

Kollmorgen Automation Suite

KAS Motion Library Reference Guide - v5.00

Reference Guide



Document Edition: JJ, July 2026

Valid for KAS Software Revision: 5.00

Valid for Kollmorgen Essentials Drive Firmware Version: 02-16-00-003

Part Number: 959716



For safe and proper use, follow these instructions.
Keep for future use.

KOLLMORGEN

A REGAL REXNORD BRAND

Table of Contents

1	Fieldbus Library	16
1.1	Fieldbus Functions and Function Blocks	17
1.1.1	EtherCAT Library	17
1.1.2	EtherNet/IP (ODVA)	17
1.2	Fieldbus - EtherCAT	19
1.2.0.1	Why use ECATReadSdo and ECATWriteSdo FBs?	19
1.2.0.2	Why use the DriveParam FBs?	20
1.2.1	EtherCAT Function Blocks that Work with Drive Parameters	21
1.2.1.1	Execution Time	21
1.2.1.1.1	Reason	21
1.2.1.1.2	Result	21
1.2.1.1.3	Solution	21
1.2.2	EtherCAT Function Blocks that Work with SDOs	23
1.2.3	DriveParamRead	24
1.2.3.0.1	Usage	25
1.2.4	DriveParamStrRead	29
1.2.4.0.1	Usage	30
1.2.5	DriveParamWrite	33
1.2.6	ECATCommErrors	38
1.2.6.0.1	ECATCommErr_ref Structure	38
1.2.6.0.2	EtherCAT Port Numbers	39
1.2.7	ECATDeviceAction	42
1.2.7.0.1	Supported Actions	43
1.2.7.0.2	Usage	43
1.2.8	ECATDeviceStatus	47
1.2.8.0.1	State Defines	48
1.2.8.0.2	LinkStatus Defines	48
1.2.9	ECATDevReadParam	52
1.2.9.0.1	EtherCAT Device Parameters	55
1.2.10	ECATGetObjVal - Deprecated	58
1.2.11	ECATMasterStatus	59
1.2.11.0.1	State Defines	59
1.2.12	ECATReadData	61
1.2.13	ECATReadPDOData	63
1.2.13.0.1	Error Codes	64
1.2.14	ECATReadSdo	68
1.2.15	ECATReadSdoData	74
1.2.16	ECATWCStatus	80
1.2.17	ECATWritePDOData	82
1.2.17.0.1	Error Codes	83
1.2.18	ECATWriteData	87
1.2.19	ECATWriteSdo	89
1.2.20	ECATWriteSdoData	95
1.2.21	FSoEParamsInit	101
1.3	Fieldbus - EtherNet/IP (ODVA)	105
1.3.1	eipAdapter	106

1.3.1.0.1 Example	106
1.3.2 eipReadAttr	107
1.3.3 eipRestartCnx	110
1.3.4 eipWriteAttr	112
2 Kollmorgen UDFBs	115
2.1 UDFB Functions and Function Blocks	117
2.1.1 Common	117
2.1.2 Pipe Network	117
2.1.3 PLCopen	118
2.1.4 PLC	118
2.2 Create or Edit UDFBs	119
2.2.1 Create a UDFB Instance	119
2.2.2 Edit UDFBs	119
2.3 UDFB - Motion / Common	121
2.3.1 FB_AKDFltRpt	122
2.3.2 FB_AxisPlsPosModulo	125
2.3.2.0.1 Timing	125
2.3.2.0.2 Hysteresis	125
2.3.3 FB_AxisPlsPosNoModulo	128
2.3.3.0.1 Timing	128
2.3.3.0.2 Hysteresis	128
2.3.4 FB_Cylinder	131
2.3.4.0.1 Usage	131
2.3.5 FB_S700FltRpt	134
2.3.5.0.1 Usage	134
2.4 UDFB - Motion / Pipe Network	137
2.4.1 MLFB_DriveFault	138
2.4.1.0.1 Usage	138
2.4.1.0.2 Bitmask Causes	139
2.4.2 MLFB_ECATRstart	141
2.4.2.0.1 Examples	141
2.4.3 Fast Homing - UDFB - Motion/Pipe Network	143
2.4.3.1 MLFB_HomeFindHomeFastInput	144
2.4.3.1.1 Usage	144
2.4.3.2 MLFB_HomeFindHomeFastInputModulo	148
2.4.3.2.1.1 Usage	148
2.4.3.3 MLFB_HomeFindLimitFastInput	152
2.4.3.3.1.1 Usage	152
2.4.3.4 MLFB_HomeFindLimitFastInputModulo	156
2.4.3.4.1.1 Usage	156
2.4.4 Homing - UDFB - Motion/Pipe Network	160
2.4.4.1 MLFB_HomeFindHomeInput	161
2.4.4.2 MLFB_HomeFindHomeInputThenZeroAngle	164
2.4.4.3 MLFB_HomeFindLimitInput	167
2.4.4.4 MLFB_HomeFindLimitInputThenZeroAngle	170
2.4.4.5 MLFB_HomeFindZeroAngle	173
2.4.4.6 MLFB_HomeMoveUntilPosErrExceeded	176
2.4.4.7 MLFB_HomeMoveUntilPosErrExceededThenZeroAngle	179

2.4.4.8 MLFB_HomeUsingCurrentPosition	182
2.4.5 MLFB_Jog	184
2.4.6 PLS - UDFB - Motion/Pipe Network	186
2.4.6.1 MLFB_PlsPosFw	187
2.4.6.1.1.1 Timing	187
2.4.6.2 MLFB_PlsPosFwBw	189
2.4.6.2.1.1 Timing	189
2.4.6.2.2.2 Hysteresis	189
2.4.6.3 MLFB_PlsTimeFw	192
2.4.6.3.1.1 Timing	192
2.5 UDFB - Motion / PLCopen	194
2.5.1 Drive Fault Reporting - UDFB - Motion/PLCopen	195
2.5.1.1 MCFB_AKDFault	196
2.5.1.1.1.1 Usage	196
2.5.1.2 MCFB_AKDFaultLookup	198
2.5.1.2.1.1 Usage	198
2.5.1.3 MCFB_DriveFault	200
2.5.1.3.1.1 Usage	200
2.5.1.3.2.2 Bitmask Causes	201
2.5.2 MCFB_ECATRestart	203
2.5.3 MCFB_StepAbsSwitchFastInput	205
2.5.3.0.1 Usage	205
2.5.4 MCFB_StepLimitSwitchFastInput	210
2.5.4.0.1 Usage	210
2.5.5 Homing - UDFB - Motion/PLCopen	214
2.5.5.1 MCFB_StepAbsolutes	215
2.5.5.2 MCFB_StepAbsSwitch	217
2.5.5.2.1.1 Usage	217
2.5.5.3 MCFB_StepBlock	222
2.5.5.3.1.1 Usage	222
2.5.5.4 MCFB_StepLimitSwitch	226
2.5.5.4.1.1 Usage	226
2.5.5.5 MCFB_StepRefPulse	231
2.5.5.5.1.1 Usage	231
2.5.6 MCFB_Jog	236
2.5.7 MCFB_GearedWebTension	238
2.5.7.0.1 Usage	238
2.5.7.0.2 PID Function in KAS	239
2.5.7.0.3 Programming Tips	240
2.6 UDFB - PLC	245
2.6.1 FB_ElapseTime	246
2.6.2 FB_FirstOrderDigitalFilter	248
2.6.2.0.1 Filter Input Delay	248
2.6.2.0.2 Filter Delay Tn	249
2.6.2.0.3 Example: Filter Input Lag - Random Input	250
2.6.3 FB_PWDutyOutput	254
2.6.3.0.1 Function Block Calculations	254
2.6.4 FB_ScaleInput - Scaling Analog IO	257

2.6.5 FB_ScaleOutput - Scaling Analog IO	259
2.6.6 FB_TemperaturePID	261
2.6.6.0.1 Tuning Process	261
2.6.6.0.2 Tuning Procedure	261
2.6.6.0.3 Start PID Controller	262
2.6.7 PipeNetwork_FFLD - Special FFLD Function	265
2.6.8 ProfilesCode_FFLD - Special FFLD Function	267
3 Motion Library	269
3.1 Motion / Pipe Network	270
3.1.1 Example: Pipe Block Description	270
3.1.2 Adder	271
3.1.2.1 Rules	271
3.1.2.1.1 Adder - Rule 1	271
3.1.2.1.2 Adder - Rule 2	271
3.1.2.2 MLAddInit	272
3.1.2.2.1 FBD Language Example	272
3.1.2.2.2 FFLD Language Example	272
3.1.2.2.3 ST Language Example	273
3.1.2.3 MLAddReadOff1	274
3.1.2.3.1 ST Language Example	274
3.1.2.4 MLAddReadOff2	275
3.1.2.5 MLAddReadRatio1	277
3.1.2.6 MLAddReadRatio2	279
3.1.2.7 MLAddWriteInput	281
3.1.2.8 MLAddWriteOff1	283
3.1.2.9 MLAddWriteOff2	285
3.1.2.10 MLAddWriteRat1	287
3.1.2.11 MLAddWriteRat2	289
3.1.3 Block	291
3.1.3.1 MLBlkCreate	292
3.1.3.2 MLBlkIsReady	294
3.1.3.3 MLBlkReadModPos	296
3.1.3.4 MLBlkReadOutVal	298
3.1.3.5 MLBlkWriteModPos	299
3.1.4 Pipe	301
3.1.4.1 MLPipeAct	302
3.1.4.2 MLPipeAddBlock	304
3.1.4.3 MLPipeCreate	306
3.1.4.4 MLPipeDeact	308
3.1.5 Axis	310
3.1.5.1 Associated Data on Positions	311
3.1.5.2 MLAxisAbs	313
3.1.5.2.1.1 Position with Modulo On	313
3.1.5.3 Forcing the Direction of Rotation	313
3.1.5.3.1.1 End Position is Less Than Start Position	313
3.1.5.3.2.2 End Position is Greater Than Start Position	314
3.1.5.3.3.3 Travel Speed Update with MLAxisAbs	314
3.1.5.4 MLAxisAdd	317

3.1.5.4.1.1 DeltaPosition Axis	317
3.1.5.5 MLAxisAddress	319
3.1.5.6 MLAxisAddTq	321
3.1.5.7 MLAxisCfgFastIn	323
3.1.5.7.1.1 Fast Homing - Inputs	323
3.1.5.7.2 Outputs	323
3.1.5.8 MLAxisCmdPos	325
3.1.5.9 MLAxisDriveNumber	327
3.1.5.10 MLAxisFBackPos	328
3.1.5.11 MLAxisGenEN	330
3.1.5.12 MLAxisGenIsEN	331
3.1.5.13 MLAxisGenIsRdy	332
3.1.5.14 MLAxisGenPos	334
3.1.5.15 MLAxisGenReadAcc	336
3.1.5.16 MLAxisGenReadDec	337
3.1.5.17 MLAxisGenReadSpd	338
3.1.5.18 MLAxisGenWriteAcc	339
3.1.5.19 MLAxisGenWriteDec	341
3.1.5.20 MLAxisGenWriteSpd	343
3.1.5.21 MLAxisInit	345
3.1.5.22 MLAxisIsCnctd	349
3.1.5.23 MLAxisIsTriggered	350
3.1.5.24 MLAxisMoveVel	352
3.1.5.25 MLAxisPipePos	354
3.1.5.26 MLAxisPower	356
3.1.5.27 MLAxisPowerDOff - Deprecated	358
3.1.5.28 MLAxisRatedTq	359
3.1.5.29 MLAxisReadActPos	361
3.1.5.30 MLAxisReadBoolParam	363
3.1.5.31 MLAxisReadFBUnit	366
3.1.5.32 MLAxisReadFEUU	367
3.1.5.33 MLAxisReadGenStatus	368
3.1.5.33.1.1 Internal Generator Status	368
3.1.5.34 MLAxisReadModPos	370
3.1.5.35 MLAxisReadParam	371
3.1.5.36 MLAxisReadTq	374
3.1.5.37 MLAxisReadUUnits	375
3.1.5.38 MLAxisReadVel	376
3.1.5.39 MLAxisReAlignRdy	378
3.1.5.40 MLAxisReAlign	379
3.1.5.41 MLAxisRel	381
3.1.5.41.1.1 DeltaPosition Axis	381
3.1.5.42 MLAxisResetErrors	383
3.1.5.43 MLAxisRstFastIn	384
3.1.5.44 MLAxisStatus	386
3.1.5.44.1.1 Bit Descriptions	386
3.1.5.45 MLAxisStop	389
3.1.5.46 MLAxisTimeStamp	392

3.1.5.47	MLAxisWriteBoolParam	394
3.1.5.48	MLAxisWriteModPos	396
3.1.5.49	MLAxisWriteParam	397
3.1.5.50	MLAxisWritePipPos	399
3.1.5.51	MLAxisWritePos	401
3.1.5.52	MLAxisWriteUUnits	404
3.1.5.53	MLPNAxisCreate	405
3.1.5.54	Axis Function Examples	407
3.1.5.54.1	MLAxisMoveVel(Speed)	407
3.1.5.54.2	MLAxisMoveVel(0.0)	407
3.1.6	CAM Profile	408
3.1.6.1	Shape Specification	408
3.1.6.1.1	Cam's INPUT-OUTPUT Transfer Function	408
3.1.6.2	MLCamInit	411
3.1.6.3	MLCamSwitch	413
3.1.6.4	MLPrfReadIOffset	415
3.1.6.5	MLPrfReadIScale	417
3.1.6.6	MLPrfReadOOffset	419
3.1.6.7	MLPrfReadOScale	421
3.1.6.8	MLPrfWriteIOffset	423
3.1.6.9	MLPrfWriteIScale	425
3.1.6.10	MLPrfWriteOOffset	427
3.1.6.11	MLPrfWriteOScale	429
3.1.7	Comparator	431
3.1.7.1	Examples: Comparator Functions	431
3.1.7.2	MLCompCheck	433
3.1.7.3	MLCompInit	435
3.1.7.4	MLCompReadRef	438
3.1.7.5	MLCompReset	439
3.1.7.6	MLCompWriteRef	440
3.1.8	Convertor	442
3.1.8.1	MLCNVConECAT	443
3.1.8.2	MLCNVConnect	445
3.1.8.3	MLCNVConnectEx	447
3.1.8.3.1.1	Precedence Rules	447
3.1.8.4	MLCNVDisconnect	450
3.1.8.5	MLCNVInit	452
3.1.9	Delay	454
3.1.9.1	MLDelayInit	455
3.1.10	Derivator	456
3.1.10.1	MLDerInit	457
3.1.10.2	MLDerReadInModPos	460
3.1.10.3	MLDerWriteInModPos	462
3.1.11	Gear	464
3.1.11.1	Examples: Gear Functions	464
3.1.11.2	MLGearInit	467
3.1.11.2.1.1	Modulo Value	469
3.1.11.3	Avoid Modulo Calculation Problems	469

3.1.11.4	MLGearReadOffset	471
3.1.11.5	MLGearReadOffSlp	472
3.1.11.6	MLGearReadRatio	473
3.1.11.7	MLGearReadRatSlp	474
3.1.11.8	MLGearWriteOff	475
3.1.11.9	MLGearWriteOSlp	477
3.1.11.10	MLGearWriteRatio	479
3.1.11.11	MLGearWriteRatSlp	481
3.1.11.11.1	RatioSlope Offset	481
3.1.11.12	With Offset	481
3.1.11.13	Without Offset	482
3.1.12	Integrator	483
3.1.12.1	MLIntWriteOutVal	484
3.1.12.2	MLIntInit	486
3.1.13	Master	488
3.1.13.1	Examples: Master Function	488
3.1.13.2	MLMstAbs	490
3.1.13.2.1.1	Position with Modulo On	490
3.1.13.3	Forcing the Direction of Rotation	491
3.1.13.3.1.1	End Position is Less Than Start Position	491
3.1.13.3.2.2	End Position is Greater Than Start Position	491
3.1.13.3.3.3	Travel Speed Update with MLAxisAbs	492
3.1.13.4	MLMstAdd	495
3.1.13.5	MLMstForcePos	497
3.1.13.6	MLMstInit	499
3.1.13.7	MLMstReadAccel	502
3.1.13.8	MLMstReadDecel	504
3.1.13.9	MLMstReadInitPos	505
3.1.13.10	MLMstReadSpeed	506
3.1.13.11	MLMstRel	508
3.1.13.12	MLMstRun	510
3.1.13.13	MLMstStatus	512
3.1.13.13.1.1	Status Definitions	512
3.1.13.14	MLMstWriteAccel	514
3.1.13.15	MLMstWriteDecel	516
3.1.13.16	MLMstWriteInitPos	518
3.1.13.17	MLMstWriteSpeed	519
3.1.14	Phaser	521
3.1.14.1	Examples: Phaser Function	521
3.1.14.2	MLPhaInit	523
3.1.14.3	MLPhaReadActPhase	525
3.1.14.4	MLPhaReadPhase	527
3.1.14.5	MLPhaReadSlope	528
3.1.14.6	MLPhaWritePhase	529
3.1.14.7	MLPhaWriteSlope	531
3.1.15	PMP	533
3.1.15.1	MLPmpAbs	534
3.1.15.2	MLPmpForcePos	536

3.1.15.3	MLPmpInit	538
3.1.15.4	MLPmpReadAccel	541
3.1.15.5	MLPmpReadFstSpd	543
3.1.15.6	MLPmpReadInitPos	545
3.1.15.7	MLPmpReadJerk	546
3.1.15.8	MLPmpReadLstSpd	547
3.1.15.9	MLPmpRel	549
3.1.15.10	MLPmpRun	551
3.1.15.11	MLPmpStatus	553
3.1.15.12	MLPmpWriteAccel	555
3.1.15.13	MLPmpWriteFstSpd	557
3.1.15.14	MLPmpWriteJerk	559
3.1.15.15	MLPmpWriteLstSpd	561
3.1.16	Sampler	563
3.1.16.1	MLSmpConECAT	564
3.1.16.2	MLSmpConnect	566
3.1.16.3	MLSmpConPLCAxis	568
3.1.16.4	MLSmpConPNAxis	570
3.1.16.5	MLSmplnit	572
3.1.16.5.1.1	Modulo Affects on the Signal Scaling	574
3.1.17	Synchronizer	577
3.1.17.1	Example: Synchronizer Functions	577
3.1.17.2	MLSynclnit	579
3.1.17.3	MLSyncReadDeltaS	581
3.1.17.4	MLSyncStart	583
3.1.17.5	MLSyncStop	585
3.1.17.6	MLSyncWriteDeltaS	587
3.1.18	Trigger	589
3.1.18.1	Example: Trigger Functions	589
3.1.18.2	MLTrigClearFlag	591
3.1.18.2.1.1	Fast Homing - Inputs	591
3.1.18.3	MLTrigInit	593
3.1.18.3.1.1	Fast Homing - Inputs	593
3.1.18.4	MLTrigIsTriggered	595
3.1.18.4.1.1	Fast Homing - Inputs	595
3.1.18.5	MLTrigReadDelay	597
3.1.18.5.1.1	Fast Homing - Inputs	597
3.1.18.6	MLTrigReadPos	599
3.1.18.6.1.1	Fast Homing - Inputs	599
3.1.18.7	MLTrigReadTime	601
3.1.18.7.1.1	Fast Homing - Inputs	601
3.1.18.8	MLTrigSetEdge	603
3.1.18.8.1.1	Fast Homing - Inputs	603
3.1.18.9	MLTrigWriteDelay	605
3.1.18.9.1.1	Fast Homing - Inputs	605
3.2	Motion / PLCopen Library	607
3.2.1	Control	607
3.2.2	Information	607

3.2.3 I/O	607
3.2.4 PLCOpenMotion	608
3.2.5 Profile	608
3.2.6 Reference	609
3.2.7 Registration	609
3.2.8 Control	610
3.2.8.1 MC_ClearFaults	611
3.2.8.1.1 Usage	611
3.2.8.2 MC_CreatePLCAxis	613
3.2.8.3 MC_EStop	617
3.2.8.4 MC_InitAxis	619
3.2.8.5 MC_InitAxisFeedback	621
3.2.8.6 MC_Power	623
3.2.8.7 MC_ResetError	625
3.2.8.8 MC_Stop	627
3.2.9 I/O	630
3.2.9.1 MC_AbortTrigger	631
3.2.9.2 MC_TouchProbe	633
3.2.9.2.1 Elements	635
3.2.9.2.2 Usage	636
3.2.9.2.3 Limitations	636
3.2.9.2.4 Fast Homing - Inputs	636
3.2.9.2.5 AKD Support with MC_TouchProbe	638
3.2.9.2.6 AKD2G Support With MC_TouchProbe	639
3.2.10 Information	641
3.2.10.1 MC_ReadActPos	642
3.2.10.2 MC_ReadActVel	644
3.2.10.3 MC_ReadAxisErr	646
3.2.10.3.1 Output Bits	646
3.2.10.4 MC_ReadBoolPar	648
3.2.10.5 MC_ReadParam	650
3.2.10.6 MC_ReadStatus	652
3.2.10.7 MC_WriteBoolPar	654
3.2.10.8 MC_WriteParam	656
3.2.11 PLCOpen Motion	658
3.2.11.1 MC_AddSuperAxis	659
3.2.11.2 MC_Halt	661
3.2.11.3 MC_MoveAbsolute	664
3.2.11.4 MC_MoveAdditive	669
3.2.11.5 MC_MoveContVel	673
3.2.11.6 MC_MoveRelative	677
3.2.11.7 MC_MoveSuperimp	681
3.2.11.7.1 Usage	681
3.2.11.8 MC_MoveVelocity	685
3.2.11.9 MC_RemSuperAxis	688
3.2.11.10 MC_SetOverride	690
3.2.12 Profile	692
3.2.12.1 MC_CamIn	693

3.2.12.1.1.1 Abort Camming	695
3.2.12.1.2.2 Usage	695
3.2.12.1.3.3 Periodic or Not Periodic	695
3.2.12.1.4.4 Adjustments Computation is Done	695
3.2.12.1.5.5 Dynamically Change a Cam Profile	695
3.2.12.1.6 Examples	696
3.2.12.1.7.1 Example 1	696
3.2.12.1.8.2 Example 2	697
3.2.12.1.9.3 Example 3	697
3.2.12.2 MC_CamOut	700
3.2.12.2.1.1 Cancel the Cam Motion	701
3.2.12.3 MC_CamResumePos	703
3.2.12.4 MC_CamStartPos	705
3.2.12.5 MC_CamTblSelect	707
3.2.12.5.1.1 Usage	707
3.2.12.6 MC_GearIn	710
3.2.12.6.1.1 Aborting Gearing	711
3.2.12.6.2.2 Time to Reach the Target Velocity	711
3.2.12.7 MC_GearInPos	714
3.2.12.7.1.1 Aborting Gearing	719
3.2.12.7.2.2 Time to Reach the Target Velocity	719
3.2.12.7.3.3 Example	720
3.2.12.8 MC_GearOut	722
3.2.12.9 MC_Phasing	725
3.2.12.10 MC_SyncSlaves	728
3.2.13 Reference	730
3.2.13.1 MC_Reference	731
3.2.13.1.1.1 Usage	733
3.2.13.1.2.2 ST Language Example	734
3.2.13.2 MC_SetPos	736
3.2.13.3 MC_SetPosition - Deprecated	739
3.2.14 Registration	740
3.2.14.1 MC_MachRegist	741
3.2.14.1.1.1 Transition and Registration	745
3.2.14.1.2.2 Distance, Tolerance, Ignore, Target Inputs	745
3.2.14.1.3.3 Position and Option Inputs	745
3.2.14.1.4.4 MC_MachRegist Options Table	746
3.2.14.2 MC_MarkRegist	749
3.2.14.2.1.1 Transition and Registration	751
3.2.14.2.2.2 Distance, Tolerance, Ignore, Target Inputs	752
3.2.14.2.3.3 Position and Option Inputs	752
3.2.14.2.4 MC_MachRegist Options Table	753
3.2.14.3 MC_StopRegist	755
3.3 Motion / Common	757
3.3.1 Coordinated Motion	757
3.3.2 Info	757
3.3.3 Motion	757
3.3.4 Profiles	757

3.3.5 Motion / Common - Coordinated Motion	758
3.3.5.1 Group Control	758
3.3.5.2 Info	758
3.3.5.3 Motion	759
3.3.5.4 Reference	759
3.3.5.5 Coordinated Motion - Group Control	760
3.3.5.5.1 MC_AddAxisToGrp	761
3.3.5.5.2 MC_CreateAxesGrp	764
3.3.5.5.3 MC_GrpDisable	767
3.3.5.5.4 MC_GrpEnable	769
3.3.5.5.5 MC_GrpReadBoolPar	771
3.3.5.5.6 MC_GrpReadParam	773
3.3.5.5.7 MC_GrpReset	775
3.3.5.5.8 MC_GrpStop	777
3.3.6 Related Functions	779
3.3.6.0.1 MC_GrpWriteBoolPar	780
3.3.6.0.2 MC_GrpWriteParam	782
3.3.6.0.3 MC_InitAxesGrp	784
3.3.6.0.4 MC_RemAxisFromGrp	786
3.3.6.0.5 MC_SetKinTra	788
3.3.6.0.6 MC_UngroupAllAxes	791
3.3.6.0.7 MC_KIN_REF Structure	793
3.3.6.0.8.1 General MC_KIN_REF Structure	793
3.3.6.0.9.2 Robotic Transform Parameters	793
3.3.7 MC_TRANSFORM_TYPE_GANTRY	793
3.3.8 MC_TRANSFORM_TYPE_GANTRY_WITH_SKEW	793
3.3.9 MC_TRANSFORM_TYPE_HBOT	794
3.3.10 MC_TRANSFORM_TYPE_TBOT	794
3.3.11 MC_TRANSFORM_TYPE_DELTA (*Experimental*)	795
3.3.12 MC_TRANSFORM_TYPE_SCARA_ELBOW_POS (*Experimental*)	797
3.3.13 MC_TRANSFORM_TYPE_SCARA_ELBOW_NEG (*Experimental*)	797
3.3.13.1 Coordinated Motion - Info	799
3.3.13.1.1 MC_GrpReadActAcc	800
3.3.13.1.2 MC_GrpReadActPos	803
3.3.13.1.3 MC_GrpReadActVel	806
3.3.13.1.4 MC_GrpReadCmdPos	809
3.3.13.1.5 MC_GrpReadCmdVel	812
3.3.13.1.6 MC_GrpReadError	815
3.3.13.1.7 MC_GrpReadStatus	817
3.3.13.2 Coordinated Motion - Motion	821
3.3.13.2.1 MC_AxisSetDefaults	822
3.3.13.2.2 MC_GrpHalt	825
3.3.13.2.3 MC_GrpSetOverride	828
3.3.13.2.4 MC_MoveCircAbs	830
3.3.13.2.5 MC_MoveCircRel	836
3.3.13.2.6 MC_MoveDirAbs	842
3.3.13.2.7 MC_MoveDirRel	845
3.3.13.2.8 MC_MoveLinAbs	848

3.3.13.2.9 MC_MoveLinRel	853
3.3.13.3 Coordinated Motion - Reference	858
3.3.13.3.1 MC_GrpSetPos	859
3.3.14 Motion / Common - Info	862
3.3.14.1 MC_ErrorDescription	863
3.3.15 Motion / Common - Motion	864
3.3.15.1 State Machine	865
3.3.15.2 MLMotionCycleTime	866
3.3.15.3 MLMotionInit	867
3.3.15.4 MLMotionRstErr	870
3.3.15.5 MLMotionStart	872
3.3.15.6 MLMotionStatus	875
3.3.15.7 MLMotionStop	878
3.3.16 Motion / Common - Profiles	880
3.3.16.1 MLPProfileBuild	881
3.3.16.1.1	882
3.3.16.1.2.1 CamProps_Ref Structure	883
3.3.16.1.3.2 CamData_Ref Structure	883
3.3.16.1.4.3 Example: Use MLPProfileBuild	885
3.3.16.1.5.4 1. Create a Profile	886
3.3.16.1.6.5 2. Define the Profile Data	886
3.3.16.1.7.6 3. Define the Profile Properties	886
3.3.16.1.8.7 4. Call the MLPProfileBuild Function Block	886
3.3.16.1.9.8 5. Use the New Profile	887
3.3.16.1.10.9 Example: Building a Parabolic Cam Profile	887
3.3.16.2 MLPProfileCreate	890
3.3.16.3 MLPProfileInit	892
3.3.16.4 MLPProfileRelease	894
3.3.16.4.1	894
3.3.16.4.2.1 In-use Definitions	894
4 System Library	897
4.1 Controller	897
4.2 File	897
4.3 Profiling	898
4.4 Recipe Function Blocks	898
4.5 TCP/IP Sockets	898
4.6 UDP	898
4.7 Controller Functions	900
4.7.1 ClearCtrlErrors	901
4.7.2 CtrlReadBoolPar	902
4.7.3 GetCtrlErrors	904
4.7.3.0.1 Status Meaning	904
4.7.4 GetCtrlInfo	906
4.7.5 GetCtrlPerf	909
4.8 File Function Blocks	912
4.8.1 FileClose	913
4.8.2 FileCopy	915
4.8.3 FileDelete	917

4.8.4 FileEOF	919
4.8.5 FileExists	921
4.8.6 FileOpenA	923
4.8.7 FileOpenR	925
4.8.8 FileOpenW	927
4.8.9 FileReadBinData	929
4.8.10 FileReadLine	931
4.8.11 FileRename	933
4.8.12 FileSeek	935
4.8.13 FileSize	937
4.8.14 FileWriteBinData	939
4.8.15 FileWriteLine	941
4.8.15.0.1 String Escape Sequences	941
4.9 Profiling Function Blocks	944
4.9.1 Stopwatch	945
4.10 Recipe Function Blocks	949
4.10.1 RecipeApplyInstance	950
4.10.2 RecipeClose	952
4.10.3 RecipeOpen	954
4.10.4 RecipeQuickLoad	956
4.10.5 RecipeSave	958
4.10.6 RecipeSaveRetainVars	960
4.10.7 RecipeStoreInstance	962
4.11 TCP/IP Function Blocks	964
4.11.1 TcpAccept	965
4.11.2 TcpBinReceive	967
4.11.3 TcpBinSend	969
4.11.4 TcpClose	971
4.11.5 TcpConnect	973
4.11.6 TcpIsConnected	976
4.11.7 TcpIsValid	978
4.11.8 TcpListen	980
4.11.9 TcpSend	982
4.12 UDP Functions / Function Blocks	984
4.12.1 UDP Functions	984
4.12.2 udpAddrMake	985
4.12.3 udpClose	987
4.12.4 udpCreate	988
4.12.5 udplsValid	989
4.12.6 udpRcvFrom	990
4.12.7 udpRcvFromArray	992
4.12.8 udpRcvFromVar	994
4.12.9 udpSendTo	996
4.12.10 udpSendToArray	998
4.12.11 udpSendToVar	1000
4.13 PrintMessage	1002
4.13.0.1 Source	1002
4.13.0.2 Level	1002

