>.config

12.3.115.1 bufferLocation

NC → PLC	<Toolholder>.config.bufferLocation
正在应答	

说明

在注册了刀柄实例后，此处提供了缓存刀库中刀夹的刀位编号。在 <Toolholder>.config.bufferMagazine 中可提供缓存刀库的刀库编号。

数据类型和更新

UInt，一次

值

刀位号

关联

- <Toolholder>.config.bufferMagazine
  刀夹所处刀库编号（缓存刀库）

更多信息

- 功能手册：SINUMERIK ONE 刀具管理

12.3.115.2 bufferMagazine

NC → PLC	<Toolholder>.config.bufferMagazine
正在应答	

说明

在注册了刀柄实例后，此处提供了缓存刀库的刀库编号。在 <Toolholder>.config.bufferLocation 中可提供缓存刀库中刀夹的刀位编号。

数据类型和更新

UInt，一次

值

刀库编号

关联

- <Toolholder>.config.bufferLocation
  缓存刀库中刀夹的刀位编号

更多信息

- 功能手册：SINUMERIK ONE 刀具管理

12.3.116 <Toolholder>.ack.in

12.3.116.1 busy

NC → PLC	<Toolholder>.ack.in.busy
正在应答	

数据类型和更新

BOOL

12.3.116.2 完成

NC → PLC	<Toolholder>.ack.in.done
应答成功	

数据类型和更新

BOOL

12.3.116.3 error

NC → PLC	<Toolholder>.ack.in.error
应答被拒，见状态	

数据类型和更新

BOOL

12.3.116.4 status

NC → PLC	<Toolholder>.ack.in.status
应答失败后的状态信息	

数据类型和更新

Word

12.3.117 <Toolholder>.ack.out

12.3.117.1 execute

PLC → NC	<Toolholder>.ack.out.execute
开始应答换刀	

数据类型和更新

BOOL

12.3.117.2 newMultitoolLocation

PLC → NC	<Toolholder>.ack.out.newMultitoolLocation
新刀具 Multitool 内的当前刀位编号	

数据类型和更新

Int

12.3.117.3 newToolLocation

PLC → NC	<Toolholder>.ack.out.newToolLocation
新刀具当前所处刀库刀位编号	

数据类型和更新

Int

12.3.117.4 newToolMagazine

PLC → NC	<Toolholder>.ack.out.newToolMagazine
新刀具当前所处刀库编号	

数据类型和更新

Int

12.3.117.5 oldToolLocation

PLC → NC	<Toolholder>.ack.out.oldToolLocation
老刀具当前所处刀库刀位编号	

数据类型和更新

Int

12.3.117.6 oldToolMagazine

PLC → NC	<Toolholder>.ack.out.oldToolMagazine
老刀具当前所处刀库编号	

数据类型和更新

Int

12.3.117.7 status

PLC → NC	<Toolholder>.ack.out.status
换刀状态	

状态信息

包含应答状态 1..7 的信号结束指令，包含应答状态 103..105 的信号使指令继续保持活动状态。

说明

1：刀具管理任务已完成。

2:无法提供新刀具。

3：出错。

4：旧刀具最好安置在 oldToolMagazine 和 oldToolLocation 中。

5：完成定位或新刀具到达指定位置。

7:重复“准备换刀”指令。

103:可装入新刀具。

104:就位或新刀具到达指定位置。

105:所有相关刀具都已到达指定刀位。

数据类型和更新

USInt

12.3.117.8 totalComplete

PLC → NC	<Toolholder>.ack.out.totalComplete
执行正面应答，使用状态 1 或 5，忽略刀具位置参数和状态参数	

数据类型与更新

BOOL

12.3.117.9 totalFailed

PLC → NC	<Toolholder>.ack.out.totalFailed
执行负面应答，使用状态 3，忽略刀具位置参数和状态参数	

数据类型与更新

BOOL

12.3.118 <Toolholder>.command.in

12.3.118.1 multitoolRequest

NC → PLC	<Toolholder>.command.in.multitoolRequest
已请求使用 Multitool 换刀，用于 Multitool 的信号有效。只有为 <Toolholder>.command.in.request 赋值后才有效	

数据类型与更新

BOOL，循环

12.3.118.2 request

NC → PLC	<Toolholder>.command.in.request
NC 请求换刀，指令信号有效	

数据类型与更新

BOOL

12.3.118.3 userParameters

NC → PLC	<Toolholder>.command.in.userParameters
使用指令 ($P_VDITCP) 传递的用户自定义参数。只有为<Toolholder>.command.in.request 赋值后才有效	

数据类型和更新

DInt

12.3.119 <Toolholder>.command.in.basic

12.3.119.1 chan

NC → PLC	<Toolholder>.command.in.basic.chan
相关通道编号	

数据类型和更新

USInt

12.3.119.2 newTool

NC → PLC	<Toolholder>.command.in.basic.newTool
新刀具的刀具号	

数据类型与更新

UInt

12.3.119.3 newToolSourceLocation

NC → PLC	<Toolholder>.command.in.basic.newToolSourceLocation
新刀具所处刀位编号，或者必须定位在刀夹的刀位编号	

数据类型与更新

Int

12.3.119.4 newToolSourceMagazine

NC → PLC	<Toolholder>.command.in.basic.newToolSourceMagazine
新刀具所处刀库编号，或者必须定位在刀夹的刀库编号	

数据类型与更新

Int

12.3.119.5 newToolTargetLocation

NC → PLC	<Toolholder>.command.in.basic.newToolTargetLocation
新刀具目标刀位编号，或者所需定位的刀夹刀位编号	

数据类型与更新

Int

12.3.119.6 newToolTargetMagazine

NC → PLC	<Toolholder>.command.in.basic.newToolTargetMagazine
新刀具目标刀库编号，或者所需定位的刀夹刀库编号	

数据类型和更新

Int

12.3.119.7 oldToolSourceLocation

NC → PLC	<toolholder>.command.in.basic.oldToolSourceLocation
老刀具所处刀位编号	

数据类型和更新

INT

12.3.119.8 oldToolSourceMagazine

NC → PLC	<toolholder>.command.in.basic.oldToolSourceMagazine
老刀具所处的刀库编号	

数据类型和更新

INT

12.3.119.9 oldToolTargetLocation

NC → PLC	<Toolholder>.command.in.basic.oldToolTargetLocation
老刀具目标刀位编号	

数据类型与更新

Int

12.3.119.10 oldToolTargetMagazine

NC → PLC	<Toolholder>.command.in.basic.oldToolTargetMagazine
老刀具目标刀库编号	

数据类型与更新

Int

12.3.119.11 perform

NC → PLC	<Toolholder>.command.in.basic.perform
已请求“执行换刀”	

数据类型与更新

BOOL

12.3.119.12 position

NC → PLC	<Toolholder>.command.in.basic.position
已请求在刀夹“定位刀库”	

数据类型与更新

BOOL

12.3.119.13 prepare

NC → PLC	<Toolholder>.command.in.basic.prepare
已请求“准备换刀”	

数据类型与更新

BOOL

12.3.120 <Toolholder>.command.in.options

12.3.120.1 byNcProg

NC → PLC	<Toolholder>.command.in.options.byNcProg
某个 NC 程序已触发换刀请求。	

数据类型与更新

BOOL

12.3.120.2 newManualTool

NC → PLC	<Toolholder>.command.in.options.newManualTool
必须手动装载新刀具（手动更换刀具）	

数据类型与更新

BOOL

12.3.120.3 noNewTool

NC → PLC	<Toolholder>.command.in.options.noNewTool
换刀时不使用新刀具，已编程设置 T0	

数据类型与更新

BOOL

12.3.120.4 oldManualTool

NC → PLC	<Toolholder>.command.in.options.oldManualTool
必须手动更换老刀具（手动更换刀具）	

数据类型与更新

BOOL

12.3.120.5 oldToolNotInToolholder

NC → PLC	<Toolholder>.command.in.options.oldToolNotInToolholder
老刀具不再位于刀夹内	

数据类型与更新

BOOL

12.3.120.6 toolRemainsInToolholder

NC → PLC	<Toolholder>.command.in.options.toolRemainsInToolholder
新刀具已在刀夹中	

数据类型与更新

BOOL

12.3.121 <Toolholder>.command.in.toolProperties

12.3.121.1 toolLocationType

NC → PLC	<Toolholder>.command.in.toolProperties.toolLocationType
刀具的刀位类型	

数据类型与更新

UInt

12.3.121.2 toolOriginLocation

NC → PLC	<Toolholder>.command.in.toolProperties.toolOriginLocation
刀库内所有者刀位的刀位编号	

数据类型与更新

UInt

12.3.121.3 toolOriginMagazine

NC → PLC	<Toolholder>.command.in.toolProperties.toolOriginMagazine
所有者刀库的刀具编号	

数据类型与更新

UInt

12.3.121.4 toolSizeBottom

NC → PLC	<Toolholder>.command.in.toolProperties.toolSizeBottom
刀具向下使用的半刀位个数	

数据类型与更新

UInt

12.3.121.5 toolSizeLeft

NC → PLC	<Toolholder>.command.in.toolProperties.toolSizeLeft
刀具向左使用的半刀位个数	

数据类型与更新

UInt

12.3.121.6 toolSizeRight

NC → PLC	<Toolholder>.command.in.toolProperties.toolSizeRight
刀具向右使用的半刀位个数	

数据类型与更新

UInt

12.3.121.7 toolSizeTop

NC → PLC	<Toolholder>.command.in.toolProperties.toolSizeTop
刀具向上使用的半刀位个数	

数据类型与更新

UInt

12.3.122 <Toolholder>.command.in.toolStatus

12.3.122.1 active

NC → PLC	<Toolholder>.command.in.toolStatus.active
刀具处于活动状态	

数据类型和更新

BOOL

值 TRUE

刀具处于活动状态

值 FALSE

刀具未激活

12.3.122.2 changeActive

NC → PLC	<Toolholder>.command.in.toolStatus.changeActive
更换刀具	

数据类型和更新

BOOL

值 TRUE

更换刀具

值 FALSE

不更换刀具

12.3.122.3 changeOneToOne

NC → PLC	<Toolholder>.command.in.toolStatus.changeOneToOne
1:1 更换刀具	

数据类型和更新

BOOL

12.3.122.4 disabled

NC → PLC	<Toolholder>.command.in.toolStatus.disabled
刀具已禁用	

数据类型和更新

BOOL

值 TRUE

刀具已禁用

值 FALSE

刀具未锁定

12.3.122.5 disableStateIgnored

NC → PLC	<Toolholder>.command.in.toolStatus.disableStateIgnored
忽略刀具锁定状态	

数据类型和更新

BOOL

值 TRUE

忽略刀具锁定状态

值 FALSE

不忽略刀具锁定状态

12.3.122.6 enabled

NC → PLC	<Toolholder>.command.in.toolStatus.enabled
刀具正在被监控	

数据类型和更新

BOOL

值 TRUE

刀具正在被监控

值 FALSE

刀具未使能

12.3.122.7 fixedLocation

NC → PLC	<Toolholder>.command.in.toolStatus.fixedLocation
刀具已进行固定位置编码	

数据类型和更新

BOOL

值 TRUE

刀具已进行固定位置编码

值 FALSE

刀具未进行固定位置编码

12.3.122.8 manualTool

NC → PLC	<Toolholder>.command.in.toolStatus.manualTool
刀具为手动更换刀具	

数据类型和更新

BOOL

12.3.122.9 masterTool

NC → PLC	<Toolholder>.command.in.toolStatus.masterTool
刀具为主刀具	

数据类型和更新

BOOL

值 TRUE

刀具为主刀具

值 FALSE

刀具不是主刀具

12.3.122.10 measured

NC → PLC	<Toolholder>.command.in.toolStatus.measured
刀具已测量	

数据类型和更新

BOOL

值 TRUE

刀具已测量

值 FALSE

刀具未测量

12.3.122.11 prewarning

NC → PLC	<Toolholder>.command.in.toolStatus.prewarning
达到刀具预警极限	

数据类型和更新

BOOL

值 TRUE

达到刀具预警极限

值 FALSE

未达到刀具预警极限

12.3.122.12 toBeLoaded

NC → PLC	<Toolholder>.command.in.toolStatus.toBeLoaded
必须装载刀具	

数据类型和更新

BOOL

值 TRUE

必须装载刀具

值 FALSE

无需装载刀具

12.3.122.13 toBeUnloaded

NC → PLC	<Toolholder>.command.in.toolStatus.toBeUnloaded
必须卸载刀具	

数据类型和更新

BOOL

值 TRUE

必须卸载刀具

值 FALSE

无需卸载刀具

12.3.122.14 toolAtDisabledLocation

NC → PLC	<Toolholder>.command.in.toolStatus.toolAtDisabledLocation
刀具处于锁定的刀库刀位	

数据类型和更新

BOOL

12.3.122.15 toolInBuffer

NC → PLC	<Toolholder>.command.in.toolStatus.toolInBuffer
刀具位于缓存刀库中	

数据类型和更新

BOOL

值 TRUE

刀具位于缓存刀库中

值 FALSE

刀具不在缓存刀库中

12.3.122.16 used

NC → PLC	<Toolholder>.command.in.toolStatus.used
刀具已使用	

数据类型和更新

BOOL

值 TRUE

刀具已使用

值 FALSE

刀具未使用

12.3.123 <Toolholder>.command.in.multitoolCommand

12.3.123.1 newTool

NC → PLC	<Toolholder>.command.in.multitoolCommand.newTool
新 Multitool 的刀具号	

数据类型与更新

UInt

12.3.123.2 newToolLocation

NC → PLC	<Toolholder>.command.in.multitoolCommand.newToolLocation
新刀具所处 Multitool 内的刀位编号	

数据类型与更新

Int

12.3.123.3 oldTool

NC → PLC	<Toolholder>.command.in.multitoolCommand.oldTool
老 Multitool 的刀具号	

数据类型与更新

UInt

12.3.123.4 oldToolLocation

NC → PLC	<Toolholder>.command.in.multitoolCommand.oldToolLocation
老刀具所处 Multitool 内的刀位编号	

数据类型与更新

Int

12.3.123.5 position

NC → PLC	<Toolholder>.command.in.multitoolCommand.position
请求定位 Multitool	

数据类型与更新

BOOL

12.3.124 <Toolholder>.command.in.multitoolProperties

12.3.124.1 distance

NC → PLC	<Toolholder>.command.in.multitoolProperties.distance
根据间距编码方式，Multitool 内的刀位间距	

数据类型与更新

Real

12.3.124.2 distanceType

NC → PLC	<Toolholder>.command.in.multitoolProperties.distanceType
Multitool 的间距编码方式	

数据类型与更新

UInt

12.3.124.3 numberOfLocations

NC → PLC	<Toolholder>.command.in.multitoolProperties.numberOfLocations
多刀的位置数量	

数据类型与更新

UInt

12.3.124.4 toolLocationType

NC → PLC	<Toolholder>.command.in.multitoolProperties.toolLocationType
Multitool 的刀位类型	

数据类型与更新

UInt

12.3.124.5 toolOriginLocation

NC → PLC	<Toolholder>.command.in.multitoolProperties.toolOriginLocation
刀库内所有者刀位的刀位编号	

数据类型与更新

UInt

12.3.124.6 toolOriginMagazine

NC → PLC	<Toolholder>.command.in.multitoolProperties.toolOriginMagazine
Multitool 的所有者刀库的刀库编号	

数据类型与更新

UInt

12.3.124.7 toolSizeBottom

NC → PLC	<Toolholder>.command.in.multitoolProperties.toolSizeBottom
Multitool 向下使用的半刀位个数	

数据类型与更新

UInt

12.3.124.8 toolSizeLeft

NC → PLC	<Toolholder>.command.in.multitoolProperties.toolSizeLeft
Multitool 向左使用的半刀位个数	

数据类型与更新

UInt

12.3.124.9 toolSizeRight

NC → PLC	<Toolholder>.command.in.multitoolProperties.toolSizeRight
Multitool 向右使用的半刀位个数	

数据类型与更新

UInt

12.3.124.10 toolSizeTop

NC → PLC	<Toolholder>.command.in.multitoolProperties.toolSizeTop
Multitool 向上使用的半刀位个数	

数据类型与更新

UInt

12.3.125 <Toolholder>.command.in.multitoolStatus

12.3.125.1 active

NC → PLC	<Toolholder>.command.in.multitoolStatus.active
刀具处于活动状态	

数据类型和更新

BOOL

值 TRUE

刀具处于活动状态

值 FALSE

刀具未激活

12.3.125.2 changeActive

NC → PLC	<Toolholder>.command.in.multitoolStatus.changeActive
更换刀具	

数据类型和更新

BOOL

值 TRUE

更换刀具

值 FALSE

不更换刀具

12.3.125.3 changeOneToOne

NC → PLC	<Toolholder>.command.in.multitoolStatus.changeOneToOne
1:1 更换刀具	

数据类型和更新

BOOL

12.3.125.4 disabled

NC → PLC	<Toolholder>.command.in.multitoolStatus.disabled
刀具已禁用	

数据类型和更新

BOOL

值 TRUE

刀具已禁用

值 FALSE

刀具未锁定

12.3.125.5 disableStateIgnored

NC → PLC	<Toolholder>.command.in.multitoolStatus.disableStateIgnored
忽略刀具锁定状态	

数据类型和更新

BOOL

值 TRUE

忽略刀具锁定状态

值 FALSE

不忽略刀具锁定状态

12.3.125.6 enabled

NC → PLC	<Toolholder>.command.in.multitoolStatus.enabled
刀具正在被监控	

数据类型和更新

BOOL

值 TRUE

刀具正在被监控

值 FALSE

刀具未使能

12.3.125.7 fixedLocation

NC → PLC	<Toolholder>.command.in.multitoolStatus.fixedLocation
刀具已进行固定位置编码	

数据类型和更新

BOOL

值 TRUE

刀具已进行固定位置编码

值 FALSE

刀具未进行固定位置编码

12.3.125.8 manualTool

NC → PLC	<Toolholder>.command.in.multitoolStatus.manualTool
刀具为手动更换刀具	

数据类型和更新

BOOL

12.3.125.9 masterTool

NC → PLC	<Toolholder>.command.in.multitoolStatus.masterTool
刀具为主刀具	

数据类型和更新

BOOL

值 TRUE

刀具为主刀具

值 FALSE

刀具不是主刀具

12.3.125.10 measured

NC → PLC	<Toolholder>.command.in.multitoolStatus.measured
刀具已测量	

数据类型和更新

BOOL

值 TRUE

刀具已测量

值 FALSE

刀具未测量

12.3.125.11 prewarning

NC → PLC	<Toolholder>.command.in.multitoolStatus.prewarning
达到刀具预警极限	

数据类型和更新

BOOL

值 TRUE

达到刀具预警极限

值 FALSE

未达到刀具预警极限

12.3.125.12 toBeLoaded

NC → PLC	<Toolholder>.command.in.multitoolStatus.toBeLoaded
必须装载刀具	

数据类型和更新

BOOL

值 TRUE

必须装载刀具

值 FALSE

无需装载刀具

12.3.125.13 toBeUnloaded

NC → PLC	<Toolholder>.command.in.multitoolStatus.toBeUnloaded
必须卸载刀具	

数据类型和更新

BOOL

值 TRUE

必须卸载刀具

值 FALSE

无需卸载刀具

12.3.125.14 toolAtDisabledLocation

NC → PLC	<Toolholder>.command.in.multitoolStatus.toolAtDisabledLocation
刀具处于锁定的刀库刀位	

数据类型和更新

BOOL

12.3.125.15 toolInBuffer

NC → PLC	<Toolholder>.command.in.multitoolStatus.toolInBuffer
刀具位于缓存刀库中	

数据类型和更新

BOOL

值 TRUE

刀具位于缓存刀库中

值 FALSE

刀具不在缓存刀库中

12.3.125.16 used

NC → PLC	<Toolholder>.command.in.multitoolStatus.used
刀具已使用	

数据类型和更新

BOOL

值 TRUE

刀具已使用

值 FALSE

刀具未使用

12.4 PLC 指令 SINU_*

12.4.1 概览

说明

指令的数据库版本和版本 CNC-SW

在 TIA-Portal V18 中，对于 CNC-SW V6.20 推荐库版本 V10.1，对于 CNC-SW V6.21 推荐库版本 V11.0。

下表按字母顺序列举了 SINUMERIK PLC 基本程序的各个指令。

名称	说明
SINU_AsyncTransfer (页 1092)	报告刀具或刀库更改后的位置，且无 NCK 任务。
SINU_CompleteRegistration (页 1090)	在全局数据块"LSINU_Hmi"中，向 HMI 提供所有已注册实例的信息。
SINU_CtrlStarDelta (页 1247)	主轴的星角切换
SINU_Init (页 1087)	基本程序在启动时初始化
SINU_NcActivateBlockSearch (页 1148)	将通道切换为程序段搜索模式
SINU_NcAssignInterrupt (页 1140)	为 NC 程序分配中断
SINU_NcCancelAlarms (页 1134)	确认分类为"取消报警"的所有报警
SINU_NcCreateCuttingEdge (页 1193)	使用刀具的下一个可用刀刃号生成刀刃
SINU_NcCreateCuttingEdgeWithD (页 1195)	接入一个新刀沿
SINU_NcCreateMultiTool (页 1181)	通过指定一个标识符或所指定 TOA 单元中的刀具号接入多刀
SINU_NcCreateTool (页 1159)	根据指定的 TOA 单元中的刀具号创建刀具
SINU_NcCreateToolByName (页 1161)	根据指定的 TOA 单元中的刀具名称创建刀具
SINU_NcDeleteCuttingEdge (页 1197)	删除指定刀具中的刀沿或带有绝对刀沿号的刀沿
SINU_NcDeleteMultiTool (页 1190)	删除多刀
SINU_NcDeleteTool (页 1179)	根据刀具号删除刀具
SINU_NcGetToolNumber (页 1164)	可通过刀具名称和副编号确定刀具的刀具号
SINU_NcLogin (页 1130)	将指定的密码传输至 NC
SINU_NcLogout (页 1133)	重置最后传输至 NC 的密码
SINU_NcMoveTool (页 1173)	在指定的 TOA 单元中装载、卸载和设置刀具
SINU_NcPositionMagazine (页 1199)	将刀库刀位定位在指定的位置上
SINU_NcPositionMultitool (页 1184)	在指定的 TOA 单元中定位多刀

名称	说明
SINU_NcReconfigureMachineData (页 1138)	立即激活生效级为“NEW_CONF” (cf) 的机床数据
SINU_NcReset (页 1136)	触发 NC 复位 (Power On)
SINU_NcResetActiveToolStatus (页 1177)	将所有刀具状态为“激活”和“已禁用”的刀具的状态设为“未激活”
SINU_NcResetMonitoringValues (页 1157)	将指定刀具刀沿的监控值重置为设定值（初始值）
SINU_NcRetractTool (页 1171)	选择 JOG-Retract 运行方式
SINU_NcSearchLoadingLocation (页 1206)	在 TOA 单元中执行空位搜索以装载具有刀具号的刀具。
SINU_NcSearchLocation (页 1202)	在 TOA 单元指定的刀库内执行空位搜索。
SINU_NcSearchMultitoolLocation (页 1187)	搜索多刀内的空刀位
SINU_NcSearchTool (页 1167)	搜索带特定属性的刀具
SINU_NcSelectPartProgram (页 1144)	选择 NC 程序以处理某个通道
SINU_NcSetSemaphore (页 1152)	为 HMI 和 PLC 检查或设置信号量
SINU_NcSetSemaphoreByToa (页 1212)	检查或设置用于 HMI 和 PLC 的信号量，以便在执行某些指令时为某些关键数据区域提供保护
SINU_NcSetUserDataActive (页 1150)	在选择的通道中，将刀具补偿、基本框架、可设置的框架和全局框架等用户数据设为活动状态
SINU_NcSetWorkpieceCounterInc (页 1209)	在指定 TOA 单元中设置用于主轴计件器的增量值
SINU_OscillateSpindle (页 1230)	开始或停止摆动某个主轴
SINU_PositionSpindle (页 1236)	控制主轴的位置。
SINU_ProcessNckEvent (页 1091)	处理 NCK 事件，以便在 PLC 程序中使用
SINU_ReadCompileCycleData (页 1259)	读取编译循环数据的指定范围
SINU_ReadUserDataInt (页 1252)	从 NC 中读取 Int 类型的 PLC 机床数据。
SINU_ReadUserDataHex (页 1254)	从 NC 中读取 Hex (Byte) 类型的 PLC 机床数据。
SINU_ReadUserDataFloat (页 1256)	从 NC 中读取 Real 类型的 PLC 机床数据。

名称	说明
SINU_RegisterAxis (页 1110)	注册一个与加工轴进行数据交换的实例。 轴通过数据类型"NCK_typeAxis"的实例来指定。
SINU_RegisterAxisByInst (页 1113)	注册一个与带所需的轴编号的加工轴进行数据交换的实例。
SINU_RegisterAxisByName (页 1117)	注册一个与指定的轴名称的加工轴进行数据交换的实例。
SINU_RegisterAxisByInstWithHmi (页 1115)	注册一个与加工轴进行数据交换的实例，其名称应与指定的实例的符号名称一致，并分配一个实例用于与 HMI 进行数据交换。
SINU_RegisterAxisByNameWithHmi (页 1119)	注册一个与指定的轴名称的加工轴进行数据交换的实例，并分配一个实例用于与 HMI 进行数据交换。
SINU_RegisterChan (页 1106)	注册一个与 NC 通道进行数据交换的实例
SINU_RegisterChanWithHmi (页 1107)	注册一个与指定的 NC 通道进行数据交换的实例，并分配一个实例用于与 HMI 进行数据交换。
SINU_RegisterLoadingPoint (页 1123)	注册一个与 TOA 单元中指定的装载位进行数据交换的实例。
SINU_RegisterModeGroup (页 1102)	注册一个与指定的运行方式组进行数据交换的实例。
SINU_RegisterModeGroupWithHmi (页 1103)	注册一个与所需的运行方式组进行数据交换的实例，并分配一个实例用于与 HMI 进行数据交换。
SINU_RegisterNc (页 1098)	注册一个进行通用 NC 数据交换的实例。
SINU_RegisterNCwithHmi (页 1099)	注册一个进行通用 NC 数据交换的实例，并分配一个实例用于与 HMI 进行数据交换。
SINU_RegisterRelocate (页 1126)	注册一个在 TOA 单元中实施数据交换的实例。
SINU_RegisterToaUnit (页 1121)	注册一个指定的 NCK TOA 单元的实例。
SINU_RegisterToolholder (页 1124)	注册一个与 TOA 单元中指定的刀架进行数据交换的实例。
SINU_RequestAsup (页 1094)	启动 NC 中预定义的子程序（异步子程序）
SINU_ReRegisterAxis (页 1120)	请求指定的轴实例的数据更新。
SINU_ReRegisterChan (页 1109)	请求指定的通道实例的数据更新。
SINU_ReRegisterLoadingPoint (页 1127)	请求指定的装载位实例的数据更新。
SINU_ReRegisterModeGroup (页 1105)	请求已注册的运行方式组实例的数据更新。

名称	说明
SINU_ReRegisterNc (页 1101)	请求已注册的 NC 实例的数据更新。
SINU_ReRegisterRelocateUpdate (页 1129)	请求指定的实施实例的数据更新。
SINU_ReRegisterToolholder (页 1128)	请求指定的刀架实例的数据更新。
SINU_RotateSpindle (页 1241)	启动或停止主轴的旋转。
SINU_PublishUserProgramVersion	应用用户 PLC 程序的版本信息，以便在 SINUMERIK Operate 中进行显示
SINU_TransferHandwheelData (页 1279)	从 PLC 向 NCK 传输手轮信号（手轮的旋转脉冲）
SINU_TraverseIndexingAxis (页 1223)	控制分度轴。
SINU_TraversePosAxis (页 1216)	控制定位轴。
SINU_Update (页 1089)	已更新所有已注册实例。
SINU_UpdateEthernetMcp (页 1271)	通过工业以太网与操作组件通信
SINU_WriteCompileCycleData (页 1261)	将数值写入编译循环数据的指定范围内
SINU_WriteRecordOverEthernet (页 1263)	将数据记录 (Record) 传输到机床控制面板中

参见

SINU_PublishUserProgramVersion (页 1096)

12.4.2 SINU_Init

说明

自固件版本 V6.21 起的特点和指令版本 V11.0：

指令包含在 HMI SINUMERIK Operate 中显示 PLC 基本程序版本的附加信息。

说明

在启动过程中初始化基本程序。

该指令可初始化基本程序，调用基本程序的所有功能（比如注册实例等）之前，必须先调用块。

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块开头部分

说明

建议

在 Startup 组织块开头部分，所有其他指令前面调用“SINU_Init”。

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
800F	另一个基本程序已进行初始化。	使用 PLC Basic Program plus 时，务必保证不要调用“LBP_ConfigBP”。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

12.4.3 SINU_Update

说明

自固件版本 V6.21 起的特点和指令版本 V11.0：

为支持 HMI SINUMERIK Operate 中的轨迹拓展了指令。

说明

该指令可以更新所有已注册的实例。

只有当 NCK 和 PLC 处于循环运行中，交换过生命符号且 PLC 程序未发生变更时，才会进行更新。

如果不是这种情况，所有实例中的数据都会失效，“ncCpuReady”输出亦不置位。

“reRegisterNeeded”输出置位后，必须重新连接之前注册的各个实例（参见“SINU_ReRegisterNc”、“SINU_ReRegisterModeGroup”等）。之后必须再次调用“SINU_Update”，以此实现所有实例的更新。

调用时间点

- 每个 PLC 循环调用一次，比如 Main 组织块开头部分

说明

建议

在 Main 组织块开头部分，所有其他指令前面调用“SINU_Update”。

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
ncCpuReady	Output	Bool	TRUE:已更新所有已请求实例的数据且数据有效。
reRegisterUpdateNeeded	Output	Bool	TRUE:必须为所有已注册实例请求重新关联，因为 PLC 程序发生变更。之后必须再次调用“SINU_Update”。
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
8008	至少有一个已注册实例未请求更新数据。	为所有已注册实例申请更新数据。
8402	NCK 未启动，无循环运行。	-
8403	NCK 不再有运行迹象，不再有循环运行。	-
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

12.4.4 SINU_CompleteRegistration

说明

在全局数据块“LSINU_Hmi”中，向 HMI 提供所有已注册实例的信息。为了让 HMI SINUMERIK Operate 能使用接口信号，必须在项目中创建名称为“LSINU_Hmi”、数据类型为“NCK_typeHmi”的全局数据块。不允许使用其他名称和数据类型。

所有实例完成注册后，必须调用该指令。

调用时间点

- 在 PLC 引导启动中完成所有注册后调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
hmi	InOut	NCK_typeHmi	HMI 数据块“LSINU_Hmi”
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
8201	参数"hmi"中的数据块名称不允许使用。	使用"LSINU_Hmi"作为 HMI 数据块的名称。
8401	存储空间不足导致部分实例名称未能传输到 HMI。	使用已注册实例的简称。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

12.4.5 SINU_ProcessNckEvent

说明

该指令可以处理 NCK 事件以在 PLC 程序中使用。比如，处理刀具管理的辅助功能和指令等。

关于辅助功能和 NC 信号的详细信息，参见 NCK 信号的传输和应答时间 (页 71)章节。

为此，请在您于 NCK 过程报警事件 (页 91)中定义的，经过组态的过程报警组织块中调用指令。

调用时间点

- 每次中断调用一次，NCK 事件的过程报警组织块开头

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
8010	无可用的 NCK 事件	只在 NCK 事件的过程报警 OB 中调用指令一次。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

此例展示如何在 NCK 事件的过程报警组织块中进行调用。

SCL

```
// tempError: Bool
// tempStatus: Word
"SINU_ProcessNckEvent"(error => #tempError,
                       status => #tempStatus);
```

12.4.6 SINU_AsyncTransfer

说明

在没有 NCK 任务的情况下，该指令可以报告更改后的刀具或刀库位置。使用此块的不同实例时，该指令可以同时运行多个报告。

报告启动后不能取消。报告启动后，不允许更改参数。

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动发送至 NCK 的报告。只有输出 "done" 或 "error" 后，才允许复位。
toolSourceMagazine	Input	Int	刀具或待定位刀库此前位置的刀库和刀位编号。
toolSourceLocation	Input	Int	
toolTargetMagazine	Input	Int	刀具当前位置的刀库和刀位编号，或者刀库要定位的位置。
toolTargetLocation	Input	Int	
multitoolTargetLocation	Input	Int	Multitool 内的位置； 无 Multitool 时为 0
fixedLocation	Input	Bool	TRUE:刀具从实际刀库传输到周转位置后，为刀具执行刀位预留。
ackStatus	Input	UInt	1：传输刀具 5：定位刀库
toaUnit	InOut	NCK_typeToaUnit	注册的 TOA 单元实例，更改后的位置报告给此实例
busy	Output	Bool	TRUE:正在报告位置，不复位 "execute"
done	Output	Bool	TRUE:已成功报告位置，可以复位 "execute"
error	Output	Bool	TRUE:出现错误，可以复位 "execute"
status	Output	Word	执行报告时的状态编号。 出错时的错误编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-

状态/错误代码* (W#16#...)	说明	解决办法
7002	消息已发送至 NCK，等待应答。	-
8005	TOA 单元实例未注册。	开始使用 TOA 单元 前，先通过 “SINU_RegisterToaUnit”进行注册。
19xx	NCK 报警	从 NCK 传递的涉及刀具管理传输问题的错误代码。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

12.4.7 SINU_RequestAsup

描述

使用指令“SINU_RequestAsup”，可在 NC 中启动预定义的子程序 (ASUB)。在启动之前，必须为 NC 程序中的 ASUB 分配一个中断编号和优先级。在 PLC 中，可通过调用指令 “SINU_NcAssignInterrupt”进行此分配。分配后，可根据需要多次调用 ASUB。

该通道中运行的 NC 程序被 ASUB 中断。可在一个 PLC 循环中启动多个 ASUB，这些 ASUB 在 NC 中依次执行。无法在具有活动通道复位的通道中启动任务。

任务启动后无法更改参数。任务开始后，不能通过指令取消。

如果每个任务使用专用实例，则可同时开始多个任务。

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

参数

下表列出了该指令的参数：

参数	声明	数据类型	描述
execute	Input	Bool	TRUE:启动功能。
interruptNumber	Input	UInt	中断编号 1...8 和 11...32（可能不使用 9、10）
chan	InOut	NCK_typeChan	NC 通道的注册实例
status	Output	Word	发生错误时的状态编号或错误编号

错误、状态和补救措施

状态/错误代码* (W#16#...)	描述	补救措施
0000	无错误	-
7000	调用而不执行	-
7001	ASUB 处理已开始	-
7002	中断已识别	-
7005	子程序运行	-
8005	通道实例未注册	在使用"SINU_RegisterChan"之前注册 chan 单元或使用注册的通道实例。
8209	中断编号无效	指定一个有效的中断编号。
8410	子程序启动时出错	-
8411	执行子程序时出错	-
* 状态代码：7xxx（信息、报警），参数错误=FALSE；错误代码：8xxx（错误），参数错误 = TRUE		

示例：启动子程序

在此示例中，异步子程序在通道 1 中启动，中断编号为 1。实例"NckInst".chan1 被分配了通道 1。

SCL

```
// #SINU_RequestAsup_Instance:SINU_RequestAsup;
// statExecute :布尔型
// tempBusy :布尔型
// tempDone :布尔型
// tempError :布尔型
// tempStatus :Word
// job start: #statStart := TRUE
#SINU_RequestAsup_Instance(execute := #statExecute,
                           interruptNumber := 1,
                           busy => #tempBusy,
                           done => #tempDone,
                           error => #tempError,
                           status => #tempStatus,
                           chan := "NckInst".chan1);
IF #tempDone OR #tempError THEN
      #statExecute := FALSE;
END_IF;
```

12.4.8 SINU_PublishUserProgramVersion

说明

指令自以下版本起可用：

CNC 软件 ≥ V6.21 以及指令库版本 ≥ V11.0

说明

该指令可以应用用户 PLC 程序的版本信息，以便在 SINUMERIK Operate 中进行显示借此传输名称或标题、版本、日期、时间和注释。所有参数的内容都可以自由选择，或者留空。

该指令可以多次调用。之前传输的数据将被完全覆盖，不会保留。

SINUMERIK Operate 中，给出的参数值在所有语言版本中均保持不变。

调用时间点

- 任意
- **建议：**
  在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
name	Input	String	用户 PLC 程序的名称或标题
version	Input	String	版本标识，例如“10.1.569”
versionDate	Input	String	日期，例如发布日期：“2022-12-24”
versionTime	Output	String	时间，例如发布时间：“23:59”
comment	Input	String	注释或补充信息
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
820a	参数值总长过长	全部所提交信息的总长度不得超过 570 个字符。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

提交 42.1001 版中的“终极 PLC 程序”的版本信息，但未指定时间。

SCL

```
// tempError : Bool
// tempStatus : Word

SINU_PublishUserProgramVersion(name := 'Ultimate PLC program',
                               version := '42.1001',
                               versionDate := '2022-12-24',
                               versionTime := '',
                               comment := 'Universally applicable',
                               error => tempError,
                               status => tempStatus);
```

12.4.9 Registration

12.4.9.1 SINU_RegisterNc

说明

该指令可以注册实例以交换一般 NC 数据。

注册成功后，在“config”区域存储一般数据并设置“registered”旗标。所有实例都注册完成后，必须调用“SINU_CompleteRegistration”。通过循环调用“SINU_Update”实现实例更新。

含 HMI 实例的 NC 实例备选注册方案：

- "SINU_RegisterNcWithHmi"

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
nc	InOut	NCK_typeNc	运行方式组编号
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已注册一个（其他）实例。	每个 NC 只注册一个实例。
8003	实例已注册。	一个实例只注册给一个 NC。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

SINU_CompleteRegistration (页 1090)

12.4.9.2 SINU_RegisterNcWithHmi

说明

该指令可以注册实例以交换一般 NC 数据，并分配一个实例与 HMI 进行数据交换。

注册成功后，在“config”区域存储一般数据并设置“registered”旗标。所有实例都注册完成后，必须调用“SINU_CompleteRegistration”。通过循环调用“SINU_Update”实现实例更新。

无 HMI 实例的 NC 实例备选注册方案：

- "SINU_RegisterNc"

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
nc	InOut	NCK_typeNc	运行方式组编号
hmiNc	InOut	NCK_typeHmiNc	要分配给 NC 的 HMI 实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已注册一个（其他）实例。	每个 NC 只注册一个实例。
8003	实例已注册。	一个实例只注册给一个 NC。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

SINU_CompleteRegistration (页 1090)

12.4.9.3 SINU_ReRegisterNc

说明

该指令可以重新关联已注册的 NC 实例。只有通过向"SINU_Update"发送 reRegisterNeeded = TRUE 信号这种方式发出请求后，才需要调用。

在"SINU_ReRegister*"后面，通过下一次调用"SINU_Update"更新实例。

调用时间点

- 发布请求（通过向"SINU_Update"发送 reRegisterNeeded = TRUE 信号）后，在 PLC 循环中为每个 NC 实例调用一次，比如在 Main 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
nc	InOut	NCK_typeNc	要重新关联的 NC 实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
8005	实例未注册。	在重新关联实例之前，先注册一个 NC 实例。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

12.4.9.4 SINU_RegisterModeGroup

说明

该指令可以注册实例以与所需的运行方式组进行数据交换。

注册成功后，在“config”区域存储相应的运行方式组编号并设置“registered”和“modeGroupActive”旗标。所有实例都注册完成后，必须调用“SINU_CompleteRegistration”。通过循环调用“SINU_Update”实现实例更新。

含 HMI 实例的运行方式组实例备选注册方案：

- "SINU_RegisterModeGroupWithHmi"

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
number	Input	UInt	运行方式组编号
modeGroup	InOut	NCK_typeModeGroup	要为运行方式组注册的运行方式组实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已为此运行方式组注册一个实例。	每个运行方式组只注册一个实例。
8003	此实例已注册给其他运行方式组。	一个实例只注册给一个运行方式组。
8004	指定的运行方式组未激活。	只为处于活动状态的运行方式组注册实例。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

SINU_CompleteRegistration (页 1090)

12.4.9.5 SINU_RegisterModeGroupWithHmi

说明

该指令可以注册实例以与所需的运行方式组进行数据交换，并分配一个实例与 HMI 进行数据交换。

注册成功后，在“config”区域存储相应的运行方式组编号并设置“registered”和“modeGroupActive”旗标。所有实例都注册完成后，必须调用“SINU_CompleteRegistration”。通过循环调用“SINU_Update”实现实例更新。

无 HMI 实例的运行方式组实例备选注册方案：

- "SINU_RegisterModeGroup"

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
number	Input	UInt	运行方式组编号
modeGroup	InOut	NCK_typeModeGroup	要为运行方式组注册的运行方式组实例
hmiModeGroup	InOut	NCK_typeHmiModeGroup	要分配给运行方式组的 HMI 实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已为此运行方式组注册一个实例。	每个运行方式组只注册一个实例。
8003	此实例已注册给其他运行方式组。	一个实例只注册给一个运行方式组。
8004	指定的运行方式组未激活。	只为处于活动状态的运行方式组注册实例。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

SINU_CompleteRegistration (页 1090)

12.4.9.6 SINU_ReRegisterModeGroup

说明

该指令可以重新关联已注册的运行方式组实例。只有通过向"SINU_Update"发送 reRegisterNeeded = TRUE 信号这种方式发出请求后，才需要调用。

在"SINU_ReRegister*"后面，通过下一次调用"SINU_Update"更新实例。

调用时间点

- 发布请求（通过向"SINU_Update"发送 reRegisterNeeded = TRUE 信号）后，在 PLC 循环中为每个运行方式组实例调用一次，比如在 Main 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
modeGroup	InOut	NCK_typeModeGroup	要重新关联的运行方式组实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
8004	所需的运行方式组未激活。	只重新关联处于活动状态的运行方式组。
8005	实例未注册。	在重新关联实例之前，先注册一个运行方式组实例。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

12.4.9.7 SINU_RegisterChan

说明

该指令可以注册实例以与指定的 NC 通道进行数据交换。

注册成功后，在“config”区域存储相应的通道编号并设置“registered”和“chanActive”旗标。所有实例都注册完成后，必须调用“SINU_CompleteRegistration”。通过循环调用“SINU_Update”实现实例更新。

含 HMI 实例的通道实例备选注册方案：

- "SINU_RegisterChanWithHmi"

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
number	Input	UInt	通道的编号
chan	InOut	NCK_typeChan	要为通道注册的通道实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已为此通道注册一个实例。	每个通道只注册一个实例。
8003	此实例已注册给其他通道。	一个实例只注册给一个通道。

状态/错误代码* (W#16#...)	说明	解决办法
8004	指定的通道未激活。	只为处于活动状态的通道注册实例。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

SINU_CompleteRegistration (页 1090)

12.4.9.8 SINU_RegisterChanWithHmi

说明

该指令可以注册实例以与指定的 NC 通道进行数据交换，并分配一个实例与 HMI 进行数据交换。

注册成功后，在“config”区域存储相应的通道编号并设置“registered”和“chanActive”旗标。所有实例都注册完成后，必须调用“SINU_CompleteRegistration”。通过循环调用“SINU_Update”实现实例更新。

无 HMI 实例的通道实例备选注册方案：

- "SINU_RegisterChan"

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
number	Input	UInt	通道的编号
chan	InOut	NCK_typeChan	要为通道注册的通道实例
hmiChan	InOut	NCK_typeHmiChan	要分配给通道的 HMI 实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已为此通道注册一个实例。	每个通道只注册一个实例。
8003	此实例已注册给其他通道。	一个实例只注册给一个通道。
8004	指定的通道未激活。	只为处于活动状态的通道注册实例。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

SINU_CompleteRegistration (页 1090)

12.4.9.9 SINU_ReRegisterChan

说明

该指令可以重新关联已注册的通道实例。只有通过向"SINU_Update"发送 reRegisterNeeded = TRUE 信号这种方式发出请求后，才需要调用。

在"SINU_ReRegister*"后面，通过下一次调用"SINU_Update"更新实例。

调用时间点

- 发布请求（通过向"SINU_Update"发送 reRegisterNeeded = TRUE 信号）后，在 PLC 循环中为每个通道实例调用一次，比如在 Main 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
chan	InOut	NCK_typeChan	应重新关联的通道实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
8004	所需通道未激活。	只重新关联处于活动状态的通道。
8005	实例未注册。	在重新关联实例之前，先注册一个通道实例。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

12.4.9.10 SINU_RegisterAxis

说明

该指令可以注册实例以与具有所需轴编号的机床轴进行数据交换。

注册成功后，在“config”区域存储相应的机床轴编号并设置“”registered”和“axisActive”旗标。所有实例都注册完成后，必须调用“SINU_CompleteRegistration”。通过循环调用“SINU_Update”实现实例更新。

轴实例备选注册方案：

- "SINU_RegisterAxisByName"
- "SINU_RegisterAxisByInst"

含相应 HMI 实例的轴实例备选注册方案：

- "SINU_RegisterAxisWithHmi"

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
number	Input	UInt	机床轴的编号
axis	InOut	NCK_typeAxis	要为机床轴注册的轴实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已为此机床轴注册一个实例。	每个机床轴只注册一个实例。
8003	此实例已注册给其他机床轴。	一个实例只注册给一个机床轴。
8004	指定的机床轴未激活。	只为处于活动状态的机床轴注册实例。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

SINU_CompleteRegistration (页 1090)

12.4.9.11 SINU_RegisterAxisWithHmi

说明

该指令可以注册实例以与具有所需轴编号的机床轴进行数据交换，并分配一个实例与 HMI 进行数据交换。

注册成功后，在“config”区域存储相应的机床轴编号并设置“registered”和“axisActive”旗标。

所有实例都注册完成后，必须调用“SINU_CompleteRegistration”。通过循环调用“SINU_Update”实现实例更新。

含相应 HMI 实例的轴实例备选注册方案：

- "SINU_RegisterAxisByNameWithHmi"
- "SINU_RegisterAxisByInstWithHmi"

无 HMI 实例的轴实例备选注册方案：

- "SINU_RegisterAxis"

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
number	Input	UInt	机床轴的编号
轴	InOut	NCK_typeAxis	要为机床轴注册的轴实例。
hmiAxis	InOut	NCK_typeHmiAxis	要分配给机床轴的 HMI 实例。
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已为此机床轴注册一个实例。	每个机床轴只注册一个实例。
8003	此实例已注册给其他机床轴。	一个实例只注册给一个机床轴。
8004	指定的机床轴未激活。	只为处于活动状态的机床轴注册实例。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

背景数据块 "NckInst" 包含一个数据类型为 "NCK_typeAxis" 的机床轴 X1。出于数据交换目的注册机床轴 X1。

SCL

```
// tempError: Bool
// tempStatus: Word
SINU_RegisterAxisWithHmi(number := 1,
                         error => #tempError,
                         status => #tempStatus,
                         axis := "NckInst".axisX1,
                         hmiAxis := "NckInst".hmiX1);
```

12.4.9.12 SINU_RegisterAxisByInst

说明

该指令可以注册实例以与满足以下条件的机床轴进行数据交换：机床轴名称与所指定实例的符号名称一致。轴名称对应机床数据 MD10000 $MN_AXCONF_MACHAX_NAME_TAB 中的条目。

说明

符号名称必须与轴名称精确对应，包括大小写都要一致。

注册成功后，在“config”区域存储相应的机床轴编号并设置“registered”和“axisActive”旗标。所有实例都注册完成后，必须调用“SINU_CompleteRegistration”。通过循环调用“SINU_Update”实现实例更新。

轴实例备选注册方案：

- "SINU_RegisterAxis"
- "SINU_RegisterAxisByName"

含相应 HMI 实例的轴实例备选注册方案：

- "SINU_RegisterAxisByInstWithHmi"

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
axis	InOut	NCK_typeAxis	要为机床轴注册的轴实例，且其符号名称与机床轴名称一致。
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已为此机床轴注册一个实例。	每个机床轴只注册一个实例。
8003	此实例已注册给其他机床轴。	一个实例只注册给一个机床轴。
8004	指定的机床轴未激活。	只为处于活动状态的机床轴注册实例。
800B	未找到具有指定名称的轴。	使用在 NC 中组态的机床轴名称进行注册。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

将实例 "X1" 分配给机床轴 X1。出于数据交换目的注册机床轴 X1。

SCL

```
// tempError: Bool
// tempStatus: Word
```

SCL

```
SINU_RegisterAxisByInst(error => tempError,
                        status => tempStatus,
                        axis := "X1"
                        );
```

参见

错误代码和状态代码 (页 1345)

SINU_CompleteRegistration (页 1090)

12.4.9.13 SINU_RegisterAxisByInstWithHmi

说明

该指令可以注册实例以与满足以下条件的机床轴进行数据交换：机床轴名称与所指定实例的符号名称一致；此外，分配一个实例与 HMI 进行数据交换。轴名称对应机床数据 MD10000 $MN_AXCONF_MACHAX_NAME_TAB 中的条目。

说明

符号名称必须与轴名称精确对应，包括大小写都要一致。与 HMI 进行数据交换的实例可以任意命名。

注册成功后，在“config”区域存储相应的机床轴编号并设置“registered”和“axisActive”旗标。所有实例都注册完成后，必须调用“SINU_CompleteRegistration”。通过循环调用“SINU_Update”实现实例更新。

含相应 HMI 实例的轴实例备选注册方案：

- "SINU_RegisterAxisWithHmi"
- "SINU_RegisterAxisByNameWithHmi"

无 HMI 实例的轴实例备选注册方案：

- "SINU_RegisterAxisByInst"

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
axis	InOut	NCK_typeAxis	要为机床轴注册的轴实例，且其符号名称与机床轴名称一致。
hmiAxis	InOut	NCK_typeHmiAxis	要分配给机床轴的 HMI 实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已为此机床轴注册一个实例。	每个机床轴只注册一个实例。
8003	此实例已注册给其他机床轴。	一个实例只注册给一个机床轴。
8004	指定的机床轴未激活。	只为处于活动状态的机床轴注册实例。
800B	未找到具有指定名称的轴。	使用在 NC 中组态的机床轴名称进行注册。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

将实例 "X1" 分配给机床轴 X1。出于数据交换目的注册机床轴 X1。

SCL

```
// tempError: Bool
// tempStatus: Word
```

SCL

```
SINU_RegisterAxisByInstWithHmi(error => #tempError,,
                                status => #tempStatus,
                                axis := "X1",
                                hmiAxis := "NckInst".hmiX1);
```

参见

SINU_CompleteRegistration (页 1090)

12.4.9.14 SINU_RegisterAxisByName

说明

该指令可以注册实例以与具有指定轴名称的机床轴进行数据交换。轴名称对应机床数据 MD10000 $MN_AXCONF_MACHAX_NAME_TAB 中的条目。

说明

符号名称必须与轴名称精确对应，包括大小写都要一致。

注册成功后，在“config”区域存储相应的机床轴编号并设置“registered”和“axisActive”旗标。所有实例都注册完成后，必须调用“SINU_CompleteRegistration”。通过循环调用“SINU_Update”实现实例更新。

轴实例备选注册方案：

- "SINU_RegisterAxis"
- "SINU_RegisterAxisByInst"

含相应 HMI 实例的轴实例备选注册方案：

- "SINU_RegisterAxisByNameWithHmi"

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
name	Input	String	机床轴名称
axis	InOut	NCK_typeAxis	要为机床轴注册的轴实例，且其符号名称与机床轴名称一致。
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已为此机床轴注册一个实例。	每个机床轴只注册一个实例。
8003	此实例已注册给其他机床轴。	一个实例只注册给一个机床轴。
8004	指定的机床轴未激活。	只为处于活动状态的机床轴注册实例。
800B	未找到具有指定名称的轴。	使用在 NC 中组态的机床轴名称进行注册。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

SINU_CompleteRegistration (页 1090)

12.4.9.15 SINU_RegisterAxisByNameWithHmi

说明

该指令可以注册实例以与具有指定轴名称的机床轴进行数据交换，并分配一个实例与 HMI 进行数据交换。轴名称对应机床数据 MD10000 $MN_AXCONF_MACHAX_NAME_TAB 中的条目。

说明

符号名称必须与轴名称精确对应，包括大小写都要一致。与 HMI 进行数据交换的实例可以任意命名。

注册成功后，在“config”区域存储相应的机床轴编号并设置“registered”和“axisActive”旗标。所有实例都注册完成后，必须调用“SINU_CompleteRegistration”。通过循环调用“SINU_Update”实现实例更新。

含相应 HMI 实例的轴实例备选注册方案：

- "SINU_RegisterAxisWithHmi"
- "SINU_RegisterAxisByInstWithHmi"

无 HMI 实例的轴实例备选注册方案：

- "SINU_RegisterAxisByName"

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
name	Input	String	机床轴名称
axis	InOut	NCK_typeAxis	要为机床轴注册的轴实例，且其符号名称与机床轴名称一致。
hmiAxis	InOut	NCK_typeHmiAxis	要分配给机床轴的 HMI 实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已为此机床轴注册一个实例。	每个机床轴只注册一个实例。
8003	此实例已注册给其他机床轴。	一个实例只注册给一个机床轴。
8004	指定的机床轴未激活。	只为处于活动状态的机床轴注册实例。
800B	未找到具有指定名称的轴。	使用在 NC 中组态的机床轴名称进行注册。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

SINU_CompleteRegistration (页 1090)

12.4.9.16 SINU_ReRegisterAxis

说明

该指令可以重新关联已注册的轴实例。只有通过向“SINU_Update”发送 reRegisterNeeded = TRUE 信号这种方式发出请求后，才需要调用。

在“SINU_ReRegister*”后面，通过下一次调用“SINU_Update”更新实例。

调用时间点

- 发布请求（通过向“SINU_Update”发送 reRegisterNeeded = TRUE 信号）后，在 PLC 循环中为每个轴实例调用一次，比如在 Main 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
axis	InOut	NCK_typeAxis	要重新关联的轴实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
8004	所需的轴未激活。	只重新关联处于活动状态的机床轴。
8005	实例未注册。	在重新关联实例之前，先注册一个轴实例。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

12.4.9.17 SINU_RegisterToaUnit

说明

该指令可以注册所指定 NCK TOA 单元的一个实例。

注册成功后，在“config”区域存储 NCK TOA 单元的相关信息并设置“registered”旗标。所有实例都注册完成后，必须调用“SINU_CompleteRegistration”。未给 TOA 单元的实例规定数据循环交换。

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
number	Input	UInt	NCK TOA 单元的编号（参考 MD28085 MM_LINK_TOA_UNIT）
toaUnit	InOut	NCK_typeToaUnit	要为 NCK TOA 单元注册的 TOA 单元实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已为此 TOA 单元注册一个实例。	每个 TOA 单元只注册一个实例。
8003	此实例已注册给其他 TOA 单元。	一个实例只注册给一个 TOA 单元。
8004	指定的 TOA 单元未激活。	只为处于活动状态的 TOA 单元注册实例。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

SINU_CompleteRegistration (页 1090)

12.4.9.18 SINU_RegisterLoadingPoint

说明

该指令可以注册实例以与 TOA 单元内指定的装载位进行数据交换。

说明

TOA 单元实例必须已通过“SINU_RegisterToaUnit”进行注册。

注册成功后，在“config”区域存储相应的装载位编号并设置“registered”旗标。所有实例都注册完成后，必须调用“SINU_CompleteRegistration”。通过循环调用“SINU_Update”实现实例更新。

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
number	Input	UInt	装载位或装载站的编号，对应装载库内的刀位编号（装载位/装载站 **x** 参考 $TC_MPP1[9999,**x**]=6 或 7，前提是 $TC_MAP1[9999]=9）
loadingPoint	InOut	NCK_typeLoadingPoint	要为 NCK 装载位注册的装载位实例
toaUnit	InOut	NCK_typeToaUnit	要注册装载位的 TOA 单元实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已为此装载位注册一个实例。	每个装载位只注册一个实例。
8003	此实例已注册给其他装载位。	一个实例只注册给一个装载位。
8004	指定的装载位未激活。	只为处于活动状态的装载位注册实例。
8005	TOA 单元实例未注册。	开始使用 TOA 单元 前，先通过“SINU_RegisterToaUnit”进行注册。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

SINU_CompleteRegistration (页 1090)

12.4.9.19 SINU_RegisterToolholder

说明

该指令可以注册实例以与 TOA 单元内指定的刀夹进行数据交换。

说明

TOA 单元实例必须已通过“SINU_RegisterToaUnit”进行注册。

注册成功后，在“config”区域存储相应的刀夹编号并设置“registered”旗标。所有实例都注册完成后，必须调用“SINU_CompleteRegistration”。通过循环调用“SINU_Update”实现实例更新。

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
number	Input	UInt	刀夹编号或主轴编号（刀夹 **x** 参考 $TC_MPP5[9998,1]=**x**，前提是 $TC_MPP1[9998,1]=2 且 $TC_MAP1[9998]=7）
toolholder	InOut	NCK_typeToolholder	要为刀夹注册的刀夹实例
toaUnit	InOut	NCK_typeToaUnit	要注册刀夹的 TOA 单元实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已为此刀夹注册一个实例。	每个刀夹只注册一个实例。
8003	此实例已注册给其他刀夹。	一个实例只注册给一个刀夹。
8004	指定的刀夹未激活。	只为处于活动状态的刀夹注册实例。
8005	TOA 单元实例未注册。	开始使用 TOA 单元 前，先通过“SINU_RegisterToaUnit”进行注册。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

SINU_CompleteRegistration (页 1090)

12.4.9.20 SINU_RegisterRelocate

说明

针对 TOA 单元内的移刀，该指令可以注册一个数据交换实例。

说明

TOA 单元实例必须已通过"SINU_RegisterToaUnit"进行注册。

注册成功后，在"config"区域存储相应的编号并设置"registered"旗标。所有实例都注册完成后，必须调用"SINU_CompleteRegistration"。通过循环调用"SINU_Update"实现实例更新。

调用时间点

- 在 PLC 引导启动中调用一次，比如在 Startup 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
relocate	InOut	NCK_typeRelocate	要注册的移刀实例
toaUnit	InOut	NCK_typeToaUnit	要为移刀注册的 TOA 单元实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7004	存储空间不足导致未能保存实例名称。参见“SINU_CompleteRegistration”和错误 16#8401。	所有实例均用简称进行注册。
8002	已为此移刀注册一个实例。	每个移刀只注册一个实例。
8003	此实例已注册给其他移刀。	一个实例只注册给一个移刀。
8004	指定的移刀未激活。	只为处于活动状态的移刀注册实例。
8005	TOA 单元实例未注册。	开始使用 TOA 单元 前，先通过“SINU_RegisterToaUnit”进行注册。
800E	注册已结束，调用“SINU_CompleteRegistration”函数后，不允许再进行注册。	开始调用“SINU_CompleteRegistration”之前，先执行完所有注册。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

SINU_CompleteRegistration (页 1090)

12.4.9.21 SINU_ReRegisterLoadingPoint

说明

该指令可以重新关联已注册的装载位实例。只有通过向“SINU_Update”发送 reRegisterNeeded = TRUE 信号这种方式发出请求后，才需要调用。

在“SINU_ReRegister*”后面，通过下一次调用“SINU_Update”更新实例。

调用时间点

- 发布请求（通过向“SINU_Update”发送 reRegisterNeeded = TRUE 信号）后，在 PLC 循环中为每个装载位实例调用一次，比如在 Main 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
loadingPoint	InOut	NCK_typeLoadingPoint	要重新关联的装载位实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
8005	实例未注册。	在重新关联实例之前，先注册一个装载位实例。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

12.4.9.22 SINU_ReRegisterToolholder

说明

该指令可以重新关联已注册的刀夹实例。只有通过向"SINU_Update"发送 reRegisterNeeded = TRUE 信号这种方式发出请求后，才需要调用。

在"SINU_ReRegister*"后面，通过下一次调用"SINU_Update"更新实例。

调用时间点

- 发布请求（通过向"SINU_Update"发送 reRegisterNeeded = TRUE 信号）后，在 PLC 循环中为每个刀夹实例调用一次，比如在 Main 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
toolholder	InOut	NCK_typeToolholder	要重新关联的刀夹实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
8005	实例未注册。	在重新关联实例之前，先注册一个刀夹实例。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

12.4.9.23 SINU_ReRegisterRelocate

说明

该指令可以重新关联已注册的移刀实例。只有通过向"SINU_Update"发送 reRegisterNeeded = TRUE 信号这种方式发出请求后，才需要调用。

在"SINU_ReRegister*"后面，通过下一次调用"SINU_Update"更新实例。

调用时间点

- 发布请求（通过向"SINU_Update"发送 reRegisterNeeded = TRUE 信号）后，在 PLC 循环中为每个移刀实例调用一次，比如在 Main 组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
relocate	InOut	NCK_typeRelocate	要重新关联的移刀实例
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
8005	实例未注册。	在重新关联实例之前，先注册一个移刀实例。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

参见

错误代码和状态代码 (页 1345)

12.4.10 NC commands

12.4.10.1 SINU_NcLogin

描述

该指令将指定密码传输给 NC，并激活相应的 NCK 访问级别。

注意
PLC 项目密码的可见性 要使用该指令，必须在 PLC 程序中保存 NCK 访问级别的密码。这意味着，有权访问 PLC 程序的每个人都可以看到 NCK 访问级别的密码。

密码包括至少 8 个字符。对于较短的密码，可在字符串中添加空格，以达到 8 个字符。在某些情况下，不接受标准密码和不符合安全规范的密码，即使它们属于 NCK 访问级别。

任务启动后无法更改参数。任务开始后，不能通过指令取消。

指令"SINU_NcLogout"可用于重置 NCK 访问级别和最后设置的密码。

NC 命令

该指令的调用对应于 PI 服务：

- LOGIN

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

参数

下表列出了该指令的参数：

参数	声明	数据类型	描述
execute	Input	Bool	TRUE：开始任务
password	Input	String	所需 NCK 访问级别的密码
busy	Output	Bool	TRUE：任务已执行
done	Output	Bool	TRUE：任务已成功完成，“执行”可重置
error	Output	Bool	TRUE：已发生错误，“执行”可重置
status	Output	Word	任务执行后的状态编号。 发生错误时的错误编号。

错误、状态和补救措施

状态/错误代码* (W#16#...)	描述	补救措施
0000	无错误	-
7001	等待 NCK 的自由接口。	-
7002	任务发送到 NCK，等待响应。	-
7005	任务发送到 NCK，等待完成。	-

状态/错误代码* (W#16#...)	描述	补救措施
8017	新任务在活动任务完成之前开始	仅在活动任务完成后，将“执行”设置回 TRUE
820a	参数过长	使用较短的密码
820b	密码错误。	使用正确的强密码
820c	参数超出有效范围	遵守参数的有效值范围
* 状态代码：7xxx（信息、报警），参数错误=FALSE；错误代码：8xxx（错误），参数错误 = TRUE		

示例：密码已传输

密码“MeinKennwort.42”传输到 NC。这是在设置静态变量#statExecute := TRUE 后发生的。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcLogin_Instance : SINU_NcLogin
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcLogin_Instance(execute := #statExecute,
                       password := 'MeinKennwort.42',
                       busy => #tempBusy,
                       done => #tempDone,
                       error => #tempError,
                       status => #tempStatus);
IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

更多信息

- SINU_NcLogout (页 1133)

12.4.10.2 SINU_NcLogout

描述

该指令重置最后传输到 NC 的密码，因此重置相应的 NCK 访问级别。

任务开始后，不能通过指令取消。

指令“SINU_NcLogin”可用于使用密码设置新的 NCK 访问级别。

NC 命令

该指令的调用对应于 PI 服务：

- LOGOUT

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

参数

下表列出了该指令的参数：

参数	声明	数据类型	描述
execute	Input	Bool	TRUE：开始任务
busy	Output	Bool	TRUE：任务已执行
done	Output	Bool	TRUE：任务已成功完成，“执行”可重置
error	Output	Bool	TRUE：已发生错误，“执行”可重置
status	Output	Word	任务执行后的状态编号。 发生错误时的错误编号。

错误、状态和补救措施

状态/错误代码* (W#16#...)	描述	补救措施
0000	无错误	-
7001	等待 NCK 的自由接口。	-
7002	任务发送到 NCK，等待响应。	-

状态/错误代码* (W#16#...)	描述	补救措施
7005	任务发送到 NCK，等待完成。	-
8017	新任务在活动任务完成之前开始	仅在活动任务完成后，将“执行”设置回 TRUE
* 状态代码：7xxx（信息、报警），参数错误=FALSE；错误代码：8xxx（错误），参数错误 = TRUE		

示例：密码已传输

当前访问级别和关联密码应在 NC 中重置。这是在设置静态变量#statExecute := TRUE 后实现的。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcLogout_Instance : SINU_NcLogout
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcLogout_Instance(execute := #statExecute,
                        busy => #tempBusy,
                        done => #tempDone,
                        error => #tempError,
                        status => #tempStatus);
IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

更多信息

- SINU_NcLogin (页 1130)

12.4.10.3 SINU_NcCancelAlarms

说明

该指令可以确认分类为“取消报警”的所有报警。无法进行某些报警的个别确认。

一旦任务开始，就无法通过指令取消。

NC：命令

对指令的调用与 PI 服务相对应：

- CANCEL

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置"execute"
error	Output	Bool	TRUE:出错，可以重置"execute"
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将"execute"设置为 TRUE。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

删除分类为“取消报警”的所有报警。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcCacncel_Instance : SINU_NcCancel
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcCancel_Instance(execute := #statExecute,
                        busy => #tempBusy,
                        done => #tempDone,
                        error => #tempError,
                        status => #tempStatus);

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

12.4.10.4 SINU_NcReset

说明

本指令将触发 NC 复位 (Power On)。

一旦任务开始，就无法通过指令取消。

NC：命令

对指令的调用与 PI 服务相对应：

- IBN_SS 带参数 0 (Power On)

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置“execute”
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将“execute”设置为 TRUE。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

应触发 NC 复位。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcReset_Instance : SINU_NcReset
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#SINU_NcReset_Instance(execute := #statExecute,
                       busy => #tempBusy,
                       done => #tempDone,
                       error => #tempError,
                       status => #tempStatus);

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

12.4.10.5 SINU_NcReconfigureMachineData

说明

此指令能够立即激活由操作人员或 PLC 按顺序输入的生效级为"NEW_CONF"(cf) 的机床数据。

此指令只能在控制系统复位或程序中断（程序段交界处 NC 停止）状态下激活。

一旦任务开始，就无法通过指令取消。

NC：命令

对指令的调用与 PI 服务相对应：

- CONFIG

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置"execute"

参数	声明	数据类型	说明
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将“execute”设置为 TRUE。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

激活分类为“NEW_CONF”的机床数据。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcReconfigureMachineData_Instance : SINU_NcReconfigureMachineData
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#SINU_NcReconfigureMachineData_Instance(execute := #statExecute,
                        busy => #tempBusy,
                        done => #tempDone,
                        error => #tempError,
                        status => #tempStatus);

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

12.4.10.6 SINU_NcAssignInterrupt

描述

该指令根据中断编号和指定通道中的优先级将中断分配给 NC 程序。

只可在指定通道的 RESET 状态下进行调用 (<Chan>.basic.in.stateReset)。

作业启动后无法更改参数。任务开始后，不能通过指令取消。

如果每个作业使用专用实例，则可在不同通道中同时启动多个作业。

中断分配保持到下一次热重启或直到分配被显式删除（使用参数 `progPath := ''` 调用）.