4.13.0.3 Usage	1002
4.14 File, Recipe, and TCP/IP Function Block ErrorIDs	1005
4.14.1 File ErrorIDs	1005
4.14.2 Recipe ErrorIDs	1006
4.14.3 TCP/IP ErrorIDs	1006
5 Data Types	1007
5.1 Different Data Types	1007
5.2 List of Data Types	1007
5.3 Arrays	1008
5.4 Bitfields	1008
5.5 Enumerations	1009
5.6 Structure	1009
5.6.1 Limitation	1009
6 Kollmorgen Documentation Legal Information	1010
6.1 KAS - Copyrights, Trademarks, Patents, Licenses, and Disclaimers	1011
6.2 Kollmorgen Trademarks	1011
6.3 Kollmorgen Patents	1011
6.4 PCMM2G	1012
6.5 Other Trademarks	1013
6.6 3rd-party Licenses	1014
6.7 Disclaimers	1016
Support and Services	1018

1 Fieldbus Library

1.1 Fieldbus Functions and Function Blocks	17
1.1.1 EtherCAT Library	17
1.1.2 EtherNet/IP (ODVA)	17
1.2 Fieldbus - EtherCAT	19
1.2.1 EtherCAT Function Blocks that Work with Drive Parameters	21
1.2.2 EtherCAT Function Blocks that Work with SDOs	23
1.2.3 DriveParamRead	24
1.2.4 DriveParamStrRead	29
1.2.5 DriveParamWrite	33
1.2.6 ECATCommErrors	38
1.2.7 ECATDeviceAction	42
1.2.8 ECATDeviceStatus	47
1.2.9 ECATDevReadParam	52
1.2.10 ECATGetObjVal - Deprecated	58
1.2.11 ECATMasterStatus	59
1.2.12 ECATReadData	61
1.2.13 ECATReadPDOData	63
1.2.14 ECATReadSdo	68
1.2.15 ECATReadSdoData	74
1.2.16 ECATWCStatus	80
1.2.17 ECATWritePDOData	82
1.2.18 ECATWriteData	87
1.2.19 ECATWriteSdo	89
1.2.20 ECATWriteSdoData	95
1.2.21 FSoEParamsInit	101
1.3 Fieldbus - EtherNet/IP (ODVA)	105
1.3.1 eipAdapter	106
1.3.2 eipReadAttr	107
1.3.3 eipRestartCnx	110
1.3.4 eipWriteAttr	112

1.1 Fieldbus Functions and Function Blocks

These tables list the Fieldbus functions and function blocks:

- "EtherCAT Library" (→ p. 17)
- "EtherNet/IP (ODVA)" (→ p. 17)

1.1.1 EtherCAT Library

Name	Description
DriveParamRead	Reads a drive parameter by sending an ASCII command to a drive.
DriveParamStrRead	Reads a single drive parameter by sending an ASCII command to a drive.
DriveParamWrite	Writes a drive parameter by sending an ASCII command to a drive.
ECATCommErrors	Returns a list of bad EtherCAT connections.
ECATDeviceAction	Performs an action on an EtherCAT device.
ECATDeviceStatus	Provides the EtherCAT state and the port link status information for the EtherCAT device.
ECATDevReadParam	Returns the EtherCAT device-specific information.
ECATGetObjVal	Deprecated.
ECATMasterStatus	Reads the EtherCAT master state and the lost frame counter to determine if EtherCAT is running normally.
ECATReadData	Reads cyclic parameter (byte offset format).
ECATReadPDOData	Copies the Input (Tx) PDO object data from an EtherCAT cyclic frame to a byte array.
ECATReadSdo	Reads a 32-bit word from I/O nodes using a CANopen SDO read command.
ECATReadSdoData	Reads data from an EtherCAT device using CANopen SDO read command.
ECATWCStatus	Returns the current number of working counter errors for the Sync unit.
ECATWriteData	Writes cyclic parameter (byte offset format).
ECATWritePDOData	Writes data to an Output Rx PDO object of the EtherCAT device in the cyclic EtherCAT frame.
ECATWriteSdo	Writes a 32-bit word to I/O nodes using a CANopen SDO write command.
ECATWriteSdoData	Writes data to an EtherCAT device using a CANopen SDO write command..
FSOEParamsInit	Transfers safety parameters from the safety master to safety slave devices to initialize the safety network.

1.1.2 EtherNet/IP (ODVA)

Name	Description
eipAdapter	Provides information about the current state of the Scanner connection.
eipReadAttr	Sends an explicit message (UCMM) to an EtherNet/IP adapter to read a single CIP attribute.
eipRestartCnx	The EIP I/O connection is restarted when this function is called.

Name	Description
eipWriteAttr	Sends an explicit message (UCMM) to an EtherNet/IP adapter to write a single CIP attribute.

1.2 Fieldbus - EtherCAT

Name	Description
DriveParamRead	Reads a drive parameter by sending an ASCII command to a drive.
DriveParamStrRead	Reads a single drive parameter by sending an ASCII command to a drive.
DriveParamWrite	Writes a drive parameter by sending an ASCII command to a drive.
ECATCommErrors	Returns a list of bad EtherCAT connections.
ECATDeviceAction	Performs an action on an EtherCAT device.
ECATDeviceStatus	Provides the EtherCAT state and the port link status information for the EtherCAT device.
ECATDevReadParam	Returns the EtherCAT device-specific information.
ECATGetObjVal	Deprecated.
ECATMasterStatus	Reads the EtherCAT master state and the lost frame counter to determine if EtherCAT is running normally.
ECATReadData	Reads cyclic parameter (byte offset format).
ECATReadPDOData	Copies the Input (Tx) PDO object data from an EtherCAT cyclic frame to a byte array.
ECATReadSdo	Reads a 32-bit word from I/O nodes using a CANopen SDO read command.
ECATReadSdoData	Reads data from an EtherCAT device using CANopen SDO read command.
ECATWCStatus	Returns the current number of working counter errors for the Sync unit.
ECATWriteData	Writes cyclic parameter (byte offset format).
ECATWritePDOData	Writes data to an Output Rx PDO object of the EtherCAT device in the cyclic EtherCAT frame.
ECATWriteSdo	Writes a 32-bit word to I/O nodes using a CANopen SDO write command.
ECATWriteSdoData	Writes data to an EtherCAT device using a CANopen SDO write command..
FSOEParamsInit	Transfers safety parameters from the safety master to safety slave devices to initialize the safety network.

Remarks

The EtherCAT SDO function blocks are activated by the CANopen over EtherCAT ([CoE](#)) protocol in a client/server mode.

- The client (aka EtherCAT master) is the KAS Runtime application.
- The servers (aka EtherCAT slaves) are the drives and I/O nodes where data can be retrieved.
- The SDO function blocks only support the reading and writing of 32-bit values.
 - It is the fundamental size of CANopen SDO calls.

1.2.0.1 Why use ECATReadSdo and ECATWriteSdo FBs?

- The "ECATReadSdo" (→ p. 68) and "ECATWriteSdo" (→ p. 89) response time is faster.
- The "ECATReadSdoData" (→ p. 74) and "ECATWriteSdoData" (→ p. 95) are used when the data size is greater than 4 bytes.
- They are preferred over "DriveParamRead" (→ p. 24) and "DriveParamWrite" (→ p. 33).

1.2.0.2 Why use the DriveParam FBs?

Reasons to prefer the DriveParam FBs are:

- They allow direct use of the parameter name.
 - Example: IL.LIMITP instead of the SDO index: 356Eh
- They can be used to setup a drive terminal in the HMI application.
 - This is similar to the [Terminal](#) view available in the AKD widget embedded in the KAS-IDE.

1.2.1 EtherCAT Function Blocks that Work with Drive Parameters

These function blocks are used to work with drive parameters not supported by ML and MC function blocks.

- They support reading and writing drive parameters using the non-cyclic SDO channel in the EtherCAT network.
- The ASCII name for the parameter is used as an input.

1.2.1.1 Execution Time

These function blocks typically take a longer time to execute (up to ten cycles to finish executing). It takes the same amount of time to Read or Write a parameter.

NOTE

It takes more than one cycle to execute these function blocks (but less than 100 ms).

1.2.1.1.1 Reason

It is not only linked to the ASCII SDO communication. Because these FBs are waiting for the AKD drive to respond, the execution time can also increase due to the load of the AKD firmware at the time you call them.

1.2.1.1.2 Result

The PLC code is overrunning the cycle duration. as explained in Tasking Model / Scheduling. As a consequence, you can see this message in the Controller Log window:

The Virtual Machine missed 1 cycle(s) of PLC execution.

1.2.1.1.3 Solution

When this happens we recommend:

- Use these function blocks sparingly in programs
- Rely on the EtherCAT read/write SDO function blocks whenever possible
- Smooth the load of the PLC code by executing these function blocks at the required update rate.

Stats about the FB Execution Time

- There is a small difference in timing when running EtherCAT at 2ms compared to other frequencies.

	0.25, 0.5, 1ms	2ms
Mean	9ms	14ms
Min	3ms	8ms
Max	16ms	24ms

- **Max** time to consider when executing a single SDO command, (i.e., before the Done output becomes TRUE): **24ms**.
- **Max** time to consider when executing a single Drive Parameter command (i.e. before the Done output becomes TRUE): **60 ms**.

	4 kHz	1 kHz
Mean	20 ms	11 ms
Min	15 ms	9 ms
Max	45 ms	58 ms

- When sending multiple commands to a single drive, only one command can be sent at a time.
 - The time to execute multiple commands is:
Number of commands x Execution time of a single command

- When commands are sent to different AKD drives at the same time, the requests do not interfere with each other.
 - The function finishes execution in the same maximum time as one drive.

1.2.2 EtherCAT Function Blocks that Work with SDOs

"ECATReadSdo" (→ p. 68), "ECATReadSdoData" (→ p. 74), "ECATWriteSdo" (→ p. 89), and "ECATWriteSdoData" (→ p. 95) are used to work with drive or remote I/O parameters that are not supported by ML and MC function blocks.

Drive or remote I/O parameters that have an associated SDO number can be read and written using these function blocks.

NOTE

It takes more than one cycle to execute these function blocks (but less than 100 ms).

Stats about the FB Execution Time

- There is a small difference in timing when running EtherCAT at 2ms compared to other frequencies.

	0.25, 0.5, 1ms	2ms
Mean	9ms	14ms
Min	3ms	8ms
Max	16ms	24ms

- Max** time to consider when executing a single SDO command, (i.e., before the Done output becomes TRUE): **24ms**.
- Max** time to consider when executing a single Drive Parameter command (i.e. before the Done output becomes TRUE): **60 ms**.

	4 kHz	1 kHz
Mean	20 ms	11 ms
Min	15 ms	9 ms
Max	45 ms	58 ms

- When sending multiple commands to a single drive, only one command can be sent at a time.
 - The time to execute multiple commands is:
Number of commands x Execution time of a single command
- When commands are sent to different AKD drives at the same time, the requests do not interfere with each other.
 - The function finishes execution in the same maximum time as one drive.

1.2.3 DriveParamRead

PLCopen ✓

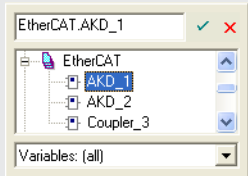
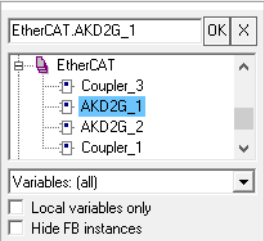
Pipe Network ✓

 **Function Block** - Reads a drive parameter by sending an ASCII command to a drive.

Notes

- This function block **uses and reserves** the EtherCAT SDO Channel.
- The SDO Channel remains reserved until the done output is TRUE.
- This FB should be called at each cycle until either the Done or Error output is TRUE.
 - If it is not called at each cycle, the rest of SDO communication (e.g., the AKD GUI Views) is blocked.
- Using this FB in SFC P0 or P1 steps is not recommended because these steps are executed only once.
- If this FB is used in P0 or P1, then it must be used in an SFC N step to ensure the FB completes.
- It takes multiple cycles to complete this function block.
- Typically only **one DriveParamRead** or **DriveParamWrite** function should be active for **each axis** at one time.
- If executing this function block continuously or if multiple times is required, add code that waits for this function block to complete (e.g., Done bit = 1) before executing it again.
 - See **stats** about the "Execution Time" (→ p. 21).
- See "EtherCAT Function Blocks that Work with Drive Parameters" (→ p. 21) about function blocks not supported by ML and MC function blocks.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge of Execute, a drive parameter is read. • The function block processes only one request at a time. • If Execute is toggled quickly so that another rising edge occurs before the function block has completed, the function block does not issue a second read command.
Drive	INT	-32768 to +32767	N/A	No default	The address of the drive where data is read. • The first node usually has the value 1001. • The second node usually has the value 1002. TIP Use the members of the EtherCAT structure to specify a device's address.  
Param	STRING	No range	N/A	No default	The parameter to read.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.
Error	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed with error.
ErrorID	DINT	No range	N/A	The DriveParamRead error result if Error is TRUE. - See "Error Codes" (→ p. 25). - Upon success, Error is set to 0 (zero).
Value	LREAL	No range	N/A	- The value of the drive parameter. - Value is only set when the function block has successfully completed.
Units	STRING	No range	N/A	- The value of the drive parameter. - Value is only set when the function block has successfully completed.

1.2.3.0.1 Usage

Use this function block to read drive parameters that are not supported by other function blocks.

Examples include:

- Drive bus voltage
- Drive display
- Fault history
- Motor temperature
- Present drive limit settings
- Present regeneration loading

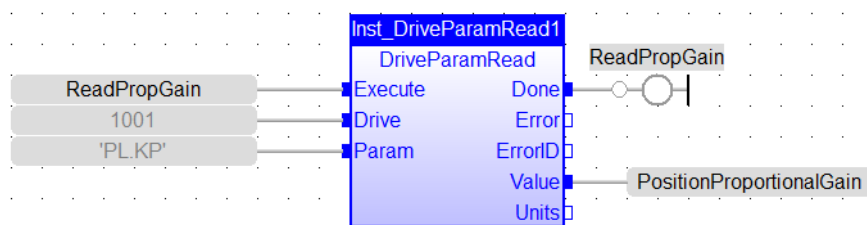
Error Codes

Value Dec (hex)	Error Code	Description
0	ECERR_OK	The SDO call succeeded.
1000 (0x3E8)	ECERR_DEVICE_ACTION_IS_INVALID	The requested Device Action is invalid.
1003 (0x3EB)	ECERR_DEVICE_ACTION_MINIMUM_ONE_NODE_REQD	A minimum of one device must be present in the EtherCAT network.
1004 (0x3EC)	ECERR_DEVICE_ACTION_MINIMUM_ONE_DC_NODE_REQD	A minimum of one Distributed clock capable device must be present in the EtherCAT network.

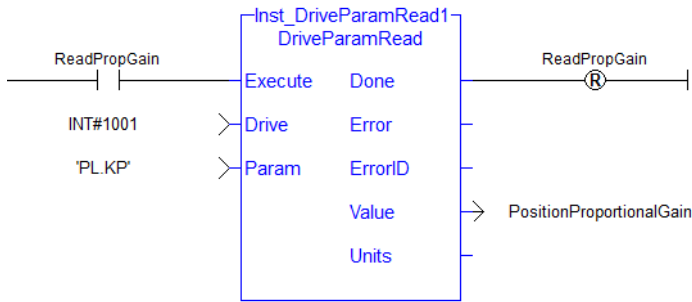
Value Dec (hex)	Error Code	Description
107 = 0x6B	ECERR_INVALID_ARRAY_SIZE	The size of the connections is too small. - When the array size is smaller than the number of bad connections: - The array is filled with the data to the size of the array. - The error ECERR_INVALID_ARRAY_SIZE is set. - In this scenario, the output ConnectionCount is set to the size of the array and it is smaller than the number of actual bad connections.
1792 (0x700)	ECERR_DEVICE_ERROR	EtherCAT device is not accessible.
1794 (0x702)	ECERR_DEVICE_INVALIDCMD	Invalid mailbox command.
1795 (0x703)	ECERR_DEVICE_INVALIDINDEX	An invalid value for the Index input was specified.
1796 (0x704)	ECERR_DEVICE_INVALIDACCESS	Reading of the variable is not permitted.
1797 (0x705)	ECERR_DEVICE_INVALIDSIZE	An invalid size for the parameter was specified.
1798 (0x706)	ECERR_DEVICE_INVALIDDATA	Invalid parameter value(s) in SDO index and/or sub-index.
1799 (0x707)	ECERR_DEVICE_NOTREADY	Device is not in a ready state. The network is not in operational.
1800 (0x708)	ECERR_DEVICE_BUSY	Device is not available to respond.
1801 (0x709)	ECERR_DEVICE_INVALIDCONTEXT	Device responded with an illegal error code, indicating the command is not allowed under the present conditions.
1802 (0x70A)	ECERR_DEVICE_NOMEMORY	EtherCAT mailbox is out of memory or device is out of disk space.
1803 (0x70B)	ECERR_DEVICE_INVALIDPARM	EtherCAT mailbox request was not valid.
1804 (0x70C)	ECERR_DEVICE_NOTFOUND	EtherCAT device not found.
1805 (0x70D)	ECERR_DEVICE_SYNTAX	An unexpected error occurred.
1810 (0x712)	ECERR_DEVICE_INVALIDSTATE	The EtherCAT device is in an invalid state.
1817 (0x719)	ECERR_DEVICE_TIMEOUT	The EtherCAT device failed to respond, timing out.
1826 (0x722)	ECERR_DEVICE_INSERTMAILBOX	Error while inserting the mailbox command into internal FIFO.
1827 (0x723)	ECERR_DEVICE_INVALIDOFFSET	An invalid value for the SubIndex input was specified.
1828 (0x724)	ECERR_DEVICE_UNKNOWNMAILBOXCMD	The master sent an unknown mailbox command to the slave.

Value Dec (hex)	Error Code	Description
1829 (0x725)	ECERR_DEVICE_ACCESSDENIED	Device responded with an invalid access error code, indicating the command is not allowed.
1832 (0x728)	ECERR_DEVICE_INVALIDADDR	The specified EtherCAT node address is invalid.
1836 (0x72c)	ECERR_DEVICE_NOT_A_FSOE_MASTER	Device is not a FSoE master.
1837 (0x072D)	ECERR_DEVICE_DISCONNECTED	The EtherCAT device is disconnected.
1920 (0x780)	ECERR_DEVICE_PARAM_ACCESS_ERROR	Unknown error occurred while accessing parameter.
1921 (0x781)	ECERR_DEVICE_PARAM_NOT_FOUND	Parameter was not found.
1922 (0x782)	ECERR_DEVICE_PARAM_NOT_INTEGER	Parameter is a floating-point value. Integer value required.
1923 (0x783)	ECERR_DEVICE_VALUE_IS_NEGATIVE	No negative values allowed. Value specified was negative.
1924 (0x784)	ECERR_DEVICE_VALUE_OUT_OF_RANGE	Value is out of data-range.
1925 (0x785)	ECERR_DEVICE_VALUE_GREATER_THAN_MAX	Value bigger than maximum.
1926 (0x786)	ECERR_DEVICE_VALUE_LOWER_THAN_MIN	Value lower than minimum.
2048 (0x800)	ECERR_CLIENT_ERROR	Error in Mailbox response to a previously sent mailbox command.
2049 (0x801)	ECERR_CLIENT_TIMEOUT	The SDO command timed out.
2050 (0x802)	ECERR_CLIENT_INVALIDPARM	An invalid value was specified.
2051 (0x803)	ECERR_CLIENT_INVALIDSIZE	An invalid value for the size input was specified.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* Read PL.KP on first AKD Drive on EtherCAT network *)

(* The code continually calls the FB (without re-executing it) until the
first execution is done, then reads the returned value from the drive and
reset the FB *)

IF ReadPropGain then
    Inst_DriveParamRead1( 1, 1001, 'PL.KP' );
End_If;

On Inst_DriveParamRead1.Done do
    Inst_DriveParamRead1( 0, 1001, 'PL.KP' );
    PositionProportionalGain := Inst_DriveParamRead1.Value; (* Reads the
returned value from the drive *)
    ReadPropGain := 0; (* Reset the FB *)
End_DO;
```

See Also

"DriveParamWrite" (→ p. 33)

1.2.4 DriveParamStrRead

PLCopen ✓

Pipe Network ✓



Function Block - Reads a single drive parameter by sending an ASCII command to a drive.

Notes

- This function block **uses and reserves** the EtherCAT SDO Channel.
- The SDO Channel remains reserved until the done output is TRUE.
- This FB should be called at each cycle until either the Done or Error output is TRUE.
 - If it is not called at each cycle, the rest of SDO communication (e.g., the AKD GUI Views) is blocked.
- Using this FB in SFC P0 or P1 steps is not recommended because these steps are executed only once.
- If this FB is used in P0 or P1, then it must be used in an SFC N step to ensure the FB completes.
- The value returned is the string response from the drive.
- See "EtherCAT Function Blocks that Work with Drive Parameters" (→ p. 21) about function blocks not supported by ML and MC function blocks.

NOTE

- This differs from "DriveParamRead" (→ p. 24) because the drive response is not parsed.
- **DriveParamRead** parses the drive response and returns the numeric value of the parameter and the units found that represent that parameter.
- Since **DriveParamStrRead** returns the drive response directly, it can be used to read parameters that have string representations (e.g., the AKD2G's AXIS#.FAULTMSG# parameters).

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	Executes the function block.
Drive	INT	-32768 to +32767	N/A	No default	The address of the drive from which data is read.
Param	STRING	No range	N/A	No default	The parameter to read.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.
Error	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed with error.
ErrorID	DINT	No range	N/A	• The DriveParamStrRead error result if Error is TRUE. • Upon success, Error is set to 0 (zero). • See "Error Codes" (→ p. 30).
Value	STRING	No range	N/A	• The value of the drive parameter. • Value is only set when the function block has successfully completed.

1.2.4.0.1 Usage

Use this function block to read drive parameters that are not supported by other function blocks.

Examples include:

- Drive bus voltage
- Drive display
- Fault history
- Motor temperature
- Present drive limit settings
- Present regeneration loading

Error Codes

Value Dec (hex)	Error Code	Description
0	ECERR_OK	The SDO call succeeded.
1000 (0x3E8)	ECERR_DEVICE_ACTION_IS_INVALID	The requested Device Action is invalid.
1003 (0x3EB)	ECERR_DEVICE_ACTION_MINIMUM_ONE_NODE_REQD	A minimum of one device must be present in the EtherCAT network.
1004 (0x3EC)	ECERR_DEVICE_ACTION_MINIMUM_ONE_DC_NODE_REQD	A minimum of one Distributed clock capable device must be present in the EtherCAT network.
107 = 0x6B	ECERR_INVALID_ARRAY_SIZE	The size of the connections is too small. • When the array size is smaller than the number of bad connections: • The array is filled with the data to the size of the array. • The error ECERR_INVALID_ARRAY_SIZE is set. • In this scenario, the output ConnectionCount is set to the size of the array and it is smaller than the number of actual bad connections.
1792 (0x700)	ECERR_DEVICE_ERROR	EtherCAT device is not accessible.
1794 (0x702)	ECERR_DEVICE_INVALIDCMD	Invalid mailbox command.
1795 (0x703)	ECERR_DEVICE_INVALIDINDEX	An invalid value for the Index input was specified.
1796 (0x704)	ECERR_DEVICE_INVALIDACCESS	Reading of the variable is not permitted.
1797 (0x705)	ECERR_DEVICE_INVALIDSIZE	An invalid size for the parameter was specified.
1798 (0x706)	ECERR_DEVICE_INVALIDDATA	Invalid parameter value(s) in SDO index and/or sub-index.
1799 (0x707)	ECERR_DEVICE_NOTREADY	Device is not in a ready state. The network is not in operational.
1800 (0x708)	ECERR_DEVICE_BUSY	Device is not available to respond.

Value Dec (hex)	Error Code	Description
1801 (0x709)	ECERR_DEVICE_INVALIDCONTEXT	Device responded with an illegal error code, indicating the command is not allowed under the present conditions.
1802 (0x70A)	ECERR_DEVICE_NOMEMORY	EtherCAT mailbox is out of memory or device is out of disk space.
1803 (0x70B)	ECERR_DEVICE_INVALIDPARM	EtherCAT mailbox request was not valid.
1804 (0x70C)	ECERR_DEVICE_NOTFOUND	EtherCAT device not found.
1805 (0x70D)	ECERR_DEVICE_SYNTAX	An unexpected error occurred.
1810 (0x712)	ECERR_DEVICE_INVALIDSTATE	The EtherCAT device is in an invalid state.
1817 (0x719)	ECERR_DEVICE_TIMEOUT	The EtherCAT device failed to respond, timing out.
1826 (0x722)	ECERR_DEVICE_INSERTMAILBOX	Error while inserting the mailbox command into internal FIFO.
1827 (0x723)	ECERR_DEVICE_INVALIDOFFSET	An invalid value for the SubIndex input was specified.
1828 (0x724)	ECERR_DEVICE_UNKNOWNMAILBOXCMD	The master sent an unknown mailbox command to the slave.
1829 (0x725)	ECERR_DEVICE_ACCESSDENIED	Device responded with an invalid access error code, indicating the command is not allowed.
1832 (0x728)	ECERR_DEVICE_INVALIDADDR	The specified EtherCAT node address is invalid.
1836 (0x72c)	ECERR_DEVICE_NOT_A_FSOE_MASTER	Device is not a FSoE master.
1837 (0x072D)	ECERR_DEVICE_DISCONNECTED	The EtherCAT device is disconnected.
1920 (0x780)	ECERR_DEVICE_PARAM_ACCESS_ERROR	Unknown error occurred while accessing parameter.
1921 (0x781)	ECERR_DEVICE_PARAM_NOT_FOUND	Parameter was not found.
1922 (0x782)	ECERR_DEVICE_PARAM_NOT_INTEGER	Parameter is a floating-point value. Integer value required.
1923 (0x783)	ECERR_DEVICE_VALUE_IS_NEGATIVE	No negative values allowed. Value specified was negative.
1924 (0x784)	ECERR_DEVICE_VALUE_OUT_OF_RANGE	Value is out of data-range.
1925 (0x785)	ECERR_DEVICE_VALUE_GREATER_THAN_MAX	Value bigger than maximum.
1926 (0x786)	ECERR_DEVICE_VALUE_LOWER_THAN_MIN	Value lower than minimum.
2048 (0x800)	ECERR_CLIENT_ERROR	Error in Mailbox response to a previously sent mailbox command.

Value Dec (hex)	Error Code	Description
2049 (0x801)	ECERR_CLIENT_TIMEOUT	The SDO command timed out.
2050 (0x802)	ECERR_CLIENT_INVALIDPARM	An invalid value was specified.
2051 (0x803)	ECERR_CLIENT_INVALIDSIZE	An invalid value for the size input was specified.

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* Read AXIS1.FAULTMSG1 on first AKD2G Drive on EtherCAT network *)

(* The code continually calls the FB (without re-executing it) until the
first execution is done, then reads the returned value from the drive and
reset the FB *)

IF ReadFaultMsg Then
    Inst_DriveParamStrRead1(True, 1001, 'AXIS1.FAULTMSG1' );
End_If;

On Inst_DriveParamStrRead1 Do
    FaultMsg := Inst_DriveParamStrRead1.Value; (* Reads the returned value
from the drive *)
    Inst_DriveParamStrRead1(False, 1001, 'AXIS1.FAULTMSG1');
    ReadFaultMsg := False; (* Reset the FB *)
End_DO;
```

See Also

- Differences between Functions and Function Blocks
- "DriveParamRead" (→ p. 24)
- "DriveParamWrite" (→ p. 33)

1.2.5 DriveParamWrite

PLCopen 

Pipe Network 



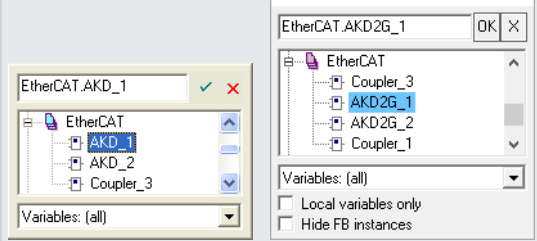
Function Block - Writes a drive parameter by sending an ASCII command to a drive.

Notes

- This function block **uses and reserves** the EtherCAT SDO Channel.
- The SDO Channel remains reserved until the done output is TRUE.
- This FB should be called at each cycle until either the Done or Error output is TRUE.
 - If it is not called at each cycle, the rest of SDO communication (e.g., the AKD GUI Views) is blocked.
- Using this FB in SFC P0 or P1 steps is not recommended because these steps are executed only once.
- If this FB is used in P0 or P1, then it must be used in an SFC N step to ensure the FB completes.
- Use this function block to change drive parameters.
Examples include:
 - Tuning parameters.
 - Changing drive limits (i.e., peak current).
- It takes multiple cycles to complete this function block.
- Typically only **one DriveParamRead** or **DriveParamWrite** function should be active for **each axis** at one time.
- If executing this function block continuously or if multiple times is required, add code that waits for this function block to complete (e.g., Done bit = 1) before executing it again.
 - See **stats** about the "Execution Time" (→ p. 21).
- See "EtherCAT Function Blocks that Work with Drive Parameters" (→ p. 21) about function blocks not supported by ML and MC function blocks.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge of Execute, a drive parameter is set. • The function block processes only one request at a time. • If Execute is toggled quickly so that another rising edge occurs before the function block has completed, the function block does not issue a second write command.

Input	Data Type	Range	Unit	Default	Description
Drive	INT	-32768 to +32767	N/A	No default	The address of the drive where data is written. - The first node usually has the value 1001. - The second node usually has the value 1002. TIP Use the members of the EtherCAT structure to specify a device's address. 
Param	STRING	No range	N/A	No default	The parameter to write.
Value	LREAL	No range	N/A	No default	The value to set the drive parameter to.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.
Error	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed with error.
ErrorID	DINT	No range	N/A	The DriveParamWrite error result if Error is TRUE. Upon success, Error is set to 0 (zero). See "Error Codes" (→ p. 34).

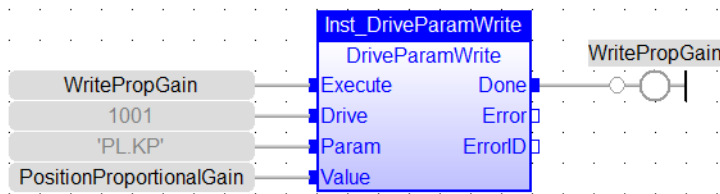
Error Codes

Value Dec (hex)	Error Code	Description
0	ECERR_OK	The SDO call succeeded.
1000 (0x3E8)	ECERR_DEVICE_ACTION_IS_INVALID	The requested Device Action is invalid.
1003 (0x3EB)	ECERR_DEVICE_ACTION_MINIMUM_ONE_NODE_REQD	A minimum of one device must be present in the EtherCAT network.
1004 (0x3EC)	ECERR_DEVICE_ACTION_MINIMUM_ONE_DC_NODE_REQD	A minimum of one Distributed clock capable device must be present in the EtherCAT network.

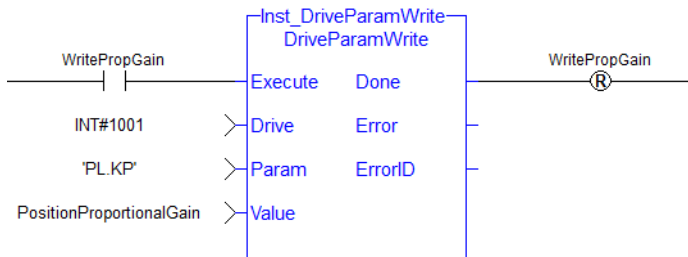
Value Dec (hex)	Error Code	Description
107 = 0x6B	ECERR_INVALID_ARRAY_SIZE	The size of the connections is too small. - When the array size is smaller than the number of bad connections: - The array is filled with the data to the size of the array. - The error ECERR_INVALID_ARRAY_SIZE is set. - In this scenario, the output ConnectionCount is set to the size of the array and it is smaller than the number of actual bad connections.
1792 (0x700)	ECERR_DEVICE_ERROR	EtherCAT device is not accessible.
1794 (0x702)	ECERR_DEVICE_INVALIDCMD	Invalid mailbox command.
1795 (0x703)	ECERR_DEVICE_INVALIDINDEX	An invalid value for the Index input was specified.
1796 (0x704)	ECERR_DEVICE_INVALIDACCESS	Reading of the variable is not permitted.
1797 (0x705)	ECERR_DEVICE_INVALIDSIZE	An invalid size for the parameter was specified.
1798 (0x706)	ECERR_DEVICE_INVALIDDATA	Invalid parameter value(s) in SDO index and/or sub-index.
1799 (0x707)	ECERR_DEVICE_NOTREADY	Device is not in a ready state. The network is not in operational.
1800 (0x708)	ECERR_DEVICE_BUSY	Device is not available to respond.
1801 (0x709)	ECERR_DEVICE_INVALIDCONTEXT	Device responded with an illegal error code, indicating the command is not allowed under the present conditions.
1802 (0x70A)	ECERR_DEVICE_NOMEMORY	EtherCAT mailbox is out of memory or device is out of disk space.
1803 (0x70B)	ECERR_DEVICE_INVALIDPARM	EtherCAT mailbox request was not valid.
1804 (0x70C)	ECERR_DEVICE_NOTFOUND	EtherCAT device not found.
1805 (0x70D)	ECERR_DEVICE_SYNTAX	An unexpected error occurred.
1810 (0x712)	ECERR_DEVICE_INVALIDSTATE	The EtherCAT device is in an invalid state.
1817 (0x719)	ECERR_DEVICE_TIMEOUT	The EtherCAT device failed to respond, timing out.
1826 (0x722)	ECERR_DEVICE_INSERTMAILBOX	Error while inserting the mailbox command into internal FIFO.
1827 (0x723)	ECERR_DEVICE_INVALIDOFFSET	An invalid value for the SubIndex input was specified.
1828 (0x724)	ECERR_DEVICE_UNKNOWNMAILBOXCMD	The master sent an unknown mailbox command to the slave.

Value Dec (hex)	Error Code	Description
1829 (0x725)	ECERR_DEVICE_ACCESSDENIED	Device responded with an invalid access error code, indicating the command is not allowed.
1832 (0x728)	ECERR_DEVICE_INVALIDADDR	The specified EtherCAT node address is invalid.
1836 (0x72c)	ECERR_DEVICE_NOT_A_FSOE_MASTER	Device is not a FSoE master.
1837 (0x072D)	ECERR_DEVICE_DISCONNECTED	The EtherCAT device is disconnected.
1920 (0x780)	ECERR_DEVICE_PARAM_ACCESS_ERROR	Unknown error occurred while accessing parameter.
1921 (0x781)	ECERR_DEVICE_PARAM_NOT_FOUND	Parameter was not found.
1922 (0x782)	ECERR_DEVICE_PARAM_NOT_INTEGER	Parameter is a floating-point value. Integer value required.
1923 (0x783)	ECERR_DEVICE_VALUE_IS_NEGATIVE	No negative values allowed. Value specified was negative.
1924 (0x784)	ECERR_DEVICE_VALUE_OUT_OF_RANGE	Value is out of data-range.
1925 (0x785)	ECERR_DEVICE_VALUE_GREATER_THAN_MAX	Value bigger than maximum.
1926 (0x786)	ECERR_DEVICE_VALUE_LOWER_THAN_MIN	Value lower than minimum.
2048 (0x800)	ECERR_CLIENT_ERROR	Error in Mailbox response to a previously sent mailbox command.
2049 (0x801)	ECERR_CLIENT_TIMEOUT	The SDO command timed out.
2050 (0x802)	ECERR_CLIENT_INVALIDPARM	An invalid value was specified.
2051 (0x803)	ECERR_CLIENT_INVALIDSIZE	An invalid value for the size input was specified.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* Write 58.000 to PL.KP of first AKD Drive on EtherCAT network *)
Inst_DriveParamWrite( TRUE, 1001, 'PL.KP', 58 );
```

See Also

"DriveParamRead" (→ p. 24)

1.2.6 ECATCommErrors

PLCopen 



Function Block - Returns a list of bad EtherCAT connections.

If EtherCAT network communication is shutdown, the failed connections are based on information that was taken at the time the network was shutdown.



TIP

See Check the Connections for Errors for an example of implementing this function.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	Read the communication errors on the rising edge.
Connection	ECATCommErr_ref See the "ECATCommErr_ref Structure" (→ p. 38) table.	N = 0 to 2 times the number of EtherCAT devices.	N/A	No default	Array of bad connections. The safe size for the list is: [2 * (number of devices) - 1].

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.
Error	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed with error.
ErrorID	DINT	No range	N/A	The function call error result, if Error is TRUE. • See "Error Codes" (→ p. 39). • Upon success, Error is set to 0 (zero).
ConnectionCount	UINT	No range	N/A	The number of bad connections. Valid indices in the Connection array range from 0 (zero) to ConnectionCount -1 (assuming ConnectionCount != 0).

1.2.6.0.1 ECATCommErr_ref Structure

Parameter	Type	Description
CommErrorCounter	UINT	The Communication Error Counter for this port.

Parameter	Type	Description
ConnectedSlaveAddress	INT	The EtherCAT address of the connected device.
ConnectedSlavePortID	UINT	The port number of the connected device.
LostLinkCounter	UINT	The Lost Link Counter for this port.
SlaveAddress	INT	The EtherCAT address of the device that owns the port.
SlavePortID	UINT	The port number.

1.2.6.0.2 EtherCAT Port Numbers

Define	Port
#define EC_PORT_A	0 (* Port A *)
#define EC_PORT_B	1 (* Port B *)
#define EC_PORT_C	2 (* Port C *)
#define EC_PORT_D	3 (* Port D*)

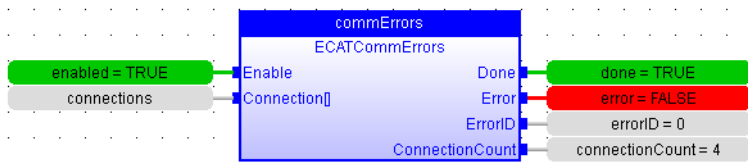
Error Codes

Value Dec (hex)	Error Code	Description
0	ECERR_OK	The SDO call succeeded.
1000 (0x3E8)	ECERR_DEVICE_ACTION_IS_INVALID	The requested Device Action is invalid.
1003 (0x3EB)	ECERR_DEVICE_ACTION_MINIMUM_ONE_NODE_REQD	A minimum of one device must be present in the EtherCAT network.
1004 (0x3EC)	ECERR_DEVICE_ACTION_MINIMUM_ONE_DC_NODE_REQD	A minimum of one Distributed clock capable device must be present in the EtherCAT network.
107 = 0x6B	ECERR_INVALID_ARRAY_SIZE	The size of the connections is too small. - When the array size is smaller than the number of bad connections: - The array is filled with the data to the size of the array. - The error ECERR_INVALID_ARRAY_SIZE is set. - In this scenario, the output ConnectionCount is set to the size of the array and it is smaller than the number of actual bad connections.
1792 (0x700)	ECERR_DEVICE_ERROR	EtherCAT device is not accessible.
1794 (0x702)	ECERR_DEVICE_INVALIDCMD	Invalid mailbox command.
1795 (0x703)	ECERR_DEVICE_INVALIDINDEX	An invalid value for the Index input was specified.
1796 (0x704)	ECERR_DEVICE_INVALIDACCESS	Reading of the variable is not permitted.

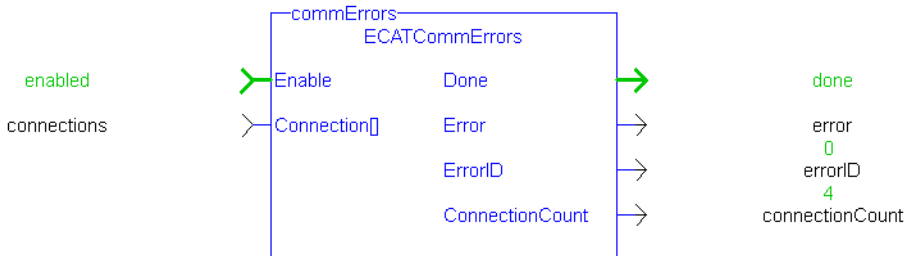
Value Dec (hex)	Error Code	Description
1797 (0x705)	ECERR_DEVICE_INVALIDSIZE	An invalid size for the parameter was specified.
1798 (0x706)	ECERR_DEVICE_INVALIDDATA	Invalid parameter value(s) in SDO index and/or sub-index.
1799 (0x707)	ECERR_DEVICE_NOTREADY	Device is not in a ready state. The network is not in operational.
1800 (0x708)	ECERR_DEVICE_BUSY	Device is not available to respond.
1801 (0x709)	ECERR_DEVICE_INVALIDCONTEXT	Device responded with an illegal error code, indicating the command is not allowed under the present conditions.
1802 (0x70A)	ECERR_DEVICE_NOMEMORY	EtherCAT mailbox is out of memory or device is out of disk space.
1803 (0x70B)	ECERR_DEVICE_INVALIDPARM	EtherCAT mailbox request was not valid.
1804 (0x70C)	ECERR_DEVICE_NOTFOUND	EtherCAT device not found.
1805 (0x70D)	ECERR_DEVICE_SYNTAX	An unexpected error occurred.
1810 (0x712)	ECERR_DEVICE_INVALIDSTATE	The EtherCAT device is in an invalid state.
1817 (0x719)	ECERR_DEVICE_TIMEOUT	The EtherCAT device failed to respond, timing out.
1826 (0x722)	ECERR_DEVICE_INSERTMAILBOX	Error while inserting the mailbox command into internal FIFO.
1827 (0x723)	ECERR_DEVICE_INVALIDOFFSET	An invalid value for the SubIndex input was specified.
1828 (0x724)	ECERR_DEVICE_UNKNOWNMAILBOXCMD	The master sent an unknown mailbox command to the slave.
1829 (0x725)	ECERR_DEVICE_ACCESSDENIED	Device responded with an invalid access error code, indicating the command is not allowed.
1832 (0x728)	ECERR_DEVICE_INVALIDADDR	The specified EtherCAT node address is invalid.
1836 (0x72c)	ECERR_DEVICE_NOT_A_FSOE_MASTER	Device is not a FSoE master.
1837 (0x072D)	ECERR_DEVICE_DISCONNECTED	The EtherCAT device is disconnected.
1920 (0x780)	ECERR_DEVICE_PARAM_ACCESS_ERROR	Unknown error occurred while accessing parameter.
1921 (0x781)	ECERR_DEVICE_PARAM_NOT_FOUND	Parameter was not found.
1922 (0x782)	ECERR_DEVICE_PARAM_NOT_INTEGER	Parameter is a floating-point value. Integer value required.
1923 (0x783)	ECERR_DEVICE_VALUE_IS_NEGATIVE	No negative values allowed. Value specified was negative.

Value Dec (hex)	Error Code	Description
1924 (0x784)	ECERR_DEVICE_VALUE_OUT_OF_RANGE	Value is out of data-range.
1925 (0x785)	ECERR_DEVICE_VALUE_GREATER_THAN_MAX	Value bigger than maximum.
1926 (0x786)	ECERR_DEVICE_VALUE_LOWER_THAN_MIN	Value lower than minimum.
2048 (0x800)	ECERR_CLIENT_ERROR	Error in Mailbox response to a previously sent mailbox command.
2049 (0x801)	ECERR_CLIENT_TIMEOUT	The SDO command timed out.
2050 (0x802)	ECERR_CLIENT_INVALIDPARM	An invalid value was specified.
2051 (0x803)	ECERR_CLIENT_INVALIDSIZE	An invalid value for the size input was specified.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```

(*****)
Read EtherCAT communication errors.
(*****)
commErrors( TRUE, Connection);

```

See Also

- "ECATDeviceStatus" (→ p. 47)
- "ECATMasterStatus" (→ p. 59)

1.2.7 ECATDeviceAction

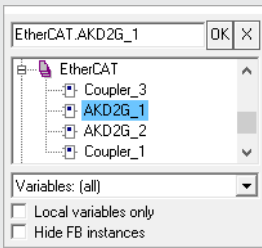
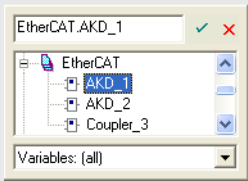
PLCopen ✓

Pipe Network ✓

 **Function Block** - Performs an action on an EtherCAT device.

- This function returns immediately.
- It can be called multiple times in one cycle.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	When Execute is set to TRUE, an action is performed.
SlaveAddress	INT	-32768 to +32767	N/A	No default	The address of the device to perform an action on. • The first node usually has the value 1001. • The second node usually has the value 1002. TIP Use the members of the EtherCAT structure to specify a device's address. 
Action	INT	-32768 to +32767	N/A	No default	The action to be performed. Action can be one of these values: • DEVICE_ACTION_CONNECT • Connects a slave into the EtherCAT network. • This action can only be executed when the motion engine is stopped. • DEVICE_ACTION_DISCONNECT • Disconnects a slave from the EtherCAT network. • This action can only be executed when the motion engine is stopped.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.
Error	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed with error.
ErrorID	DINT	No range	N/A	The ECATDeviceAction error result if Error is TRUE. • See "Error Codes" (→ p. 43). • Upon success, Error is set to 0 (zero).

1.2.7.0.1 Supported Actions

DEVICE_ACTION_DISCONNECT

- The EtherCAT master is notified to expect the node to be removed from the EtherCAT when ECATDeviceAction is called with DEVICE_ACTION_DISCONNECT.
- Any Axis/Axes mapped to a drive node is automatically simulated.
- The last position from the physical drive is carried over to the simulation when the EtherCAT restarts.

DEVICE_ACTION_CONNECT

- This action connects the already disconnected node to the EtherCAT Network.
- The axis (or axes) that was acting as simulated axis becomes a normal axis when the drive node is connected.
- The axis/axes position comes directly from the configured drive feedback.
 - The position is **not** automatically transferred from the simulated axis.

1.2.7.0.2 Usage

The EtherCAT network needs to be stopped to use this function block.

- This is achieved by either:
 - Calling **ECATDeviceAction** prior to calling "MLMotionStart" (→ p. 872).
 - First calling "MLMotionStop" (→ p. 878) to stop the network.
- See Modular EtherCAT Concept for more information on modular machine design.

Error Codes

Value Dec (hex)	Error Code	Description
0	ECERR_OK	The SDO call succeeded.
1000 (0x3E8)	ECERR_DEVICE_ACTION_IS_INVALID	The requested Device Action is invalid.
1003 (0x3EB)	ECERR_DEVICE_ACTION_MINIMUM_ONE_NODE_REQD	A minimum of one device must be present in the EtherCAT network.
1004 (0x3EC)	ECERR_DEVICE_ACTION_MINIMUM_ONE_DC_NODE_REQD	A minimum of one Distributed clock capable device must be present in the EtherCAT network.

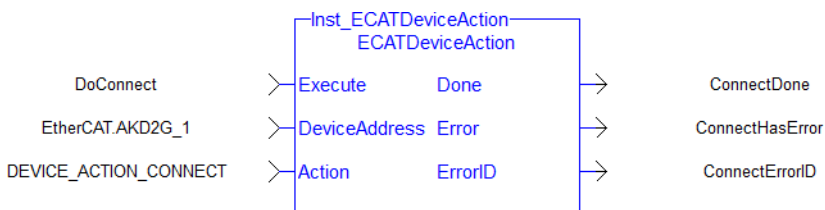
Value Dec (hex)	Error Code	Description
107 = 0x6B	ECERR_INVALID_ARRAY_SIZE	The size of the connections is too small. - When the array size is smaller than the number of bad connections: - The array is filled with the data to the size of the array. - The error ECERR_INVALID_ARRAY_SIZE is set. - In this scenario, the output ConnectionCount is set to the size of the array and it is smaller than the number of actual bad connections.
1792 (0x700)	ECERR_DEVICE_ERROR	EtherCAT device is not accessible.
1794 (0x702)	ECERR_DEVICE_INVALIDCMD	Invalid mailbox command.
1795 (0x703)	ECERR_DEVICE_INVALIDINDEX	An invalid value for the Index input was specified.
1796 (0x704)	ECERR_DEVICE_INVALIDACCESS	Reading of the variable is not permitted.
1797 (0x705)	ECERR_DEVICE_INVALIDSIZE	An invalid size for the parameter was specified.
1798 (0x706)	ECERR_DEVICE_INVALIDDATA	Invalid parameter value(s) in SDO index and/or sub-index.
1799 (0x707)	ECERR_DEVICE_NOTREADY	Device is not in a ready state. The network is not in operational.
1800 (0x708)	ECERR_DEVICE_BUSY	Device is not available to respond.
1801 (0x709)	ECERR_DEVICE_INVALIDCONTEXT	Device responded with an illegal error code, indicating the command is not allowed under the present conditions.
1802 (0x70A)	ECERR_DEVICE_NOMEMORY	EtherCAT mailbox is out of memory or device is out of disk space.
1803 (0x70B)	ECERR_DEVICE_INVALIDPARM	EtherCAT mailbox request was not valid.
1804 (0x70C)	ECERR_DEVICE_NOTFOUND	EtherCAT device not found.
1805 (0x70D)	ECERR_DEVICE_SYNTAX	An unexpected error occurred.
1810 (0x712)	ECERR_DEVICE_INVALIDSTATE	The EtherCAT device is in an invalid state.
1817 (0x719)	ECERR_DEVICE_TIMEOUT	The EtherCAT device failed to respond, timing out.
1826 (0x722)	ECERR_DEVICE_INSERTMAILBOX	Error while inserting the mailbox command into internal FIFO.
1827 (0x723)	ECERR_DEVICE_INVALIDOFFSET	An invalid value for the SubIndex input was specified.
1828 (0x724)	ECERR_DEVICE_UNKNOWNMAILBOXCMD	The master sent an unknown mailbox command to the slave.

Value Dec (hex)	Error Code	Description
1829 (0x725)	ECERR_DEVICE_ACCESSDENIED	Device responded with an invalid access error code, indicating the command is not allowed.
1832 (0x728)	ECERR_DEVICE_INVALIDADDR	The specified EtherCAT node address is invalid.
1836 (0x72c)	ECERR_DEVICE_NOT_A_FSOE_MASTER	Device is not a FSoE master.
1837 (0x072D)	ECERR_DEVICE_DISCONNECTED	The EtherCAT device is disconnected.
1920 (0x780)	ECERR_DEVICE_PARAM_ACCESS_ERROR	Unknown error occurred while accessing parameter.
1921 (0x781)	ECERR_DEVICE_PARAM_NOT_FOUND	Parameter was not found.
1922 (0x782)	ECERR_DEVICE_PARAM_NOT_INTEGER	Parameter is a floating-point value. Integer value required.
1923 (0x783)	ECERR_DEVICE_VALUE_IS_NEGATIVE	No negative values allowed. Value specified was negative.
1924 (0x784)	ECERR_DEVICE_VALUE_OUT_OF_RANGE	Value is out of data-range.
1925 (0x785)	ECERR_DEVICE_VALUE_GREATER_THAN_MAX	Value bigger than maximum.
1926 (0x786)	ECERR_DEVICE_VALUE_LOWER_THAN_MIN	Value lower than minimum.
2048 (0x800)	ECERR_CLIENT_ERROR	Error in Mailbox response to a previously sent mailbox command.
2049 (0x801)	ECERR_CLIENT_TIMEOUT	The SDO command timed out.
2050 (0x802)	ECERR_CLIENT_INVALIDPARM	An invalid value was specified.
2051 (0x803)	ECERR_CLIENT_INVALIDSIZE	An invalid value for the size input was specified.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLMotionStop();  
MotionEngineStatus := MLMotionStatus();  
On MotionEngineStatus = MLSTATUS_STOPPED Do  
    Inst_ECATDeviceAction(True, EtherCAT.AKD_3, DEVICE_ACTION_DISCONNECT);  
    If Inst_ECATDeviceAction.Error Then // Handle Error  
        // Error ID value is in Inst_ECATDeviceAction.ErrorEnd_If;  
End_Do;
```

See Also

- "MLMotionStart" (→ p. 872)
- "MLMotionStop" (→ p. 878)

1.2.8 ECATDeviceStatus

PLCopen 

Pipe Network 



Function Block - Provides the EtherCAT state and the port link status information for the EtherCAT device.

- If the EtherCAT network communication is not running due to a shutdown, the device status contains information taken at the time the network was shutdown.
- This function block is useful in locating the devices with communication errors when the "ECATWCStatus" (→ p. 80) function indicates there are EtherCAT communication errors.

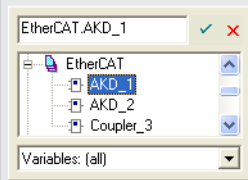
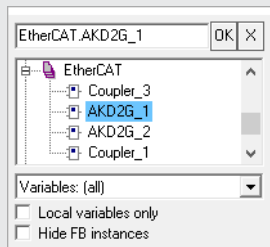
TIP

See Check the Device States in EtherCAT Communication Diagnosis Steps for an example of implementing this function.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	Read the device status on the rising edge.
DeviceAddress	INT	-32768 to +32767	N/A	No default	The address of the device data is read from. • The first node usually has the value 1001. • The second node usually has the value 1002.

 **TIP**
 Use the members of the EtherCAT structure to specify a device's address.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.

Output	Data Type	Range	Unit	Description
State	UINT	No range	N/A	Indicates the EtherCAT state of the device. • See "State Defines" (→ p. 48). • A value of 0 (zero) indicates there is no communication with the device and the state is unknown. • Bits 3:0 indicate the actual state of the device. • An EC_STATE_ERROR (bit 4 set to 1) indicates the device is not in the EtherCAT master requested state due to error conditions (e.g., loss of communication).
LinkStatus	UINT	No range	N/A	Provides the physical link status of the device's ports. • See "LinkStatus Defines" (→ p. 48). • If no communication is possible with the device, bit 0 is set to 1. • If a link is detected on a port (A-D), the corresponding bit (4-7) is set to 1. • If no link is detected the corresponding bit is set to 0.
Error	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed with error.
ErrorID	DINT	No range	N/A	The function call error result, if Error is TRUE. • See "Error Codes" (→ p. 49). • Upon success, Error is set to 0 (zero).

1.2.8.0.1 State Defines

```

#define EC_STATE_NO_COMMUNICATION 0 (* 0x00 = No Communication to
device *)
#define EC_STATE_INIT 1 (* 0x01 = Device in Init state *)
#define EC_STATE_PREOP 2 (* 0x02 = Device in Pre-operational
state *)
#define EC_STATE_BOOTSTRAP 3 (* 0x03 = Device in Bootstrap state
*)
#define EC_STATE_SAFEOP 4 (* 0x04 = Device in Safe-
Operational state *)
#define EC_STATE_OP 8 (* 0x08 = Device in Operational
state *)
#define EC_STATE_ERROR 16 (* 0x10 bit 4 set to 1; Device not
in requested state error *)

```

1.2.8.0.2 LinkStatus Defines

```

#define EC_LINK_NO_COMMUNICATION 1 (* 0x1 = No communication to device;
bit 0 set to 1 *)
#define EC_LINK_PORT_A 16 (* 0x10 = Link detected on Port A;
bit 4 set to 1 *)
#define EC_LINK_PORT_B 32 (* 0x20 = Link detected on Port B;
bit 5 set to 1 *)
#define EC_LINK_PORT_C 64 (* 0x40 = Link detected on Port C;

```

```

bit 6 set to 1 *)
#define EC_LINK_PORT_D 128 (* 0x80 = Link detected on Port D;
bit 7 set to 1 *)

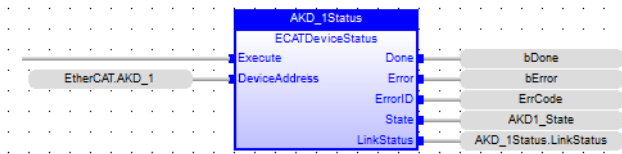
```

Error Codes

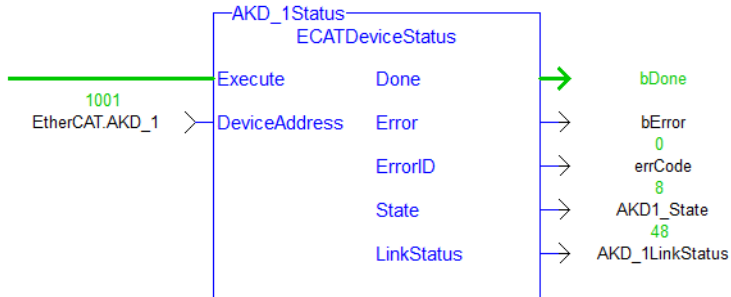
Value Dec (hex)	Error Code	Description
0	ECERR_OK	The SDO call succeeded.
1000 (0x3E8)	ECERR_DEVICE_ACTION_IS_INVALID	The requested Device Action is invalid.
1003 (0x3EB)	ECERR_DEVICE_ACTION_MINIMUM_ONE_NODE_REQD	A minimum of one device must be present in the EtherCAT network.
1004 (0x3EC)	ECERR_DEVICE_ACTION_MINIMUM_ONE_DC_NODE_REQD	A minimum of one Distributed clock capable device must be present in the EtherCAT network.
107 = 0x6B	ECERR_INVALID_ARRAY_SIZE	The size of the connections is too small. - When the array size is smaller than the number of bad connections: - The array is filled with the data to the size of the array. - The error ECERR_INVALID_ARRAY_SIZE is set. - In this scenario, the output ConnectionCount is set to the size of the array and it is smaller than the number of actual bad connections.
1792 (0x700)	ECERR_DEVICE_ERROR	EtherCAT device is not accessible.
1794 (0x702)	ECERR_DEVICE_INVALIDCMD	Invalid mailbox command.
1795 (0x703)	ECERR_DEVICE_INVALIDINDEX	An invalid value for the Index input was specified.
1796 (0x704)	ECERR_DEVICE_INVALIDACCESS	Reading of the variable is not permitted.
1797 (0x705)	ECERR_DEVICE_INVALIDSIZE	An invalid size for the parameter was specified.
1798 (0x706)	ECERR_DEVICE_INVALIDDATA	Invalid parameter value(s) in SDO index and/or sub-index.
1799 (0x707)	ECERR_DEVICE_NOTREADY	Device is not in a ready state. The network is not in operational.
1800 (0x708)	ECERR_DEVICE_BUSY	Device is not available to respond.
1801 (0x709)	ECERR_DEVICE_INVALIDCONTEXT	Device responded with an illegal error code, indicating the command is not allowed under the present conditions.
1802 (0x70A)	ECERR_DEVICE_NOMEMORY	EtherCAT mailbox is out of memory or device is out of disk space.

Value Dec (hex)	Error Code	Description
1803 (0x70B)	ECERR_DEVICE_INVALIDPARAM	EtherCAT mailbox request was not valid.
1804 (0x70C)	ECERR_DEVICE_NOTFOUND	EtherCAT device not found.
1805 (0x70D)	ECERR_DEVICE_SYNTAX	An unexpected error occurred.
1810 (0x712)	ECERR_DEVICE_INVALIDSTATE	The EtherCAT device is in an invalid state.
1817 (0x719)	ECERR_DEVICE_TIMEOUT	The EtherCAT device failed to respond, timing out.
1826 (0x722)	ECERR_DEVICE_INSERTMAILBOX	Error while inserting the mailbox command into internal FIFO.
1827 (0x723)	ECERR_DEVICE_INVALIDOFFSET	An invalid value for the SubIndex input was specified.
1828 (0x724)	ECERR_DEVICE_UNKNOWNMAILBOXCMD	The master sent an unknown mailbox command to the slave.
1829 (0x725)	ECERR_DEVICE_ACCESSDENIED	Device responded with an invalid access error code, indicating the command is not allowed.
1832 (0x728)	ECERR_DEVICE_INVALIDADDR	The specified EtherCAT node address is invalid.
1836 (0x72c)	ECERR_DEVICE_NOT_A_FSOE_MASTER	Device is not a FSoE master.
1837 (0x072D)	ECERR_DEVICE_DISCONNECTED	The EtherCAT device is disconnected.
1920 (0x780)	ECERR_DEVICE_PARAM_ACCESS_ERROR	Unknown error occurred while accessing parameter.
1921 (0x781)	ECERR_DEVICE_PARAM_NOT_FOUND	Parameter was not found.
1922 (0x782)	ECERR_DEVICE_PARAM_NOT_INTEGER	Parameter is a floating-point value. Integer value required.
1923 (0x783)	ECERR_DEVICE_VALUE_IS_NEGATIVE	No negative values allowed. Value specified was negative.
1924 (0x784)	ECERR_DEVICE_VALUE_OUT_OF_RANGE	Value is out of data-range.
1925 (0x785)	ECERR_DEVICE_VALUE_GREATER_THAN_MAX	Value bigger than maximum.
1926 (0x786)	ECERR_DEVICE_VALUE_LOWER_THAN_MIN	Value lower than minimum.
2048 (0x800)	ECERR_CLIENT_ERROR	Error in Mailbox response to a previously sent mailbox command.
2049 (0x801)	ECERR_CLIENT_TIMEOUT	The SDO command timed out.
2050 (0x802)	ECERR_CLIENT_INVALIDPARAM	An invalid value was specified.
2051 (0x803)	ECERR_CLIENT_INVALIDSIZE	An invalid value for the size input was specified.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(*****)
(* Read AKD_1 device state and link status*)
(*****)

Inst_EcDeviceStatus(TRUE, EtherCAT.AKD_1);
```

See Also

- "ECATMasterStatus" (→ p. 59)
- "ECATWCStatus" (→ p. 80)

1.2.9 ECATDevReadParam

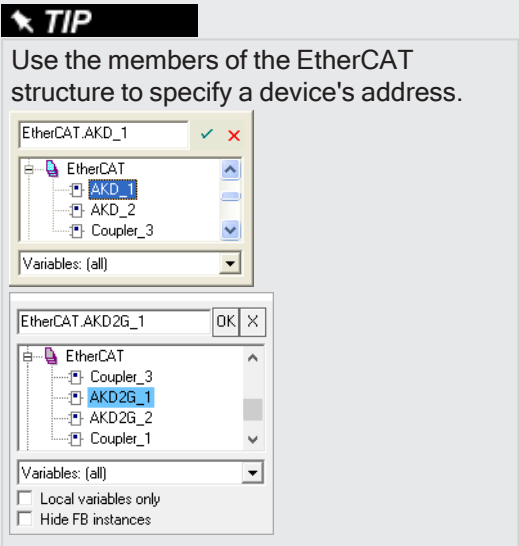
PLCopen ✓

Pipe Network ✓

 **Function Block** - Returns the EtherCAT device-specific information.

See "EtherCAT Function Blocks that Work with Drive Parameters" (→ p. 21) about function blocks not supported by ML and MC function blocks.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	Requests to read the EtherCAT device-specific parameter.
DeviceAddress	INT	-32768 to +32767	N/A	No default	The address of the device data is read from. • The first node usually has the value 1001. • The second node usually has the value 1002. 
ParameterNumber	INT	-32768 to +32767	N/A	No default	Parameter number. See "EtherCAT Device Parameters" (→ p. 55).