说明

更多信息

有关在参数"progPath"处指定 NC 程序的详细信息，请参见 *NC 编程手册中的章节"工作准备">"文件和程序管理">"程序存储"*。

分配后，可使用指令"SINU_RequestAsup"启动 PLC 的中断，其中使用相同的中断编号和相同的 NC 通道。

NC 命令

指令调用对应于以下 NC 命令：

- SETINT
- SETINT LIFTFAST，参数 `liftFast := TRUE`
- SETINT BLSYNC，参数 `blockSync := TRUE`

- CLRINT，参数 `progPath := ''`
- 或 PI 服务
  ASUB

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

参数

下表列出了该指令的参数：

参数	声明	数据类型	描述
execute	Input	Bool	TRUE：开始任务
progPath	Input	String	NC 程序的名称或 NC 程序的完整路径 更多信息： *NC 编程手册中的章节“工作准备”>“文件和程序管理”>“程序存储”* ''（空路径）：删除分配 示例：“_N_MYPROG_MPF”或“/_N_MPF_DIR/_N_MYPROG_MPF”
interruptNumber	Input	USInt	中断编号 1...8 和 11...32（可能不使用 9、10）
priority	Input	USInt	1..128：优先级 1 为最高优先级
liftFast	Input	Bool	TRUE：从轮廓快速回退
blockSync	Input	Bool	TRUE：“progPath”中的 NC 程序应在中断启动后从下一个程序块更改开始
chan	InOut	NCK_typeChan	应在其中分配中断的 NCK 通道的注册实例。
busy	Output	Bool	TRUE：任务已执行
done	Output	Bool	TRUE：任务已成功完成，“执行”可重置
error	Output	Bool	TRUE：已发生错误，“执行”可重置
status	Output	Word	作业执行后的状态编号。 发生错误时的错误编号。

错误、状态和补救措施

状态/错误代码* (W#16#...)	**描述**	**补救措施**
0000	无错误	-
7001	等待 NCK 的自由接口。	-
7002	作业发送到 NCK，等待响应。	-
7005	作业发送到 NCK，等待完成。	-
8005	通道实例未注册。	先注册一个通道实例
8017	新任务在活动任务完成之前开始	仅在活动任务完成后，将“执行”设置回 TRUE
8019	功能已生效，无法再次启动	重复作业/不同时对一个通道执行多个调用，而是逐个进行
820a	参数过长	在参数“progPath”处使用较短的路径
820b	未找到程序，或通道未处于 RESET 状态	对现有 NC 程序使用正确的路径和程序名称，并重置通道 **注：** 如果在热重启时调用 SINU_NcAssignInterrupt，则可能在第一次调用时，状态 = 820b。热重启后，ProgEvents 可能会启动，这就是通道没有值“stateReset”的原因。 **补救措施：** • 如果状态为 820b，再次调用 SINU_NcAssignInterrupt • 仅在 NC 使用 OPI 变量“anPoweronState”指示 ProgEvents 已完成时启动 SINU_NcAssignInterrupt
820c	参数超出有效范围	遵守参数的有效值范围
* 状态代码：7xxx（信息、报警），参数错误=FALSE；错误代码：8xxx（错误），参数错误 = TRUE		

示例：分配中断

在注册实例“Chan1”的通道中，应将优先级为 2 的中断 5 分配给 NC 程序 MYPROG.SPF。这是在设置静态变量#statExecute := TRUE 后执行的。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcAssignInterrupt_Instance : SINU_NcAssignInterrupt
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcAssignInterrupt_Instance(execute := #statExecute,
                                 interruptNumber := 5,
                                 priority := 2,
                                 liftFast := FALSE,
                                 blockSync := FALSE,
                                 progPath := '/_N_SPF_DIR/_N_MYPROG_SPF',
                                 busy => #tempBusy,
                                 done => #tempDone,
                                 error => #tempError,
                                 status => #tempStatus,
                                 chan := "Chan1");
IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

示例：为 NC 程序动态分配中断

在注册实例"Chan1"的通道中，优先级为 10 的中断 24 应分配给变量 #myProg 中的 NC 程序，该变量位于目录 #myDir 中。这是在设置静态变量#statExecute := TRUE 后执行的。

使用函数 CONCAT() 组合完整的程序路径。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcAssignInterrupt_Instance : SINU_NcAssignInterrupt
// myDir : String
// myProg : String
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#myDir := '_N_MPF_DIR';
#myProg := '_N_MYPROG_MPF';

#SINU_NcAssignInterrupt_Instance(
    execute := #statExecute,
    interruptNumber := 24,
    priority := 10,
    liftFast := FALSE,
    blockSync := FALSE,
    progPath := CONCAT(IN1 := '/', IN2 := #myDir, IN3 := '/', IN4 := #myProg),
    busy => #tempBusy,
    done => #tempDone,
    error => #tempError,
    status => #tempStatus,
    chan := "Chan1");
IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

更多信息

- SINU_RequestAsup (页 1094)
- *NC 编程手册中的章节“工作准备”>“文件和程序管理”>“程序存储”*

12.4.10.7 SINU_NcSelectPartProgram

描述

该指令选择一个通道的 NC 程序，用于执行。

作业启动后无法更改参数。任务开始后，不能通过指令取消。

如果每个任务使用专用实例，则可同时开始多个任务。

说明

更多信息

有关在参数“progPath”处指定 NC 程序的详细信息，请参见 *NC 编程手册中的章节“工作准备”>“文件和程序管理”>“程序存储”*。

NC 命令

该指令的调用对应于 PI 服务：

- SELECT

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

参数

下表列出了该指令的参数：

参数	声明	数据类型	描述
execute	Input	Bool	TRUE：开始任务
progPath	Input	String	要选择的 NC 程序名称或 NC 程序完整路径 更多信息： *NC 编程手册中的章节“工作准备”>“文件和程序管理”>“程序存储”* 示例：“_N_MYPROG_MPF”或“/_N_MPF_DIR/_N_MYPROG_MPF” **注：** 程序分配无法删除。
chan	InOut	NCK_typeChan	应在其中选择程序的 NCK 通道的注册实例。
busy	Output	Bool	TRUE：任务已执行
done	Output	Bool	TRUE：任务已成功完成，“执行”可重置
error	Output	Bool	TRUE：已发生错误，“执行”可重置
status	Output	Word	作业执行后的状态编号。 发生错误时的错误编号。

错误、状态和补救措施

状态/错误代码* (W#16#...)	**描述**	**补救措施**
0000	无错误	-
7001	等待 NCK 的自由接口。	-
7002	作业发送到 NCK，等待响应。	-
7005	作业发送到 NCK，等待完成。	-
8005	通道实例未注册。	先注册一个通道实例
8017	新任务在活动任务完成之前开始	仅在活动任务完成后，将“执行”设置回 TRUE
820a	参数过长	在参数“progPath”处使用较短的路径
820b	程序未找到	对现有 NC 程序使用正确的路径和程序名称
820c	参数超出有效范围	遵守参数的有效值范围
* 状态代码：7xxx（信息、报警），参数错误=FALSE；错误代码：8xxx（错误），参数错误 = TRUE		

示例：选择程序

在注册实例“Chan1”的通道中，应选择 NC 程序 MYPROG.SPF。这是在设置静态变量 #statExecute := TRUE 后执行的。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcSelectPartProgram_Instance : SINU_NcSelectPartProgram
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#SINU_NcSelectPartProgram_Instance(execute := #statExecute,
                                   progPath := '/_N_SPF_DIR/_N_MYPROG_SPF',
                                   busy => #tempBusy,
                                   done => #tempDone,
                                   error => #tempError,
                                   status => #tempStatus,
                                   chan := "Chan1");
IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

示例：动态选择程序

在注册实例"Chan1"的通道中，NC 程序应在变量 #myProg 中选择，该变量位于文件夹 #myDir 中。这是在设置静态变量#statExecute := TRUE 后执行的。

使用函数 CONCAT() 组合完整的程序路径。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcSelectPartProgram_Instance : SINU_NcSelectPartProgram
// myDir : String
// myProg : String
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#myDir := '_N_MPF_DIR';
#myProg := '_N_MYPROG_MPF';

#SINU_NcSelectPartProgram_Instance(
    execute := #statExecute,
    progPath := CONCAT(IN1 := '/', IN2 := #myDir, IN3 := '/', IN4 := #myProg),
    busy => #tempBusy,
    done => #tempDone,
    error => #tempError,
    status => #tempStatus,
    chan := "Chan1");
IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

更多信息

- *NC 编程手册中的章节“工作准备”>“文件和程序管理”>“程序存储”*

12.4.10.8 SINU_NcActivateBlockSearch

说明

该指令可以将通道切换为程序段搜索模式。在这个时间点上，程序段搜索指针必须已在 NC 中生效。

可随时通过 NC 复位终止程序段搜索。

程序段搜索完成后将自动重新激活正常的执行模式。以找到的搜索目标为起点进行 NC 启动。

任务开始后无法更改参数。一旦任务开始，就无法通过指令取消。

如果针对每项任务编制了单独的实例，则可以同时启动多项任务。

NC：命令

对指令的调用与 PI 服务相对应：

- FINDBL

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
mode	Input	USInt	预处理模式 1：不进行计算 2:进行计算和定位 4：进行计算，不定位 5：进行程序测试
chan	InOut	NCK_typeChan	NCK 通道的已注册实例，需在其中执行指令
busy	Output	Bool	TRUE:已执行任务

参数	声明	数据类型	说明
done	Output	Bool	TRUE:已成功结束任务，可以重置“execute”
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	通道实例未注册。	首先注册通道实例
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将“execute”设置为 TRUE。
820c	参数超出有效范围	遵守参数的有效值域
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在已注册实例“Chan1”的通道中，激活程序段搜索模式，且不需要进行计算。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcActivateBlockSearch_Instance : SINU_NcActivateBlockSearch
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#SINU_NcActivateBlockSearch_Instance(execute := #statExecute,
                                     mode := 1, // 搜索，无计算
                                     busy => #tempBusy,
                                     done => #tempDone,
                                     error => #tempError,
                                     status => #tempStatus,
                                     chan := "Chan1);

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

12.4.10.9 SINU_NcSetUserDataActive

描述

该指令激活用户数据，例如刀具偏移、基准及选定通道中的可调与全局框架。

用户数据仅在通道的 RESET 或 STOP 状态下激活，并仅可使用下一个 NC 程序块更新。

作业启动后无法更改参数。任务开始后，不能通过指令取消。

如果每个任务使用专用实例，则可同时开始多个任务。

NC 命令

该指令的调用对应于 PI 服务：

- SETUDT

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

参数

下表列出了该指令的参数：

参数	声明	数据类型	描述
execute	Input	Bool	TRUE：开始任务
userDataType	Input	USInt	应设置为激活状态的用户数据 1：更新活动刀具偏移 2：更新基本框架 3：更新活动可调框架 4：更新活动全局基本框架 5：更新活动全局可调框架
chan	InOut	NCK_typeChan	应在其中将用户数据设置为激活状态的 NC 通道的注册实例。
busy	Output	Bool	TRUE：任务已执行
done	Output	Bool	TRUE：任务已成功完成，“执行”可重置
error	Output	Bool	TRUE：已发生错误，“执行”可重置
status	Output	Word	作业执行后的状态编号。 发生错误时的错误编号。

错误、状态和补救措施

状态/错误代码* (W#16#...)	描述	补救措施
0000	无错误	-
7001	等待 NCK 的自由接口。	-
7002	作业发送到 NCK，等待响应。	-
7005	作业发送到 NCK，等待完成。	-
8005	通道实例未注册。	先注册一个通道实例
8017	新任务在活动任务完成之前开始	仅在活动任务完成后，将“执行”设置回 TRUE
820b	通道未处于 STOP 或 RESET 状态	在通道中执行 NC 停止或重置
820c	userDataType 的值无效	遵守 userDataType 的有效值范围
* 状态代码：7xxx（信息、报警），参数错误=FALSE；错误代码：8xxx（错误），参数错误 = TRUE		

示例：激活刀具偏移的用户数据

在注册实例"Chan1"的通道中，刀具偏移的用户数据应设置为激活状态。这是在设置静态变量 #statExecute := TRUE 后执行的。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcSetUserDataActive_Instance : SINU_NcSetUserDataActive
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcSetUserDataActive_Instance(execute := #statExecute,
                                   userDataType:= 1,   // 刀具偏移
                                   busy => #tempBusy,
                                   done => #tempDone,
                                   error => #tempError,
                                   status => #tempStatus,
                                   chan := "Chan1");
IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

12.4.10.10 SINU_NcSetSemaphore

描述

该指令检查或设置 HMI 和 PLC 的信号量，以在执行某些指令时保护关键数据区域。信号量也可以重置。

调用指令时，信号量仅在此前未设置时进行设置。如果信号量已设置，则该指令会发出错误信号。

说明

在刀具管理的 PI 服务的上下文中，指令 SINU_NcSetSempahoreByToa 优于 SINU_NcSetSempahore。

作业启动后无法更改参数。任务开始后，不能通过指令取消。

如果每个作业使用专用实例，则可在不同的 TOA 单元中或为不同的信号量同时启动多个作业。

在执行以下指令之一之前，必须检查并设置关联的信号量（testAndSet := TRUE）：

- SINU_NcCreateToolByName
- SINU_NcSearchLoadingLocation
- SINU_NcMoveTool
- SINU_NcSearchLocation
- SINU_NcGetToolNumber
- SINU_NcSearchTool
- SINU_NcSearchMultitoolLocation
- SINU_NcPositionMultitool
- SINU_NcPositionMagazine
- SINU_NcCreateMultiTool

信号量必须始终设置在分配给 TOA 单元的编号最小的通道中。

如果信号量已成功设置，返回的变量也只能被读取。从相应的 NC 变量中读取指令的返回后，必须重置信号量（testAndSet := FALSE）。只要信号量仍重置，就不能再次对其进行设置。

设置信号量后，相关指令可调用任意次数，而不需要复位和设置信号量。

可自由使用未分配给指令的附加信号量。

NC 命令

该指令的调用对应于 PI 服务：

- MMCSEM

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

参数

下表列出了该指令的参数：

参数	声明	数据类型	描述
execute	Input	Bool	TRUE：开始任务
semaphoreNumber	Input	USInt	1：用于 SINU_NcCreateToolByName 1：用于 SINU_NcCreateMultiTool 2：用于 SINU_NcSearchLocation 2：用于 SINU_NcSearchLoadingLocation 2：用于 SINU_NcSearchMultitoolLocation 3：用于 SINU_NcMoveTool 3：用于 SINU_NcPositionMultitool 3：用于 SINU_NcPositionMagazine 4..5:可自由使用 6：用于 SINU_NcGetToolNumber 6：用于 SINU_NcSearchTool 7..10:可自由使用
testAndSet	Input	Bool	TRUE：检查信号量是否未设置，然后进行设置 FALSE：重置信号量
chan	InOut	NCK_typeChan	应在其中设置信号量的 NCK 通道的注册实例。
busy	Output	Bool	TRUE：任务已执行
done	Output	Bool	TRUE：任务已成功完成，“执行”可重置
error	Output	Bool	TRUE：已发生错误，“执行”可重置
status	Output	Word	作业执行后的状态编号。 发生错误时的错误编号。

错误、状态和补救措施

状态/错误代码* (W#16#...)	描述	补救措施
0000	无错误	-
7001	等待 NCK 的自由接口。	-

状态/错误代码* (W#16#...)	描述	补救措施
7002	作业发送到 NCK，等待响应。	-
7005	作业发送到 NCK，等待完成。	-
8005	通道实例未注册。	先注册一个通道实例
8017	新任务在活动任务完成之前开始	仅在活动任务完成后，将“执行”设置回 TRUE
820b	信号量已设置，无法设置	重置信号量（如已设置）或等待信号量再次可用
820c	参数超出有效范围	遵守参数的有效值范围
* 状态代码：7xxx（信息、报警），参数错误=FALSE；错误代码：8xxx（错误），参数错误 = TRUE		

示例：设置信号量

在注册实例“Chan1”的通道中，应设置空位置搜索（“SINU_NcSearchLocation”）的信号量。在设置静态变量#statExecute := TRUE 后进行此尝试。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

如果已设置信号量，则再次尝试设置信号量。如果无法成功设置信号量，则设置 #statSemaphore := TRUE。

SCL

```
// SINU_NcSetSemaphore_Instance : SINU_NcSetSemaphore
// statExecute : Bool
// statRetry : Bool
// statSemaphore : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#SINU_NcSetSemaphore_Instance(execute := #statExecute,
空位置搜索的                                    semaphoreNumber := 2, //
                              testAndSet := TRUE,   // 测试和设置
                              busy => #tempBusy,
                              done => #tempDone,
                              error => #tempError,
                              status => #tempStatus,
                              chan := "Chan1");
IF #statRetry THEN
    #statExecute := TRUE;
    #statRetry := FALSE;
ELSIF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;

    #statRetry := #tempError AND #statSetOrReset;
    #statSemaphore := #tempDone;
END_IF;
```

示例：重置信号量

在注册实例"Chan1"的通道中，应重置空位置搜索（"SINU_NcSearchLocation"）的信号量。这是在设置静态变量#statExecute := TRUE 后执行的。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcSetSemaphore_Instance : SINU_NcSetSemaphoree
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcSetSemaphore_Instance(execute := #statExecute,
空位置搜索的                                    semaphoreNumber := 2, //
                              testAndSet := FALSE,  // 重置
                              busy => #tempBusy,
                              done => #tempDone,
                              error => #tempError,
                              status => #tempStatus,
                              chan := "Chan1");
IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

更多信息

SINU_NcSetSemaphoreByToa (页 1212)

12.4.10.11 SINU_NcResetMonitoringValues

说明

该指令可将指定刀具刀沿的监控值重置为设定值（初始值）。其只适用于激活了监控的刀具。

如果没有选择专用的监控类型，所有生效的刀具监控都将被重置。

任务开始后无法更改参数。一旦任务开始，就无法通过指令取消。

如果针对每项任务编制了单独的实例，则可以同时启动多项任务。

NC：命令

对指令的调用与 PI 服务相对应：

- TRESMO

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
tool	Input	Int	1..32000：刀具的刀具号 -1：加工所有刀具
cuttingEdge	Input	Int	1、2、...刀沿的刀沿号 -1：复位所有刀沿的监控
sisterTools	Input	Bool	FALSE：仅编辑拥有此刀具号的刀具 TRUE:编辑该刀具的所有姐妹刀具（仅适用于刀具号 1..9999）。
serviceLifeMonitoring	Input	Bool	TRUE:重置刀具使用寿命监控 如果没有设置监控类型，所有生效的刀具监控都将被重置

参数	声明	数据类型	说明
quantityMonitoring	Input	Bool	TRUE:重置工件数量监控 如果没有设置监控类型，所有生效的刀具监控都将被重置
wearingMonitoring	Input	Bool	TRUE:重置刀具磨损监控 如果没有设置监控类型，所有生效的刀具监控都将被重置
sumCorrectionMonitoring	Input	Bool	TRUE:重置总和校正监控 如果没有设置监控类型，所有生效的刀具监控都将被重置
toaUnit	InOut	NCK_typeToaUnit	TOA 单元的已注册实例，需在其中执行任务
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置"execute"
error	Output	Bool	TRUE:出错，可以重置"execute"
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	TOA 实例未注册。	首先注册 TOA 实例
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将"execute"设置为 TRUE。
820c	参数值无效	遵守参数的有效值域
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在注册实例"Toa1"的 TOA 单元中，将重置针对刀具 42 所有刀沿的磨损和使用寿命的监控。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcResetMonitoringValues_Instance : SINU_NcResetMonitoringValues
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcResetMonitoringValues_Instance(execute := #statExecute,
                                       tool := 42,       //刀具号
                                       cuttingEdge := -1,
                                       sisterTools := FALSE,
                                       serviceLiveMonitoring := TRUE,
                                       quantityMonitoring := FALSE,
                                       wearingMonitoring := TRUE,
                                       sumCorrectionMonitoring := FALSE,
                                       busy => #tempBusy,
                                       done => #tempDone,
                                       error => #tempError,
                                       status => #tempStatus,
                                       toaUnit := "Toa1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

12.4.10.12 SINU_NcCreateTool

描述

该指令基于指定 TOA 单元中的 T 编号装刀。如果 TOA 单元中已存在具有此 T 编号的刀具，则该指令会发出错误 16#820b 的信号。

说明

该功能在带和不带刀具管理的控制系统中均可用。

作业启动后无法更改参数。任务开始后，不能通过指令取消。

如果每个任务使用专用实例，则可同时开始多个任务。

NC 命令

该指令的调用对应于 PI 服务：

- CREATO

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

参数

下表列出了该指令的参数：

参数	声明	数据类型	描述
execute	Input	Bool	TRUE：开始任务
tool	Input	UInt	1..32000：新刀具的 T 编号
toaUnit	InOut	NCK_typeToaUnit	应在其中创建刀具的 TOA 单元的注册实例
busy	Output	Bool	TRUE：任务已执行
done	Output	Bool	TRUE：任务已成功完成，“执行”可重置
error	Output	Bool	TRUE：已发生错误，“执行”可重置
status	Output	Word	作业执行后的状态编号。 发生错误时的错误编号。

错误、状态和补救措施

状态/错误代码* (W#16#...)	描述	补救措施
0000	无错误	-
7001	等待 NCK 的自由接口。	-
7002	作业发送到 NCK，等待响应。	-
7005	作业发送到 NCK，等待完成。	-
8005	TOA 实例未注册。	先注册一个 TOA 实例
8017	新任务在活动任务完成之前开始	仅在活动任务完成后，将“执行”设置回 TRUE
820b	T 编号已存在	使用另一个 T 编号

状态/错误代码* (W#16#...)	描述	补救措施
820c	T 编号的值无效	遵守 T 编号的有效值范围
* 状态代码：7xxx（信息、报警），参数错误=FALSE；错误代码：8xxx（错误），参数错误 = TRUE		

示例：创建刀具

在注册实例"Toa1"的 TOA 单元中，应创建 T 编号为 42 的新刀具。这是在设置静态变量 #statExecute := TRUE 后发生的。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcCreateTool_Instance : SINU_NcCreateTool
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcCreateTool_Instance(execute := #statExecute,
                            tool := 42,              //T 编号
                            busy => #tempBusy,
                            done => #tempDone,
                            error => #tempError,
                            status => #tempStatus,
                            toaUnit := "Toa1");
IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

12.4.10.13 SINU_NcCreateToolByName

描述

该指令基于指定 TOA 单元中的刀具标识符装刀。此外，还可指定 T 编号和 Duplo 编号。

如果 TOA 单元中没有具有此 T 编号的刀具，则该指令会发出错误信号。执行成功后，新建刀具的 T 编号位于 NC 变量"TnumWZV"中。

说明

执行指令之前，为保护 NC 变量 TnumWZV，必须设置关联的信号量：

- 指令"SINU_NcSetSemaphoreByToa"，带信号量编号 := 1, testAndSet := TRUE 和参数"toaUnit"中 TOA 单元的相同实例

读取 NC 变量 TnumWZV 后，必须重置关联的信号量：

- 带信号量编号:= 1 和 testAndSet := FALSE 的指令"SINU_NcSetSemaphoreByToa"

如果指令在执行前发出错误信号，则仍可使用设置的信号量重复该步骤。如果不重复执行，则必须重置信号量。

任务启动后无法更改参数。任务开始后，不能通过指令取消。

如果每个作业使用专用实例，则可在不同的 TOA 单元中同时启动多个作业。

NC 命令

该指令的调用对应于 PI 服务：

- TMCRTO

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

参数

下表列出了该指令的参数：

参数	声明	数据类型	描述
execute	Input	Bool	TRUE：开始任务
toolName	Input	String	新刀具的刀具标识符，最多 31 个字符
toolNumber	Input	Int	1..32000：新刀具的 T 编号 -1:由 NC 分配的 T 编号
duploNumber	Input	Int	1..32000：新刀具的 Duplo 编号 -1:由 NC 分配的 Duplo 编号
toaUnit	InOut	NCK_typeToaUnit	应在其中创建刀具的 TOA 单元的注册实例
busy	Output	Bool	TRUE：任务已执行，不要重置“执行”
done	Output	Bool	TRUE：任务已成功完成，“执行”可重置

参数	声明	数据类型	描述
error	Output	Bool	TRUE：已发生错误，“执行”可重置
status	Output	Word	任务执行后的状态编号。 发生错误时的错误编号。

错误、状态和补救措施

状态/错误代码* (W#16#...)	描述	补救措施
0000	无错误	-
7001	等待 NCK 的自由接口。	-
7002	任务发送到 NCK，等待响应。	-
7005	任务发送到 NCK，等待完成。	-
8005	TOA 实例未注册。	先注册一个 TOA 实例
8017	新任务在活动任务完成之前开始	仅在活动任务完成后，将“执行”设置回 TRUE
820b	具有 T 编号或双编号的刀具已存在	使用另一个 T 编号或另一个 duplo 编号
820c	参数的值无效	遵守参数的有效值范围
* 状态代码：7xxx（信息、报警），参数错误=FALSE；错误代码：8xxx（错误），参数错误 = TRUE		

示例：创建刀具

在注册实例“Toa1”的 TOA 单元中，应创建一个新刀具“MYTOOL”。这是在设置静态变量 #statExecute := TRUE 后执行的。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcCreateToolByName_Instance : SINU_NcCreateToolByName
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#SINU_NcCreateToolByName_Instance(execute := #statExecute,
                                  toolName := 'MYTOOL',
                                  toolNumber := -1,
                                  duploNumber := -1,
                                  busy => #tempBusy,
                                  done => #tempDone,
                                  error => #tempError,
                                  status => #tempStatus,
                                  toaUnit := "Toa1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

更多信息

- SINU_NcSetSemaphoreByToa (页 1212)

12.4.10.14 SINU_NcGetToolNumber

说明

该指令可通过刀具名称和副编号确定刀具的刀具号。

成功执行指令后，结果将出现在以下 NC 变量中：

- resultNrOfTools
  - 0：未找到指定刀具
  - 1：已找到指定刀具
- resultToolNr：所指定刀具的刀具号

说明

在执行指令之前，必须设置所属的信号量以为 NC 变量提供保护：

- 指令“SINU_NcSetSemaphoreByToa”，带 semaphoreNumber := 6、testAndSet := TRUE，并且在参数“toaUnit”中有与 TOA 单元相同的实例

在读取 NC 变量之后，所属的信号量必须重置：

- 指令“SINU_NcSetSemaphoreByToa”，带 semaphoreNumber := 6 和 testAndSet := FALSE

如果在执行指令的过程中报告了一个错误，则可以在依旧设置有信号量的情况下重复该过程。如果未重复执行指令，必须重置信号量。

任务开始后无法更改参数。一旦任务开始，就无法通过指令取消。

如果针对每项任务编制了单独的实例，则可以在不同的 TOA 单元中同时启动多项任务。

NC：命令

对指令的调用与 PI 服务相对应：

- TMGETT

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
toolName	Input	String	刀具的名称，最多 31 个字符
duploNumber	Input	UInt	1..32000：刀具的副编号
toaUnit	InOut	NCK_typeToaUnit	TOA 单元的已注册实例，应确定相应刀具号
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置“execute”
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-

状态/错误代码* (W#16#...)	说明	解决办法
8005	TOA 实例未注册。	首先注册 TOA 实例
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将“execute”设置为 TRUE。
820a	刀具名称过长	刀具名称最长可为 31 个字符
820c	无效参数	遵守参数的有效值域
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在已注册实例“Toa1”的 TOA 单元中，应确定名称为“TOOL”和副编号为 6 的刀具的刀具号。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcGetToolNumber_Instance : SINU_NcGetToolNumber
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcGetToolNumber_Instance(execute := #statExecute,
                               toolName := 'TOOL',
                               duploNumber := 6,
                               busy => #tempBusy,
                               done => #tempDone,
                               error => #tempError,
                               status => #tempStatus,
                               toaUnit := "Toa1");

IF #statExecute THEN
   IF #tempError OR #tempDone THEN
       #statExecute := FALSE;
   END_IF;
END_IF;
```

更多信息

SINU_NcSetSemaphoreByToa (页 1212)

12.4.10.15 SINU_NcSearchTool

说明

该指令用于在一个搜索区域（一个或多个刀库，从一个特定刀位开始，到一个特定刀位结束）内搜索具有指定属性的刀具。只有激活刀具管理后，搜索才可用。

说明

在执行指令之前，必须设置所属的信号量以为 NC 变量提供保护：

- 指令“SINU_NcSetSemaphoreByToa”，带 semaphoreNumber := 6、testAndSet := TRUE，并且在参数“toaUnit”中有与 TOA 单元相同的实例

在读取 NC 变量之后，所属的信号量必须重置：

- 指令“SINU_NcSetSemaphoreByToa”，带 semaphoreNumber := 6 和 testAndSet := FALSE

如果在执行指令的过程中报告了一个错误，则可以在依旧设置有信号量的情况下重复该过程。如果未重复执行指令，必须重置信号量。

任务开始后无法更改参数。一旦任务开始，就无法通过指令取消。

如果针对每项任务编制了单独的实例，则可以在不同的 TOA 单元中同时启动多项任务。

设定方式

- 查找方向
- 搜索下一个具有指定属性的刀具
- 搜索所有具有指定属性的刀具

可联接性和刀具属性的参数设置

只提供了逻辑与运算用于筛选属性。逻辑或运算必须由用户通过多次调用 PI 服务并逐一分析结果才可实现。

通过以下 NC 变量设置找到的刀具的属性：

- "parMask<X>" 参数设置窗口
- "parData<X>" 比较值

其中 <X> = TAD、TAO、TAS、TD

对称搜索

只有满足以下前提条件才能根据刀库位置执行对称搜索。

- 搜索区域中只包含一个刀库：“magazineNumberStart” == “magazineNumberEnd”
- 给定围绕其进行对称搜索的参考刀位，即刀库位：“magazineReference”和“locationReference”

- 对于参考刀位，必须在 TPM 模块中为其配置对待搜索刀库的重复指定
- “searchDirection” == 3

参考刀位是周转位置刀位，即周转位置刀库中或者内部装刀刀库中的刀位，例如切换位置、抓刀器、装刀位。对称搜索会以刀库位置为基准运行至参考刀位前。

若参考刀位前的刀库位置超出了搜索区域，那么该指令的特性将与未找到匹配刀位时相同。

结果列表

指令无故障结束后，搜索结果位于以下 NC 变量中：

- “resultCuttingEdgeNrUsed”：从上一次工件计数起所用刀沿的刀沿号
- “resultNrOfCutEdgesUsed”：从上一次工件计数起的刀沿数量
- "resultNrOfTools" 已找到的刀具的数量
- “resultToolNr”：所找到刀具的刀具号
- “resultToolNrUsed”：从上一次工件计数起所用刀沿的刀具号

如果未找到刀具，已找到的刀具数为“resultNrOfTools” == 0。

NC：命令

对指令的调用与 PI 服务相对应：

- TSEARC

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
firstMagazine	Input	Int	1..32000：从刀库编号 -1 自第一个刀库开始搜索
firstLocation	Input	Int	1..32000：从刀位编号 -1：自第一个刀位开始搜索
lastMagazine	Input	Int	1..32000：结束刀库编号 -1：搜索至最后一个刀库

参数	声明	数据类型	说明
lastLocation	Input	Int	1..32000：到刀位编号 -1：搜索至最后一个刀位
referenceMagazine	Input	Int	1..32000：参考刀库的编号（仅涉及对称搜索，搜索方向 == 3） -1：未使用参考刀库
referenceLocation	Input	Int	1..32000：参考刀位的编号（仅涉及对称搜索，搜索方向 == 3） -1：未使用参考刀位
searchDir	Input	USInt	0：搜索方案像 $TC_MAMP2 中一样设置 1：向前搜索 2:向后搜索 3：对称搜索
allTools	Input	Bool	TRUE:找到所有刀具 FALSE：找到第一个刀具
allEdges	Input	Bool	TRUE:检查全部刀沿 FALSE：检查特定刀沿
toaUnit	InOut	NCK_typeToaUnit	TOA 单元的已注册实例，需在其中搜索刀具
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置“execute”
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-

状态/错误代码* (W#16#...)	说明	解决办法
8005	TOA 实例未注册。	首先注册 TOA 实例
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将"execute"设置为 TRUE。
820c	参数值无效	注意参数的有效值域
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在已注册实例"Toa1"的 TOA 单元中，将以指定参数搜索全部刀具。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcSearchTool_Instance : SINU_NcSearchTool
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcSearchTool_Instance(execute := #statExecute,
                            firstMagazine := -1,
                            firstLocation := -1,
                            lastMagazine := -1,
                            lastLocation := -1,
                            referenceMagazine := -1,
                            referenceLocation := -1,
                            searchDir := 1,
                            allTools := TRUE,
                            allEdges := TRUE,
                            busy => #tempBusy,
                            done => #tempDone,
                            error => #tempError,
                            status => #tempStatus,
                            toaUnit := "Toa1");

IF #statExecute THEN
   IF #tempError OR #tempDone THEN
       #statExecute := FALSE;
   END_IF;
END_IF;
```

更多信息

INU_NcSetSemaphoreByToa (页 1212)

12.4.10.16 SINU_NcRetractTool

说明

指令选择了 JOG-Retract 运行方式。退刀轴（即用于执行回退的几何轴）可由 NC 自动选定或显式设定。

采用自动选定时，系统会将垂直于当前选择的工作平面的几何轴选作退刀轴。

- G17:退刀轴 → 第 3 几何轴 (Z)
- G18:退刀轴 → 第 2 几何轴 (Y)
- G19:退刀轴 → 第 1 几何轴 (X)

可通过 OPI 变量"retractState.Bit 2/3"读取生效的退刀轴。

该指令只能在 JOG 运行方式中“复位”状态下激活。

更多信息：

*功能手册：进给轴和主轴*请参见关键词“刀具方向中的退刀运动 (JOG-Retract)”。

任务开始后无法更改参数。一旦任务开始，就无法通过指令取消。

如果针对每项任务编制了单独的实例，则可以同时启动多项任务。

NC：命令

对指令的调用与 PI 服务相对应：

- RETRAC

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
geoAxis	Input	SInt	-1：由 NC 自动选定退刀轴 1：退刀轴为 WCS 的第 1 工件坐标系的几何轴 2:退刀轴为 WCS 的第 2 工件坐标系的几何轴 3：退刀轴为 WCS 的第 3 工件坐标系的几何轴
chan	InOut	NCK_typeChan	NCK 通道的已注册实例，需在其中执行指令
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置“execute”
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	通道实例未注册。	首先注册通道实例
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将“execute”设置为 TRUE。
820b	未满足 JOG-Retract 的前提条件	更多信息 *功能手册：进给轴和主轴*请参见关键词 “刀具方向中的退刀运动 (JOG-Retract)”
820c	参数超出有效范围	遵守参数的有效值域
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在已注册实例"Chan1"的通道中，需为第 1 个几何轴选择 JOG-Retract 运行方式。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcRetractTool_Instance : SINU_NcRetractTool
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcRetractTool_Instance(execute := #statExecute,
                             geoAxis := 1,  // 第 1 个几何轴
                             busy => #tempBusy,
                             done => #tempDone,
                             error => #tempError,
                             status => #tempStatus,
                             chan := "Chan1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

12.4.10.17 SINU_NcMoveTool

描述

该指令用于在指定 TOA 单元中装载、卸载和重新定位刀具。可使用 T 编号或刀具所在的刀库和位置编号来指定刀具。装载时，刀具必须已位于真实刀库中的装载点。

使用刀库和位置编号指定刀具的目标位置。如果未指定特定的目标位置，则执行空位置搜索。

PLC 中的接口：

- 装载/卸载：装载点
- 其它运动：重新定位

说明

执行指令之前，为防止竞争访问，必须设置关联的信号量：

- 指令“SINU_NcSetSemaphoreByToa”，带信号量编号 := 3, testAndSet := TRUE 和参数 toaUnit 中 TOA 单元的相同实例

执行后必须重置关联的信号量：

- 带信号量编号:= 3 和 testAndSet := FALSE 的指令“SINU_NcSetSemaphoreByToa”

作业启动后无法更改参数。任务开始后，不能通过指令取消。

如果每个作业使用专用实例，则可在不同的 TOA 单元中同时启动多个作业。

NC 命令

该指令的调用对应于 PI 服务：

- TMMVTL

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

参数

下表列出了该指令的参数：

参数	声明	数据类型	描述
execute	Input	Bool	TRUE：开始任务
tool	Input	Int	1..32000：刀具的 T 编号 -1:工具基于 toolSourceMagazine 和 toolSourceLocation 定义
toolSourceMagazine	Input	Int	当前刀位的刀库编号 -1:参数刀具会考虑刀具的 T 编号
toolSourceLocation	Input	Int	当前刀位的位置编号 -1:参数刀具会考虑刀具的 T 编号
toolTargetMagazine	Input	Int	刀具目标位置的刀库编号
toolTargetLocation	Input	Int	刀具目标位置的位置编号 -1:搜索刀库中的空位置，然后使用该位置
toaUnit	InOut	NCK_typeToaUnit	应在其中移动刀具的 TOA 单元的注册实例

参数	声明	数据类型	描述
busy	Output	Bool	TRUE：任务已执行
done	Output	Bool	TRUE：任务已成功完成，“执行”可重置
error	Output	Bool	TRUE：已发生错误，“执行”可重置
status	Output	Word	作业执行后的状态编号。 发生错误时的错误编号。

错误、状态和补救措施

状态/错误代码* (W#16#...)	描述	补救措施
0000	无错误	-
7001	等待 NCK 的自由接口。	-
7002	作业发送到 NCK，等待响应。	-
7005	作业发送到 NCK，等待完成。	-
8005	TOA 实例未注册。	先注册一个 TOA 实例
8017	新任务在活动任务完成之前开始	仅在活动任务完成后，将“执行”设置回 TRUE
820c	参数的值无效	遵守参数的有效值范围
* 状态代码：7xxx（信息、报警），参数错误=FALSE；错误代码：8xxx（错误），参数错误 = TRUE		

示例：卸载

在注册实例“Toa1”的 TOA 单元中，T 编号为 42 的刀具应移动到装载站（刀库 9999，位置 1）进行卸载。这是在设置静态变量#statExecute := TRUE 后执行的。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcMoveTool_Instance : SINU_NcMoveTool
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#SINU_NcMoveTool_Instance(execute := #statExecute,
                          tool := 42,               // T 编号
                          toolSourceMagazine := -1, //  因 T 编号而具有唯一性
                          toolSourceLocation := -1, //  因 T 编号而具有唯一性
                          toolTargetMagazine := 9999,
                          toolTargetLocation := 1,
                          busy => #tempBusy,
                          done => #tempDone,
                          status => #tempStatus,
                          toaUnit := "Toa1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

示例：重新定位

在注册实例"Toa1"的 TOA 单元中，刀库 1 中的刀具应从位置 5 移动到位置 8。这是在设置静态变量#statExecute := TRUE 后执行的。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcMoveTool_Instance : SINU_NcMoveTool
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#SINU_NcMoveTool_Instance(execute := #statExecute,
                          tool := -1,              //  不是 T 编号,
                          toolSourceMagazine := 1, //  使用刀库和
                          toolSourceLocation := 5, //  位置
                          toolTargetMagazine := 1,
                          toolTargetLocation := 8,
                          busy => #tempBusy,
                          done => #tempDone,
                          error => #tempError,
                          status => #tempStatus,
                          toaUnit := "Toa1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

更多信息

- SINU_NcSetSemaphoreByToa (页 1212)

12.4.10.18 SINU_NcResetActiveToolStatus

说明

此指令将所有刀具状态为“生效”和“禁用”的刀具的状态设为“未生效”。

复位刀具状态的正确时间点是在发生以下事件时。

- NC/PLC 接口信号“刀具禁用未生效”的下降沿
- 程序结束
- 通道复位

任务开始后无法更改参数。一旦任务开始，就无法通过指令取消。

如果针对每项任务编制了单独的实例，则可以同时启动多项任务。

NC：命令

对指令的调用与 PI 服务相对应：

- TMRASS

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
toaUnit	InOut	NCK_typeToaUnit	TOA 单元的已注册实例，应删除其中的刀具
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置“execute”
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	TOA 实例未注册。	首先注册 TOA 实例
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将“execute”设置为 TRUE。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在已注册实例“Toa1”的 TOA 单元中，刀具状态将重置。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcResetActiveToolStatus_Instance : SINU_NcResetActiveToolStatus
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcResetActiveToolStatus_Instance(execute := #statExecute,
                                       busy => #tempBusy,
                                       done => #tempDone,
                                       error => #tempError,
                                       status => #tempStatus,
                                       toaUnit := "Toa1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

12.4.10.19 SINU_NcDeleteTool

描述

该指令删除具有 T 编号的刀具以及指定 TOA 单元中的所有关联刀沿。如果 TOA 单元中没有具有此 T 编号的刀具，则该指令会发出错误信号。

作业启动后无法更改参数。任务开始后，不能通过指令取消。

如果每个任务使用专用实例，则可同时开始多个任务。

NC 命令

该指令的调用对应于 PI 服务：

- DELETO

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

参数

下表列出了该指令的参数：

参数	声明	数据类型	描述
execute	Input	Bool	TRUE：开始任务
tool	Input	UInt	1..32000：刀具的 T 编号
toaUnit	InOut	NCK_typeToaUnit	应在其中删除刀具的 TOA 单元的注册实例
busy	Output	Bool	TRUE：任务已执行
done	Output	Bool	TRUE：任务已成功完成，“执行”可重置
error	Output	Bool	TRUE：已发生错误，“执行”可重置
status	Output	Word	作业执行后的状态编号。 发生错误时的错误编号。

错误、状态和补救措施

状态/错误代码* (W#16#...)	描述	补救措施
0000	无错误	-
7001	等待 NCK 的自由接口。	-
7002	作业发送到 NCK，等待响应。	-
7005	作业发送到 NCK，等待完成。	-
8005	TOA 实例未注册。	先注册一个 TOA 实例
8017	新任务在活动任务完成之前开始	仅在活动任务完成后，将“执行”设置回 TRUE
820b	具有 T 编号的刀具不存在	使用现有的 T 编号
820c	T 编号的值无效	遵守 T 编号的有效值范围
* 状态代码：7xxx（信息、报警），参数错误=FALSE；错误代码：8xxx（错误），参数错误 = TRUE		

示例：删除刀具

在注册实例“Toa1”的 TOA 单元中，应删除 T 编号为 42 的刀具。这是在设置静态变量 #statExecute := TRUE 后执行的。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcDeleteTool_Instance : SINU_NcDeleteTool
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcDeleteTool_Instance(execute := #statExecute,
                            tool := 42,              //T 编号
                            busy => #tempBusy,
                            done => #tempDone,
                            error => #tempError,
                            status => #tempStatus,
                            toaUnit := "Toa1");
IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

12.4.10.20 SINU_NcCreateMultiTool

说明

该指令可以通过指定一个标识符或所指定 TOA 单元中的刀具号接入多刀。

如果在 TOA 单元中已存在带有该刀具号的刀具，该指令将报告错误 16#820b。

通过该调用影响下列激活的文件系统模块：

- 多刀目录 MTV：针对该多刀所输入的状态为存在
- 块 T/MTV 中的变量“MTnumWZV”包含最后所生成的多刀的编号

说明

在执行指令之前，必须设置所属的信号量以为 NC 变量提供保护：

- 指令“SINU_NcSetSemaphoreByToa”，带 semaphoreNumber := 1、testAndSet := TRUE，并且在参数“toaUnit”中有与 TOA 单元相同的实例

在读取 NC 变量之后，所属的信号量必须重置：

- 指令“SINU_NcSetSemaphoreByToa”，带 semaphoreNumber := 1 和 testAndSet := FALSE

如果在执行指令的过程中报告了一个错误，则可以在依旧设置有信号量的情况下重复该过程。如果未重复执行指令，必须重置信号量。

任务开始后无法更改参数。一旦任务开始，就无法通过指令取消。

如果针对每项任务编制了单独的实例，则可以在不同的 TOA 单元中同时启动多项任务。

NC：命令

对指令的调用与 PI 服务相对应：

- TMCRMT

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
multitoolName	Input	String	多刀的名称，最多 31 个字符
multitoolNumber	Input	Int	1..32000：多刀的刀具号 -1：由 NC 分配刀具号
multitoolLocations	Input	USInt	2...$MN_MAX_TOOLS_PER_MULTITOOL：多刀中的刀位数量
distanceMode	Input	USInt	距离编码方式 1：刀位号 2:长度 3：角度
toaUnit	InOut	NCK_typeToaUnit	TOA 单元的已注册实例，多刀需接入其中
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置“execute”
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	TOA 实例未注册。	首先注册 TOA 实例
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将“execute”设置为 TRUE。
820a	刀具名称过长	刀具名称最长可为 31 个字符
820b	刀具所带刀具号或者刀具名称已存在	应用未使用的刀具号或者刀具名称
820c	无效参数	遵守参数的有效值域
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在已注册实例“Toa1”的 TOA 单元中，应接入一个新的多刀，其多刀编号为 42，标识为“MINIREVOLVER”且有 4 个刀位。距离编码的方式应为“刀位号”。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcCreateMultiTool_Instance : SINU_NcCreateMultiTool
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#SINU_NcCreateMultiTool_Instance(execute := #statExecute,
                                 multitoolName := 'MINIREVOLVER',
                                 multitoolNumber := 42,
                                 multitoolLocations := 4,
                                 distanceMode := 1,
                                 busy => #tempBusy,
                                 done => #tempDone,
                                 error => #tempError,
                                 status => #tempStatus,
                                 toaUnit := "Toa1");

IF #statExecute THEN
   IF #tempError OR #tempDone THEN
       #statExecute := FALSE;
   END_IF;
END_IF;
```

详细信息

SINU_NcSetSemaphoreByToa (页 1212)

12.4.10.21 SINU_NcPositionMultitool

说明

该指令可将指定 TOA 单元中的多刀定位在已编程的刀位上，或者将其定位在位于多刀刀位中的已编程刀具上。刀具可通过刀具号指定，也可通过名称和副编号指定。

仅当多刀位于一个刀套刀位（例如：主轴）上，且无基于该刀套刀位的刀具补偿生效时，才能对其进行定位。

通过刀具名称和副编号确定相应数据：

- toolName
- duploNumber
  (toolNumber := -1，location := -1)

通过刀具号确定相应数据：

- toolNumber
  (toolName := ''，duploNumber := -1，location := -1)

通过多刀刀位确定相应数据：

- 本地
  (toolName := ''，duploNumber := -1，toolNumber := -1)

其他未使用的参数要用 -1 或空字符串来标定。

说明

在执行指令之前，必须设置所属的信号量以防止并发的访问操作。

- 指令"SINU_NcSetSemaphoreByToa"，带 semaphoreNumber := 3、testAndSet := TRUE，并且在参数"toaUnit"中有与 TOA 单元相同的实例

在执行该指令之后，所属的信号量必须重置：

- 指令"SINU_NcSetSemaphoreByToa"，带 semaphoreNumber := 3 和 testAndSet := FALSE

如果在执行指令的过程中报告了一个错误，则可以在依旧设置有信号量的情况下重复该过程。如果未重复执行指令，必须重置信号量。

任务开始后无法更改参数。一旦任务开始，就无法通过指令取消。

如果针对每项任务编制了单独的实例，则可以在不同的 TOA 单元中同时启动多项任务。

NC：命令

对指令的调用与 PI 服务相对应：

- POSMT

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
toolName	Input	String	多刀的名称，最多 31 个字符 "（空字符串）：刀具名称未使用
toolholder	Input	UInt	1..32000：刀夹编号
toolNumber	Input	Int	1..32000：刀具的刀具号 -1：刀具号未使用
duploNumber	Input	Int	1..32000：多刀中需定位的刀具的副编号 -1：副编号未使用
location	Input	Int	定位的目标刀套刀位的多刀刀位编号 -1：刀位编号未使用

参数	声明	数据类型	说明
toaUnit	InOut	NCK_typeToaUnit	TOA 单元的已注册实例，多刀需定位在其中
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置“execute”
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	TOA 实例未注册。	首先注册 TOA 实例
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将“execute”设置为 TRUE。
820a	刀具名称过长	刀具名称最长可为 31 个字符
820c	参数值无效	遵守参数的有效值域
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在已注册实例“Toa1”的 TOA 单元中，带刀具 MT 编号的多刀应定位在 MT 刀位 6 上，该多刀现位于刀夹 3 中。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcPositionMultitool_Instance : SINU_NcPositionMultitool
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcPositionMultitool_Instance(execute := #statExecute,
                                   toolName := '',
                                   toolHolder := 3,   // 刀夹 3
                                   toolNumber := -1,
                                   duploNumber := -1,
                                   sourceLocation := -1,
                                   placeNumber := 6,  // 多刀中的刀位 6
                                   busy => #tempBusy,
                                   done => #tempDone,
                                   error => #tempError,
                                   status => #tempStatus,
                                   toaUnit := "Toa1");

IF #statExecute THEN
   IF #tempError OR #tempDone THEN
       #statExecute := FALSE;
   END_IF;
END_IF;
```

更多信息

SINU_NcSetSemaphoreByToa (页 1212)

12.4.10.22 SINU_NcSearchMultitoolLocation

说明

该指令用于在多刀内搜索一个用于接收指定刀具的未占用的刀位，或检查指定的用于接收指定刀具的刀位是否被占用。

刀具可通过刀具号设定，也可通过名称和副编号设定。

搜索完成时，结果可在 NC 变量中读取：

- magCMCmdPar1（多刀编号）
- magCMCmdPar2（多刀刀位编号）

说明

在执行指令之前，必须设置所属的信号量以为 NC 变量提供保护：

- 指令“SINU_NcSetSemaphoreByToa”，带 semaphoreNumber := 2、testAndSet := TRUE，并且在参数“toaUnit”中有与 TOA 单元相同的实例

在读取 NC 变量之后，所属的信号量必须重置：

- 指令“SINU_NcSetSemaphoreByToa”，带 semaphoreNumber := 2 和 testAndSet := FALSE

如果在执行指令的过程中报告了一个错误，则可以在依旧设置有信号量的情况下重复该过程。如果未重复执行指令，必须重置信号量。

任务开始后无法更改参数。一旦任务开始，就无法通过指令取消。

如果针对每项任务编制了单独的实例，则可以在不同的 TOA 单元中同时启动多项任务。

NC：命令

对指令的调用与 PI 服务相对应：

- FDPLMT

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
toolName	Input	String	"（空字符串）：已使用刀具号
toolNumber	Input	Int	-1：已使用刀具名称和副编号
duploNumber	Input	Int	1..32000：刀具的副编号 -1：已使用刀具号
multitoolNumber	Input	Int	1..32000：多刀的刀具号，在其中搜索或检查刀具的位置。 -1：针对所有多刀搜索空刀位，或针对所有多刀检查用于接收刀具的指定刀位是否被占用。
multitoolLocation	Input	Int	应进行检查的多刀中的刀位 -1：在多刀内搜索任意一个空刀位

参数	声明	数据类型	说明
toaUnit	InOut	NCK_typeToaUnit	TOA 单元的已注册实例，需在其中搜索刀位
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置“execute”
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	TOA 实例未注册。	首先注册 TOA 实例
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将“execute”设置为 TRUE。
820a	刀具名称过长	刀具名称最长可为 31 个字符
820c	参数超出有效范围	遵守参数的有效值域
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在已注册实例“Toa1”的 TOA 单元中，需检查多刀 12345 中的刀位 6 是否可用于放置刀具 4711。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcSearchMultitoolLocation_Instance : SINU_NcSearchMultitoolLocation
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcSearchMultitoolLocation_Instance(execute := #statExecute,
                                toolName := '',
                                toolNumber := 4711, // 刀具的刀具号
                                duploNumber := -1,
                                multitoolNumber := 12345, // 多刀的刀具号
                                multitoolLocationNumber := 6, // 多刀中的刀位
                                busy => #tempBusy,
                                done => #tempDone,
                                error => #tempError,
                                status => #tempStatus,
                                toaUnit := "Toa1");

IF #statExecute THEN
   IF #tempError OR #tempDone THEN
       #statExecute := FALSE;
   END_IF;
END_IF;
```

更多信息

SINU_NcSetSemaphoreByToa (页 1212)

12.4.10.23 SINU_NcDeleteMultiTool

说明

该指令可以通过指定一个标识符或所指定 TOA 单元中的刀具号删除多刀。

如果没有删除，之后多刀中配备的刀具将不再包含在多刀中且不会载入，但定义保留。

任务开始后无法更改参数。一旦任务开始，就无法通过指令取消。

如果针对每项任务编制了单独的实例，则可以同时启动多项任务。

NC：命令

对指令的调用与 PI 服务相对应：

- TMDLMT

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
multitoolName	Input	String	多刀的名称，最多 31 个字符
multitoolNumber	Input	Int	1..32000：删除带该刀具号的多刀 -1：删除指定名称为“multitoolName”的多刀
deleteTools	Input	Bool	TRUE:删除包含的刀具 FALSE：未删除包含的刀具
toaUnit	InOut	NCK_typeToaUnit	TOA 单元的已注册实例，应删除其中的多刀
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置“execute”
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-

状态/错误代码* (W#16#...)	说明	解决办法
7005	已将任务发送至 NCK，等待结束。	-
8005	TOA 实例未注册。	首先注册 TOA 实例
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将“execute”设置为 TRUE。
820a	刀具名称过长	刀具名称最长可为 31 个字符
820b	刀具所带刀具号或者刀具名称已存在	应用未使用的刀具号或者刀具名称
820c	无效参数	遵守参数的有效值域
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在已注册实例“Toa1”的 TOA 单元中，应删除标识为“MINIREVOLVER”的多刀。不允许删除所包含的刀具。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcDeleteMultiTool_Instance : SINU_NcDeleteMultiTool
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcDeleteMultiTool_Instance(execute := #statExecute,
                                 multitoolName := 'MINIREVOLVER',
                                 multitoolNumber := 1,
                                 deleteTools := FALSE,
                                 busy => #tempBusy,
                                 done => #tempDone,
                                 error => #tempError,
                                 status => #tempStatus,
                                 toaUnit := "Toa1");

IF #statExecute THEN
   IF #tempError OR #tempDone THEN
       #statExecute := FALSE;
   END_IF;
END_IF;
```

12.4.10.24 SINU_NcCreateCuttingEdge

描述

该指令为具有指定 TOA 单元中具有 T 编号的刀具，创建具有下一个自由 D 编号的刀沿。如果 TOA 单元中没有具有此 T 编号的刀具，则该指令会发出错误信号。

作业启动后无法更改参数。任务开始后，不能通过指令取消。

如果每个任务使用专用实例，则可同时开始多个任务。

NC 命令

该指令的调用对应于 PI 服务：

- CREACE

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

参数

下表列出了该指令的参数：

参数	声明	数据类型	描述
execute	Input	Bool	TRUE：开始任务
tool	Input	UInt	1..32000：刀具的 T 编号
toaUnit	InOut	NCK_typeToaUnit	应在其中创建刀具的 TOA 单元的注册实例
busy	Output	Bool	TRUE：任务已执行
done	Output	Bool	TRUE：任务已成功完成，“执行”可重置
error	Output	Bool	TRUE：已发生错误，“执行”可重置
status	Output	Word	作业执行后的状态编号。 发生错误时的错误编号。

错误、状态和补救措施

状态/错误代码* (W#16#...)	描述	补救措施
0000	无错误	-
7001	等待 NCK 的自由接口。	-
7002	作业发送到 NCK，等待响应。	-
7005	作业发送到 NCK，等待完成。	-
8005	TOA 实例未注册。	先注册一个 TOA 实例
8017	新任务在活动任务完成之前开始	仅在活动任务完成后，将“执行”设置回 TRUE
820b	具有 T 编号的刀具不存在	使用现有的 T 编号
820c	T 编号的值无效	遵守 T 编号的有效值范围
* 状态代码：7xxx（信息、报警），参数错误=FALSE；错误代码：8xxx（错误），参数错误 = TRUE		

示例：创建刀沿

在注册实例“Toa1”的 TOA 单元中，应为 T 编号为 42 的刀具创建新的刀沿。这是在设置静态变量#statExecute := TRUE 后执行的。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcCreateCuttingEdge_Instance : SINU_NcCreateCuttingEdge
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcCreateCuttingEdge_Instance(execute := #statExecute,
                            tool := 42,              //T 编号
                            busy => #tempBusy,
                            done => #tempDone,
                            error => #tempError,
                            status => #tempStatus,
                            toaUnit := "Toa1");
IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

12.4.10.25 SINU_NcCreateCuttingEdgeWithD

说明

该指令可以接入一个新刀沿。如果指定了刀具号，则可以针对该现有刀具接入新的刀沿。未指定刀具号 (tool := -1) 时，根据绝对刀沿号的模式接入刀沿。

已接入刀具的刀沿的刀沿号为 1...9。未接入刀具（绝对刀沿号的模式）的刀沿的刀沿号为 1...32000。

如果不存在带有该刀具号的刀具，或在 TOA 单元中已存在带有指定刀沿号的刀沿，该指令将报告一个错误。

任务开始后无法更改参数。一旦任务开始，就无法通过指令取消。

如果针对每项任务编制了单独的实例，则可以同时启动多项任务。

NC：命令

对指令的调用与 PI 服务相对应：

- CRCEDN

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
tool	Input	Int	1..32000：刀具的刀具号 -1：使用“cuttingEdge”中的绝对刀沿号
cuttingEdge	Input	Int	1、2...：刀沿的刀沿号 -1：刀沿号由 NC 分配
toaUnit	InOut	NCK_typeToaUnit	TOA 单元的已注册实例，需在其中接入刀沿
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置“execute”

参数	声明	数据类型	说明
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	TOA 实例未注册。	首先注册 TOA 实例
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将“execute”设置为 TRUE。
820b	刀具所带刀具号不存在或已存在带刀沿号的刀沿	使用现有的刀具号或使用空闲的刀沿号
820c	刀具号或刀沿号的无效值	遵守刀具号或刀沿号的有效值域
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在已注册实例“Toa1”的 TOA 单元中，应为刀具号为 42 的刀具接入一个刀沿号为 3 的新刀沿。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcCreateCuttingEdgeWithD_Instance : SINU_NcCreateCuttingEdgeWithD
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#SINU_NcCreateCuttingEdgeWithD_Instance(execute := #statExecute,
                                        tool := 42,       //刀具号
                                        cuttingEdge := 3, // 刀沿号
                                        busy => #tempBusy,
                                        done => #tempDone,
                                        error => #tempError,
                                        status => #tempStatus,
                                        toaUnit := "Toa1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

12.4.10.26 SINU_NcDeleteCuttingEdge

说明

该指令可以删除指定刀具中的刀沿或带有绝对刀沿号的刀沿当使用绝对刀沿号时，必须在参数 tool :=-1 中设置刀具号。

如果不存在带有该刀具号的刀具，或在 TOA 单元中不存在带有该刀沿号的刀沿，该指令将报告一个错误。

任务开始后无法更改参数。一旦任务开始，就无法通过指令取消。

如果针对每项任务编制了单独的实例，则可以同时启动多项任务。

NC：命令

对指令的调用与 PI 服务相对应：

- DELECE

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
tool	Input	Int	1..32000：刀具的刀具号 -1：使用"cuttingEdge"中的绝对刀沿号
cuttingEdge	Input	UInt	1、2...：刀沿的刀沿号
toaUnit	InOut	NCK_typeToaUnit	TOA 单元的已注册实例，应删除其中的刀具
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置"execute"
error	Output	Bool	TRUE:出错，可以重置"execute"
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	TOA 实例未注册。	首先注册 TOA 实例
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将"execute"设置为 TRUE。
820b	刀具所带刀具号不存在	使用现有刀具的刀具号
820c	刀具号或刀沿号的无效值	遵守刀具号或刀沿号的有效值域
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在已注册实例"Toa1"的 TOA 单元中，应删除刀具号为 42 的刀具中刀沿号为 3 的刀沿。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcDeleteCuttingEdge_Instance : SINU_NcDeleteCuttingEdge
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcDeleteCuttingEdge_Instance(execute := #statExecute,
                                   tool := 42,       //刀具号
                                   cuttingEdge := 3, // 刀沿号
                                   busy => #tempBusy,
                                   done => #tempDone,
                                   error => #tempError,
                                   status => #tempStatus,
                                   toaUnit := "Toa1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

12.4.10.27 SINU_NcPositionMagazine

说明

该指令能运行刀库，使刀库位置定位至规定的位置（例如：至装刀位）。可直接指定刀库位置或者通过位于刀位上的刀具指定。该指令可通过刀具名称和副编号确定刀具的刀具号。

通过刀具名称和副编号确定相应数据：

- toolName
- duploNumber
  (toolNumber := -1，sourceLocation := -1，sourceMagazine := -1)

通过刀具号确定相应数据：

- toolNumber
  (toolName := ''，duploNumber := -1，sourceLocation := -1，sourceMagazine := -1)

通过刀库和刀位确定相应数据：

- sourceLocation
- sourceMagazine
  (toolName := ''，duploNumber := -1，toolNumber := -1)

其他未使用的参数要用 -1 或空字符串来标定。

说明

在执行指令之前，必须设置所属的信号量以防止并发的访问操作。

- 指令"SINU_NcSetSemaphoreByToa"，带 semaphoreNumber := 3、testAndSet := TRUE，并且在参数"toaUnit"中有与 TOA 单元相同的实例

在执行该指令之后，所属的信号量必须重置：

- 指令"SINU_NcSetSemaphoreByToa"，带 semaphoreNumber := 3 和 testAndSet := FALSE

如果在执行指令的过程中报告了一个错误，则可以在依旧设置有信号量的情况下重复该过程。如果未重复执行指令，必须重置信号量。

任务开始后无法更改参数。一旦任务开始，就无法通过指令取消。

如果针对每项任务编制了单独的实例，则可以在不同的 TOA 单元中同时启动多项任务。

NC：命令

对指令的调用与 PI 服务相对应：

- TMPOSM

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
toolName	Input	String	刀具的名称，最多 31 个字符
toolNumber	Input	Int	1..32000：刀具的刀具号 -1：刀具号未使用
duploNumber	Input	Int	1..32000：刀具的副编号 -1：副编号未使用

参数	声明	数据类型	说明
sourceMagazine	Input	Int	应定位的刀库 -1：刀库未使用
sourceLocation	Input	Int	应定位的刀库中的刀位 -1：刀位未使用
referenceMagazine	Input	UInt	刀库，刀具或刀库位需定位到此
referenceLocation	Input	UInt	刀位，刀具或刀库位需定位到此
toaUnit	InOut	NCK_typeToaUnit	TOA 单元的已注册实例，刀库需定位在其中
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置“execute”
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	TOA 实例未注册。	首先注册 TOA 实例
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将“execute”设置为 TRUE。
820a	刀具名称过长	刀具名称最长可为 31 个字符
820c	参数值无效	遵守参数的有效值域
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在已注册实例"Toa1"的 TOA 单元中，刀具号为 1 的刀具所在的刀库位将被移到装载刀库 (9999) 的装刀刀位 2 前面。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcPositionMagazine_Instance : SINU_NcPositionMagazine
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcPositionMagazine_Instance(execute := #statExecute,
                                  toolName := '',
                                  toolNumber := 1,     // 刀具号
                                  duploNumber := -1,
                                  sourceMagazine := -1,
                                  sourceLocation := -1,
                                  referenceMagazine := 9999, //装载刀库
                                  referenceLocation := 2,    // 目标刀位
                                  busy => #tempBusy,
                                  done => #tempDone,
                                  error => #tempError,
                                  status => #tempStatus,
                                  toaUnit := "Toa1");

IF #statExecute THEN
   IF #tempError OR #tempDone THEN
       #statExecute := FALSE;
   END_IF;
END_IF;
```

更多信息

SINU_NcSetSemaphoreByToa (页 1212)

12.4.10.28 SINU_NcSearchLocation

描述

该指令在 TOA 单元的指定刀库中执行空位置搜索。执行此操作时，会考虑指定的位置类型和刀具尺寸。

执行成功后，刀库编号位于 NC 变量 magCMCmdPar1 中，位置编号位于 magCMCmdPar2 中，后者可用作刀具的目标位置。

如果没有为刀具找到合适的目标位置，则该指令会发出错误 16#820b 的信号。

说明

执行指令之前，为保护 NC 变量 magCMCmdPar1 和 magCMCmdPar2，必须设置关联的信号量：

- 指令“SINU_NcSetSemaphoreByToa”，带信号量编号 := 2, testAndSet := TRUE 和参数 toaUnit 中 TOA 单元的相同实例

读取 NC 变量 magCMCmdPar1 和 magCMCmdPar2 后，必须重置关联的信号量：

- 带信号量编号:= 2 和 testAndSet := FALSE 的指令“SINU_NcSetSemaphoreByToa”

如果指令在执行前发出错误信号，则仍可使用设置的信号量重复该步骤。如果不重复执行，则必须重置信号量。

作业启动后无法更改参数。任务开始后，不能通过指令取消。

如果每个作业使用专用实例，则可在不同的 TOA 单元中同时启动多个作业。

NC 命令

该指令的调用对应于 PI 服务：

- TMFPBP

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

参数

下表列出了该指令的参数：

参数	声明	数据类型	描述
execute	Input	Bool	TRUE：开始任务
firstSearchMagazine	Input	Int	要从中执行搜索的刀库的编号 -1:从第一个开始搜索所有刀库
firstSearchLocation	Input	Int	要从中执行搜索的刀库中的位置编号 -1:搜索此刀库中的所有位置
lastSearchMagazine	Input	Int	要执行搜索的刀库的编号 -1:搜索所有刀库，直到最后一个

参数	声明	数据类型	描述
lastSearchLocation	Input	Int	要执行搜索的刀库中的位置编号 -1:搜索此刀库中的所有位置
referenceMagazine	Input	Int	刀库编号，应用作搜索参考 -1:无参考的搜索策略
referenceLocation	Input	Int	刀库中的位置编号，应用作搜索参考 -1:无参考的搜索策略
toolSizeLeft	Input	UInt	左侧所需的半位置数
toolSizeRight	Input	UInt	右侧所需的半位置数
toolSizeTop	Input	UInt	向上方向所需的半位置数
toolSizeBottom	Input	UInt	向下方向所需的半位置数
toolLocationType	Input	UInt	所需位置类型
searchDir	Input	UInt	0：搜索策略，如 $TC_MAMP2 中所定义的 1：正向搜索 2：反向搜索 3：对称搜索
toaUnit	InOut	NCK_typeToaUnit	应在其中执行空位置搜索的 TOA 单元的注册实例
busy	Output	Bool	TRUE：任务已执行
done	Output	Bool	TRUE：任务已成功完成，“执行”可重置
error	Output	Bool	TRUE：已发生错误，“执行”可重置
status	Output	Word	作业执行后的状态编号。 发生错误时的错误编号。

错误、状态和补救措施

状态/错误代码* (W#16#...)	描述	补救措施
0000	无错误	-
7001	等待 NCK 的自由接口。	-
7002	作业发送到 NCK，等待响应。	-

状态/错误代码* (W#16#...)	描述	补救措施
7005	作业发送到 NCK，等待完成。	-
8005	TOA 实例未注册。	先注册一个 TOA 实例
8017	新任务在活动任务完成之前开始	仅在活动任务完成后，将“执行”设置回 TRUE
820b	未找到该刀具的适当空位置	在刀库中留出空间或扩大搜索范围 检查刀具和刀库的位置类型 ($TC_MPP7) 考虑相邻位置的数据 > 0 ($TC_MPP3..6)
820c	参数的值无效	遵守参数的有效值范围
* 状态代码：7xxx（信息、报警），参数错误=FALSE；错误代码：8xxx（错误），参数错误 = TRUE		

示例：查找空位置

在注册实例“Toa1”的 TOA 单元中，应为位置类型为 5 和尺寸为两个半位置的刀具找到适当的空位置。这是在设置静态变量#statExecute := TRUE 后执行的。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcSearchLocation_Instance : SINU_NcSearchLocation
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#SINU_NcSearchLocation_Instance(execute := #statExecute,
                                firstSearchMagazine := 1, // 在刀库 1 中搜索
                                firstSearchLocation := -1,// 位置
                                lastSearchMagazine := 1,
                                lastSearchLocation := -1,
                                referenceMagazine := -1,
                                referenceLocation := -1,
                                toolSizeLeft := 2,       // 2 个半位置
                                toolSizeRight := 2,
                                toolSizeTop := 2,
                                toolSizeBottom := 2,
                                toolLocationType := 5,  // 位置类型 5
                                searchDir := 1,         // 正向搜索
                                busy => #tempBusy,
                                done => #tempDone,
                                error => #tempError,
                                status => #tempStatus,
                                toaUnit := "Toa1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

更多信息

- SINU_NcSetSemaphoreByToa (页 1212)
- SINU_NcSearchLoadingLocation (页 1206)

12.4.10.29 SINU_NcSearchLoadingLocation

描述

该指令执行空位置搜索，以加载具有指定 TOA 单元中的 T 编号的刀具。

执行成功后，刀库编号位于 NC 变量 magCMCmdPar1 中，位置编号位于 magCMCmdPar2 中，后者可用作装载刀具的目标位置。

如果没有为刀具找到合适的目标位置，则该指令会发出错误 16#820b 的信号。

说明

执行指令之前，为保护 NC 变量 magCMCmdPar1 和 magCMCmdPar2，必须设置关联的信号量：

- 指令“SINU_NcSetSemaphore”，带 semaphoreNumberByToa := 2, testAndSet := TRUE 和参数 toaUnit 中 TOA 单元的相同实例

读取 NC 变量 magCMCmdPar1 和 magCMCmdPar2 后，必须重置关联的信号量：

- 带 semaphoreNumberByToa := 2 和 testAndSet := FALSE 的指令“SINU_NcSetSemaphore”

如果指令在执行前发出错误信号，则仍可使用设置的信号量重复该步骤。如果不重复执行，则必须重置信号量。

作业启动后无法更改参数。任务开始后，不能通过指令取消。

如果每个作业使用专用实例，则可在不同的 TOA 单元中同时启动多个作业。

NC 命令

该指令的调用对应于 PI 服务：

- TMFDPL

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

参数

下表列出了该指令的参数：

参数	声明	数据类型	描述
execute	Input	Bool	TRUE：开始任务
tool	Input	UInt	1..32000：刀具的 T 编号
searchMagazine	Input	Int	空位置搜索的刀库编号 -1:在 TOA 单元的所有刀库中搜索
searchLocation	Input	Int	空位置检查的位置编号 -1:搜索刀库中的空位置
referenceMagazine	Input	Int	刀库编号，应用作搜索参考 -1:无参考的搜索策略

参数	声明	数据类型	描述
referenceLocation	Input	Int	刀库中的位置编号，应用作搜索参考 -1:无参考的搜索策略
toaUnit	InOut	NCK_typeToaUnit	应在其中装载刀具的 TOA 单元的注册实例
busy	Output	Bool	TRUE：任务已执行
done	Output	Bool	TRUE：任务已成功完成，“执行”可重置
error	Output	Bool	TRUE：已发生错误，“执行”可重置
status	Output	Word	作业执行后的状态编号。 发生错误时的错误编号。

错误、状态和补救措施

状态/错误代码* (W#16#...)	描述	补救措施
0000	无错误	-
7001	等待 NCK 的自由接口。	-
7002	作业发送到 NCK，等待响应。	-
7005	作业发送到 NCK，等待完成。	-
8005	TOA 实例未注册。	先注册一个 TOA 实例
8017	新任务在活动任务完成之前开始	仅在活动任务完成后，将“执行”设置回 TRUE
820b	未找到该刀具的适当空位置	在刀库中留出空间或扩大搜索范围 检查刀具和刀库的位置类型 ($TC_MPP7) 考虑相邻位置的数据 > 0 ($TC_MPP3..6)
820c	参数的值无效	遵守参数的有效值范围
* 状态代码：7xxx（信息、报警），参数错误=FALSE；错误代码：8xxx（错误），参数错误 = TRUE		

示例：空位置搜索和装载

在注册实例“Toa1”的 TOA 单元中，应为 T 编号为 42 的刀具找到一个适当的空位置，以便随后装载刀具。这是在设置静态变量#statExecute := TRUE 后执行的。

反馈保存在临时变量#tempBusy、#tempDone、#tempError 和#tempStatus 中。

SCL

```
// SINU_NcSearchLoadingLocation_Instance : SINU_NcSearchLoadingLocation
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcSearchLoadingLocation_Instance(
                          execute := #statExecute,
                          tool := 42,           // T 编号
                          searchMagazine := 1,  //  在刀库 1 中搜素
                          searchLocation := -1, //  位置
                          referenceMagazine := -1,
                          referenceLocation := -1,
                          busy => #tempBusy,
                          done => #tempDone,
                          error => #tempError,
                          status => #tempStatus,
                          toaUnit := "Toa1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

更多信息

- SINU_NcSetSemaphoreByToa (页 1212)
- SINU_NcSearchLocation (页 1202)

12.4.10.30 SINU_NcSetWorkpieceCounterInc

说明

该指令可在指定 TOA 单元中设置用于主轴计件器的增量值。

任务开始后无法更改参数。一旦任务开始，就无法通过指令取消。

如果针对每项任务编制了单独的实例，则可以同时启动多项任务。

NC：命令

对指令的调用与 PI 服务相对应：

- TMPCIT

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
spindle	Input	SInt	1、2、...主轴编号；对应刀位数据中的类型索引，在通道中中间存储刀库的刀位类型为主轴时。 -1：主主轴
increment	Input	UInt	0..32000：增量值；设定计件器在多少次主轴旋转后增量提升计数
toaUnit	InOut	NCK_typeToaUnit	TOA 单元的已注册实例，需在其中设置增量值
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置“execute”
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-

状态/错误代码* (W#16#...)	说明	解决办法
8005	TOA 实例未注册。	首先注册 TOA 实例
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将"execute"设置为 TRUE。
820c	参数值无效	遵守参数的有效值域
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在已注册实例"Toa1"的 TOA 单元中，主轴 1 的计件器将在主轴每转 2500 圈后增量提升计数。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcSetWorkpieceCounterInc_Instance : SINU_NcSetWorkpieceCounterInc
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word

#SINU_NcSetWorkpieceCounterInc_Instance(execute := #statExecute,
                                        spindle := 1,
                                        increment := 2500,
                                        busy => #tempBusy,
                                        done => #tempDone,
                                        error => #tempError,
                                        status => #tempStatus,
                                        toaUnit := "Toa1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

12.4.10.31 SINU_NcSetSemaphoreByToa

说明

该指令可检查或设置用于 HMI 和 PLC 的信号量，以便在执行某些指令时为某些关键数据区域提供保护。同样必须重置信号量。

在与刀具管理的 PI 服务有关的情况下，优先使用该指令而不是 SINU_NcSetSemaphore。

调用指令时，如果没有事先进行设置，才会设置信号量。如果已设置信号量，该指令会报告一个错误。

任务开始后无法更改参数。一旦任务开始，就无法通过指令取消。

如果针对每项任务使用了单独的实例，则可以同时在不同的 TOA 单元中或针对不同的信号量启动多项任务。

在执行以下任何指令之前，必须使用相同的 TOA 单元检查并设置所属的信号量 (testAndSet := TRUE)：

- SINU_NcCreateToolByName
- SINU_NcSearchLoadingLocation
- SINU_NcMoveTool
- SINU_NcSearchLocation
- SINU_NcGetToolNumber
- SINU_NcSearchTool
- SINU_NcSearchMultitoolLocation
- SINU_NcPositionMultitool
- SINU_NcPositionMagazine
- SINU_NcCreateMultiTool

该指令可在 TOA 单元所属的通道（编号最小）中设置信号量。

只有在成功设置信号量后，才能读取用于回传的变量。在从相应的 NC 变量中读取回传的指令之后，必须再次重置信号量 (testAndSet := FALSE)。如果未重置信号量，就无需重新设置。

利用已设置的信号量可以根据需要频繁调用相关指令，而不必每次都重新设置信号量。

其它没有分配给指令的信号量也可以自由使用。

NC：命令

对指令的调用与 PI 服务相对应：

- MMCSEM

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:启动任务
semaphoreNumber	Input	USInt	1：für SINU_NcCreateToolByName 1：用于 SINU_NcCreateMultiTool 2:用于 SINU_NcSearchLocation 2:用于 SINU_NcSearchLoadingLocation 2:用于 SINU_NcSearchMultitoolLocation 3：用于 SINU_NcMoveTool 3：用于 SINU_NcPositionMultitool 3：用于 SINU_NcPositionMagazine 4..5：可自由使用 6:用于 SINU_NcGetToolNumber 6:用于 SINU_NcSearchTool 7..10：可自由使用
testAndSet	Input	Bool	TRUE:检查是否未设置信号量，然后设置 FALSE：复位信号量
toaUnit	InOut	NCK_typeToaUnit	TOA 单元的已注册实例。在第一个或编号最小的通道（已分配给 TOA 单元）中设置信号量。
busy	Output	Bool	TRUE:已执行任务
done	Output	Bool	TRUE:已成功结束任务，可以重置“execute”
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	TOA 单元的实例未注册。	首先请注册 TOA 单元的实例
8017	在当前任务完成前开始的新任务	在当前任务结束之前，不得再次将“execute”设置为 TRUE。
820b	信号量已设置，无法再进行设置	如果已自行设置，请重置信号量，或者等待信号量再次空闲可用
820c	参数超出有效范围	遵守参数的有效值域
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例：设置信号量

在已注册实例“Toa2”的 TOA 单元中，应设置用于空刀位搜索 (SINU_NcSearchLocation) 的信号量。此操作在设置静态变量 #statExecute := TRUE 后执行。已在所属的且编号最小的 TOA 单元通道中设置了信号量。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

如果已设置信号量，则会重复尝试设置信号量。如果信号量已成功设置，将设置 #statSemaphore := TRUE。

SCL

```
// SINU_NcRetractTool_Instance : SINU_NcRetractTool
// statExecute : Bool
// statRetry : Bool
// statSemaphore : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#SINU_NcSetSemaphoreByToa_Instance(execute := #statExecute,
                                   semaphoreNumber := 2, // 用于空刀位搜索
                                   testAndSet := TRUE,   // 检查和设置
                                   busy => #tempBusy,
                                   done => #tempDone,
                                   error => #tempError,
                                   status => #tempStatus,
                                   toaUnit := "Toa2");

IF #statRetry THEN
    #statExecute := TRUE;
    #statRetry := FALSE;
ELSIF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;

    #statRetry := #tempError AND #statSetOrReset;
    #statSemaphore := #tempDone;
END_IF;
```

示例：复位信号量

在已注册实例"Toa2"的 TOA 单元中，应重置用于空刀位搜索 (SINU_NcSearchLocation) 的信号量。此操作在设置静态变量 #statExecute := TRUE 后执行。已在所属的且编号最小的 TOA 单元通道中重置了信号量。

反馈信息存储在临时变量 #tempBusy、#tempDone、#tempError 和 #tempStatus 中。

SCL

```
// SINU_NcSetSemaphoreByToa_Instance : SINU_NcSetSemaphoreByToa
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#SINU_NcSetSemaphoreByToa_Instance(execute := #statExecute,
                                   semaphoreNumber := 2, // 用于空刀位搜索
                                   testAndSet := FALSE,  // 重置
                                   busy => #tempBusy,
                                   done => #tempDone,
                                   error => #tempError,
                                   status => #tempStatus,
                                   toaUnit := "Toa2");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

更多信息

SINU_NcSetSemaphore (页 1152)

12.4.11 轴与主轴功能

12.4.11.1 SINU_TraversePosAxis

说明

该指令可控制定位轴。除了目标位置或要移动的距离外，还可以指定接近方向和速度。

为了借助该指令移动定位轴，轴的控制必须首先从 NC 传输到 PLC。参见子章节“PLC 轴和轴交换”。

任务开始后无法更改参数。一旦任务开始执行，就无法通过指令取消。要停止移动的轴，请改用“删除剩余行程 (页 1251)”功能（参见 <Axis>.basic.out.resetMovement）。

可以为同一加工轴编程该指令的多个实例。在这种情况下，依次处理多个同时开始的任务。如果该指令与其他指令同时用于同一加工轴的轴/主轴控制，也同样如此。

可同时控制不同的加工轴。必须为每个加工轴使用单独的指令实例，并且必须对调用进行编程。

借助参数“ImperialUnits := TRUE”指定的是英制单位（inch 和 inch/min）而不是公制，并且与 NC 中的当前设置无关。

参数“modeAbsolute”、“modeIncremental”、“modeShortest”、“modeNegativeDir”、“modePositiveDir”之一必须设置为 TRUE。不能将这些参数中的超过一个设置为 TRUE。

NC：命令

对指令的调用与以下 NC 命令相对应：

- POS[axis] = pos oder POS[axis] = AC(pos) 带参数 modeAbsolute := TRUE
- POS[axis] = IC(pos) 带参数 modeIncremental := TRUE
- POS[axis] = DC(pos) 带参数 modeShortest := TRUE
- POS[axis] = ACN(pos) 带参数 modeNegativeDir := TRUE
- POS[axis] = ACP(pos) 带参数 modePositiveDir := TRUE
- FDA[axis] = feedRate 带参数 handwheelOvr := TRUE

PLC 轴和轴交换

在轴可以由 PLC 控制之前，必须将轴激活为 PLC 轴（从 NC 到 PLC 的轴交换）。

使用以下信号来进行轴交换，并注意相应的信号：

- `<axis>.interchange.out.assignToPlc = TRUE;`
- `<axis>.interchange.out.requireSwitchOver = TRUE;`

控制回转后（从 PLC 到 NC 的轴交换），轴可以再次由 NC 程序控制。

使用以下信号来进行轴交换，并注意相应的信号：

- `<axis>.interchange.out.assignToPlc = FALSE;`
- `<axis>.interchange.out.requireSwitchOver = TRUE;`

在引导启动期间，固定分配的 PLC 轴（MD30460 位 5 = 1，<Axis>.plcCtrl.in.plcCtrlPermanent = TRUE）变为“中立轴”。在运行任务中，定位轴自动成为 PLC 轴，无需提前进行轴交换。

轴禁止

如果设置了轴禁用 <Axis>.basic.out.disable 或 <Axis>.basic.in.disabled，通过该指令控制的轴不会移动，仅生成模拟的实际值。当轴被 NC 禁用时，行为与轴运行时相同。

耦联组中的轴

如果轴处于耦联组中，那么这些轴只能借助指令通过引导轴来移动。无法通过跟随轴来控制耦联组。

只有设置 <Axis>.coupling.out.enableFollowOverlay := TRUE 后，才能控制跟随轴。然后仅跟随轴本身受控。

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明	
execute	Input	Bool	TRUE:用于控制轴的任务	
modeAbsolute	Input	Bool	TRUE:定位到绝对位置	这些参数之一必须为 TRUE
modeShortest	Input	Bool	TRUE:以最短路径定位到绝对位置	
modePositiveDir	Input	Bool	TRUE:沿正接近方向定位到绝对位置	
modeNegativeDir	Input	Bool	TRUE:沿负接近方向定位到绝对位置	
modeIncremental	Input	Bool	TRUE:增量式定位：“pos”是距离，不是绝对位置	
pos	Input	Real	绝对位置（mm 或度）。 "modeIncremental" := TRUE:待移动的距离（mm 或度）。 "imperialUnits" := TRUE:用“inch”代替“mm”。	
feedRate	Input	Real	定位速度（mm/min 或 rpm）。 "imperialUnits" := TRUE:用 inch/min，而不是 mm/min 0：使用 MD32060 $MA_POS_AX_VELO 中的速度	
imperialUnits	Input	Bool	TRUE：用 inch/min，而不是 mm/min 来指定“speed”	
handwheelOvr	Input	Bool	TRUE:手轮的手轮脉冲作为定位速度的速度叠加起作用	
axis	InOut	NCK_typeAxis	要控制的 NCK 加工轴的已注册实例	
busy	Output	Bool	TRUE:执行用于控制轴的任务	
done	Output	Bool	TRUE:用于控制轴的任务顺利完成，可以重置“execute”	
error	Output	Bool	TRUE:出错，可以重置“execute”	
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。	

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	实例未注册。	在控制轴之前，请先注册轴实例。
8012	轴不是 PLC 轴。	在控制轴之前，请先进行轴交换。 参见子章节"PLC 轴和轴交换"
8204	模式不具有唯一性，	请将模式参数之一设置为 TRUE。
0001-00FF	来自 NC 的故障标识，参见表格"来自 NC 的故障标识"。	必要时参见 NC 报警
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

来自 NC 的故障标识

故障标识 (W#16#...)	说明
0001	已同时激活多个轴/主轴功能
0014	启动了一个功能，但未到达位置
001E	轴/主轴的控制权在运动结束前即已移交给 NC
0028	通过 NC 程序对轴进行编程，NC 内部故障。
0032	固定分配的 PLC 轴：运行 (JOG) 或回参考点
003C	固定分配的 PLC 轴：通道状态不允许开始
0064	对轴/主轴编程时的位置错误（对应报警 16830）
0065	编写的转速过高
0066	恒定切削速度的取值范围错误（对应报警 14840）
0068	副主轴:不允许的编程（对应报警 22030）
0069	无测量系统（对应报警 16770）
006A	轴定位尚在生效（对应报警 22052）
006B	未找到参考标记（对应报警 22051）

故障标识 (W#16#...)	说明
006C	从转速闭环控制到位置闭环控制没有过渡（对应报警 22050）
006D	未找到参考标记（对应报警 22051）
006E	速度/转速为负
006F	设定转速 == 零
0070	无效的齿轮档
0073	未到达编程的位置
0075	在 NC 中 G96/G961 未生效
0076	在 NC 中 G96/G961 仍有效
0078	该轴不是分度轴（对应报警 20072）
0079	分度位置错误（对应于报警 17510）
007D	DC（最短行程）无法启用（对应报警 16800）
007E	无法启用负绝对值（对应报警 16820）
007F	无法启用正绝对值（对应报警 16810）
0080	无用于直径编程的平面轴（对应报警 16510）
0082	软件限位开关正（对应报警 20070）
0083	软件限位开关负（对应报警 20070）
0084	工作区域限制正（对应报警 20071）
0085	工作区域限制负（对应报警 20071）
0086	框架不可用于分度轴
0087	配备“端面齿”的分度轴生效（对应报警 17501）
0088	配备“端面齿”的分度轴生效，且轴未回参考点（对应报警 17503）
0089	不可为经过转换的主轴/轴启动主轴模式（对应报警 22290）
008A	轴：超出特定坐标系中的正向工作区域限制（对应报警 20082）
008B	轴：超出特定坐标系中的负向工作区域限制（对应报警 20082）
00C8	系统故障：对应于报警 450007

示例：增量式移动定位轴

属于已注册实例“X1”的轴将相对于当前位置移动 100 mm (modeIncremental := TRUE)。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈保存在临时变量“tempBusy”、“tempDone”、“tempError”和“tempStatus”中。

到达位置 (tempDone = TRUE) 或出现故障 (tempError = TRUE) 后，会自动通过 #statExecute := FALSE 结束任务。

SCL

```
// SINU_TraversePosAxis_Instance : SINU_TraversePosAxis
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
#SINU_TraversePosAxis_Instance(execute := #statExecute,
                               modeAbsolute := FALSE,
                               modeShortest := FALSE,
                               modePositiveDir := FALSE,
                               modeNegativeDir := FALSE,
                               modeIncremental := TRUE,
                               pos := 100.0,        // 100 mm
                               feedRate := 600.0,   // 600 mm/min
                               imperialUnits := FALSE,
                               handwheelOvr := FALSE,
                               busy => #tempBusy,
                               done => #tempDone,
                               error => #tempError,
                               status => #tempStatus,
                               axis := "X1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

示例：为旋转轴定位

属于已注册实例“A1”的主轴将以 1 rpm 的速度定位在正接近方向上 90° 处 (modePositiveDir := TRUE)。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈保存在临时变量“tempBusy”、“tempDone”、“tempError”和“tempStatus”中。

到达位置 (tempDone = TRUE) 或出现故障 (tempError = TRUE) 后，会自动通过 #statExecute := FALSE 结束任务。

SCL

```
// SINU_TraversePosAxis_Instance : SINU_TraversePosAxis
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
#SINU_TraversePosAxis_Instance(execute := #statExecute,
                               modeAbsolute := FALSE,
                               modeShortest := FALSE,
                               modePositiveDir := TRUE,
                               modeNegativeDir := FALSE,
                               modeIncremental := FALSE,
                               pos := 90.0,       // 90°
                               feedRate := 1.0,   // 1 rpm
                               imperialUnits := FALSE,
                               handwheelOvr := FALSE,
                               busy => #tempBusy,
                               done => #tempDone,
                               error => #tempError,
                               status => #tempStatus,
                               axis := "A1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

示例：以英制单位为轴定位

属于已注册实例“Y1”的轴应以 50 inch/min 的速度定位在位置 10 inch 处（modeAbsolute := TRUE 且 imperialUnits := TRUE）。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈保存在临时变量“tempBusy”、“tempDone”、“tempError”和“tempStatus”中。

到达位置 (tempDone = TRUE) 或出现故障 (tempError = TRUE) 后，会自动通过 #statExecute := FALSE 结束任务。

SCL

```
// SINU_TraversePosAxis_Instance : SINU_TraversePosAxis
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
#SINU_TraversePosAxis_Instance(execute := #statExecute,
                               modeAbsolute := TRUE,
                               modeShortest := FALSE,
                               modePositiveDir := FALSE,
                               modeNegativeDir := FALSE,
                               modeIncremental := FALSE,
                               pos := 10.0,        // 10 inch
                               feedRate := 50.0,   // 50 inch/min
                               imperialUnits := TRUE,
                               handwheelOvr := FALSE,
                               busy => #tempBusy,
                               done => #tempDone,
                               error => #tempError,
                               status => #tempStatus,
                               axis := "Y1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

参见

错误代码和状态代码 (页 1345)

12.4.11.2 SINU_TraverseIndexingAxis

说明

该指令可控制分度轴。除了目标位置或要移动的距离外，还可以指定接近方向和速度。

为了借助该指令移动分度轴，轴的控制必须首先从 NC 传输到 PLC。参见子章节“PLC 轴和轴交换”。

任务开始后无法更改参数。一旦任务开始执行，就无法通过指令取消。要停止移动的轴，请改用“删除剩余行程 (页 1251)”功能（参见 <Axis>.basic.out.resetMovement）。

可以为同一加工轴编程该指令的多个实例。在这种情况下，依次处理多个同时开始的任务。如果该指令与其他指令同时用于同一加工轴的轴/主轴控制，也同样如此。

可同时控制不同的加工轴。必须为每个加工轴使用单独的指令实例，并且必须对调用进行编程。

借助参数“ImperialUnits := TRUE”指定的是英制单位 (inch/min) 而不是公制，并且与 NC 中的当前设置无关。

参数“modeAbsolute”、“modeIncremental”、“modeShortest”、“modeNegativeDir”、“modePositiveDir”之一必须设置为 TRUE。不能将这些参数中的超过一个设置为 TRUE。

NC：命令

对指令的调用与以下 NC 命令相对应：

- POS[axis] = CAC(codedPosition) 带参数 modeAbsolute := TRUE
- POS[axis] = CIC(codedPosition) 带参数 modeIncremental := TRUE
- POS[axis] = CDC(codedPosition) 带参数 modeShortest := TRUE
- POS[axis] = CACN(codedPosition) 带参数 modeNegativeDir := TRUE
- POS[axis] = CACP(codedPosition) 带参数 modePositiveDir := TRUE

PLC 轴和轴交换

在轴可以由 PLC 控制之前，必须将轴激活为 PLC 轴（从 NC 到 PLC 的轴交换）。

使用以下信号来进行轴交换，并注意相应的信号：

- `<axis>.interchange.out.assignToPlc = TRUE;`
- `<axis>.interchange.out.requireSwitchOver = TRUE;`

控制回转后（从 PLC 到 NC 的轴交换），轴可以再次由 NC 程序控制。

使用以下信号来进行轴交换，并注意相应的信号：

- `<axis>.interchange.out.assignToPlc = FALSE;`
- `<axis>.interchange.out.requireSwitchOver = TRUE;`

在引导启动期间，固定分配的 PLC 轴（MD30460 位 5 = 1，<Axis>.plcCtrl.in.plcCtrlPermanent = TRUE）变为“中立轴”。在运行任务中，定位轴自动成为 PLC 轴，无需提前进行轴交换。

轴禁止

如果设置了轴禁用 <Axis>.basic.out.disable 或 <Axis>.basic.in.disabled，通过该指令控制的轴不会移动，仅生成模拟的实际值。当轴被 NC 禁用时，行为与轴运行时相同。

耦联组中的轴

如果轴处于耦联组中，那么这些轴只能借助指令通过引导轴来移动。无法通过跟随轴来控制耦联组。

只有设置 <Axis>.coupling.out.enableFollowOverlay := TRUE 后，才能控制跟随轴。然后仅跟随轴本身受控。

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明	
execute	Input	Bool	TRUE:用于控制轴的任务	
modeAbsolute	Input	Bool	TRUE:定位到绝对位置	这些参数之一必须为 TRUE
modeShortest	Input	Bool	TRUE:以最短路径定位到绝对位置	
modePositiveDir	Input	Bool	TRUE:沿正接近方向定位到绝对位置	
modeNegativeDir	Input	Bool	TRUE:沿负接近方向定位到绝对位置	
modeIncremental	Input	Bool	TRUE:增量式定位：“indexingPos”是距离，不是绝对位置	
indexingPos	Input	DInt	绝对分度位置。 "modeIncremental" := TRUE:待移动距离（分度位置）。	
speed	Input	Real	定位速度（mm/min 或 rpm）。 "imperialUnits" := TRUE:用 inch/min，而不是 mm/min 0：使用 MD32060 $MA_POS_AX_VELO 中的速度	
imperialUnits	Input	Bool	TRUE：用 inch/min，而不是 mm/min 来指定“speed”	
axis	InOut	NCK_typeAxis	要控制的 NCK 加工轴的已注册实例	
busy	Output	Bool	TRUE:执行用于控制轴的任务	

参数	声明	数据类型	说明
done	Output	Bool	TRUE:用于控制轴的任务顺利完成，可以重置“execute”
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	实例未注册。	在控制轴之前，请先注册轴实例。
8012	轴不是 PLC 轴。	在控制轴之前，请先进行轴交换。 参见子章节“PLC 轴和轴交换”
8204	模式不具有唯一性，	请将模式参数之一设置为 TRUE。
0001-00FF	来自 NC 的故障标识，参见表格“来自 NC 的故障标识”。	必要时参见 NC 报警
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

来自 NC 的故障标识

故障标识 (W#16#...)	说明
0001	已同时激活多个轴/主轴功能
0014	启动了一个功能，但未到达位置
001E	轴/主轴的控制权在运动结束前即已移交给 NC
0028	通过 NC 程序对轴进行编程，NC 内部故障。
0032	固定分配的 PLC 轴：运行 (JOG) 或回参考点
003C	固定分配的 PLC 轴：通道状态不允许开始

故障标识 (W#16#...)	说明
0064	对轴/主轴编程时的位置错误（对应报警 16830）
0065	编写的转速过高
0066	恒定切削速度的取值范围错误（对应报警 14840）
0068	副主轴:不允许的编程（对应报警 22030）
0069	无测量系统（对应报警 16770）
006A	轴定位尚在生效（对应报警 22052）
006B	未找到参考标记（对应报警 22051）
006C	从转速闭环控制到位置闭环控制没有过渡（对应报警 22050）
006D	未找到参考标记（对应报警 22051）
006E	速度/转速为负
006F	设定转速 == 零
0070	无效的齿轮档
0073	未到达编程的位置
0075	在 NC 中 G96/G961 未生效
0076	在 NC 中 G96/G961 仍有效
0078	该轴不是分度轴（对应报警 20072）
0079	分度位置错误（对应于报警 17510）
007D	DC（最短行程）无法启用（对应报警 16800）
007E	无法启用负绝对值（对应报警 16820）
007F	无法启用正绝对值（对应报警 16810）
0080	无用于直径编程的平面轴（对应报警 16510）
0082	软件限位开关正（对应报警 20070）
0083	软件限位开关负（对应报警 20070）
0084	工作区域限制正（对应报警 20071）
0085	工作区域限制负（对应报警 20071）
0086	框架不可用于分度轴
0087	配备“端面齿”的分度轴生效（对应报警 17501）
0088	配备“端面齿”的分度轴生效，且轴未回参考点（对应报警 17503）
0089	不可为经过转换的主轴/轴启动主轴模式（对应报警 22290）
008A	轴：超出特定坐标系中的正向工作区域限制（对应报警 20082）

故障标识 (W#16#...)	说明
008B	轴：超出特定坐标系中的负向工作区域限制（对应报警 20082）
00C8	系统故障：对应于报警 450007

示例：为分度轴进行增量式定位

属于已注册实例"A1"的轴应相对于当前分度位置在负方向上移动 3 个分度位置 (modeIncremental := TRUE)。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈保存在临时变量"tempBusy"、"tempDone"、"tempError"和"tempStatus"中。

到达位置 (tempDone = TRUE) 或出现故障 (tempError = TRUE) 后，会自动通过 #statExecute := FALSE 结束任务。

SCL

```
// SINU_TravIndAxis_Instance : SINU_TraverseIndexingAxis
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
#SINU_TravIndAxis_Instance(execute := #statExecute,
                           modeAbsolute := FALSE,
                           modeShortest := FALSE,
                           modePositiveDir := FALSE,
                           modeNegativeDir := FALSE,
                           modeIncremental := TRUE,
                           indexingPos := -3, // 返回 3 个分度
                           speed := 1000.0, // 1000 mm/min
                           imperialUnits := FALSE,
                           busy => #tempBusy,
                           done => #tempDone,
                           error => #tempError,
                           status => #tempStatus,
                           axis := "A1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

示例：为旋转的分度轴定位

属于已注册实例"A1"的旋转轴（是一个分度轴）应以 12 rpm 的速度定位在正接近方向上的分度位置 6 (modePositiveDir := TRUE)。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈保存在临时变量"tempBusy"、"tempDone"、"tempError"和"tempStatus"中。

到达位置 (tempDone = TRUE) 或出现故障 (tempError = TRUE) 后，会自动通过 #statExecute := FALSE 结束任务。

SCL

```
// SINU_TravIndAxis_Instance : SINU_TraverseIndexingAxis
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
#SINU_TravIndAxis_Instance(execute := #statExecute,
                           modeAbsolute := FALSE,
                           modeShortest := FALSE,
                           modePositiveDir := TRUE,
                           modeNegativeDir := FALSE,
                           modeIncremental := FALSE,
                           indexingPos := 6,     // 分度位置 6°
                           speed := 12.0,        // 12 rpm
                           imperialUnits := FALSE,
                           busy => #tempBusy,
                           done => #tempDone,
                           error => #tempError,
                           status => #tempStatus,
                           axis := "A1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

示例：以英制单位为分度轴定位

属于已注册实例"B1"的分度轴应以 72 inch/min 的速度定位在分度位置 3 处（modeAbsolute := TRUE 且 imperialUnits := TRUE）。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈保存在临时变量"tempBusy"、"tempDone"、"tempError"和"tempStatus"中。

到达位置 (tempDone = TRUE) 或出现故障 (tempError = TRUE) 后，会自动通过 #statExecute := FALSE 结束任务。

SCL

```
// SINU_TravIndAxis_Instance : SINU_TraverseIndexingAxis
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
#SINU_TravIndAxis_Instance(execute := #statExecute,
                           modeAbsolute := TRUE,
                           modeShortest := FALSE,
                           modePositiveDir := FALSE,
                           modeNegativeDir := FALSE,
                           modeIncremental := FALSE,
                           indexingPos := 3,      // 分度位置 3°
                           speed := 72.0,         // 72 inch/min
                           imperialUnits := TRUE,
                           busy => #tempBusy,
                           done => #tempDone,
                           error => #tempError,
                           status => #tempStatus,
                           axis := "B1");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

参见

错误代码和状态代码 (页 1345)

12.4.11.3 SINU_OscillateSpindle

说明

该指令可开始或停止摆动某个主轴。为此，可指定齿轮级。

为了借助指令启动或停止主轴，轴的控制必须首先从 NC 传输到 PLC。参见子章节“PLC 轴和轴交换”。

任务开始后无法更改参数。启动或停止任务一旦开始，就不能通过指令来终止。

可以为同一加工轴编程该指令的多个实例。在这种情况下，依次处理多个同时开始的任务。如果该指令与其他指令同时用于同一加工轴的轴/主轴控制，也同样如此。

可同时控制不同的加工轴。必须为每个加工轴使用单独的指令实例，并且必须对调用进行编程。

不能使用该指令来控制非主轴的轴。

NC：命令

对指令的调用与以下 NC 命令相对应：

- M40 及参数 gearStage := 0，用于自动选择齿轮级
- M41...M45 及参数 gearStage := 1...5，用于齿轮级 1 ... 5

PLC 轴和轴交换

在轴可以由 PLC 控制之前，必须将轴激活为 PLC 轴（从 NC 到 PLC 的轴交换）。

使用以下信号来进行轴交换，并注意相应的信号：

- `<axis>.interchange.out.assignToPlc = TRUE;`
- `<axis>.interchange.out.requireSwitchOver = TRUE;`

控制回转后（从 PLC 到 NC 的轴交换），轴可以再次由 NC 程序控制。

使用以下信号来进行轴交换，并注意相应的信号：

- `<axis>.interchange.out.assignToPlc = FALSE;`
- `<axis>.interchange.out.requireSwitchOver = TRUE;`

在引导启动期间，固定分配的 PLC 轴（MD30460 位 5 = 1，<Axis>.plcCtrl.in.plcCtrlPermanent = TRUE）变为“中立轴”。在运行任务中，定位轴自动成为 PLC 轴，无需提前进行轴交换。

轴禁止

如果设置了轴禁用 <Axis>.basic.out.disable 或 <Axis>.basic.in.disabled，通过该指令控制的轴不会移动，仅生成模拟的实际值。当轴被 NC 禁用时，行为与轴运行时相同。

耦联组中的轴

如果轴处于耦联组中，那么这些轴只能借助指令通过引导轴来移动。无法通过跟随轴来控制耦联组。

只有设置 <Axis>.coupling.out.enableFollowOverlay := TRUE 后，才能控制跟随轴。然后仅跟随轴本身受控。

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
start	Input	Bool	TRUE:用于启动主轴摆动的任务 不得与“stop”一起设置。
stop	Input	Bool	TRUE:用于停止主轴摆动的任务 不得与“start”一起设置。
gearStage	Input	UInt	0：自动选择齿轮级 1...5：齿轮级
axis	InOut	NCK_typeAxis	要控制的 NCK 加工轴的已注册实例。
busy	Output	Bool	TRUE:执行用于控制轴的任务
done	Output	Bool	TRUE:用于控制轴的任务顺利完成，可以重置“start”或“stop”。
error	Output	Bool	TRUE:出错，可以重置“start”或“stop”。
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	实例未注册。	在控制轴之前，请先注册轴实例。
800D	带有“start”和“stop”的任务组合无效	不要同时使用“start”和“stop”。

状态/错误代码* (W#16#...)	说明	解决办法
8012	轴不是 PLC 轴。	在控制轴之前，请先进行轴交换。 参见子章节"PLC 轴和轴交换"
0001-00FF	来自 NC 的故障标识，参见表格"来自 NC 的故障标识"。	必要时参见 NC 报警
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

来自 NC 的故障标识

故障标识 (W#16#...)	说明
0001	已同时激活多个轴/主轴功能
0014	启动了一个功能，但未到达位置
001E	轴/主轴的控制权在运动结束前即已移交给 NC
0028	通过 NC 程序对轴进行编程，NC 内部故障。
0032	固定分配的 PLC 轴：运行 (JOG) 或回参考点
003C	固定分配的 PLC 轴：通道状态不允许开始
0064	对轴/主轴编程时的位置错误（对应报警 16830）
0065	编写的转速过高
0066	恒定切削速度的取值范围错误（对应报警 14840）
0068	副主轴:不允许的编程（对应报警 22030）
0069	无测量系统（对应报警 16770）
006A	轴定位尚在生效（对应报警 22052）
006B	未找到参考标记（对应报警 22051）
006C	从转速闭环控制到位置闭环控制没有过渡（对应报警 22050）
006D	未找到参考标记（对应报警 22051）
006E	速度/转速为负
006F	设定转速 == 零
0070	无效的齿轮档
0073	未到达编程的位置
0075	在 NC 中 G96/G961 未生效。

故障标识 (W#16#...)	说明
0076	在 NC 中 G96/G961 仍有效。
0078	该轴不是分度轴（对应报警 20072）
0079	分度位置错误（对应于报警 17510）
007D	DC（最短行程）无法启用（对应报警 16800）
007E	无法启用负绝对值（对应报警 16820）
007F	无法启用正绝对值（对应报警 16810）
0080	无用于直径编程的平面轴（对应报警 16510）
0082	软件限位开关正（对应报警 20070）
0083	软件限位开关负（对应报警 20070）
0084	工作区域限制正（对应报警 20071）
0085	工作区域限制负（对应报警 20071）
0086	框架不可用于分度轴
0087	配备“端面齿”的分度轴生效（对应报警 17501）
0088	配备“端面齿”的分度轴生效，且轴未回参考点（对应报警 17503）
0089	不可为经过转换的主轴/轴启动主轴模式（对应报警 22290）
008A	轴：超出特定坐标系中的正向工作区域限制（对应报警 20082）
008B	轴：超出特定坐标系中的负向工作区域限制（对应报警 20082）
00C8	系统故障：对应于报警 450007

示例：启动主轴摆动

属于已注册实例“SP”的主轴应摆动。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈保存在临时变量“tempBusy”、“tempDone”、“tempError”和“tempStatus”中。

开始旋转 (tempDone = TRUE) 或出现故障 (tempError = TRUE) 后，会自动通过 #statExecute := FALSE 结束任务。如果任务顺利完成，主轴继续摆动。

SCL

```
// SINU_OscillateSpindle_Instance : SINU_OscillateSpindle
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#SINU_OscillateSpindle_Instance(start := #statExecute,
                                stop := FALSE,
                                gearStage := 3,   // 齿轮箱挡位 3
                                busy => #tempBusy,
                                done => #tempDone,
                                error => #tempError,
                                status => #tempStatus,
                                axis := "SP");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

示例：停止主轴摆动

属于已注册实例“SP”的主轴应停止。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈保存在临时变量“tempBusy”、“tempDone”、“tempError”和“tempStatus”中。

到达位置 (tempDone = TRUE) 或出现故障 (tempError = TRUE) 后，会自动通过 #statExecute := FALSE 结束任务。

SCL

```
// SINU_OscillateSpindle_Instance : SINU_OscillateSpindle
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
#SINU_OscillateSpindle_Instance(start := FALSE,
                                stop := #statExecute,
                                gearStage  := 0,
                                busy => #tempBusy,
                                done => #tempDone,
                                error => #tempError,
                                status => #tempStatus,
                                axis := "SP");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

参见

错误代码和状态代码 (页 1345)

12.4.11.4 SINU_PositionSpindle

说明

该指令可控制主轴的位置。除了目标位置或要移动的距离外，还可以指定接近方向和速度。

为了借助该指令移动主轴，轴的控制必须首先从 NC 传输到 PLC。参见子章节"PLC 轴和轴交换"。

任务开始后无法更改参数。一旦开始执行定位任务，就无法通过指令取消。要停止主轴运行，请改用 "删除剩余行程 (页 1251)"功能（参见 <Axis>.basic.out.resetMovement）。

可以为同一加工轴编程该指令的多个实例。在这种情况下，依次处理多个同时开始的任务。如果该指令与其他指令同时用于同一加工轴的轴/主轴控制，也同样如此。

可同时控制不同的加工轴。必须为每个加工轴使用单独的指令实例，并且必须对调用进行编程。

不能使用该指令来控制非主轴的轴。

参数"modeIncremental"、"modeShortest"、"modeNegativeDir"、"modePositiveDir"之一必须设置为 TRUE。但是，不能将这些参数中的超过一个设置为 TRUE。

NC：命令

对指令的调用与以下 NC 命令相对应：

- SPOS[spindle] = IC(angularPos) 带参数 modeIncremental := TRUE
- SPOS[spindle] = DC(angularPos) 带参数 modeShortest := TRUE
- SPOS[spindle] = ACN(angularPos) 带参数 modeNegativeDir := TRUE
- SPOS[spindle] = ACP(angularPos) 带参数 modePositiveDir := TRUE

PLC 轴和轴交换

在轴可以由 PLC 控制之前，必须将轴激活为 PLC 轴（从 NC 到 PLC 的轴交换）。

使用以下信号来进行轴交换，并注意相应的信号：

- `<axis>.interchange.out.assignToPlc = TRUE;`
- `<axis>.interchange.out.requireSwitchOver = TRUE;`

控制回转后（从 PLC 到 NC 的轴交换），轴可以再次由 NC 程序控制。

使用以下信号来进行轴交换，并注意相应的信号：

- `<axis>.interchange.out.assignToPlc = FALSE;`
- `<axis>.interchange.out.requireSwitchOver = TRUE;`

在引导启动期间，固定分配的 PLC 轴（MD30460 位 5 = 1，<Axis>.plcCtrl.in.plcCtrlPermanent = TRUE）变为“中立轴”。在运行任务中，定位轴自动成为 PLC 轴，无需提前进行轴交换。

轴禁止

如果设置了轴禁用 <Axis>.basic.out.disable 或 <Axis>.basic.in.disabled，通过该指令控制的轴不会移动，仅生成模拟的实际值。当轴被 NC 禁用时，行为与轴运行时相同。

耦联组中的轴

如果轴处于耦联组中，那么这些轴只能借助指令通过引导轴来移动。无法通过跟随轴来控制耦联组。

只有设置 <Axis>.coupling.out.enableFollowOverlay := TRUE 后，才能控制跟随轴。然后仅跟随轴本身受控。

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明	
execute	Input	Bool	TRUE:用于控制轴的任务	
modeShortest	Input	Bool	TRUE:以最短路径定位到绝对位置	这些参数之一必须为 TRUE
modePositiveDir	Input	Bool	TRUE:沿正接近方向定位到绝对位置	
modeNegativeDir	Input	Bool	TRUE:沿负接近方向定位到绝对位置	
modeIncremental	Input	Bool	TRUE:增量式定位：“angularPos”是距离，不是绝对位置	
angularPos	Input	Real	绝对位置（度）。 "modeIncremental" := TRUE:待移动的角度（度）。	

参数	声明	数据类型	说明
spindleSpeed	Input	Real	定位速度 (rpm)。 0：使用 MD35300 $MA_SPIND_POSCTRL_VELO 中的速度
axis	InOut	NCK_typeAxis	要控制的 NCK 加工轴的已注册实例
busy	Output	Bool	TRUE:执行用于控制轴的任务
done	Output	Bool	TRUE:用于控制轴的任务顺利完成，可以重置“execute”
error	Output	Bool	TRUE:出错，可以重置“execute”
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	实例未注册。	在控制轴之前，请先注册轴实例。
8012	轴不是 PLC 轴。	在控制轴之前，请先进行轴交换。 参见子章节“PLC 轴和轴交换”
8204	模式不具有唯一性，	请将模式参数之一设置为 TRUE。
0001-00FF	来自 NC 的故障标识，参见表格“来自 NC 的故障标识”。	必要时参见 NC 报警
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

来自 NC 的故障标识

故障标识 (W#16#...)	说明
0001	已同时激活多个轴/主轴功能
0014	启动了一个功能，但未到达位置

故障标识 (W#16#...)	说明
001E	轴/主轴的控制权在运动结束前即已移交给 NC
0028	通过 NC 程序对轴进行编程，NC 内部故障。
0032	固定分配的 PLC 轴：运行 (JOG) 或回参考点
003C	固定分配的 PLC 轴：通道状态不允许开始
0064	对轴/主轴编程时的位置错误（对应报警 16830）
0065	编写的转速过高
0066	恒定切削速度的取值范围错误（对应报警 14840）
0068	副主轴:不允许的编程（对应报警 22030）
0069	无测量系统（对应报警 16770）
006A	轴定位尚在生效（对应报警 22052）
006B	未找到参考标记（对应报警 22051）
006C	从转速闭环控制到位置闭环控制没有过渡（对应报警 22050）
006D	未找到参考标记（对应报警 22051）
006E	速度/转速为负
006F	设定转速 == 零
0070	无效的齿轮档
0073	未到达编程的位置
0075	在 NC 中 G96/G961 未生效。
0076	在 NC 中 G96/G961 仍有效。
0078	该轴不是分度轴（对应报警 20072）
0079	分度位置错误（对应于报警 17510）
007D	DC（最短行程）无法启用（对应报警 16800）
007E	无法启用负绝对值（对应报警 16820）
007F	无法启用正绝对值（对应报警 16810）
0080	无用于直径编程的平面轴（对应报警 16510）
0082	软件限位开关正（对应报警 20070）
0083	软件限位开关负（对应报警 20070）
0084	工作区域限制正（对应报警 20071）
0085	工作区域限制负（对应报警 20071）
0086	框架不可用于分度轴
0087	配备“端面齿”的分度轴生效（对应报警 17501）

故障标识 (W#16#...)	说明
0088	配备“端面齿”的分度轴生效，且轴未回参考点（对应报警 17503）
0089	不可为经过转换的主轴/轴启动主轴模式（对应报警 22290）
008A	轴：超出特定坐标系中的正向工作区域限制（对应报警 20082）
008B	轴：超出特定坐标系中的负向工作区域限制（对应报警 20082）
00C8	系统故障：对应于报警 450007

示例：增量式定位主轴

属于已注册实例“SP”的主轴将相对于当前位置移动 15° (modeIncremental := TRUE)。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈保存在临时变量“tempBusy”、“tempDone”、“tempError”和“tempStatus”中。

到达位置 (tempDone = TRUE) 或出现故障 (tempError = TRUE) 后，会自动通过 #statExecute := FALSE 结束任务。

SCL

```
// SINU_PositionSpindel_Instance : SINU_PositionSpindel
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
#SINU_PositionSpindel_Instance(execute := #statExecute,
                               modeShortest := FALSE,
                               modePositiveDir := FALSE,
                               modeNegativeDir := FALSE,
                               modeIncremental := TRUE,
                               angularPos := 15.0, // 移动 15°
                               spindleSpeed := 50.0, // 50 rpm
                               busy => #tempBusy,
                               done => #tempDone,
                               error => #tempError,
                               status => #tempStatus,
                               axis := "SP");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

示例：主轴绝对定位

属于已注册实例“SP”的主轴将以 120 rpm 的速度定位在正接近方向上 90° 处 (modePositiveDir := TRUE)。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈保存在临时变量“tempBusy”、“tempDone”、“tempError”和“tempStatus”中。

到达位置 (tempDone = TRUE) 或出现故障 (tempError = TRUE) 后，会自动通过 #statExecute := FALSE 结束任务。

SCL

```
// SINU_PositionSpindel_Instance : SINU_PositionSpindel
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
#SINU_PositionSpindel_Instance(execute := #statExecute,
                               modeShortest := FALSE,
                               modePositiveDir := TRUE,
                               modeNegativeDir := FALSE,
                               modeIncremental := FALSE,
                               angularPos := 90.0,    // 在 90°
                               spindleSpeed := 120.0, // 120 rpm
                               busy => #tempBusy,
                               done => #tempDone,
                               error => #tempError,
                               status => #tempStatus,
                               axis := "SP");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

参见

错误代码和状态代码 (页 1345)

12.4.11.5 SINU_RotateSpindle

说明

该指令可启动或停止主轴的旋转。除了旋转方向之外，还可指定速度。

为了借助指令启动或停止主轴，轴的控制必须首先从 NC 传输到 PLC。参见子章节"PLC 轴和轴交换"。

任务开始后无法更改参数。启动或停止任务一旦开始，就不能通过指令来终止。

可以为同一加工轴编程该指令的多个实例。在这种情况下，依次处理多个同时开始的任务。如果该指令与其他指令同时用于同一加工轴的轴/主轴控制，也同样如此。

可同时控制不同的加工轴。必须为每个加工轴使用单独的指令实例，并且必须对调用进行编程。

不能使用该指令来控制非主轴的轴。

NC：命令

对指令的调用与以下 NC 命令相对应：

- S<spindle> M3 带参数 start := TRUE 和 dirM4 := FALSE
- S<spindle> M4 带参数 start := TRUE 和 dirM4 := TRUE
- M5 带参数 stop := TRUE

PLC 轴和轴交换

在轴可以由 PLC 控制之前，必须将轴激活为 PLC 轴（从 NC 到 PLC 的轴交换）。

使用以下信号来进行轴交换，并注意相应的信号：

- `<axis>.interchange.out.assignToPlc = TRUE;`
- `<axis>.interchange.out.requireSwitchOver = TRUE;`

控制回转后（从 PLC 到 NC 的轴交换），轴可以再次由 NC 程序控制。

使用以下信号来进行轴交换，并注意相应的信号：

- `<axis>.interchange.out.assignToPlc = FALSE;`
- `<axis>.interchange.out.requireSwitchOver = TRUE;`

在引导启动期间，固定分配的 PLC 轴（MD30460 位 5 = 1，<Axis>.plcCtrl.in.plcCtrlPermanent = TRUE）变为"中立轴"。在运行任务中，定位轴自动成为 PLC 轴，无需提前进行轴交换。

轴禁止

如果设置了轴禁用 <Axis>.basic.out.disable 或 <Axis>.basic.in.disabled，通过该指令控制的轴不会移动，仅生成模拟的实际值。当轴被 NC 禁用时，行为与轴运行时相同。

耦联组中的轴

如果轴处于耦联组中，那么这些轴只能借助指令通过引导轴来移动。无法通过跟随轴来控制耦联组。

只有设置 <Axis>.coupling.out.enableFollowOverlay := TRUE 后，才能控制跟随轴。然后仅跟随轴本身受控。

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
start	Input	Bool	TRUE:用于启动主轴旋转的任务 不得与“stop”一起设置。
stop	Input	Bool	TRUE:用于停止主轴旋转的任务。 不得与“start”一起设置。
spindleSpeed	Input	Real	主轴转速 (rpm)
dirM4	Input	Bool	FALSE：旋转方向同 M3 TRUE:旋转方向同 M4
autoGearStage	Input	Bool	TRUE:自动选择齿轮级
axis	InOut	NCK_typeAxis	要控制的 NCK 加工轴的已注册实例。
busy	Output	Bool	TRUE:执行用于控制轴的任务
done	Output	Bool	TRUE:用于控制轴的任务顺利完成，可以重置“start”或“stop”。
error	Output	Bool	TRUE:出错，可以重置“start”或“stop”。
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	**说明**	**解决办法**
0000	无错误	-
7001	等待 NCK 的空闲接口。	-
7002	已将任务发送至 NCK，等待应答。	-
7005	已将任务发送至 NCK，等待结束。	-
8005	实例未注册。	在控制轴之前，请先注册轴实例。
800D	带有“start”和“stop”的任务组合无效	不要同时使用“start”和“stop”。
8012	轴不是 PLC 轴。	在控制轴之前，请先进行轴交换。 参见子章节“PLC 轴和轴交换”
0001-00FF	来自 NC 的故障标识，参见表格“来自 NC 的故障标识”。	必要时参见 NC 报警
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

来自 NC 的故障标识

故障标识 (W#16#...)	**说明**
0001	已同时激活多个轴/主轴功能
0014	启动了一个功能，但未到达位置
001E	轴/主轴的控制权在运动结束前即已移交给 NC
0028	通过 NC 程序对轴进行编程，NC 内部故障。
0032	固定分配的 PLC 轴：运行 (JOG) 或回参考点
003C	固定分配的 PLC 轴：通道状态不允许开始
0064	对轴/主轴编程时的位置错误（对应报警 16830）
0065	编写的转速过高
0066	恒定切削速度的取值范围错误（对应报警 14840）
0068	副主轴:不允许的编程（对应报警 22030）
0069	无测量系统（对应报警 16770）
006A	轴定位尚在生效（对应报警 22052）
006B	未找到参考标记（对应报警 22051）

故障标识 (W#16#...)	说明
006C	从转速闭环控制到位置闭环控制没有过渡（对应报警 22050）
006D	未找到参考标记（对应报警 22051）
006E	速度/转速为负
006F	设定转速 == 零
0070	无效的齿轮档
0073	未到达编程的位置
0075	在 NC 中 G96/G961 未生效。
0076	在 NC 中 G96/G961 仍有效。
0078	该轴不是分度轴（对应报警 20072）
0079	分度位置错误（对应于报警 17510）
007D	DC（最短行程）无法启用（对应报警 16800）
007E	无法启用负绝对值（对应报警 16820）
007F	无法启用正绝对值（对应报警 16810）
0080	无用于直径编程的平面轴（对应报警 16510）
0082	软件限位开关正（对应报警 20070）
0083	软件限位开关负（对应报警 20070）
0084	工作区域限制正（对应报警 20071）
0085	工作区域限制负（对应报警 20071）
0086	框架不可用于分度轴
0087	配备“端面齿”的分度轴生效（对应报警 17501）
0088	配备“端面齿”的分度轴生效，且轴未回参考点（对应报警 17503）
0089	不可为经过转换的主轴/轴启动主轴模式（对应报警 22290）
008A	轴：超出特定坐标系中的正向工作区域限制（对应报警 20082）
008B	轴：超出特定坐标系中的负向工作区域限制（对应报警 20082）
00C8	系统故障：对应于报警 450007

示例：启动主轴旋转

属于已注册实例“SP”的主轴应以 1,000 rpm 的转速旋转。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈保存在临时变量“tempBusy”、“tempDone”、“tempError”和“tempStatus”中。

开始旋转 (tempDone = TRUE) 或出现故障 (tempError = TRUE) 后，会自动通过 #statExecute := FALSE 结束任务。如果任务顺利完成，主轴继续以 1,000 rpm 的转速旋转。

SCL

```
// SINU_RotateSpindle_Instance : SINU_RotateSpindle
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
#SINU_RotateSpindle_Instance(start := #statExecute,
                             stop := FALSE,
                             spindleSpeed := 1000.0,   // 1000 rpm
                             dirM4 := FALSE,           // 旋转方向同 M3
                             autoGearStage := FALSE,
                             busy => #tempBusy,
                             done => #tempDone,
                             error => #tempError,
                             status => #tempStatus,
                             axis := "SP");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

示例：停止主轴旋转

属于已注册实例"SP"的主轴应停止。此操作在设置静态变量 #statExecute := TRUE 后执行。

反馈保存在临时变量"tempBusy"、"tempDone"、"tempError"和"tempStatus"中。

到达位置 (tempDone = TRUE) 或出现故障 (tempError = TRUE) 后，会自动通过 #statExecute := FALSE 结束任务。

SCL

```
// SINU_RotateSpindle_Instance : SINU_RotateSpindle
// statExecute : Bool
// tempBusy : Bool
// tempDone : Bool
// tempError : Bool
// tempStatus : Word
```

SCL

```
#SINU_RotateSpindle_Instance(start := FALSE,
                             stop := #statExecute,
                             spindleSpeed := 0.0,
                             dirM4 := FALSE,
                             autoGearStage := FALSE,
                             busy => #tempBusy,
                             done => #tempDone,
                             error => #tempError,
                             status => #tempStatus,
                             axis := "SP");

IF #statExecute THEN
    IF #tempError OR #tempDone THEN
        #statExecute := FALSE;
    END_IF;
END_IF;
```

参见

错误代码和状态代码 (页 1345)

12.4.11.6 SINU_CtrlStarDelta

说明

该指令用于主轴的星角切换。该指令根据输入值“deltaOn”，在创建的两个电机数据组 (<Axis>.drive.out.motorDataSetSelection) 之间切换。

在切换开始时，会检查电机数据集的有效性(<Axis>.drive.in.numberOfDriveDataSetsValid)。所用的主轴必须在 NC 中参数设置为主轴，由 PLC 用户程序 ("SINU_Register") 激活和注册。如果满足了这些条件，接口位 <Axis>.drive.out.motorChangeoverDone 以及两个输出端“star”和“delta”才会在切换过程中设置为 FALSE。

在参数化的切换时间“timeValue”过去之后，设置星形或三角形运行，并且在切换时间再次到期后，再次释放脉冲 (<Axis>.drive.in.pulsesEnabled)。

切换的基本步骤

1. 将 <Axis>.drive.in.pulsesEnabled 设置为 FALSE。
2. 将输出端“star”和“delta”设置为 FALSE。
3. 选择星形或者三角形接触器，具体取决于输入端“deltaOn”。
4. 将 <Axis>.drive.out.motorChangeoverDone 和 <Axis>.drive.in.pulsesEnabled 设置为 TRUE。

准确的信号流参见脉冲图。

说明

如果在切换过程中为"enabled"输入赋值 FALSE，这种情况下仍会运行活动的切换过程，直至完成指令处理。尽管 enabled = FALSE，输出端"star"、"delta"和接口信号仍保持其最后的状态。

如果在切换过程中出错，则处理会立即中断并且过程结束。主轴处于未定义状态。在重新启动切换过程前，消除故障。

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明	
enable	Input	Bool	TRUE:设置输入端，以便处理指令	
deltaOn	Input	Bool	星形/三角形切换 • 星形 = 0 • 三角形 = 1	
timeValue	Input	Time	切换时间（值范围：0, 50 ms, ...）	
			配置	**内部生效**
			0	100 ms
			< 50 ms	50 ms
			≥ 50 ms	项目设计的值 (ms)
busy	Output	Bool	TRUE:切换过程生效	
valid	Output	Bool	TRUE:正在处理指令，未出错	
error	Output	Bool	TRUE:出错	
status	Output	Word	出错时的状态编号或错误编号	
star	Output	Bool	TRUE:对星形接触器的控制	
delta	Output	Bool	TRUE:对三角形接触器的控制	
spindle	InOut	NCK_typeAxis	需要更新的主轴/轴实例	