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified.

Output	Data Type	Range	Unit	Description
ErrorID	INT	-32768 to +32767	N/A	Indicates the error if Error output is TRUE. See "Error Codes" (→ p. 53).
Value	LREAL	No range	N/A	Value of the parameter.

Error Codes

Value Dec (hex)	Error Code	Description
0	ECERR_OK	The SDO call succeeded.
1000 (0x3E8)	ECERR_DEVICE_ACTION_IS_INVALID	The requested Device Action is invalid.
1003 (0x3EB)	ECERR_DEVICE_ACTION_MINIMUM_ONE_NODE_REQD	A minimum of one device must be present in the EtherCAT network.
1004 (0x3EC)	ECERR_DEVICE_ACTION_MINIMUM_ONE_DC_NODE_REQD	A minimum of one Distributed clock capable device must be present in the EtherCAT network.
107 = 0x6B	ECERR_INVALID_ARRAY_SIZE	The size of the connections is too small. - When the array size is smaller than the number of bad connections: - The array is filled with the data to the size of the array. - The error ECERR_INVALID_ARRAY_SIZE is set. - In this scenario, the output ConnectionCount is set to the size of the array and it is smaller than the number of actual bad connections.
1792 (0x700)	ECERR_DEVICE_ERROR	EtherCAT device is not accessible.
1794 (0x702)	ECERR_DEVICE_INVALIDCMD	Invalid mailbox command.
1795 (0x703)	ECERR_DEVICE_INVALIDINDEX	An invalid value for the Index input was specified.
1796 (0x704)	ECERR_DEVICE_INVALIDACCESS	Reading of the variable is not permitted.
1797 (0x705)	ECERR_DEVICE_INVALIDSIZE	An invalid size for the parameter was specified.
1798 (0x706)	ECERR_DEVICE_INVALIDDATA	Invalid parameter value(s) in SDO index and/or sub-index.
1799 (0x707)	ECERR_DEVICE_NOTREADY	Device is not in a ready state. The network is not in operational.
1800 (0x708)	ECERR_DEVICE_BUSY	Device is not available to respond.
1801 (0x709)	ECERR_DEVICE_INVALIDCONTEXT	Device responded with an illegal error code, indicating the command is not allowed under the present conditions.

Value Dec (hex)	Error Code	Description
1802 (0x70A)	ECERR_DEVICE_NOMEMORY	EtherCAT mailbox is out of memory or device is out of disk space.
1803 (0x70B)	ECERR_DEVICE_INVALIDPARM	EtherCAT mailbox request was not valid.
1804 (0x70C)	ECERR_DEVICE_NOTFOUND	EtherCAT device not found.
1805 (0x70D)	ECERR_DEVICE_SYNTAX	An unexpected error occurred.
1810 (0x712)	ECERR_DEVICE_INVALIDSTATE	The EtherCAT device is in an invalid state.
1817 (0x719)	ECERR_DEVICE_TIMEOUT	The EtherCAT device failed to respond, timing out.
1826 (0x722)	ECERR_DEVICE_INSERTMAILBOX	Error while inserting the mailbox command into internal FIFO.
1827 (0x723)	ECERR_DEVICE_INVALIDOFFSET	An invalid value for the SubIndex input was specified.
1828 (0x724)	ECERR_DEVICE_UNKNOWNMAILBOXCMD	The master sent an unknown mailbox command to the slave.
1829 (0x725)	ECERR_DEVICE_ACCESSDENIED	Device responded with an invalid access error code, indicating the command is not allowed.
1832 (0x728)	ECERR_DEVICE_INVALIDADDR	The specified EtherCAT node address is invalid.
1836 (0x72c)	ECERR_DEVICE_NOT_A_FSOE_MASTER	Device is not a FSoE master.
1837 (0x072D)	ECERR_DEVICE_DISCONNECTED	The EtherCAT device is disconnected.
1920 (0x780)	ECERR_DEVICE_PARAM_ACCESS_ERROR	Unknown error occurred while accessing parameter.
1921 (0x781)	ECERR_DEVICE_PARAM_NOT_FOUND	Parameter was not found.
1922 (0x782)	ECERR_DEVICE_PARAM_NOT_INTEGER	Parameter is a floating-point value. Integer value required.
1923 (0x783)	ECERR_DEVICE_VALUE_IS_NEGATIVE	No negative values allowed. Value specified was negative.
1924 (0x784)	ECERR_DEVICE_VALUE_OUT_OF_RANGE	Value is out of data-range.
1925 (0x785)	ECERR_DEVICE_VALUE_GREATER_THAN_MAX	Value bigger than maximum.
1926 (0x786)	ECERR_DEVICE_VALUE_LOWER_THAN_MIN	Value lower than minimum.
2048 (0x800)	ECERR_CLIENT_ERROR	Error in Mailbox response to a previously sent mailbox command.
2049 (0x801)	ECERR_CLIENT_TIMEOUT	The SDO command timed out.
2050 (0x802)	ECERR_CLIENT_INVALIDPARM	An invalid value was specified.

Value Dec (hex)	Error Code	Description
2051 (0x803)	ECERR_CLIENT_INVALIDSIZE	An invalid value for the size input was specified.

1.2.9.0.1 EtherCAT Device Parameters

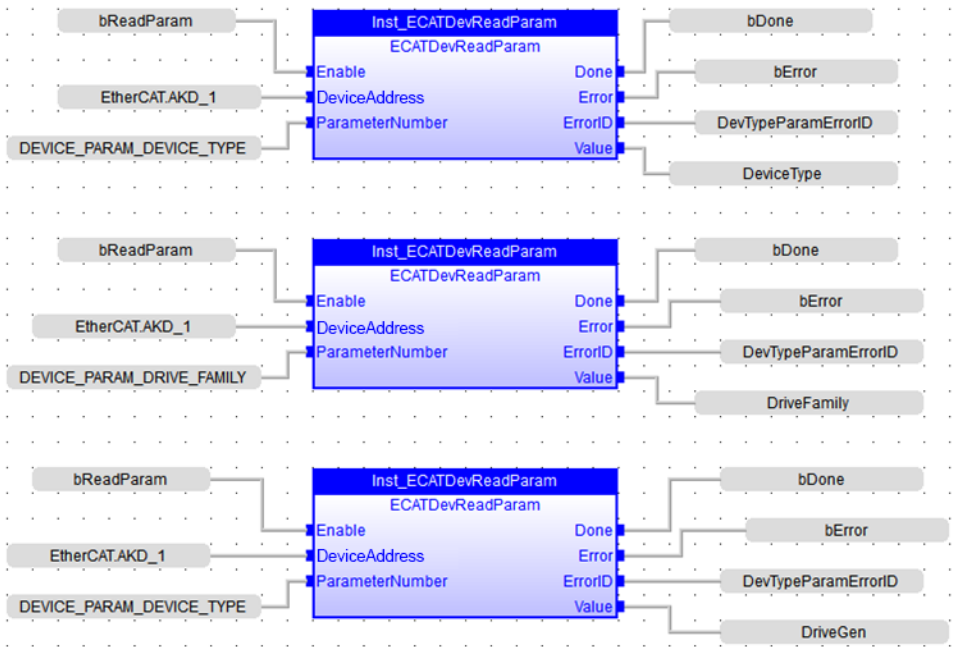
This is a list of supported parameters read by **ECATDevReadParam**.

[Click to view the EtherCAT Device Parameters](#)

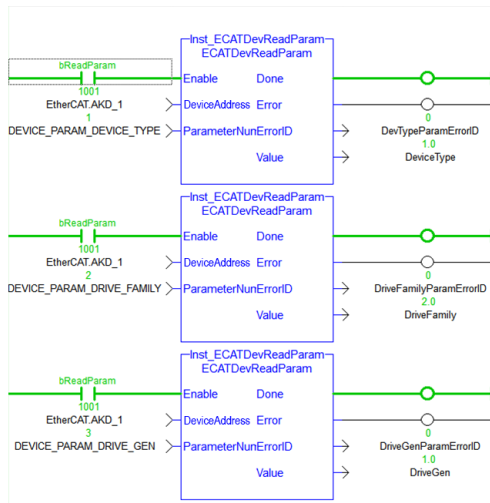
Parameter	ID	Name	R/W	Description		
DEVICE_PARAM_DEVICE_TYPE	1	Device Type	Read Only	EtherCAT Device Type		
				Output Value	Numerical Value	Description
				DEVICE_TYPE_OTHER	0	The EtherCAT device is not a drive.
				DEVICE_TYPE_DRIVE	1	The EtherCAT device is a drive.
DEVICE_PARAM_DRIVE_FAMILY	2	Drive Family	Read Only	EtherCAT device drive family details.		
				Output Value	Numerical Value	Description
				DEVICE_NOT_A_DRIVE	-1	The EtherCAT device is not a drive.
				DRIVE_FAMILY_OTHER	0	The drive family cannot be determined.
				DRIVE_FAMILY_S300_S700	1	The device is an S300/S700 family drive.
				DRIVE_FAMILY_AKD	2	The device is in the AKD2G, AKD-N, or AKD servo drive family.
				DRIVE_FAMILY_AKT2G_STEPPER	3	The device is an AKT2G Stepper family drive.

Parameter	ID	Name	R/W	Description		
DEVICE_PARAM_DRIVE_GEN	3	Drive Generation	Read Only	EtherCAT drive generation details.		
				Output Value	Numerical Value	Description
				DEVICE_NOT_A_DRIVE	-1	The EtherCAT device is not a drive.
				DRIVE_GEN_UNKNOWN	0	The drive generation cannot be determined.
				DRIVE_GEN_1	1	The device is a 1 st generation drive (e.g., AKD, AKD-N, S300, S700, AKT2G-SM-L15, AKT2G-SM-L50).
DRIVE_GEN_2	2	The device is a 2 nd generation drive (e.g., AKD2G).				

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* ECATDevReadParam ST example *)
Inst_ECATDevReadParam( TRUE, EtherCAT.AKD_2, DEVICE_PARAM_DEVICE_TYPE );
DeviceType := Inst_ECATDevReadParam.Value;

Inst_ECATDevReadParam( TRUE, EtherCAT.AKD_2, DEVICE_PARAM_DRIVE_FAMILY );
DeviceFamily := Inst_ECATDevReadParam.Value;

Inst_ECATDevReadParam( TRUE, EtherCAT.AKD_2, DEVICE_PARAM_DRIVE_GEN );
DriveGen := Inst_ECATDevReadParam.Value;
```


1.2.10 ECATGetObjVal - Deprecated

PLCopen ✓

Pipe Network ✓

NOTE



Function - Deprecated as of KAS v2.7.
The recommended best practice is to map a PLC variable to a PDO object.

1.2.11 ECATMasterStatus

PLCopen ✓

Pipe Network ✓



Function Block - Reads the EtherCAT master state and the lost frame counter to determine if EtherCAT is running normally.

TIP

See EtherCAT Communication Diagnosis Steps.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	- Request to read the EtherCAT master state and the lost frame count. - Continuously reads the master state and the lost frame count as long as the Enable remains high.

Outputs

Output	Data Type	Range	Unit	Description
Valid	BOOL	FALSE, TRUE	N/A	Indicates the values at the State and LostFrameCount outputs are valid.
Error	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed with error.
ErrorID	DINT	No range	N/A	Error code when the function block failed due to error.
State	UINT	No range	N/A	- Indicates the EtherCAT state of the Master. - See "State Defines" (→ p. 59) for more information.
LostFrameCount	UDINT	No range	N/A	Total cumulative number of cyclic frames sent with no-response since the EtherCAT started by calling the MLMotionStart. - Missing return frames will generate an A38 alarm. - The Counter is reset to 0 (zero) when the MLMotionStart is called.

1.2.11.0.1 State Defines

```

#define EC_STATE_NO_COMMUNICATION 0 (* 0x00 = No Communication to
device *)
#define EC_STATE_INIT 1 (* 0x01 = Device in Init state *)
#define EC_STATE_PREOP 2 (* 0x02 = Device in Pre-operational
state *)
#define EC_STATE_BOOTSTRAP 3 (* 0x03 = Device in Bootstrap state
*)
#define EC_STATE_SAFEOP 4 (* 0x04 = Device in Safe-
```

```
Operational state *)
#define    EC_STATE_OP                8    (* 0x08 = Device in Operational
state *)
```

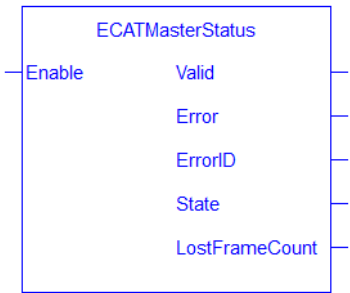
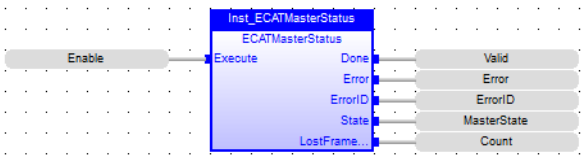
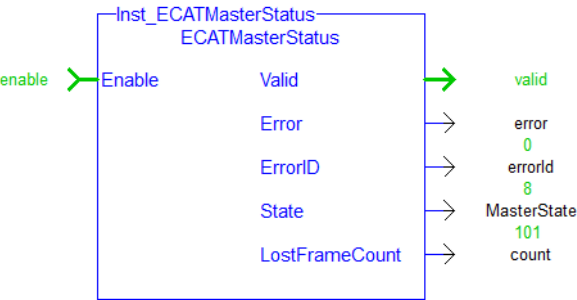


Figure 2-1: ECATMasterStatus

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
// ECATMasterStatus
Inst_ECATMasterStatus( True );
MasterState := Inst_ECATMasterStatus.State;
MasterLastFrameCount := Inst_ECATMasterStatus.LostFrameCount;
```

See Also

- "ECATDeviceStatus" (→ p. 47)
- "ECATWCStatus" (→ p. 80)

1.2.12 ECATReadData

PLCopen ✓

Pipe Network ✓



Function - Reads cyclic parameter (byte offset format).

ⓘ IMPORTANT

This is a low level function and should be used carefully by **advanced users**.

NOTE

- The valid ranges for the **Value** parameter are:
 - For 1 byte: 0 to 255
 - For 2 bytes: 0 to 65535
 - For 4 bytes: -2147483648 to 2147483647
(The sign bit represents the most significant bit in the data word).
- Allows a direct access to the memory EtherCAT Process Image of the EtherCAT frame which is sent or received when you need to debug the application.
- Access the EtherCAT image element by giving the offset in the image and the size of the element.
- If you have a device other than the drive, this function is used for more than just debug.
 - It is used to get the status of the module (e.g., Stepper I/O slice) in the case the project is based on an external XML file because it contains unsupported EtherCAT device.
 - See Add an Unsupported EtherCAT Device.

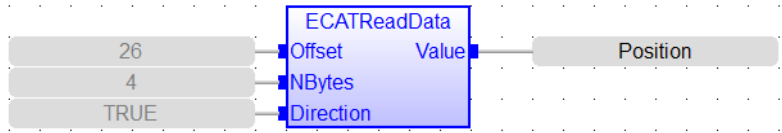
Inputs

Input	Data Type	Range	Unit	Default	Description
Offset	UINT	0 to 1500	Bytes	No default	Offset in bytes from the beginning of the frame. The maximum size of an Ethernet frame is 1500.
ⓘ IMPORTANT - The Offset value required to access may change when: - The firmware for any device on the EtherCAT network is updated or Whenever the EtherCAT network topology changes. - When performing an update of a network device or changing the network topology, export the ENI file and check the Offset value needed to access the desired information.					
Nbytes	SINT	1, 2, or 4	Bytes	No default	Number of bytes to read.
Direction	BOOL	FALSE, TRUE	N/A	No default	Direction of the frame. - TRUE = output image. - FALSE = input image.

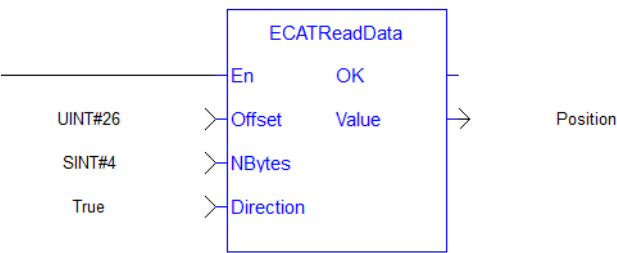
Outputs

Output	Data Type	Range	Unit	Description
Value	DINT	No range	N/A	Value of the EtherCAT frame.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
// Read 4 bytes starting at offset 26 of the output image
Position := ECATReadData(26, 4, true);
```

See Also

"ECATWriteData" (→ p. 87)

1.2.13 ECATReadPDOData

PLCopen ✓

Pipe Network ✓

This function/function block was added in KAS v4.05.

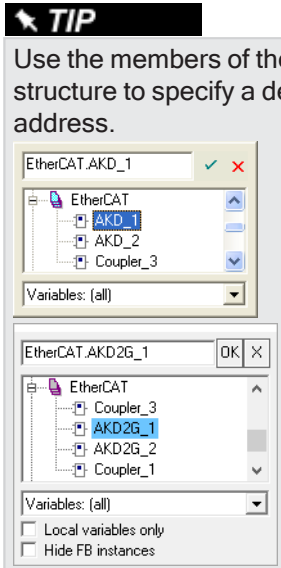


Function Block - Copies the Input (Tx) PDO object data from an EtherCAT cyclic frame to a byte array.

- When the Enable input is TRUE, then on every cycle, the Input (Tx) PDO data is copied to the **CoEDataArray**.
- The EtherCAT Device and the EtherCAT Master should be in Operational State (EC_STATE_OP).
- Serializeln can be used to extract the value from the **CoEDataArray**.
 - Endianness: EtherCAT transmits the data in little endian format.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	When Enable is TRUE, bytes are copied from the Input (Tx) PDO object to CoEDataArray . • The PDO object is specified by its CoE Index and SubIndex.
Index	UINT	0 to 65535	N/A	No default	The CoE index of the source Input (Tx) PDO object.
SubIndex	USINT	0 to 255	N/A	No default	The CoE sub-index of the source Input (Tx) PDO object.
Size	USINT	1 to 255	BYTE	No default	• The size of the CoE object to read. • The size should be the exact size of the CoE object.

Input	Data Type	Range	Unit	Default	Description
DeviceAddress	INT	0 to 65535	N/A	No default	The EtherCAT address of the device from which data is read from. - The first node usually has the value 1001. - The second node usually has the value 1002.  TIP Use the members of the EtherCAT structure to specify a device's address.
CoEdataArray[]	USINT[]	- Dimension: $\geq$ Size. - Elements: No range	N/A	No default	Input (Tx) PDO data is copied to this array.

Outputs

Output	Data Type	Range	Unit	Description
Valid	BOOL	FALSE, TRUE	N/A	If TRUE, the data returned is valid.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	DINT	No range	N/A	Indicates the error if Error output is TRUE. - See "Error Codes" (→ p. 64). - Upon success, Error is set to 0 (zero).

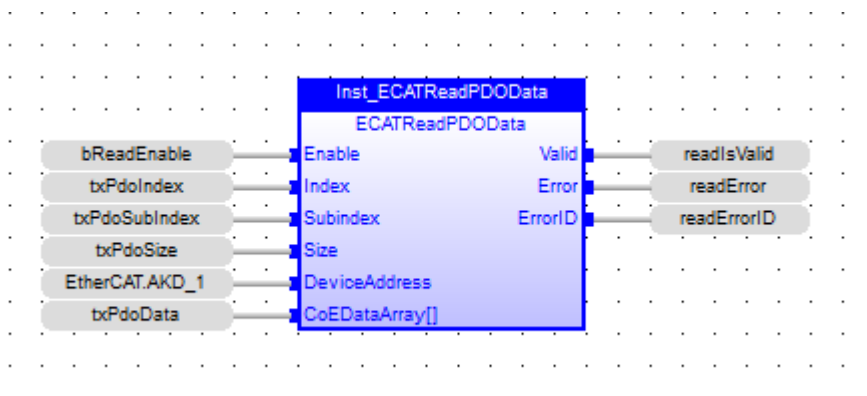
1.2.13.0.1 Error Codes

Value Dec (hex)	Error Code	Description
0	ECERR_OK	The SDO call succeeded.
1000 (0x3E8)	ECERR_DEVICE_ACTION_IS_INVALID	The requested Device Action is invalid.
1003 (0x3EB)	ECERR_DEVICE_ACTION_MINIMUM_ONE_NODE_REQD	A minimum of one device must be present in the EtherCAT network.

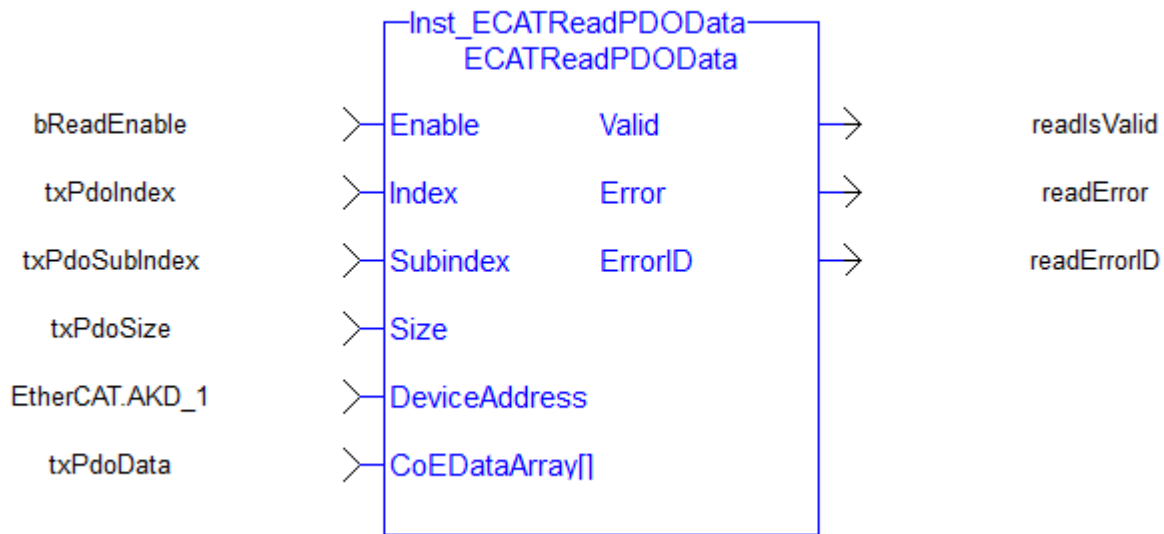
Value Dec (hex)	Error Code	Description
1004 (0x3EC)	ECERR_DEVICE_ACTION_MINIMUM_ONE_DC_NODE_REQD	A minimum of one Distributed clock capable device must be present in the EtherCAT network.
107 = 0x6B	ECERR_INVALID_ARRAY_SIZE	The size of the connections is too small. - When the array size is smaller than the number of bad connections: - The array is filled with the data to the size of the array. - The error ECERR_INVALID_ARRAY_SIZE is set. - In this scenario, the output ConnectionCount is set to the size of the array and it is smaller than the number of actual bad connections.
1792 (0x700)	ECERR_DEVICE_ERROR	EtherCAT device is not accessible.
1794 (0x702)	ECERR_DEVICE_INVALIDCMD	Invalid mailbox command.
1795 (0x703)	ECERR_DEVICE_INVALIDINDEX	An invalid value for the Index input was specified.
1796 (0x704)	ECERR_DEVICE_INVALIDACCESS	Reading of the variable is not permitted.
1797 (0x705)	ECERR_DEVICE_INVALIDSIZE	An invalid size for the parameter was specified.
1798 (0x706)	ECERR_DEVICE_INVALIDDATA	Invalid parameter value(s) in SDO index and/or sub-index.
1799 (0x707)	ECERR_DEVICE_NOTREADY	Device is not in a ready state. The network is not in operational.
1800 (0x708)	ECERR_DEVICE_BUSY	Device is not available to respond.
1801 (0x709)	ECERR_DEVICE_INVALIDCONTEXT	Device responded with an illegal error code, indicating the command is not allowed under the present conditions.
1802 (0x70A)	ECERR_DEVICE_NOMEMORY	EtherCAT mailbox is out of memory or device is out of disk space.
1803 (0x70B)	ECERR_DEVICE_INVALIDPARM	EtherCAT mailbox request was not valid.
1804 (0x70C)	ECERR_DEVICE_NOTFOUND	EtherCAT device not found.
1805 (0x70D)	ECERR_DEVICE_SYNTAX	An unexpected error occurred.
1810 (0x712)	ECERR_DEVICE_INVALIDSTATE	The EtherCAT device is in an invalid state.
1817 (0x719)	ECERR_DEVICE_TIMEOUT	The EtherCAT device failed to respond, timing out.
1826 (0x722)	ECERR_DEVICE_INSERTMAILBOX	Error while inserting the mailbox command into internal FIFO.
1827 (0x723)	ECERR_DEVICE_INVALIDOFFSET	An invalid value for the SubIndex input was specified.

Value Dec (hex)	Error Code	Description
1828 (0x724)	ECERR_DEVICE_UNKNOWNMAILBOXCMD	The master sent an unknown mailbox command to the slave.
1829 (0x725)	ECERR_DEVICE_ACCESSDENIED	Device responded with an invalid access error code, indicating the command is not allowed.
1832 (0x728)	ECERR_DEVICE_INVALIDADDR	The specified EtherCAT node address is invalid.
1836 (0x72c)	ECERR_DEVICE_NOT_A_FSOE_MASTER	Device is not a FSoE master.
1837 (0x072D)	ECERR_DEVICE_DISCONNECTED	The EtherCAT device is disconnected.
1920 (0x780)	ECERR_DEVICE_PARAM_ACCESS_ERROR	Unknown error occurred while accessing parameter.
1921 (0x781)	ECERR_DEVICE_PARAM_NOT_FOUND	Parameter was not found.
1922 (0x782)	ECERR_DEVICE_PARAM_NOT_INTEGER	Parameter is a floating-point value. Integer value required.
1923 (0x783)	ECERR_DEVICE_VALUE_IS_NEGATIVE	No negative values allowed. Value specified was negative.
1924 (0x784)	ECERR_DEVICE_VALUE_OUT_OF_RANGE	Value is out of data-range.
1925 (0x785)	ECERR_DEVICE_VALUE_GREATER_THAN_MAX	Value bigger than maximum.
1926 (0x786)	ECERR_DEVICE_VALUE_LOWER_THAN_MIN	Value lower than minimum.
2048 (0x800)	ECERR_CLIENT_ERROR	Error in Mailbox response to a previously sent mailbox command.
2049 (0x801)	ECERR_CLIENT_TIMEOUT	The SDO command timed out.
2050 (0x802)	ECERR_CLIENT_INVALIDPARM	An invalid value was specified.
2051 (0x803)	ECERR_CLIENT_INVALIDSIZE	An invalid value for the size input was specified.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_ECATReadPDOData1( TRUE, 16#6063, 0, 4, EtherCAT.AKD_1, txPdoData );
IF Inst_ECATReadPDOData1.Valid THEN
    // Data is valid and can be used.
    // Extract the value from the byte array. Data is in little endian format.
    SerializeIn( txPdoData, txObjValue1, 0, FALSE);
END_IF;
```

See Also

- "ECATReadSdo" (→ p. 68)
- "ECATWritePDOData" (→ p. 82)
- `SerializeIn`

1.2.14 ECATReadSdo

PLCopen 

Pipe Network 



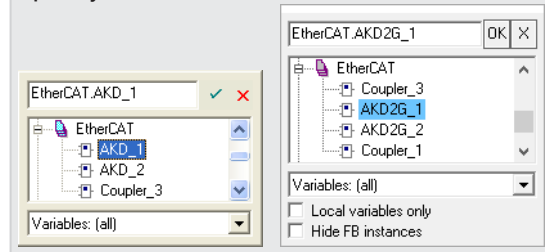
Function Block - Reads a 32-bit word from I/O nodes using a CANopen SDO read command.