脉冲图

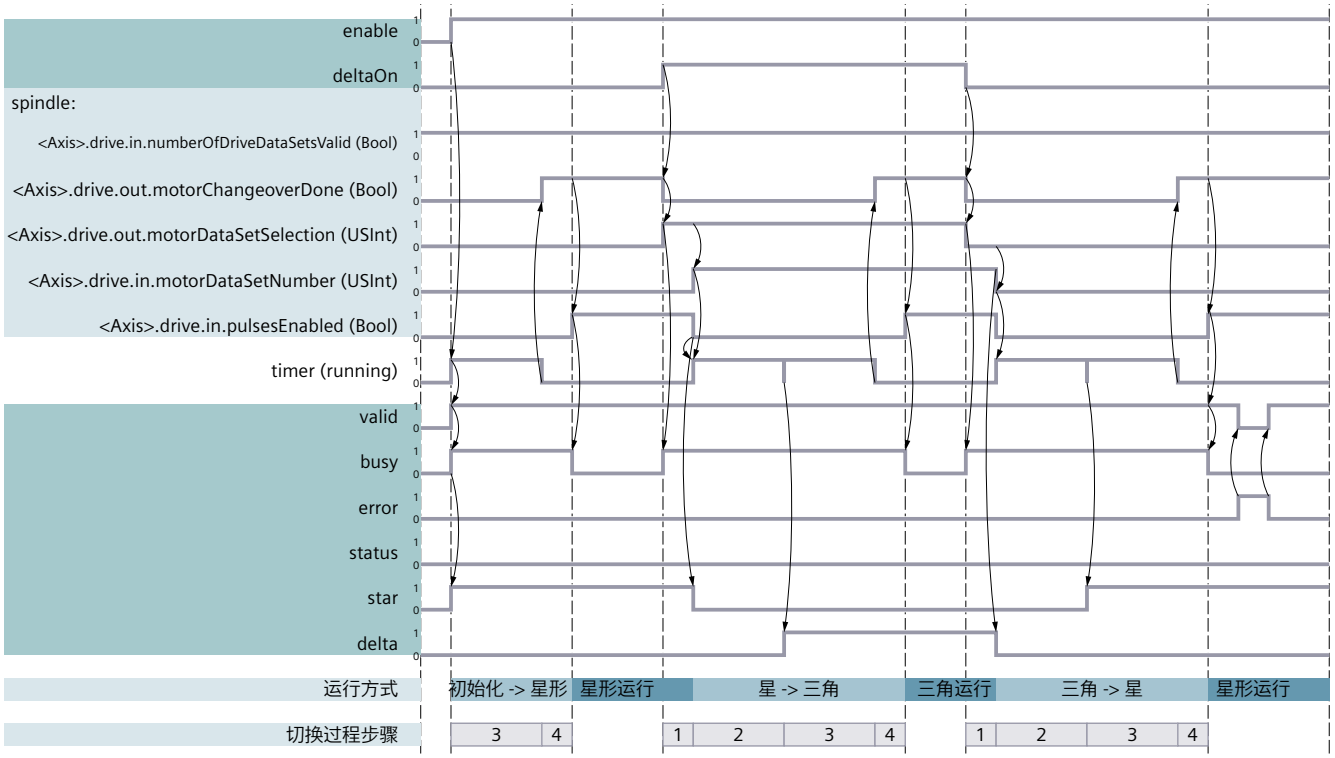


图 12-1　星角切换信号图 SINU_CtrlStarDelta

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7000	未执行功能，因为 enable = FALSE	-
7110	引导启动期间无有效的驱动数据集	等待，直至驱动准备就绪 (<Axis>.drive.in.ready = TRUE)
7111	切换过程步骤 1	-
7112	切换过程步骤 2	-
7113	切换过程步骤 3	-
7114	切换过程步骤 4	-
7121	切换过程步骤 1 最后执行 (enable = FALSE)	-

状态/错误代码* (W#16#...)	说明	解决办法
7122	切换过程步骤 2 最后执行 (enable = FALSE)	-
7123	切换过程步骤 3 最后执行 (enable = FALSE)	-
7124	切换过程步骤 4 最后执行 (enable = FALSE)	-
8001	无效/错误的实例	使用正确的实例，将指令与参数“spindle”关联在一起。
8004	所需的轴未激活。	激活加工轴。
8005	该实例未注册。	注册加工轴。
8110	所传输的轴不是主轴。	在参数“spindle”中将传输的轴定义为主轴 (MD35000 = 1)。
8111	无有效的驱动数据集 (<Axis>.drive.in.numberOfDriveDataSetsValid)	将有效的电机数据集输入驱动参数中。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

说明

如果在切换过程中出现 8001、8005 或 8111 这几个故障之一，那么过程将被中断。在解决所出现的故障之后，切换过程会从同样的位置继续。

示例

所传输的实例“NckInst”.SP 已分配给主轴 SP。在本例中，“DB_StarDelta”是一个全局数据块，其中存储了用于对接输入和输出参数的数据。

SCL

```
// Call SINU_CtrlStarDelta
```

SCL

```
"inst_SINU_CtrlStarDelta"(enable:="DB_StarDelta".enable,
                          deltaOn:="DB_StarDelta".YDelta,
                          timeValue:=T#500ms,
                          busy=>"DB_StarDelta".busy,
                          valid=>"DB_StarDelta".valid,
                          error=>"DB_StarDelta".error,
                          status=>"DB_StarDelta".status,
                          star=>"DB_StarDelta".StarRelay,
                          delta=>"DB_StarDelta".DeltaRelay,
                          spindle:="NckInst".SP);
```

参见

错误代码和状态代码 (页 1345)

12.4.11.7 中断轴的运动，删除剩余行程

中断 PLC 轴的运行

任务开始后无法更改指令参数。因此，不能通过指令取消已启动的用于移动轴的任务。要停止移动的轴，请改用“删除剩余行程”功能（参见 <Axis>.basic.out.resetMovement）。如果已设置信号 <Axis>.basic.out.resetMovement := TRUE，则运行中断，设置“error” := TRUE，并输出“status” := 16#0x1E（见“脉冲图”章节）。

脉冲图

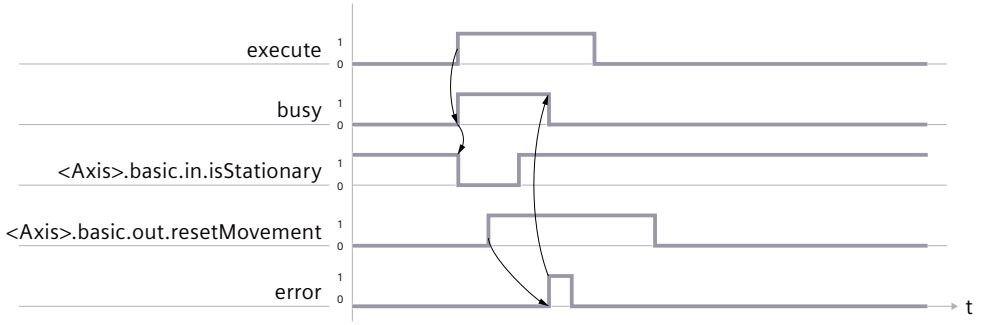


图 12-2 删除信号图剩余行程

参见

SINU_TraverseIndexingAxis (页 1223)

SINU_PositionSpindle (页 1236)

SINU_TraversePosAxis (页 1216)

12.4.12 NC 数据传输

12.4.12.1 SINU_ReadUserDataInt

说明

您可将 PLC 专用机床数据存储在 NC 中，在启动过程中或启动后的某个时间读取这些数据，使用它们实现不同的用户选项、机床扩展级别和机床组态等。

通过此指令可以读取 Int 类型的 PLC 机床数据。

定义传输的 PLC 机床数据

在 NC 侧通过以下一般机床数据定义待传输的 PLC 机床数据。

可能的 PLC 机床数据数量：

- **适用于：CNC 软件 ≥ V6.21 和指令 ≥ V11.0**
  MD14504 $MN_MAXNUM_USER_DATA_INT（INT 类型用户数据的数量），最大值：1024
- **适用于：CNC 软件 < V6.21 和指令 < V11.0**
  MD14504 $MN_MAXNUM_USER_DATA_INT（INT 类型用户数据的数量），最大值：512

待传输 PLC 机床数据的值：

- MD14510[<n>] $MN_USER_DATA_INT（INT 类型的用户数据）

调用时间点

- 必要时在 PLC 引导启动中或 PLC 循环中调用，比如 Startup 组织块或 Main 组织块中

参数

下表介绍函数的参数：

参数	声明	数据类型	说明
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号
maxUserData	Output	UInt	各种类型的可用 PLC 机床数据数量，对应机床数据 MD14504
userData	Output	Variant	用于存储 PLC 机床数据的存储区，是 MD14510[<n>] 中的值。用于存储所有现有 PLC 机床数据的阵列或结构的容量，不能小于可用数据的数量，也不能超过可设置的 PLC 机床数据最大数量。 如果阵列或结构的容量小于相应类型的现有 PLC 机床数据数量，这种情况下只传输部分数据，并输出状态 => 16#7003。 如果可用的数组或结构大于对应类型的现有 PLC 机床数据的数量，则对应数据类型的所有现有数据均被传输至 PLC。阵列或结构中现有的附加元素会被删除。 如果使用预设值 NULL，则各函数只提供 PLC 可用机床数据的数量，并输出状态 => 16#7003。 允许的数据存储区为 E、A、D 和指令接口 (Input、Output、Static、Temp)。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7003	显示参数"userData"的数据存储区太小。只能传输各类型的部分 PLC 可用机床数据。	使用有足够数据量的数据存储区。
8202	参数 userData 指向一个较大的数据存储区。	使用包含不超过 1024 个 Int 或不超过 2048 字节的数据存储区

状态/错误代码* (W#16#...)	说明	解决办法
8203	参数"userData"指向一个无效的数据存储区。	使用数据存储区 E、A、D 或指令接口 (Input、Output、Static、Temp)。
出错后不传输 PLC 机床数据。 * 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

SCL

```
// tempError: Bool
// tempStatus: Word
// tempMaxDataInt: UInt
// tempUserDataInt: Array[1..1024] of Int
SINU_ReadUserDataInt(error => tempError,
                     status => tempStatus,
                     maxUserData => tempMaxDataInt,
                     userData => tempUserDataInt
                     );
```

12.4.12.2 SINU_ReadUserDataHex

说明

您可将 PLC 专用机床数据存储在 NC 中，在启动过程中或启动后的某个时间读取这些数据，使用它们实现不同的用户选项、机床扩展级别和机床组态等。

通过此指令可以读取十六进制的 PLC 机床数据。对应字节型数据。

定义传输的 PLC 机床数据

在 NC 侧通过以下一般机床数据定义待传输的 PLC 机床数据。

可能的 PLC 机床数据数量：

- **适用于：CNC 软件 ≥ V6.21 和指令 ≥ V11.0**
  MD14506 $MN_MAXNUM_USER_DATA_HEX（HEX 类型用户数据的数量），最大值：1024
- **适用于：CNC 软件 < V6.21 和指令 < V11.0**
  MD14506 $MN_MAXNUM_USER_DATA_HEX（HEX 类型用户数据的数量），最大值：512

待传输 PLC 机床数据的值：

- MD14512[<n>] $MN_USER_DATA_HEX（十六进制的用户数据）

调用时间点

- 必要时在 PLC 引导启动中或 PLC 循环中调用，比如 Startup 组织块或 Main 组织块中

参数

下表介绍函数的参数：

参数	声明	数据类型	说明
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号
maxUserData	Output	UInt	各种类型的可用 PLC 机床数据数量，对应机床数据 MD14506（将数量四舍五入为偶数）。
userData	Output	Variant	用于存储 PLC 机床数据的存储区，是 MD14512[<n>] 中的值。用于存储所有现有 PLC 机床数据的阵列或结构的容量，不能小于可用数据的数量，也不能超过可设置的 PLC 机床数据最大数量。 如果阵列或结构的容量小于相应类型的现有 PLC 机床数据数量，这种情况下只传输部分数据，并输出状态 => 16#7003。 如果可用的数组或结构大于对应类型的现有 PLC 机床数据的数量，则对应数据类型的所有现有数据均被传输至 PLC。阵列或结构中现有的附加元素会被删除。 如果使用预设值 NULL，则各函数只提供 PLC 可用机床数据的数量，并输出状态 => 16#7003。 允许的数据存储区为 E、A、D 和指令接口 (Input、Output、Static、Temp)。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7003	显示参数"userData"的数据存储区太小。只能传输各类型的部分 PLC 可用机床数据。	使用有足够数据量的数据存储区。
8202	参数 userData 指向一个较大的数据存储区。	使用包含不超过 1024 字节或不超过 1024 字节的数据存储区
8203	参数"userData"指向一个无效的数据存储区。	使用数据存储区 E、A、D 或指令接口 (Input、Output、Static、Temp)。
出错后不传输 PLC 机床数据。 * 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

SCL

```
// tempError: Bool
// tempStatus: Word
// tempMaxDataHex: UInt
// tempUserDataHex: Array[1..1024] of Byte
SINU_ReadUserDataHex(error => tempError,
                     status => tempStatus,
                     maxUserData => tempMaxDataHex,
                     userData => tempUserDataHex
                     );
```

12.4.12.3 SINU_ReadUserDataFloat

说明

您可将 PLC 专用机床数据存储在 NC 中，在启动过程中或启动后的某个时间读取这些数据，使用它们实现不同的用户选项、机床扩展级别和机床组态等。

通过此指令可以读取 Real 类型的 PLC 机床数据。

定义传输的 PLC 机床数据

在 NC 侧通过以下一般机床数据定义待传输的 PLC 机床数据。

可能的 PLC 机床数据数量：

- **适用于：CNC 软件 ≥ V6.21 和指令 ≥ V11.0**
  MD14508 $MN_MAXNUM_USER_DATA_FLOAT（FLOAT 类型用户数据的数量），最大值：1024
- **适用于：CNC 软件 < V6.21 和指令 < V11.0**
  MD14508 $MN_MAXNUM_USER_DATA_FLOAT（FLOAT 类型用户数据的数量），最大值：128

待传输 PLC 机床数据的值：

- MD14514[<n>] $MN_USER_DATA_FLOAT（FLOAT 类型的用户数据）

调用时间点

- 必要时在 PLC 引导启动中或 PLC 循环中调用，比如 Startup 组织块或 Main 组织块中

参数

下表介绍函数的参数：

参数	声明	数据类型	说明
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

参数	声明	数据类型	说明
maxUserData	Output	UInt	各种类型的可用 PLC 机床数据数量，对应机床数据 MD14508。
userData	Output	Variant	用于存储 PLC 机床数据的存储区，是 MD14514[<n>] 中的值。用于存储所有现有 PLC 机床数据的阵列或结构的容量，不能小于可用数据的数量，也不能超过可设置的 PLC 机床数据最大数量。 如果阵列或结构的容量小于相应类型的现有 PLC 机床数据数量，这种情况下只传输部分数据，并输出状态 => 16#7003。 如果可用的数组或结构大于对应类型的现有 PLC 机床数据的数量，则对应数据类型的所有现有数据均被传输至 PLC。阵列或结构中现有的附加元素会被删除。 如果使用预设值 NULL，则各函数只提供 PLC 可用机床数据的数量，并输出状态 => 16#7003。 允许的数据存储区为 E、A、D 和指令接口 (Input、Output、Static、Temp)。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
7003	显示参数“userData”的数据存储区太小。只能传输各类型的部分 PLC 可用机床数据。	使用有足够数据量的数据存储区。
8202	参数 userData 指向一个较大的数据存储区。	使用包含不超过 1024 个 Float 或不超过 4096 字节的数据存储区
8203	参数 userData 指向一个无效的数据存储区。	使用数据存储区 E、A、D 或指令接口 (Input、Output、Static、Temp)。
出错后不传输 PLC 机床数据。 * 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

SCL

```
// tempError: Bool
// tempStatus: Word
// tempMaxDataFloat: UInt
// tempUserDataFloat: Array[1..1024] of Real
SINU_ReadUserDataFloat(error => tempError,
                       status => tempStatus,
                       maxUserData => tempMaxDataFloat,
                       userData => tempUserDataFloat
                       );
```

12.4.12.4 SINU_ReadCompileCycleData

说明

该指令可读取编译循环数据的指定范围。

供 PLC 读取的编译循环数据可以通过机床数据 MD10410 $MN_CC_VDI_OUT_DATA 进行配置，最大可达 4096 字节。

调用时间点

- 必要时在 PLC 循环中，例如 Main 组织块中

参数

下表介绍函数的参数：

参数	声明	数据类型	说明
sourceOffset	Input	UInt	0..4095：字节偏移量，自其开始读取编译循环数据
count	Input	UInt	自 sourceOffset 起所读取的字节数量
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号
compileCycleData	Output	Variant	用于所读取数据的目标区域。该区域的大小必须与待读取的数据量相对应（参数计数）

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
8011	所指定的范围在编译循环数据之外。	选择“sourceOffset”和“count”，使所生成的范围完全在编译循环数据范围内。
8202	“compileCycleData”所指定的数据存储区过大。	指定一个数据存储区，其大小等于“count”的大小。
8203	“compileCycleData”所指定的数据存储区无效。	指定一个数据存储区，其范围应在 E、A、D 或指令接口 (Input、Output、Static、Temp) 之内。
820D	“compileCycleData”所指定的数据存储区过小。	指定一个数据存储区，其大小等于“count”的大小。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

将用于供 PLC 读取编译循环数据的区域配置为 4096 字节。编译循环数据的最后 3072 字节将被读入数组“tempData”中。

SCL

```
// tempError: Bool
// tempStatus: Word
// tempData: Array[1..3072] of Byte
SINU_ReadCompileCycleData(sourceOffset := 1024,
                          count := 3072,
                          error => tempError,
                          status => tempStatus,
                          compileCycleData => tempData
                          );
```

更多信息

SINU_WriteCompileCycleData (页 1261)

12.4.12.5 SINU_WriteCompileCycleData

说明

该指令可将数值写入编译循环数据的指定范围内。

供 PLC 写入的编译循环数据可以通过机床数据 MD10400 $MN_CC_VDI_IN_DATA 进行配置，大小可达 4096 字节。

调用时间点

- 必要时在 PLC 循环中，例如 Main 组织块中

参数

下表介绍函数的参数：

参数	声明	数据类型	说明
destinationOffset	Input	UInt	0..4095：字节偏移量，自其开始写入编译循环数据
count	Input	UInt	自 destinationOffset 起所写入的字节数量
compileCycleData	Input	Variant	用于待写入数据的源区域。该区域的大小必须与待写入的数据量相对应（参数计数）
error	Output	Bool	TRUE:出错
status	Output	Word	出错时的状态编号或错误编号

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
8011	所指定的范围在编译循环数据之外。	选择“destinationOffset”和“count”，使所生成的范围完全在编译循环数据范围内。
8202	“compileCycleData”所指定的数据存储区过大。	指定一个数据存储区，其大小等于“count”的大小。

状态/错误代码* (W#16#...)	说明	解决办法
8203	“compileCycleData”所指定的数据存储区无效。	指定一个数据存储区，其范围应在 E、A、D 或指令接口 (Input、Output、Static、Temp) 之内。
820D	“compileCycleData”所指定的数据存储区过小。	指定一个数据存储区，其大小等于“count”的大小。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

将用于供 PLC 写入编译循环数据的区域配置为 4096 字节。将数组“tempArray”中的数据写入到编译循环数据的前 2048 个字节中。

SCL

```
// tempError: Bool
// tempStatus: Word
// tempData: Array[1..2048] of Byte
//将数据填入 tempData 中
SINU_WriteCompileCycleData(destinationOffset := 0,
                        count := 2048,
                        compileCycleData => tempData,
                        error => tempError,
                        status => tempStatus
                        );
```

更多信息

SINU_ReadCompileCycleData (页 1259)

12.4.13 操作单元

12.4.13.1 SINU_WriteRecordOverEthernet

说明

该指令将数据记录 (Record) 传输到机床控制面板（通过以太网与 SINUMERIK ONE 相连）中。

说明
此功能在虚拟 SINUMERIK 中不可用！

更多信息
对于通过 PROFINET IO 连接的机床控制面板，可在 PLC 上使用指令“WRREC”（写数据记录）。

关于“WRREC”的更多信息：关键词“WRREC”中的 TIA Portal 在线技术支持。

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
execute	Input	Bool	TRUE:传输数据记录，参见脉冲图
address	Input	USInt	以太网机床控制面板的地址 缺省值：192
udpPort	Input	UInt	PLC 上本地的 UDP 端口 不得同时用于其他指令。 缺省值：16001
connectionId	Input	CONN_OUC	PLC 上的连接 ID 不得同时用于其他指令。 缺省值：1001
interfaceId	Input	HW_ANY	用于连接机床控制面板的网络接口的接口 ID，也可参见硬件配置。 缺省值：262

参数	声明	数据类型	说明
slot	Input	UInt	所发送数据记录 RECORD 所涉及的 SINUMERIK ONE MCP 按键模块的插槽。 缺省值：1
record	Input	UInt	待写入数据记录的数据记录号。128，机床控制面板的组态数据 缺省值：128
busy	Output	Bool	TRUE:写入过程正在进行或尚未完成。
done	Output	Bool	TRUE:数据记录已成功传输/保存。
error	Output	Bool	TRUE:写入过程中出现错误。 “status”参数下显示导致错误的原因。
status	Output	Word	指令状态或错误信息。 关于“status”参数的解释，请参见状态和错误标识表格。
diagnosis	Output	Bool	针对附加信息（状态相关）的诊断输出端
data	InOut	Variant	应发送的数据记录。

参数“slot”的详细说明

通过在数量可变的情况下将按键模块以固定的顺序布置，产生 MCP 模块或输入/输出映像的布置。

模块的固定顺序：

1. 基本（始终存在，槽 1）
2. 按键模块（1 至 4，槽 2 起）
3. Powerride 倍率开关（1 至 3）
4. 2 个手轮（始终存在）

基本模块	**按键模块 1**	**按键模块 2**	**按键模块 3**	**按键模块 4**	Powerride 1	Powerride 2	Powerride 3	手轮 1	手轮 2
插槽 1	**插槽 2**	**插槽 3**	**插槽 4**	**插槽 5**					

针对参数 "data" 的数据记录 128 (16#80) 的结构

说明

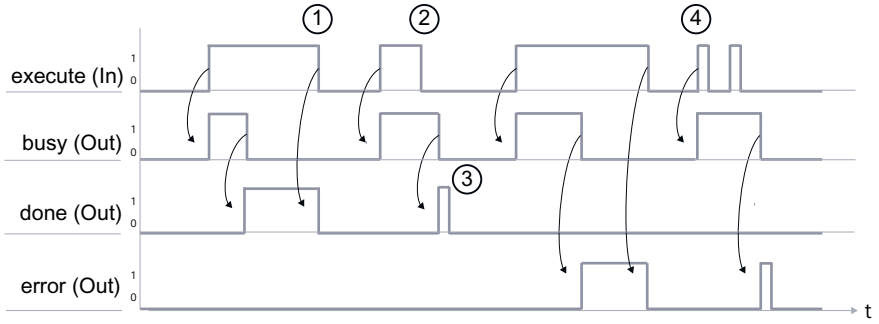
针对数据记录 RECORD 的结构，使用 PLC 基本程序库中提供的数据类型“Modular MCP data types”。

- SINUMERIK ONE：数据类型“LBP_typeModularMcpKeypadRecord”

在一个数据记录中最多可以调整 21 个 LED 按键的数值。如果想在 PLC 用户程序中更改 LED 按键的颜色值，需针对每个按键模块创建一个用于数据记录 128 的实例。

更多有关数据记录 128 结构的信息，请参阅 SINUMERIK ONE MCP 产品手册中的“适用于整个模块的参数数据记录 128 的结构”一章。

脉冲图



任务在“execute”参数为上升沿时开始。应用输入参数的值，并在整个任务持续期间保持有效状态。

应持续调用该指令，直到它重新报告“done”或“error”。

无法中断任务。

在任务结束后，“execute”须为上升沿，借此可启动新任务。

可以通过“SINU_WriteRecordOverEthernet”的不同实例启动多个并行调用。该指令会在内部逐个处理这些调用，而不是并行处理。

① “done”和“error”将随着“execute”的下降沿进行重置。

② 功能块的功能不会随着“execute”的下降沿而停止。如果在设置输出端“done”之前重置了输入端“execute”，则输出端“done”只设置一个循环。

③ 如果“execute”已经是 FALSE 状态，则“done”和“error”只存在于一个循环内。

④ 当指令仍在处理中时 ("busy" = TRUE)，“execute”的上升沿将请求开始一个新任务。旧任务以任务开始时存在的参数结束。

图 12-3 脉冲图 SINU_WriteRecordOverEthernet

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	---
7000	等待任务开始	---
7132	将记录发送到机床控制面板上	---
7133	等待机床控制面板的应答	---
8017	写入过程正在进行或尚未结束	如果“busy”为 FALSE 状态，则“execute”已设置
8116	机床控制面板未在指定时间内应答	检查与机床控制面板的连接
8119	连接时出错：已创建使用该 ID 的连接	使用其它连接 ID，参数“connectionId”
811A	连接时出错：端口已使用或被系统预留	检查参数“udpPort”
8202	参数“data”中的记录数据过长	检查数据表“data”的长度
8203	参数“data”中不允许的存储区	检查参数“data”
8234	连接时出错：参数“interfaceId”中的接口 ID 错误	检查参数“interfaceId”
8235	连接时出错：参数“connectionId”中的连接 ID 错误	检查参数“connectionId”
8236	参数“data”中的记录版本无效（主要信息）	检查参数“data”中的记录版本
8237	参数“record”中的记录编号无效	在参数“record”中使用正确的记录编号
8238	针对 "record" :=128：针对按键的无效配置，无效的按键索引输出到“diagnosis”输出端中。	参数“data”中的按键参数化正确
8239	无效配置，检查参数	检查参数“slot”、“record”和“data”
8240	参数“data”为零	检查参数“data”
8242	发送时出错：机床控制面板地址错误	检查参数“address”
86xx	内部错误	请联系西门子。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例：设置 SINUMERIK ONE MCP 的按键颜色

在以下示例中，与默认颜色分配方案相比，3x5 按键模块的 LED 按键的颜色将改为激活和禁用两种状态。

按键模块是 SINUMERIK ONE MCP1900 的组成部分，已作为机床控制面板 1（通过“SINU_UpdateEthernetMcp”调用）进行了配置。按键模块 2 的按键 4 和 12 是相匹配的，客户键盘（见下图）。

图 12-4　按键模块 2，插槽 3

以下数值应适用于这两个按键：

按键号	用于激活状态的颜色代码	用于禁用状态的颜色代码
4	(0.0.255) 蓝色	(255.255.0) 黄色
12	(0.0.255) 蓝色	(255.255.0) 黄色

PLC 用户项目中的准备工作

1. 在全局数据块“myMCP1900keypads”中针对 SINUMERIK ONE MCP 所需按键模块 2 的数据记录创建一个实例。

myMCP1900Keypads

	Name	Data type	Default value	Comment
1	Static			
2	keypadModeGroup	"LBP_typeModularMcpKeypadRecord"		Keypad 1 for ModeGroup Slot 2
3	keypadOEM	"LBP_typeModularMcpKeypadRecord"		Keypad 2 for OEM keys Slot 3
4	keypadAxis	"LBP_typeModularMcpKeypadRecord"		Keypad 3 for axis keys Slot 4

图 12-5　数据块，带符号名称“myMCP1900Keypads”

2. 在数据块“dataMcp”中创建以下控制变量和状态变量。

19	Mcp1WriteRecord	Struct		
20	execute	Bool	FALSE	
21	slot	UInt	3	
22	record	UInt	128	
23	data	"LBP_typeModularMcpKeypadRecord"		Abbild der modularen MCP für Tasteneigenschaften
24	recordVersion	Byte	16#10	Version 1.0
25	keyActiveAttrib...	USInt	2	Anzahl gültige Einträge in keyActiveAttributes
26	keyInactiveAttr...	USInt	2	Anzahl gültige Einträge in keyInactiveAttributes
27	reserved	Byte	16#0	Nicht verwendete Variable
28	keyActiveAttrib...	Array[0..20] of "LBP_typeModularMc...		Farbattribute für Tasten im Status "active"
29	keyInactiveAttr...	Array[0..20] of "LBP_typeModularMc...		Farbattribute für Tasten im Status "inactive"
30	busy	Bool	FALSE	
31	done	Bool	FALSE	
32	error	Bool	FALSE	
33	status	Word	16#0000	
34	diagnosis	Word	16#0000	
35	reset	Bool	FALSE	

图 12-6　数据块“dataMcp”中的控制变量和状态变量

3. 在已创建的数据块 ("myMCP1900Keypads".keypadOEM) 中，如下所示调整按键模块 2（插槽 3）按键 4 和 12 的数值。可以在应用程序中或直接在数据块中分配这些数值。

名称	数据类型	待设置的值	注释
主要信息			
keypadOEM.recordVersion	字节	B#16#10	版本 1.0
keypadOEM.keyActiveAttribCount	USInt	2	keyActiveAttributes 中有效条目的数量
keypadOEM.keyInactiveAttribCount	USInt	2	keyInactiveAttributes 中有效条目的数量
参数块 1 适用于最多 21 个 LED 的激活状态			
keypadOEM.keyActiveAttributes[0].keyIndex	USInt	4	密钥索引
keypadOEM.keyActiveAttributes[0].red	USInt	0	红色 LED 的亮度
keypadOEM.keyActiveAttributes[0].green	USInt	0	绿色 LED 的亮度
keypadOEM.keyActiveAttributes[0].blue	USInt	255	蓝色 LED 的亮度
keypadOEM.keyActiveAttributes[1].keyIndex	USInt	12	密钥索引
keypadOEM.keyActiveAttributes[1].red	USInt	0	红色 LED 的亮度
keypadOEM.keyActiveAttributes[1].green	USInt	0	绿色 LED 的亮度
keypadOEM.keyActiveAttributes[1].blue	USInt	255	蓝色 LED 的亮度
参数块 2 适用于最多 21 个 LED 的禁用状态			
keypadOEM.keyInactiveAttributes[0].keyIndex	USInt	4	密钥索引
keypadOEM.keyInactiveAttributes[0].red	USInt	255	红色 LED 的亮度
keypadOEM.keyInactiveAttributes[0].green	USInt	255	绿色 LED 的亮度

名称	数据类型	待设置的值	注释
keypadOEM.keyInactiveAttributes[0].blue	USInt	0	蓝色 LED 的亮度
keypadOEM.keyInactiveAttributes[1].keyIndex	USInt	12	密钥索引
keypadOEM.keyInactiveAttributes[1].red	USInt	255	红色 LED 的亮度
keypadOEM.keyInactiveAttributes[1].green	USInt	255	绿色 LED 的亮度
keypadOEM.keyInactiveAttributes[1].blue	USInt	0	蓝色 LED 的亮度
...			

说明

实例数组字段中的索引与按键编号不一致。在“keyIndex”中规定按键编号。

示例

在第一个存储位置（索引为 0），应配置按键 4。

KeypadOEM.keyActiveAttributes[0].keyIndex := 4

4. 在第 3 项中介绍了参数“slot”和“record”。在示例中连接参数“address”、“udpPort”、“connectionId”和“interfaceId”均为默认值，必要时必须进行调整。在数据块“dataMcp”中设置位“dataMcp".Mcp1WriteRecord.execute”，然后开始写入过程。通过“busy”，可以看出该进程是否仍在进行。写入过程成功结束后，设置为“done”。出现错误，“error”位为 TRUE，错误状态通过“status”输出。如果两个位“done”或“error”中的一个为 TRUE，则可以重置“execute”（见脉冲图）。

调用 SINU_WriteRecordOverEthernet

SCL

```
// 将数据记录 128 写入 SINUMERIK ONE MCP 的客户按键模块 2（插槽 3）中
```

SCL

```
#InstWriteRecordOverEthernet(execute := "dataMcp".Mcp1WriteRecord.execute,
                                address := 192,
                                udpPort := 16001,
                                connectionId := 1001,
                                interfaceId := 262,
                                slot := "dataMcp".Mcp1WriteRecord.slot, // 插槽 3
                               record := "dataMcp".Mcp1WriteRecord.record, // 记录编号 128
                                usy => "dataMcp".Mcp1WriteRecord.busy,
                                done => "dataMcp".Mcp1WriteRecord.done,
                                error => "dataMcp".Mcp1WriteRecord.error,
                                status => "dataMcp".Mcp1WriteRecord.status,
                                diagnosis=>"dataMcp".Mcp1WriteRecord.diagnosis,
                                data := "dataMcp".Mcp1WriteRecord.data);

IF "dataMcp".Mcp1WriteRecord.done THEN          // 写入任务顺利完成
  "dataMcp".Mcp1WriteRecord.execute := FALSE;   // 重置"execute"
END_IF;
IF "dataMcp".Mcp1WriteRecord.error AND "dataMcp".Mcp1WriteRecord.reset THEN
  // 写入任务结束但报错
  // 通过输出参数"status"和"diagnosis"进行错误诊断
  // 出现"error"时通过复位信号"reset"重置"execute"
  "dataMcp".Mcp1WriteRecord.execute := FALSE;
END_IF;
```

参见

SINU_TransferHandwheelData (页 1279)

12.4.13.2 SINU_UpdateEthernetMcp

说明

说明

此功能在虚拟 SINUMERIK 中不可用！

该指令用于通过工业以太网 (IE) 与机床控制面板 (MCP)、手持操作设备 (HT)、直接键模块等操作部件进行通信。

对于每一个该操作组件，必须循环调用"SINU_UpdateEthernetMcp"指令，并相应使用单独的实例。

连接参数 MCP 地址、连接 ID、接口 ID 和本地 UDP 端口都是输入参数的一部分。此外，可在输入端上指定其是否是直接键模块。连接参数和输入端"directKeys"只有在 enable = FALSE 时才能更改。可以周期性地更改输入端"disableDataToMcp"（不再向操作组件发送数据）。

如果通过"enable"建立连接，则在第一步中识别操作组件。如果输出参数"identified"为 TRUE，则可以读取机床控制面板型号和所需输入/输出数据的长度。在"设备型号"一章中可以找到已知型号和支持型号的清单。如果识别成功，就可以建立起循环式的连接，输出端设为 mcpReady = TRUE。

在循环式数据交换中，将参数"toMcp"的数据发送到操作组件中，从操作组件接收的数据将被写入参数"fromMcp"。数据以设定的时间间隔进行发送和接收。当有新的数据包可用时，均将设置输出端"newData"。

模块化机床控制面板或其他 HMI 设备提供的数据类型可用于在"fromMcp"和"toMcp"参数中构建输入和输出映像。自有类型或变量也可以进行参数化。

如果 HMI 设备发生故障或与 HMI 设备的连接中断，故障信息将通过"error"和"status"输出。此外，连接中断时，删除输入映像，将输出端"newData"设为 TRUE 一次，以便能够对故障作出响应。如果保持 enable = TRUE 状态，将自动重新尝试建立连接。

为断开连接，必须设置为 enable = FALSE。这样就断开了已激活的以太网连接，无法再处理指令。输入映像会保留最后的状态，而不会像出错时那样被删除。

更多信息

有关工程组态和使用的详细信息请参见章节"以太网机床控制面板和 HT 10 的通讯组态 (页 68)"。

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
enable	Input	Bool	TRUE:创建连接和周期性通信
directKeys	Input	Bool	TRUE:连接到直接键，而不连接到机床控制面板
address	Input	USInt	以太网机床控制面板的地址
udpPort	Input	UInt	PLC 上本地的 UDP 端口 不得同时用于其他指令。针对每个连接使用一个单独的 UDP 端口。

参数	声明	数据类型	说明
connectionId	Input	CONN_OUC	PLC 上的连接 ID 不得同时用于其他指令。针对每个连接使用一个单独的连接 ID。
interfaceId	Input	HW_ANY	用于连接机床控制面板的网络接口的接口 ID，可参见硬件配置。
disableDataToMcp	Input	Bool	TRUE:未将数据发送至机床控制面板
mcpReady	Output	Bool	TRUE:正在进行与操作组件的循环通信
newData	Output	Bool	TRUE:新的数据包可用
identified	Output	Bool	TRUE:已成功识别
mcpType	Output	Byte	机床控制面板的类型标识 （参见章节“设备类型”）
lengthIn	Output	USInt	机床控制面板数据的长度
lengthOut	Output	USInt	机床控制面板上数据的长度
handwheel1	Output	SINU_typeHandwheelData	手轮 1 的数据：active，recent，stopped，pulses
handwheel2	Output	SINU_typeHandwheelData	手轮 2 的数据：active，recent，stopped，pulses
busy	Output	Bool	TRUE:与操作组件的连接已经建立
warning	Output	Bool	TRUE:接收数据的时间长于预期
error	Output	Bool	TRUE:循环通信或建立连接期间出错。 “status”参数下显示导致错误的原因。
status	Output	Word	指令状态或错误信息。 关于参数“status”的解释，参见“错误、状态和补救措施”一章。
fromMcp	InOut	Variant	机床控制面板的数据
toMcp	InOut	Variant	机床控制面板上的数据

脉冲示意图

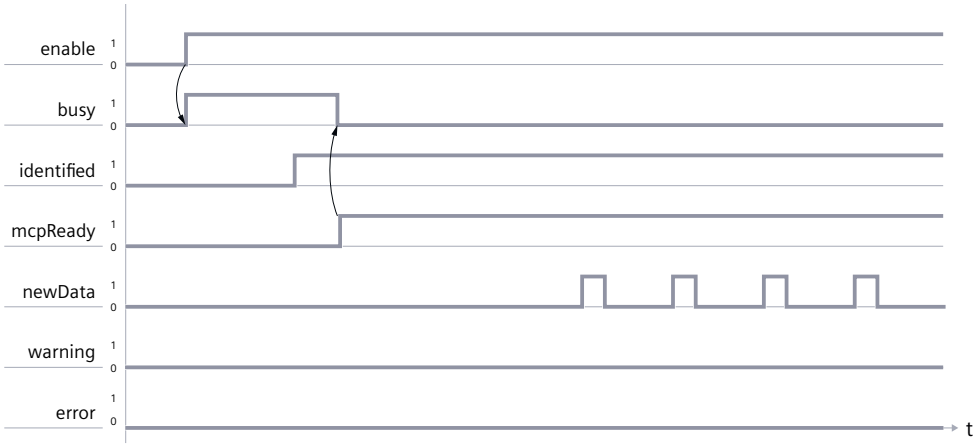


图 12-7　SINU_UpdateEthernetMcp：在一个模块化的机床控制面板上建立循环连接

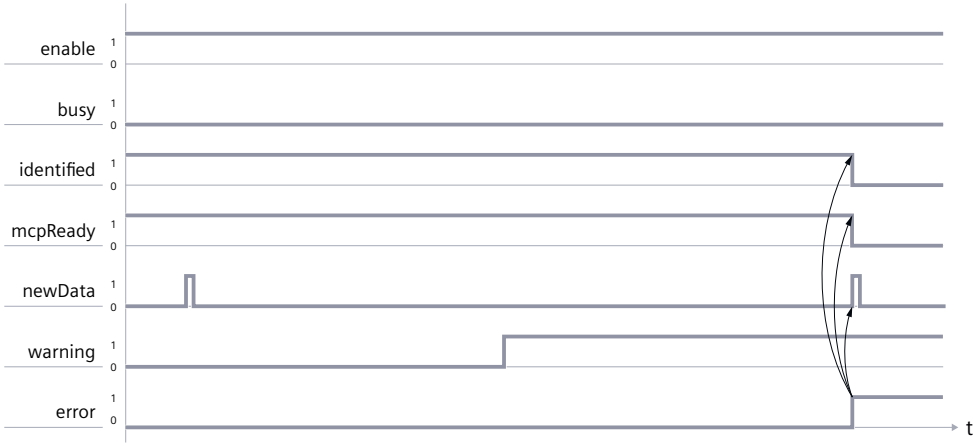


图 12-8　SINU_UpdateEthernetMcp：警告和超时错误

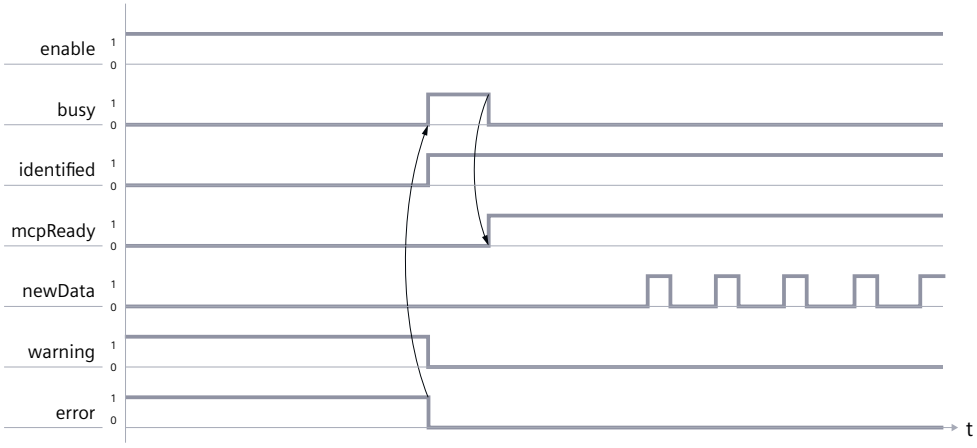


图 12-9　SINU_UpdateEthernetMcp：超时之后建立连接

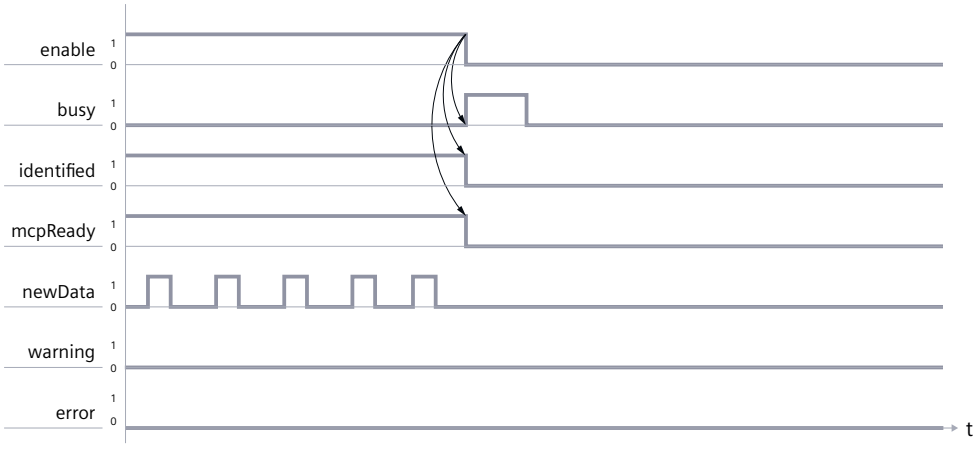


图 12-10　SINU_UpdateEthernetMcp：enable = "False"，连接断开

设备类型

在 SINUMERIK ONE (NC 63) (https://mall.industry.siemens.com/mall/en/WW/Catalog/Products/10028455) 目录中，可以找到 SINUMERIK ONE 与 PLC Basic Program plus 都支持的操作组件。

除了目录中列出的操作组件外，还可以通过指令"SINU_UpdateEthernetMcp"识别出更多的操作组件。下面是已知操作组件的列表，其中有相应的类型标识（对应于输出端"mcpType"）。

类型标识	名称
16#80	MCP 483C
16#81	MCP 483

类型标识	名称
16#82	MCP 310
16#83	MCP OEM
16#84	MCP DMG
16#85	HT 8
16#86	TCU DT
16#87	MCP MPP
16#88	HT 2
16#89	OP 08T
16#8A	MCP 310C PN
16#8B	MCP Index OEM
16#8C	RCU/TCU30
16#8D	MCP Interface PN
16#8E	VDCK
16#90	MCP 398C
16#91	HT 10
16#A0	Modular MCP

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	---
7000	等待任务开始	---
7131	与机床控制面板的连接已创建	---
7132	数据发送到机床控制面板中	---
7133	从机床控制面板接收数据	---
7134	与机床控制面板的连接已断开	---
7135	参数“fromMcp”的范围过小。数据丢失。	为参数“fromMcp”选择范围，至少应与“lengthIn”一样大。
8202	参数“fromMcp”或“toMcp”的范围过长	检查参数“fromMcp”和“toMcp”的长度变量最长可为 224 字节。

状态/错误代码* (W#16#...)	说明	解决办法
8203	参数“fromMcp”或“toMcp”中不允许的存储区	检查参数“fromMcp”和“toMcp”。使用数据存储区 E、A、D 或指令块接口 (Input、Output、Static、Temp)。
8116	进行识别时机床控制面板未在指定时间内应答	检查指定的机床控制面板地址和与机床控制面板的连接
8117	进行参数化时机床控制面板未在指定时间内应答	检查与机床控制面板的连接
8118	进行循环通信时机床控制面板未在指定时间内应答	检查与机床控制面板的连接
8119	连接时出错：已创建使用该 ID 的连接	使用未使用的连接 ID，检查参数“connectionId”
811A	连接时出错：端口已使用或被系统预留	使用未使用的 UDP 端口，检查参数“udpPort”
8234	连接时出错：参数“interfaceId”中的接口 ID 错误	检查参数“interfaceId”
8235	连接时出错：参数“connectionId”中的连接 ID 错误	检查参数“connectionId”
8240	参数“toMcp”为零	检查参数“toMcp”
8241	参数“fromMcp”为零	检查参数“fromMcp”
8242	发送时出错：机床控制面板地址错误	参数“address”，检查与机床控制面板的连接
8243	操作组件无 IPv4 地址	检查是否支持操作组件
8244	无法确定针对此操作组件的参数设置数据	检查该操作组件是否在已知类型的列表中
8245	机床控制面板参数设置数据无效	检查针对待确定的类型所存储的机床控制面板参数设置数据是否正确。 针对“SINU_UpdateEthernetMcp”重新生成实例。
8246	机床控制面板的参数化已被操作组件拒绝	检查针对待确定的类型所存储的机床控制面板参数设置数据是否正确。 针对“SINU_UpdateEthernetMcp”重新生成实例。
86xx	内部错误	请联系西门子。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在以下示例中，利用实例数据块“InstUpdateEthernetMcp”对“SINU_UpdateEthernetMcp”的调用进行了编程。在全局数据块“dataMcp”中创建控制数据和状态数据。针对带 1 个基本模块的模块化机床控制面板、4 个按键模块和 2 个 Powerride 倍率开关选择 InOut 参数“toMcp”和“fromMcp”的长度。

图中的观察值显示了成功识别模块化机床控制面板后循环通信步骤中的输出端。

	Name	Datentyp	Startwert	Beobachtungswert
16	Mcp1Data	Struct		
17	enable	Bool	FALSE	TRUE
18	directKeys	Bool	FALSE	FALSE
19	disableDataToMcp	Bool	FALSE	FALSE
20	toMcp	Array[1..30] of Byte		
21	fromMcp	Array[1..26] of Byte		
22	handwheel1	"SINU_typeHandwheelData"		
23	active	Bool	false	TRUE
24	recent	Bool	false	FALSE
25	stopped	Bool	false	FALSE
26	pulses	Int	0	0
27	handwheel2	"SINU_typeHandwheelData"		
28	newData	Bool	FALSE	FALSE
29	identified	Bool	FALSE	TRUE
30	mcpType	USInt	0	160
31	lenghtIn	USInt	0	26
32	lenghtOut	USInt	0	30
33	mcpReady	Bool	FALSE	TRUE
34	busy	Bool	FALSE	FALSE
35	warning	Bool	FALSE	FALSE
36	error	Bool	FALSE	FALSE
37	status	Word	16#0000	16#0000
38	diagnosis	Word	16#0000	16#0000

图 12-11　数据块“dataMcp”中的控制变量和状态变量

调用 SINU_UpdateEthernetMcp

SCL

```
#InstUpdateEthernetMcp(enable := "dataMcp".Mcp1Data.enable,
                       directKeys := "dataMcp".Mcp1Data.directKeys,
                       address := 192,
                       udpPort := 16001,
                       connectionId := 1001,
                       interfaceId := 262,
                       disableDataToMcp := "dataMcp".Mcp1Data.disable,
                       newData => "dataMcp".Mcp1Data.newData,
                       identified => "dataMcp".Mcp1Data.identified,
                       mcpType => "dataMcp".Mcp1Data.mcpType,
                       lengthIn => "dataMcp".Mcp1Data.lenghtIn,
                       lengthOut => "dataMcp".Mcp1Data.lenghtOut,
                       mcpReady => "dataMcp".Mcp1Data.mcpReady,
                       busy => "dataMcp".Mcp1Data.busy,
                       warning => "dataMcp".Mcp1Data.warning,
                       error => "dataMcp".Mcp1Data.error,
                       status => "dataMcp".Mcp1Data.status,
                       diagnosis => "dataMcp".Mcp1Data.diagnosis,
                       fromMcp := "dataMcp".Mcp1Data.fromMcp,
                       toMcp := "dataMcp".Mcp1Data.toMcp,
                       handwheel1 => "dataMcp".Mcp1Data.handwheel1,
                       handwheel2 => "dataMcp".Mcp1Data.handwheel2);
```

更多信息

SINU_TransferHandwheelData (页 1279)

SINU_WriteRecordOverEthernet (页 1263)

12.4.13.3 SINU_TransferHandwheelData

说明

“SINU_TransferHandwheelData”指令从 PLC 向 NCK 传输手轮信号（手轮的旋转脉冲）。通过“SINU_UpdateEthernetMcp”指令，将手轮信号从机床控制面板 (MCP) 或手持操作设备 (HT) 传输到 PLC。向输出端“handwheel1”和“handwheel2”（类型均为 SINU_typeHandwheelData）输出最多 2 个手轮的手轮数据。该 SINU_typeHandwheelData 型的手轮数据可作为指令 SINU_TransferHandwheelData 的输入端使用。最多可以传输 6 个不同操作组件手轮的数据。不需要的手轮数据输入端用零来填充。一旦收到来自机床控制面板或手持终端的新数据，该功能就会开始传输手轮信号。

为了用 NCK 处理手轮信号，必须针对每个所需的手轮"i"设置机床数据 MD11350[i] :=6 和 MD11351[i] :=1，通过机床数据 MD11352[i] 将传输接口 1 至 6 分配给 NCK 的手轮。针对 1:1 的分配，可以针对手轮 1 至 6 设置 MD11352[i] :=i。

更多信息

更多关于参数化的信息可参见*功能手册：进给轴和主轴*中的章节“通过手轮进行手动移动”。

调用时间点

- 每个 PLC 循环调用一次，比如 Main 组织块中

说明

该调用应在调用"SINU_UpdateEthernetMcp"之后进行。借此最新的手轮数据将始终可用。

参数

下表显示了该指令的参数：

参数	声明	数据类型	说明
handwheel1	Input	Variant	手轮 1 的数据 取值范围："SINU_typeHandwheelData"类型的数据或零
handwheel2	Input	Variant	手轮 2 的数据 取值范围："SINU_typeHandwheelData"类型的数据或零
handwheel3	Input	Variant	手轮 3 的数据 取值范围："SINU_typeHandwheelData"类型的数据或零
handwheel4	Input	Variant	手轮 4 的数据 取值范围："SINU_typeHandwheelData"类型的数据或零
handwheel5	Input	Variant	手轮 5 的数据 取值范围："SINU_typeHandwheelData"类型的数据或零
handwheel6	Input	Variant	手轮 6 的数据 取值范围："SINU_typeHandwheelData"类型的数据或零
error	Output	Bool	TRUE:传输手轮数据时出错
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

故障、状态和解决办法

状态/错误代码* (W#16#...)	说明	解决办法
0000	无错误	-
8251	手轮 1 输入端上的错误类型	检查参数 handwheel1，必须为类型“SINU_typeHandwheelData”或零
8252	手轮 2 输入端上的错误类型	检查参数 handwheel2，必须为类型“SINU_typeHandwheelData”或零
8253	手轮 3 输入端上的错误类型	检查参数 handwheel3，必须为类型“SINU_typeHandwheelData”或零
8254	手轮 4 输入端上的错误类型	检查参数 handwheel4，必须为类型“SINU_typeHandwheelData”或零
8255	手轮 5 输入端上的错误类型	检查参数 handwheel5，必须为类型“SINU_typeHandwheelData”或零
8256	手轮 6 输入端上的错误类型	检查参数 handwheel6，必须为类型“SINU_typeHandwheelData”或零
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

示例

在以下示例中，共传输了 3 个手轮的数据。MCP1 的 2 个手轮和 HT10 的 1 个手轮。

SCL

```
// tempError : Bool
// tempStatus : Word

SINU_TransferHandwheelData(handwheel1:="dataMcp".Mcp1Data.handwheel1,
                           handwheel2:="dataMcp".Mcp1Data.handwheel2,
                           handwheel3:="dataMcp".HT10Data.handwheel1,
                           handwheel4:=NULL,
                           handwheel5:=NULL,
                           handwheel6:=NULL,
                           error=>#tempError,
                           status=>#tempStatus);
```

更多信息

SINU_UpdateEthernetMcp (页 1271)

SINU_WriteRecordOverEthernet (页 1263)

12.5 PLC 数据类型：全局库

12.5.1 概览

下表列出了通过 TIA 系统库提供的 PLC 数据类型。为了使用这些数据类型，必须在项目中插入 SINUMERIK ONE（版本 > 6.20）。

名称	说明
SINU_typeHandwheelData (页 1282)	提供针对手轮的脉冲数据。

12.5.2 SINU_typeHandwheelData

说明

PLC 数据类型"SINU_typeHandwheelData"可提供针对手轮的脉冲数据。该数据类型可与指令"SINU_UpdateEthernetMcp"和"SINU_TransferHandwheelData"一起使用。该数据类型在全局库中找不到，必须手动输入后才能使用。

active	
说明	TRUE:手轮已激活/手轮存在
数据类型	Bool

最新	
说明	TRUE:可用的手轮新数据
数据类型	Bool

已停止	
说明	TRUE:手轮停止
数据类型	Bool

脉冲	
说明	最后一次传输到 NCK 时的脉冲偏差
数据类型	Int

12.6 PLC 块 LSINU_* 全局库

12.6.1 概览

下表按字母顺序列举了 SINUMERIK PLC 基本程序的各个块。

名称	说明
LSINU_GenerateSinuUserAlarm (页 1283)	生成 SINUMERIK 用户报警，该报警可由各种报警客户端（HMI Operate、TIA Portal...）进行显示。
LSINU_GetReinitSinuUserAlarms (页 1289)	检测报警适配器的重新初始化，并在一个周期内在输出端上进行报告。
LSINU_GetSinuUserAlarmCounters (页 1291)	输出现场所有报警计数器的内容。
LSINU_Powerride (页 1294)	在 SINUMERIK 系统中显示旋钮开关 (SINUMERIK Powerride)。
LSINU_PowerrideConfig (页 1304)	预配置的倍率设置。
LSINU_ReadGUD (页 1305)	指令用于从 NC 读取全局用户变量 (GUD) 以及用于确定 GUD 变量地址。
LSINU_ReadVar (页 1313)	指令用于读取 NC 的变量。
LSINU_TransferSelData (页 1322)	指令用于在 PLC 和 NC 之间进行数据交换。
LSINU_WriteVar (页 1331)	指令用于写入 NC 的变量。

12.6.2 LSINU_GenerateSinuUserAlarm

通过模块"LSINU_GenerateSinuUserAlarm"生成 SINUMERIK 用户报警，该报警可由各种报警客户端（HMI Operate、TIA Portal...）进行显示。

该模块只允许在出现报警或报警消除时调用。如果在没有报警变化的情况下进行非必要的模块调用，可能会导致运行时间出问题。

参数“alarmCome”和“alarmGone”在 PLC 循环中不能同时具有值 1，否则会导致故障 16#8201（“alarmCome”和“alarmGo”同时生效）。因此，只有通过两个参数的交替上升沿控制才能实现模块的正确处理。

说明

在 SINUMERIK 用户报警的 AppNote 中，您可以找到 SINUMERIK 用户报警的一些定义示例。AppNote 请参见西门子工业在线支持 (https://support.industry.siemens.com/cs/ww/en/view/109812364)页面。

调用时间点

- 每个 PLC 循环调用一次，比如 Main 组织块中。

说明

只允许从“Program cycle”类型的执行层级调用块。

参数

下表列出了模块的参数：

参数	声明	数据类型	说明
alarmCome	Input	Bool	TRUE: 发生 FALSE: 未发生
alarmGo	Input	Bool	TRUE: 消除 FALSE: 未消除
timestamp	Input	LDT	LDT#1970-01-01-00:00:00:时间戳在内部确定/其他值：使用调用者传递的时间戳
alarmNumber	Input	UDInt	报警号码 取值范围：500,000 ... 999,999
alarmIndex	Input	UInt	报警索引 0=No indexing, 1..(max)=indexed 通过 @3%<format>@ 嵌入报警文本

参数	声明	数据类型	说明
alarmCategory	Input	USInt	报警类别 取值范围： 1=Error 2=Message with Acknowledgement 3=Message without Acknowledgement 4=Info
alarmCounter	Input	USInt	0=不计数 1..16 计数器
associatedValue7	Input	Variant	第 1 个附值，通过 @7%<format>@ 嵌入报警文本
associatedValue8	Input	Variant	第 2 个附值，通过 @8%<format>@ 嵌入报警文本
associatedValue9	Input	Variant	第 3 个附值，通过 @9%<format>@ 嵌入报警文本
associatedValue10	Input	Variant	第 4 个附值，通过 @10%<format>@ 嵌入报警文本
alarm	InOut	LSINU_typeAlarmInternalData	内部数据，不得更改！
error	Output	Bool	TRUE:出错
statusId	Output	UInt	故障源 指定要使用“状态”参数评估的故障出现在哪个模块中： 0：无故障 1：模块本身 2: RD_SINFO 11：“LSINU_InstInternalAlarmAdapter”中的故障 12：来自下级警报功能的错误
status	Output	Word	故障代码，只能结合“statusId”进行评估（见表“任务状态”）.

说明

参数“alarm”中的变量应以在 PLC 循环期间始终保留的方式进行保存。这一过程可例如在功能块（背景数据块）的静态区域或者在共享数据块中进行。其中的内容必须至少保留到报告报警已成功消除。

任务状态

statusId	状态/错误代码* (W#16#...)	含义	提示
0	0000	任务结束，无警告或者其他细节	---
1	8208	错误的报警类别	检查参数“alarmCatgegory”。 取值范围： 1=Error 2=Message with Acknowledgement 3=Message without Acknowledgement 4=Info
1	8013	“alarmCome”和“alarmGo”同时以值 1 生效	在一个 PLC 循环中，这两个参数不得同时提供值 1。
1	8014	错误的组织块流程级	仅可在流程级“Program cycle”中调用 FC。
2	8081	当前组织块的启动信息与指定的系统数据类型不匹配	标准模块“RD_SINFO”的故障代码摘录；另请参见 TIA Portal 在线技术支持。
2	8083	最后启动的启动 OB 的启动信息与指定的系统数据类型不匹配	标准模块“RD_SINFO”的故障代码摘录；另请参见 TIA Portal 在线技术支持。
11	8611-8615	内部错误	请初始化数据块“LSINU_InstInternalAlarmAdapter” 如果仍然出错，请拨打服务热线。
11	8205	错误的“alarmNumber”	检查参数“alarmNumber” 取值范围：500,000 ... 999,999
11	8206	错误的“uniqueAlarmId”	检查参数“alarm”中的元素“internalData.uniqueAlarmId” 取值范围 0 ... 65534 如要重新进行初始化，参数“alarm”中的两个元素“internalData.uniqueAlarmId”和“internalData.runtimeUid”都应设置为 0。
11	8207	错误的“alarmCounter”	检查参数“alarmCounter” 取值范围：0...16
11	8409	已达到同时存在的报警的最大数量	减少存在的 SINUMERIK 用户报警
11	8015	已消除的消息，在其之前没有出现消息	检查参数“alarmNumber”、“alarmCategory”和“alarmIndex”

statusId	状态/错误代码* (W#16#...)	含义	提示
11	8016	报警相同，同一报警类别内已经存在具有相同报警编号和报警索引的报警。	检查参数“alarmNumber”、“alarmCategory”和“alarmIndex” 如果两个报警的上述三个参数相同，即表示两个报警相同，也就是“重复”。请检查三个参数并更改至少一个参数值。
12	0085	“仅供参考”类型的信息	来自下级警报功能的错误。
12	8001	无效的静态报告信息	来自下级警报功能的错误。
12	8002	没有有效的静态报告信息	来自下级警报功能的错误。
12	8004	已达到信息附值的最大大小512字节。	来自下级警报功能的错误。
12	8005	参数SIG中有一个正信号沿，并且已经存在一条尚未确认的信息。	来自下级警报功能的错误。
12	8007	已消除的消息，在其之前没有出现消息	来自下级警报功能的错误。
12	8087	已禁用静态信息	来自下级警报功能的错误。
12	8089	信息过长	来自下级警报功能的错误。
12	80Ax	参数 SD_i 中的值无效。	来自下级警报功能的错误。
12	80C1	CPU 当前未生成任何信息，因为初始化例行程序正在运行（例如，在 RUN 中下载后就是这种情况）。请稍后重试。	来自下级警报功能的错误。
12	80C2	已发送每个时间单位的最大信息数。请稍后重试。	来自下级警报功能的错误。
12	80C3	将使用所有动态信息实例。请稍后重试。	来自下级警报功能的错误。
12	80C4	信息将被输出并且无法覆盖。请稍后重试。	来自下级警报功能的错误。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE			

示例

SCL

```
VAR
  statAlarmCome : Bool;
  statAlarmComeOld : Bool;
  statAlarm : "LSINU_typeAlarmInternalData"
END_VAR

VAR_TEMP
  tempAssociatedValue7 : UDInt;
  tempAssociatedValue8 : Real;
  tempError : Bool;
  tempStatusId : UInt;
  tempStatus : Word;
END_VAR

VAR CONSTANT
  LDT_DEFAULT : LDT := LDT#1970-01-01-00:00:00;
END_VAR
BEGIN
  IF #statAlarmCome <> #statAlarmComeOld THEN

    // set associated values
    #tempAssociatedValue7 := 999;
    #tempAssociatedValue8 := 1.234;

    // generate alarm
    "LSINU_GenerateSinuUserAlarm"(alarmCome := #statAlarmCome,
                                  alarmGo:=NOT #statAlarmCome,
                                  timestamp := #LDT_DEFAULT,
                                  alarmNumber:= 500000,  // Alarmnumber
                                  alarmIndex:= 0,        // no index
                                  alarmCategory:= 1,     // Error
                                  alarmCounter:=#alarmCounter,
                                  associatedValue7 := #tempAssociatedValue7,
                                  associatedValue8 := #tempAssociatedValue8,
                                  associatedValue9 := NULL,
                                  associatedValue10 := NULL,
                                  error => #tempError,
                                  statusId => #tempStatusId,
                                  status => #tempStatus,
                                  alarm := #statAlarm
                                  );

  // set edge detection in case of success
  IF NOT #tempError THEN
    #statAlarmComeOld := #statAlarmCome;
  END_IF;

END_FUNCTION_BLOCK
```

12.6.3 LSINU_GetReinitSinuUserAlarms

块“LSINU_GetReinitSinuUserAlarms”检测所有已使用报警的重新初始化情况，并通过输出端“reinitNeeded”在一个循环内持续显示检测结果。在“示例”章节中有一份程序节选，介绍了输出端“reinitNeeded”与 AppNote 函数“LOEM_ScanDb2LikeAlarms”互连的方式。

如果在报警文本更改后，将 RUN 中的模块加载到控制系统中，则必须重新初始化所有已使用的报警。

调用时间点

- 每个 PLC 循环调用一次，比如 Main 组织块中。

说明

只允许从“Program cycle”类型的执行层级调用块。

参数

下表列出了模块的参数：

参数	声明	数据类型	说明
reinitNeeded	Output	Bool	TRUE:已检测到报警适配器的重新初始化。 所有待处理的报警均必须由用户重新初始化。信号存在一个周期。
error	Output	Bool	故障状态 TRUE:任务受到消极应答或无法执行。
statusId	Output	UInt	故障源 指定要使用“状态”参数评估的故障出现在哪个模块中： 0：无故障 1：模块本身 2: RD_SINFO
status	Output	Word	故障代码，只能结合“statusId”进行评估（见表“任务状态”）.

任务状态

statusId	状态/错误代码* (W#16#...)	含义	提示
0	0000	任务结束，无警告或者其他细节	---
1	8214	错误的组织块流程级	仅可在流程级“Program cycle”中调用 FC。
2	8081	当前组织块的启动信息与指定的系统数据类型不匹配	标准模块“RD_SINFO”的故障代码摘录；另请参见 TIA Portal 在线技术支持。
2	8083	最后启动的启动 OB 的启动信息与指定的系统数据类型不匹配	标准模块“RD_SINFO”的故障代码摘录；另请参见 TIA Portal 在线技术支持。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE			

示例

下面是“LOEM_ScanDb2LikeAlarms”和“LSINU_GetReinitSinuUserAlarms”循环功能调用的组织块示例。组织块监控并生成 SINUMERIK 用户报警或检测重新初始化。

说明

在 SINUMERIK 用户报警的 AppNote 中，您可以找到 SINUMERIK 用户报警的一些定义示例。AppNote 请参见西门子工业在线支持页面。

SCL

```
ORGANIZATION_BLOCK "LOEM_Main"

// Example organisation block for cyclic execution calls function "LOEM_ScanDb2LikeAlarms"
// and LSINU_GetReinitSinuUserAlarms for monitoring
// and generating SINUMERIK user alarms and detect reinitialization

VAR_TEMP
   tempTimestamp : LDT;
   tempRetValRdSysT : Int;
   tempError : Bool;
   tempStatusId : UInt;
   tempStatus : Word;
   tempReinitNeeded : Bool;
END_VAR
```

SCL

```
BEGIN

   REGION raise alarms
      REGION detect reinitialization
         "LSINU_GetReinitSinuUserAlarms"(reinitNeeded => #tempReinitNeeded,
                                         error => #tempError,
                                         statusId => #tempStatusId,
                                         status => #tempStatus);
      END_REGION

      REGION raise Db2Like alarms
         //call function from AppNote
        "LOEM_ScanDb2LikeAlarms"(ack := "LOEM_Db2LikeAlarms".ack,
                                 reinitNeeded:=#tempReinitNeeded,
                                 timestamp := #tempTimestamp,
                                 error => #tempError,
                                 statusId => #tempStatusId,
                                 status => #tempStatus,
                                 db2LikeAlarms := "LOEM_Db2LikeAlarms".db2likeAlarms);
      END_REGION

   END_REGION
END_ORGANIZATION_BLOCK
```

参见

SIOS AppNote User Alarms SINUMERIK ONE (https://support.industry.siemens.com/cs/ww/en/view/109812364)

12.6.4 LSINU_GetSinuUserAlarmCounters

使用模块“LSINU_GetSinuUserAlarmCounters” 以阵列的形式输出所有报警计数器的内容。

总共定义并输出了 16 个计数器，无论 SINUMERIK 用户报警是否实际使用这些计数器进行计数。通过这 16 个计数器，可将 SINUMERIK 用户报警分配给例如机器部件或功能。因此，可以为这些机器部件或功能计算已生效的 SINUMERIK 用户报警。

在通过“LSINU_GenerateSinuUserAlarm”生成 SINUMERIK 用户报警时，通过参数“alarmCounter”确定，在报警出现或消除时增加或减少 16 个可用计数器中的哪一个。如果规定“计数器 = 0”，意味着该报警不影响任何计数器。

说明

在 SINUMERIK 用户报警的 AppNote 中，您可以找到 SINUMERIK 用户报警的一些定义示例。AppNote 请参见西门子工业在线支持 (https://support.industry.siemens.com/cs/ww/en/view/109812364)页面。

调用时间点

- 每个 PLC 循环调用一次，比如 Main 组织块中。

说明

只允许从“Program cycle”类型的执行层级调用块。

参数

下表列出了模块的参数：

参数	声明	数据类型	说明
alarmCounters	Output	Array[1..16] of UInt	已生效的 SINUMERIK 用户报警的计数器阵列
error	Output	Bool	故障状态 TRUE:任务受到消极应答或无法执行。
statusId	Output	UInt	故障源 指定要使用“状态”参数评估的故障出现在哪个模块中： 0：无故障 1：模块本身 2: RD_SINFO
status	Output	Word	故障代码，只能结合“statusId”进行评估（见表“任务状态”）.

任务状态

statusId	状态/错误代码* (W#16#...)	含义	提示
0	0000	任务结束，无警告或者其他细节	---
1	8014	错误的组织块流程级	仅可在流程级"Program cycle"中调用 FC。
2	8081	当前组织块的启动信息与指定的系统数据类型不匹配	标准模块"RD_SINFO"的故障代码摘录；另请参见 TIA Portal 在线技术支持。
2	8083	最后启动的启动 OB 的启动信息与指定的系统数据类型不匹配	标准模块"RD_SINFO"的故障代码摘录；另请参见 TIA Portal 在线技术支持。
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE			

示例

SCL

```
VAR
  valueOfCounter13 : UInt;
END_VAR

VAR_TEMP
  tempAlarmCounters : Array[1..16] of UInt;
  tempEerror : Bool;
  tempStatusId : Word;
  tempStatus : UInt;
END_VAR
BEGIN

  "LSINU_GetSinuUserAlarmCounters"(alarmCounters => #alarmCounters,
                                   error => #error,
                                   statusId => #statusId,
                                   status => #status
                                   );

  #valueOfCounter13 := #alarmCounters[13];

END_FUNCTION_BLOCK
```

12.6.5 LSINU_Powerride

说明

功能块 "LSINU_Powerride" 将旋钮开关 (SINUMERIK Powerride) 集成到 SINUMERIK 系统中。

通过 Powerride，能够以百分比的形式更改进给率，并立即对机器产生作用。为此，将 Powerride 的补偿值乘以编程的进给率。

更改进给率的接口：

- 通道实例中的单个接口
- 轴实例中的单个接口

有关 Powerride 硬件的详细信息，请参见 *SINUMERIK ONE 机床控制面板产品手册*。

您可通过功能块的参数查询数据，并设置 Powerride 的行为。

在数据块“LSINU_PowerrideConfig”中完成组态后，即可立即使用 Powerride。通过参数“powerrideConfig”参数将所需组态传输到功能块。对于具体的调整，使用“LSINU_typePowerrideConfig”类型和您的组态设置创建您自己的数据块。

如需自己编程 Powerride 的行为，您可直接使用参数“powerrideInputs”查询信号，或使用“powerrideOutputs”进行控制。

调用时间点

- 每个实例的每个 PLC 循环各一次，例如在主组织块中

参数

下表列出了模块的参数：

参数	声明	数据类型	说明
enable	Input	Bool	激活 Powerride 功能 该参数仅可在上电（控制系统启动）完成后设置为 TRUE。 TRUE: 检查组态数据，如果顺利，则开始处理功能块。上电后，如果背景数据块剩余，则最后一个“switchPos”将生效。 FALSE： 不处理功能块，保留初始值。
preset	Input	Bool	更改“switchPos” 在正脉冲沿，将应用“presetValue”作为当前的“switchPos”。为此，必须对“presetValue”进行有效参数化。将输出补偿值“powerrideConfig.switchPos[1].value”。 将更新“grayCode”和“ovrFactor”形式的显示内容。
forceSwitchPos1	Input	Bool	将开关调至位置 1。 在正脉冲沿，开关将被设置在位置 1，并且输出补偿值“powerrideConfig.switchPos[1].value”。 通过设置 presetValue = 1 和 preset = TRUE 可以获得相同的结果。就此而言，提供此参数是为了方便起见。如果“presetValue”包含有效值，同时设置了“preset”，则执行“forceSwitchPos1”的功能。 将更新“grayCode”和“ovrFactor”形式的显示内容。
buttonColor	Input	USInt	规定开关 LED 的颜色。目前，只有绿色可以通过设置位 1 来实现。 取值范围： 0、1、4、5：无颜色选择 2、3、6、7：绿色

参数	声明	数据类型	说明
presetValue	Input	USInt	“switchPos”的新值 取值范围：1...36 在“preset”的正脉冲沿，该值被应用为新的“switchPos” **特殊情况 1** “presetValue”介于“powerrideConfig.pos100”和“powerrideConfig.posMax”之间，并且“switchPos”（当前位置）小于“powerrideConfig.pos100” 出于安全方面的考量，这种情况下通过“preset”应用“powerrideConfig.pos100”的值。 **特殊情况 2** “presetValue”大于“powerrideConfig.posMax”并且小于 36 通过“preset”应用“powerrideConfig.posMax”的值。
powerrideInputs	InOut[1)]	LSINU_typePowerrideInputs	Powerride 发送至 PLC 的信号
powerrideOutputs	InOut[1)]	LSINU_typePowerrideOutputs	PLC 发送至 Powerride 的信号
powerrideConfig	InOut[1)]	LSINU_PowerrideConfig	组态数据
buttonPressed	Output	Bool	显示是否已按下开关。
ovrFactor[2)]	Output	USInt	将输出在“powerrideConfig.switchPos[1..36].value”下参数设置的值。 取值范围：0...200
grayCode[2)]	Output	USInt	将以格雷码的形式输出在“powerrideConfig.switchPos[1..36].value”下参数设置的值。
switchPos	Output	USInt	显示 Powerride 倍率开关的当前开关位置 取值范围：1...powerrideConfig.posMax

参数	声明	数据类型	说明
error	Output	Bool	显示是否存在故障 “status”参数下显示导致错误的原因。 TRUE:出错，将复位其他输出端 FALSE：如果“enable” = FALSE 或者不存在故障
status	Output	Word	执行任务时的状态编号。 出现故障时的故障编号。

1) 这些参数也可以定义为纯输入 (powerrideInputs, powerrideConfig) 或输出 (powerrideOutputs)。但是，此时使用 InOut 具有运行时间优势，因为 InOut 参数是作为指示地址而不是副本来传输。

2) 在机床数据中设置对补偿值的编译（例如：MD12020 $MN_OVR_FEED_IS_GRAY_CODE）。

脉冲图

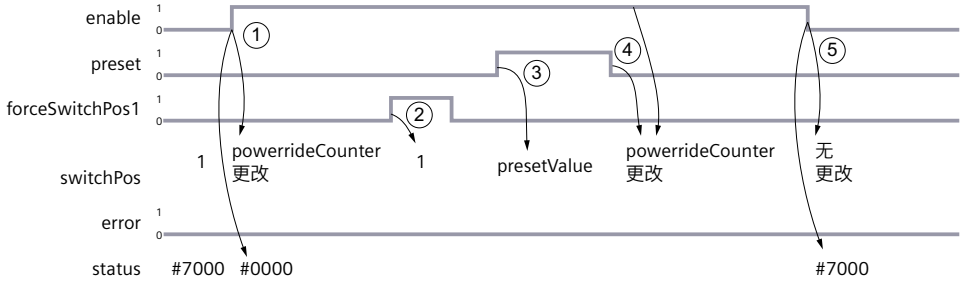


图 12-12 LSINU_Powerride：在旋转开关上进行更改后的信号走向

1. 将设置参数“enable”：
   顺利检查配置后，模块将旋钮 (“powerrideCounter”) 上的更改传输至“switchPos”。输出端“ovrFactor”和“grayCode”也将根据传输的组态进行设置。
2. 将设置参数“forceSwitchPos1”：
   输出端“switchPos”将被设置为 1。输出端“ovrFactor”和“grayCode”将根据“switchPos[1]”的组态进行设置。
3. 将设置参数“preset”，参数“forceSwitchPos”为 0：
   在该输出端上将根据“presetValue”输出“switchPos”，并设置“ovrFactor”和“grayCode”。
4. 参数“forceSwitchPos1”和“preset”均为 0：
   模块再次将旋钮上的更改传输至“switchPos”，“ovrFactor”和“grayCode”将根据组态进行设置。
5. 将复位参数“enable”：
   这些输出端保留其最后的数值。

故障、状态和解决办法

状态/错误代码* (W#16#...)	**说明**	**解决办法**
0000	无错误	-
7000	不执行该功能	将“enable”设置为 TRUE
7101	警告 [1]：“buttonColor”无效	使用 0...7 参见“LED 色卡”，在补偿后无需复位“enable”
7102	报警：powerrideConfig.switchPos[].ledColor 无效	使用 0...7 参见 LED 色卡 (页 1299)，执行“enable”的复位，以便重新读入修正后的组态
7103	报警： powerrideConfig.switchPos[].vibrationIndex ungültig[4]	使用 0...7 执行“enable”的复位，以便重新读入修正后的组态
8101	正“preset”脉冲沿，但“presetValue”[3] 大于 36	使用 1...36 将“preset”复位
8102	正“preset”脉冲沿，但“presetValue”小于 1	使用 1...36 将“preset”复位
8212	组态错误 [2]：“posMax”大于 36	使用 1...36 将“enable”复位
8221	组态错误：“pos100”小于 1	使用 1...36 将“enable”复位
8222	组态错误：“posMax”小于 1	使用 1...36 将“enable”复位
8223	组态错误：“pos100”大于“posMax”	使用 pos100 ≤ posMax 将“enable”复位
8231	组态错误：“switchPos[x].value”在一个或多个数据集中超出有效范围 (0 ... 200%)	使用 0...200 修正数据集，将“enable”复位
* 状态代码：7xxx（信息、警告），参数 error = FALSE；故障代码：8xxx（故障），参数 error = TRUE		

关于表格的提示

[1] 如果出现警告，Powerride 的其他功能不受限制。

[2] 如果出现组态错误，出于安全原因，输出 switchPos[1]。消除组态错误后，必须将输入端“enable”处的信号设置为 FALSE，然后再设置为 TRUE，以便再次读取组态数据并重置错误。

[3] 如果参数"presetValue"出现错误，在补偿后，必须将输入端"preset"设置为 FALSE，然后再设置为 TRUE，以便在正脉冲沿应用现在正确的"presetValue"。

[4] 可选的振动模式参见 *SINUMERIK ONE 机床控制面板产品手册第一部分：机床控制面板 xxxx* 章节“调试”下的段落“Powerride 振动模式”。

更多信息

关于 Powerride 与 PLC 之间的信号传输的详细说明，请参见 *SINUMERIK ONE 机床控制面板产品手册第一部分：机床控制面板 xxxx* 章节“调试”下的段落“输入映像”和“输出映像”。

关于 LED 颜色的说明，请参见章节“LED 色卡 (页 1299)”。

12.6.5.1 LED 色板

下表列举介绍蓝色、绿色和红色三种单色可以构成的颜色组合。

说明

针对开关（"LSINU_Powrride"的输入端"buttonColor"），目前只能选择绿色。

组合色	蓝色 (Bit 2)	绿色 (Bit 1)	红色 (Bit 0)
-	-	-	-
红色	-	-	x
绿色	-	x	-
黄色	-	x	x
蓝色	x	-	-
洋红色	x	-	x
青色	x	x	-
白色	x	x	x

12.6.5.2 示例：为通过机床控制面板集成 Powerride 倍率开关做好准备

前提条件

项目中包含 **SINUMERIK ONE PLC Basic Program plus** 库的下列块：

- LSINU_Powerride, LSINU_PowerrideConfig

项目中包含 **SINUMERIK ONE PLC Basic Program plus** 库的下列数据类型：

- LSINU_typePowerrideInputs, LSINU_typePowerrideOutputs, LSINU_typePowerrideConfig, LSINU_typePowerrideConfigPos

项目中包含 **SINUMERIK ONE PLC Basic Program** 库的下列数据类型：

- LBP_typeModularMcpBaseInput, LBP_typeModularMcpUniversalInput, LBP_typeHandwheel
- LBP_typeModularMcpBaseOutput, LBP_typeModularMcpUniversalOutput

通过机床控制面板集成 Powerride 倍率开关

示例所用的 SINUMERIK ONE MCP，具有两个通过以太网 (IE) 连接在一起的 Powerride 模块。

全局数据块“ModularMcpData”用于输入和输出映像。此块不是库的构成部分。以“ModularMcpData”为例介绍如何借助库中随附提供的数据类型，为 SINUMERIK ONE MCP 映射输入映像和输出映像，以及如何使用这个数据块。

ModularMcpData

	Name	Datentyp	Offset	Startwert	Remanenz	Kommentar
1	Static					
2	in	Struct	0.0			
3	base	"LBP_typeModularMcpBaseInput"	0.0			Eingangsabbild von Basissignalen für modulare MCP
4	keypad1	"LBP_typeModularMcpUniversalInput"	6.0			Eingangsabbild des Universal-Bedienfelds für modulare MCP
5	keypad2	"LBP_typeModularMcpUniversalInput"	10.0			Eingangsabbild des Universal-Bedienfelds für modulare MCP
6	keypad3	"LBP_typeModularMcpUniversalInput"	14.0			Eingangsabbild des Universal-Bedienfelds für modulare MCP
7	keypad4	"LBP_typeModularMcpUniversalInput"	18.0			Eingangsabbild des Universal-Bedienfelds für modulare MCP
8	powerride1	"LSINU_typePowerrideInputs"	22.0			Eingangsabbild von Powerride
9	powerride2	"LSINU_typePowerrideInputs"	24.0			Eingangsabbild von Powerride
10	powerride3	"LSINU_typePowerrideInputs"	26.0			Eingangsabbild von Powerride
11	handwheel1	"LBP_typeHandwheel"	28.0			Anwender-Datentyp für Handrad
12	handwheel2	"LBP_typeHandwheel"	32.0			Anwender-Datentyp für Handrad
13	out	Struct	36.0			
14	base	"LBP_typeModularMcpBaseOutput"	36.0			Ausgangsabbild von Basissignalen für modulare MCP
15	keypad1	"LBP_typeModularMcpUniversalOutput"	38.0			Ausgangsabbild des Universal-Bedienfelds für modulare MCP
16	keypad2	"LBP_typeModularMcpUniversalOutput"	42.0			Ausgangsabbild des Universal-Bedienfelds für modulare MCP
17	keypad3	"LBP_typeModularMcpUniversalOutput"	46.0			Ausgangsabbild des Universal-Bedienfelds für modulare MCP
18	keypad4	"LBP_typeModularMcpUniversalOutput"	50.0			Ausgangsabbild des Universal-Bedienfelds für modulare MCP
19	powerride1	"LSINU_typePowerrideOutputs"	54.0			Ausgangsabbild von Powerride
20	powerride2	"LSINU_typePowerrideOutputs"	60.0			Ausgangsabbild von Powerride
21	powerride3	"LSINU_typePowerrideOutputs"	66.0			Ausgangsabbild von Powerride

图 12-13　数据块：ModularMcpData

下表举例介绍输入和输出地址：

	数据字节编号
输入数据	"ModularMcpData".in.powerride1 和 "ModularMcpData".in.powerride2
输出数据	"ModularMcpData".out.powerride1 和 "ModularMcpData".out.powerride2

说明

可以通过固件版本，更改机床控制面板输入和输出映像中的 Powerride 数据地址。当前地址分配参见 *SINUMERIK ONE MCP 产品手册*。

12.6.5.3 示例：调用 LSINU_Powerride

说明

示例介绍如何调用“LSINU_Powerride”。SINUMERIK ONE MCP 的输入和输出映像必须事先分配给“SINU_UpdateEthernetMcp”块的参数“fromMcp”和“toMcp”。

在示例中，全局数据块“ModularMcpData”包含输入和输出映像。相关信息参见 *SINUMERIK ONE MCP 产品手册*。

通过数据块“PowerrideDb”为“LSINU_Powerride”的输入和输出参数提供数据。

PowerrideDb

Name	Datentyp	Startwert
Static		
pr1enable	Bool	false
pr1disable	Bool	false
pr1preset	Bool	false
pr1forceSwitchPos1	Bool	false
pr1buttonColor	USInt	0
pr1presetValue	USInt	1
pr1error	Bool	false
pr1buttonPressed	Bool	false
pr1ovrFactor	USInt	0
pr1grayCode	USInt	0
pr1switchPos	USInt	0
pr1status	Word	16#0
devicePR1In	"LSINU_typePowerrideInputs"	
devicePR1Out	"LSINU_typePowerrideOutputs"	

图 12-14 数据块：PowerrideDb

为参数“powerrideConfig”使用库中提供的“configFeed”组态。

以下代码所示，为在“Main [OB1]”中调用“LSINU_Powerride”。首先，调用指令“SINU_UpdateEthernetMcp”。该指令用于通过工业以太网与操作元件通信。Powerride 倍率开关块本身只有在与机床控制面板进行循环通信时才会激活 ("mcpReady" = TRUE)，并且不应该以任何其他方式明确禁用 ("pr1disable")

代码示例

SCL

```
#InstUpdateEthernetMcp(enable := TRUE,
                       address := 198,

                       udpPort := 16001,

                       connectionId := 1001,

                       interfaceId := 262,

                       mcpReady => #tempMcpReady,

                       newData => #tempNewData,

                       identified => #tempIdentified,

                       mcpType => #tempMcpType,

                       lengthIn => #tempLengthIn,

                       lengthOut => #tempLengthOut,

                       busy => #tempBusy,

                       warning => #tempWarning,

                       error => #tempError,

                       status => #tempStatus,

                       fromMcp := "ModularMcpData".in,

                       toMcp := "ModularMcpData".out);

"PowerrideDb".pr1enable := #tempMcpReady
                           AND NOT "PowerrideDb".pr1disable;

"InstPowerride1"(enable := "PowerrideDb".pr1enable,
                 preset := "PowerrideDb".pr1preset,
                 forceSwitchPos1 := "PowerrideDb".pr1forceSwitchPos1,
                 buttonColor := "PowerrideDb".pr1buttonColor,
                 presetValue := "PowerrideDb".pr1presetValue,
                 error => "PowerrideDb".pr1error,
                 buttonPressed => "PowerrideDb".pr1buttonPressed,
                 ovrFactor => "PowerrideDb".pr1ovrFactor,
                 grayCode => "PowerrideDb".pr1grayCode,
                 switchPos => "PowerrideDb".pr1switchPos,
                 status => "PowerrideDb".pr1status,
                 powerrideInputs := "ModularMcpData".in.powerride1,
```

SCL

```
powerrideOutputs := "ModularMcpData".out.powerride1,
powerrideConfig := "LSINU_PowerrideConfig".configFeed);
```

参见

示例：为通过机床控制面板集成 Powerride 倍率开关做好准备 (页 1299)

12.6.6 LSINU_PowerrideConfig

说明

数据块“LSINU_PowerrideConfig”包含 Powerride 的预组态设置。您可以通过参数“powerrideConfig”，将提供的组态“configFeed”或“configSpindle”传递到功能块“LSINU_Powerride”。

使用包含您组态设置的“LSINU_typePowerrideConfig”类型创建数据块，可以实现用户自定义组态。

说明

升级 **SINUMERIK ONE PLC Basic Program plus** 库时，数据块“LSINU_PowerrideConfig”会被覆盖。

针对 Powerride 的预组态设置

configFeed	
说明	进给组态
数据类型	LSINU_typePowerrideConfig

采用此组态时，Powerride 的行为如下所述：

- 位置 19 对应 100 %。
- 最大位置是 23，对应补偿值 120 %。也就是说，继续向右转动不会提高补偿值。
- 到达位置 19 后，延迟 500 ms，以模式 2 振动。
- 在位置 1 (0 %) 处，LED 灯带熄灭，以模式 1 振动。
- 在位置 2 至 18（1 % 至 95 %），LED 灯带亮白光。
- 在位置 19 (100 %)，LED 灯带亮绿光。

- 在位置 20 至 22（105 % 至 115 %），LED 灯带亮黄光。
- 从位置 23 (120 %) 开始，LED 灯带亮红光，到达位置 23 后，以模式 3 振动。
- “在 Powerride 上快速左转 (> "quickTurnSpeedThreshold") 时设置位置 1”功能禁用（参见 LSINU_typePowerrideConfig 提示信息）。

configSpindle	
说明	主轴配置
数据类型	LSINU_typePowerrideConfig

采用此组态时，Powerride 的行为如下所述：

- 位置 11 对应 100 %。
- 最大位置是 15，对应补偿值 120 %。也就是说，继续向右转动不会提高补偿值。
- 到达位置 11 后，延迟 500ms，以模式 2 振动。
- 在位置 1 (0 %) 处，LED 灯带熄灭，以模式 1 振动。
- 在位置 1 至 10（50 % 至 95 %），LED 灯带亮白光。
- 在位置 11 (100 %)，LED 灯带亮绿光。
- 在位置 12 至 14（105 % 至 115 %），LED 灯带亮黄光。
- 从位置 15 (120 %) 开始，LED 灯带亮红光，到达位置 15 后，以模式 3 振动。
- “在 Powerride 上快速左转 (> "quickTurnSpeedThreshold") 时设置位置 1”功能禁用（参见 LSINU_typePowerrideConfig 提示信息）。

参见

错误代码和状态代码 (页 1345)

LSINU_typePowerrideConfig (页 1340)

12.6.7 LSINU_ReadGUD

说明

配置所需的数据类型不包含在库基本程序升级版中。必须将这些数据类型从“SINUMERIK ONE PLC Basic Program V6.x”库复制到 TIA Portal 项目中（命名约定：“LBP_type*”）。

功能

功能块 "LSINU_ReadGUD" 用于读取 NC 或通道区域中的全局用户数据 (GUD)。

对于范围类型，请参阅下方的调用组合表及 NC-VAR 选择器的在线帮助。

调用 "LSINU_ReadGUD" 时，NC 变量的名称以字符串的形式传输：参数 "Addr" = <DB name>.<S7 name>。

请求读取 NC 变量

使用正向信号边沿调用 "LSINU_ReadGUD"，参数“Req”= 0 → 1

NC 的 S7 名称：参数 "Addr" = <DB name>.<S7 name>

参数“Area”、“Unit”、“Index1”和“Index2”：有关变量寻址的更多信息，请参见部分："LSINU_ReadVar" (页 1313) 中的说明和 NC-VAR 选择器的在线帮助。

GUD 变量还可通过 "LSINU_ReadVar" 读取并使用 "LSINU_WriteVar" 写入。为此，需要一个地址令牌，其在激活参数“CnvtToken”时生成，并在调用 "LSINU_ReadVar" 和“LSINU_WriteVar”，参数“Addr1”...“Addr8” = <Token> 时用作变量地址。如果 GUD 变量包含具有多个值的数组，则还必须为参数“Line1”...“Line8”= <Index> 分配数组索引。

完成读取请求

读取请求成功完成：参数 "Done" == 1。

读取请求完成但出现错误：参数“Error”== 1，错误原因 **RD := "DB_GUD".ValueRD := "DB_GUD".Value**“State”中的 RD :="DB_GUD"

一般条件

- "LSINU_ReadGUD" 具有多实例能力。
  每次调用 "LSINU_ReadGUD" 都必须从用户区分配一个单独的实例或背景数据块。
- 只允许在 PLC 基本程序的循环部分读取 GUD 变量（"LSINU_ReadGUD"以 Req = 1 调用 ）（z. B. "Main [OB1]"）。如果任务未启动（以 Req = 0 调用 "LSINU_ReadGUD"），参数也可写入 PLC 基本程序（"Startup [OB100]"）的启动部分。（z. B. "Main [OB1]"）然而，通过以 Req = 1 调用 "LSINU_ReadGUD"，可使用之前写入 PLC 基本程序循环部分中的参数执行任务。
- GUD 变量的名称只能使用大写字母。
- GUD 变量的读取通常会持续多个 PLC 循环。
- 一次调用最多可传输 230 字节。

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

形式参数的描述

信号	类别	类型	取值范围	含义
Req	I	BOOL	---	任务以正向信号边沿开始
Addr	I	VARIANT	"<DBName>". <VarName>	STRING 类型变量中的名称
Area	I	BYTE	0, 2	NC 区域 0 通道域 2
Unit	I	BYTE	1, 2, ... 10	区域== NC：1 区域 == 通道：通道号
Index1	I	INT	0, 1, 2, ... <max. array index>	数组索引 1 对于无数组索引的变量：0
Index2	I	INT	0, 1, 2, ... <max. array index>	数组索引 2 对于无第二个数组索引的变量：0
CnvtToken	I	BOOL	---	激活 10 字节变量令牌的生成
VarToken	I	VARIANT	"<DB name>". <S7 name>	10 字节令牌的结构（参见示例）
Error	O	BOOL	---	TRUE：任务否定应答或任务无法执行
DONE	O	BOOL	---	TRUE：任务成功执行。
State	O	WORD	---	请参见“错误代码”段落
RD	I/O	VARIANT		读取数据的目标区域，例如与要读取变量具有相同数据类型的 DB 区或位存储器

错误代码

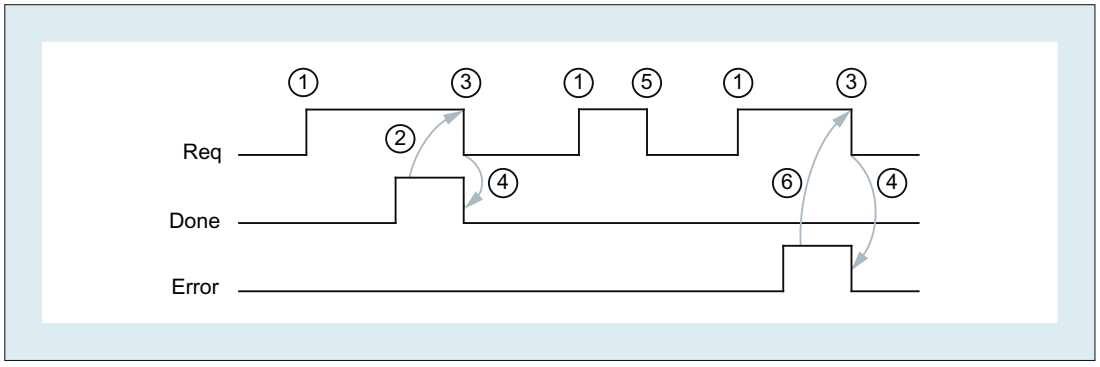
State	含义	说明
WORD L		
1	访问错误	
2	任务出错	任务中的变量编译错误
3	否定应答，任务不可执行	内部错误，可能的补救措施： NC RESET

State	含义	说明
WORD L		
4	数据区域或数据类型不匹配	检查 要在 RD 中读取的数据
4	可用的本地用户内存不足	读取的变量比 RD 中指定的变量长
5	格式转换错误	地址参数必须作为字符串传输。
6	FIFO 已满	必须重复任务， 因为队列已满
10	寻址出错	单元包含值 0
11	变量地址无效	检查地址（或变量名称）、区域、单元
15	字符串超过 32 个字符	GUD 变量名称过长

配置步骤

要想读取 GUD 变量，它的名称必须存储在字符串变量中。

流程图



① 用户：设置请求，Req = 0 → 1

② "LSINU_ReadGUD":读取成功完成，Done = 1

③ 用户：如果 Done == 1，重置请求：1 → 0

④ "LSINU_ReadGUD":重置任务确认，Done = 0

⑤ 用户：如果 Done == 0 且 Error == 0，不允许重置请求 Req = 1 → 0

⑥ "LSINU_ReadGUD":读取服务完成但出现错误，Error = 1
用户：重置请求，如果 Error == 1 且 Req = 0，可能的进一步错误处理

调用示例 1

从全局区域读取全局变量：

- 名称"_SC_NCK_ROU_R[0]"
- 类型：REAL
- 用 VarToken 填充一个 10 字节结构，以便稍后使用"LSINU_WriteVar"写入变量时使用等

读取 GUD 变量：带背景数据块 "LSINU_ReadGUD_DB" 的"LSINU_ReadGUD"

在本例中，全局数据块 "DB_GUD" 包含要读取变量的名称、由调用填充的"VarToken"的结构和用于获取读取值的 REAL 值。

```
DATA_BLOCK "DB_GUD"
VERSION : 0.1
NON_RETAIN
   VAR
   GUD_real : String
   Token_real : Struct
    SyntaxID : Byte;
    AreaUnit : Byte;
    ColIndex : Word;
    RowIndex : Word;
    BlockType : Byte;
    NumRows : Byte;
    S7Typ : Byte;
    NcLength : Byte;
   END_STRUCT;
Value_real : Real;
END_VAR
BEGIN
   GUD_real := '_SC_NCK_ROU_R';
END_DATA_BLOCK


// Test "LSINU_ReadGUD"
// real, NC GUD
"LSINU_ReadGUD_DB"(Req :="Tag_1", // Bool
     Addr := "DB_GUD".GUD_real,
```

```
Area := 0,                          // Global GUD, from area NC
Addr2 := "NCVAR".C1_RP_rpa11_1,
Unit := 1,
Index1 := 0,
Index2 := 0,
CnvtToken := TRUE,
VarToken := "DB_GUD".Token_real,
Error => "Tag_2",                   // Bool
Done => "Tag_3",                    // Bool
State => "Tag_4",                   // Word
RD := "DB_GUD".Value_real);
```

在本 SCL 示例中，"LSINU_ReadGUD" 由设置“Tag_1”激活。该函数无错误地执行，即"Tag_2" = false, "Tag_3" = true, "Tag_4" = 0。“DB_GUD”中的“Token_real”结构由 "LSINU_ReadGUD" 填充，如下所示：

SyntaxID = 16#82

AreaUnit = 16#01

ColIndex = 16#0002

RowIndex = 16#0000

BlockType = 16#36

NumRows = 16#01

S7Typ = 16#0F

NcLength = 16#08

_SC_NCK_ROU_R[0] 中的值写入“DB_GUD”.Value_real。

调用示例 2

从通道 1 读取 GUD 变量：

- 名称“SC_CONT_NAME[0]”
- 类型：STRING[32]
- 用 VarToken 填充一个 10 字节结构，以便稍后使用“LSINU_WriteVar”写入变量时使用等

读取 GUD 变量：带背景数据块 "LSINU_ReadGUD_DB" 的“LSINU_ReadGUD”

在本例中，全局数据块 "DB_GUD" 包含要读取变量的名称、由调用填充的"VarToken"的结构和用于获取读取值的字符串值。

```
DATA_BLOCK "DB_GUD"
VERSION : 0.1
NON_RETAIN
   VAR
   GUD_string : String
   Token_string : Struct
    SyntaxID : Byte;
    AreaUnit : Byte;
    ColIndex : Word;
    RowIndex : Word;
    BlockType : Byte;
    NumRows : Byte;
    S7Typ : Byte;
    NcLength : Byte;
   END_STRUCT;
Value_string : String
END_VAR
BEGIN
   GUD_string := '_SC_CONT_NAME';
END_DATA_BLOCK


// Test "LSINU_ReadGUD"
// string, channel GUD
"LSINU_ReadGUD_DB"(Req :="Tag_1", // Bool
     Addr := "DB_GUD".GUD_string,
     Area := 2,                   //channel GUD
     Unit := 1,                   // GUD from channel 1
     Index1 := 0,                 // _SC_CONT_NAME[0]
     Index2 := 0,
     CnvtToken := TRUE,
     VarToken := "DB_GUD".Token_string,
     Error => "Tag_2",            // Bool
```

```
    Done => "Tag_3",                    // Bool
    State => "Tag_4",                   // Word
    RD := "DB_GUD".Value_string);
```

在本 SCL 示例中， "LSINU_ReadGUD" 由设置 Tag_1 激活。该函数无错误地执行，即"Tag_2" = false, "Tag_3" = true, "Tag_4" = 0。“DB_GUD”中的“Token_string”结构由 "LSINU_ReadGUD" 填充，如下所示：

SyntaxID = 16#82

AreaUnit = 16#41

ColIndex = 16#001A

RowIndex = 16#0000

BlockType = 16#36

NumRows = 16#01

S7Typ = 16#13

NcLength = 16#8D

_SC_CONT_NAME[0] 中的值写入 "DB_GUD".Value_string。

写入 GUD 变量：带背景数据块 "LSINU_WriteVar_DB" 的“LSINU_WriteVar”

来自 "LSINU_ReadGUD" 的 GUD 变量令牌，参数：用“LSINU_WriteVar”写入“VarToken”，参数“Addr1”。

```
// Test "LSINU_WriteVar"
// 1x string
"LSINU_WriteVar_DB"(Req :="Tag_5",          // Bool
            NumVar := 1,
            Addr1 := "DB_GUD".Token_string,
            Line1 := 1                      // _SC_CONT_NAME[0]
            Addr2 := NULL,
            Addr3 := NULL,
            Addr4 := NULL,
            Addr5 := NULL,
            Addr6 := NULL,
            Addr7 := NULL,
            Addr8 := NULL,
```

```
Error => "Tag_6",                    // Bool
Done => "Tag_7",                     // Bool
State => "Tag_8",                    // Word
SD1 := "DB_GUD".Value_string,
SD2 := NULL,
SD3 := NULL,
SD4 := NULL,
SD5 := NULL,
SD6 := NULL,
SD7 := NULL,
SD8 := NULL;
```

在本 SCL 示例中，“LSINU_WriteVar” 由设置“Tag_1”激活。使用 VarToken "DB_GUD ".Token_string"（之前由 "LSINU_ReadGUD" 填充）。该函数无错误地执行，
即"Tag_2" = false, "Tag_3" = true, "Tag_4" = 0。"DB_GUD".Value_string 中的值写入变量 SC_CONT_NAME[0]。

说明

参数索引

用于 "LSINU_ReadGUD"时，参数 index1 从 0 开始读取 SC_CONT_NAME[**0**]。
而参数 Line<n> 从 1 开始，以便写入 SC_CONT_NAME[**0**]。

12.6.8 LSINU_ReadVar

12.6.8.1 LSINU_ReadVar

说明

组态所需的数据类型不包含在库基本程序升级版中。必须将这些数据类型从“SINUMERIK ONE PLC Basic Program V6.x”库复制到 TIA Portal 项目中（命名约定：“LBP_type*”）。

功能

功能块 "LSINU_ReadVar" 从 NC 区域读取变量。

请求读取 NC 变量

以正向信号边沿调用“LSINU_ReadVar””：参数“Req”= 0 → 1

NC 变量的 S7 名称：参数“Addr1”到“Addr8”=<DB name>.<S7 name>。未使用的地址必须分配“ZERO”。

完成读取请求

读取请求成功完成：参数“NDR”== 1。

读取请求因错误而终止：参数 "Error"== 1，"State" 的错误原因

要求

已生成访问 NC 变量的数据块，如 "ncvar [DB120]".。另请参见 生成数据块以访问 NC 变量 (页 1321)。

在此数据块中，NC 变量在参数“Addr<x>”中指定，如 Addr1: = "ncvar".<NC-Variable>

一般条件

- "LSINU_ReadVar" 具有多实例能力。
  每次调用 "LSINU_ReadVar" 都必须从用户区分配一个单独的实例或背景数据块。
- 读取**通道专用**变量后，在调用 "LSINU_ReadVar" 时，不能通过“Addr1”到“Addr8”寻址来自不同通道的变量。

说明

错误案例

读取来自不同通道的变量时，会输出一条故障消息：

- "Error" == TRUE
- "State" == W#16#02

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

变量寻址

对于某些 NC 变量，需要选择“区域号”和/或 NC VAR 选择器中的“行”或“列”。

可选择基本类型，即“区域号”、“行”和“列”预分配“0”。在 "LSINU_ReadVar" 中检查 NC VAR 选择器指定的“区域号”、“行”和“列”的值是否为“0”如果 NC VAR 选择器值 == “0”，则采用相应的参数值。为此，必须在调用 "LSINU_ReadVar" 之前写入参数“Unit<n>”、“Column<n>”和“Line<n>”(<n> = 1, 2, ... - 8)

以下示例说明了如何通过 R 参数使用“区域号”、“行”和“列”。每个通道都有 100 个 R 参数可用。若要在其 PLC 程序中读取其中几个参数，或在编程时不清楚要在运行期间实际读取哪些 R 参数，借助 NC Var 选择器为每个 R 参数生成参考条目会非常耗时。相反，只为变量 R 参数“基本类型”创建一个条目。

如果在 NC VAR 选择器中，“1”设为“区域号”且“0”指定为“行”，则为通道 1 的所有 R 参数创建一个条目。行参数指定调用 "LSINU_ReadVar" 时要读取的 R 参数。

如果在 NC VAR 选择器中，“区域号”和“行”指定为“0”，则为所有通道中的所有 R 参数创建一个条目。“单元”参数指定通道，“行”参数指定调用 "LSINU_ReadVar" 时要读取的 R 参数的索引。请注意，值 Index +1 必须指定为“行”参数：

来自通道 1 的 R 参数 R[0]：单元 = 1，行 = 1

来自通道 2 的 R 参数 R[10]：单元 = 2，行 = 11

更多说明，另请参见 NC VAR 选择器的在线帮助。

表格 12-2　参数匹配

“LSINU_ReadVar”参数	**NC VAR 选择器**
Unit<n>	区域号
Column<n>	列
Line<n>	行

来自不同区域的变量可在调用中进行如下组合：

表格 12-3　调用组合

调用	**区域**				
1	C[1]	N	B	A	T
2	C[2]	N	B	A	T
与示例中显示的调用 1 和调用 2 相同的规则适用于通道 3 到 10： A = 轴特定数据，区域号对应于轴编号 B = 方式组数据，区域号对应于轴编号 C = 通道数据，区域号对应于轴编号 N = NC 数据，与区域号无关（不能在 NC VAR 选择器中更改） T = 刀具管理数据，区域号对应于轴编号					

说明

同时读取多个“字符串”类型的变量时，可用变量的数目可能少于八个，具体取决于字符串长度。这是因为一次调用最多可传输 230 字节。未读取任何内容并输出以下错误消息：

- "Error" == TRUE
- "State" == W#16#03

形式参数的描述

参数	类别	类型	取值范围	含义
Req	I	BOOL	---	任务以正向信号边沿开始
NumVar	I	INT	1 ... 8	要读取的变量数目："Addr1" - "Addr8"
Addr1 Addr2 ... Addr8	I	类型	<DB name>.<S7 name>	来自 **NC Var 选择器**的变量标识符
Unit1 Unit2 ... Unit8	I	BYTE	---	区域地址，可选用于变量寻址
Column1 Column2 ... Column8	I	WORD	---	列地址，可选用于变量寻址
Line1 Line2 ... Line8	I	WORD	---	行地址，可选用于变量寻址
Error	O	BOOL	---	TRUE：任务否定应答或任务无法执行
NDR	O	BOOL	---	TRUE：任务成功执行。数据可用
State	O	WORD	---	请参见“错误代码”段落
RD1 RD2 ... RD8	I/O	类型	---	读取数据的目标区域， 例如与要读取变量具有相同数据类型的 DB 区或位存储器

错误代码

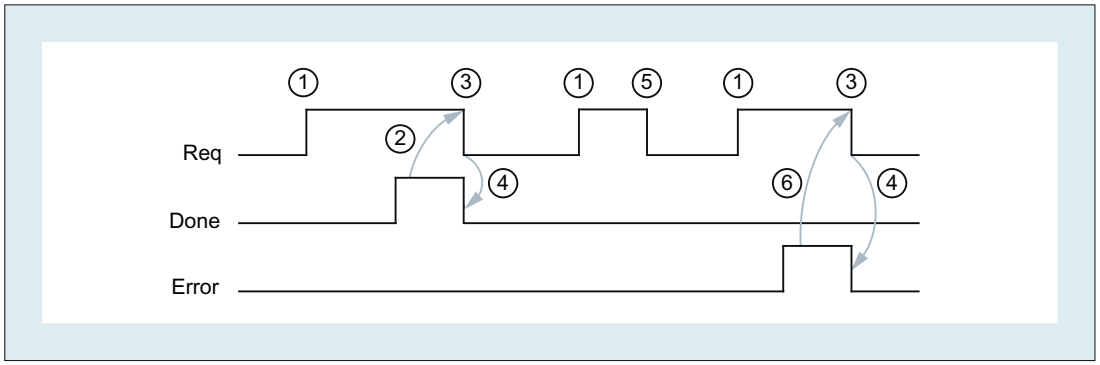
NC 变量

State		含义	说明
高位字节 1)	低位字节		
1 - 8	1	访问错误	无法访问此变量。 可能的原因： • NCK 中不存在变量（可能必须先设置机床数据） • “行”参数错误（0 只可用于某些变量，对于 Word 变量，该参数必须是偶数）
0	2	任务出错	任务中的变量编译错误
0	3	否定应答，任务不可执行	内部错误，可能的补救措施： • 检查任务数据 • NC 复位 • 一次调用中的字符串变量较少
1 - 8	4	可用的本地用户内存不足	读取变量的数据类型大于“RD1”-“RD8”中指定的数据类型
1 - 8	5	格式转换错误	变量类型转换时出错
0	6	FIFO 已满	必须重复任务，因为队列已满
1 - 8	10	变量寻址时出错	“单元”或“列”/“行”包含值 0
0	11	变量地址无效	检查“地址”（或变量名称）、“区域”、“单元”
0	12	NumVar == 0	检查参数“NumVar”
1) 高位字节 > 0 ⇒ 发生错误的变量的编号			

按照以下步骤读取 NC 变量：

- 使用 NC VAR 选择器选择变量
- 将选定变量保存在 *.VAR 文件中。
- 创建 STL 源
- 通过将 STL 源集成为 TIA Portal 项目中的外部源，创建具有相关地址数据的数据块
- 用户可更改数据块的名称及其中包含的 S7 名称
- "LSINU_ReadVar" 的参数化

脉冲图



① 用户：设置请求：Req = 0 → 1

② "LSINU_ReadVar":读取成功完成：NDR == 1

③ 用户：如果 NDR == 1，重置请求：Req = 1 → 0

④ "LSINU_ReadVar":重置任务确认：NDR == 0

⑤ 用户：如果 NDR == 0 且 Error == 0，不允许重置请求 Req = 1 → 0

⑥ "LSINU_ReadVar":读取完成但出现错误：Error == 1
用户：重置请求，如果 Error == 1 且 Req = 0，根据状态进行可能的进一步错误处理

调用示例

读取三个机床数据，它们的地址数据存储在 "ncvar [DB120]" 中。

数据规范

使用“NC-Var-Selector”工具选择数据并从中生成 "ncvar [DB120]" 数据块。另请参见 生成数据块以访问 NC 变量。

区域	程序块	名称	类型	编号	字节	S7 符号
C[1]	M	MD20070 $MC_AXCONF_MACHAX_USED[1]	CHAR	20070	1	C1_M_AXCONF_MACHAX_USED1
C[1]	M	MD20070 $MC_AXCONF_MACHAX_USED[2]	CHAR	20070	1	C1_M_AXCONF_MACHAX_USED2
N	M	MD10000 $MN_AXCONF_MACHAX_NAME_TAB[1]	STRING	10000	1	N_M_AXCONF_MACHAX_NAME_1

数据快 "ncvar" 必须传输到 PLC。读取值在此处写入 "DB_Read"，必须为正确的类型执行此操作。这意味着，要读取的 CHAR 值必须写入 "DB_Read" 中 CHAR 类型的数据区域。

使用背景数据块 "LSINU_ReadVar" 参数化 “LSINU_ReadVar_DB”：

```
/  Test "LSINU_ReadVar"
/
/  2x char, 1x string
/
"LSINU_ReadVar_DB"(Req :="Tag_1",          // Bool
           NumVar := 3,
           Addr1 := "ncvar".C1_M_AXCONF_MACHAX_USED1,
           Addr2 := "ncvar".C1_M_AXCONF_MACHAX_USED2,
           Addr3 := "ncvar".N_M_AXCONF_MACHAX_NAME_1,
           Addr4 := NULL,
           Addr5 := NULL,
           Addr6 := NULL,
           Addr7 := NULL,
           Addr8 := NULL,
           Error => "Tag_2",               // Bool
           NDR => "Tag_3",                 // Bool
           State => "Tag_4",               // Word
           RD1 := "DB_Read".CharValue[0],
           RD2 := "DB_Read".CharValue[1],
           RD3 := "DB_Read".StringValue,
           RD4 := NULL,
           RD5 := NULL,
           RD6 := NULL,
           RD7 := NULL,
           RD8 := NULL;
```

在本 SCL 示例中，“LSINU_ReadVar”由设置 Tag_1 激活。

该函数无错误地执行，即"Tag_2" = false, "Tag_3" = true, "Tag_4" = 0。

读取值以正确的类型写入 "DB_Read"。

示例：变量寻址

从通道 1 读取两个 R 参数，其地址数据作为“基本类型”存储在数据块“ncvar”中。

R 参数编号通过参数“Line<x>”进行参数化。

数据块 "ncvar" 中的 R 参数定义由“NC VAR 选择器”创建：

```
C1_RP_rpa0_1:
   STRUCT
   SYNTAX_ID:            BYTE := B#16#82;
   area_and_unit :       BYTE := B#16#41;
   column :              WORD := W#16#1;
   line :                WORD := W#16#0;        // variable addressing
   block type :          BYTE := B#16#15;
   NO.OF LINES :         BYTE := B#16#1;
   type :                BYTE := B#16#F;
   length :              BYTE := B#16#8;
   END_STRUCT;


// Test "LSINU_ReadVar"
// R Parameter
   "LSINU_ReadVar_DB"(Req := "Tag_1",          // Bool
               NumVar := 2,
               Addr1 := "ncvar".C1_RP_rpa0_0,
               Line1 := 1                      // Reading R0
               Addr2 := "ncvar".C1_RP_rpa0_1,
               Line3 := 2                      // Reading R1
               Addr3 := NULL,
               Addr4 := NULL,
               Addr5 := NULL,
               Addr6 := NULL,
               Addr7 := NULL,
               Addr8 := NULL,
               Error => "Tag_2",               // Bool
               NDR => "Tag_3",                 // Bool
               State => "Tag_4",               // Word
               RD1 := "DB_Read".RealValue[0],
               RD2 := "DB_Read".RealValue[1],
               RD3 := NULL,
               RD4 := NULL,
```

```
RD5 := NULL,
RD6 := NULL,
RD7 := NULL,
RD8 := NULL;
```

在本 SCL 示例中，“LSINU_ReadVar”由设置“Tag_1”激活。该函数无错误地执行，即"Tag_2" = false, "Tag_3" = true, "Tag_4" = 0。读取值以正确的类型写入 "DB_Read"。

数据类型分类

NC 数据类型	S7 数据类型
DOUBLE	LREAL
FLOAT	REAL
LONG	DINT
INTEGER	DINT
UINT_32	DWORD
INT_16	INT
UINT_16	WORD
UNSIGNED	WORD
CHAR	CHAR or BYTE
STRING	STRING
BOOL	BOOL
DATETIME	DATE_AND_TIME

示例

想要不调整格式便能读取类型为 DOUBLE 的 NC 变量，必须在目标区域“RDx”（如："DB_Read".RealValue）中指定类型为 REAL 的数据区域。

12.6.8.2 生成用于访问 NC 变量的数据块

功能块“LSINU_ReadVar” (页 1313)和“LSINU_WriteVar” (页 1331)允许用户程序访问 NC 变量。为此，这些功能块需要相应的结构化数据块。

可通过两种方式生成数据块。优选通过 TIA Portal 生成。

在 TIA Portal 中创建数据块

要生成用于访问 NC 变量的数据块，请执行以下操作：

1. 在项目导航中，添加一个新的用户自定义变量表，路径：“CNC_1 > NCK > 配置的 NC 变量”。
2. 将所需条目从“可用 NC 变量列表”复制到用户自定义变量表中。
3. 右键单击用户自定义变量表，并选择“创建 PLC 数据块”(Create PLC data block) 命令。

详细说明参见 PLC 手册，路径：NC 变量 > 配置 NC 变量。

使用“NC-VAR-Selector”工具创建数据块

要生成用于访问 NC 变量的数据块，请执行以下操作：

1. 打开“NC-VAR-Selector”工具。
2. 使用“插入项目”按钮选择相关的 NC 变量。
3. 按“生成”按钮可生成 STL 源，例如 "ncvar.awl"。
4. 在 TIA Portal 的项目导航中，打开文件夹“外部资源”。
5. 双击条目“添加新的外部文件”。
6. 选择带生成的 STL 源的文件，例如 "ncvar.awl".
   使用“NC-VAR-Selector”生成的数据块作为外部源导入。
7. 为导入的源选择快捷菜单项“从源生成程序块”。
   例如，可使用“程序块”文件夹中的导入源生成名为 "DB120 [DB120]" 的数据块。
8. 为生成的数据块分配一个符号名，例如 "ncvar".

12.6.9 LSINU_TransferSelData

功能

程序块“LSINU_TransferSelData”“可在 PLC 和 NC 之间交换数据。无论何时调用程序块，数据都会立即传输，与循环数据交换无关。

通过参数“使能”= TRUE 调用程序块来激活数据传输。

该程序块提供以下功能：

- 快速数据交换 NC → PLC
- 快速数据交换 PLC → NC

说明

"LSINU_TransferSelData"仅支持功能 3 和 4（Funct := 3 和 Funct := 4）。对于不支持的功能，将输出 ErrCode = 1 的错误。

调用时间

- 如需要，在 PLC 循环中，如在主 OB 中

形式参数的描述

信号	类别	类型	取值范围	含义
Enable	I	BOOL	---	TRUE：传输数据
Funct	I	BYTE	3 ... 4	3：快速数据交换，读取
				4：快速数据交换，写入
S7Var	I	VARIANT	---	取决于"Funct"
IVAR1	I	INT	---	取决于"Funct"
IVAR2	I	INT	---	取决于"Funct"
Error	O	BOOL	---	TRUE：已出错
ErrCode	O	INT	---	1："Funct"无效 否则：取决于"Funct"

12.6.9.1 功能 3、4：快速数据交换 PLC-NC

简介

提供一个单独的内部数据区域，用于在 NC 和 PLC 之间快速交换信息。此接口包含 10240 字节。接口中的数据结构可自由定义。

可使用 NC 程序访问界面中的数据，如下所示：

$A_DBB[x] 代表字节　Offset x=0,1,2,3 ... 10239
$A_DBW[x] 代表字　Offset x=0,2,4,6 ... 10238
$A_DBD[x] 代表 DWord　Offset x=0,4,8,12 ... 10236
$A_DBR[x] 代表 Real　Offset x=0,4,8,12 ... 10236

所有这些变量都使用相同的存储区域。索引"x"表示数据区域中的偏移地址（0 到 10239）。

相同的偏移用于从 PLC 访问。

对于从 PLC 访问，如果只传输一个值，建议使用基本数据类型 BYTE、CHAR、WORD、INT、DWORD、DINT 或 REAL；如果传输多个值，则建议使用这些数据类型的数组（...的数组）。原则上，数据接口能够以任何方式构建 (STRUCT)。但在这种情况下，数据将始终按字节传输。必须考虑 PLC 程序中由对齐产生的任何填充字节以及包含 SWAP 命令的必要性。参见示例 3。

信号量

信号量字节可用于确保数据传输期间的一致性。在此概念中，数据在完全写入之前无法由对方读取。

信号量字节位于同一数据区域（0 到 10239）。因此，不得在用于传输数据的区域之外。PLC 上的读取操作必须使用与 NC 上关联写入操作相同的信号量字节。PLC 上的写入操作必须使用与 NC 上关联写入操作相同的信号量字节。

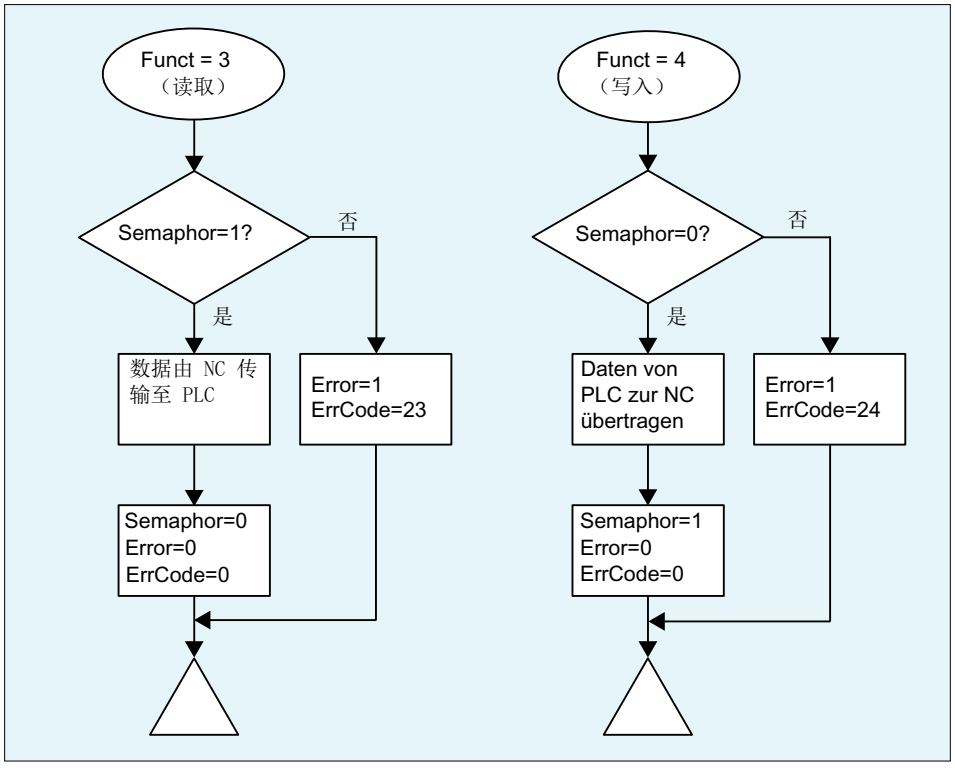
此外，传输 NC → PLC 和传输 PLC → NC 必须使用单独的信号量字节。各种读写访问操作的可用信号量字节数目仅受数据区域大小的限制。

该程序块在读取或写入操作之前检查信号量字节，并在读取或写入操作完成时将其写入。

信号量机制必须在 NC 程序中编程（见下文）。

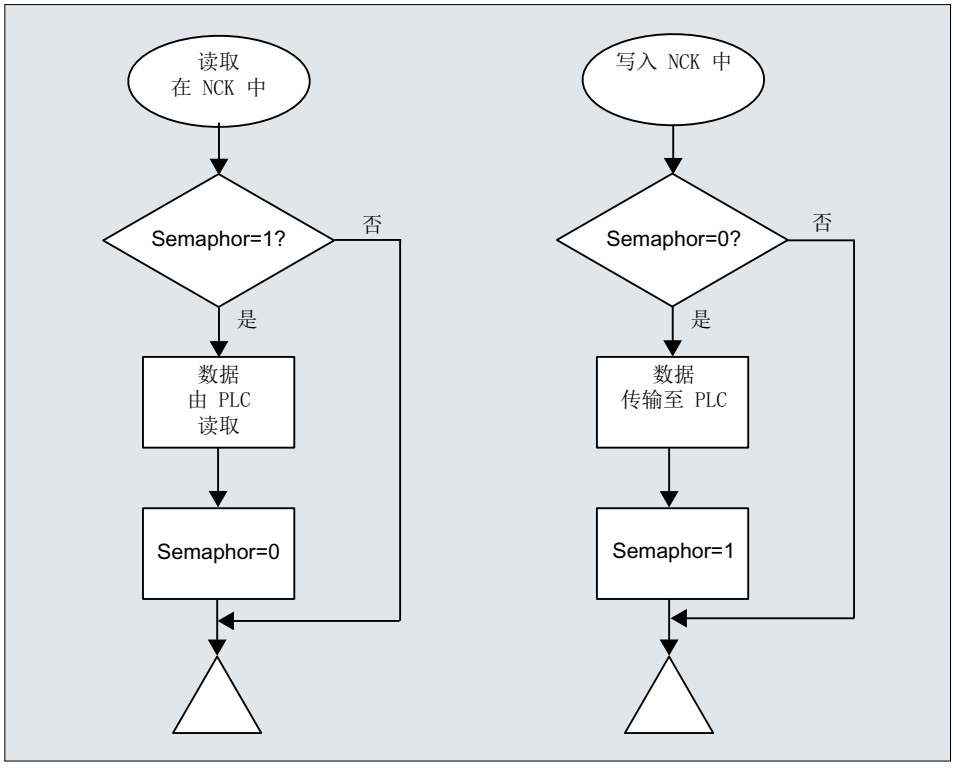
在 PLC 中使用信号量

该程序块将控制并设置信号量字节。



在 NC 中使用信号量

信号量字节的控制和设置必须明确执行。



变量值范围

信号	类别	类型	取值范围	含义
Enable	I	BOOL	---	TRUE：传输数据
Funct	I	BYTE	3, 4	3：快速数据交换，读取
				4：快速数据交换，写入
S7Var	I	类型	---	源数据区域（针对"Funct"：= 4） 目标数据区域（针对"Funct"：= 3）
IVAR1	I	INT	0 ... 10239	偏移
IVAR2	I	INT	-1, 0 ... 10239	信号量字节偏移量 -1:无信号量传输
Error	O	BOOL	---	TRUE：已出错

信号	类别	类型	取值范围	含义
ErrCode	O	INT	---	20:Word/DWord/Real 的对齐错误
				21：非法移位偏移量
				22:非法信号量字节
				23:无可供读取的新数据（信号量）
				24：无法写入新数据（信号量）
				26:S7Var 数据区域无效 (ZERO)
				28:S7Var 指向数据区域 > 10240 字节
				30:S7Var 显示无效数据区域 (<> ZERO)

在大端字节序和小端字节序之间切换

对于以下数据类型，如果此类变量在 LSINU_TransferSelData 中直接参数化，则在 LSINU_TransferSelData 中更改字节序。

- Word
- Array[] of Word
- Int
- Array[] of Int
- DWord
- Array[] of DWord
- Dint
- Array[] of Dint
- Real
- Array[] of Real

对于其他类型、类型数组、结构等，字节序不会改变，如需要，必须在 PLC 用户程序中执行。

示例 1：读取

PLC 在存储区域读取偏移量 4 的 DWord，以进行快速数据交换。字节 0 用作信号量。

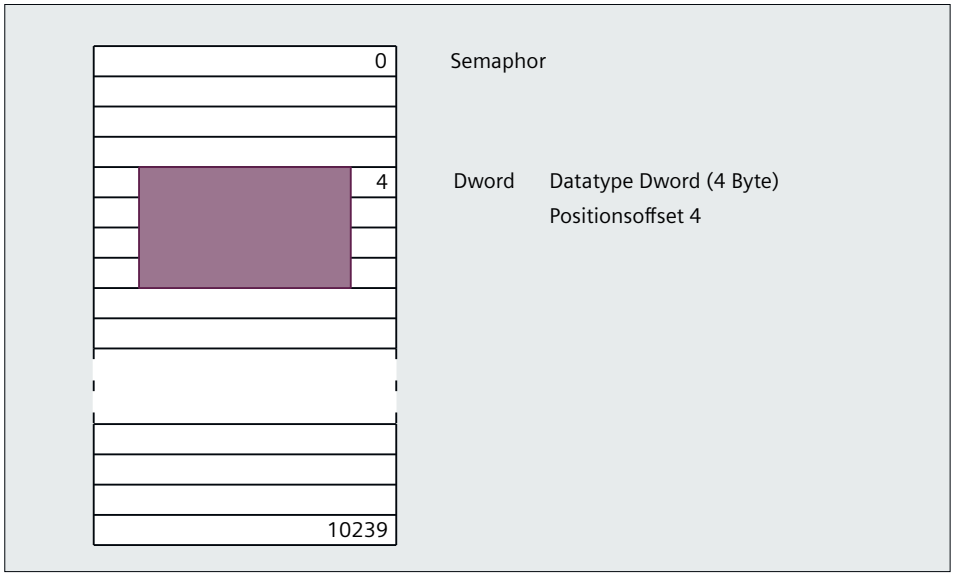


图 12-15　调用示例

PLC 编程

```
//  #tempData: DWord
//  #tempError: Bool
//  #tempStatus: Int
"LSINU_TransferSelData"(Enable := TRUE,          // 传输数据
                Funct := 3,                      // 从 NC 读取
                S7Var := #tempData,              // 写入数据的变量
                IVAR1 := 4,                      // 从偏移量 4 读取
                IVAR2 := 0,                      // 字节 0 中的信号量
                Error => #tempError,
                ErrCode => #tempStatus);
```

说明

示例 1 使用了数据类型 DWord。其中，字节序由 LSINU_TransferSelData 自动更改，这一点不必在 PLC 用户程序中实现（另请参见本章中字节序更改的示例 3）。

使用同步动作对 NC 进行编程

将数据写入 PLC。字节 0 也必须用作信号量。

```
ID=1 WHENEVER $A_DBB[0] == 0 DO $A_DBD[4] = $R1 $A_DBB[0] = 1
```

示例 2：写入

从偏移量 8 开始的 10 个字写入存储区域，以进行快速数据交换字节 1 用作信号量。

PLC 编程

```
//  #tempData: Array [1..10] of Word
//  #tempError: Bool
//  #tempStatus: Int
"LSINU_TransferSelData"(Enable := TRUE,          // 传输数据
                Funct := 4,                      // 写入 NC
                S7Var := #tempData,              // 读取数据的变量
                IVAR1 := 8,                      // 从偏移量 8 写入
                IVAR2 := 1,                      // 字节 1 中的信号量
                Error => #tempError,
                ErrCode => #tempStatus);
```

使用同步动作进行 NC 编程

从 PLC 读取数据，字节 1 也必须用作信号量。

```
ID=2 WHENEVER $A_DBB[1] == 1 DO $R1 = $A_DBW[8] $R2 = $A_DBW[10]
$A_DBB[1] = 0
```

示例 3：使用字节序更改的 STRUCT 读取

在用于快速数据交换的存储区域中，使用了读取用结构。PLC 用户程序 (SWAP) 中更改了字节序。

PLC 编程

```
//  #MyVar : Struct
//              first : DWord;
//              second : Word;
//              third : Byte;
//  END_STRUCT;
//  #tempError: Bool
//  #tempStatus: Int
"LSINU_TransferSelData"(Enable := TRUE,          // 传输数据
                Funct := 3,                      // 从 NC 读取
                S7Var := #MyVar,                 // 写入数据的变量
                IVAR1 := 0,                      // 从偏移量 0 读取
```

```
                IVAR2 := -1,                    // 无信号量
                Error => #tempError,
                ErrCode => #tempStatus);

IF NOT #tempError THEN                          // 读取后更改字节序
                MyVar.first := SWAP(MyVar.first);
                MyVar.second:= SWAP(MyVar.second);
END_IF;
```