Notes

- This function block **uses and reserves** the EtherCAT SDO Channel.
- The SDO Channel remains reserved until the done output is TRUE.
- This FB should be called at each cycle until either the Done or Error output is TRUE.
 - If it is not called at each cycle, the rest of SDO communication (e.g., the AKD GUI Views) is blocked.
- Using this FB in SFC P0 or P1 steps is not recommended because these steps are executed only once.
- If this FB is used in P0 or P1, then it must be used in an SFC N step to ensure the FB completes.
- Is typically used to query the status of inputs.
- See "EtherCAT Function Blocks that Work with SDOs" (→ p. 23) about function blocks used to work with drive or remote I/O parameters that are not supported by ML and MC function blocks.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge of Execute, an SDO read command is issued. • The function block only handles one SDO command at a time. • If Execute is toggled quickly so that another rising edge occurs before the SDO command has completed, the function block does not issue a second SDO command.
Index	INT	No range	N/A	No default	The object directory index of the data to be read. See: • Communication SDOs. • Manufacturer specific SDOs. • Profile specific SDOs. To read/write an SDO object with an index greater than 16#7FFF (32767), the value must be entered in this form: <code>any_to_int(index # in hex format).</code> Example: <code>any_to_int(16#8321).</code>

Input	Data Type	Range	Unit	Default	Description
SubIndex	SINT	No range	N/A	No default	The sub-index of the object directory variable to be read. See: • Communication SDOs. • Manufacturer specific SDOs. • Profile specific SDOs. To read/write an SDO object with an index greater than 16#7FFF (32767), the value must be entered in this form: <code>any_to_int</code>(index # in hex format). Example: <code>any_to_int</code>(16#8321).
Size	SINT	1 to 4	N/A	No default	The size (i.e., number of bytes) to read.
DeviceAddress	INT	-32768 to +32767	N/A	No default	The EtherCAT address of the device from which data is read from. • The first node usually has the value 1001. • The second node usually has the value 1002. TIP Use the members of the EtherCAT structure to specify a device's address. 

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.
Error	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed with error.
ErrorID	DINT	No range	N/A	The SDO call error result, if Error is TRUE. • See "Error Codes" (→ p. 70). • Upon success, Error is set to 0 (zero).
Value	DINT	No range	N/A	• The value of the object directory variable being read. • Value is only set when an SDO read command has successfully completed.

State Diagram

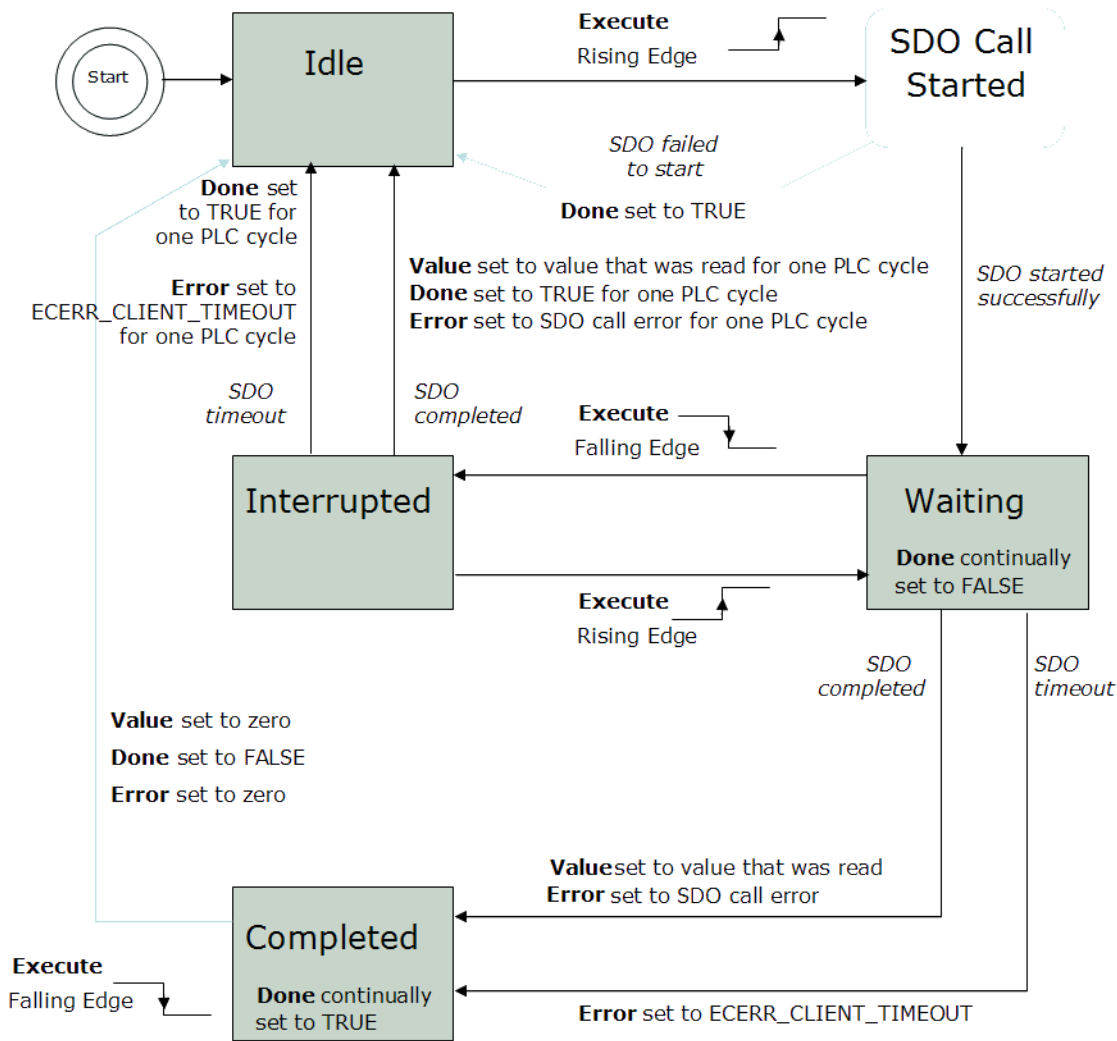


Figure 2-2: ECATReadSDO State Diagram

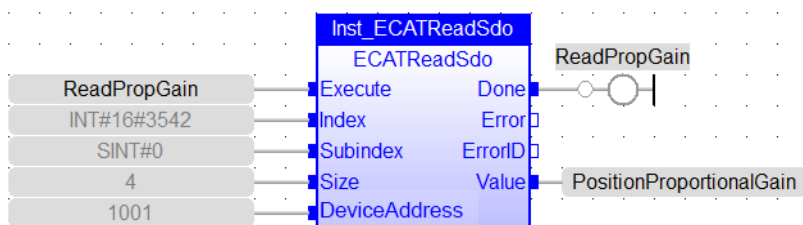
Error Codes

Value Dec (hex)	Error Code	Description
0	ECERR_OK	The SDO call succeeded.
1000 (0x3E8)	ECERR_DEVICE_ACTION_IS_INVALID	The requested Device Action is invalid.
1003 (0x3EB)	ECERR_DEVICE_ACTION_MINIMUM_ONE_NODE_REQD	A minimum of one device must be present in the EtherCAT network.
1004 (0x3EC)	ECERR_DEVICE_ACTION_MINIMUM_ONE_DC_NODE_REQD	A minimum of one Distributed clock capable device must be present in the EtherCAT network.

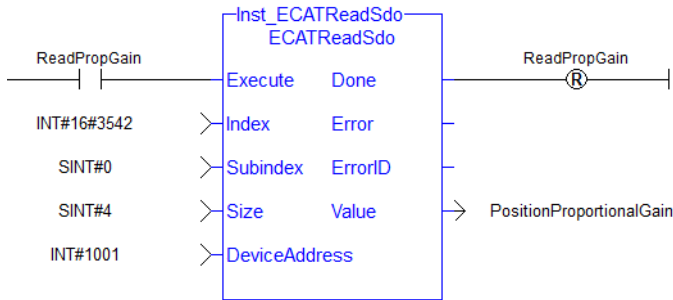
Value Dec (hex)	Error Code	Description
107 = 0x6B	ECERR_INVALID_ARRAY_SIZE	The size of the connections is too small. - When the array size is smaller than the number of bad connections: - The array is filled with the data to the size of the array. - The error ECERR_INVALID_ARRAY_SIZE is set. - In this scenario, the output ConnectionCount is set to the size of the array and it is smaller than the number of actual bad connections.
1792 (0x700)	ECERR_DEVICE_ERROR	EtherCAT device is not accessible.
1794 (0x702)	ECERR_DEVICE_INVALIDCMD	Invalid mailbox command.
1795 (0x703)	ECERR_DEVICE_INVALIDINDEX	An invalid value for the Index input was specified.
1796 (0x704)	ECERR_DEVICE_INVALIDACCESS	Reading of the variable is not permitted.
1797 (0x705)	ECERR_DEVICE_INVALIDSIZE	An invalid size for the parameter was specified.
1798 (0x706)	ECERR_DEVICE_INVALIDDATA	Invalid parameter value(s) in SDO index and/or sub-index.
1799 (0x707)	ECERR_DEVICE_NOTREADY	Device is not in a ready state. The network is not in operational.
1800 (0x708)	ECERR_DEVICE_BUSY	Device is not available to respond.
1801 (0x709)	ECERR_DEVICE_INVALIDCONTEXT	Device responded with an illegal error code, indicating the command is not allowed under the present conditions.
1802 (0x70A)	ECERR_DEVICE_NOMEMORY	EtherCAT mailbox is out of memory or device is out of disk space.
1803 (0x70B)	ECERR_DEVICE_INVALIDPARM	EtherCAT mailbox request was not valid.
1804 (0x70C)	ECERR_DEVICE_NOTFOUND	EtherCAT device not found.
1805 (0x70D)	ECERR_DEVICE_SYNTAX	An unexpected error occurred.
1810 (0x712)	ECERR_DEVICE_INVALIDSTATE	The EtherCAT device is in an invalid state.
1817 (0x719)	ECERR_DEVICE_TIMEOUT	The EtherCAT device failed to respond, timing out.
1826 (0x722)	ECERR_DEVICE_INSERTMAILBOX	Error while inserting the mailbox command into internal FIFO.
1827 (0x723)	ECERR_DEVICE_INVALIDOFFSET	An invalid value for the SubIndex input was specified.
1828 (0x724)	ECERR_DEVICE_UNKNOWNMAILBOXCMD	The master sent an unknown mailbox command to the slave.

Value Dec (hex)	Error Code	Description
1829 (0x725)	ECERR_DEVICE_ACCESSDENIED	Device responded with an invalid access error code, indicating the command is not allowed.
1832 (0x728)	ECERR_DEVICE_INVALIDADDR	The specified EtherCAT node address is invalid.
1836 (0x72c)	ECERR_DEVICE_NOT_A_FSOE_MASTER	Device is not a FSoE master.
1837 (0x072D)	ECERR_DEVICE_DISCONNECTED	The EtherCAT device is disconnected.
1920 (0x780)	ECERR_DEVICE_PARAM_ACCESS_ERROR	Unknown error occurred while accessing parameter.
1921 (0x781)	ECERR_DEVICE_PARAM_NOT_FOUND	Parameter was not found.
1922 (0x782)	ECERR_DEVICE_PARAM_NOT_INTEGER	Parameter is a floating-point value. Integer value required.
1923 (0x783)	ECERR_DEVICE_VALUE_IS_NEGATIVE	No negative values allowed. Value specified was negative.
1924 (0x784)	ECERR_DEVICE_VALUE_OUT_OF_RANGE	Value is out of data-range.
1925 (0x785)	ECERR_DEVICE_VALUE_GREATER_THAN_MAX	Value bigger than maximum.
1926 (0x786)	ECERR_DEVICE_VALUE_LOWER_THAN_MIN	Value lower than minimum.
2048 (0x800)	ECERR_CLIENT_ERROR	Error in Mailbox response to a previously sent mailbox command.
2049 (0x801)	ECERR_CLIENT_TIMEOUT	The SDO command timed out.
2050 (0x802)	ECERR_CLIENT_INVALIDPARM	An invalid value was specified.
2051 (0x803)	ECERR_CLIENT_INVALIDSIZE	An invalid value for the size input was specified.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* Read PL.KP on first AKD Drive on EtherCAT network *)
Inst_ECATReadSdo(TRUE, 16#3542, 0, 4, 1001 );
IF Inst_ECATReadSdo.Done OR Inst_ECATReadSdo.Error THEN
    Inst_ECATReadSdo(FALSE, 16#3542, 0, 4, 1001 );
    PositionProportionalGain := Inst_ECATReadSdo.Value;
END_IF;
```

See Also

- "ECATReadSdoData" (→ p. 74)
- "ECATWriteSdo" (→ p. 89)

1.2.15 ECATReadSdoData

PLCopen 

Pipe Network 



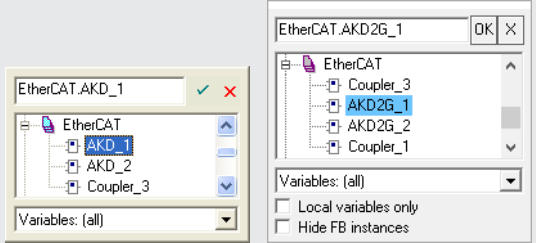
Function Block - Reads data from an EtherCAT device using CANopen SDO read command.

Notes

- This function block **uses and reserves** the EtherCAT SDO Channel.
- The SDO Channel remains reserved until the done output is TRUE.
- This FB should be called at each cycle until either the Done or Error output is TRUE.
 - If it is not called at each cycle, the rest of SDO communication (e.g., the AKD GUI Views) is blocked.
- Using this FB in SFC P0 or P1 steps is not recommended because these steps are executed only once.
- If this FB is used in P0 or P1, then it must be used in an SFC N step to ensure the FB completes.
- Is typically used to query the status of inputs.
- See "EtherCAT Function Blocks that Work with SDOs" (→ p. 23) about function blocks used to work with drive or remote I/O parameters that are not supported by ML and MC function blocks.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge of Execute, an SDO read command is issued. • The function block only handles one SDO command at a time. • If Execute is toggled quickly so that another rising edge occurs before the SDO command has completed, the function block does not issue a second SDO command.
Index	INT	-32768 to +32767	N/A	No default	The object directory index of the data to be read. See: • Communication SDOs. • Manufacturer specific SDOs. • Profile specific SDOs. To read/write an SDO object with an index greater than 16#7FFF (32767), the value must be entered in this form: <code>any_to_int(index # in hex format)</code>. Example: <code>any_to_int(16#8321)</code>.

Input	Data Type	Range	Unit	Default	Description
SubIndex	SINT	No range	N/A	No default	The sub-index of the object directory variable to be read. See: • Communication SDOs. • Manufacturer specific SDOs. • Profile specific SDOs. To read/write an SDO object with an index greater than 16#7FFF (32767), the value must be entered in this form: <code>any_to_int(index # in hex format)</code>. Example: <code>any_to_int(16#8321)</code>.
Size	UINT	1 to 255	N/A	No default	• The size (i.e., number of bytes) of the CoE object to read. • The size should be the exact size of the CoE object not exceeding 255 bytes.
DeviceAddress	INT	-32768 to +32767	N/A	No default	The EtherCAT address of the device from which data is read from. • The first node usually has the value 1001. • The second node usually has the value 1002. TIP Use the members of the EtherCAT structure to specify a device's address. 
CoEDataArray[]	USINT	1 to 255	N/A	No default	• The CoE object data being read. • The size of the array should be equal to or greater than the specified size. • The data is only set when an SDO read command has successfully completed.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.
Error	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed with error.

Output	Data Type	Range	Unit	Description
ErrorID	DINT	No range	N/A	The SDO call error result, if Error is TRUE. - See "Error Codes" (→ p. 76). - Upon success, Error is set to 0 (zero).

State Diagram

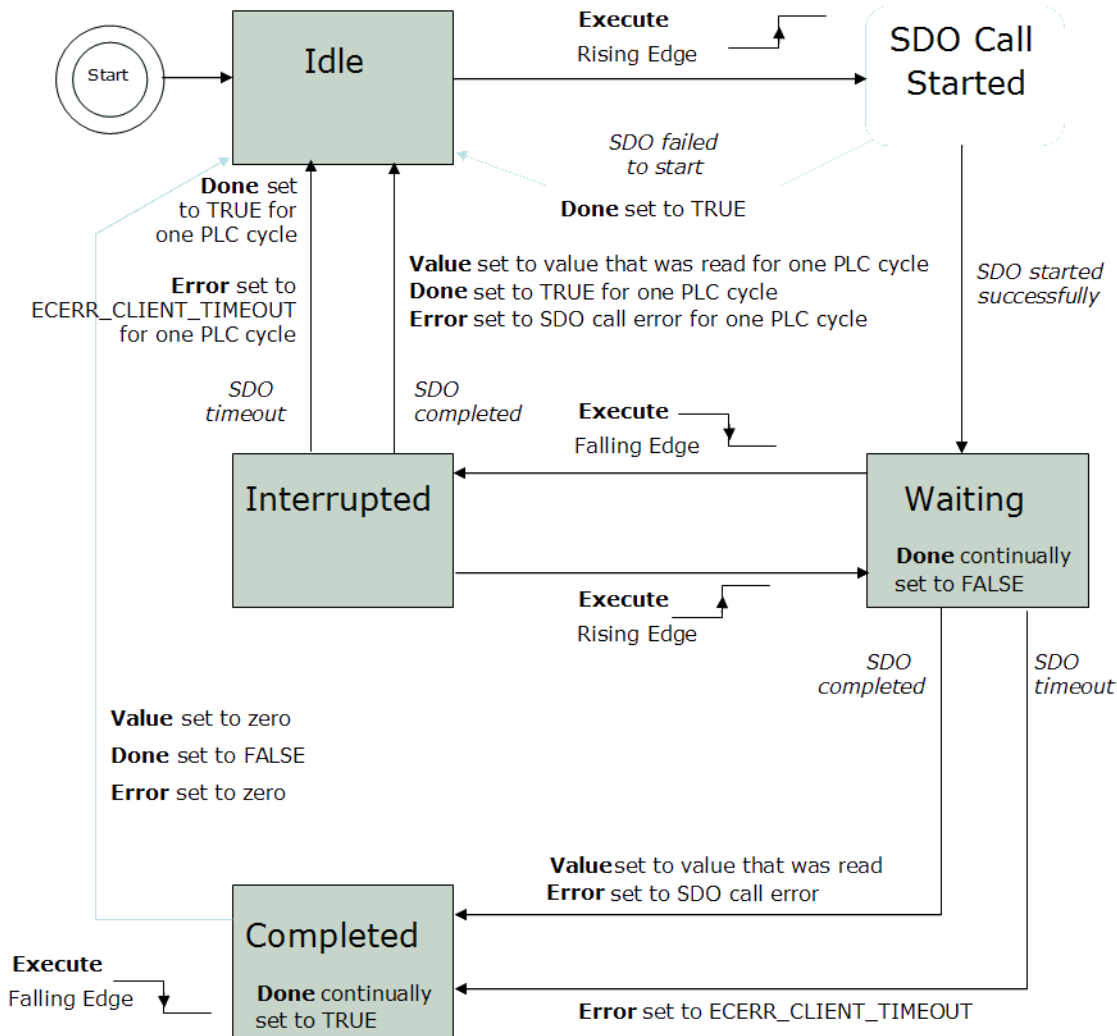


Figure 2-3: ECATReadSdoData State Diagram

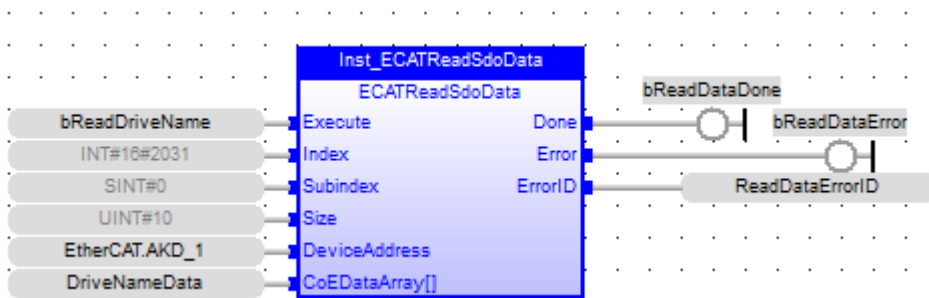
Error Codes

Value Dec (hex)	Error Code	Description
0	ECERR_OK	The SDO call succeeded.
1000 (0x3E8)	ECERR_DEVICE_ACTION_IS_INVALID	The requested Device Action is invalid.
1003 (0x3EB)	ECERR_DEVICE_ACTION_MINIMUM_ONE_NODE_REQD	A minimum of one device must be present in the EtherCAT network.

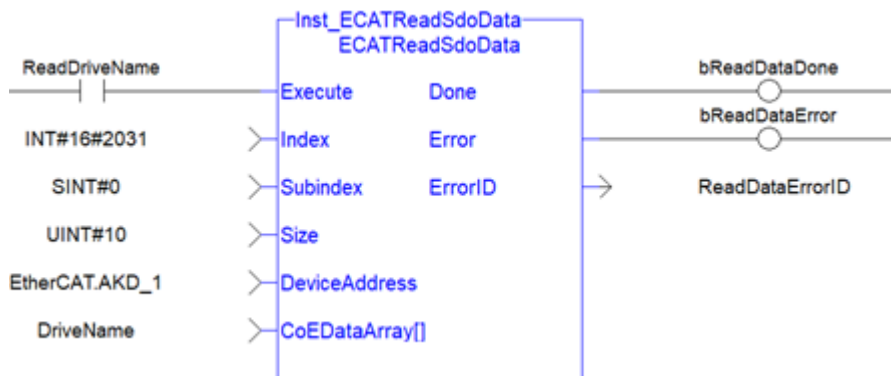
Value Dec (hex)	Error Code	Description
1004 (0x3EC)	ECERR_DEVICE_ACTION_MINIMUM_ONE_DC_NODE_REQD	A minimum of one Distributed clock capable device must be present in the EtherCAT network.
107 = 0x6B	ECERR_INVALID_ARRAY_SIZE	The size of the connections is too small. - When the array size is smaller than the number of bad connections: - The array is filled with the data to the size of the array. - The error ECERR_INVALID_ARRAY_SIZE is set. - In this scenario, the output ConnectionCount is set to the size of the array and it is smaller than the number of actual bad connections.
1792 (0x700)	ECERR_DEVICE_ERROR	EtherCAT device is not accessible.
1794 (0x702)	ECERR_DEVICE_INVALIDCMD	Invalid mailbox command.
1795 (0x703)	ECERR_DEVICE_INVALIDINDEX	An invalid value for the Index input was specified.
1796 (0x704)	ECERR_DEVICE_INVALIDACCESS	Reading of the variable is not permitted.
1797 (0x705)	ECERR_DEVICE_INVALIDSIZE	An invalid size for the parameter was specified.
1798 (0x706)	ECERR_DEVICE_INVALIDDATA	Invalid parameter value(s) in SDO index and/or sub-index.
1799 (0x707)	ECERR_DEVICE_NOTREADY	Device is not in a ready state. The network is not in operational.
1800 (0x708)	ECERR_DEVICE_BUSY	Device is not available to respond.
1801 (0x709)	ECERR_DEVICE_INVALIDCONTEXT	Device responded with an illegal error code, indicating the command is not allowed under the present conditions.
1802 (0x70A)	ECERR_DEVICE_NOMEMORY	EtherCAT mailbox is out of memory or device is out of disk space.
1803 (0x70B)	ECERR_DEVICE_INVALIDPARM	EtherCAT mailbox request was not valid.
1804 (0x70C)	ECERR_DEVICE_NOTFOUND	EtherCAT device not found.
1805 (0x70D)	ECERR_DEVICE_SYNTAX	An unexpected error occurred.
1810 (0x712)	ECERR_DEVICE_INVALIDSTATE	The EtherCAT device is in an invalid state.
1817 (0x719)	ECERR_DEVICE_TIMEOUT	The EtherCAT device failed to respond, timing out.
1826 (0x722)	ECERR_DEVICE_INSERTMAILBOX	Error while inserting the mailbox command into internal FIFO.
1827 (0x723)	ECERR_DEVICE_INVALIDOFFSET	An invalid value for the SubIndex input was specified.

Value Dec (hex)	Error Code	Description
1828 (0x724)	ECERR_DEVICE_UNKNOWNMAILBOXCMD	The master sent an unknown mailbox command to the slave.
1829 (0x725)	ECERR_DEVICE_ACCESSDENIED	Device responded with an invalid access error code, indicating the command is not allowed.
1832 (0x728)	ECERR_DEVICE_INVALIDADDR	The specified EtherCAT node address is invalid.
1836 (0x72c)	ECERR_DEVICE_NOT_A_FSOE_MASTER	Device is not a FSoE master.
1837 (0x072D)	ECERR_DEVICE_DISCONNECTED	The EtherCAT device is disconnected.
1920 (0x780)	ECERR_DEVICE_PARAM_ACCESS_ERROR	Unknown error occurred while accessing parameter.
1921 (0x781)	ECERR_DEVICE_PARAM_NOT_FOUND	Parameter was not found.
1922 (0x782)	ECERR_DEVICE_PARAM_NOT_INTEGER	Parameter is a floating-point value. Integer value required.
1923 (0x783)	ECERR_DEVICE_VALUE_IS_NEGATIVE	No negative values allowed. Value specified was negative.
1924 (0x784)	ECERR_DEVICE_VALUE_OUT_OF_RANGE	Value is out of data-range.
1925 (0x785)	ECERR_DEVICE_VALUE_GREATER_THAN_MAX	Value bigger than maximum.
1926 (0x786)	ECERR_DEVICE_VALUE_LOWER_THAN_MIN	Value lower than minimum.
2048 (0x800)	ECERR_CLIENT_ERROR	Error in Mailbox response to a previously sent mailbox command.
2049 (0x801)	ECERR_CLIENT_TIMEOUT	The SDO command timed out.
2050 (0x802)	ECERR_CLIENT_INVALIDPARAM	An invalid value was specified.
2051 (0x803)	ECERR_CLIENT_INVALIDSIZE	An invalid value for the size input was specified.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* Read DRV.NAME on first AKD Drive on EtherCAT network *)
Inst_ECATReadSdoData(TRUE, 16#2031, 0, 10, EtherCAT.AKD_1, DriveName );
IF Inst_ECATReadSdoData.Done OR Inst_ECATReadSdoData.Error THEN
    Inst_ECATReadSdoData(FALSE, 16#2031, 0, 10, EtherCAT.AKD_1, DriveName );
END_IF;
```

See Also

- "ECATReadSdo" (→ p. 68)
- "ECATWriteSdoData" (→ p. 95)

1.2.16 ECATWCStatus

PLCopen ✓

Pipe Network ✓

 **Function** - Returns the current number of working counter errors for the Sync unit.

- The working counter errors are cleared to zero when the EtherCAT network is taken from **Init** to **OP** state.
- Value **0 (zero)** means no working counter errors.
 - When the value is non zero, the master automatically reduces the count by **1** for every thousand good frames received.
- When the working counter error exceeds the Working Counter Error Limit, the EtherCAT network is stopped.

TIP

See Check for Working Counter Errors in EtherCAT Communication Diagnosis Steps for an example of implementing this function.

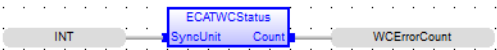
Inputs

Input	Data Type	Range	Unit	Default	Description
SyncUnit	INT	-32768 to +32767	N/A	No default	Sync Unit Index

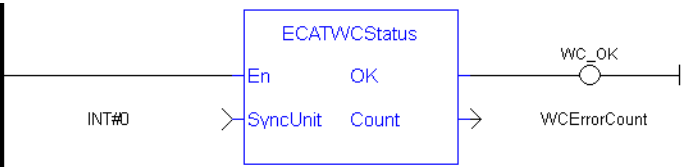
Outputs

Output	Data Type	Range	Unit	Description
Count	UDINT	No range	N/A	Working Counter error.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(*****)  
(* read Ethercat Working counter value *)
```

```
*****)  
ErrorCounter := ECATWCStatus( 0 );
```

See Also

- "ECATDeviceStatus" (→ p. 47)
- "ECATMasterStatus" (→ p. 59)

1.2.17 ECATWritePDOData



This function/function block was added in KAS v4.05.



Function Block - Writes data to an Output Rx PDO object of the EtherCAT device in the cyclic EtherCAT frame.

- When the Enable input is TRUE, then on every cycle, the **CoEDataArray** is copied to cyclic EtherCAT frame.
- The EtherCAT Device and the EtherCAT Master should be in Operational State (EC_STATE_OP).
- SerializeOut can be used to convert the value in the bytes to **CoEDataArray**.
 - Endianness: EtherCAT transmits the data in little endian format.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	When Enable is TRUE, bytes are copied from CoEDataArray to the output Output Rx PDO object.
Index	UINT	0 to 65535	N/A	No default	The CoE index of the destination Input (Tx) PDO object.
SubIndex	USINT	0 to 255	N/A	No default	The CoE sub-index of the destination Input (Tx) PDO object.
Size	USINT	1 to 255	BYTE	No default	• The size of the CoE object to write. • The size should be the exact size of the CoE object.
DeviceAddress	INT	0 to 65535	N/A	No default	The EtherCAT address of the device from which data is read from. • The first node usually has the value 1001. • The second node usually has the value 1002.

TIP

Use the members of the EtherCAT structure to specify a device's address.

The screenshot shows two windows of the EtherCAT structure. The top window, titled 'EtherCAT.AKD_1', shows a tree view with 'EtherCAT' as the root, containing 'AKD_1', 'AKD_2', and 'Coupler_3'. The bottom window, titled 'EtherCAT.AKD2G_1', shows a tree view with 'EtherCAT' as the root, containing 'Coupler_3', 'AKD2G_1', 'AKD2G_2', and 'Coupler_1'. Both windows have a 'Variables: (all)' dropdown at the bottom.

Input	Data Type	Range	Unit	Default	Description
CoEDataArray[]	USINT[]	- Dimension: $\geq$ Size. - Elements: No range	N/A	No default	The data in this array is copied to the Output Rx PDO data.

Outputs

Output	Data Type	Range	Unit	Description
Status	BOOL	FALSE, TRUE	N/A	If TRUE, indicates data was written without error.
Error	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed with error.
ErrorID	DINT	No range	N/A	The function block error result, if the Error is TRUE. - Indicates the error if Error output is TRUE. - See "Error Codes" (→ p. 83). - Upon success, Error is set to 0 (zero).

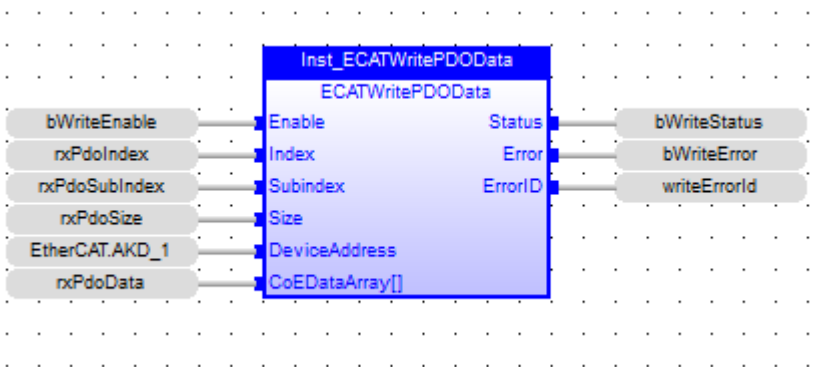
1.2.17.0.1 Error Codes

Value Dec (hex)	Error Code	Description
0	ECERR_OK	The SDO call succeeded.
1000 (0x3E8)	ECERR_DEVICE_ACTION_IS_INVALID	The requested Device Action is invalid.
1003 (0x3EB)	ECERR_DEVICE_ACTION_MINIMUM_ONE_NODE_REQD	A minimum of one device must be present in the EtherCAT network.
1004 (0x3EC)	ECERR_DEVICE_ACTION_MINIMUM_ONE_DC_NODE_REQD	A minimum of one Distributed clock capable device must be present in the EtherCAT network.
107 = 0x6B	ECERR_INVALID_ARRAY_SIZE	The size of the connections is too small. - When the array size is smaller than the number of bad connections: - The array is filled with the data to the size of the array. - The error ECERR_INVALID_ARRAY_SIZE is set. - In this scenario, the output ConnectionCount is set to the size of the array and it is smaller than the number of actual bad connections.
1792 (0x700)	ECERR_DEVICE_ERROR	EtherCAT device is not accessible.
1794 (0x702)	ECERR_DEVICE_INVALIDCMD	Invalid mailbox command.
1795 (0x703)	ECERR_DEVICE_INVALIDINDEX	An invalid value for the Index input was specified.

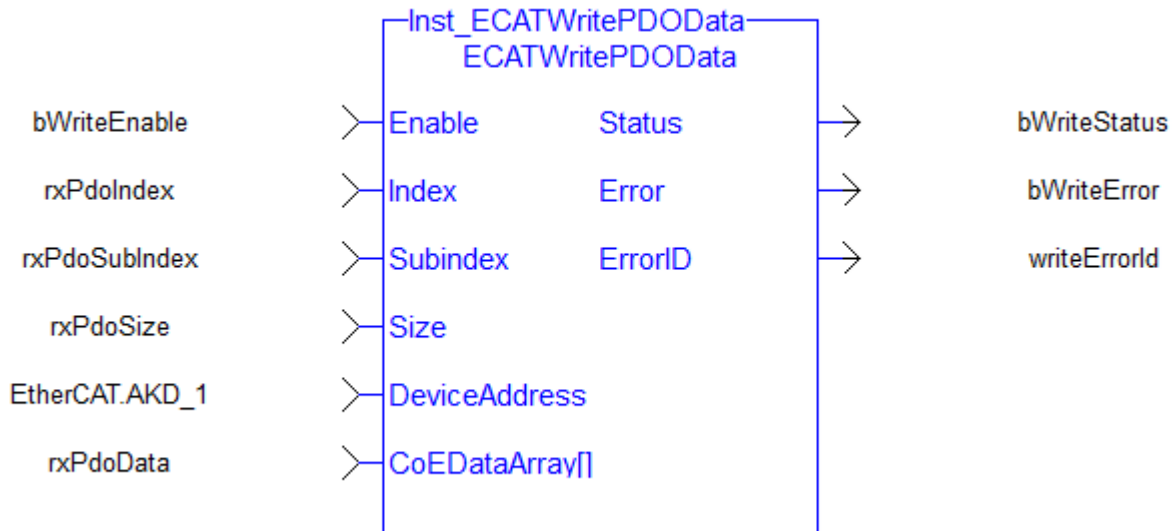
Value Dec (hex)	Error Code	Description
1796 (0x704)	ECERR_DEVICE_INVALIDACCESS	Reading of the variable is not permitted.
1797 (0x705)	ECERR_DEVICE_INVALIDSIZE	An invalid size for the parameter was specified.
1798 (0x706)	ECERR_DEVICE_INVALIDDATA	Invalid parameter value(s) in SDO index and/or sub-index.
1799 (0x707)	ECERR_DEVICE_NOTREADY	Device is not in a ready state. The network is not in operational.
1800 (0x708)	ECERR_DEVICE_BUSY	Device is not available to respond.
1801 (0x709)	ECERR_DEVICE_INVALIDCONTEXT	Device responded with an illegal error code, indicating the command is not allowed under the present conditions.
1802 (0x70A)	ECERR_DEVICE_NOMEMORY	EtherCAT mailbox is out of memory or device is out of disk space.
1803 (0x70B)	ECERR_DEVICE_INVALIDPARM	EtherCAT mailbox request was not valid.
1804 (0x70C)	ECERR_DEVICE_NOTFOUND	EtherCAT device not found.
1805 (0x70D)	ECERR_DEVICE_SYNTAX	An unexpected error occurred.
1810 (0x712)	ECERR_DEVICE_INVALIDSTATE	The EtherCAT device is in an invalid state.
1817 (0x719)	ECERR_DEVICE_TIMEOUT	The EtherCAT device failed to respond, timing out.
1826 (0x722)	ECERR_DEVICE_INSERTMAILBOX	Error while inserting the mailbox command into internal FIFO.
1827 (0x723)	ECERR_DEVICE_INVALIDOFFSET	An invalid value for the SubIndex input was specified.
1828 (0x724)	ECERR_DEVICE_UNKNOWNMAILBOXCMD	The master sent an unknown mailbox command to the slave.
1829 (0x725)	ECERR_DEVICE_ACCESSDENIED	Device responded with an invalid access error code, indicating the command is not allowed.
1832 (0x728)	ECERR_DEVICE_INVALIDADDR	The specified EtherCAT node address is invalid.
1836 (0x72c)	ECERR_DEVICE_NOT_A_FSOE_MASTER	Device is not a FSoE master.
1837 (0x072D)	ECERR_DEVICE_DISCONNECTED	The EtherCAT device is disconnected.
1920 (0x780)	ECERR_DEVICE_PARAM_ACCESS_ERROR	Unknown error occurred while accessing parameter.
1921 (0x781)	ECERR_DEVICE_PARAM_NOT_FOUND	Parameter was not found.
1922 (0x782)	ECERR_DEVICE_PARAM_NOT_INTEGER	Parameter is a floating-point value. Integer value required.

Value Dec (hex)	Error Code	Description
1923 (0x783)	ECERR_DEVICE_VALUE_IS_NEGATIVE	No negative values allowed. Value specified was negative.
1924 (0x784)	ECERR_DEVICE_VALUE_OUT_OF_RANGE	Value is out of data-range.
1925 (0x785)	ECERR_DEVICE_VALUE_GREATER_THAN_MAX	Value bigger than maximum.
1926 (0x786)	ECERR_DEVICE_VALUE_LOWER_THAN_MIN	Value lower than minimum.
2048 (0x800)	ECERR_CLIENT_ERROR	Error in Mailbox response to a previously sent mailbox command.
2049 (0x801)	ECERR_CLIENT_TIMEOUT	The SDO command timed out.
2050 (0x802)	ECERR_CLIENT_INVALIDPARM	An invalid value was specified.
2051 (0x803)	ECERR_CLIENT_INVALIDSIZE	An invalid value for the size input was specified.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
// Convert the variable to byte array.
SerializeOut( rxPdoData, rxTargetTorque, 0, FALSE );
// Write the bytes to the cyclic etherCAT rxPDO frame.
Inst_ECATWritePDOData1( TRUE, 16#6071, 0, 4, EtherCAT.AKD_1, rxPdoData );
IF Inst_ECATWritePDOData1.Status THEN
    // Data write is successful. Perform next actions.
END_IF;
```

See Also

- "ECATReadPDOData" (→ p. 63)
- "ECATWriteSdo" (→ p. 89)
- SerializeOut

1.2.18 ECATWriteData

PLCopen ✓

Pipe Network ✓



Function - Writes cyclic parameter (byte offset format).

ⓘ IMPORTANT

This is a low level function and should be used carefully by **advanced users**.

NOTE

- The valid ranges for the **Value** parameter are:
 - For 1 byte: 0 to 255
 - For 2 bytes: 0 to 65535
 - For 4 bytes: -2147483648 to 2147483647
(The sign bit represents the most significant bit in the data word).
- Modify the EtherCAT process image by directly writing values in it.
- If you have a device other than the drive, this function is used for more than just debug.
 - It is used to set the status of the module (e.g., Stepper I/O slice) in the case the project is based on an external XML file because it contains unsupported EtherCAT device.
 - See Add an Unsupported EtherCAT Device.

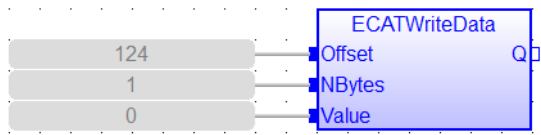
Inputs

Input	Data Type	Range	Unit	Default	Description
Offset	UINT	0 to 1500	Bytes	No default	Offset in bytes from the beginning of the frame. The maximum size of an Ethernet frame is 1500.
ⓘ IMPORTANT - The Offset value required to access may change when: - The firmware for any device on the EtherCAT network is updated or Whenever the EtherCAT network topology changes. - When performing an update of a network device or changing the network topology, export the ENI file and check the Offset value needed to access the desired information.					
Nbytes	SINT	1, 2, or 4	Bytes	No default	Number of bytes to write.
Value	DINT	-2147483648 to 2147483647	N/A	No default	Value to be written in the image. Only the number of bytes specified by Nbytes is copied.

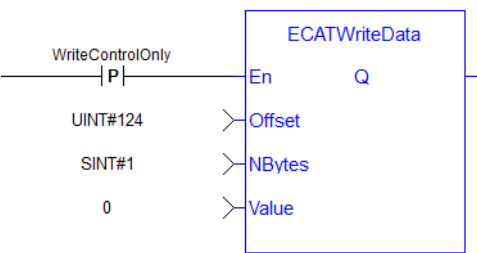
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• TRUE if the data was written. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//For use with Kollmorgen Thermocouple slice I/O to read in deg C
//Lookup offset by exporting ENI file after EtherCAT network is scanned
//Use offst 124 (byte) to write 0 in control word to allow temperature to be
shown on status byte

ON WriteControlOnly DO
  ECATWriteData( 124, 1, 0 );
END_DO
```

See Also

"ECATReadData" (→ p. 61)

1.2.19 ECATWriteSdo

PLCopen ✓

Pipe Network ✓



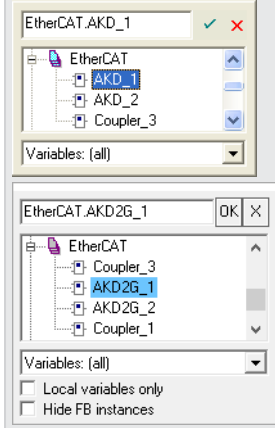
Function Block - Writes a 32-bit word to I/O nodes using a CANopen SDO write command.

Notes

- This function block **uses and reserves** the EtherCAT SDO Channel.
- The SDO Channel remains reserved until the done output is TRUE.
- This FB should be called at each cycle until either the Done or Error output is TRUE.
 - If it is not called at each cycle, the rest of SDO communication (e.g., the AKD GUI Views) is blocked.
- Using this FB in SFC P0 or P1 steps is not recommended because these steps are executed only once.
- If this FB is used in P0 or P1, then it must be used in an SFC N step to ensure the FB completes.
- See When can I expect my SDO command to be completed? in the FAQs on how to set up the update rate for SDO communication.
- See "EtherCAT Function Blocks that Work with SDOs" (→ p. 23) about function blocks used to work with drive or remote I/O parameters that are not supported by ML and MC function blocks.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge of Execute, an SDO write command is issued. • The function block only handles one SDO command at a time. • If Execute is toggled quickly so that another rising edge occurs before the SDO command has completed, the function block does not issue a second SDO command.
Index	INT	-32768 to +32767	N/A	No default	The object directory index of the data to be written to. See: • Communication SDOs. • Manufacturer specific SDOs. • Profile specific SDOs. To read/write an SDO object with an index greater than 16#7FFF (32767), the value must be entered in this form: <code>any_to_int(index # in hex format).</code> Example: <code>any_to_int(16#8321).</code>

Input	Data Type	Range	Unit	Default	Description
SubIndex	SINT	No range	N/A	No default	The sub-index of the object directory variable to be written to. See: • Communication SDOs. • Manufacturer specific SDOs. • Profile specific SDOs. To read/write an SDO object with an index greater than 16#7FFF (32767), the value must be entered in this form: <code>any_to_int</code>(index # in hex format). Example: <code>any_to_int</code>(16#8321).
Size	SINT	1 to 4	N/A	No default	The size (i.e., number of bytes) to write.
DeviceAddress	INT	-32768 to +32767	N/A	No default	The EtherCAT address of the device from which data is written to. • The first node usually has the value 1001. • The second node usually has the value 1002. TIP Use the members of the EtherCAT structure to specify a device's address. 
Value	DINT	-2147483648 to 2147483647	N/A	No default	The value to write to the object directory variable.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.

Output	Data Type	Range	Unit	Description
Error	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed with error.
ErrorID	DINT	No range	N/A	The SDO call error result, if Error is TRUE. • See "Error Codes" (→ p. 91). • Upon success, Error is set to 0 (zero).

State Diagram

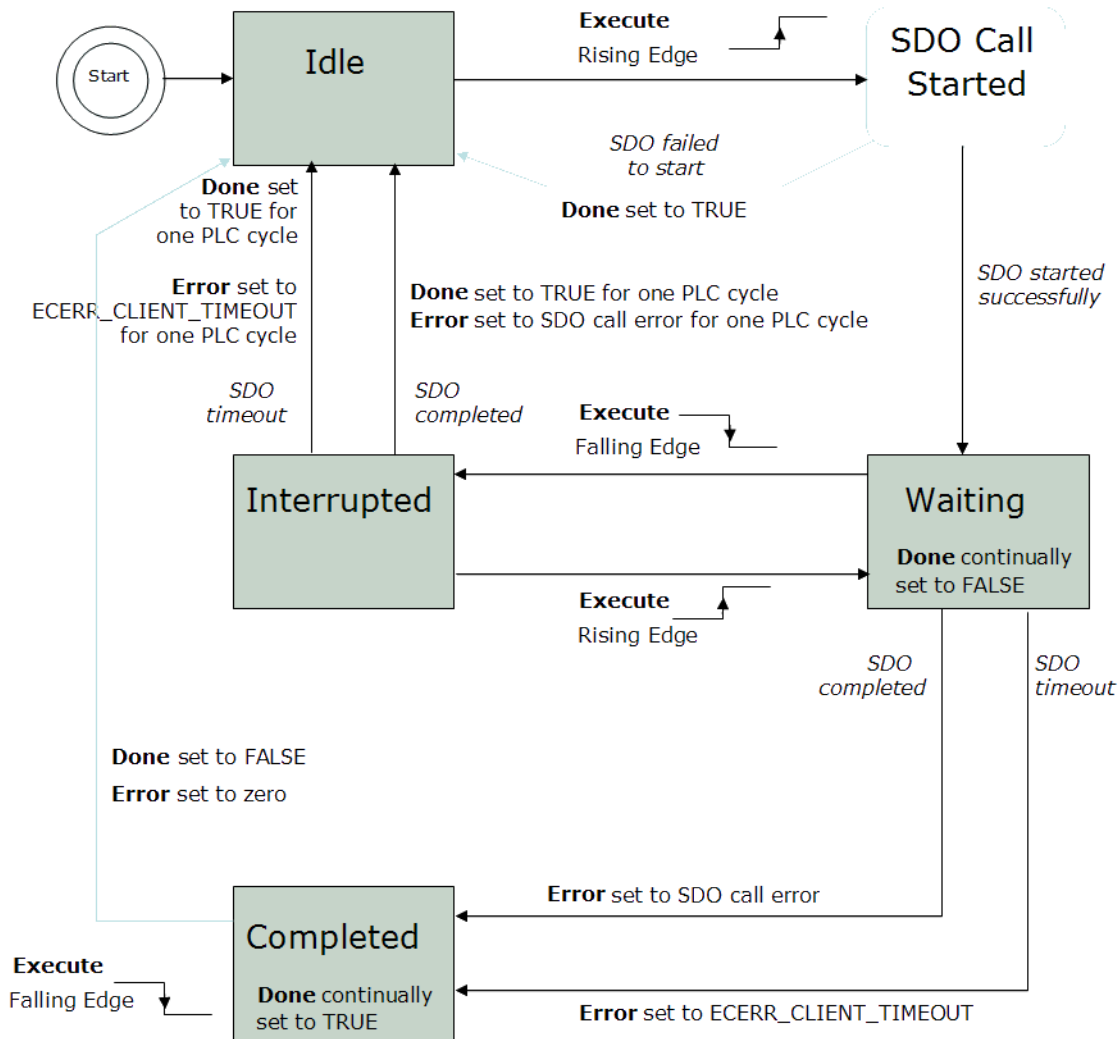


Figure 2-4: ECATWriteSdo State Diagram

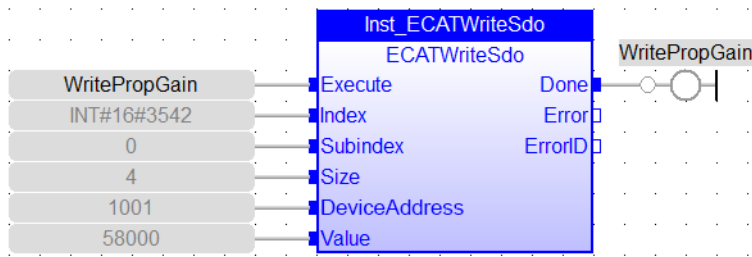
Error Codes

Value Dec (hex)	Error Code	Description
0	ECERR_OK	The SDO call succeeded.
1000 (0x3E8)	ECERR_DEVICE_ACTION_IS_INVALID	The requested Device Action is invalid.

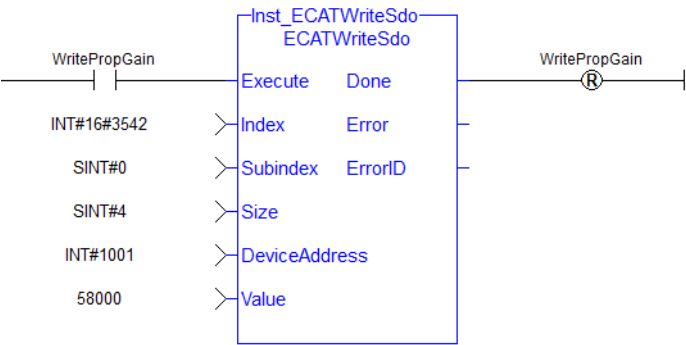
Value Dec (hex)	Error Code	Description
1003 (0x3EB)	ECERR_DEVICE_ACTION_MINIMUM_ONE_NODE_REQD	A minimum of one device must be present in the EtherCAT network.
1004 (0x3EC)	ECERR_DEVICE_ACTION_MINIMUM_ONE_DC_NODE_REQD	A minimum of one Distributed clock capable device must be present in the EtherCAT network.
107 = 0x6B	ECERR_INVALID_ARRAY_SIZE	The size of the connections is too small. - When the array size is smaller than the number of bad connections: - The array is filled with the data to the size of the array. - The error ECERR_INVALID_ARRAY_SIZE is set. - In this scenario, the output ConnectionCount is set to the size of the array and it is smaller than the number of actual bad connections.
1792 (0x700)	ECERR_DEVICE_ERROR	EtherCAT device is not accessible.
1794 (0x702)	ECERR_DEVICE_INVALIDCMD	Invalid mailbox command.
1795 (0x703)	ECERR_DEVICE_INVALIDINDEX	An invalid value for the Index input was specified.
1796 (0x704)	ECERR_DEVICE_INVALIDACCESS	Reading of the variable is not permitted.
1797 (0x705)	ECERR_DEVICE_INVALIDSIZE	An invalid size for the parameter was specified.
1798 (0x706)	ECERR_DEVICE_INVALIDDATA	Invalid parameter value(s) in SDO index and/or sub-index.
1799 (0x707)	ECERR_DEVICE_NOTREADY	Device is not in a ready state. The network is not in operational.
1800 (0x708)	ECERR_DEVICE_BUSY	Device is not available to respond.
1801 (0x709)	ECERR_DEVICE_INVALIDCONTEXT	Device responded with an illegal error code, indicating the command is not allowed under the present conditions.
1802 (0x70A)	ECERR_DEVICE_NOMEMORY	EtherCAT mailbox is out of memory or device is out of disk space.
1803 (0x70B)	ECERR_DEVICE_INVALIDPARM	EtherCAT mailbox request was not valid.
1804 (0x70C)	ECERR_DEVICE_NOTFOUND	EtherCAT device not found.
1805 (0x70D)	ECERR_DEVICE_SYNTAX	An unexpected error occurred.
1810 (0x712)	ECERR_DEVICE_INVALIDSTATE	The EtherCAT device is in an invalid state.
1817 (0x719)	ECERR_DEVICE_TIMEOUT	The EtherCAT device failed to respond, timing out.

Value Dec (hex)	Error Code	Description
1826 (0x722)	ECERR_DEVICE_INSERTMAILBOX	Error while inserting the mailbox command into internal FIFO.
1827 (0x723)	ECERR_DEVICE_INVALIDOFFSET	An invalid value for the SubIndex input was specified.
1828 (0x724)	ECERR_DEVICE_UNKNOWNMAILBOXCMD	The master sent an unknown mailbox command to the slave.
1829 (0x725)	ECERR_DEVICE_ACCESSDENIED	Device responded with an invalid access error code, indicating the command is not allowed.
1832 (0x728)	ECERR_DEVICE_INVALIDADDR	The specified EtherCAT node address is invalid.
1836 (0x72c)	ECERR_DEVICE_NOT_A_FSOE_MASTER	Device is not a FSoE master.
1837 (0x072D)	ECERR_DEVICE_DISCONNECTED	The EtherCAT device is disconnected.
1920 (0x780)	ECERR_DEVICE_PARAM_ACCESS_ERROR	Unknown error occurred while accessing parameter.
1921 (0x781)	ECERR_DEVICE_PARAM_NOT_FOUND	Parameter was not found.
1922 (0x782)	ECERR_DEVICE_PARAM_NOT_INTEGER	Parameter is a floating-point value. Integer value required.
1923 (0x783)	ECERR_DEVICE_VALUE_IS_NEGATIVE	No negative values allowed. Value specified was negative.
1924 (0x784)	ECERR_DEVICE_VALUE_OUT_OF_RANGE	Value is out of data-range.
1925 (0x785)	ECERR_DEVICE_VALUE_GREATER_THAN_MAX	Value bigger than maximum.
1926 (0x786)	ECERR_DEVICE_VALUE_LOWER_THAN_MIN	Value lower than minimum.
2048 (0x800)	ECERR_CLIENT_ERROR	Error in Mailbox response to a previously sent mailbox command.
2049 (0x801)	ECERR_CLIENT_TIMEOUT	The SDO command timed out.
2050 (0x802)	ECERR_CLIENT_INVALIDPARG	An invalid value was specified.
2051 (0x803)	ECERR_CLIENT_INVALIDSIZE	An invalid value for the size input was specified.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* Write 58.000 to PL.KP of first AKD Drive on EtheCAT network *)
Inst_ECATWriteSdo(TRUE, 16#3542, 0, 4, 1001, 58000 );
IF Inst_ECATWriteSdo.Done OR Inst_ECATWriteSdo.Error THEN
    Inst_ECATWriteSdo(FALSE, 16#3542, 0, 4, 1001, 58000 );
END_IF;
```

See Also

- "ECATReadSdo" (→ p. 68)
- "ECATWriteSdoData" (→ p. 95)

1.2.20 ECATWriteSdoData

PLCopen ✓

Pipe Network ✓



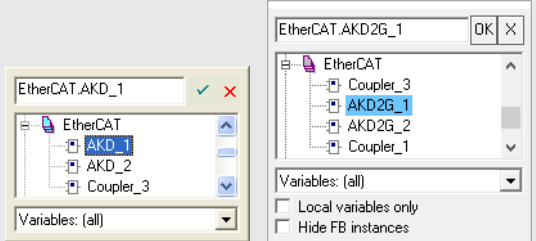
Function Block - Writes data to an EtherCAT device using a CANopen SDO write command.

Notes

- This function block **uses and reserves** the EtherCAT SDO Channel.
- The SDO Channel remains reserved until the done output is TRUE.
- This FB should be called at each cycle until either the Done or Error output is TRUE.
 - If it is not called at each cycle, the rest of SDO communication (e.g., the AKD GUI Views) is blocked.
- Using this FB in SFC P0 or P1 steps is not recommended because these steps are executed only once.
- If this FB is used in P0 or P1, then it must be used in an SFC N step to ensure the FB completes.
- See When can I expect my SDO command to be completed? in the FAQs on how to set up the update rate for SDO communication.
- See "EtherCAT Function Blocks that Work with SDOs" (→ p. 23) about function blocks used to work with drive or remote I/O parameters that are not supported by ML and MC function blocks.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge of Execute, an SDO write command is issued. • The function block only handles one SDO command at a time. • If Execute is toggled quickly so that another rising edge occurs before the SDO command has completed, the function block does not issue a second SDO command.
Index	INT	-32768 to +32767	N/A	No default	The object directory index of the data to be written to. See: • Communication SDOs. • Manufacturer specific SDOs. • Profile specific SDOs. To read/write an SDO object with an index greater than 16#7FFF (32767), the value must be entered in this form: <code>any_to_int(index # in hex format).</code> Example: <code>any_to_int(16#8321).</code>

Input	Data Type	Range	Unit	Default	Description
SubIndex	SINT	No range	N/A	No default	The sub-index of the object directory variable to be written to. See: • Communication SDOs. • Manufacturer specific SDOs. • Profile specific SDOs. To read/write an SDO object with an index greater than 16#7FFF (32767), the value must be entered in this form: <code>any_to_int(index # in hex format)</code>. Example: <code>any_to_int(16#8321)</code>.
Size	UINT	1 to 255	N/A	No default	The size (i.e., number of bytes) of the CoE object to write. The size should be the exact size of the CoE object not exceeding 255 bytes.
DeviceAddress	INT	-32768 to +32767	N/A	No default	The EtherCAT address of the device from which data is written to. • The first node usually has the value 1001. • The second node usually has the value 1002. TIP Use the members of the EtherCAT structure to specify a device's address. 
CoEdataArray[]	USINT	1 to 255	N/A	No default	The data to write to the CoE object. The size of the array should be equal to or greater than the specified size. The data is only set when an SDO write command has successfully completed.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.
Error	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed with error.

Output	Data Type	Range	Unit	Description
ErrorID	DINT	No range	N/A	The SDO call error result, if Error is TRUE. • See "Error Codes" (→ p. 97). • Upon success, Error is set to 0 (zero).