示例 4：使用 STRUCT 读取

在用于快速数据交换的存储区域中，使用了读取用结构。

NC 值

$A_DBB[0] = 11, $A_DBB[1] = 22, $A_DBW[2] = B#1a2b, $A_DBD[2] = B#3c4d5e6f

PLC 编程

```
//  S7Var_Struct : Struct
//              S7Var_Byte : Byte;
//              S7Var_Word : Word;
//              S7Var_DInt : DWord;
//  END_STRUCT;
//  #tempError: Bool
//  #tempStatus: Int
"LSINU_TransferSelData"(Enable := TRUE,         // 传输数据
                Funct := 3,                     // 从 NC 读取
                S7Var := #S7Var_Struct,         // 写入数据的结构
                IVAR1 := 0,                     // 从偏移量 0 写入
                IVAR2 := -1,                    // 无信号量
                Error => #tempError,
                ErrCode => #tempStatus);
```

函数执行后，结构填充如下：

```
$A_DBB[1] → #S7Var_Struct.S7Var_Byte = 11
$A_DBW[2] → #S7Var_Struct.S7Var_Word = 16#2B1A
$A_DBD[4] → #S7Var_Struct.S7Var_DWord = 16#6F5E4D3C
```

$A_DBB[1] 不能在 PLC 中以符号方式寻址，但位于结构中，作为 #S7Var_Struct.S7Var_Byte 和 #S7Var_Struct.S7Var_Word 之间的填充字节。

要正确传输 $A_DBW[2] 和 $A_DBD[4]，#S7Var_Struct.S7Var_Word 和 #S7Var_Struct.S7Var_DWord 中必须更改字节序列：

```
#statS7Var_Struct.S7Var_Word :=
SWAP_WORD(#statS7Var_Struct.S7Var_Word);
#statS7Var_Struct.S7Var_DWord :=
SWAP_DWORD(#statS7Var_Struct.S7Var_DWord);
```

12.6.10 LSINU_WriteVar

说明

组态所需的数据类型不包含在库基本程序升级版中。必须将这些数据类型从“SINUMERIK ONE PLC Basic Program V6.x”库复制到 TIA Portal 项目中（命名约定：“LBP_type*”）。

功能

功能块 "LSINU_WriteVar" 从 NC 区域写入变量。

请求写入 NC 变量

使用正向信号边沿调用 "LSINU_WriteVar"，参数“Req”= 0 → 1

NC 变量的 S7 名称：参数“Addr1”到“Addr8”=<DB name>.<S7 name>。未使用的地址必须分配“ZERO”。

从中读取要写入的 NC 变量值的变量：

参数“SD1”到“SD8”。未使用的参数“SD<n>”必须分配“ZERO”。

完成写入请求

写入取请求成功完成：参数 "Done" == 1。

写入取请求完成但出现错误：参数 "Error"== 1，"State" 的错误原因

要求

已生成访问 NC 变量的数据块，如"ncvar [DB120]".。另请参见 “生成数据块以访问 NC 变量” (页 1321)。

在此数据块中，NC 变量在参数“Addr<x>”中指定，如 Addr1: = "ncvar".<NC-Variable>

一般条件

- "LSINU_WriteVar" 具有多实例能力。
  每次调用 "LSINU_WriteVar" 都必须从用户区分配一个单独的实例或背景数据块。
- 若要在没有密码的情况下定义机床数据和 GUD，要访问数据的保护级别必须重新定义为最低级别。
  参考资料：
  – NC 编程手册；“工作准备”部分
- 写入**通道专用**变量时，当使用“Addr1”到“Addr8”调用 "LSINU_WriteVar"时，不允许对来自不同通道的变量进行寻址。

说明

错误案例
写入来自不同通道的变量时，会输出一条故障消息：
- "Error" == TRUE
- "State" == W#16#02

调用时间

- 每个实例的每个 PLC 循环一次，如在主 OB 中

变量寻址

对于某些 NC 变量，需要选择“区域号”和/或 NC VAR 选择器中的“行”或“列”。可选择基本类型，即“区域号”、“行”和“列”预分配“0”。在 "LSINU_WriteVar" 中检查 NC VAR 选择器指定的“区域号”、“行”和“列”的值是否为“0”如果 NC VAR 选择器值 == “0”，则采用相应的"LSINU_WriteVar"参数值。为此，必须在调用 "LSINU_WriteVar" 之前将参数“Unit<n>”、“Column<n>”和“Line<n>”写入为 <n> = 1, 2, ... 8。

有关基本类型使用的更多说明，另请参见 NC VAR 选择器的在线帮助。

"LSINU_WriteVar" 参数	NC VAR 选择器
Unit<n>	区域号
Column<n>	列
Line<n>	行

来自不同区域的 NC 变量可在调用中进行如下组合：

调用	区域				
1	C[1]	N	B	A	T
2	C[2]	N	B	A	T
与示例中显示的调用 1 和调用 2 相同的规则适用于通道 3 到 10。 A = 轴特定数据，区域号对应于轴编号 B = 方式组数据，区域号对应于轴编号 C = 通道数据，区域号对应于轴编号 N = NC 数据，与区域号无关（不能在 NC VAR 选择器中更改） T = 刀具管理数据，区域号对应于轴编号					

说明

同时写入多个“字符串”类型的变量时，可用变量的数目可能少于八个，具体取决于字符串长度。

这是因为一次调用最多可传输 230 字节。未写入任何内容并输出以下错误消息，例如：

- "Error" == TRUE
- "State" == W#16#060F（第六个变量出错，用户数据过长）

形式参数的描述

参数	类别	类型	取值范围	含义
Req	I	BOOL	---	任务以正向信号边沿开始
NumVar	I	INT	1 ... 8	要读取的变量数目："Addr1" - "Addr8"
Addr1 Addr2 ... Addr8	I	类型	<DB name>. <S7 name>	来自 **NC Var 选择器**的变量标识符
Unit1 Unit2 ... Unit8	I	BYTE	---	区域地址，可选用于变量寻址

参数	类别	类型	取值范围	含义
Column1 Column2 ... Column8	I	WORD	---	列地址，可选用于变量寻址
Line1 Line2 ... Line8	I	WORD	---	行地址，可选用于变量寻址
Error	O	BOOL	---	TRUE：任务否定应答或任务无法执行
DONE	O	BOOL	---	TRUE：任务成功执行。
State	O	WORD	---	请参见“错误代码”段落
SD1 SD2 ... SD8	I/O	类型	---	要写入的数据， 例如与要写入变量具有相同数据类型的 DB 区或位存储器

错误代码

NC 变量

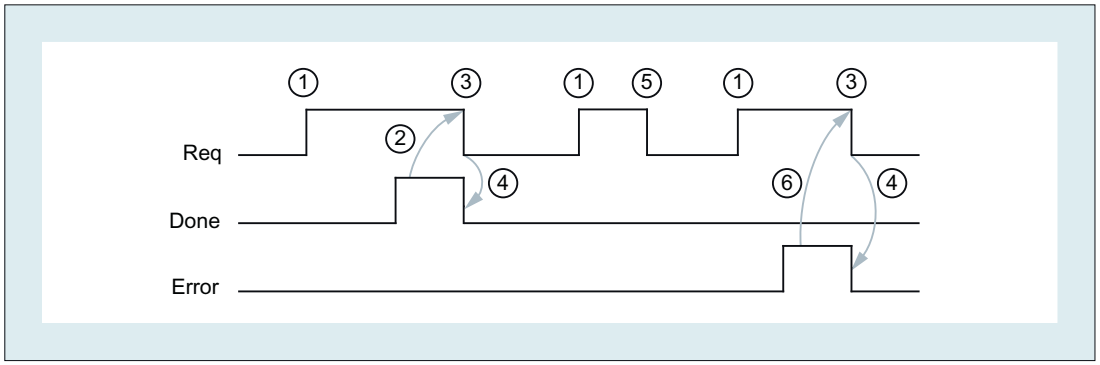
State		含义	说明
高位字节 [1)]	低位字节		
1 - 8	1	访问错误	无法访问此变量。 可能的原因： • NCK 中不存在变量（可能必须先设置机床数据） • “行”参数错误（0 只可用于某些变量，对于 Word 变量，该参数必须是偶数） • 此时即尝试写入无效值
0	2	任务出错	任务中的变量编译错误
0	3	否定应答，任务不可执行	内部错误，可能的补救措施： • 检查任务 • NC 复位

State		含义	说明
高位字节 1)	低位字节		
1 - 8	4	数据区域或数据类型不匹配或字符串为空	检查"SD1"-"SD8"中的数据
1 - 8	5	格式转换错误	变量类型转换时出错
0	6	FIFO 已满	必须重复任务，因为队列已满
1 - 8	10	变量寻址时出错	"单元"或"列"/"行"包含值 0
0	11	变量地址无效或变量为只读模式	检查"地址"（或变量名称）、"区域"、"单元"
0	12	NumVar == 0	检查参数"NumVar"
1 - 8	15	用户数据过长	补救措施：每个任务写入更少的变量或使用更短的字符串变量
1) 高位字节 > 0 ⇒ 发生错误的变量的编号			

组态步骤

要写入 NC 变量，需要执行与读取 NC 变量相同的组态步骤。有助于将所有要读取或写入的 NC 变量的地址数据存储在 DB 中。

脉冲图



① 用户：设置请求，Req = 0 → 1

② "LSINU_WriteVar":写入成功完成，Done = 1

③ 用户：如果 Done == 1，重置请求：1 → 0

④ "LSINU_WriteVar":重置任务确认，Done = 0

⑤ 用户：如果 Done == 0 且 Error == 0，不允许重置请求 Req = 1 → 0

⑥ "LSINU_WriteVar":写入完成但出现错误，Error = 1
用户：重置请求，如果 Error == 1 且 Req = 0，根据状态进行可能的进一步错误处理

调用示例

写入三个机床数据，它们的地址存储在 "ncvar [DB120]" 中。

数据规范

使用“NC-Var-Selector”工具选择数据并从中生成 "ncvar [DB120]" 数据块。另请参见 “生成数据块以访问 NC 变量” (页 1321)。

区域	程序块	名称	类型	S7 名称
C[1]	RP	rpa[5]	REAL	C1_RP_rpa5_1
C[1]	RP	rpa[11]	REAL	C1_RP_rpa11_1
C[1]	RP	rpa[14)	REAL	C1_RP_rpa14_1

数据快 "ncvar" 必须传输到 PLC。要写入的值从 "DB_Write" 读取，必须为正确的类型执行此操作。这意味着，要写入的 REAL 值必须从 "DB_Write" 中 REAL 类型的数据区域读取。

使用背景数据块 LSINU_WriteVar_DB 调用和参数化"LSINU_WriteVar" ：

在本 SCL 示例中，"LSINU_WriteVar" 通过设置“Tag_1”使用背景数据块 LSINU_WriteVar_DB 激活。该函数无错误地执行，即"Tag_2" = false, "Tag_3" = true, "Tag_4" = 0。“DB_Write”中的值写入相应的 R 参数。

使用背景数据块 "LSINU_WriteVar" 参数化 “LSINU_WriteVar_DB”：

```
// Test "LSINU_WriteVar"
// 3x real
"LSINU_WriteVar_DB"(Req :="Tag_1",          // Bool
            NumVar := 3,
            Addr1 := "ncvar".C1_RP_rpa5_1,
            Addr2 := "ncvar".C1_RP_rpa11_1,
            Addr3 := "ncvar".C1_RP_rpa14_1,
            Addr4 := NULL,
            Addr5 := NULL,
            Addr6 := NULL,
            Addr7 := NULL,
            Addr8 := NULL,
            Error => "Tag_2",               // Bool
            Done => "Tag_3",                // Bool
            State => "Tag_4",               // Word
            SD1 := "DB_Write".RealValue[0],
            SD2 := "DB_Write".RealValue[1],
            SD3 := "DB_Write".RealValue[2],
            SD4 := NULL,
            SD5 := NULL,
            SD6 := NULL,
            SD7 := NULL,
            SD8 := NULL;
```

在本 SCL 示例中，"LSINU_WriteVar" 由设置“Tag_1”激活。

该函数无错误地执行，即"Tag_2" = false, "Tag_3" = true, "Tag_4" = 0。

"DB_Write“ 中的值写入相应的 R 参数。

示例：变量寻址

与前面的示例相比，来自通道 1 的 R 参数的地址数据仅作为基本类型存储在 "ncvar" 中。R 参数编号通过参数 Line<n> 参数化。

"ncvar" 中的 R 参数定义由“NC VAR 选择器”创建：

```
C1_RP_rpa0_1:
STRUCT
   SYNTAX_ID:           BYTE := B#16#82;
   area_and_unit :      BYTE := B#16#41;
   column :             WORD := W#16#1;
   line :               WORD := W#16#0;       // variable addressing
   block type :         BYTE := B#16#15;
   NO.OF LINES :        BYTE := B#16#1;
   type :               BYTE := B#16#F;
   length :             BYTE := B#16#8;
   END_STRUCT ;


// Test "LSINU_WriteVar"
// 3x real, variabel adressiert
"LSINU_WriteVar_DB"(Req :="Tag_1",      // Bool
             NumVar := 3,
             Addr1 := "ncvar".C1_RP_rpa0_1,
             Line1 := 5,
             Addr2 := "ncvar".C1_RP_rpa0_1,
             Line2 := 11,
             Addr3 := "ncvar".C1_RP_rpa0_1,
             Line3 := 14,
             Addr4 := NULL,
             Addr5 := NULL,
             Addr6 := NULL,
             Addr7 := NULL,
             Addr8 := NULL,
             Error => "Tag_2",          // Bool
             NDR => "Tag_3",            // Bool
             State => "Tag_4",          // Word
             SD1 := "DB_Write".RealValue[0],
             SD2 := "DB_Write".RealValue[1],
             SD3 := "DB_Write".RealValue,
```

```
SD4 := NULL,
SD5 := NULL,
SD6 := NULL,
SD7 := NULL,
SD8 := NULL;
```

在本 SCL 示例中，"LSINU_WriteVar" 由设置“Tag_1”激活。

该函数无错误地执行，即"Tag_2" = false, "Tag_3" = true, "Tag_4" = 0。

"DB_Write" 中的值写入相应的 R 参数。

12.7 PLC 数据类型：全局库

12.7.1 概览

下表列举介绍 SINUMERIK PLC 基本程序的全局库 PLC 数据类型。将复制模板从全局库复制到项目中后，即可使用 PLD 数据类型。

名称	说明
LSINU_typeAlarmInternalData (页 1340)	用于存储内部数据以识别报警。
LSINU_typePowerrideConfig (页 1340)	为 Powerride 提供所有组态数据
LSINU_typePowerrideConfigPos (页 1342)	为 Powerride 的某个位置提供组态数据
LSINU_typePowerrideInputs (页 1343)	为“LSINU_Powerride”提供来自 Powerride 的输入数据
LSINU_typePowerrideOutputs (页 1344)	为“LSINU_Powerride”提供发送至 Powerride 的输出数据

12.7.2 LSINU_typeAlarmInternalData

描述

PLC 数据类型“LSINU_typeAlarmInternalData”用于保存内部数据，以识别报警。它是每个报警的一部分。

说明

不允许更改数据类型的内容。

12.7.3 LSINU_typePowerrideConfig

说明

PLC 数据类型“LSINU_typePowerrideConfig”为 Powerride 提供所有组态数据。过程中为各位置使用数据类型“LSINU_typePowerrrideConfig”。

pos100	
说明	代表补偿值 100 % 的开关位置 [1 - 36]
数据类型	USInt

posMax	
说明	代表最大补偿值的开关位置 [1 - 36]
数据类型	USInt

delay100	
说明	超过 100 % 补偿值的时间延迟。 达到“pos100”的开关位置后，启用时间延迟。延迟时间结束后，才能继续提升开关位置。也就是说，延迟时间未结束时，Powerride 不响应向右继续旋转。
数据类型	Time

quickTurnSpeedThreshold[1)]	
说明	用于快速调整的速度阈值（增量/毫秒） 值 = 0 代表禁用“在 Powerride 上快速左转 (> "quickTurnSpeedThreshold") 时设置位置 1”这项功能。
数据类型	Int

quickTurnIncThreshold[1)]	
说明	用于快速调整的增量阈值 值 = 0 代表禁用“在 Powerride 上快速左转 (> "quickTurnSpeedThreshold") 时设置位置 1”这项功能。
数据类型	Int

restTimeAfterQuickTurn[1)]	
说明	开始处理其他旋转运动之前，在位置 1 的停留时间。 此值规定快速向左转动旋转开关后设置的位置 1 保持多长时间。此值不可设为 0。
数据类型	Time

switchPos	
说明	定义开关位置 阵列中的每个数据集代表一个开关位置。您可为每个开关位置规定相应的属性（补偿值、LED 灯带颜色、振动模式）。
数据类型	LSINU_typePowerrideConfigPos 的阵列 [1 - 36]

[1)]“快速调整”提示信息

“quickTurnSpeedThreshold”、“quickTurnIncThreshold”和“restTimeAfterQuickTurn”三个参数可以实现“在 Powerride 上快速左转 (> "quickTurnSpeedThreshold") 时设置位置 1”这项功能。如果开关位置 1 被设备补偿值 0 %，即 switchPos[1].value = 0，可以在紧急情况下使各轴快速静止。“quickTurnSpeedThreshold”和“quickTurnIncThreshold”的参数值取决于所需的灵敏度和程序循环时间（比如 OB1）。值越小，灵敏度越高。如果“quickTurnSpeedThreshold”或“quickTurnIncThreshold”的参数值为零，说明功能被禁用。

如果不是上述情况，计算规则如下所示：

最后 30 个程序循环的时间和开关设置 (powerrideCounter) 始终保存在各 LSINU_Powerride 实例的静态区。如果在最后 30 个程序循环中，旋转开关向左转动的步数超过“quickTurnIncThreshold”，需要检查下列不等式：
步数差 *1000 / 时间差 [ms] > quickTurnIncThreshold。

如果不等式成立，会立刻输出开关位置 1，保持时间为在“restTimeAfterQuickTurn”中预设的时间。只有保持时间结束后，才能重新调整开关位置。

通过“restTimeAfterQuickTurn”和“quickTurnIncThreshold”两个参数的值，可以为“在 Powerride 上快速左转 (>"quickTurnSpeedThreshold") 时设置位置 1”这项功能规定在最多 30 个程序循环内，必须向左转动多少步，以及允许持续多长时间。

示例

循环时间约 20ms 时，quickTurnSpeedThreshold = 2，quickTurnIncThreshold = 2

12.7.4 LSINU_typePowerrideConfigPos

说明

PLC 数据类型“LSINU_typePowerrideConfigPos”为 Powerride 的某个位置提供组态数据。此数据类型在“LSINU_typePowerrrideConfig”中使用。

value	
说明	补偿值 [0 - 200] 此值代表用于某个特定开关位置的补偿值（倍率系数），单位：%。
数据类型	USInt

ledColor	
说明	Powerride 上 LED 灯带的颜色，可以是颜色组合（参见 LED 色板 (页 1299)）
数据类型	USInt

vibrationIndex	
说明	振动模式 [0 - 7]，0 代表无振动 详细信息参见 *SINUMERIK ONE MCP 产品手册第 1 部分：MCP xxxx*，“调试”章节 “Powerride 振动模式”部分
数据类型	USInt

12.7.5 LSINU_typePowerrideInputs

说明

PLC 数据类型“LSINU_typePowerrideInputs”为“LSINU_Powerride”提供来自 Powerride 的输入数据。

buttonPressed	
说明	Powerride 开关的状态 TRUE：开关已按下
数据类型	Bool

powerrideCounter	
说明	转轮位置 [0 - 255]，模数 256，每一圈 36 个卡槽，向右转增加数值，向左转降低数值
数据类型	USInt

12.7.6 LSINU_typePowerrideOutputs

说明

PLC 数据类型"LSINU_typePowerrideOutputs"为"LSINU_Powerride"提供发送至 Powerride 的输出数据。

ledInfo									
说明	控制 Powerride LED 灯带的各个 LED，赋值 TRUE 时亮起相应的 LED。 **比特位分配**								
	字节	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
	0	-	-	-	-	-	-	-	LED 24
	1	LED 23	LED 22	LED 21	LED 20	LED 19	LED 18	LED 17	LED 16
	2	LED 15	LED 14	LED 13	LED 12	LED 11	LED 10	LED 9	LED 8
	3	LED 7	LED 6	LED 5	LED 4	LED 3	LED 2	LED 1	LED 0
数据类型	DWord								

scaleRed, scaleGreen, scaleBlue	
说明	LED 灯带的颜色，可以是颜色组合（参见 LED 色板 (页 1299)）
数据类型	Bool

buttonRed, buttonGreen, buttonBlue	
说明	开关的 LED 颜色，可以是颜色组合（参见 LED 色板 (页 1299)）
数据类型	Bool

vibrationPattern	
说明	振动模式 [0 - 7]，0 代表无振动 详细信息参见 *SINUMERIK ONE MCP 产品手册第 1 部分：MCP xxxx*，“调试”章节 “Powerride 振动模式”部分
数据类型	字节

12.8 错误代码和状态代码

下表列出了基本程序功能块的所有故障标识。

16#0000

ErrorID (W#16#...)	说明
0000	无错误

消息/信息/警告 16#7xxx

ErrorID (W#16#...)	说明
7000	不执行该功能
7001	等待 NCK 的空闲接口。
7002	已将任务或信息发送至 NCK，等待应答
7003	显示功能“SINU_ReadUserData”之一中的参数“userData”的数据存储区太小。
7004	由于存储空间不足，无法保存实例名称。
7005	已将任务或信息发送至 NCK，等待应答。
7101	警告：“buttonColor”无效
7102	报警：powerrideConfig.switchPos[].ledColor 无效
7103	报警：powerrideConfig.switchPos[].vibrationIndex 无效
7110	引导启动期间无有效的驱动数据集
7111	切换过程步骤 1
7112	切换过程步骤 2
7113	切换过程步骤 3
7114	切换过程步骤 4
7121	切换过程步骤 1 最后执行 (enable = FALSE)
7122	切换过程步骤 2 最后执行 (enable = FALSE)
7123	切换过程步骤 3 最后执行 (enable = FALSE)
7124	切换过程步骤 4 最后执行 (enable = FALSE)
7132	将记录发送到机床控制面板上
7133	等待机床控制面板的应答

故障 16#8xxx

ErrorID (W#16#...)	说明
8001	实例无效或者实例大小有误。
8002	参数指向一个过大的数据存储区。
8003	参数指向一个过大的数据存储区。
8004	所需的 NCK 对象未生效。
8005	该实例未注册。
8006	NCK 相关数据类型未知或版本太新
8007	未知的实例类型
8008	至少有一个已注册实例未请求更新数据。
8009	未调用初始化。
800a	系统软件不支持该模块。
800b	未找到具有指定名称的轴。
800c	不允许在没有激活命令的情况下向 NCK 发送信息。
800d	发送至 NCK 的任务或信息的组合无效。
800e	注册已完成，调用“SINU_CompleteRegistration”功能后不再允许注册。
800f	另一个基本程序已进行初始化。
8010	无 NCK 事件可用。
8011	访问数据存储区外。
8012	轴不是 PLC 轴。
8013	“alarmCome”和“alarmGo”同时以值 1 生效
8014	错误的组织块流程级
8015	离开消息，在其之前没有出现消息
8016	报警相同，同一报警类别内已经存在具有相同报警编号和报警索引的报警。
8017	在当前任务完成前开始的新任务。
8019	功能正在活跃状态，不能再次启动
8101	正“preset”脉冲沿，但“presetValue”[3] 大于 36
8102	正“preset”脉冲沿，但“presetValue”小于 1
811a	连接时出错：端口已使用或被系统预留
8110	所传输的轴不是主轴。
8111	无有效的驱动数据集 (<Axis>.drive.in.numberOfDriveDataSetsValid)
8116	机床控制面板未在指定时间内应答

ErrorID (W#16#...)	说明
8119	连接时出错：已创建使用该 ID 的连接
8201	参数“hmi”中的数据块名称不合规。
8202	显示功能“SINU_ReadUserData”之一中的参数“userData”的数据存储区太大。
8203	显示功能“SINU_ReadUserData”之一中的参数“userData”的数据存储区无效
8204	模式不具有唯一性，
8205	错误的“alarmNumber”
8206	错误的“uniqueAlarmId”
8207	错误的“alarmCounter”
8208	错误的报警类别
8209	中断编号无效
820a	参数过长
820b	无法找到/打开/创建/使用对象
820c	参数值无效
820d	所指定的数据存储区过小
8212	组态错误 [2]：“posMax”大于 36
8221	组态错误：“pos100”小于 1
8222	组态错误：“posMax”小于 1
8223	组态错误：“pos100”大于“posMax”
8231	组态错误：“switchPos[x].value”在一个或多个数据集中超出有效范围 (0 ... 200%)
8234	连接时出错：参数“interfaceId”中的接口 ID 错误
8235	连接时出错：参数“connectionId”中的连接 ID 错误
8236	参数“data”中的记录版本无效（主要信息）
8237	参数“record”中的记录编号无效
8238	针对 "record" :=128：针对按键的无效配置， 无效的按键索引输出到“diagnosis”输出端中。
8239	无效配置，检查参数
8240	参数为零
8241	参数为零
8242	发送时出错：机床控制面板地址错误
8243	操作组件无 IPv4 地址
8244	无法确定针对此操作组件的参数设置数据

ErrorID (W#16#...)	说明
8245	机床控制面板参数设置数据无效
8246	机床控制面板的参数化已被操作组件拒绝
8251	手轮 1 输入端上的错误类型
8252	手轮 2 输入端上的错误类型
8253	手轮 3 输入端上的错误类型
8254	手轮 4 输入端上的错误类型
8255	手轮 5 输入端上的错误类型
8256	手轮 6 输入端上的错误类型
8401	由于存储空间不足，无法向 HMI 传输所有实例名称。
8402	NCK 未启动，无循环运行。
8403	NCK 不再有运行迹象，不再有循环运行。
8404	无法加载 NCK 适配器。
8405	无法加载 NCK 适配器。
8409	已达到同时存在的报警的最大数量
8410	子程序启动时出错
8411	子程序执行时出错

内部故障 16#8601 - 16#87FF

ErrorID (W#16#...)	说明
8601 - 87FF	内部故障 - 请联系西门子。

缩写列表

A

A.1 缩略语列表

A	
A	输出
AFIS	Automatic Filter Switch: 自动切换滤波器
ASCII	American Standard Code for Information Interchange:美国信息互换标准码
ASIC	Application Specific Integrated Circuit:用户自行开发的专用集成电路
ASUP	异步子程序
AUTO	AUTOMATIC，一种运行方式
AUXFU	Auxiliary Function:辅助功能
AWL	指令列表

B	
BAG	运行方式组
BCD	Binary Coded Decimals:用二进制代码编码的十进制数
BICO	Binector Connector
BIN	Binary Files:二进制文件
BKS	基准坐标系
BM	运行信息
BO	Binector Output
BTSS	操作面板接口

C	
CLC	间隙控制
CNC	Computerized Numerical Control: Computerunterstützte numerische Steuerung
COM	Communication
CP	Communication Processor

C	
CPU	Central Processing Unit:中央处理器
CST	Configured Stop:配置停止

D	
DDS	Drive Data Set:驱动数据组
DIR	Directory:目录
DO	Drive Object
DRF	Differential Resolver Function:微分旋转变压器功能（手轮）
DRY	Dry Run:空运行进给
DWORD	双字（当前 32 位）

E	
E	输入
EES	Execution from External Storage
E/A	输入/输出
ESR	扩展的停止和退回
ETC	ETC 键“>”，用于在同一层菜单中扩展软键栏

F	
FDD	Feed Disable:进给禁止
FdStop	Feed Stop:进给停止
FIFO	First In First Out:存储器，工作无需地址说明，数据按存储的顺序读入
FM	故障信息
FUP	功能图（一种 PLC 编程方法）
FW	Firmware

G	
GEO	几何属性，例如几何值
GP	基本程序 (PLC)
GUD	Global User Data:全局用户数据

H	
HEX	十六进制数代号
HiFu	辅助功能
HMI	Human Machine Interface:SINUMERIK 操作介面
HSA	主轴驱动
HT	Handheld Terminal
HW	硬件

I	
IBN	调试
INC	Increment:增量尺寸
INI	Initializing Data:初始化数据
IPO	Interpolator

J	
JOG	Jogging:点动模式

K	
KOP	梯形图（一种 PLC 编程方法）

L	
LED	Light Emitting Diode:发光二极管
LMS	位置测量系统
LR	位置控制器

M	
Main	Main program:主程序 (OB1，PLC)
MCP	Machine Control Panel:机床控制面板
MD	机床数据

M	
MDA	Manual Data Automatic:手动数据输入
MDS	Motor Data Set:电机数据组
MELDW	信息字
MKS	机床坐标系
MM	Motor Module
MPF	Main Program File:主程序（NC）
MPI	Multi Point Interface
MSTT	机床控制面板

N	
NC	Numerical Control:带有程序段处理、运行范围等等的数控系统
NCU	Numerical Control Unit:NC 硬件单元
NCK	Numerical Control Kernel
NCSD	NC Start Disable
NST	接口信号
NV	零点偏移
NX	Numerical Extension:轴扩展模块

O	
OB	PLC 中组织块
OEM	Original Equipment Manufacturer
OP	Operation Panel:操作面板

P	
PCU	PC Unit:PC 主机（计算机元件）
PG	编程器
PI	程序实例
PLC	Programmable Logic Control:可编程逻辑控制器
PN	PROFINET
PO	Power On

P	
POS	位置/定位
PPO	Parameter Prozessdaten Objekt：通过 PROFIBUS–DP 和“转速可变驱动”传输时的循环数据报文
PPU	Panel Processing Unit ：基于面板的 CNC 控制系统（如 SINUMERIK 828D）的核心硬件
PROFIBUS	Process Field Bus:串行数据总线
PRT	程序测试
PTP	Point to Point:点到点
PZD	过程数据：PPO 的过程数据部分

R	
REF	返回参考点功能
REPOS	再定位功能
RESU	Retrace support
RID	Read In Disable
RP	R 参数，计算参数，预定义用户变量

S	
SA	同步动作
SBL	Single Block:单程序段
SBT	Safe Brake Test
SCC	Safety Control Channel
SCL	Structured Control Language
SD	Settingdatum：设定数据
SDI	Safe Direction
SERUPRO	Search-Run by Program Test:通过程序测试的程序段查找
SIC	Safety Info Channel
SKP	Skip:跳至零件程序段末尾
SLP	Safe Limited Position
SLS	Safely Limited Speed
SMI	Sensor Module Integrated

S	
SOS	Safe Operating Stop
SPF	Sub Program File:子程序（NC）
SS1	Safe Stop 1
SS2	Safe Stop 2
STO	Safe Torque Off
STW	控制字
SUG	砂轮外缘速度
SW	软件

T	
TCU	Thin Client Unit
TIA	Totally Integrated Automation
TM	Terminal Module (SINAMICS)
TO	Tool Offset:刀具补偿
TOA	Tool Offset Active:刀具补偿标识（文件类型）
TOFF	在线－刀具长度补偿
TRANSMIT	Transform Milling Into Turning:车床上铣削加工的坐标转换

U	
UP	子程序
USB	Universal Serial Bus

V	
VDI	NC 和 PLC 间的内部通讯接口

W	
WKS	工件坐标系
WPD	Work Piece Directory:工件目录
WZ	刀具
WZV	刀具管理

Z	
ZSW	（驱动）状态字

A.2 块、数据类型、指令和变量缩写

下列缩写词可在块名称、数据类型、指令和变量中使用。

A		
accel	acceleration	加速度
ack	acknowledge fault	应答故障
ack	confirm receipt	确认
ackWork	fault will be corrected	排除故障
amc	SINUMERIK Integrate Analyze MyCondition (AMC)	SINUMERIK Integrate Analyze MyCondition (AMC)
aMode	automatic mode (linked mode) (AUTO)	链式运行 (VK)
asup	asynchronous subprogram	异步子程序
at	advanced technology	-
auto	auto	AUTO
aux	auxiliary (e.g. auxiliary function)	帮助…（比如辅助功能等）

B		
bcs	basic coordinate system	基准坐标系
bp	basic program	基本程序

C		
ccw	counterclockwise	向左旋转
chan	channel	通道
clc	clearance control	间距控制
clg	centerless grinding	无心（未夹紧）磨削
comp	compensation	补偿

C		
con	condition	条件
config	configure / configuration	设置
cont	continuous	连贯，无缝
cpu	central processing unit	中央处理器
ctrl	control, controller	控制，控制器，闭环控制器
cw	clockwise	向右旋转

D		
dc	direct current	直流电
decel	deceleration	制动/延迟（空间）
dep	dependent	取决于
diag	diagnostics	诊断
dir	direction	方向
dp	decentralized periphery	分布式输入/输出设备
drf	differential resolver function	DRF 功能
dyn	dynamic	动态

E		
eg	electronic gearbox	电子齿轮箱
emStop	emergency stop	急停
enc	encoder	脉冲编码器
esr	extended stop and retract	扩展的停止和退回

F		
freq	frequency	频率
func	function	功能

G		
geo	geometry	几何尺寸
goRefPoint	goto reference point	回参考点

G		
gud	global user data	全局用户数据
gwps	grinding wheel peripheral speed	砂轮圆周速度 (SUG)

H		
hmi	human machine interface	人机界面
ht	handheld terminal	手动操作面板
hw	hardware	硬件
hyd	hydraulic	液压

I		
id	Identity	名称
in	incomming	输入
inc	increments	增量
incVar	increments variable	增量变量
info	information	信息

J		
jog	jog	自定义手动运行

M		
mag	magazine	刀库
max	maximum	最大值
mcp	machine control panel	机床控制面板
mcs	machine coordinate system	机床坐标系
mcsWcs	switch request MCS/WCS	机床坐标系/工件坐标系切换请求
mda	manual data automatic	手动输入数据
medOff	media off	媒体关
medOn	media on	媒体开
min	minimum	最小值
misc	miscellaneous	综合

M		
mmc	men-machine-communication	人机通信
mpi	multi point interface	多点式串行接口
mpp	machine pushbutton panel	机床按钮面板
ms	main-/sub-	主/从
msg	message	消息/通知

N		
nc(u)	numerical control (unit)	数控，包含整个 NC(U) 优先使用缩写
nck	numerical control kernel	数控核心，包含 NC(U) 的整个运动控制器部分
nok	not ok	不合格

O		
ob	organisation block	组织块
oem	original equipment manufacturer	原始设备制造商
ok	ok	合格
op	operator panel	前置操作面板
opUnit	operator unit	控制台
opUnitCom	operator unit communication	控制面板通信
oscill	oscillation	振动
out	outgoing	输出
ovr	override	倍率

P		
pc	pc	PC
plc	programmable logic controller	可编程逻辑控制器 (PLC)
pn	profinet	Profinet
pneu	pneumatic	气动
pos	position, positioning	位置，定位

P		
proc	procedure (verallgemeinert FB und FC)	程序（块）
prog	program	程序
protArea	protection area	保护区
ptp	point to point	点到点

R		
ref	reference point approach function	基准化功能
repos	repos	“重新定位”功能
resync	resynchronization	重新同步
rev	revolutional	旋转相关
rot	rotation, rotary	旋转，转动

S		
safety	Safety Integrated	Safety Integrated
sbr	Subroutine	子程序
serupro	Search–Run by Program Test	通过程序测试的程序段查找
sim	simulation	模拟
sinu	SINUMERIK	SINUMERIK
sla	safely limited acceleration	安全限制加速
slp	safely limited position	安全限制位置
sls	safely limited speed	Safely-Limited Speed
smi	sensor module integrated	集成式传感器模块
sMode	single mode (SM)	单独运行 (EB)
sto	safe torque off	安全转矩关闭
stopCyclEnd	stop after end of cycle	周期结束后停止
stopImm	immediate stop	急停
sw	software	软件
synAct	synchronous Action	同步动作
sync	synchronous, synchronized	同步，已同步

T		
tcs	tool coordinate system	刀具坐标系
tcu	thin client unit	Thin Client Unit： 薄型客户单元
toa	tool area	刀具管理区域
techno	technology	工艺
toffon	(programmable) tool offset	（可编程）刀具补偿或（可编程）刀具补偿偏移
toolMgmt	tool management	刀具管理
travers	traverse/traversing	移动/运行（轴）

V		
var	variable	可设定

W		
wcs	workpiece coordinate system	工件坐标系
wPiece	workpiece	工件

索引

<

<Axis>.basic, 136
<Axis>.basic.in, 137
<Axis>.basic.in.actToolBasePos, 147
<Axis>.basic.in.alarmActive, 139
<Axis>.basic.in.clampingToleranceExceeded, 141
<Axis>.basic.in.contTraversActive, 144
<Axis>.basic.in.disabled, 139
<Axis>.basic.in.dynBacklashCompActive, 147
<Axis>.basic.in.emergencyRetractionActive, 146
<Axis>.basic.in.enc1Active, 140
<Axis>.basic.in.enc1FreqLimitExceeded, 140
<Axis>.basic.in.enc1Restored, 140
<Axis>.basic.in.enc1Synchronized, 140
<Axis>.basic.in.enc2Active, 140
<Axis>.basic.in.enc2FreqLimitExceeded, 140
<Axis>.basic.in.enc2Restored, 141
<Axis>.basic.in.enc2Synchronized, 140
<Axis>.basic.in.fixedStopForceLimited, 146
<Axis>.basic.in.fixedStopReached, 146
<Axis>.basic.in.fixedStopRequest, 146
<Axis>.basic.in.followUpModeActive, 141
<Axis>.basic.in.handwheelAssignedNumber, 142
<Axis>.basic.in.handwheelDirInverted, 142
<Axis>.basic.in.handwheelOvrActive, 142
<Axis>.basic.in.inc10000Active, 144
<Axis>.basic.in.inc1000Active, 143
<Axis>.basic.in.inc100Active, 143
<Axis>.basic.in.inc10Active, 143
<Axis>.basic.in.inc1Active, 143
<Axis>.basic.in.incVarActive, 144
<Axis>.basic.in.indexAxisInPos, 145
<Axis>.basic.in.isPathAxis, 145
<Axis>.basic.in.isPosAxis, 145
<Axis>.basic.in.isStationary, 139
<Axis>.basic.in.jogFixedPointReachedNumber, 144
<Axis>.basic.in.jogFixedPointSelectedNumber, 144
<Axis>.basic.in.jogPosReached, 145
<Axis>.basic.in.jogToPosActive, 144
<Axis>.basic.in.lubricationPulseActive, 145
<Axis>.basic.in.measurementActive, 142
<Axis>.basic.in.moduloLimitSwitchActive, 144
<Axis>.basic.in.posAxisFeedrateValue, 146
<Axis>.basic.in.posCoarseReached, 141
<Axis>.basic.in.posCtrlActive, 141
<Axis>.basic.in.posFineReached, 141
<Axis>.basic.in.ready, 139
<Axis>.basic.in.reposDelayAck, 147
<Axis>.basic.in.reposOffsetActive, 146
<Axis>.basic.in.reposOffsetValid, 147
<Axis>.basic.in.revolutionalFeedrateActive, 142
<Axis>.basic.in.rotAxisInPos, 145
<Axis>.basic.in.speedCtrlActive, 141
<Axis>.basic.in.spindleModeActive, 145
<Axis>.basic.in.swCamsActive, 142
<Axis>.basic.in.traversMinusCommandActive, 143
<Axis>.basic.in.traversMinusRequestActive, 142
<Axis>.basic.in.traversPlusCommandActive, 143
<Axis>.basic.in.traversPlusRequestActive, 143
<Axis>.basic.in.velocityReduced, 146
<Axis>.basic.out, 137
<Axis>.basic.out.acceptExternalWorkOffset, 153
<Axis>.basic.out.clampingInProgress, 148
<Axis>.basic.out.contTravers, 150
<Axis>.basic.out.disable, 147
<Axis>.basic.out.disableParameterSetSwitchover, 154
<Axis>.basic.out.disableTraversKeys, 149
<Axis>.basic.out.dynBacklashComp, 154
<Axis>.basic.out.enableCtrl, 147
<Axis>.basic.out.enableModuloLimitSwitch, 151
<Axis>.basic.out.enableOvr, 148
<Axis>.basic.out.enc1Activation, 148
<Axis>.basic.out.enc2Activation, 148
<Axis>.basic.out.feedrateOvrFactor, 147
<Axis>.basic.out.feedStopSpindleStop, 149
<Axis>.basic.out.fixedFeedrateSelection, 153
<Axis>.basic.out.fixedStopAckReached, 153
<Axis>.basic.out.fixedStopEnable, 153
<Axis>.basic.out.fixedStopSensor, 153
<Axis>.basic.out.followUpMode, 148
<Axis>.basic.out.handwheelInvertDir, 149
<Axis>.basic.out.handwheelSelection, 149
<Axis>.basic.out.hwLimitSwitchMinus, 151
<Axis>.basic.out.hwLimitSwitchPlus, 151
<Axis>.basic.out.inc1, 150
<Axis>.basic.out.inc10, 150
<Axis>.basic.out.inc100, 150
<Axis>.basic.out.inc1000, 150
<Axis>.basic.out.inc10000, 150
<Axis>.basic.out.incVar, 150
<Axis>.basic.out.jogFixedPointSelection, 151
<Axis>.basic.out.jogToPos, 151
<Axis>.basic.out.limitSpeed, 153
<Axis>.basic.out.posCtrlParameterSetSelection, 153
<Axis>.basic.out.progTestEnableCtrl, 152

<Axis>.basic.out.progTestRequest, 152
<Axis>.basic.out.progTestSuppression, 152
<Axis>.basic.out.rapidTraversOvr, 149
<Axis>.basic.out.refPointApproachCam, 152
<Axis>.basic.out.refPointSelection, 152
<Axis>.basic.out.reposDelay, 152
<Axis>.basic.out.reposPtp, 152
<Axis>.basic.out.resetMovement, 148
<Axis>.basic.out.swCams, 148
<Axis>.basic.out.swLimitSwitch2Minus, 151
<Axis>.basic.out.swLimitSwitch2Plus, 151
<Axis>.basic.out.traversKeyMinus, 149
<Axis>.basic.out.traversKeyPlus, 149
<Axis>.config, 136
<Axis>.config.axisActive, 139
<Axis>.config.axisNumber, 139
<Axis>.config.registered, 139
<Axis>.coupling, 137
<Axis>.coupling.in, 138
<Axis>.coupling.in.accelReached, 172
<Axis>.coupling.in.accelThresholdReached, 171
<Axis>.coupling.in.correctionActive, 171
<Axis>.coupling.in.correctionDone, 171
<Axis>.coupling.in.followingSpindleActive, 169
<Axis>.coupling.in.gantryActive, 167
<Axis>.coupling.in.gantryCutOffLimitExceeded, 168
<Axis>.coupling.in.gantryLeadingActive, 168
<Axis>.coupling.in.gantrySyncOk, 168
<Axis>.coupling.in.gantrySyncRunReady, 168
<Axis>.coupling.in.gantryWarningLimitExceeded, 168
<Axis>.coupling.in.leadingSpindleActive, 169
<Axis>.coupling.in.maxAccelReached, 171
<Axis>.coupling.in.maxSpeedReached, 171
<Axis>.coupling.in.mcsCouplingActive, 169
<Axis>.coupling.in.mcsFollowingAxisModeActive, 169
<Axis>.coupling.in.mcsMirroringActive, 169
<Axis>.coupling.in.mcsOffsetChanged, 169
<Axis>.coupling.in.motionOverlaid, 170
<Axis>.coupling.in.msActive, 168
<Axis>.coupling.in.msCompCtrlActive, 168
<Axis>.coupling.in.msSpeedDifferenceCoarseReached, 169
<Axis>.coupling.in.msSpeedDifferenceFineReached, 169
<Axis>.coupling.in.processActive, 171
<Axis>.coupling.in.speedThresholdReached, 171
<Axis>.coupling.in.syncCoarse2Reached, 170
<Axis>.coupling.in.syncCoarseReached, 170
<Axis>.coupling.in.syncFine2Reached, 170
<Axis>.coupling.in.syncFineReached, 170
<Axis>.coupling.in.syncSpindleCtrlMode, 170
<Axis>.coupling.out, 138
<Axis>.coupling.out.enableFollowOverlay, 173
<Axis>.coupling.out.gantryDisableSync, 172
<Axis>.coupling.out.gantryStartSyncRun, 172
<Axis>.coupling.out.mcsCollisionProtection, 172
<Axis>.coupling.out.mcsDecouple, 172
<Axis>.coupling.out.msActivate, 172
<Axis>.coupling.out.msTorqueComp, 172
<Axis>.coupling.out.syncCorrection, 173
<Axis>.coupling.out.syncResetCorrection, 173
<Axis>.coupling.out.syncSpindleResync, 173
<Axis>.coupling.out.syncSpindleSyncDisable, 173
<Axis>.drive, 136
<Axis>.drive.in, 137
<Axis>.drive.in.autonomousMotionActive, 155
<Axis>.drive.in.classCAlarmActive, 154
<Axis>.drive.in.currentCtrlActive, 158
<Axis>.drive.in.driveCtrlActive, 158
<Axis>.drive.in.driveDataSetNumber, 155
<Axis>.drive.in.driveTestTraversRequested, 158
<Axis>.drive.in.enabled, 154
<Axis>.drive.in.esrDcVoltageAlarmActive, 157
<Axis>.drive.in.esrGeneratorOperationActive, 157
<Axis>.drive.in.esrSpeedBelowMin, 157
<Axis>.drive.in.heatSinkTemperatureWarning, 155
<Axis>.drive.in.holdingBrakeReleased, 155
<Axis>.drive.in.motorDataSetNumber, 155
<Axis>.drive.in.motorTemperatureWarning, 154
<Axis>.drive.in.numberOfDriveDataSets, 156
<Axis>.drive.in.numberOfDriveDataSetsValid, 156
<Axis>.drive.in.posCtrlParameterSetNumber, 157
<Axis>.drive.in.pulsesEnabled, 154
<Axis>.drive.in.rapidStopActive, 155
<Axis>.drive.in.ready, 154
<Axis>.drive.in.speedAtSetpoint, 156
<Axis>.drive.in.speedBelowMin, 156
<Axis>.drive.in.speedBelowThreshold, 156
<Axis>.drive.in.speedCtrlIntegratorDisabled, 155
<Axis>.drive.in.speedSetpointReached, 156
<Axis>.drive.in.torqueBelowThreshold, 156
<Axis>.drive.in.varSignalingFunc1Active, 157
<Axis>.drive.in.varSignalingFunc2Active, 157
<Axis>.drive.in.varSignalingFunc3Active, 157
<Axis>.drive.out, 137
<Axis>.drive.out.disableSpeedCtrlIntegrator, 158
<Axis>.drive.out.driveDataSetSelection, 159
<Axis>.drive.out.enableDriveTestTravers, 159
<Axis>.drive.out.enablePulses, 158
<Axis>.drive.out.motorChangeoverDone, 159
<Axis>.drive.out.motorDataSetSelection, 159
<Axis>.drive.out.rapidStop, 158

<Axis>.drive.out.releaseHoldingBrake, 158
<Axis>.drive.out.requestDriveCtrl, 159
<Axis>.interchange, 137
<Axis>.interchange.in, 138
<Axis>.interchange.in.assignedChanNumber, 181
<Axis>.interchange.in.assignedToPlc, 181
<Axis>.interchange.in.neutralAxis, 180
<Axis>.interchange.in.possible, 180
<Axis>.interchange.in.requested, 180
<Axis>.interchange.out, 138
<Axis>.interchange.out.assignToChanSelection, 181
<Axis>.interchange.out.assignToPlc, 181
<Axis>.interchange.out.requireSwitchOver, 181
<Axis>.oscillation, 137
<Axis>.oscillation.in, 138
<Axis>.oscillation.in.active, 163
<Axis>.oscillation.in.disabled, 163
<Axis>.oscillation.in.error, 163
<Axis>.oscillation.in.externalReversalActive, 164
<Axis>.oscillation.in.motionActive, 164
<Axis>.oscillation.in.sparkingOutActive, 164
<Axis>.oscillation.out, 138
<Axis>.oscillation.out.activateReversalPoint, 167
<Axis>.oscillation.out.externalReversal, 167
<Axis>.oscillation.out.selectReversalPoint, 167
<Axis>.oscillation.out.stopAtNextReversalPoint, 167
<Axis>.plcCtrl, 137
<Axis>.plcCtrl.in, 138
<Axis>.plcCtrl.in.plcCtrlActive, 173
<Axis>.plcCtrl.in.plcCtrlPermanent, 174
<Axis>.plcCtrl.in.resetDone, 174
<Axis>.plcCtrl.in.stopped, 173
<Axis>.plcCtrl.out, 138
<Axis>.plcCtrl.out.requestPlcCtrl, 174
<Axis>.plcCtrl.out.reset, 174
<Axis>.plcCtrl.out.resume, 174
<Axis>.plcCtrl.out.stop, 174
<Axis>.safety, 137
<Axis>.safety.in, 138
<Axis>.safety.in.brakeTestModeEnabled, 181
<Axis>.safety.in.infoCtrlChanActive, 181
<Axis>.safety.out, 138
<Axis>.safety.out.activateBrakeRampForSs2e, 182
<Axis>.safety.out.enableBrakeTestMode, 182
<Axis>.safety.out.setpointLimitSelection, 182
<Axis>.safetyCtrlChan, 137
<Axis>.safetyCtrlChan.out, 139
<Axis>.safetyCtrlChan.out.brakeSelection, 180
<Axis>.safetyCtrlChan.out.closeBrake, 179
<Axis>.safetyCtrlChan.out.externalBrakeClosed, 180
<Axis>.safetyCtrlChan.out.selectBrakeTest, 179
<Axis>.safetyCtrlChan.out.selectDirectionOfRotation, 180
<Axis>.safetyCtrlChan.out.selectTestSequence, 180
<Axis>.safetyCtrlChan.out.startBrakeTest, 179
<Axis>.safetyCtrlChan.out.testStop, 179
<Axis>.safetyInfoChan, 137
<Axis>.safetyInfoChan.in, 139
<Axis>.safetyInfoChan.in.acceptanceTestModeSelected, 178
<Axis>.safetyInfoChan.in.acceptanceTestSlpSelected, 178
<Axis>.safetyInfoChan.in.activeBrake, 178
<Axis>.safetyInfoChan.in.brakeTest, 178
<Axis>.safetyInfoChan.in.brakeTestActive, 178
<Axis>.safetyInfoChan.in.brakeTestCompleted, 179
<Axis>.safetyInfoChan.in.brakeTestResult, 178
<Axis>.safetyInfoChan.in.currentLoadSign, 179
<Axis>.safetyInfoChan.in.esrRetractRequested, 175
<Axis>.safetyInfoChan.in.externalBrakeRequested, 179
<Axis>.safetyInfoChan.in.safetyFaultActive, 176
<Axis>.safetyInfoChan.in.safetyMessageActive, 175
<Axis>.safetyInfoChan.in.sdiNegativeSelected, 176
<Axis>.safetyInfoChan.in.sdiPositiveSelected, 176
<Axis>.safetyInfoChan.in.setpointInput, 178
<Axis>.safetyInfoChan.in.slaActive, 174
<Axis>.safetyInfoChan.in.slpRangeSelected, 177
<Axis>.safetyInfoChan.in.slpSelectedAndUserAgreement, 177
<Axis>.safetyInfoChan.in.slsActive, 176
<Axis>.safetyInfoChan.in.slsBit0Active, 175
<Axis>.safetyInfoChan.in.slsBit1Active, 175
<Axis>.safetyInfoChan.in.slsLimit, 177
<Axis>.safetyInfoChan.in.slsSelected, 176
<Axis>.safetyInfoChan.in.sosActive, 176
<Axis>.safetyInfoChan.in.sosSelected, 176
<Axis>.safetyInfoChan.in.ss1Active, 175
<Axis>.safetyInfoChan.in.ss2Active, 175
<Axis>.safetyInfoChan.in.ss2eActive, 177
<Axis>.safetyInfoChan.in.ss2esrActive, 177
<Axis>.safetyInfoChan.in.stoActive, 175
<Axis>.safetyInfoChan.in.testStopActive, 177
<Axis>.safetyInfoChan.in.testStopRequired, 177
<Axis>.spindle, 137
<Axis>.spindle.in, 138
<Axis>.spindle.in.constantCuttingSpeedActive, 161
<Axis>.spindle.in.ctrlModeActive, 161
<Axis>.spindle.in.cwRotActive, 160
<Axis>.spindle.in.gearChangeoverRequested, 160
<Axis>.spindle.in.inPos, 162
<Axis>.spindle.in.mFuncNumber, 159
<Axis>.spindle.in.oscillModeActive, 161

<Axis>.spindle.in.posModeActive, 161
<Axis>.spindle.in.requestedGearStageNumber, 160
<Axis>.spindle.in.rigidTappingActive, 161
<Axis>.spindle.in.sFuncValue, 159
<Axis>.spindle.in.smi24, 162
<Axis>.spindle.in.smi24.sensorS1Present, 162
<Axis>.spindle.in.smi24.sensorS1StateNumber, 162
<Axis>.spindle.in.smi24.sensorS1Value, 162
<Axis>.spindle.in.smi24.sensorS4Actuated, 163
<Axis>.spindle.in.smi24.sensorS4Present, 162
<Axis>.spindle.in.smi24.sensorS5Actuated, 163
<Axis>.spindle.in.smi24.sensorS5Present, 163
<Axis>.spindle.in.smi24.sensorS6Present, 163
<Axis>.spindle.in.smi24.sensorsPresent, 162
<Axis>.spindle.in.smi24.speedLimited, 162
<Axis>.spindle.in.speedLimitationExceeded, 160
<Axis>.spindle.in.speedSetpointAboveMax, 160
<Axis>.spindle.in.speedSetpointBelowMin, 160
<Axis>.spindle.in.speedSetpointReached, 160
<Axis>.spindle.in.syncModeActive, 161
<Axis>.spindle.in.toolWithDynLimitationActive, 161
<Axis>.spindle.out, 138
<Axis>.spindle.out.autoGearStageChange, 166
<Axis>.spindle.out.deleteSValue, 165
<Axis>.spindle.out.disableSpeedMonitor, 165
<Axis>.spindle.out.gearChangeoverDone, 164
<Axis>.spindle.out.gearStageSelection, 164
<Axis>.spindle.out.invertM3M4, 165
<Axis>.spindle.out.oscillEnable, 166
<Axis>.spindle.out.oscillRequest, 166
<Axis>.spindle.out.oscillSetDirCcw, 167
<Axis>.spindle.out.oscillSetDirCw, 167
<Axis>.spindle.out.ovrFactor, 164
<Axis>.spindle.out.resyncEnc1, 164
<Axis>.spindle.out.resyncEnc1ForPos, 165
<Axis>.spindle.out.resyncEnc2, 165
<Axis>.spindle.out.resyncEnc2ForPos, 165
<Axis>.spindle.out.startCcw, 166
<Axis>.spindle.out.startCw, 166
<Axis>.spindle.out.startPosOperation, 166
<Axis>.spindle.out.stop, 166
<Axis>.spindle.out.useFeedrateOvr, 165
<Chan>.auxDFuncs, 183
<Chan>.auxDFuncs.auxDFuncs[1].in, 264
<Chan>.auxDFuncs.auxDFuncs[2].in, 265
<Chan>.auxDFuncs.auxDFuncs[3].in, 265
<Chan>.auxDFuncs.in, 185
<Chan>.auxDFuncs[1].in.extendedAddress, 264
<Chan>.auxDFuncs[1].in.fastAckActive, 264
<Chan>.auxDFuncs[1].in.hasChanged, 264
<Chan>.auxDFuncs[1].in.value, 265
<Chan>.auxDFuncs[2].in.extendedAddress, 265
<Chan>.auxDFuncs[2].in.fastAckActive, 265
<Chan>.auxDFuncs[2].in.hasChanged, 265
<Chan>.auxDFuncs[2].in.value, 265
<Chan>.auxDFuncs[3].in.extendedAddress, 266
<Chan>.auxDFuncs[3].in.fastAckActive, 266
<Chan>.auxDFuncs[3].in.hasChanged, 265
<Chan>.auxDFuncs[3].in.value, 266
<Chan>.auxDLFuncs, 184
<Chan>.auxDLFuncs.auxDLFuncs[1], 266
<Chan>.auxDLFuncs.in, 185
<Chan>.auxDLFuncs[1].in.extendedAddress, 266
<Chan>.auxDLFuncs[1].in.fastAckActive, 266
<Chan>.auxDLFuncs[1].in.hasChanged, 266
<Chan>.auxDLFuncs[1].in.value, 267
<Chan>.auxFFuncs, 184
<Chan>.auxFFuncs.auxFFuncs[1].in, 269
<Chan>.auxFFuncs.auxFFuncs[2].in, 270
<Chan>.auxFFuncs.auxFFuncs[3].in, 270
<Chan>.auxFFuncs.auxFFuncs[4].in, 271
<Chan>.auxFFuncs.auxFFuncs[5].in, 272
<Chan>.auxFFuncs.auxFFuncs[6].in, 272
<Chan>.auxFFuncs.in, 185
<Chan>.auxFFuncs[1].in.extendedAddress, 270
<Chan>.auxFFuncs[1].in.fastAckActive, 269
<Chan>.auxFFuncs[1].in.hasChanged, 269
<Chan>.auxFFuncs[1].in.value, 270
<Chan>.auxFFuncs[2].in.extendedAddress, 270
<Chan>.auxFFuncs[2].in.fastAckActive, 270
<Chan>.auxFFuncs[2].in.hasChanged, 270
<Chan>.auxFFuncs[2].in.value, 270
<Chan>.auxFFuncs[3].in.extendedAddress, 271
<Chan>.auxFFuncs[3].in.fastAckActive, 271
<Chan>.auxFFuncs[3].in.hasChanged, 271
<Chan>.auxFFuncs[3].in.value, 271
<Chan>.auxFFuncs[4].in.extendedAddress, 272
<Chan>.auxFFuncs[4].in.fastAckActive, 271
<Chan>.auxFFuncs[4].in.hasChanged, 271
<Chan>.auxFFuncs[4].in.value, 272
<Chan>.auxFFuncs[5].in.extendedAddress, 272
<Chan>.auxFFuncs[5].in.fastAckActive, 272
<Chan>.auxFFuncs[5].in.hasChanged, 272
<Chan>.auxFFuncs[5].in.value, 272
<Chan>.auxFFuncs[6].in.extendedAddress, 273
<Chan>.auxFFuncs[6].in.fastAckActive, 273
<Chan>.auxFFuncs[6].in.hasChanged, 273
<Chan>.auxFFuncs[6].in.value, 273
<Chan>.auxHFuncs, 184
<Chan>.auxHFuncs.auxHFuncs[1].in, 267
<Chan>.auxHFuncs.auxHFuncs[2].in, 267
<Chan>.auxHFuncs.auxHFuncs[3].in, 268
<Chan>.auxHFuncs.in, 185
<Chan>.auxHFuncs[1].in.extendedAddress, 267

<Chan>.auxHFuncs[1].in.fastAckActive, 267
<Chan>.auxHFuncs[1].in.hasChanged, 267
<Chan>.auxHFuncs[1].in.value, 267
<Chan>.auxHFuncs[1].in.valueTypeDint, 267
<Chan>.auxHFuncs[2].in.extendedAddress, 268
<Chan>.auxHFuncs[2].in.fastAckActive, 268
<Chan>.auxHFuncs[2].in.hasChanged, 268
<Chan>.auxHFuncs[2].in.value, 268
<Chan>.auxHFuncs[2].in.valueTypeDint, 268
<Chan>.auxHFuncs[3].in.extendedAddress, 269
<Chan>.auxHFuncs[3].in.fastAckActive, 269
<Chan>.auxHFuncs[3].in.hasChanged, 268
<Chan>.auxHFuncs[3].in.value, 269
<Chan>.auxHFuncs[3].in.valueTypeDint, 269
<Chan>.auxMDynFuncs, 183
<Chan>.auxMDynFuncs.in, 185
<Chan>.auxMDynFuncs.in.m00Decoded, 242
<Chan>.auxMDynFuncs.in.m01Decoded, 242
<Chan>.auxMDynFuncs.in.m02Decoded, 242
<Chan>.auxMDynFuncs.in.m03Decoded, 242
<Chan>.auxMDynFuncs.in.m04Decoded, 242
<Chan>.auxMDynFuncs.in.m05Decoded, 242
<Chan>.auxMDynFuncs.in.m06Decoded, 242
<Chan>.auxMDynFuncs.in.m07Decoded, 243
<Chan>.auxMDynFuncs.in.m08Decoded, 243
<Chan>.auxMDynFuncs.in.m09Decoded, 243
<Chan>.auxMDynFuncs.in.m10Decoded, 243
<Chan>.auxMDynFuncs.in.m11Decoded, 243
<Chan>.auxMDynFuncs.in.m12Decoded, 243
<Chan>.auxMDynFuncs.in.m13Decoded, 243
<Chan>.auxMDynFuncs.in.m14Decoded, 244
<Chan>.auxMDynFuncs.in.m15Decoded, 244
<Chan>.auxMDynFuncs.in.m16Decoded, 244
<Chan>.auxMDynFuncs.in.m17Decoded, 244
<Chan>.auxMDynFuncs.in.m18Decoded, 244
<Chan>.auxMDynFuncs.in.m19Decoded, 244
<Chan>.auxMDynFuncs.in.m20Decoded, 244
<Chan>.auxMDynFuncs.in.m21Decoded, 245
<Chan>.auxMDynFuncs.in.m22Decoded, 245
<Chan>.auxMDynFuncs.in.m23Decoded, 245
<Chan>.auxMDynFuncs.in.m24Decoded, 245
<Chan>.auxMDynFuncs.in.m25Decoded, 245
<Chan>.auxMDynFuncs.in.m26Decoded, 245
<Chan>.auxMDynFuncs.in.m27Decoded, 245
<Chan>.auxMDynFuncs.in.m28Decoded, 246
<Chan>.auxMDynFuncs.in.m29Decoded, 246
<Chan>.auxMDynFuncs.in.m30Decoded, 246
<Chan>.auxMDynFuncs.in.m31Decoded, 246
<Chan>.auxMDynFuncs.in.m32Decoded, 246
<Chan>.auxMDynFuncs.in.m33Decoded, 246
<Chan>.auxMDynFuncs.in.m34Decoded, 246
<Chan>.auxMDynFuncs.in.m35Decoded, 247
<Chan>.auxMDynFuncs.in.m36Decoded, 247
<Chan>.auxMDynFuncs.in.m37Decoded, 247
<Chan>.auxMDynFuncs.in.m38Decoded, 247
<Chan>.auxMDynFuncs.in.m39Decoded, 247
<Chan>.auxMDynFuncs.in.m40Decoded, 247
<Chan>.auxMDynFuncs.in.m41Decoded, 247
<Chan>.auxMDynFuncs.in.m42Decoded, 248
<Chan>.auxMDynFuncs.in.m43Decoded, 248
<Chan>.auxMDynFuncs.in.m44Decoded, 248
<Chan>.auxMDynFuncs.in.m45Decoded, 248
<Chan>.auxMDynFuncs.in.m46Decoded, 248
<Chan>.auxMDynFuncs.in.m47Decoded, 248
<Chan>.auxMDynFuncs.in.m48Decoded, 248
<Chan>.auxMDynFuncs.in.m49Decoded, 249
<Chan>.auxMDynFuncs.in.m50Decoded, 249
<Chan>.auxMDynFuncs.in.m51Decoded, 249
<Chan>.auxMDynFuncs.in.m52Decoded, 249
<Chan>.auxMDynFuncs.in.m53Decoded, 249
<Chan>.auxMDynFuncs.in.m54Decoded, 249
<Chan>.auxMDynFuncs.in.m55Decoded, 249
<Chan>.auxMDynFuncs.in.m56Decoded, 250
<Chan>.auxMDynFuncs.in.m57Decoded, 250
<Chan>.auxMDynFuncs.in.m58Decoded, 250
<Chan>.auxMDynFuncs.in.m59Decoded, 250
<Chan>.auxMDynFuncs.in.m60Decoded, 250
<Chan>.auxMDynFuncs.in.m61Decoded, 250
<Chan>.auxMDynFuncs.in.m62Decoded, 250
<Chan>.auxMDynFuncs.in.m63Decoded, 251
<Chan>.auxMDynFuncs.in.m64Decoded, 251
<Chan>.auxMDynFuncs.in.m65Decoded, 251
<Chan>.auxMDynFuncs.in.m66Decoded, 251
<Chan>.auxMDynFuncs.in.m67Decoded, 251
<Chan>.auxMDynFuncs.in.m68Decoded, 251
<Chan>.auxMDynFuncs.in.m69Decoded, 251
<Chan>.auxMDynFuncs.in.m70Decoded, 252
<Chan>.auxMDynFuncs.in.m71Decoded, 252
<Chan>.auxMDynFuncs.in.m72Decoded, 252
<Chan>.auxMDynFuncs.in.m73Decoded, 252
<Chan>.auxMDynFuncs.in.m74Decoded, 252
<Chan>.auxMDynFuncs.in.m75Decoded, 252
<Chan>.auxMDynFuncs.in.m76Decoded, 252
<Chan>.auxMDynFuncs.in.m77Decoded, 253
<Chan>.auxMDynFuncs.in.m78Decoded, 253
<Chan>.auxMDynFuncs.in.m79Decoded, 253
<Chan>.auxMDynFuncs.in.m80Decoded, 253
<Chan>.auxMDynFuncs.in.m81Decoded, 253
<Chan>.auxMDynFuncs.in.m82Decoded, 253
<Chan>.auxMDynFuncs.in.m83Decoded, 253
<Chan>.auxMDynFuncs.in.m84Decoded, 254
<Chan>.auxMDynFuncs.in.m85Decoded, 254
<Chan>.auxMDynFuncs.in.m86Decoded, 254
<Chan>.auxMDynFuncs.in.m87Decoded, 254