State Diagram

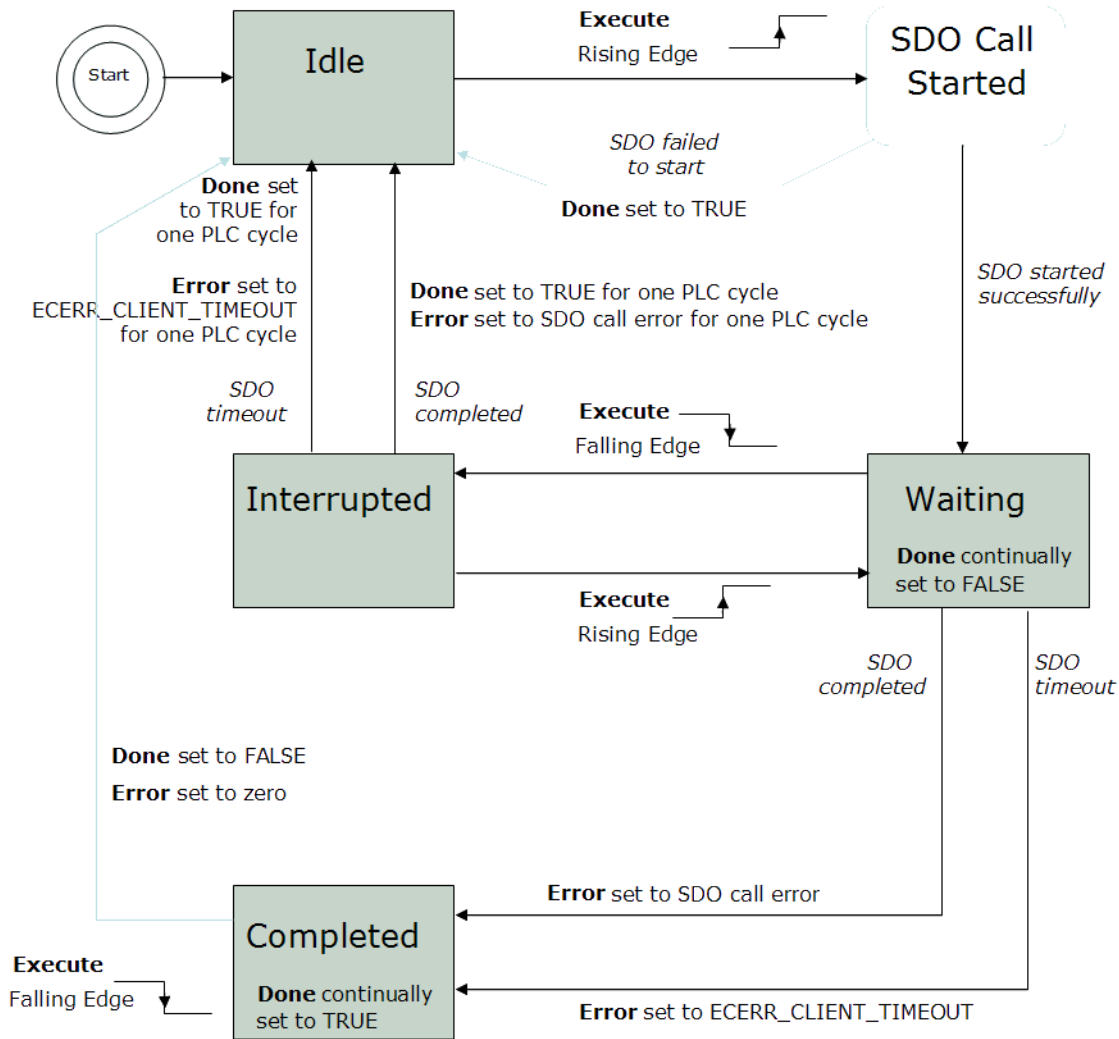


Figure 2-5: ECATWriteSdoData State Diagram

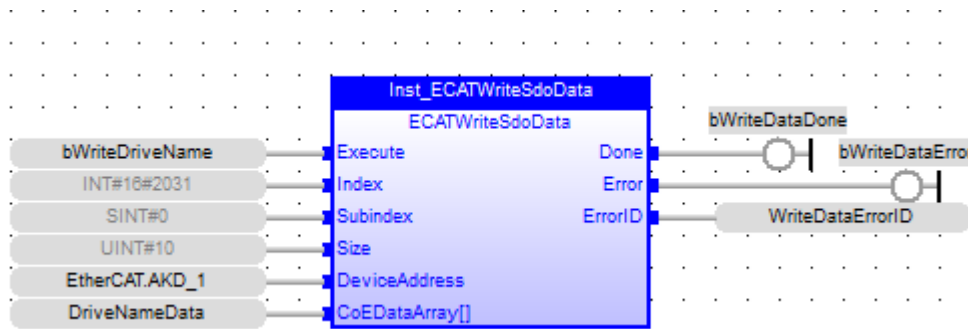
Error Codes

Value Dec (hex)	Error Code	Description
0	ECERR_OK	The SDO call succeeded.
1000 (0x3E8)	ECERR_DEVICE_ACTION_IS_INVALID	The requested Device Action is invalid.
1003 (0x3EB)	ECERR_DEVICE_ACTION_MINIMUM_ONE_NODE_REQD	A minimum of one device must be present in the EtherCAT network.

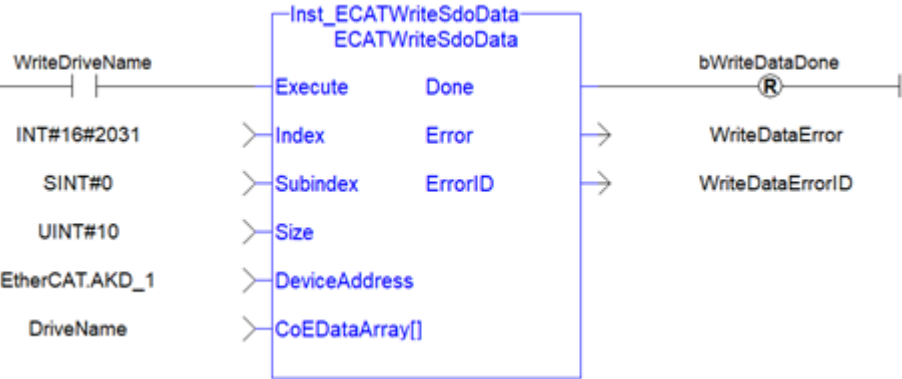
Value Dec (hex)	Error Code	Description
1004 (0x3EC)	ECERR_DEVICE_ACTION_MINIMUM_ONE_DC_NODE_REQD	A minimum of one Distributed clock capable device must be present in the EtherCAT network.
107 = 0x6B	ECERR_INVALID_ARRAY_SIZE	The size of the connections is too small. - When the array size is smaller than the number of bad connections: - The array is filled with the data to the size of the array. - The error ECERR_INVALID_ARRAY_SIZE is set. - In this scenario, the output ConnectionCount is set to the size of the array and it is smaller than the number of actual bad connections.
1792 (0x700)	ECERR_DEVICE_ERROR	EtherCAT device is not accessible.
1794 (0x702)	ECERR_DEVICE_INVALIDCMD	Invalid mailbox command.
1795 (0x703)	ECERR_DEVICE_INVALIDINDEX	An invalid value for the Index input was specified.
1796 (0x704)	ECERR_DEVICE_INVALIDACCESS	Reading of the variable is not permitted.
1797 (0x705)	ECERR_DEVICE_INVALIDSIZE	An invalid size for the parameter was specified.
1798 (0x706)	ECERR_DEVICE_INVALIDDATA	Invalid parameter value(s) in SDO index and/or sub-index.
1799 (0x707)	ECERR_DEVICE_NOTREADY	Device is not in a ready state. The network is not in operational.
1800 (0x708)	ECERR_DEVICE_BUSY	Device is not available to respond.
1801 (0x709)	ECERR_DEVICE_INVALIDCONTEXT	Device responded with an illegal error code, indicating the command is not allowed under the present conditions.
1802 (0x70A)	ECERR_DEVICE_NOMEMORY	EtherCAT mailbox is out of memory or device is out of disk space.
1803 (0x70B)	ECERR_DEVICE_INVALIDPARM	EtherCAT mailbox request was not valid.
1804 (0x70C)	ECERR_DEVICE_NOTFOUND	EtherCAT device not found.
1805 (0x70D)	ECERR_DEVICE_SYNTAX	An unexpected error occurred.
1810 (0x712)	ECERR_DEVICE_INVALIDSTATE	The EtherCAT device is in an invalid state.
1817 (0x719)	ECERR_DEVICE_TIMEOUT	The EtherCAT device failed to respond, timing out.
1826 (0x722)	ECERR_DEVICE_INSERTMAILBOX	Error while inserting the mailbox command into internal FIFO.
1827 (0x723)	ECERR_DEVICE_INVALIDOFFSET	An invalid value for the SubIndex input was specified.

Value Dec (hex)	Error Code	Description
1828 (0x724)	ECERR_DEVICE_UNKNOWNMAILBOXCMD	The master sent an unknown mailbox command to the slave.
1829 (0x725)	ECERR_DEVICE_ACCESSDENIED	Device responded with an invalid access error code, indicating the command is not allowed.
1832 (0x728)	ECERR_DEVICE_INVALIDADDR	The specified EtherCAT node address is invalid.
1836 (0x72c)	ECERR_DEVICE_NOT_A_FSOE_MASTER	Device is not a FSoE master.
1837 (0x072D)	ECERR_DEVICE_DISCONNECTED	The EtherCAT device is disconnected.
1920 (0x780)	ECERR_DEVICE_PARAM_ACCESS_ERROR	Unknown error occurred while accessing parameter.
1921 (0x781)	ECERR_DEVICE_PARAM_NOT_FOUND	Parameter was not found.
1922 (0x782)	ECERR_DEVICE_PARAM_NOT_INTEGER	Parameter is a floating-point value. Integer value required.
1923 (0x783)	ECERR_DEVICE_VALUE_IS_NEGATIVE	No negative values allowed. Value specified was negative.
1924 (0x784)	ECERR_DEVICE_VALUE_OUT_OF_RANGE	Value is out of data-range.
1925 (0x785)	ECERR_DEVICE_VALUE_GREATER_THAN_MAX	Value bigger than maximum.
1926 (0x786)	ECERR_DEVICE_VALUE_LOWER_THAN_MIN	Value lower than minimum.
2048 (0x800)	ECERR_CLIENT_ERROR	Error in Mailbox response to a previously sent mailbox command.
2049 (0x801)	ECERR_CLIENT_TIMEOUT	The SDO command timed out.
2050 (0x802)	ECERR_CLIENT_INVALIDPARM	An invalid value was specified.
2051 (0x803)	ECERR_CLIENT_INVALIDSIZE	An invalid value for the size input was specified.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* Write DRV.NAME on first AKD Drive on EtherCAT network *)
Inst_ECATWriteSdoData(TRUE, 16#2031, 0, 10, EtherCAT.AKD_1, DriveName );
IF Inst_ECATWriteSdoData.Done OR Inst_ECATWriteSdoData.Error THEN
    Inst_ECATWriteSdoData(FALSE, 16#2031, 0, 10, EtherCAT.AKD_1, DriveName );
END_IF;
```

See Also

- "ECATReadSdoData" (→ p. 74)
- "ECATWriteSdo" (→ p. 89)

1.2.21 FSoEParamsInit

PLCopen 

Pipe Network 

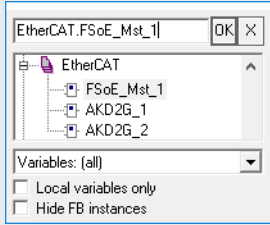


Function Block - Transfers safety parameters from the safety master to safety slave devices to initialize the safety network.

This function block reads the safety parameters from the FSoE master from the input **FSoEMasterAddress** and transfers them to the intended safety slave device using EtherCAT SDO communication.

- The function block:
 - Checks the FSoE master's register for the safety parameter transfer on the rising edge of the **Execute** input.
 - Gets the EtherCAT address of the safety slave that receives the safety parameter if the FSoE master has any safety parameters to transfer.
 - Once the address is read and validated from the FSoE master, the function block reads the actual parameter from the FSoE master and writes the parameter to the safe slave.
- The **Done** output is set to 1 when all parameters to all of the intended Safety slaves are written.
- The **Error** output is set to 1 and the appropriate ErrorID is set in the **ErrorID** output if any error occurred during this process.
- FSoE masters are supported by this function block: BBH SCU-1-EC.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge the function block initiates safety parameter transfer from the FSoE master at address FSoEMasterAddress to its safety slaves.
FSoEMasterAddress	INT	1001 to 1256	N/A	No default	The EtherCAT slave address of the FSoE master that is asked to initialize safety parameters. - The first EtherCAT slave usually has the value 1001. - The second slave usually has the value 1002. - Alternately, use the members of the EtherCAT structure to specify the EtherCAT address of the safety master. 

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.
Error	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed with error.
ErrorID	DINT	No range	N/A	The parameter initialization error result. • See "Error Codes" (→ p. 102). • Upon success, Error is set to 0 (zero).

Error Codes

Value Dec (hex)	Error Code	Description
0	ECERR_OK	The SDO call succeeded.
1000 (0x3E8)	ECERR_DEVICE_ACTION_IS_INVALID	The requested Device Action is invalid.
1003 (0x3EB)	ECERR_DEVICE_ACTION_MINIMUM_ONE_NODE_REQD	A minimum of one device must be present in the EtherCAT network.
1004 (0x3EC)	ECERR_DEVICE_ACTION_MINIMUM_ONE_DC_NODE_REQD	A minimum of one Distributed clock capable device must be present in the EtherCAT network.
107 = 0x6B	ECERR_INVALID_ARRAY_SIZE	The size of the connections is too small. • When the array size is smaller than the number of bad connections: • The array is filled with the data to the size of the array. • The error ECERR_INVALID_ARRAY_SIZE is set. • In this scenario, the output ConnectionCount is set to the size of the array and it is smaller than the number of actual bad connections.
1792 (0x700)	ECERR_DEVICE_ERROR	EtherCAT device is not accessible.
1794 (0x702)	ECERR_DEVICE_INVALIDCMD	Invalid mailbox command.
1795 (0x703)	ECERR_DEVICE_INVALIDINDEX	An invalid value for the Index input was specified.
1796 (0x704)	ECERR_DEVICE_INVALIDACCESS	Reading of the variable is not permitted.
1797 (0x705)	ECERR_DEVICE_INVALIDSIZE	An invalid size for the parameter was specified.
1798 (0x706)	ECERR_DEVICE_INVALIDDATA	Invalid parameter value(s) in SDO index and/or sub-index.
1799 (0x707)	ECERR_DEVICE_NOTREADY	Device is not in a ready state. The network is not in operational.
1800 (0x708)	ECERR_DEVICE_BUSY	Device is not available to respond.

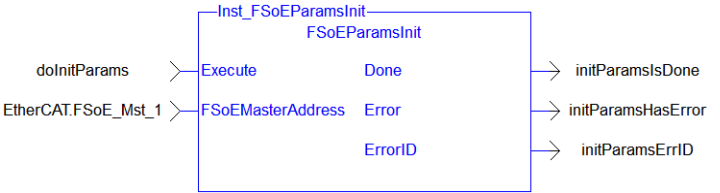
Value Dec (hex)	Error Code	Description
1801 (0x709)	ECERR_DEVICE_INVALIDCONTEXT	Device responded with an illegal error code, indicating the command is not allowed under the present conditions.
1802 (0x70A)	ECERR_DEVICE_NOMEMORY	EtherCAT mailbox is out of memory or device is out of disk space.
1803 (0x70B)	ECERR_DEVICE_INVALIDPARM	EtherCAT mailbox request was not valid.
1804 (0x70C)	ECERR_DEVICE_NOTFOUND	EtherCAT device not found.
1805 (0x70D)	ECERR_DEVICE_SYNTAX	An unexpected error occurred.
1810 (0x712)	ECERR_DEVICE_INVALIDSTATE	The EtherCAT device is in an invalid state.
1817 (0x719)	ECERR_DEVICE_TIMEOUT	The EtherCAT device failed to respond, timing out.
1826 (0x722)	ECERR_DEVICE_INSERTMAILBOX	Error while inserting the mailbox command into internal FIFO.
1827 (0x723)	ECERR_DEVICE_INVALIDOFFSET	An invalid value for the SubIndex input was specified.
1828 (0x724)	ECERR_DEVICE_UNKNOWNMAILBOXCMD	The master sent an unknown mailbox command to the slave.
1829 (0x725)	ECERR_DEVICE_ACCESSDENIED	Device responded with an invalid access error code, indicating the command is not allowed.
1832 (0x728)	ECERR_DEVICE_INVALIDADDR	The specified EtherCAT node address is invalid.
1836 (0x72c)	ECERR_DEVICE_NOT_A_FSOE_MASTER	Device is not a FSoE master.
1837 (0x072D)	ECERR_DEVICE_DISCONNECTED	The EtherCAT device is disconnected.
1920 (0x780)	ECERR_DEVICE_PARAM_ACCESS_ERROR	Unknown error occurred while accessing parameter.
1921 (0x781)	ECERR_DEVICE_PARAM_NOT_FOUND	Parameter was not found.
1922 (0x782)	ECERR_DEVICE_PARAM_NOT_INTEGER	Parameter is a floating-point value. Integer value required.
1923 (0x783)	ECERR_DEVICE_VALUE_IS_NEGATIVE	No negative values allowed. Value specified was negative.
1924 (0x784)	ECERR_DEVICE_VALUE_OUT_OF_RANGE	Value is out of data-range.
1925 (0x785)	ECERR_DEVICE_VALUE_GREATER_THAN_MAX	Value bigger than maximum.
1926 (0x786)	ECERR_DEVICE_VALUE_LOWER_THAN_MIN	Value lower than minimum.
2048 (0x800)	ECERR_CLIENT_ERROR	Error in Mailbox response to a previously sent mailbox command.

Value Dec (hex)	Error Code	Description
2049 (0x801)	ECERR_CLIENT_TIMEOUT	The SDO command timed out.
2050 (0x802)	ECERR_CLIENT_INVALIDPARM	An invalid value was specified.
2051 (0x803)	ECERR_CLIENT_INVALIDSIZE	An invalid value for the size input was specified.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_FSoEParamsInit(True, EtherCAT.FSoE_Mst_1);
```

See Also

- Set Up FSoE Master and an AKD2G with SafeMotion Monitor
- AKD2G Safety Parametrization Using FSoE with SCU-1-EC and PxMM / PCMM2G
- Troubleshooting FSoE Safety Parameters

1.3 Fieldbus - EtherNet/IP (ODVA)

These functions are used for explicit messaging.

Name	Description
eipAdapter	Provides information about the current state of the Scanner connection.
eipReadAttr	Sends an explicit message (UCMM) to an EtherNet/IP adapter to read a single CIP attribute.
eipRestartCnx	The EIP I/O connection is restarted when this function is called.
eipWriteAttr	Sends an explicit message (UCMM) to an EtherNet/IP adapter to write a single CIP attribute.

1.3.1 eipAdapter



 **Function Block** - Provides information about the current state of the Scanner connection.

- This function block is used in a program on the Adapter side.
- The servers (adapters) accessed by this block must be configured in the EtherNet/IP Scanner fieldbus configuration.
- Only one explicit message (read or write) can be sent at one time to the same server.
 - If another message is pending you receive the error report 3 (busy) after calling the block to start a new exchange.
- Consider SerializeIn and SerializeOut functions for extracting data from the read buffer.

1.3.1.0.1 Example

```
Inst_eipAdapter();    // read the EtherNet/IP adapter status

EIP_running := Inst_eipAdapter.Run;    // is it running?
EIP_connected := Inst_eipAdapter.IOCnx;    // and connected?
```

Inputs

ⓘ IMPORTANT

There are no Inputs for this function / function block.
It automatically refers to the adapter in the project.

Outputs

Output	Data Type and hyperlink for F/FB table header cellsData TypeSee Data Types .	Range	Unit	Description
Run	BOOL	FALSE, TRUE	N/A	TRUE, if the EIP stack is running.
IOcnx	BOOL	FALSE, TRUE	N/A	TRUE, if an I/O connection is established with the Scanner.

Remarks

See Also

- "eipReadAttr" (→ p. 107)
- "eipWriteAttr" (→ p. 112)

1.3.2 eipReadAttr



 **Function Block** - Sends an explicit message (UCMM) to an EtherNet/IP adapter to read a single CIP attribute.

- The servers (adapters) accessed by this block must be configured in the EtherNet/IP Scanner fieldbus configuration.
- Only one explicit message (read or write) can be sent at one time to the same server.
 - If another message is pending you receive the error report 3 (busy) after calling the block to start a new exchange.
- Consider SerializeIn and SerializeOut functions for extracting data from the read buffer.

Inputs

Input	Data Type and hyperlink for F/FB table header cellsData TypeSee Data Types .	Range	Unit	Default	Description
Snd	BOOL	FALSE, TRUE	N/A	No default	• A rising edge on this input starts the exchange. • The DONE output signals the end of exchange.
SrvIP	STRING	No range	N/A	No default	IP address of the server (adapter) (e.g., configured in the EtherNet/IP Scanner configuration).
Class	UINT	No range	N/A	No default	Class identifier of the CIP object.
Inst	UINT	1 to 65535	N/A	No default	Instance identifier of the CIP object.
Attr	UINT	No range	N/A	No default	Identifier of the CIP attribute.
Data	USINT[]	Dimension: The number of bytes to read. Data: No range	N/A	No default	• Buffer where to store the received data. • If the actual attribute length is greater than the size of this array, the value is truncated when read.

Outputs

Output	Data Type and hyperlink for F/FB table header cellsData TypeSee Data Types .	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	This output is TRUE during one cycle when the exchange is finished, whether the exchange succeeded or failed. ⚠ WARNING Warning: This output can be TRUE just after the call to block when starting a new exchange in case of invalid parameters.
RcvSize	UINT	0 to 450	N/A	- Actual size of the CIP attribute answered by the server. - If this size is greater than the size of the DATA input array, it indicates the value was truncated.
Err	UINT	No range	N/A	Main error report. - Can be one of these values: 0 = No error. 1 = Invalid input arguments. 2 = System is busy (see remarks). 3 = Timeout waiting for the answer (the timeout value is 3 seconds). 4 = UCMM error was returned by the server others = internal errors (reserved for technical support).
EmErr	UINT	No range	N/A	In case of a UCMM error, this is the CIP general status error code.
EmErrExt	UINT	No range	N/A	In case of a UCMM error, this is the CIP extended status error code.

FBD Language Example

Not available.

FPLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
// used variables
// Inst_eipReadAttr : eipReadAttr ;
// bRead : BOOL ; (* request for READ *)
// DataRead : ARRAY [0 .. 15] OF USINT ; (* read data *)
```

```

// Server identification and CIP things
#define SRVIP '192.168.33.21'
#define CLASSID UINT#100
#define INSTID_READ UINT#1
#define ATTRID UINT#3
////////////////////////////////////
////////////////////////////////////
// requested READ command
if bRead then
    Inst_eipReadAttr (bRead, SRVIP, CLASSID, INSTID_READ, ATTRID, DataRead);
end_if;
// READ answer here ?
if Inst_eipReadAttr.Done then
    // check answer - if OK answered data is in DataRead array
    if Inst_eipReadAttr.Err = 0 then
        printf ('READ ok - size = %lu bytes',
            any_to_dint (Inst_eipReadAttr.RcvSize));
    else
        printf ('READ Error %lu (UCMM Error %lu, %lu)',
            any_to_dint (Inst_eipReadAttr.Err),
            any_to_dint (Inst_eipReadAttr.EmErr),
            any_to_dint (Inst_eipReadAttr.EmErrExt));
    end_if;
    // reset READ command and block input
    Inst_eipReadAttr (false, SRVIP, CLASSID, INSTID_READ, ATTRID, DataRead);
    bRead := false;
end_if;

```

See Also

- "eipAdapter" (→ p. 106)
- "eipWriteAttr" (→ p. 112)

1.3.3 eipRestartCnx

PLCopen ✓

Pipe Network ✓

This function/function block was added in KAS v4.03.



Function - The EIP I/O connection is restarted when this function is called.

Example: ForwardClose and ForwardOpen requests are sent to the specified EIP adapter IP address. RJ45

ⓘ **IMPORTANT**

This function should be only called under certain conditions (e.g., IF statements, etc.).
It should not be called every cycle.

Inputs

Input	Data Type and hyperlink for F/FB table header cellsData TypeSee Data Types .	Range	Unit	Default	Description
IP	STRING	No range	N/A	No default	IP address of the Ethernet I/P adapter device.

Outputs

Output	Data Type and hyperlink for F/FB table header cellsData TypeSee Data Types .	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	- Result of the function call. - If FALSE, the specified IP address in the Input is incorrect.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
IF bRestartEipCnx THEN
    bOK := EipRestartCnx( '10.50.66.109');
    bRestartEipCnx := FALSE;
END_IF;
```

1.3.4 eipWriteAttr



 **Function Block** - Sends an explicit message (UCMM) to an EtherNet/IP adapter to write a single CIP attribute.

- The servers (adapters) accessed by this block must be configured in the EtherNet/IP Scanner fieldbus configuration.
- Only one explicit message (read or write) can be sent at one time to the same server.
 - If another message is pending you receive the error report 3 (busy) after calling the block to start a new exchange.
- Consider SerializeIn and SerializeOut functions for extracting data from the read buffer.

Inputs

Input	Data Type	Range	Unit	Default	Description
Snd	BOOL	FALSE, TRUE	N/A	No default	• A rising edge on this input starts the exchange. • The DONE output signals the end of exchange.
SrvIP	STRING	No range	N/A	No default	IP address of the server (adapter) (e.g., configured in the EtherNet/IP Scanner configuration).
Class	UINT	No range	N/A	No default	Class identifier of the CIP object.
Inst	UINT	1 to 65535	N/A	No default	Instance identifier of the CIP object.
Attr	UINT	No range	N/A	No default	Identifier of the CIP attribute.
Size	UINT	1 to 450	N/A	No default	• Number of bytes to write. • Cannot exceed 450 bytes.
Data	USINT[]	Dimension: The number of bytes to write. Data: No range	N/A	No default	Buffer containing the data to write.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	• This output is TRUE during one cycle when the exchange is finished, whether the exchange succeeded or failed. • Warning: This output can be TRUE just after the call to block when starting a new exchange in case of invalid parameters.

Output	Data Type	Range	Unit	Description
Err	UINT	No range	N/A	Main error report. - Can be one of these values: 0 = No error. 1 = Invalid input arguments. 2 = System is busy (see remarks). 3 = Timeout waiting for the answer (the timeout value is 3 seconds). 4 = UCMM error was returned by the server others = internal errors (reserved for technical support).
EmErr	UINT	No range	N/A	In case of a UCMM error, this is the CIP general status error code.
EmErrExt	UINT	No range	N/A	In case of a UCMM error, this is the CIP extended status error code.