<Chan>.auxMDynFuncs.in.m88Decoded, 254
<Chan>.auxMDynFuncs.in.m89Decoded, 254
<Chan>.auxMDynFuncs.in.m90Decoded, 254
<Chan>.auxMDynFuncs.in.m91Decoded, 255
<Chan>.auxMDynFuncs.in.m92Decoded, 255
<Chan>.auxMDynFuncs.in.m93Decoded, 255
<Chan>.auxMDynFuncs.in.m94Decoded, 255
<Chan>.auxMDynFuncs.in.m95Decoded, 255
<Chan>.auxMDynFuncs.in.m96Decoded, 255
<Chan>.auxMDynFuncs.in.m97Decoded, 255
<Chan>.auxMDynFuncs.in.m98Decoded, 256
<Chan>.auxMDynFuncs.in.m99Decoded, 256
<Chan>.auxMFuncs, 183
<Chan>.auxMFuncs.auxMFuncs[1].in, 256
<Chan>.auxMFuncs.auxMFuncs[2].in, 257
<Chan>.auxMFuncs.auxMFuncs[3].in, 257
<Chan>.auxMFuncs.auxMFuncs[4].in, 258
<Chan>.auxMFuncs.auxMFuncs[5].in, 259
<Chan>.auxMFuncs.in, 185
<Chan>.auxMFuncs[1].in.extendedAddress, 256
<Chan>.auxMFuncs[1].in.fastAckActive, 256
<Chan>.auxMFuncs[1].in.hasChanged, 256
<Chan>.auxMFuncs[1].in.notDecoded, 256
<Chan>.auxMFuncs[1].in.value, 257
<Chan>.auxMFuncs[2].in.extendedAddress, 257
<Chan>.auxMFuncs[2].in.fastAckActive, 257
<Chan>.auxMFuncs[2].in.hasChanged, 257
<Chan>.auxMFuncs[2].in.notDecoded, 257
<Chan>.auxMFuncs[2].in.value, 257
<Chan>.auxMFuncs[3].in.extendedAddress, 258
<Chan>.auxMFuncs[3].in.fastAckActive, 258
<Chan>.auxMFuncs[3].in.hasChanged, 258
<Chan>.auxMFuncs[3].in.notDecoded, 258
<Chan>.auxMFuncs[3].in.value, 258
<Chan>.auxMFuncs[4].in.extendedAddress, 259
<Chan>.auxMFuncs[4].in.fastAckActive, 259
<Chan>.auxMFuncs[4].in.hasChanged, 258
<Chan>.auxMFuncs[4].in.notDecoded, 259
<Chan>.auxMFuncs[4].in.value, 259
<Chan>.auxMFuncs[5].in.extendedAddress, 260
<Chan>.auxMFuncs[5].in.fastAckActive, 260
<Chan>.auxMFuncs[5].in.hasChanged, 259
<Chan>.auxMFuncs[5].in.notDecoded, 259
<Chan>.auxMFuncs[5].in.value, 260
<Chan>.auxSFuncs, 183
<Chan>.auxSFuncs.auxSFuncs[1].in, 260
<Chan>.auxSFuncs.auxSFuncs[2].in, 261
<Chan>.auxSFuncs.auxSFuncs[3].in, 261
<Chan>.auxSFuncs.in, 185
<Chan>.auxSFuncs[1].in.extendedAddress, 260
<Chan>.auxSFuncs[1].in.fastAckActive, 260
<Chan>.auxSFuncs[1].in.hasChanged, 260
<Chan>.auxSFuncs[1].in.value, 261
<Chan>.auxSFuncs[2].in.extendedAddress, 261
<Chan>.auxSFuncs[2].in.fastAckActive, 261
<Chan>.auxSFuncs[2].in.hasChanged, 261
<Chan>.auxSFuncs[2].in.value, 261
<Chan>.auxSFuncs[3].in.extendedAddress, 262
<Chan>.auxSFuncs[3].in.fastAckActive, 262
<Chan>.auxSFuncs[3].in.hasChanged, 261
<Chan>.auxSFuncs[3].in.value, 262
<Chan>.auxTFuncs, 183
<Chan>.auxTFuncs.auxTFuncs[1].in, 262
<Chan>.auxTFuncs.auxTFuncs[2].in, 263
<Chan>.auxTFuncs.auxTFuncs[3].in, 263
<Chan>.auxTFuncs.in, 185
<Chan>.auxTFuncs[1].in.extendedAddress, 262
<Chan>.auxTFuncs[1].in.fastAckActive, 262
<Chan>.auxTFuncs[1].in.hasChanged, 262
<Chan>.auxTFuncs[1].in.value, 263
<Chan>.auxTFuncs[2].in.extendedAddress, 263
<Chan>.auxTFuncs[2].in.fastAckActive, 263
<Chan>.auxTFuncs[2].in.hasChanged, 263
<Chan>.auxTFuncs[2].in.value, 263
<Chan>.auxTFuncs[3].in.extendedAddress, 264
<Chan>.auxTFuncs[3].in.fastAckActive, 264
<Chan>.auxTFuncs[3].in.hasChanged, 263
<Chan>.auxTFuncs[3].in.value, 264
<Chan>.basic, 182
<Chan>.basic.in, 185
<Chan>.basic.in.allAxesReferenced, 188
<Chan>.basic.in.allAxesStationary, 191
<Chan>.basic.in.asupActive, 195
<Chan>.basic.in.asupSilentActive, 196
<Chan>.basic.in.asupStopStateActive, 196
<Chan>.basic.in.blockSearchActionActive, 189
<Chan>.basic.in.blockSearchActive, 189
<Chan>.basic.in.blockSearchApproachActive, 190
<Chan>.basic.in.blockSearchLastActionActive, 189
<Chan>.basic.in.blockSearchSeruproActive, 190
<Chan>.basic.in.circularJogTraversActive, 195
<Chan>.basic.in.clcActive, 192
<Chan>.basic.in.clcStopAtLowerLimitActive, 192
<Chan>.basic.in.clcStopAtUpperLimitActive, 192
<Chan>.basic.in.collisionStopActive, 196
<Chan>.basic.in.configuredStopAtBlockEnd, 193
<Chan>.basic.in.configuredStopFuncActive, 193
<Chan>.basic.in.contourHandwheelAssignedNumber
, 193
<Chan>.basic.in.contourHandwheelDirInverted, 193
<Chan>.basic.in.disableReadInIgnored, 192
<Chan>.basic.in.dryRunFeedrateActive, 194
<Chan>.basic.in.executionFromExternalActive, 189

<Chan>.basic.in.externalProgLanguageModeActive, 193
<Chan>.basic.in.handwheelOvrActive, 191
<Chan>.basic.in.interruptProcessingActive, 191
<Chan>.basic.in.jogRetractActive, 195
<Chan>.basic.in.jogRetractDataAvailable, 195
<Chan>.basic.in.m00m01Active, 190
<Chan>.basic.in.m00m01AssociatedAuxFuncActive, 190
<Chan>.basic.in.m02m30Active, 190
<Chan>.basic.in.ncAlarmActive, 192
<Chan>.basic.in.ncAlarmWithProgStopActive, 192
<Chan>.basic.in.ncAlarmWithStopActive, 192
<Chan>.basic.in.ncStartRequested, 188
<Chan>.basic.in.ncStopRequested, 189
<Chan>.basic.in.orientableToolholderActive, 190
<Chan>.basic.in.overStoreActive, 194
<Chan>.basic.in.productionTargetReached, 194
<Chan>.basic.in.progStateAborted, 188
<Chan>.basic.in.progStateInterrupted, 188
<Chan>.basic.in.progStateRunning, 188
<Chan>.basic.in.progStateStopped, 188
<Chan>.basic.in.progStateWaiting, 188
<Chan>.basic.in.progTestSelected, 191
<Chan>.basic.in.protAreasNotGuaranteed, 196
<Chan>.basic.in.ptpTraversActive, 194
<Chan>.basic.in.ready, 187
<Chan>.basic.in.reducedDynamicsActive, 196
<Chan>.basic.in.refActive, 190
<Chan>.basic.in.reposDelayActive, 195
<Chan>.basic.in.reposModeChangeAck, 194
<Chan>.basic.in.reposModeSelectedNumber, 195
<Chan>.basic.in.retraceSupportActive, 189
<Chan>.basic.in.retraceSupportModeActive, 189
<Chan>.basic.in.revolutionalFeedrateActive, 191
<Chan>.basic.in.singleBlockStopAtBlockEnd, 193
<Chan>.basic.in.singleBlockStopIgnored, 193
<Chan>.basic.in.stateActive, 187
<Chan>.basic.in.stateInterrupted, 187
<Chan>.basic.in.stateReset, 187
<Chan>.basic.in.stopConditionActive, 195
<Chan>.basic.in.toffonActive, 194
<Chan>.basic.in.toffonMotionActive, 194
<Chan>.basic.in.transformationActive, 191
<Chan>.basic.in.transformationActiveNumber, 191
<Chan>.basic.out, 185
<Chan>.basic.out.abortProgLevel, 199
<Chan>.basic.out.blockSearchAckPlcActionDone, 198
<Chan>.basic.out.circularJogTravers, 201
<Chan>.basic.out.clcOvr, 197
<Chan>.basic.out.configuredStopFunc, 200
<Chan>.basic.out.contourHandwheelInvertDir, 201
<Chan>.basic.out.contourHandwheelSelection, 201
<Chan>.basic.out.contourHandwheelSim, 201
<Chan>.basic.out.contourHandwheelSimInvertDir, 201
<Chan>.basic.out.deactivateClc, 197
<Chan>.basic.out.deleteDistanceToGo, 199
<Chan>.basic.out.deleteSubProgRepetition, 199
<Chan>.basic.out.disableFeed, 196
<Chan>.basic.out.disableNcStart, 199
<Chan>.basic.out.disableReadIn, 199
<Chan>.basic.out.dryRunFeedrate, 197
<Chan>.basic.out.enableGoToStart, 202
<Chan>.basic.out.enableProtAreas, 202
<Chan>.basic.out.geoAxesTraversBcsWcsSelection, 202
<Chan>.basic.out.handwheelOffsetDrf, 201
<Chan>.basic.out.lockAllSynActs, 202
<Chan>.basic.out.m01, 197
<Chan>.basic.out.m01AssociatedAuxFunc, 201
<Chan>.basic.out.ncStart, 199
<Chan>.basic.out.ncStop, 199
<Chan>.basic.out.ncStopAxesAndSpindles, 200
<Chan>.basic.out.ncStopBlockEnd, 200
<Chan>.basic.out.pathFeedrateLimit, 198
<Chan>.basic.out.pathFeedrateOvr, 198
<Chan>.basic.out.pathFeedrateOvrFactor, 198
<Chan>.basic.out.pathFixedFeedrateSelection, 200
<Chan>.basic.out.pathRapidTraversOvr, 198
<Chan>.basic.out.pathRapidTraversOvrFactor, 198
<Chan>.basic.out.progTest, 198
<Chan>.basic.out.ptpTravers, 200
<Chan>.basic.out.refPoint, 197
<Chan>.basic.out.reposModeActivationRequest, 202
<Chan>.basic.out.reposModeSelection, 202
<Chan>.basic.out.reset, 200
<Chan>.basic.out.retraceSupportStart, 197
<Chan>.basic.out.retraceSupportTraversBackwards, 196
<Chan>.basic.out.singleBlock, 197
<Chan>.basic.out.skipBlockLevel0, 202
<Chan>.basic.out.skipBlockLevel1, 203
<Chan>.basic.out.skipBlockLevel2, 203
<Chan>.basic.out.skipBlockLevel3, 203
<Chan>.basic.out.skipBlockLevel4, 203
<Chan>.basic.out.skipBlockLevel5, 203
<Chan>.basic.out.skipBlockLevel6, 203
<Chan>.basic.out.skipBlockLevel7, 203
<Chan>.basic.out.skipBlockLevel8, 204
<Chan>.basic.out.skipBlockLevel9, 204
<Chan>.basic.out.suppressDisableNcStart, 200
<Chan>.config, 182

<Chan>.config.chanActive, 187
<Chan>.config.chanNumber, 187
<Chan>.config.registered, 187
<Chan>.cycles, 184
<Chan>.cycles.in, 184
<Chan>.cycles.in.signals, 306
<Chan>.cycles.out, 184
<Chan>.cycles.out.signals, 306
<Chan>.gCommandGroup, 182
<Chan>.gCommandGroup.in, 186
<Chan>.gCommandGroup.in.activeValues, 204
<Chan>.gCommandGroup.in.activeValues.activeValues[1], 204
<Chan>.gCommandGroup.in.activeValues.activeValues[10], 206
<Chan>.gCommandGroup.in.activeValues.activeValues[11], 206
<Chan>.gCommandGroup.in.activeValues.activeValues[12], 206
<Chan>.gCommandGroup.in.activeValues.activeValues[13], 206
<Chan>.gCommandGroup.in.activeValues.activeValues[14], 206
<Chan>.gCommandGroup.in.activeValues.activeValues[15], 206
<Chan>.gCommandGroup.in.activeValues.activeValues[16], 207
<Chan>.gCommandGroup.in.activeValues.activeValues[17], 207
<Chan>.gCommandGroup.in.activeValues.activeValues[18], 207
<Chan>.gCommandGroup.in.activeValues.activeValues[19], 207
<Chan>.gCommandGroup.in.activeValues.activeValues[2], 204
<Chan>.gCommandGroup.in.activeValues.activeValues[20], 207
<Chan>.gCommandGroup.in.activeValues.activeValues[21], 207
<Chan>.gCommandGroup.in.activeValues.activeValues[22], 208
<Chan>.gCommandGroup.in.activeValues.activeValues[23], 208
<Chan>.gCommandGroup.in.activeValues.activeValues[24], 208
<Chan>.gCommandGroup.in.activeValues.activeValues[25], 208
<Chan>.gCommandGroup.in.activeValues.activeValues[26], 208
<Chan>.gCommandGroup.in.activeValues.activeValues[27], 208
<Chan>.gCommandGroup.in.activeValues.activeValues[28], 209
<Chan>.gCommandGroup.in.activeValues.activeValues[29], 209
<Chan>.gCommandGroup.in.activeValues.activeValues[3], 204
<Chan>.gCommandGroup.in.activeValues.activeValues[30], 209
<Chan>.gCommandGroup.in.activeValues.activeValues[31], 209
<Chan>.gCommandGroup.in.activeValues.activeValues[32], 209
<Chan>.gCommandGroup.in.activeValues.activeValues[33], 209
<Chan>.gCommandGroup.in.activeValues.activeValues[34], 210
<Chan>.gCommandGroup.in.activeValues.activeValues[35], 210
<Chan>.gCommandGroup.in.activeValues.activeValues[36], 210
<Chan>.gCommandGroup.in.activeValues.activeValues[37], 210
<Chan>.gCommandGroup.in.activeValues.activeValues[38], 210
<Chan>.gCommandGroup.in.activeValues.activeValues[39], 210
<Chan>.gCommandGroup.in.activeValues.activeValues[4], 205
<Chan>.gCommandGroup.in.activeValues.activeValues[40], 211
<Chan>.gCommandGroup.in.activeValues.activeValues[41], 211
<Chan>.gCommandGroup.in.activeValues.activeValues[42], 211
<Chan>.gCommandGroup.in.activeValues.activeValues[43], 211
<Chan>.gCommandGroup.in.activeValues.activeValues[44], 211
<Chan>.gCommandGroup.in.activeValues.activeValues[45], 211
<Chan>.gCommandGroup.in.activeValues.activeValues[46], 212
<Chan>.gCommandGroup.in.activeValues.activeValues[47], 212
<Chan>.gCommandGroup.in.activeValues.activeValues[48], 212
<Chan>.gCommandGroup.in.activeValues.activeValues[49], 212
<Chan>.gCommandGroup.in.activeValues.activeValues[5], 205
<Chan>.gCommandGroup.in.activeValues.activeValues[50], 212

<Chan>.gCommandGroup.in.activeValues.activeValues[51], 212
<Chan>.gCommandGroup.in.activeValues.activeValues[52], 213
<Chan>.gCommandGroup.in.activeValues.activeValues[53], 213
<Chan>.gCommandGroup.in.activeValues.activeValues[54], 213
<Chan>.gCommandGroup.in.activeValues.activeValues[55], 213
<Chan>.gCommandGroup.in.activeValues.activeValues[56], 213
<Chan>.gCommandGroup.in.activeValues.activeValues[57], 213
<Chan>.gCommandGroup.in.activeValues.activeValues[58], 214
<Chan>.gCommandGroup.in.activeValues.activeValues[59], 214
<Chan>.gCommandGroup.in.activeValues.activeValues[6], 205
<Chan>.gCommandGroup.in.activeValues.activeValues[60], 214
<Chan>.gCommandGroup.in.activeValues.activeValues[61], 214
<Chan>.gCommandGroup.in.activeValues.activeValues[62], 214
<Chan>.gCommandGroup.in.activeValues.activeValues[63], 214
<Chan>.gCommandGroup.in.activeValues.activeValues[64], 215
<Chan>.gCommandGroup.in.activeValues.activeValues[7], 205
<Chan>.gCommandGroup.in.activeValues.activeValues[8], 205
<Chan>.gCommandGroup.in.activeValues.activeValues[9], 205
<Chan>.gCommandGroup.in.g00Active, 215
<Chan>.geoAxis.in, 186
<Chan>.geoAxis.in.contManualTraversActive, 217
<Chan>.geoAxis.in.handwheelAssignedNumber, 215
<Chan>.geoAxis.in.handwheelDirInverted, 215
<Chan>.geoAxis.in.inc10000Active, 216
<Chan>.geoAxis.in.inc1000Active, 216
<Chan>.geoAxis.in.inc100Active, 216
<Chan>.geoAxis.in.inc10Active, 216
<Chan>.geoAxis.in.inc1Active, 216
<Chan>.geoAxis.in.incVarActive, 217
<Chan>.geoAxis.in.traversMinusCommand, 216
<Chan>.geoAxis.in.traversMinusRequest, 215
<Chan>.geoAxis.in.traversPlusCommand, 216
<Chan>.geoAxis.in.traversPlusRequest, 215
<Chan>.geoAxis.out, 186
<Chan>.geoAxis.out.contManualTravers, 219
<Chan>.geoAxis.out.disableFeed, 217
<Chan>.geoAxis.out.disableTraversKey, 217
<Chan>.geoAxis.out.handwheelInvertDir, 217
<Chan>.geoAxis.out.handwheelSelection, 217
<Chan>.geoAxis.out.inc1, 218
<Chan>.geoAxis.out.inc10, 218
<Chan>.geoAxis.out.inc100, 218
<Chan>.geoAxis.out.inc1000, 218
<Chan>.geoAxis.out.inc10000, 219
<Chan>.geoAxis.out.incVar, 219
<Chan>.geoAxis.out.rapidTraversOvr, 218
<Chan>.geoAxis.out.traversKeyMinus, 218
<Chan>.geoAxis.out.traversKeyPlus, 218
<Chan>.geoAxis1, 182
<Chan>.geoAxis1.in, 215
<Chan>.geoAxis1.out, 217
<Chan>.geoAxis2, 182
<Chan>.geoAxis2.in, 219
<Chan>.geoAxis2.in.contManualTraversActive, 221
<Chan>.geoAxis2.in.handwheelAssignedNumber, 219
<Chan>.geoAxis2.in.handwheelDirInverted, 219
<Chan>.geoAxis2.in.inc10000Active, 221
<Chan>.geoAxis2.in.inc1000Active, 220
<Chan>.geoAxis2.in.inc100Active, 220
<Chan>.geoAxis2.in.inc10Active, 220
<Chan>.geoAxis2.in.inc1Active, 220
<Chan>.geoAxis2.in.incVarActive, 221
<Chan>.geoAxis2.in.traversMinusCommand, 220
<Chan>.geoAxis2.in.traversMinusRequest, 219
<Chan>.geoAxis2.in.traversPlusCommand, 220
<Chan>.geoAxis2.in.traversPlusRequest, 220
<Chan>.geoAxis2.out, 221
<Chan>.geoAxis2.out.contManualTravers, 223
<Chan>.geoAxis2.out.disableFeed, 221
<Chan>.geoAxis2.out.disableTraversKey, 222
<Chan>.geoAxis2.out.handwheelInvertDir, 221
<Chan>.geoAxis2.out.handwheelSelection, 221
<Chan>.geoAxis2.out.inc1, 222
<Chan>.geoAxis2.out.inc10, 222
<Chan>.geoAxis2.out.inc100, 222
<Chan>.geoAxis2.out.inc1000, 223
<Chan>.geoAxis2.out.inc10000, 223
<Chan>.geoAxis2.out.incVar, 223
<Chan>.geoAxis2.out.rapidTraversOvr, 222
<Chan>.geoAxis2.out.traversKeyMinus, 222
<Chan>.geoAxis2.out.traversKeyPlus, 222
<Chan>.geoAxis3, 183
<Chan>.geoAxis3.in, 223
<Chan>.geoAxis3.in.contManualTraversActive, 225

<Chan>.geoAxis3.in.handwheelAssignedNumber, 223
<Chan>.geoAxis3.in.handwheelDirInverted, 223
<Chan>.geoAxis3.in.inc10000Active, 225
<Chan>.geoAxis3.in.inc1000Active, 225
<Chan>.geoAxis3.in.inc100Active, 224
<Chan>.geoAxis3.in.inc10Active, 224
<Chan>.geoAxis3.in.inc1Active, 224
<Chan>.geoAxis3.in.incVarActive, 225
<Chan>.geoAxis3.in.traversMinusCommand, 224
<Chan>.geoAxis3.in.traversMinusRequest, 224
<Chan>.geoAxis3.in.traversPlusCommand, 224
<Chan>.geoAxis3.in.traversPlusRequest, 224
<Chan>.geoAxis3.out, 225
<Chan>.geoAxis3.out.contManualTravers, 227
<Chan>.geoAxis3.out.disableFeed, 226
<Chan>.geoAxis3.out.disableTraversKey, 226
<Chan>.geoAxis3.out.handwheelInvertDir, 225
<Chan>.geoAxis3.out.handwheelSelection, 225
<Chan>.geoAxis3.out.inc1, 226
<Chan>.geoAxis3.out.inc10, 226
<Chan>.geoAxis3.out.inc100, 227
<Chan>.geoAxis3.out.inc1000, 227
<Chan>.geoAxis3.out.inc10000, 227
<Chan>.geoAxis3.out.incVar, 227
<Chan>.geoAxis3.out.rapidTraversOvr, 226
<Chan>.geoAxis3.out.traversKeyMinus, 226
<Chan>.geoAxis3.out.traversKeyPlus, 226
<Chan>.grinding, 183
<Chan>.grinding.in, 186
<Chan>.grinding.in.requested, 304
<Chan>.grinding.in.signalsEnabled, 304
<Chan>.grinding.out, 186
<Chan>.grinding.out.ackFuncsExecution, 304
<Chan>.grinding.out.disableSignals, 304
<Chan>.grinding.out.set, 304
<Chan>.nibbelingAndPunching, 184
<Chan>.nibbelingAndPunching.in, 184
<Chan>.nibbelingAndPunching.in.strokeInitiationActive, 305
<Chan>.nibbelingAndPunching.in.strokeManualInitiationAck, 305
<Chan>.nibbelingAndPunching.out, 184
<Chan>.nibbelingAndPunching.out.delayStroke, 305
<Chan>.nibbelingAndPunching.out.enableStroke, 305
<Chan>.nibbelingAndPunching.out.manualStrokeInterface1, 306
<Chan>.nibbelingAndPunching.out.manualStrokeInterface2, 306
<Chan>.nibbelingAndPunching.out.noStroke, 305
<Chan>.nibbelingAndPunching.out.suppressStroke, 305
<Chan>.orientationAxis.in, 186
<Chan>.orientationAxis.in.contManualTraversActive, 229
<Chan>.orientationAxis.in.handwheelAssignedNumber, 227
<Chan>.orientationAxis.in.handwheelDirInverted, 228
<Chan>.orientationAxis.in.inc10000Active, 229
<Chan>.orientationAxis.in.inc1000Active, 229
<Chan>.orientationAxis.in.inc100Active, 229
<Chan>.orientationAxis.in.inc10Active, 228
<Chan>.orientationAxis.in.inc1Active, 228
<Chan>.orientationAxis.in.incVarActive, 229
<Chan>.orientationAxis.in.traversMinusCommand, 228
<Chan>.orientationAxis.in.traversMinusRequest, 228
<Chan>.orientationAxis.in.traversPlusCommand, 228
<Chan>.orientationAxis.in.traversPlusRequest, 228
<Chan>.orientationAxis.out, 187
<Chan>.orientationAxis.out.contManualTravers, 231
<Chan>.orientationAxis.out.disableFeed, 230
<Chan>.orientationAxis.out.disableTraversKey, 230
<Chan>.orientationAxis.out.handwheelInvertDir, 230
<Chan>.orientationAxis.out.handwheelSelection, 229
<Chan>.orientationAxis.out.inc1, 230
<Chan>.orientationAxis.out.inc10, 231
<Chan>.orientationAxis.out.inc100, 231
<Chan>.orientationAxis.out.inc1000, 231
<Chan>.orientationAxis.out.inc10000, 231
<Chan>.orientationAxis.out.incVar, 231
<Chan>.orientationAxis.out.rapidTraversOvr, 230
<Chan>.orientationAxis.out.traversKeyMinus, 230
<Chan>.orientationAxis.out.traversKeyPlus, 230
<Chan>.orientationAxis1, 183
<Chan>.orientationAxis1.in, 227
<Chan>.orientationAxis1.out, 229
<Chan>.orientationAxis2, 183
<Chan>.orientationAxis2.in, 231
<Chan>.orientationAxis2.in.contManualTraversActive, 233
<Chan>.orientationAxis2.in.handwheelAssignedNumber, 232
<Chan>.orientationAxis2.in.handwheelDirInverted, 232
<Chan>.orientationAxis2.in.inc10000Active, 233
<Chan>.orientationAxis2.in.inc1000Active, 233
<Chan>.orientationAxis2.in.inc100Active, 233
<Chan>.orientationAxis2.in.inc10Active, 233
<Chan>.orientationAxis2.in.inc1Active, 232

<Chan>.orientationAxis2.in.incVarActive, 233
<Chan>.orientationAxis2.in.traversMinusCommand, 232
<Chan>.orientationAxis2.in.traversMinusRequest, 232
<Chan>.orientationAxis2.in.traversPlusCommand, 232
<Chan>.orientationAxis2.in.traversPlusRequest, 232
<Chan>.orientationAxis2.out, 233
<Chan>.orientationAxis2.out.contManualTravers, 235
<Chan>.orientationAxis2.out.disableFeed, 234
<Chan>.orientationAxis2.out.disableTraversKey, 234
<Chan>.orientationAxis2.out.handwheelInvertDir, 234
<Chan>.orientationAxis2.out.handwheelSelection, 234
<Chan>.orientationAxis2.out.inc1, 235
<Chan>.orientationAxis2.out.inc10, 235
<Chan>.orientationAxis2.out.inc100, 235
<Chan>.orientationAxis2.out.inc1000, 235
<Chan>.orientationAxis2.out.inc10000, 235
<Chan>.orientationAxis2.out.incVar, 235
<Chan>.orientationAxis2.out.rapidTraversOvr, 234
<Chan>.orientationAxis2.out.traversKeyMinus, 234
<Chan>.orientationAxis2.out.traversKeyPlus, 234
<Chan>.orientationAxis3, 183
<Chan>.orientationAxis3.in, 236
<Chan>.orientationAxis3.in.contManualTraversActive, 237
<Chan>.orientationAxis3.in.handwheelAssignedNumber, 236
<Chan>.orientationAxis3.in.handwheelDirInverted, 236
<Chan>.orientationAxis3.in.inc10000Active, 237
<Chan>.orientationAxis3.in.inc1000Active, 237
<Chan>.orientationAxis3.in.inc100Active, 237
<Chan>.orientationAxis3.in.inc10Active, 237
<Chan>.orientationAxis3.in.inc1Active, 237
<Chan>.orientationAxis3.in.incVarActive, 237
<Chan>.orientationAxis3.in.traversMinusCommand, 236
<Chan>.orientationAxis3.in.traversMinusRequest, 236
<Chan>.orientationAxis3.in.traversPlusCommand, 236
<Chan>.orientationAxis3.in.traversPlusRequest, 236
<Chan>.orientationAxis3.out, 238
<Chan>.orientationAxis3.out.contManualTravers, 240
<Chan>.orientationAxis3.out.disableFeed, 238
<Chan>.orientationAxis3.out.disableTraversKey, 238
<Chan>.orientationAxis3.out.handwheelInvertDir, 238
<Chan>.orientationAxis3.out.handwheelSelection, 238
<Chan>.orientationAxis3.out.inc1, 239
<Chan>.orientationAxis3.out.inc10, 239
<Chan>.orientationAxis3.out.inc100, 239
<Chan>.orientationAxis3.out.inc1000, 239
<Chan>.orientationAxis3.out.inc10000, 239
<Chan>.orientationAxis3.out.incVar, 239
<Chan>.orientationAxis3.out.rapidTraversOvr, 238
<Chan>.orientationAxis3.out.traversKeyMinus, 238
<Chan>.orientationAxis3.out.traversKeyPlus, 239
<Chan>.progEvent, 184
<Chan>.progEvent.in, 186
<Chan>.progEvent.in.blockSearchActive, 282
<Chan>.progEvent.in.ncStartActive, 282
<Chan>.progEvent.in.partProgramEndActive, 282
<Chan>.progEvent.in.powerOnActive, 282
<Chan>.progEvent.in.resetActive, 282
<Chan>.protAreas, 184
<Chan>.protAreas.in, 185
<Chan>.protAreas.in.protAreas.in[1], 273
<Chan>.protAreas.out, 185
<Chan>.protAreas[1].in.chanPreactivated, 274
<Chan>.protAreas[1].in.chanViolated, 274
<Chan>.protAreas[1].in.ncPreactivated, 273
<Chan>.protAreas[1].in.ncViolated, 273
<Chan>.protAreas[1].out.chan, 274
<Chan>.protAreas[1].out.nc, 274
<Chan>.protAreas[10].in.chanPreactivated, 281
<Chan>.protAreas[10].in.chanViolated, 281
<Chan>.protAreas[10].in.ncPreactivated, 281
<Chan>.protAreas[10].in.ncViolated, 281
<Chan>.protAreas[10].out.chan, 282
<Chan>.protAreas[10].out.nc, 282
<Chan>.protAreas[2].in.chanPreactivated, 274
<Chan>.protAreas[2].in.chanViolated, 275
<Chan>.protAreas[2].in.ncPreactivated, 274
<Chan>.protAreas[2].in.ncViolated, 274
<Chan>.protAreas[2].out.chan, 275
<Chan>.protAreas[2].out.nc, 275
<Chan>.protAreas[3].in.chanPreactivated, 275
<Chan>.protAreas[3].in.chanViolated, 275
<Chan>.protAreas[3].in.ncPreactivated, 275
<Chan>.protAreas[3].in.ncViolated, 275
<Chan>.protAreas[3].out.chan, 276
<Chan>.protAreas[3].out.nc, 276
<Chan>.protAreas[4].in.chanPreactivated, 276
<Chan>.protAreas[4].in.chanSpecificViolated, 276
<Chan>.protAreas[4].in.ncPreactivated, 276
<Chan>.protAreas[4].in.ncViolated, 276

<Chan>.protAreas[4].out.chan, 277
<Chan>.protAreas[4].out.nc, 276
<Chan>.protAreas[5].in.chanSpecificPreactivated, 277
<Chan>.protAreas[5].in.chanViolated, 277
<Chan>.protAreas[5].in.ncPreactivated, 277
<Chan>.protAreas[5].in.ncViolated, 277
<Chan>.protAreas[5].out.chan, 277
<Chan>.protAreas[5].out.nc, 277
<Chan>.protAreas[6].in.chanPreactivated, 278
<Chan>.protAreas[6].in.chanViolated, 278
<Chan>.protAreas[6].in.ncPreactivated, 278
<Chan>.protAreas[6].in.ncViolated, 278
<Chan>.protAreas[6].out.chan, 278
<Chan>.protAreas[6].out.nc, 278
<Chan>.protAreas[7].in.chanPreactivated, 279
<Chan>.protAreas[7].in.chanViolated, 279
<Chan>.protAreas[7].in.ncPreactivated, 278
<Chan>.protAreas[7].in.ncViolated, 279
<Chan>.protAreas[7].out.chan, 279
<Chan>.protAreas[7].out.nc, 279
<Chan>.protAreas[8].in.chanPreactivated, 280
<Chan>.protAreas[8].in.chanViolated, 280
<Chan>.protAreas[8].in.ncPreactivated, 279
<Chan>.protAreas[8].in.ncViolated, 279
<Chan>.protAreas[8].out.chan, 280
<Chan>.protAreas[8].out.nc, 280
<Chan>.protAreas[9].in.chanPreactivated, 280
<Chan>.protAreas[9].in.chanViolated, 281
<Chan>.protAreas[9].in.ncPreactivated, 280
<Chan>.protAreas[9].in.ncViolated, 280
<Chan>.protAreas[9].out.chan, 281
<Chan>.protAreas[9].out.nc, 281
<Chan>.synActs, 184
<Chan>.synActs.in, 186
<Chan>.synActs.out, 186
<Chan>.synActs[1].in.lockable, 283
<Chan>.synActs[1].out.lock, 283
<Chan>.synActs[10].in.lockable, 286
<Chan>.synActs[10].out.lock, 286
<Chan>.synActs[11].in.lockable, 286
<Chan>.synActs[11].out.lock, 286
<Chan>.synActs[12].in.lockable, 286
<Chan>.synActs[12].out.lock, 286
<Chan>.synActs[13].in.lockable, 287
<Chan>.synActs[13].out.lock, 287
<Chan>.synActs[14].in.lockable, 287
<Chan>.synActs[14].out.lock, 287
<Chan>.synActs[15].in.lockable, 287
<Chan>.synActs[15].out.lock, 287
<Chan>.synActs[16].in.lockable, 288
<Chan>.synActs[16].out.lock, 288
<Chan>.synActs[17].in.lockable, 288
<Chan>.synActs[17].out.lock, 288
<Chan>.synActs[18].in.lockable, 288
<Chan>.synActs[18].out.lock, 288
<Chan>.synActs[19].in.lockable, 289
<Chan>.synActs[19].out.lock, 289
<Chan>.synActs[2].in.lockable, 283
<Chan>.synActs[2].out.lock, 283
<Chan>.synActs[20].in.lockable, 289
<Chan>.synActs[20].out.lock, 289
<Chan>.synActs[21].in.lockable, 289
<Chan>.synActs[21].out.lock, 289
<Chan>.synActs[22].in.lockable, 290
<Chan>.synActs[22].out.lock, 290
<Chan>.synActs[23].in.lockable, 290
<Chan>.synActs[23].out.lock, 290
<Chan>.synActs[24].in.lockable, 290
<Chan>.synActs[24].out.lock, 290
<Chan>.synActs[25].in.lockable, 291
<Chan>.synActs[25].out.lock, 291
<Chan>.synActs[26].in.lockable, 291
<Chan>.synActs[26].out.lock, 291
<Chan>.synActs[27].in.lockable, 291
<Chan>.synActs[27].out.lock, 291
<Chan>.synActs[28].in.lockable, 292
<Chan>.synActs[28].out.lock, 292
<Chan>.synActs[29].in.lockable, 292
<Chan>.synActs[29].out.lock, 292
<Chan>.synActs[3].in.lockable, 283
<Chan>.synActs[3].out.lock, 283
<Chan>.synActs[30].in.lockable, 292
<Chan>.synActs[30].out.lock, 292
<Chan>.synActs[31].in.lockable, 293
<Chan>.synActs[31].out.lock, 293
<Chan>.synActs[32].in.lockable, 293
<Chan>.synActs[32].out.lock, 293
<Chan>.synActs[33].in.lockable, 293
<Chan>.synActs[33].out.lock, 293
<Chan>.synActs[34].in.lockable, 294
<Chan>.synActs[34].out.lock, 294
<Chan>.synActs[35].in.lockable, 294
<Chan>.synActs[35].out.lock, 294
<Chan>.synActs[36].in.lockable, 294
<Chan>.synActs[36].out.lock, 294
<Chan>.synActs[37].in.lockable, 295
<Chan>.synActs[37].out.lock, 295
<Chan>.synActs[38].in.lockable, 295
<Chan>.synActs[38].out.lock, 295
<Chan>.synActs[39].in.lockable, 295
<Chan>.synActs[39].out.lock, 295
<Chan>.synActs[4].in.lockable, 284
<Chan>.synActs[4].out.lock, 284

<Chan>.synActs[40].in.lockable, 296
<Chan>.synActs[40].out.lock, 296
<Chan>.synActs[41].in.lockable, 296
<Chan>.synActs[41].out.lock, 296
<Chan>.synActs[42].in.lockable, 296
<Chan>.synActs[42].out.lock, 296
<Chan>.synActs[43].in.lockable, 297
<Chan>.synActs[43].out.lock, 297
<Chan>.synActs[44].in.lockable, 297
<Chan>.synActs[44].out.lock, 297
<Chan>.synActs[45].in.lockable, 297
<Chan>.synActs[45].out.lock, 297
<Chan>.synActs[46].in.lockable, 298
<Chan>.synActs[46].out.lock, 298
<Chan>.synActs[47].in.lockable, 298
<Chan>.synActs[47].out.lock, 298
<Chan>.synActs[48].in.lockable, 298
<Chan>.synActs[48].out.lock, 298
<Chan>.synActs[49].in.lockable, 299
<Chan>.synActs[49].out.lock, 299
<Chan>.synActs[5].in.lockable, 284
<Chan>.synActs[5].out.lock, 284
<Chan>.synActs[50].in.lockable, 299
<Chan>.synActs[50].out.lock, 299
<Chan>.synActs[51].in.lockable, 299
<Chan>.synActs[51].out.lock, 299
<Chan>.synActs[52].in.lockable, 300
<Chan>.synActs[52].out.lock, 300
<Chan>.synActs[53].in.lockable, 300
<Chan>.synActs[53].out.lock, 300
<Chan>.synActs[54].in.lockable, 300
<Chan>.synActs[54].out.lock, 300
<Chan>.synActs[55].in.lockable, 301
<Chan>.synActs[55].out.lock, 301
<Chan>.synActs[56].in.lockable, 301
<Chan>.synActs[56].out.lock, 301
<Chan>.synActs[57].in.lockable, 301
<Chan>.synActs[57].out.lock, 301
<Chan>.synActs[58].in.lockable, 302
<Chan>.synActs[58].out.lock, 302
<Chan>.synActs[59].in.lockable, 302
<Chan>.synActs[59].out.lock, 302
<Chan>.synActs[6].in.lockable, 284
<Chan>.synActs[6].out.lock, 284
<Chan>.synActs[60].in.lockable, 302
<Chan>.synActs[60].out.lock, 302
<Chan>.synActs[61].in.lockable, 303
<Chan>.synActs[61].out.lock, 303
<Chan>.synActs[62].in.lockable, 303
<Chan>.synActs[62].out.lock, 303
<Chan>.synActs[63].in.lockable, 303
<Chan>.synActs[63].out.lock, 303
<Chan>.synActs[64].in.lockable, 304
<Chan>.synActs[64].out.lock, 304
<Chan>.synActs[7].in.lockable, 285
<Chan>.synActs[7].out.lock, 285
<Chan>.synActs[8].in.lockable, 285
<Chan>.synActs[8].out.lock, 285
<Chan>.synActs[9].in.lockable, 285
<Chan>.synActs[9].out.lock, 285
<Chan>.toolMgmt, 183
<Chan>.toolMgmt.in, 186
<Chan>.toolMgmt.in.lastReplacementToolReached, 240
<Chan>.toolMgmt.in.lastTNumber, 241
<Chan>.toolMgmt.in.newReplacementToolActive, 240
<Chan>.toolMgmt.in.newTNumber, 241
<Chan>.toolMgmt.in.toolLimitTNumber, 241
<Chan>.toolMgmt.in.toolLimitValueReached, 240
<Chan>.toolMgmt.in.toolMissing, 240
<Chan>.toolMgmt.in.toolPrewarnLimitReached, 240
<Chan>.toolMgmt.in.toolPrewarnLimitTNumber, 240
<Chan>.toolMgmt.out, 186
<Chan>.toolMgmt.out.deactivateWearMonitoring, 241
<Chan>.toolMgmt.out.deactivateWorkpieceCounter, 241
<Chan>.toolMgmt.out.ignoreDisabledSate, 241
<Chan>.toolMgmt.out.toolLifeMonitoring, 241
<Hmi>.basic, 421
<Hmi>.basic.in, 421
<Hmi>.basic.in.activeOperatingAreaNumber, 422
<Hmi>.basic.in.cancelKeyPressed, 421
<Hmi>.basic.in.currentChanNumber, 422
<Hmi>.basic.in.currentScreenNumber, 423
<Hmi>.basic.in.dataTransferActive, 422
<Hmi>.basic.in.directKeyAddress, 423
<Hmi>.basic.in.inputDeviceActive, 422
<Hmi>.basic.in.languageId, 423
<Hmi>.basic.in.mcpAddress, 423
<Hmi>.basic.in.mcsWcsSwitched, 422
<Hmi>.basic.in.screenChangeActive, 422
<Hmi>.basic.in.screenDark, 421
<Hmi>.basic.in.simActive, 422
<Hmi>.basic.in.switchoverTcuDisabled, 423
<Hmi>.basic.in.tcuIndex, 423
<Hmi>.basic.inOut, 421
<Hmi>.basic.inOut.cancelAlarmsCleared, 425
<Hmi>.basic.inOut.plcHardkeySelection, 425
<Hmi>.basic.inOut.recallAlarmsCleared, 425
<Hmi>.basic.out, 421
<Hmi>.basic.out.analogSpindle1Percentage, 425
<Hmi>.basic.out.analogSpindle2Percentage, 425

<Hmi>.basic.out.brightenScreen, 423
<Hmi>.basic.out.clearCancelAlarms, 424
<Hmi>.basic.out.clearRecallAlarms, 424
<Hmi>.basic.out.darkenScreen, 424
<Hmi>.basic.out.disableKeys, 424
<Hmi>.basic.out.disableTeachInTransfer, 425
<Hmi>.basic.out.displayWcsValues, 424
<Hmi>.basic.out.permitRemoteMonitoringOnly, 425
<Hmi>.basic.out.rejectRemoteAccess, 424
<Hmi>.basic.out.saveActionLog, 424
<Hmi>.config, 420
<Hmi>.config.registered, 421
<Hmi>.mappingTable, 420
<Hmi>.mappingTable.in, 428
<Hmi>.mappingTable.in.error, 428
<Hmi>.mappingTable.in.ready, 428
<Hmi>.mappingTable.in.status, 428
<Hmi>.mappingTable.out, 428
<Hmi>.mappingTable.out.numberOfTableLines, 429
<Hmi>.mappingTable.out.status, 429
<Hmi>.mappingTable.out.tableLines, 428
<Hmi>.mappingTable.out.version, 429
<Hmi>.plcFunc, 421
<Hmi>.plcFunc.in, 427
<Hmi>.plcFunc.in.busy, 427
<Hmi>.plcFunc.in.errorCode, 427
<Hmi>.plcFunc.out, 427
<Hmi>.plcFunc.out.parameter1, 427
<Hmi>.plcFunc.out.requestStrobe, 427
<Hmi>.progSelection, 421
<Hmi>.progSelection.in, 426
<Hmi>.progSelection.in.completed, 426
<Hmi>.progSelection.in.error, 426
<Hmi>.progSelection.in.errorCode, 426
<Hmi>.progSelection.in.identified, 426
<Hmi>.progSelection.in.selected, 426
<Hmi>.progSelection.out, 426
<Hmi>.progSelection.out.progList, 427
<Hmi>.progSelection.out.progNumber, 427
<Hmi>.progSelection.out.request, 426
<Hmi>.userKeys, 421
<Hmi>.userKeys.in, 427
<Hmi>.userKeys.out, 428
<Hmi>.userKeys[1].in.pressed, 428
<Hmi>.userKeys[1].out.activateLed, 428
<Hmi>.userKeys[1].out.disable, 428
<HmiAxis>.basic, 429
<HmiAxis>.basic.in, 429
<HmiAxis>.basic.in.handwheelSelected, 429
<HmiAxis>.basic.in.progTestRequested, 429
<HmiAxis>.basic.in.progTestSuppressionRequested, 429
<HmiAxis>.config, 429
<HmiChan>.axisSelection, 430
<HmiChan>.axisSelection.in, 430
<HmiChan>.axisSelection.in.selectedKeyActuated, 434
<HmiChan>.axisSelection.in.selectedNumber, 434
<HmiChan>.axisSelection.in.selectedType, 434
<HmiChan>.axisSelection.out, 430
<HmiChan>.axisSelection.out.activeNumber, 434
<HmiChan>.axisSelection.out.activeType, 434
<HmiChan>.basic, 430
<HmiChan>.basic.in, 430
<HmiChan>.basic.in.atStopSetOvrZeroSelected, 431
<HmiChan>.basic.in.configuredStopFuncSelected, 431
<HmiChan>.basic.in.contourHandwheelSelected, 431
<HmiChan>.basic.in.dryRunSelected, 432
<HmiChan>.basic.in.handwheelOffsetDrfSelected, 431
<HmiChan>.basic.in.m01AssociatedAuxFuncSelected, 431
<HmiChan>.basic.in.m01Selected, 431
<HmiChan>.basic.in.progTestSelected, 432
<HmiChan>.basic.in.rapidTraversOvrSelected, 432
<HmiChan>.basic.in.reposModeChanged, 432
<HmiChan>.basic.in.reposModeSelectedNumber, 432
<HmiChan>.basic.in.skipBlockLevel0, 432
<HmiChan>.basic.in.skipBlockLevel1, 432
<HmiChan>.basic.in.skipBlockLevel2, 433
<HmiChan>.basic.in.skipBlockLevel3, 433
<HmiChan>.basic.in.skipBlockLevel4, 433
<HmiChan>.basic.in.skipBlockLevel5, 433
<HmiChan>.basic.in.skipBlockLevel6, 433
<HmiChan>.basic.in.skipBlockLevel7, 433
<HmiChan>.basic.in.skipBlockLevel8, 433
<HmiChan>.basic.in.skipBlockLevel9, 434
<HmiChan>.config, 430
<HmiChan>.geoAxis.in, 431
<HmiChan>.geoAxis.in.handwheelSelected, 434
<HmiChan>.geoAxis1, 430
<HmiChan>.geoAxis1.in.handwheelSelected, 434
<HmiChan>.geoAxis2, 430
<HmiChan>.geoAxis2.in.handwheelSelected, 435
<HmiChan>.geoAxis3, 430
<HmiChan>.geoAxis3.in.handwheelSelected, 435
<HmiChan>.orientationAxis.in, 431
<HmiChan>.orientationAxis.in.handwheelSelected, 435
<HmiChan>.orientationAxis1, 430

<HmiChan>.orientationAxis1.in.handwheelSelected, 435
<HmiChan>.orientationAxis2, 430
<HmiChan>.orientationAxis2.in.handwheelSelected, 435
<HmiChan>.orientationAxis3, 430
<HmiChan>.orientationAxis3.in.handwheelSelected, 435
<HmiModeGroup>.basic, 435
<HmiModeGroup>.basic.in, 435
<HmiModeGroup>.basic.in.autoModeRequested, 435
<HmiModeGroup>.basic.in.contTraversRequested, 437
<HmiModeGroup>.basic.in.inc10000Requested, 437
<HmiModeGroup>.basic.in.inc1000Requested, 437
<HmiModeGroup>.basic.in.inc100Requested, 436
<HmiModeGroup>.basic.in.inc10Requested, 436
<HmiModeGroup>.basic.in.inc1Requested, 436
<HmiModeGroup>.basic.in.incVarRequested, 437
<HmiModeGroup>.basic.in.jogModeRequested, 436
<HmiModeGroup>.basic.in.mdaModeRequested, 436
<HmiModeGroup>.basic.in.refRequested, 436
<HmiModeGroup>.basic.in.reposRequested, 436
<HmiModeGroup>.basic.in.teachInRequested, 436
<HmiModeGroup>.config, 435
<HmiNc>.collision, 437
<HmiNc>.collision.in, 437
<HmiNc>.collision.in.fixtureAreasAutoDeactivated, 438
<HmiNc>.collision.in.fixtureAreasJogDeactivated, 438
<HmiNc>.collision.in.machineAreasAutoDeactivated, 437
<HmiNc>.collision.in.machineAreasJogDeactivated, 438
<HmiNc>.collision.in.toolAreasAutoDeactivated, 438
<HmiNc>.collision.in.toolAreasJogDeactivated, 438
<HmiNc>.collision.in.workpieceAreasAutoDeactivated, 438
<HmiNc>.collision.in.workpieceAreasJogDeactivated, 438
<HmiNc>.config, 437
<HmiNc>.config.registered, 437
<LoadingPoint>
.error, 1025
<LoadingPoint>.ack, 413
<LoadingPoint>.ack.in, 413
<LoadingPoint>.ack.in.busy, 418
<LoadingPoint>.ack.in.done, 418
<LoadingPoint>.ack.in.error, 418
<LoadingPoint>.ack.in.status, 419
<LoadingPoint>.ack.out, 413
<LoadingPoint>.ack.out.execute, 419
<LoadingPoint>.ack.out.multitoolLocation, 419
<LoadingPoint>.ack.out.status, 419
<LoadingPoint>.ack.out.toolLocation, 419
<LoadingPoint>.ack.out.toolMagazine, 419
<LoadingPoint>.ack.out.totalComplete, 419
<LoadingPoint>.ack.out.totalFailed, 419
<LoadingPoint>.command, 413
<LoadingPoint>.command.in, 413
<LoadingPoint>.command.in.basic, 414
<LoadingPoint>.command.in.basic.chan, 415
<LoadingPoint>.command.in.basic.load, 415
<LoadingPoint>.command.in.basic.ncCommand, 414
<LoadingPoint>.command.in.basic.position, 415
<LoadingPoint>.command.in.basic.tool, 415
<LoadingPoint>.command.in.basic.toolSourceLocation, 415
<LoadingPoint>.command.in.basic.toolSourceMagazine, 415
<LoadingPoint>.command.in.basic.toolTargetLocation, 416
<LoadingPoint>.command.in.basic.toolTargetMagazine, 416
<LoadingPoint>.command.in.basic.unload, 415
<LoadingPoint>.command.in.multitoolProperties, 414
<LoadingPoint>.command.in.multitoolProperties.distance, 418
<LoadingPoint>.command.in.multitoolProperties.distanceType, 418
<LoadingPoint>.command.in.multitoolProperties.numberOfLocations, 418
<LoadingPoint>.command.in.multitoolRequest, 414
<LoadingPoint>.command.in.options, 414
<LoadingPoint>.command.in.options.byNcProg, 416
<LoadingPoint>.command.in.request, 414
<LoadingPoint>.command.in.toolProperties, 414
<LoadingPoint>.command.in.toolProperties.toolLocationType, 416
<LoadingPoint>.command.in.toolProperties.toolSizeBottom, 416
<LoadingPoint>.command.in.toolProperties.toolSizeLeft, 416
<LoadingPoint>.command.in.toolProperties.toolSizeRight, 416
<LoadingPoint>.command.in.toolProperties.toolSizeTop, 416
<LoadingPoint>.command.in.toolStatus, 414
<LoadingPoint>.command.in.toolStatus.active, 416
<LoadingPoint>.command.in.toolStatus.changeActive, 417

<LoadingPoint>.command.in.toolStatus.changeOneToOne, 418
<LoadingPoint>.command.in.toolStatus.disabled, 417
<LoadingPoint>.command.in.toolStatus.disableStateIgnored, 417
<LoadingPoint>.command.in.toolStatus.enabled, 417
<LoadingPoint>.command.in.toolStatus.fixedLocation, 417
<LoadingPoint>.command.in.toolStatus.manualTool, 418
<LoadingPoint>.command.in.toolStatus.masterTool, 417
<LoadingPoint>.command.in.toolStatus.measured, 417
<LoadingPoint>.command.in.toolStatus.prewarning, 417
<LoadingPoint>.command.in.toolStatus.toBeLoaded, 417
<LoadingPoint>.command.in.toolStatus.toBeUnloaded, 417
<LoadingPoint>.command.in.toolStatus.toolAtDisabledLocation, 418
<LoadingPoint>.command.in.toolStatus.toolInBuffer, 417
<LoadingPoint>.command.in.toolStatus.used, 417
<LoadingPoint>.command.in.userParameters, 414
<LoadingPoint>.config, 413
<LoadingPoint>.config.loadingLocation, 414
<LoadingPoint>.config.loadingMagazine, 413
<LoadingPoint>.config.loadingPoint, 413
<LoadingPoint>.config.registered, 413
<LoadingPoint>.config.toaUnit, 413
<ModeGroup>.basic, 306
<ModeGroup>.basic.in, 306
<ModeGroup>.basic.in.autoModeActive, 307
<ModeGroup>.basic.in.contTraversActive, 309
<ModeGroup>.basic.in.inc10000Active, 309
<ModeGroup>.basic.in.inc1000Active, 309
<ModeGroup>.basic.in.inc100Active, 309
<ModeGroup>.basic.in.inc10Active, 308
<ModeGroup>.basic.in.inc1Active, 308
<ModeGroup>.basic.in.incVarActive, 309
<ModeGroup>.basic.in.jogModeActive, 308
<ModeGroup>.basic.in.mdaModeActive, 307
<ModeGroup>.basic.in.nckInternalJogActive, 308
<ModeGroup>.basic.in.ready, 307
<ModeGroup>.basic.in.refActive, 308
<ModeGroup>.basic.in.reposActive, 308
<ModeGroup>.basic.in.resetAllChanDone, 307
<ModeGroup>.basic.in.resetDone, 307
<ModeGroup>.basic.in.teachInActive, 308
<ModeGroup>.basic.out, 306
<ModeGroup>.basic.out.autoMode, 310
<ModeGroup>.basic.out.contTravers, 312
<ModeGroup>.basic.out.disableModeChangeover, 310
<ModeGroup>.basic.out.inc1, 311
<ModeGroup>.basic.out.inc10, 311
<ModeGroup>.basic.out.inc100, 311
<ModeGroup>.basic.out.inc1000, 311
<ModeGroup>.basic.out.inc10000, 312
<ModeGroup>.basic.out.incVar, 312
<ModeGroup>.basic.out.jogMode, 310
<ModeGroup>.basic.out.mdaMode, 310
<ModeGroup>.basic.out.refPoint, 311
<ModeGroup>.basic.out.repos, 310
<ModeGroup>.basic.out.reset, 309
<ModeGroup>.basic.out.singleBlockTypeA, 311
<ModeGroup>.basic.out.singleBlockTypeB, 311
<ModeGroup>.basic.out.stopAll, 309
<ModeGroup>.basic.out.stopAxesOnly, 310
<ModeGroup>.basic.out.teachIn, 310
<ModeGroup>.config, 306
<ModeGroup>.config.modeGroupActive, 307
<ModeGroup>.config.modeGroupNumber, 307
<ModeGroup>.config.registered, 307
<Nc>.analogInputs, 312
<Nc>.analogInputs.in, 313
<Nc>.analogInputs.out, 313
<Nc>.analogInputs[1].in.actualValue, 376
<Nc>.analogInputs[1].out.lock, 377
<Nc>.analogInputs[1].out.settingValue, 377
<Nc>.analogInputs[1].out.target, 377
<Nc>.analogInputs[2].in.actualValue, 376
<Nc>.analogInputs[2].out.lock, 377
<Nc>.analogInputs[2].out.settingValue, 378
<Nc>.analogInputs[2].out.target, 378
<Nc>.analogInputs[3].in.actualValue, 376
<Nc>.analogInputs[3].out.lock, 378
<Nc>.analogInputs[3].out.settingValue, 378
<Nc>.analogInputs[3].out.target, 378
<Nc>.analogInputs[4].in.actualValue, 376
<Nc>.analogInputs[4].out.lock, 378
<Nc>.analogInputs[4].out.settingValue, 379
<Nc>.analogInputs[4].out.target, 378
<Nc>.analogInputs[5].in.actualValue, 376
<Nc>.analogInputs[5].out.lock, 379
<Nc>.analogInputs[5].out.settingValue, 379
<Nc>.analogInputs[5].out.target, 379
<Nc>.analogInputs[6].in.actualValue, 377
<Nc>.analogInputs[6].out.lock, 379
<Nc>.analogInputs[6].out.settingValue, 379

<Nc>.analogInputs[6].out.target, 379
<Nc>.analogInputs[7].in.actualValue, 377
<Nc>.analogInputs[7].out.lock, 380
<Nc>.analogInputs[7].out.settingValue, 380
<Nc>.analogInputs[7].out.target, 380
<Nc>.analogInputs[8].in.actualValue, 377
<Nc>.analogInputs[8].out.lock, 380
<Nc>.analogInputs[8].out.settingValue, 380
<Nc>.analogInputs[8].out.target, 380
<Nc>.analogOutputs, 313
<Nc>.analogOutputs.in, 313
<Nc>.analogOutputs.out, 314
<Nc>.analogOutputs[1].in.setpointValue, 380
<Nc>.analogOutputs[1].out.lock, 382
<Nc>.analogOutputs[1].out.overwrite, 382
<Nc>.analogOutputs[1].out.settingValue, 382
<Nc>.analogOutputs[1].out.target, 382
<Nc>.analogOutputs[2].in.setpointValue, 381
<Nc>.analogOutputs[2].out.lock, 383
<Nc>.analogOutputs[2].out.overwrite, 382
<Nc>.analogOutputs[2].out.settingValue, 382
<Nc>.analogOutputs[2].out.target, 383
<Nc>.analogOutputs[3].in.setpointValue, 381
<Nc>.analogOutputs[3].out.lock, 383
<Nc>.analogOutputs[3].out.overwrite, 383
<Nc>.analogOutputs[3].out.settingValue, 383
<Nc>.analogOutputs[3].out.target, 383
<Nc>.analogOutputs[4].in.setpointValue, 381
<Nc>.analogOutputs[4].out.lock, 384
<Nc>.analogOutputs[4].out.overwrite, 384
<Nc>.analogOutputs[4].out.settingValue, 383
<Nc>.analogOutputs[4].out.target, 384
<Nc>.analogOutputs[5].in.setpointValue, 381
<Nc>.analogOutputs[5].out.lock, 384
<Nc>.analogOutputs[5].out.overwrite, 384
<Nc>.analogOutputs[5].out.settingValue, 384
<Nc>.analogOutputs[5].out.target, 384
<Nc>.analogOutputs[6].in.setpointValue, 381
<Nc>.analogOutputs[6].out.lock, 385
<Nc>.analogOutputs[6].out.overwrite, 385
<Nc>.analogOutputs[6].out.settingValue, 385
<Nc>.analogOutputs[6].out.target, 385
<Nc>.analogOutputs[7].in.setpointValue, 381
<Nc>.analogOutputs[7].out.lock, 386
<Nc>.analogOutputs[7].out.overwrite, 385
<Nc>.analogOutputs[7].out.settingValue, 385
<Nc>.analogOutputs[7].out.target, 385
<Nc>.analogOutputs[8].in.setpointValue, 381
<Nc>.analogOutputs[8].out.lock, 386
<Nc>.analogOutputs[8].out.overwrite, 386
<Nc>.analogOutputs[8].out.settingValue, 386
<Nc>.analogOutputs[8].out.target, 386
<Nc>.basic, 312
<Nc>.basic.in, 313
<Nc>.basic.in.collisionMonitoringActive, 317
<Nc>.basic.in.drivesCyclicOperation, 316
<Nc>.basic.in.drivesReady, 316
<Nc>.basic.in.emergencyStopActive, 315
<Nc>.basic.in.imperialMetricChangeCounter, 316
<Nc>.basic.in.imperialUnitsActive, 315
<Nc>.basic.in.internalAsup10StartReq, 317
<Nc>.basic.in.internalAsup9StartReq, 317
<Nc>.basic.in.internalAsupChanNumber, 317
<Nc>.basic.in.internalAsupModeGroupNumber, 317
<Nc>.basic.in.measuringProbe1Actuated, 315
<Nc>.basic.in.measuringProbe2Actuated, 315
<Nc>.basic.in.ncAlarmActive, 316
<Nc>.basic.in.ncReady, 316
<Nc>.basic.in.ncuAirTemperatureAlarmActive, 317
<Nc>.basic.in.ncuBatteryAlarmActive, 317
<Nc>.basic.in.ncuHeatSinkAlarmActive, 316
<Nc>.basic.in.ncuLinkActive, 315
<Nc>.basic.in.sinuOperateReady, 316
<Nc>.basic.in.swCams, 317
<Nc>.basic.in.swCams[1].minus, 318
<Nc>.basic.in.swCams[1].plus, 318
<Nc>.basic.in.swCams[10].minus, 320
<Nc>.basic.in.swCams[10].plus, 320
<Nc>.basic.in.swCams[11].minus, 320
<Nc>.basic.in.swCams[11].plus, 321
<Nc>.basic.in.swCams[12].minus, 321
<Nc>.basic.in.swCams[12].plus, 321
<Nc>.basic.in.swCams[13].minus, 321
<Nc>.basic.in.swCams[13].plus, 321
<Nc>.basic.in.swCams[14].minus, 321
<Nc>.basic.in.swCams[14].plus, 321
<Nc>.basic.in.swCams[15].minus, 322
<Nc>.basic.in.swCams[15].plus, 322
<Nc>.basic.in.swCams[16].minus, 322
<Nc>.basic.in.swCams[16].plus, 322
<Nc>.basic.in.swCams[17].minus, 322
<Nc>.basic.in.swCams[17].plus, 322
<Nc>.basic.in.swCams[18].minus, 322
<Nc>.basic.in.swCams[18].plus, 323
<Nc>.basic.in.swCams[19].minus, 323
<Nc>.basic.in.swCams[19].plus, 323
<Nc>.basic.in.swCams[2].minus, 318
<Nc>.basic.in.swCams[2].plus, 318
<Nc>.basic.in.swCams[20].minus, 323
<Nc>.basic.in.swCams[20].plus, 323
<Nc>.basic.in.swCams[21].minus, 323
<Nc>.basic.in.swCams[21].plus, 323
<Nc>.basic.in.swCams[22].minus, 324
<Nc>.basic.in.swCams[22].plus, 324

<Nc>.basic.in.swCams[23].minus, 324
<Nc>.basic.in.swCams[23].plus, 324
<Nc>.basic.in.swCams[24].minus, 324
<Nc>.basic.in.swCams[24].plus, 324
<Nc>.basic.in.swCams[25].minus, 324
<Nc>.basic.in.swCams[25].plus, 325
<Nc>.basic.in.swCams[26].minus, 325
<Nc>.basic.in.swCams[26].plus, 325
<Nc>.basic.in.swCams[27].minus, 325
<Nc>.basic.in.swCams[27].plus, 325
<Nc>.basic.in.swCams[28].minus, 325
<Nc>.basic.in.swCams[28].plus, 325
<Nc>.basic.in.swCams[29].minus, 326
<Nc>.basic.in.swCams[29].plus, 326
<Nc>.basic.in.swCams[3].minus, 318
<Nc>.basic.in.swCams[3].plus, 318
<Nc>.basic.in.swCams[30].minus, 326
<Nc>.basic.in.swCams[30].plus, 326
<Nc>.basic.in.swCams[31].minus, 326
<Nc>.basic.in.swCams[31].plus, 326
<Nc>.basic.in.swCams[32].minus, 326
<Nc>.basic.in.swCams[32].plus, 327
<Nc>.basic.in.swCams[4].minus, 318
<Nc>.basic.in.swCams[4].plus, 319
<Nc>.basic.in.swCams[5].minus, 319
<Nc>.basic.in.swCams[5].plus, 319
<Nc>.basic.in.swCams[6].minus, 319
<Nc>.basic.in.swCams[6].plus, 319
<Nc>.basic.in.swCams[7].minus, 319
<Nc>.basic.in.swCams[7].plus, 319
<Nc>.basic.in.swCams[8].minus, 320
<Nc>.basic.in.swCams[8].plus, 320
<Nc>.basic.in.swCams[9].minus, 320
<Nc>.basic.in.swCams[9].plus, 320
<Nc>.basic.out, 313
<Nc>.basic.out.emergencyStop, 327
<Nc>.basic.out.emergencyStopAck, 327
<Nc>.basic.out.enableReducedDynamics, 328
<Nc>.basic.out.internalAsupInfo, 328
<Nc>.basic.out.internalAsupReference, 328
<Nc>.basic.out.internalAsupStatus, 328
<Nc>.basic.out.keyOperatedSwitchPos0, 327
<Nc>.basic.out.keyOperatedSwitchPos1, 327
<Nc>.basic.out.keyOperatedSwitchPos2, 327
<Nc>.basic.out.keyOperatedSwitchPos3, 327
<Nc>.basic.out.selectIncFromModeGroup, 328
<Nc>.basic.out.transferActToolBasePos, 328
<Nc>.collision, 313
<Nc>.collision.out, 314
<Nc>.collision.out.deactivateFixtureAreasAuto, 387
<Nc>.collision.out.deactivateFixtureAreasJog, 387
<Nc>.collision.out.deactivateMachineAreasAuto, 387
<Nc>.collision.out.deactivateMachineAreasJog, 387
<Nc>.collision.out.deactivateToolAreasAuto, 387
<Nc>.collision.out.deactivateToolAreasJog, 387
<Nc>.collision.out.deactivateWorkpieceAreasAuto, 387
<Nc>.collision.out.deactivateWorkpieceAreasJog, 388
<Nc>.collisionProtAreas, 313
<Nc>.collisionProtAreas.in, 314
<Nc>.collisionProtAreas.out, 314
<Nc>.collisionProtAreas[0].in.active, 386
<Nc>.collisionProtAreas[0].out.activate, 386
<Nc>.config, 312
<Nc>.config.activeAxes, 315
<Nc>.config.activeChans, 314
<Nc>.config.activeModeGroups, 314
<Nc>.config.isVirtual, 314
<Nc>.config.maxAxis, 314
<Nc>.config.maxChan, 314
<Nc>.config.maxModeGroup, 314
<Nc>.config.ncCycleTimeout, 315
<Nc>.config.ncRunupTimeout, 315
<Nc>.config.registered, 314
<Nc>.digitalInputs, 312
<Nc>.digitalInputs.in, 313
<Nc>.digitalInputs.out, 313
<Nc>.digitalInputs[1].in.active, 330
<Nc>.digitalInputs[1].out.lock, 336
<Nc>.digitalInputs[1].out.set, 336
<Nc>.digitalInputs[10].in.active, 331
<Nc>.digitalInputs[10].out.lock, 338
<Nc>.digitalInputs[10].out.set, 338
<Nc>.digitalInputs[11].in.active, 331
<Nc>.digitalInputs[11].out.lock, 339
<Nc>.digitalInputs[11].out.set, 339
<Nc>.digitalInputs[12].in.active, 332
<Nc>.digitalInputs[12].out.lock, 339
<Nc>.digitalInputs[12].out.set, 339
<Nc>.digitalInputs[13].in.active, 332
<Nc>.digitalInputs[13].out.lock, 339
<Nc>.digitalInputs[13].out.set, 339
<Nc>.digitalInputs[14].in.active, 332
<Nc>.digitalInputs[14].out.lock, 339
<Nc>.digitalInputs[14].out.set, 340
<Nc>.digitalInputs[15].in.active, 332
<Nc>.digitalInputs[15].out.lock, 340
<Nc>.digitalInputs[15].out.set, 340
<Nc>.digitalInputs[16].in.active, 332
<Nc>.digitalInputs[16].out.lock, 340
<Nc>.digitalInputs[16].out.set, 340
<Nc>.digitalInputs[17].in.active, 332
<Nc>.digitalInputs[17].out.lock, 340

<Nc>.digitalInputs[17].out.set, 340
<Nc>.digitalInputs[18].in.active, 332
<Nc>.digitalInputs[18].out.lock, 341
<Nc>.digitalInputs[18].out.set, 341
<Nc>.digitalInputs[19].in.active, 333
<Nc>.digitalInputs[19].out.lock, 341
<Nc>.digitalInputs[19].out.set, 341
<Nc>.digitalInputs[2].in.active, 330
<Nc>.digitalInputs[2].out.lock, 336
<Nc>.digitalInputs[2].out.set, 336
<Nc>.digitalInputs[20].in.active, 333
<Nc>.digitalInputs[20].out.lock, 341
<Nc>.digitalInputs[20].out.set, 341
<Nc>.digitalInputs[21].in.active, 333
<Nc>.digitalInputs[21].out.lock, 341
<Nc>.digitalInputs[21].out.set, 342
<Nc>.digitalInputs[22].in.active, 333
<Nc>.digitalInputs[22].out.lock, 342
<Nc>.digitalInputs[22].out.set, 342
<Nc>.digitalInputs[23].in.active, 333
<Nc>.digitalInputs[23].out.lock, 342
<Nc>.digitalInputs[23].out.set, 342
<Nc>.digitalInputs[24].in.active, 333
<Nc>.digitalInputs[24].out.lock, 342
<Nc>.digitalInputs[24].out.set, 342
<Nc>.digitalInputs[25].in.active, 333
<Nc>.digitalInputs[25].out.lock, 343
<Nc>.digitalInputs[25].out.set, 343
<Nc>.digitalInputs[26].in.active, 334
<Nc>.digitalInputs[26].out.lock, 343
<Nc>.digitalInputs[26].out.set, 343
<Nc>.digitalInputs[27].in.active, 334
<Nc>.digitalInputs[27].out.lock, 343
<Nc>.digitalInputs[27].out.set, 343
<Nc>.digitalInputs[28].in.active, 334
<Nc>.digitalInputs[28].out.lock, 343
<Nc>.digitalInputs[28].out.set, 344
<Nc>.digitalInputs[29].in.active, 334
<Nc>.digitalInputs[29].out.lock, 344
<Nc>.digitalInputs[29].out.set, 344
<Nc>.digitalInputs[3].in.active, 330
<Nc>.digitalInputs[3].out.lock, 336
<Nc>.digitalInputs[3].out.set, 336
<Nc>.digitalInputs[30].in.active, 334
<Nc>.digitalInputs[30].out.lock, 344
<Nc>.digitalInputs[30].out.set, 344
<Nc>.digitalInputs[31].in.active, 334
<Nc>.digitalInputs[31].out.lock, 344
<Nc>.digitalInputs[31].out.set, 344
<Nc>.digitalInputs[32].in.active, 334
<Nc>.digitalInputs[32].out.lock, 345
<Nc>.digitalInputs[32].out.set, 345
<Nc>.digitalInputs[33].in.active, 335
<Nc>.digitalInputs[33].out.lock, 345
<Nc>.digitalInputs[33].out.set, 345
<Nc>.digitalInputs[34].in.active, 335
<Nc>.digitalInputs[34].out.lock, 345
<Nc>.digitalInputs[34].out.set, 345
<Nc>.digitalInputs[35].in.active, 335
<Nc>.digitalInputs[35].out.lock, 345
<Nc>.digitalInputs[35].out.set, 346
<Nc>.digitalInputs[36].in.active, 335
<Nc>.digitalInputs[36].out.lock, 346
<Nc>.digitalInputs[36].out.set, 346
<Nc>.digitalInputs[37].in.active, 335
<Nc>.digitalInputs[37].out.lock, 346
<Nc>.digitalInputs[37].out.set, 346
<Nc>.digitalInputs[38].in.active, 335
<Nc>.digitalInputs[38].out.lock, 346
<Nc>.digitalInputs[38].out.set, 346
<Nc>.digitalInputs[39].in.active, 335
<Nc>.digitalInputs[39].out.lock, 347
<Nc>.digitalInputs[39].out.set, 347
<Nc>.digitalInputs[4].in.active, 330
<Nc>.digitalInputs[4].out.lock, 337
<Nc>.digitalInputs[4].out.set, 337
<Nc>.digitalInputs[40].in.active, 336
<Nc>.digitalInputs[40].out.lock, 347
<Nc>.digitalInputs[40].out.set, 347
<Nc>.digitalInputs[5].in.active, 331
<Nc>.digitalInputs[5].out.lock, 337
<Nc>.digitalInputs[5].out.set, 337
<Nc>.digitalInputs[6].in.active, 331
<Nc>.digitalInputs[6].out.lock, 337
<Nc>.digitalInputs[6].out.set, 337
<Nc>.digitalInputs[7].in.active, 331
<Nc>.digitalInputs[7].out.lock, 337
<Nc>.digitalInputs[7].out.set, 338
<Nc>.digitalInputs[8].in.active, 331
<Nc>.digitalInputs[8].out.lock, 338
<Nc>.digitalInputs[8].out.set, 338
<Nc>.digitalInputs[9].in.active, 331
<Nc>.digitalInputs[9].out.lock, 338
<Nc>.digitalInputs[9].out.set, 338
<Nc>.digitalOutputs, 312
<Nc>.digitalOutputs.in, 313
<Nc>.digitalOutputs.out, 313
<Nc>.digitalOutputs[1].in.setpointState, 347
<Nc>.digitalOutputs[1].out.lock, 353
<Nc>.digitalOutputs[1].out.overwrite, 353
<Nc>.digitalOutputs[1].out.settingState, 353
<Nc>.digitalOutputs[1].out.target, 353
<Nc>.digitalOutputs[10].in.setpointState, 348
<Nc>.digitalOutputs[10].out.lock, 359

<Nc>.digitalOutputs[10].out.overwrite, 358
<Nc>.digitalOutputs[10].out.settingState, 358
<Nc>.digitalOutputs[10].out.target, 358
<Nc>.digitalOutputs[11].in.setpointState, 349
<Nc>.digitalOutputs[11].out.lock, 359
<Nc>.digitalOutputs[11].out.overwrite, 359
<Nc>.digitalOutputs[11].out.settingState, 359
<Nc>.digitalOutputs[11].out.target, 359
<Nc>.digitalOutputs[12].in.setpointState, 349
<Nc>.digitalOutputs[12].out.lock, 360
<Nc>.digitalOutputs[12].out.overwrite, 359
<Nc>.digitalOutputs[12].out.settingState, 359
<Nc>.digitalOutputs[12].out.target, 360
<Nc>.digitalOutputs[13].in.setpointState, 349
<Nc>.digitalOutputs[13].out.lock, 360
<Nc>.digitalOutputs[13].out.overwrite, 360
<Nc>.digitalOutputs[13].out.settingState, 360
<Nc>.digitalOutputs[13].out.target, 360
<Nc>.digitalOutputs[14].in.setpointState, 349
<Nc>.digitalOutputs[14].out.lock, 361
<Nc>.digitalOutputs[14].out.overwrite, 361
<Nc>.digitalOutputs[14].out.settingState, 360
<Nc>.digitalOutputs[14].out.target, 361
<Nc>.digitalOutputs[15].in.setpointState, 349
<Nc>.digitalOutputs[15].out.lock, 361
<Nc>.digitalOutputs[15].out.overwrite, 361
<Nc>.digitalOutputs[15].out.settingState, 361
<Nc>.digitalOutputs[15].out.target, 361
<Nc>.digitalOutputs[16].in.setpointState, 349
<Nc>.digitalOutputs[16].out.lock, 362
<Nc>.digitalOutputs[16].out.overwrite, 362
<Nc>.digitalOutputs[16].out.settingState, 362
<Nc>.digitalOutputs[16].out.target, 362
<Nc>.digitalOutputs[17].in.setpointState, 349
<Nc>.digitalOutputs[17].out.lock, 363
<Nc>.digitalOutputs[17].out.overwrite, 362
<Nc>.digitalOutputs[17].out.settingState, 362
<Nc>.digitalOutputs[17].out.target, 362
<Nc>.digitalOutputs[18].in.setpointState, 350
<Nc>.digitalOutputs[18].out.lock, 363
<Nc>.digitalOutputs[18].out.overwrite, 363
<Nc>.digitalOutputs[18].out.settingState, 363
<Nc>.digitalOutputs[18].out.target, 363
<Nc>.digitalOutputs[19].in.setpointState, 350
<Nc>.digitalOutputs[19].out.lock, 364
<Nc>.digitalOutputs[19].out.overwrite, 363
<Nc>.digitalOutputs[19].out.settingState, 363
<Nc>.digitalOutputs[19].out.target, 364
<Nc>.digitalOutputs[2].in.setpointState, 347
<Nc>.digitalOutputs[2].out.lock, 354
<Nc>.digitalOutputs[2].out.overwrite, 354
<Nc>.digitalOutputs[2].out.settingState, 354
<Nc>.digitalOutputs[2].out.target, 354
<Nc>.digitalOutputs[20].in.setpointState, 350
<Nc>.digitalOutputs[20].out.lock, 364
<Nc>.digitalOutputs[20].out.overwrite, 364
<Nc>.digitalOutputs[20].out.settingState, 364
<Nc>.digitalOutputs[20].out.target, 364
<Nc>.digitalOutputs[21].in.setpointState, 350
<Nc>.digitalOutputs[21].out.lock, 365
<Nc>.digitalOutputs[21].out.overwrite, 365
<Nc>.digitalOutputs[21].out.settingState, 364
<Nc>.digitalOutputs[21].out.target, 365
<Nc>.digitalOutputs[22].in.setpointState, 350
<Nc>.digitalOutputs[22].out.lock, 365
<Nc>.digitalOutputs[22].out.overwrite, 365
<Nc>.digitalOutputs[22].out.settingState, 365
<Nc>.digitalOutputs[22].out.target, 365
<Nc>.digitalOutputs[23].in.setpointState, 350
<Nc>.digitalOutputs[23].out.lock, 366
<Nc>.digitalOutputs[23].out.overwrite, 366
<Nc>.digitalOutputs[23].out.settingState, 366
<Nc>.digitalOutputs[23].out.target, 366
<Nc>.digitalOutputs[24].in.setpointState, 350
<Nc>.digitalOutputs[24].out.lock, 367
<Nc>.digitalOutputs[24].out.overwrite, 366
<Nc>.digitalOutputs[24].out.settingState, 366
<Nc>.digitalOutputs[24].out.target, 366
<Nc>.digitalOutputs[25].in.setpointState, 351
<Nc>.digitalOutputs[25].out.lock, 367
<Nc>.digitalOutputs[25].out.overwrite, 367
<Nc>.digitalOutputs[25].out.settingState, 367
<Nc>.digitalOutputs[25].out.target, 367
<Nc>.digitalOutputs[26].in.setpointState, 351
<Nc>.digitalOutputs[26].out.lock, 368
<Nc>.digitalOutputs[26].out.overwrite, 367
<Nc>.digitalOutputs[26].out.settingState, 367
<Nc>.digitalOutputs[26].out.target, 368
<Nc>.digitalOutputs[27].in.setpointState, 351
<Nc>.digitalOutputs[27].out.lock, 368
<Nc>.digitalOutputs[27].out.overwrite, 368
<Nc>.digitalOutputs[27].out.settingState, 368
<Nc>.digitalOutputs[27].out.target, 368
<Nc>.digitalOutputs[28].in.setpointState, 351
<Nc>.digitalOutputs[28].out.lock, 369
<Nc>.digitalOutputs[28].out.overwrite, 369
<Nc>.digitalOutputs[28].out.settingState, 368
<Nc>.digitalOutputs[28].out.target, 369
<Nc>.digitalOutputs[29].in.setpointState, 351
<Nc>.digitalOutputs[29].out.lock, 369
<Nc>.digitalOutputs[29].out.overwrite, 369
<Nc>.digitalOutputs[29].out.settingState, 369
<Nc>.digitalOutputs[29].out.target, 369
<Nc>.digitalOutputs[3].in.setpointState, 347

<Nc>.digitalOutputs[3].out.lock, 355
<Nc>.digitalOutputs[3].out.overwrite, 354
<Nc>.digitalOutputs[3].out.settingState, 354
<Nc>.digitalOutputs[3].out.target, 354
<Nc>.digitalOutputs[30].in.setpointState, 351
<Nc>.digitalOutputs[30].out.lock, 370
<Nc>.digitalOutputs[30].out.overwrite, 370
<Nc>.digitalOutputs[30].out.settingState, 370
<Nc>.digitalOutputs[30].out.target, 370
<Nc>.digitalOutputs[31].in.setpointState, 351
<Nc>.digitalOutputs[31].out.lock, 371
<Nc>.digitalOutputs[31].out.overwrite, 370
<Nc>.digitalOutputs[31].out.settingState, 370
<Nc>.digitalOutputs[31].out.target, 370
<Nc>.digitalOutputs[32].in.setpointState, 352
<Nc>.digitalOutputs[32].out.lock, 371
<Nc>.digitalOutputs[32].out.overwrite, 371
<Nc>.digitalOutputs[32].out.settingState, 371
<Nc>.digitalOutputs[32].out.target, 371
<Nc>.digitalOutputs[33].in.setpointState, 352
<Nc>.digitalOutputs[33].out.lock, 372
<Nc>.digitalOutputs[33].out.overwrite, 371
<Nc>.digitalOutputs[33].out.settingState, 371
<Nc>.digitalOutputs[33].out.target, 372
<Nc>.digitalOutputs[34].in.setpointState, 352
<Nc>.digitalOutputs[34].out.lock, 372
<Nc>.digitalOutputs[34].out.overwrite, 372
<Nc>.digitalOutputs[34].out.settingState, 372
<Nc>.digitalOutputs[34].out.target, 372
<Nc>.digitalOutputs[35].in.setpointState, 352
<Nc>.digitalOutputs[35].out.lock, 373
<Nc>.digitalOutputs[35].out.overwrite, 373
<Nc>.digitalOutputs[35].out.settingState, 372
<Nc>.digitalOutputs[35].out.target, 373
<Nc>.digitalOutputs[36].in.setpointState, 352
<Nc>.digitalOutputs[36].out.lock, 373
<Nc>.digitalOutputs[36].out.overwrite, 373
<Nc>.digitalOutputs[36].out.settingState, 373
<Nc>.digitalOutputs[36].out.target, 373
<Nc>.digitalOutputs[37].in.setpointState, 352
<Nc>.digitalOutputs[37].out.lock, 374
<Nc>.digitalOutputs[37].out.overwrite, 374
<Nc>.digitalOutputs[37].out.settingState, 374
<Nc>.digitalOutputs[37].out.target, 374
<Nc>.digitalOutputs[38].in.setpointState, 352
<Nc>.digitalOutputs[38].out.lock, 375
<Nc>.digitalOutputs[38].out.overwrite, 374
<Nc>.digitalOutputs[38].out.settingState, 374
<Nc>.digitalOutputs[38].out.target, 374
<Nc>.digitalOutputs[39].in.setpointState, 353
<Nc>.digitalOutputs[39].out.lock, 375
<Nc>.digitalOutputs[39].out.overwrite, 375
<Nc>.digitalOutputs[39].out.settingState, 375
<Nc>.digitalOutputs[39].out.target, 375
<Nc>.digitalOutputs[4].in.setpointState, 348
<Nc>.digitalOutputs[4].out.lock, 355
<Nc>.digitalOutputs[4].out.overwrite, 355
<Nc>.digitalOutputs[4].out.settingState, 355
<Nc>.digitalOutputs[4].out.target, 355
<Nc>.digitalOutputs[40].in.setpointState, 353
<Nc>.digitalOutputs[40].out.lock, 376
<Nc>.digitalOutputs[40].out.overwrite, 375
<Nc>.digitalOutputs[40].out.settingState, 375
<Nc>.digitalOutputs[40].out.target, 376
<Nc>.digitalOutputs[5].in.setpointState, 348
<Nc>.digitalOutputs[5].out.lock, 356
<Nc>.digitalOutputs[5].out.overwrite, 355
<Nc>.digitalOutputs[5].out.settingState, 355
<Nc>.digitalOutputs[5].out.target, 356
<Nc>.digitalOutputs[6].in.setpointState, 348
<Nc>.digitalOutputs[6].out.lock, 356
<Nc>.digitalOutputs[6].out.overwrite, 356
<Nc>.digitalOutputs[6].out.settingState, 356
<Nc>.digitalOutputs[6].out.target, 356
<Nc>.digitalOutputs[7].in.setpointState, 348
<Nc>.digitalOutputs[7].out.lock, 357
<Nc>.digitalOutputs[7].out.overwrite, 357
<Nc>.digitalOutputs[7].out.settingState, 356
<Nc>.digitalOutputs[7].out.target, 357
<Nc>.digitalOutputs[8].in.setpointState, 348
<Nc>.digitalOutputs[8].out.lock, 357
<Nc>.digitalOutputs[8].out.overwrite, 357
<Nc>.digitalOutputs[8].out.settingState, 357
<Nc>.digitalOutputs[8].out.target, 357
<Nc>.digitalOutputs[9].in.setpointState, 348
<Nc>.digitalOutputs[9].out.lock, 358
<Nc>.digitalOutputs[9].out.overwrite, 358
<Nc>.digitalOutputs[9].out.settingState, 358
<Nc>.digitalOutputs[9].out.target, 358
<Nc>.handwheels, 312
<Nc>.handwheels.in, 314
<Nc>.handwheels[1].in.movedCounter, 328
<Nc>.handwheels[1].in.stationary, 328
<Nc>.handwheels[2].in.movedCounter, 329
<Nc>.handwheels[2].in.stationary, 329
<Nc>.handwheels[3].in.movedCounter, 329
<Nc>.handwheels[3].in.stationary, 329
<Nc>.handwheels[4].in.movedCounter, 329
<Nc>.handwheels[4].in.stationary, 329
<Nc>.handwheels[5].in.movedCounter, 329
<Nc>.handwheels[5].in.stationary, 330
<Nc>.handwheels[6].in.movedCounter, 330
<Nc>.handwheels[6].in.stationary, 330
<Relocate>.ack, 407

<Relocate>.ack.in, 407
<Relocate>.ack.in.busy, 412
<Relocate>.ack.in.done, 412
<Relocate>.ack.in.error, 412
<Relocate>.ack.in.status, 412
<Relocate>.ack.out, 407
<Relocate>.ack.out.execute, 412
<Relocate>.ack.out.status, 413
<Relocate>.ack.out.toolLocation, 412
<Relocate>.ack.out.toolMagazine, 412
<Relocate>.ack.out.totalComplete, 412
<Relocate>.ack.out.totalFailed, 412
<Relocate>.command, 407
<Relocate>.command.in, 407
<Relocate>.command.in.basic, 408
<Relocate>.command.in.basic.chan, 409
<Relocate>.command.in.basic.ncCommand, 408
<Relocate>.command.in.basic.relocate, 408
<Relocate>.command.in.basic.tool, 409
<Relocate>.command.in.basic.toolSourceLocation, 409
<Relocate>.command.in.basic.toolSourceMagazine, 409
<Relocate>.command.in.basic.toolTargetLocation, 409
<Relocate>.command.in.basic.toolTargetMagazine, 409
<Relocate>.command.in.multitoolProperties, 408
<Relocate>.command.in.multitoolProperties.distance, 411
<Relocate>.command.in.multitoolProperties.distanceType, 411
<Relocate>.command.in.multitoolProperties.numberOfLocations, 412
<Relocate>.command.in.multitoolRequest, 407
<Relocate>.command.in.options, 408
<Relocate>.command.in.options.byNcProg, 409
<Relocate>.command.in.request, 407
<Relocate>.command.in.toolProperties, 408
<Relocate>.command.in.toolProperties.toolLocationType, 409
<Relocate>.command.in.toolProperties.toolSizeBottom, 410
<Relocate>.command.in.toolProperties.toolSizeLeft, 410
<Relocate>.command.in.toolProperties.toolSizeRight, 410
<Relocate>.command.in.toolProperties.toolSizeTop, 410
<Relocate>.command.in.toolStatus, 408
<Relocate>.command.in.toolStatus.active, 410
<Relocate>.command.in.toolStatus.changeActive, 410
<Relocate>.command.in.toolStatus.changeOneToOne, 411
<Relocate>.command.in.toolStatus.disabled, 410
<Relocate>.command.in.toolStatus.disableStateIgnored, 411
<Relocate>.command.in.toolStatus.enabled, 410
<Relocate>.command.in.toolStatus.fixedLocation, 410
<Relocate>.command.in.toolStatus.manualTool, 411
<Relocate>.command.in.toolStatus.masterTool, 411
<Relocate>.command.in.toolStatus.measured, 410
<Relocate>.command.in.toolStatus.prewarning, 410
<Relocate>.command.in.toolStatus.toBeLoaded, 411
<Relocate>.command.in.toolStatus.toBeUnloaded, 411
<Relocate>.command.in.toolStatus.toolAtDisabledLocation, 411
<Relocate>.command.in.toolStatus.toolInBuffer, 411
<Relocate>.command.in.toolStatus.used, 410
<Relocate>.command.in.userParameters, 408
<Relocate>.config, 407
<Relocate>.config.registered, 408
<Relocate>.config.toaUnit, 408
<ToaUnit>.config, 420
<ToaUnit>.config.associatedChans, 420
<ToaUnit>.config.bufferMagazine, 420
<ToaUnit>.config.loadingMagazine, 420
<ToaUnit>.config.magazineCount, 420
<ToaUnit>.config.magazines, 420
<ToaUnit>.config.magazines[0].locationCount, 419
<ToaUnit>.config.magazines[0].magazineNumber, 419
<ToaUnit>.config.magazines[0].magazineType, 419
<ToaUnit>.config.maxToolholder, 420
<ToaUnit>.config.registered, 420
<ToaUnit>.config.toaUnitNumber, 420
<Toolholder>.ack, 388
<Toolholder>.ack.in, 388
<Toolholder>.ack.in.busy, 405
<Toolholder>.ack.in.done, 405
<Toolholder>.ack.in.error, 406
<Toolholder>.ack.in.status, 406
<Toolholder>.ack.out, 388
<Toolholder>.ack.out.execute, 406
<Toolholder>.ack.out.newMultitoolLocation, 406
<Toolholder>.ack.out.newToolLocation, 406
<Toolholder>.ack.out.newToolMagazine, 406
<Toolholder>.ack.out.oldToolLocation, 407
<Toolholder>.ack.out.oldToolMagazine, 406
<Toolholder>.ack.out.status, 407

<Toolholder>.ack.out.totalComplete, 406
<Toolholder>.ack.out.totalFailed, 406
<Toolholder>.command, 388
<Toolholder>.command.in, 388
<Toolholder>.command.in.basic, 389
<Toolholder>.command.in.basic.chan, 391
<Toolholder>.command.in.basic.ncCommand, 390
<Toolholder>.command.in.basic.newTool, 391
<Toolholder>.command.in.basic.newToolSourceLocation, 392
<Toolholder>.command.in.basic.newToolSourceMagazine, 392
<Toolholder>.command.in.basic.newToolTargetLocation, 391
<Toolholder>.command.in.basic.newToolTargetMagazine, 391
<Toolholder>.command.in.basic.oldToolSourceLocation, 392
<Toolholder>.command.in.basic.oldToolSourceMagazine, 392
<Toolholder>.command.in.basic.oldToolTargetLocation, 392, 393
<Toolholder>.command.in.basic.oldToolTargetMagazine, 392
<Toolholder>.command.in.basic.perform, 390
<Toolholder>.command.in.basic.position, 390, 391
<Toolholder>.command.in.basic.prepare, 390
<Toolholder>.command.in.multitoolCommand, 389
<Toolholder>.command.in.multitoolCommand.newTool, 399
<Toolholder>.command.in.multitoolCommand.newToolLocation, 399
<Toolholder>.command.in.multitoolCommand.oldTool, 399
<Toolholder>.command.in.multitoolCommand.oldToolLocation, 400
<Toolholder>.command.in.multitoolCommand.position, 399
<Toolholder>.command.in.multitoolProperties, 389
<Toolholder>.command.in.multitoolProperties.distance, 400
<Toolholder>.command.in.multitoolProperties.distanceType, 400
<Toolholder>.command.in.multitoolProperties.numberOfLocations, 400
<Toolholder>.command.in.multitoolProperties.toolLocationType, 400, 401
<Toolholder>.command.in.multitoolProperties.toolOriginLocation, 402
<Toolholder>.command.in.multitoolProperties.toolOriginMagazine, 402
<Toolholder>.command.in.multitoolProperties.toolSizeBottom, 401, 402
<Toolholder>.command.in.multitoolProperties.toolSizeLeft, 401
<Toolholder>.command.in.multitoolProperties.toolSizeRight, 401
<Toolholder>.command.in.multitoolProperties.toolSizeTop, 401
<Toolholder>.command.in.multitoolRequest, 388, 389
<Toolholder>.command.in.multitoolStatus, 389
<Toolholder>.command.in.multitoolStatus.active, 402
<Toolholder>.command.in.multitoolStatus.changeActive, 403
<Toolholder>.command.in.multitoolStatus.changeOneToOne, 405
<Toolholder>.command.in.multitoolStatus.disabled, 403
<Toolholder>.command.in.multitoolStatus.disableStateIgnored, 404
<Toolholder>.command.in.multitoolStatus.enabled, 402
<Toolholder>.command.in.multitoolStatus.fixedLocation, 403, 404
<Toolholder>.command.in.multitoolStatus.manualTool, 405
<Toolholder>.command.in.multitoolStatus.masterTool, 405
<Toolholder>.command.in.multitoolStatus.measured, 403
<Toolholder>.command.in.multitoolStatus.prewarning, 403
<Toolholder>.command.in.multitoolStatus.toBeLoaded, 405
<Toolholder>.command.in.multitoolStatus.toBeUnloaded, 404
<Toolholder>.command.in.multitoolStatus.toolAtDisabledLocation, 405
<Toolholder>.command.in.multitoolStatus.toolInBuffer, 404
<Toolholder>.command.in.multitoolStatus.used, 404
<Toolholder>.command.in.options, 389
<Toolholder>.command.in.options.byNcProg, 394
<Toolholder>.command.in.options.newManualTool, 393
<Toolholder>.command.in.options.noNewTool, 393
<Toolholder>.command.in.options.oldManualTool, 393
<Toolholder>.command.in.options.oldToolNotInToolholder, 393

<Toolholder>.command.in.options.rejectingToolAllowed, 394
<Toolholder>.command.in.options.toolRemainsInToolholder, 394
<Toolholder>.command.in.request, 388
<Toolholder>.command.in.toolProperties, 389
<Toolholder>.command.in.toolProperties.toolLocationType, 394
<Toolholder>.command.in.toolProperties.toolOriginLocation, 395
<Toolholder>.command.in.toolProperties.toolOriginMagazine, 395
<Toolholder>.command.in.toolProperties.toolSizeBottom, 395
<Toolholder>.command.in.toolProperties.toolSizeLeft, 394
<Toolholder>.command.in.toolProperties.toolSizeRight, 394, 395
<Toolholder>.command.in.toolProperties.toolSizeTop, 395
<Toolholder>.command.in.toolStatus, 389
<Toolholder>.command.in.toolStatus.active, 395, 396
<Toolholder>.command.in.toolStatus.changeActive, 397
<Toolholder>.command.in.toolStatus.changeOneToOne, 398
<Toolholder>.command.in.toolStatus.disabled, 396
<Toolholder>.command.in.toolStatus.disableStateIgnored, 397
<Toolholder>.command.in.toolStatus.enabled, 396
<Toolholder>.command.in.toolStatus.fixedLocation, 397
<Toolholder>.command.in.toolStatus.manualTool, 398
<Toolholder>.command.in.toolStatus.masterTool, 398
<Toolholder>.command.in.toolStatus.measured, 396
<Toolholder>.command.in.toolStatus.prewarning, 396
<Toolholder>.command.in.toolStatus.toBeLoaded, 398
<Toolholder>.command.in.toolStatus.toBeUnloaded, 398
<Toolholder>.command.in.toolStatus.toolAtDisabledLocation, 399
<Toolholder>.command.in.toolStatus.toolInBuffer, 397
<Toolholder>.command.in.toolStatus.used, 397
<Toolholder>.command.in.userParameters, 389
<Toolholder>.config, 388
<Toolholder>.config.bufferLocation, 390
<Toolholder>.config.bufferMagazine, 390
<Toolholder>.config.registered, 389
<Toolholder>.config.toaUnit, 390
<Toolholder>.config.toolholder, 390

A

Axis
.slsLimit, 633

C

Chan
.circularJogTraversActive, 682
.configuredStopAtBlockEnd, 686
.configuredStopFunc, 730
.configuredStopFuncActive, 686
.contourHandwheelSimInvertDir, 732
.dryRunFeedrateActive, 689
.singleBlockStopAtBlockEnd, 717

H

hmi
.error, 985
.mcpAddress, 962
.numberOfTableLines, 986
.ready, 985
.status, 986
.tableLines, 986
.version, 987
Hmi
.activateLed, 987
.activeOperatingAreaNumber, 959
.analogSpindle1Percentage, 966
.analogSpindle2Percentage, 967
.brightenScreen, 967
.busy, 973
.cancelAlarmsCleared, 965
.cancelKeyPressed, 960
.clearCancelAlarms, 967
.clearRecallAlarms, 968
.currentChanNumber, 960
.currentScreenNumber, 960
.darkenScreen, 968
.dataTransferActive, 961
.directKeyAddress, 961
.disable, 987
.disableKeys, 969
.disableTeachInTransfer, 970
.displayWcsValues, 971

.error, 978
.errorCode, 973, 980
.Hmi, 976
.identified, 977
.inputDeviceActive, 962
.mcsWcsSwitched, 962
.parameter1, 974
.permitRemoteMonitoringOnly, 971
.plcHardkeySelection, 965
.pressed, 987
.progList, 982
.progNumber, 983
.recallAlarmsCleared, 966
.rejectRemoteAccess, 972
.request, 985
.saveActionLog, 972
.screenChangeActive, 963
.screenDark, 963
.selected, 981
.selection, 975
.simActive, 964
.switchoverTcuDisabled, 964
.tcuIndex, 965
hmiAxis
.handwheelSelected, 988
HmiAxis
.progTestRequested, 989
.progTestSuppressionRequested, 990
hmiChan
.contourHandwheelSelected, 993
.geoAxis1.in.handwheelSelected, 1006
.geoAxis2.in.handwheelSelected, 1006
.geoAxis3.in.handwheelSelected, 1006
.orientationAxis1.in, 1007
.orientationAxis2.in, 1007
.orientationAxis3.in, 1007
HmiChan
.activeNumber, 1004
.activeType, 1005
.atStopSetOvrZeroSelected, 992
.configuredStopFuncSelected, 992
.dryRunSelected, 994
.handwheelOffsetDrfSelected, 995
.m01AssociatedAuxFuncSelected, 996
.m01Selected, 996
.progTestSelected, 997
.rapidTraversOvrSelected, 998
.reposModeChanged, 1000
.selectedKeyActuated, 1001
.selectedNumber, 1002
.selectedType, 1003
.Signal, 999
.skipBlockLevel0...9, 1001
HmiModeGroup
.autoModeRequested, 1008
.contTraversRequested, 1009
.disableModeChangeover, 1019
.inc10000Requested, 1014
.inc1000Requested, 1013
.inc100Requested, 1012
.inc10Requested, 1011
.inc1Requested, 1010
.incVarRequested, 1015
.jogModeRequested, 1016
.mdaModeRequested, 1009
.refRequested, 1017
.reposRequested, 1017
.teachInRequested, 1018
HmiNc
.fixtureAreasAutoDeactivated, 1020
.fixtureAreasJogDeactivated, 1020
.machineAreasAutoDeactivated, 1019
.machineAreasJogDeactivated, 1020
.toolAreasAutoDeactivated, 1020
.toolAreasJogDeactivated, 1020
.workpieceAreasAutoDeactivated, 1020
.workpieceAreasJogDeactivated, 1020

L

loadingPoint
.byNcProg, 1032
.distance, 1039
.distanceType, 1039
.numberOfLocations, 1039
.toolLocationType, 1032
.toolSizeBottom, 1032
.toolSizeLeft, 1032
.toolSizeRight, 1033
.toolSizeTop, 1033
LoadingPoint
.busy, 1025
.chan, 1029
.done, 1025
.execute, 1026
.load, 1029
.loadingLocation, 1024
.loadingMagazine, 1024
.multitoolLocation, 1026
.multitoolRequest, 1028
.position, 1030
.request, 1028
.Signal, 1030

.status, 1026
.toolLocation, 1027
.toolMagazine, 1027
.toolStatus.active, 1033
.toolStatus.changeActive, 1034
.toolStatus.changeOneToOne, 1034
.toolStatus.disabled, 1034
.toolStatus.disableStateIgnored, 1035
.toolStatus.enabled, 1035
.toolStatus.fixedLocation, 1035
.toolStatus.manualTool, 1036
.toolStatus.masterTool, 1036
.toolStatus.measured, 1036
.toolStatus.prewarning, 1037
.toolStatus.toBeLoaded, 1037
.toolStatus.toBeUnloaded, 1037
.toolStatus.toolAtDisabledLocation, 1038
.toolStatus.toolInBuffer, 1038
.toolStatus.used, 1038
.toolTargetLocation, 1031
.toolTargetMagazine, 1031
.totalComplete, 1027
.totalFailed, 1028
.unload, 1031
.userParameters, 1029
LSINU_GenerateSinuUserAlarm, 1283
LSINU_GetReinitSinuUserAlarms, 1289
LSINU_GetSinuUserAlarmCounters, 1291
LSINU_Hmi, 63
LSINU_Powerride, 1294
LED 色板, 1299
示例：调用 LSINU_Powerrride, 1301
通过 MCP 嵌套 Powerride 示例, 1299
LSINU_PowerrideConfig, 1304
LSINU_ReadGUD（读取 GUD 变量）, 1306
LSINU_ReadVar（读取 NC 变量）, 1313
LSINU_TransferSelData, 1322
LSINU_typeAlarmInternalData, 1340
LSINU_typePowerrideConfig, 1340
LSINU_typePowerrideConfigPos, 1342
LSINU_typePowerrideInputs, 1343
LSINU_typePowerrideOutputs, 1344
LSINU_WriteVar（写入 NC 变量, 1331

M

MD22510, 765

N

Nc
.activate, 948
.active, 929, 946
.actualValue, 937
.collisionMonitoringActive, 913
.deactivateFixtureAreasAuto, 949
.deactivateFixtureAreasJog, 950
.deactivateMachineAreasAuto, 951
.deactivateMachineAreasJog, 953
.deactivateToolAreasAuto, 954
.deactivateToolAreasJog, 955
.deactivateWorkpieceAreasAuto, 956
.deactivateWorkpieceAreasJog, 957
.drivesCyclicOperation, 914
.drivesReady, 914
.emergencyStop, 922
.emergencyStopAck, 923
.emergencyStopActive, 915
.enableReducedDynamics, 924
.imperialMetricChangeCounter, 915
.imperialUnitsActive, 916
.keyOperatedSwitchPos0, 924
.keyOperatedSwitchPos1, 925
.keyOperatedSwitchPos2, 926
.keyOperatedSwitchPos3, 926
.lock, 930, 933, 938, 943
.measuringProbe1Actuated, 916
.measuringProbe2Actuated, 917
.movedCounter, 928
.ncAlarmActive, 917
.ncReady, 918
.ncuAirTemperatureAlarmActive, 919
.ncuHeatSinkAlarmActive, 920
.overwrite, 934, 944
.selectIncFromModeGroup, 927
.set, 931
.setpointState, 932
.setpointValue, 941
.settingState, 935
.settingValue, 940, 942
.sinuOperateReady, 920
.stationary, 929
.swCams[x].minus, 920
.swCams[x].plus, 921
.target, 936, 939, 945
.transferActToolBasePos, 928
NC
故障, 115
NCK_typeAxis, 440

NCK_typeChan, 673
NCK_typeHmi, 958
NCK_typeHmiAxis, 988
NCK_typeHmiChan, 991
NCK_typeHmiModeGroup, 1008
NCK_typeHmiNc, 1019
NCK_typeLoadingPoint, 1023
NCK_typeModeGroup, 874
NCK_typeNc, 912
NCK_typeRelocate, 1040
NCK_typeToaUnit, 1022
NCK_typeToolholder, 1055

P

PLC 基本程序, 41

R

Relocate
.busy, 1040
.byNcProg, 1046
.chan, 1044
.distance, 1054
.distanceType, 1054
.done, 1041
.error, 1041
.execute, 1041
.multitoolRequest, 1043
.numberOfLocations, 1054
.relocate, 1045
.request, 1044
.status, 1041, 1042
.tool, 1045
.toolLocation, 1042
.toolLocationType, 1047
.toolMagazine, 1042
.toolSizeBottom, 1047
.toolSizeLeft, 1047
.toolSizeRight, 1048
.toolSizeTop, 1048
.toolSourceLocation, 1045
.toolSourceMagazine, 1045
.toolStatus.active, 1048
.toolStatus.changeActive, 1049
.toolStatus.changeOneToOne, 1049
.toolStatus.disabled, 1049
.toolStatus.disableStateIgnored, 1050
.toolStatus.enabled, 1050
.toolStatus.fixedLocation, 1050
.toolStatus.manualTool, 1051
.toolStatus.masterTool, 1051
.toolStatus.measured, 1051
.toolStatus.prewarning, 1052
.toolStatus.toBeLoaded, 1052
.toolStatus.toBeUnloaded, 1052
.toolStatus.toolAtDisabledLocation, 1053
.toolStatus.toolInBuffer, 1053
.toolStatus.used, 1053
.toolTargetLocation, 1046
.toolTargetMagazine, 1046
.totalComplete, 1043
.totalFailed, 1043
.userParameters, 1044

S

SINU_AsyncTransfer, 1092
SINU_CompleteRegistration, 1090
SINU_CtrlStarDelta, 1247
SINU_Init, 1088
SINU_NcActivateBlockSearch, 1134, 1148
SINU_NcAssignInterrupt, 1140
SINU_NcCreateCuttingEdge, 1193
SINU_NcCreateCuttingEdgeWithD, 1195
SINU_NcCreateMultiTool, 1181
SINU_NcCreateTool, 1159
SINU_NcCreateToolByName, 1161
SINU_NcDeleteCuttingEdge, 1197
SINU_NcDeleteMultiTool, 1190
SINU_NcDeleteTool, 1179
SINU_NcGetToolNumber, 1164
SINU_NcLogin, 1130
SINU_NcLogout, 1133
SINU_NcMoveTool, 1173
SINU_NcPositionMagazine, 1199
SINU_NcPositionMultitool, 1184
SINU_NcReconfigureMachineData, 1138
SINU_NcReset, 1136
SINU_NcResetActiveToolStatus, 1177
SINU_NcResetMonitoringValues, 1157
SINU_NcRetractTool, 1171
SINU_NcSearchLoadingLocation, 1206
SINU_NcSearchLocation, 1202
SINU_NcSearchMultitoolLocation, 1187
SINU_NcSearchTool, 1167
SINU_NcSelectPartProgram, 1144
SINU_NcSetSemaphore, 1152
SINU_NcSetSemaphoreByToa, 1212
SINU_NcSetUserDataActive, 1150
SINU_NcSetWorkpieceCounterInc, 1209
SINU_OscillateSpindle, 1230
SINU_PositionSpindle, 1236

SINU_ProcessNckEvent, 1091
SINU_PublishUserProgramVersion, 1096
SINU_ReadCompileCycleData, 1259
SINU_ReadUserDataFloat, 1256
SINU_ReadUserDataHex, 1254
SINU_ReadUserDataInt, 1252
SINU_RegisterAxis, 1110
SINU_RegisterAxisByInst, 1113
SINU_RegisterAxisByInstWithHmi, 1115
SINU_RegisterAxisByName, 1117
SINU_RegisterAxisByNameWithHmi, 1119
SINU_RegisterAxisWithHmi, 1111
SINU_RegisterChan, 1106
SINU_RegisterChanWithHmi, 1107
SINU_RegisterLoadingPoint, 1123
SINU_RegisterModeGroup, 1102
SINU_RegisterModeGroupWithHmi, 1103
SINU_RegisterNc, 1098
SINU_RegisterNcWithHmi, 1099
SINU_RegisterRelocate, 1126
SINU_RegisterToaUnit, 1121
SINU_RegisterToolholder, 1124
SINU_RequestAsup, 1094
SINU_ReRegisterAxis, 1120
SINU_ReRegisterChan, 1109
SINU_ReRegisterLoadingPoint, 1127
SINU_ReRegisterModeGroup, 1105
SINU_ReRegisterNc, 1101
SINU_ReRegisterRelocate, 1129
SINU_ReRegisterToolholder, 1128
SINU_RotateSpindle, 1241
SINU_TransferHandwheelData, 1279
SINU_TraverseIndexingAxis, 1223
SINU_TraversePosAxis, 1216
SINU_typeHandwheelData, 1282
SINU_Update, 1089
SINU_UpdateEthernetMcp, 1271
SINU_WriteCompileCycleData, 1261
SINU_WriteRecordOverEthernet, 1263
SiNUMERIK Operate
  LSINU_Hmi, 63

T

toolholder
  .oldToolSourceLocation, 1063
  .oldToolSourceMagazine, 1063
Toolholder
  .bufferLocation, 1056
  .bufferMagazine, 1056
  .busy, 1057
  .byNcProg, 1065
  .chan, 1062
  .distance, 1075
  .distanceType, 1075
  .done, 1057
  .error, 1058
  .execute, 1058
  .multitoolRequest, 1061
  .multitoolStatus.active, 1078
  .multitoolStatus.changeActive, 1078
  .multitoolStatus.changeOneToOne, 1078
  .multitoolStatus.disabled, 1079
  .multitoolStatus.disableStateIgnored, 1079
  .multitoolStatus.enabled, 1079
  .multitoolStatus.fixedLocation, 1080
  .multitoolStatus.manualTool, 1080
  .multitoolStatus.masterTool, 1080
  .multitoolStatus.measured, 1081
  .multitoolStatus.prewarning, 1081
  .multitoolStatus.toBeLoaded, 1081
  .multitoolStatus.toBeUnloaded, 1082
  .multitoolStatus.toolAtDisabledLocation, 1082
  .multitoolStatus.toolInBuffer, 1082
  .multitoolStatus.used, 1083
  .newManualTool, 1065
  .newMultitoolLocation, 1058
  .newTool, 1062, 1074
  .newToolLocation, 1059, 1074
  .newToolMagazine, 1059
  .newToolSourceLocation, 1062
  .newToolSourceMagazine, 1062
  .newToolTargetLocation, 1063
  .newToolTargetMagazine, 1063
  .noNewTool, 1065
  .numberOfLocations, 1076
  .oldManualTool, 1066
  .oldTool, 1074
  .oldToolLocation, 1059, 1075
  .oldToolMagazine, 1059
  .oldToolNotInToolholder, 1066
  .oldToolTargetLocation, 1064
  .oldToolTargetMagazine, 1064
  .perform, 1064
  .position, 1064, 1075
  .prepare, 1065
  .request, 1061
  .status, 1058, 1060
  .toolLocationType, 1066, 1076
  .toolOriginLocation, 1067, 1076
  .toolOriginMagazine, 1067, 1076
  .toolRemainsInToolholder, 1066
  .toolSizeBottom, 1067, 1077
  .toolSizeLeft, 1067, 1077

.toolSizeRight, 1068, 1077
.toolSizeTop, 1068, 1077
.toolStatus.active, 1068
.toolStatus.changeActive, 1069
.toolStatus.changeOneToOne, 1069
.toolStatus.disabled, 1069
.toolStatus.disableStateIgnored, 1070
.toolStatus.enabled, 1070
.toolStatus.fixedLocation, 1070
.toolStatus.manualTool, 1071
.toolStatus.masterTool, 1071
.toolStatus.measured, 1071
.toolStatus.prewarning, 1072
.toolStatus.toBeLoaded, 1072
.toolStatus.toBeUnloaded, 1072
.toolStatus.toolAtDisabledLocation, 1073
.toolStatus.toolInBuffer, 1073
.toolStatus.used, 1073
.totalComplete, 1060
.totalFailed, 1061
.userParameters, 1061

程

程序块
复制操作时的冲突, 48

升

升级项目, 55

数

数据类型"HMI"
NCK_typeHmi, 958
NCK_typeHmiAxis, 988
NCK_typeHmiChan, 991
NCK_typeHmiModeGroup, 1008
NCK_typeHmiNc, 1019
数据类型"NC"
NCK_typeNc, 912
数据类型“端子模块”
NCK_typeLoadingPoint, 1023
NCK_typeRelocate, 1040
NCK_typeToaUnit, 1022
NCK_typeToolholder, 1055
数据类型“通道”
NCK_typeChan, 673
数据类型“轴”
NCK_typeAxis, 440

添

添加指令, 52

通

通道
.abortProgLevel, 726
.ackFuncsExecution, 866
.activeValues[x], 765
.allAxesReferenced, 675
.allAxesStationary, 676
.asupActive, 676
.asupSilentActive, 677
.asupStopStateActive, 677
.blockSearchAckPlcActionDone, 726
.blockSearchActionActive, 678
.blockSearchActive, 679, 861
.blockSearchApproachActive, 680
.blockSearchLastActionActive, 680
.blockSearchSeruproActive, 681
.chan, 859
.chanPreactivated, 855
.chanViolated, 856
.circularJogTravers, 728
.clcActive, 682
.clcOvr, 729
.clcStopAtLowerLimitActive, 683
.clcStopAtUpperLimitActive, 684
.collisionStopActive, 685
.contManualTravers, 779, 808
.contManualTraversActive, 766, 796
.contourHandwheelActiveNumber, 687
.contourHandwheelDirInverted, 688
.contourHandwheelInvertDir, 730
.contourHandwheelSelection, 731
.contourHandwheelSim, 732
.deactivateClc, 733
.deactivateWearMonitoring, 830
.deactivateWorkpieceCounter, 831
.delayStroke, 869
.deleteDistanceToGo, 733
.deleteSubProgRepetition, 734
.disableFeed, 735, 780, 809
.disableNcStart, 736
.disableReadIn, 737
.disableReadInIgnored, 688
.disableSignals, 867
.disableTraversKey, 781, 810
.dryRunFeedrate, 737

.enableGoToStart, 738
.enableProtAreas, 739
.enableStroke, 870
.executionFromExternalActive, 690
.extendedAddress, 835, 838, 841, 844, 846, 848, 852
.externalProgLanguageModeActive, 690
.fastAckActive, 836, 839, 841, 844, 847, 849, 853
.g00Active, 766
.geoAxesTraversBcsWcsSelection, 740
.handwheelAssignedNumber, 767, 797
.handwheelDirInverted, 768, 798
.handwheelInvertDir, 781, 811
.handwheelOffsetDrf, 740
.handwheelOvrActive, 691
.handwheelSelection, 782, 810
.hasChanged, 836, 839, 842, 845, 847, 849, 853
.ignoreDisabledSate, 832
.inc1, 783, 812
.inc10, 784, 813
.inc100, 786, 815
.inc1000, 787, 816
.inc10000, 789, 818
.inc10000Active, 773, 802
.inc1000Active, 772, 801
.inc100Active, 771, 800
.inc10Active, 770, 799
.inc1Active, 769, 798
.incVar, 790, 819
.incVarActive, 774, 803
.interruptProcessingActive, 691
.jogRetractActive, 692
.jogRetractDataAvailable, 693
.lastReplacementToolReached, 825
.lastTNumber, 826
.lock, 865
.lockable, 864
.lockAllSynActs, 741
.m00Decoded, 834
.m00m01Active, 693
.m00m01AssociatedAuxFuncActive, 694
.m01, 742
.m01AssociatedAuxFunc, 742
.m02m30Active, 695
.manualStrokeInterface1, 870
.manualStrokeInterface2, 871
.nc, 860
.ncAlarmActive, 696
.ncAlarmWithProgStopActive, 697
.ncAlarmWithStopActive, 698
.ncPreactivated, 857
.ncStart, 743
.ncStartActive, 862
.ncStop, 744
.ncStopAxesAndSpindles, 745
.ncStopBlockEnd, 746
.ncViolated, 858
.newReplacementToolActive, 826
.newTNumber, 827
.noStroke, 872
.notDecoded, 837
.orientableToolholderActive, 699
.overStoreActive, 699
.partProgramEndActive, 862
.pathFeedrateLimit, 747
.pathFeedrateOvr, 748
.pathFeedrateOvrFactor, 749
.pathFixedFeedrateSelection, 751
.pathRapidTraversOvr, 752
.pathRapidTraversOvrFactor, 753
.powerOnActive, 863
.productionTargetReached, 699
.progStateAborted, 700
.progStateInterrupted, 701
.progStateRunning, 703
.progStateWaiting, 706
.progTest, 755
.progTestSelected, 707
.protAreasNotGuaranteed, 708
.ptpTravers, 757
.ptpTraversActive, 708
.rapidTraversOvr, 792, 821
.ready, 709
.reducedDynamicsActive, 710
.ref, 757
.refActive, 710
.reposDelayActive, 711
.reposModeActivationRequest, 758
.reposModeChangeAck, 712
.reposModeSelectedNumber, 714
.requested, 865
.reset, 761
.resetActive, 863
.retraceSupportActive, 715
.retraceSupportModeActive, 716
.retraceSupportStart, 762
.retraceSupportTraversBackwards, 763
.revolutionalFeedrateActive, 717
.set, 867
.signals, 873, 874
.signalsEnabled, 866
.singleBlock, 763
.singleBlockStopIgnored, 718

.skipBlockLevel0, 763
.skipBlockLevel1, 763
.skipBlockLevel2, 763
.skipBlockLevel3, 763
.skipBlockLevel4, 763
.skipBlockLevel5, 763
.skipBlockLevel6, 763
.skipBlockLevel7, 763
.skipBlockLevel8, 763
.skipBlockLevel9, 763
.stateActive, 719
.stateInterrupted, 720
.stateReset, 721
.stopConditionActive, 722
.strokeInitiationActive, 868
.strokeManualInitiationAck, 868
.suppressDisableNcStart, 764
.suppressStroke, 873
.toffonActive, 722
.toffonMotionActive, 723
.toolLifeMonitoring, 833
.toolLimitTNumber, 827
.toolLimitValueReached, 828
.toolMissing, 829
.toolPrewarnLimitReached, 829
.toolPrewarnLimitTNumber, 830
.transformationActive, 724
.transformationActiveNumber, 725
.traversKeyMinus, 793, 822
.traversKeyPlus, 794, 823
.traversMinusCommand, 775, 806
.traversMinusRequest, 776, 804
.traversPlusCommand, 777, 807
.traversPlusRequest, 778, 805
.value, 837, 840, 842, 845, 848, 850, 853
.valueTypeDint, 851
.xx, 704

运

运行方式组
., 885
.autoMode, 891
.autoModeActive, 875
.contTravers, 892
.contTraversActive, 875
.disableModeChangeover, 893
.inc1, 894
.inc10, 895
.inc100, 896
.inc1000, 898
.inc10000, 899
.inc10000Active, 882
.inc1000Active, 880
.inc100Active, 879
.inc10Active, 877
.inc1Active, 876
.incVar, 901
.incVarActive, 883
.jogMode, 902
.mdaMode, 903
.mdaModeActive, 886
.nckInternalJogActive, 886
.ready, 887
.refActive, 888
.refPoint, 904
.repos, 904
.reposActive, 888
.reset, 905
.resetAllChanDone, 889
.resetDone, 889
.singleBlockTypeA, 906
.singleBlockTypeB, 907
.stopAll, 908
.stopAxesOnly, 910
.teachIn, 911
.teachInActive, 890
运行方式组数据类型
NCK_typeModeGroup, 874

轴

轴
.AccelReached, 641
.AccelThresholdReached, 641
.acceptExternalWorkOffset, 491
.activateBrakeRampForSs2e, (安全)
.activateReversalPoint, 620
.active, 616
.actToolBasePos, 441
.alarmActive, 442
.assignedChanNumber, 568
.assignedToPlc, 569
.assignToChanSelection, 572
.assignToPlc, 573
.autoGearStageChange, 599
.autonomousMotionActive, 540
.brakeTestModeEnabled, (安全)
.chanAssignmentSelection, 668
.clampingInProgress, 492
.clampingToleranceExceeded, 443
.classCAlarmActive, 541
.constantCuttingSpeedActive, 577
.contTraversActive, 443, 493

.correctionActive, 642
.correctionDone, 642
.ctrlModeActive, 577
.currentCtrlActive, 541
.cwRotActive, 578
.deleteSValue, 600
.disable, 493
.disabled, 444, 617
.disableParameterSetSwitchover, 496
.disableSpeedCtrlIntegrator, 561
.disableSpeedMonitor, 600, 601
.disableTraversKeys, 497
.driveCtrlActive, 542
.driveDataSetNumber, 543
.driveDataSetSelection, 561
.driveTestTraversRequested, 544
.dynBacklashComp, 497
.dynBacklashCompActive, 445
.emergencyRetractionActive, 445
.enableBrakeTestMode, (安全)
.enableCtrl, 498
.enabled, 544
.enableDriveTestTravers, 563
.enableFollowOverlay, 658
.enableModuloLimit Switch, 500
.enableOvr, 501
.enablePulses, 563
.enc1Activation, 502
.enc1Active, 445
.enc1FreqLimitExceeded, 446
.enc1Restored, 447
.enc1Synchronized, 448
.enc2Activation, 504
.enc2Active, 449
.enc2FreqLimitExceeded, 450
.enc2Restored, 451
.enc2Synchronized, 451
.error, 617
.esrDcVoltageAlarmActive, 545
.esrGeneratorOperationActive, 545
.esrSpeedBelowMin, 545
.externalReversal, 621
.externalReversalActive, 618
.feedrateOvrFactor, 507
.feedStopSpindleStop, 510
.fixedFeedrateSelection, 511
.fixedStopAckReached, 513
.fixedStopEnable, 514
.fixedStopForceLimited, 452
.fixedStopReached, 453
.fixedStopRequest, 454
.fixedStopSensor, 515
.followingSpindleActive, 643
.followUpMode, 516
.followUpModeActive, 454
.gantryActive, 644
.gantryCutOffLimitExceeded, 644
.gantryDisableSync, 659
.gantryLeadingActive, 645
.gantryStartSyncRun, 659
.gantrySyncOk, 645
.gantrySyncRunReady, 645
.gantryWarningLimitExceeded, 646
.gearChangeoverDone, 601
.gearChangeoverRequested, 579
.gearStageSelection, 602
.handwheelAssignedNumber, 456
.handwheelDirInverted, 456
.handwheelInvertDir, 518
.handwheelOvrActive, 457
.handwheelSelection, 519
.heatSinkTemperatureWarning, 546
.holdingBrakeReleased, 547
.hwLimitSwitchMinus, 520
.hwLimitSwitchPlus, 520
.inc1, 521
.inc10, 522
.inc100, 523
.inc1000, 523
.inc10000, 523
.inc10000Active, 463
.inc1000Active, 461
.inc100Active, 460
.inc10Active, 459
.inc1Active, 458
.incVar, 523
.incVarActive, 464
.indexAxisInPos, 465
.infoCtrlChanActive, (安全)
.inPos, 580
.invertM3M4, 603
.isPathAxis, 466
.isPosAxis, 467
.isStationary, 468
.jogFixedPointReachedNumber, 468
.jogFixedPointSelectedNumber, 469
.jogFixedPointSelection, 523
.jogPosReached, 470
.jogToPos, 524
.jogToPosActive, 471
.leadingSpindleActive, 646
.limitSpeed, 525
.lubricationPulseActive, 472
.maxAccelReached, 647

.maxSpeedReached, 647
.mcsCollisionProtection, 660
.mcsCouplingActive, 648
.mcsDecouple, 660
.mcsFollowingAxisModeActive, 648
.mcsMirroringActive, 648
.mcsOffsetChanged, 649
.measurementActive, 472
.mFuncNumber, 581
.moduloLimitSwitchActive, 473
.motionActive, 618
.motionOverlaid, 649
.motorChangeoverDone, 564
.motorDataSetNumber, 547
.motorDataSetSelection, 565
.motorTemperatureWarning, 548
.msActivate, 660
.msActive, 650
.msCompCtrlActive, 651
.msSpeedDifferenceCoarseReached, 651
.msSpeedDifferenceFineReached, 652
.msTorqueComp, 661
.neutralAxis, 570
.numberOfDriveDataSets, 549
.numberOfDriveDataSetsValid, 550
.oscillEnable, 604
.oscillModeActive, 582
.oscillRequest, 605
.oscillSetDirCcw, 606
.oscillSetDirCw, 607
.ovrFactor, 608
.plcCtrlActive, 665
.plcCtrlPermanent, 665
.posAxisFeedrateValue, 473
.posCoarseReached, 474
.posCtrlActive, 475
.posCtrlParameterSetNumber, 551
.posCtrlParameterSetSelection, 526
.posFineReached, 476
.posModeActive, 582
.possible, 571
.processActive, 653
.progTestEnableCtrl, 527
.progTestRequest, 527
.progTestSuppression, 528
.pulsesEnabled, 551
.rapidStop, 566
.rapidStopActive, 552
.rapidTraversOvr, 529
.ready, 477, 553
.refPointApproachCam, 530
.refPointSelection, 530
.releaseHoldingBrake, 567
.reposDelay, 531
.reposDelayAck, 478
.reposOffsetActive, 480
.reposOffsetValid, 481
.reposPtp, 533
.requestCtrlByPlc, 605
.requestDriveCtrl, 568
.requested, 571
.requestedGearStageNumber, 583
.requestPlcCtrl, 575, 669
.requireSwitchOver, 574
.resetDone, 666
.resetMovement, 533
.resume, 670
.resyncEnc1, 611
.resyncEnc1ForPos, 612
.resyncEnc2, 612
.resyncEnc2ForPos, 613
.revolutionalFeedrateActive, 483
.rigidTappingActive, 584
.rotAxisInPos, 484
.selectReversalPoint, 621
.setpointLimitSelection, (安全)
.sFuncValue, 585
.Signal, 976
.smi24.sensorS1Present, 592
.smi24.sensorS1StateNumber, 593
.smi24.sensorS1Value, 594
.smi24.sensorS4Actuated, 595
.smi24.sensorS4Present, 596
.smi24.sensorS5Actuated, 596
.smi24.sensorS5Present, 597
.smi24.sensorS6Present, 598
.smi24.sensorsPresent, 598
.smi24.speedLimited, 599
.sparkingOutActive, 619
.speedAtSetpoint, 554
.speedBelowMin, 554
.speedBelowThreshold, 555
.speedCtrlActive, 484
.speedCtrlIntegratorDisabled, 556
.speedLimitationExceeded, 586
.speedSetpointAboveMax, 587
.speedSetpointBelowMin, 589
.speedSetpointReached, 556, 590
.SpeedThresholdReached, 653
.spindleModeActive, 485
.startCcw, 614
.startCw, 614
.startPosOperation, 615
.stop, 615, 671

.stopAtNextReversalPoint, 622
.stopped, 667
.swCams, 535
.swCamsActive, 486
.swLimitSwitch2Minus, 536
.swLimitSwitch2Plus, 536
.syncCoarse2Reached, 654
.syncCoarseReached, 655
.syncCorrection, 662
.syncFine2Reached, 656
.syncFineReached, 656
.syncModeActive, 591
.syncResetCorrection, 663
.syncSpindleCtrlMode, 657
.syncSpindleResync, 663
.syncSpindleSyncDisable, 664
.toolWithDynLimitationActive, 592
.torqueBelowThreshold, 557
.traversKeyMinus, 537
.traversKeyPlus, 538
.traversMinusCommandActive, 486
.traversMinusRequestActive, 487
.traversPlusCommandActive, 489
.traversPlusRequestActive, 490
.useFeedrateOvr, 616
.varSignalingFunc1Active, 558
.varSignalingFunc2Active, 559
.varSignalingFunc3Active, 560
.velocityReduced, 491
acceptanceTestModeSelected, 624
acceptanceTestSlpSelected, 625
activeBrake, 625
brakeSelection, 637
brakeTest, 626
brakeTestActive, 626
brakeTestCompleted, 626
brakeTestResult, 627
closeBrake, 638
currentLoadSign, 627
esrRetractRequested, 628
externalBrakeClosed, 638
externalBrakeRequested, 628
safetyFaultActive, 628
safetyMessageActive, 629
sdiNegativeSelected, 629
sdiPositiveSelected, 630
selectBrakeTest, 638
selectDirectionOfRotation, 639
selectTestSequence, 639
setpointInput, 630
slaActive, 630
slpRangeSelected, 631
slpSelectedAndUserAgreement, 631
slsActive, 632
slsBit0Active, 632
slsBit1Active, 632
slsSelected, 633
sosActive, 634
sosSelected, 634
ss1Active, 634
ss2Active, 635
ss2eActive, 635
ss2esrActive, 636
startBrakeTest, 640
stoActive, 636
sw2b.testStopActive, 636
sw2b.testStopRequired, 637
testStop, 640