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
// used variables
// Inst_eipWriteAttr : eipWriteAttr ;
// bWrite : BOOL ; (* request for WRITE *)
// DataWrite : ARRAY [0 .. 15] OF USINT; (* written data *)
// uiSizeWrite : UINT := UINT#16 ; (* number of bytes to read *)
// Server identification and CIP things
#define SRVIP '192.168.33.21'
#define CLASSID UINT#100
#define INSTID_WRITE UINT#2
#define ATTRID UINT#3
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
// requested WRITE command
if bWrite then
    Inst_eipWriteAttr (bWrite, SRVIP, CLASSID, INSTID_WRITE, ATTRID,
                      uiSizeWrite, DataWrite);
end_if;
// WRITE answer here ?
if Inst_eipWriteAttr.Done then
    // check answer
```



```

    if Inst_eipWriteAttr.Err = 0 then
        printf ('WRITE ok');
    else
        printf ('WRITE Error %lu - (UCMM Error %lu, %lu)',
            any_to_dint (Inst_eipWriteAttr.Err),
            any_to_dint (Inst_eipWriteAttr.EmErr),
            any_to_dint (Inst_eipWriteAttr.EmErrExt));
    end_if;
    // reset WRITE command and block input
    Inst_eipWriteAttr (false, SRVIP, CLASSID, INSTID_WRITE, ATTRID,
        uiSizeWrite, DataWrite);
    bWrite := false;
end_if;

```

See Also

- "eipAdapter" (→ p. 106)
- "eipReadAttr" (→ p. 107)

2 Kollmorgen UDFBs

A Kollmorgen UDFB is a pre-defined function block created by Kollmorgen to simplify certain tasks or demonstrate a particular function.

- A Kollmorgen UDFB must be instantiated before it can be used.
- The code inside a Kollmorgen UDFB can be modified by creating an unlocked copy in the subprogram section in the project tree.

2.1 UDFB Functions and Function Blocks	117
2.1.1 Common	117
2.1.2 Pipe Network	117
2.1.3 PLCopen	118
2.1.4 PLC	118
2.2 Create or Edit UDFBs	119
2.2.1 Create a UDFB Instance	119
2.2.2 Edit UDFBs	119
2.3 UDFB - Motion / Common	121
2.3.1 FB_AKDFltRpt	122
2.3.2 FB_AxisPlsPosModulo	125
2.3.3 FB_AxisPlsPosNoModulo	128
2.3.4 FB_Cylinder	131
2.3.5 FB_S700FltRpt	134
2.4 UDFB - Motion / Pipe Network	137
2.4.1 MLFB_DriveFault	138
2.4.2 MLFB_ECATRestart	141
2.4.3 Fast Homing - UDFB - Motion/Pipe Network	143
2.4.4 Homing - UDFB - Motion/Pipe Network	160
2.4.5 MLFB_Jog	184
2.4.6 PLS - UDFB - Motion/Pipe Network	186
2.5 UDFB - Motion / PLCopen	194
2.5.1 Drive Fault Reporting - UDFB - Motion/PLCopen	195
2.5.2 MCFB_ECATRestart	203
2.5.3 MCFB_StepAbsSwitchFastInput	205
2.5.4 MCFB_StepLimitSwitchFastInput	210
2.5.5 Homing - UDFB - Motion/PLCopen	214
2.5.6 MCFB_Jog	236
2.5.7 MCFB_GearedWebTension	238

2.6 UDFB - PLC	245
2.6.1 FB_ElapseTime	246
2.6.2 FB_FirstOrderDigitalFilter	248
2.6.3 FB_PWDutyOutput	254
2.6.4 FB_ScaleInput - Scaling Analog IO	257
2.6.5 FB_ScaleOutput - Scaling Analog IO	259
2.6.6 FB_TemperaturePID	261
2.6.7 PipeNetwork_FFLD - Special FFLD Function	265
2.6.8 ProfilesCode_FFLD - Special FFLD Function	267

2.1 UDFB Functions and Function Blocks

These are the UDFB functions and function blocks:

- "Common" (→ p. 117)
- "Pipe Network" (→ p. 117)
- "PLCopen" (→ p. 118)
- "PLC" (→ p. 118)

2.1.1 Common

Name	Description
FB_AKDFltRpt	Outputs AKD drive fault information.
FB_AxisPlsPosModulo	Used for any position of a modulo axis in both directions.
FB_AxisPlsPosNoModulo	Used for any position of a non-modulo axis in both directions.
FB_Cylinder	Used to control a cylinder and the Limit Switches.
FB_S700FltRpt	Outputs S700 drive fault Information.

2.1.2 Pipe Network

Name	Description
MLFB_DriveFault	Returns the fault status, number, and description of the requested axis mapped to a Kollmorgen drive.
MLFB_ECATRstart	Reinitializes the EtherCAT network and the motion engine.
MLFB_HomeFindHomeFastInput	Fast homing to a home switch.
MLFB_HomeFindHomeFastInputModulo	Fast homing to a home switch: Modulo mode.
MLFB_HomeFindHomeInput	Homing to a home switch.
MLFB_HomeFindHomeInputThenZeroAngle	Homing to a home switch plus zero angle.
MLFB_HomeFindLimitFastInput	Fast homing to a limit switch.
MLFB_HomeFindLimitFastInputModulo	Fast homing to a limit switch: Modulo mode.
MLFB_HomeFindLimitInput	Homing to a limit switch.
MLFB_HomeFindLimitInputThenZeroAngle	Homing to a limit switch plus zero angle.
MLFB_HomeFindZeroAngle	Homing to zero angle reference.
MLFB_HomeMoveUntilPosErrExceeded	Homing to the position error is exceeded.
MLFB_HomeMoveUntilPosErrExceededThenZeroAngle	Homing to the position error is exceeded plus zero angle.
MLFB_HomeUsingCurrentPosition	Homing using current position.
MLFB_Jog	Define to jog an axis in the selected direction at a defined speed.
MLFB_PlsPosFw	Forward position range indicator.
MLFB_PlsPosFwBw	Forward/backward position range indicator.
MLFB_PlsTimeFw	Forward/backward Position/Time range indicator.

2.1.3 PLCopen

Name	Description
MCFB_AKDFault	Outputs AKD drive fault information.
MCFB_AKDFaultLookup	String message of the corresponding AKD drive fault number.
MCFB_DriveFault	Returns the fault status, number, and description of the requested axis mapped to a Kollmorgen drive.
MCFB_ECATRestart	Reinitializes the EtherCAT network and the motion engine.
MCFB_GearedWebTension	Facilitates dancer and tension control in an electronic geared master/slave machine design.
MCFB_Jog	Define to jog an axis in the selected direction at a defined speed.
MCFB_StepAbsolutes	Homing by setting Actual Position to the position of the feedback.
MCFB_StepAbsSwitch	Homing to a home switch.
MCFB_StepAbsSwitchFastInput	Fast homing to a home switch.
MCFB_StepBlock	Homing to a physical object, mechanically blocking the movement.
MCFB_StepLimitSwitch	Homing to a limit switch.
MCFB_StepLimitSwitchFastInput	Fast homing to a limit switch.
MCFB_StepRefPulse	Homing to a zero angle reference.

2.1.4 PLC

Name	Description
FB_ElapseTime	Keeps track of the time (oTotalOnTime) a Boolean input variable is on.
FB_FirstOrderDigitalFilter	Defined to filter an Analog signal.
FB_PWDutyOutput	Accepts an input value between the minimum and maximum input range and converts this to a duty cycle percentage.
FB_ScaleInput - Scaling Analog IO	Converts un-scaled DINT values from Analog Inputs into user units of type LREAL.
FB_ScaleOutput - Scaling Analog IO	Converts un-scaled LREAL values from a PLC program into units of type DINT.
FB_TemperaturePID	Provides PID temperature control with auto tuning.
PipeNetwork_FFLD - Special Function	Calls the PNCode function block in FFLD POU's.
ProfilesCode_FFLD - Special Function	Calls the Profiles Code Function Block in FFLD POU's.

2.2 Create or Edit UDFBs

2.2.1 Create a UDFB Instance

1. Open the PLC code.
2. Select the UDFB in the [Library](#) tree.
3. Drag-and-drop the UDFB in the PLC editor to create the instance of the UDFB.

An instance of the UDFB has now been created in **Subprograms**.

NOTE

An instance of the UDFB cannot be created directly from the dictionary or from the PLC Editor.

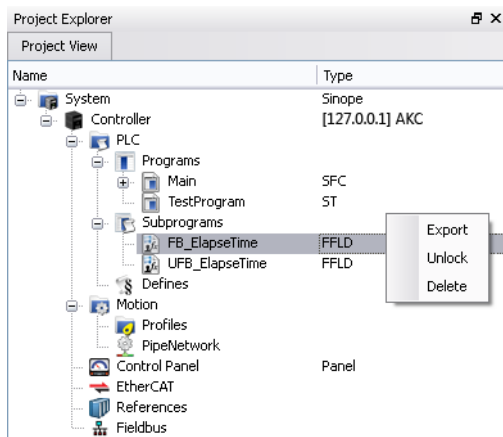
2.2.2 Edit UDFBs

By default, all Kollmorgen UDFBs are protected; they cannot be modified or renamed.

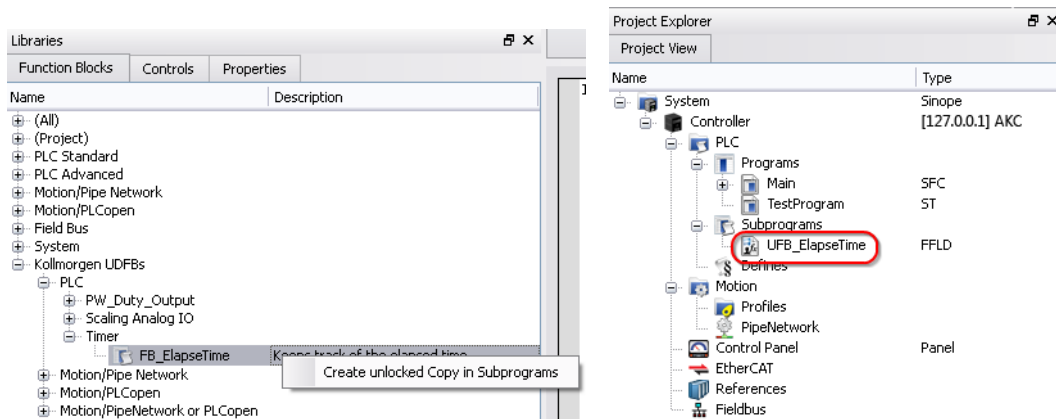
A Kollmorgen UDFB dropped into an instance cannot be edited.

There are two solutions to edit a UDFB:

- Right click a Kollmorgen UDFB that was dropped into an instance (in **Subprograms**) and click **Unlock**. This creates an unlocked version of the UDFB with the name **U<sequence number><UDFB name>**.



- Instead of dropping a Kollmorgen UDFB into an instance, right-click the UDFB and click **Create unlocked copy in Subprograms**. This creates an unlocked instance of the UDFB with the name **U<sequence number><Kollmorgen UDFB name>**.



- After a Kollmorgen UDFB is unlocked, right-click the UDFB to rename or exported it.

- Renamed UDFBs must have unique names.
- Importing a saved UDFB increments the UDFB's name.

 TIP

For a UDFB to modify a structure or array based on the output, it must first be defined it as an input. The input is automatically set as an INOUT parameter. This is because OUTs are strictly simple types.

2.3 UDFB - Motion / Common

These are the Kollmorgen UDFB Motion/Common function and function blocks.

Name	Description
FB_AKDFltRpt	Outputs AKD drive fault information.
FB_AxisPlsPosModulo	Used for any position of a modulo axis in both directions.
FB_AxisPlsPosNoModulo	Used for any position of a non-modulo axis in both directions.
FB_Cylinder	Used to control a cylinder and the Limit Switches.
FB_S700FltRpt	Outputs S700 drive fault Information.

2.3.1 FB_AKDFltRpt

PLCopen ✓

Pipe Network ✓



Function Block - Outputs AKD drive fault information.

- Provide drive fault information the application program uses to determine next steps.
 - Example: Perform a machine-controlled stop or perform an immediate disable of the servo drives.
- In the application program, sends output fault information from this UDFB to the HMI for review by the machine operator.

TIP

- This function block lists the **earliest occurring** fault first.
 - This may not be the same fault as is being reported on an AKD's display, which is based on priority.
 - The "MCFB_AKDFault" (→ p. 196) function block may be preferred as it reports the same error as displayed on the drive.
- The **oFAULT** output turns TRUE when the selected drive goes into a fault state.
 - This function block outputs the:
 - Total number of faults in the drive fault history variable. (Pre-Defined Error Field Object 1003h)
 - Fault number and message for the last three drive faults.
 - Each fault has two outputs: the fault number and a fault message.
 - The fault number is the same number reported on the display of the drive.
 - The fault message provides a short description of the fault.
 - Example: If the first fault is a feedback error with a F401 is shown on the front of the drive, the output of this FB are:
 - **oFirstFaultNumber** = 401.
 - **oFirstFaultMessage** = Failed To Set Feedback Type.
 - **iResetfaultHistory** resets the faults reported by the FB.
 - **oDriveNotUsed** outputs 1 (True) if the axis is configured to Simulated in the ProjectEtherCAT setup screen.
 - This function block can be used with either the PLCopen or Pipe Network Motion engines.

Inputs

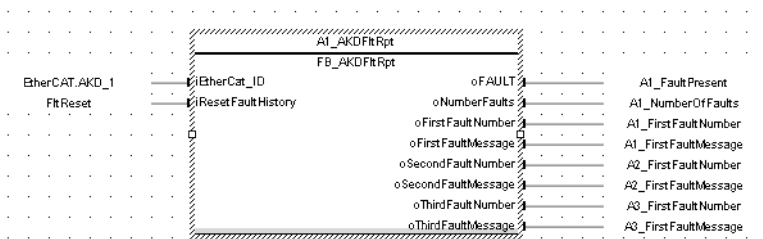
Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables the Kollmorgen UDFB. Used in the FFLD editor only.
iEtherCat_ID	INT	No range	N/A	No default	EtherCAT address of AKD Drive. Example: 1001 or AKD_1.
iRstFltHist	BOOL	FALSE, TRUE	N/A	No default	When input is TRUE, clears all Faults saved to drives history.

Outputs

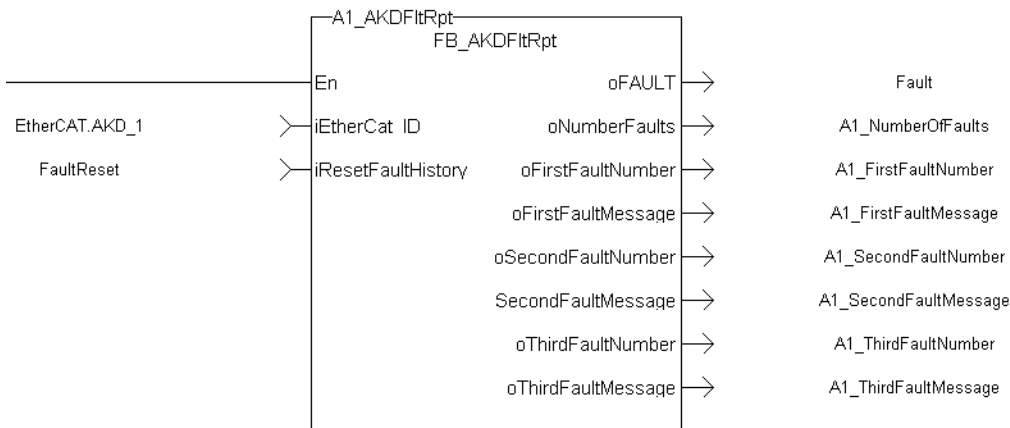
Output	Data Type	Range	Unit	Description
oFAULT	BOOL	FALSE, TRUE	N/A	TRUE if selected drive currently has a Fault.
oNumberFaults	DINT	0 to 10	N/A	Number of faults saved in the drive's history.

Output	Data Type	Range	Unit	Description
oFirstFaultNumber	DINT	100 to 999	N/A	Three-digit fault identifier.
oFirstFaultMessage	STRING	No range	N/A	Description of the fault.
oSecondFaultNumber	DINT	100 to 999	N/A	Three-digit fault identifier.
oSecondFaultMessage	STRING	No range	N/A	Description of the fault.
oThirdFaultNumber	DINT	100 to 999	N/A	Three-digit fault identifier.
oThirdFaultMessage	STRING	No range	N/A	Description of the fault.
oDriveNotUsed	BOOL	FALSE, TRUE	N/A	Is this drive: 0 (zero) = Real. 1 = Simulated.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Execute the Function Block
1_AKDFltRpt (1001, resetFaultHistST);

//Read Function Block Outputs
```

```

AKD1_Fault:= A1_AKDFltRpt.oFault;
AKD1_NumFault:= A1_AKDFltRpt.oNumberFaults;
AKD1_FirstFaultNumber:= A1_AKDFltRpt.oFirstFaultNumber;
AKD1_FirstFaultMessage:= A1_AKDFltRpt.oFirstFaultMessage;
AKD1_SecondFaultNumber:= A1_AKDFltRpt.oSecondFaultNumber;
AKD1_SecondFaultMessage:= A1_AKDFltRpt.oSecondFaultMessage;
AKD1_ThirdFaultNumber:= A1_AKDFltRpt.oThirdFaultNumber;
AKD1_ThirdFaultMessage:= A1_AKDFltRpt.oThirdFaultMessage;
;

```

NOTE

A1_FaultReporting is an instance of the **FB_S700FltRpt** function block.

See Also

- "MC_ReadStatus" (→ p. 652) (PLCopen Motion Engine)
- "MCFB_AKDFault" (→ p. 196)
- "MLAxisStatus" (→ p. 386) (Pipe Network Motion Engine)

2.3.2 FB_AxisPlsPosModulo

PLCopen ✓

Pipe Network ✓



Function Block - Used for any position of a modulo axis in both directions.

- The Boolean output **oPLS** is set to:
 - TRUE if the position has crossed the start position.
 - FALSE if the position has crossed the end position.
- The function block is executed cyclically.
 - The function block has the possibility to compensate a delay time of the connected device (e.g., glue nozzles).
- It is possible to define a hysteresis for switching on and off of the PLS.

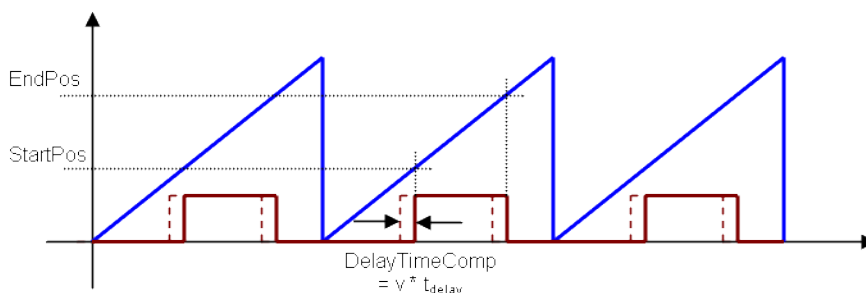
Inputs

Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL	FALSE, TRUE	N/A	No default	Enable PLS.
iPosition	LREAL	No range	User units	No default	Any position of a modulo axis.
iModuloPosition	LREAL	No range	User units	No default	Modulo position of the axis.
iStartPos	LREAL	No range	User units	No default	Start position of PLS.
iEndPos	LREAL	No range	User units	No default	End position of PLS.
iDelayTime	TIME	0ms - 24hr	TIME	No default	Delay time for compensation.
iHysteresis	LREAL	No range	User units	No default	Hysteresis.
ibForce	BOOL	FALSE, TRUE	N/A	No default	Force PLS.

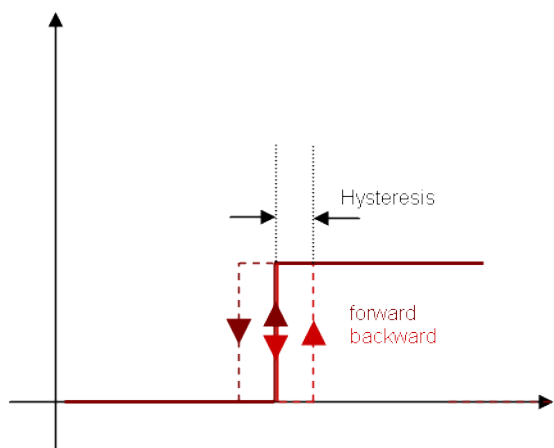
Outputs

Output	Data Type	Range	Unit	Description
oPLS	BOOL	FALSE, TRUE	N/A	Position limit switch.

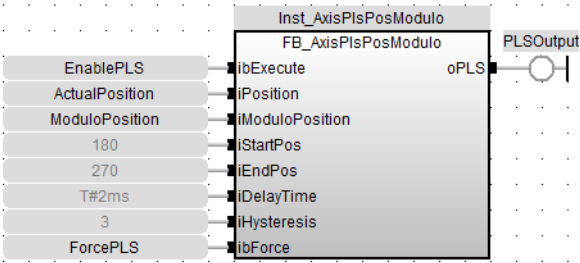
2.3.2.0.1 Timing



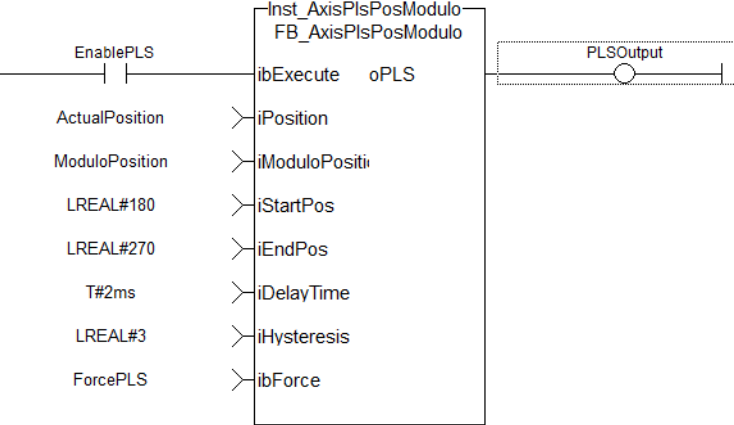
2.3.2.0.2 Hysteresis



FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//PLSOutput is True when position input is between 180 and 270 with a T#2ms delay
//Can also force the output to be true with ForceOuput variable
//Hysteresis is on for 3 user units in case direction changes around start
```

```
point
```

```
Inst_FB_AxisPlsPosModulo( EnablePLS, ActualPosition, ModuloPosition, 180,  
270, T#2ms, 3, ForcePLS );
```

```
PLSOutput := Inst_FB_AxisPlsPosModulo.oPLS;
```

2.3.3 FB_AxisPlsPosNoModulo

PLCopen ✓

Pipe Network ✓

 **Function Block** - Used for any position of a non-modulo axis in both directions.

- The Boolean output **oPLS** is set to:
 - TRUE if the position has crossed the start position.
 - FALSE if the position has crossed the end position.
- The function block has the possibility to compensate a delay time of the connected device (e.g., glue nozzles).
- It is possible to define a hysteresis for switching on and off of the PLS.

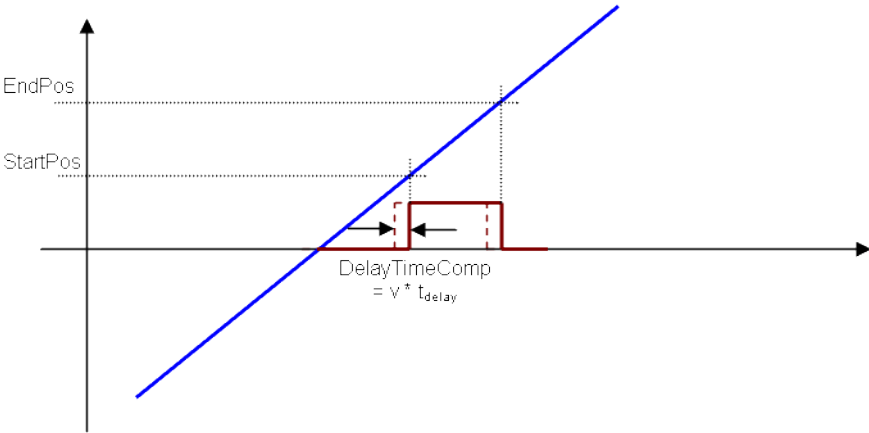
Inputs

Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL	FALSE, TRUE	N/A	No default	Enable PLS.
iPosition	LREAL	No range	User units	No default	Any position of a non-modulo axis.
iStartPos	LREAL	No range	User units	No default	Start position of PLS.
iEndPos	LREAL	No range	User units	No default	End position of PLS.
iDelayTime	TIME	0ms - 24hr	TIME	No default	Delay time for compensation.
iHysteresis	LREAL	No range	User units	No default	Hysteresis.
ibForce	BOOL	FALSE, TRUE	N/A	No default	Force PLS.

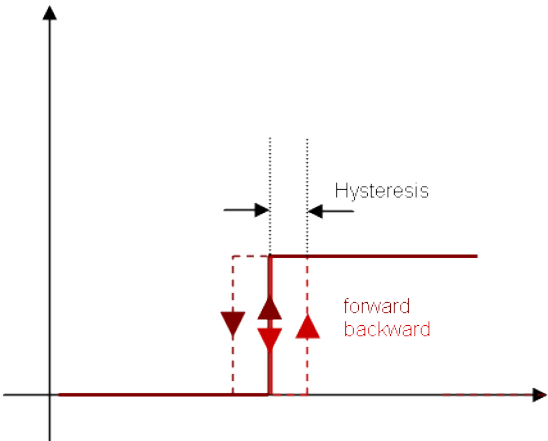
Outputs

Output	Data Type	Range	Unit	Description
oPLS	BOOL	FALSE, TRUE	N/A	Position limit switch.

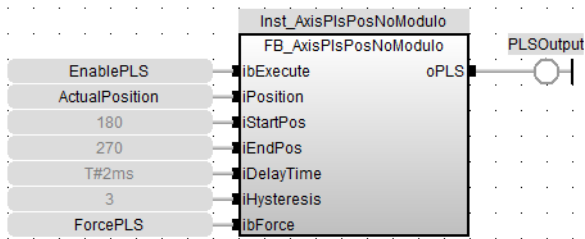
2.3.3.0.1 Timing



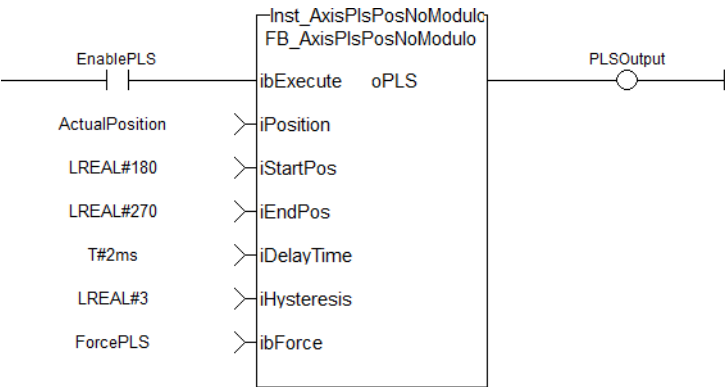
2.3.3.0.2 Hysteresis



FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//PLSOutput is True when position input is between 180 and 270 with a T#2ms
delay
//Can also force the output to be true with ForceOutput variable
//Hysteresis is on for 3 user units in case direction changes around start
point
```



```
Inst_FB_AxisPlsPosNoModulo( EnablePLS, ActualPosition, 180, 270, T#2ms, 3,  
ForcePLS );  
  
PLSOutput := Inst_FB_AxisPlsPosNoModulo.oPLS;
```

2.3.4 FB_Cylinder

PLCopen ✓

Pipe Network ✓



Function Block - Used to control a cylinder and the Limit Switches.

- There are two inputs, **iLnA** and **iLnB**, to set the direction of the movement and the belonging LimSwitches iLsA and iLsB.
- If iLnA is set to TRUE, the output oDirA is set to TRUE.
 - After a time value defined by CtrlTime, the iLsA has to become TRUE otherwise a fault FaultLsA appears.
 - Just as in direction B.
- If both iLsA and iLsB are TRUE, then a Fault depending of the output is set.
- If both iLnA and iLnB are given (e.g., to stop the cylinder movement), no limit switch is controlled.
- All faults can be reset by input iResetFault.

2.3.4.0.1 Usage

- The signal flow is valid for both directions (A and B)
- If oDirA AND oDirB are active there is no Fault Control.
- The Fault can be reset by iRestFault = True.

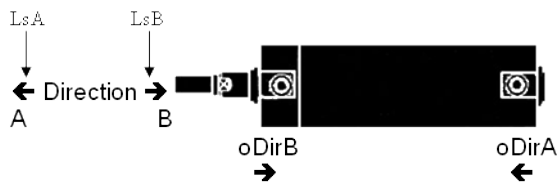


Figure 3-1: FB_Cylinder Usage 1



Figure 3-2: FB_Cylinder Usage 2

Inputs

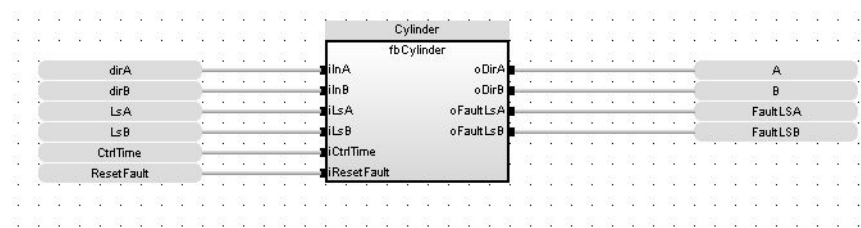
Input	Data Type	Range	Unit	Default	Description
iLnA	BOOL	FALSE, TRUE	N/A	No default	Set direction A.
iLnB	BOOL	FALSE, TRUE	N/A	No default	Set direction B.
iLsA	BOOL	FALSE, TRUE	N/A	No default	Limit Switch at end of direction A.

Input	Data Type	Range	Unit	Default	Description
iLsB	BOOL	FALSE, TRUE	N/A	No default	Limit Switch at end of direction B.
iCtrlTime	TIME	0ms - 24hr	TIME	No default	Max time until LimitSwitch has to be reached.
iResetFault	BOOL	FALSE, TRUE	N/A	No default	Reset Fault (is set to FALSE by UDFB!).

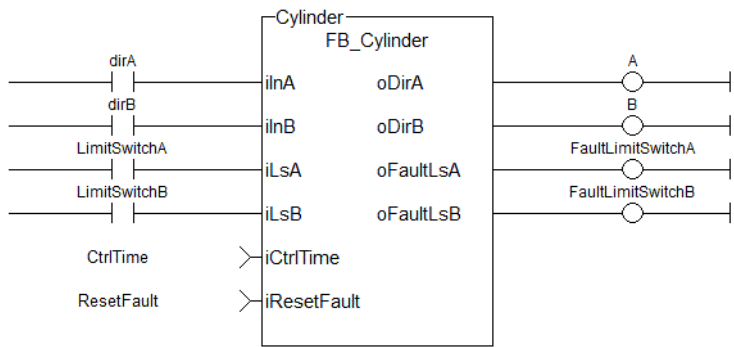
Outputs

Output	Data Type	Range	Unit	Description
oDirA	BOOL	FALSE, TRUE	N/A	Direction A.
oDirB	BOOL	FALSE, TRUE	N/A	Direction B.
oFaultLsA	BOOL	FALSE, TRUE	N/A	Fault of LimitSwitch at the end of direction A.
oFaultLsB	BOOL	FALSE, TRUE	N/A	Fault of LimitSwitch at the end of direction B.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Electric Cylinder with limit switch controls
Inst_FB_Cylinder( dirA, dirB, LimitSwitchA, LimitSwitchB, CtrlTime,
```

```
ResetFault );  
A := Inst_FB_Cylinder.oDirA;  
B := Inst_FB_Cylinder.oDirB;  
FaultLimitSwitchA := Inst_FB_Cylinder.oFaultLsA;  
FaultLimitSwitchB := Inst_FB_Cylinder.oFaultLsB;
```

2.3.5 FB_S700FltRpt

PLCopen ✓

Pipe Network ✓



Function Block - Outputs S700 drive fault Information.

- The **oFAULT** output turns TRUE when the selected drive goes into a fault state.
- This function block outputs the:
 - Total number of faults in the drive fault history variable. (FLTHIST)
 - Fault number and message for the last three drive faults.
- Each fault has two outputs: the fault number and a fault message.
 - The fault number is the same number reported on the display of the drive.
 - The fault message provides a short description of the fault.
 - Example: If the first fault is a feedback error with a F04 is displayed on the front of the drive, the output of this FB are:
 - **oFirstFaultNumber** = 04
 - **oFirstFaultMessage** = Feedback Error
 - **iResetfaultHistory** resets the faults reported by the FB.
 - **oDriveNotUsed** outputs 1 (True) if the axis is configured to Simulated in the ProjectEtherCAT setup screen.
- This function block can be used with either the PLCopen or Pipe Network Motion engines.

2.3.5.0.1 Usage

- Provide drive fault information the application program uses to determine next steps.
 - Example: Perform a machine-controlled stop or perform an immediate disable of the servo drives.
- In the application program, sends output fault information from this UDFB to the HMI for review by the machine operator.

Inputs

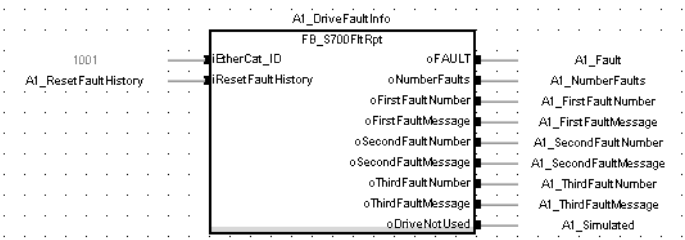
Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables the Kollmorgen UDFB. Used in the FFLD editor only.
iEtherCat_ID	INT	-32768 to +32767	N/A	No default	EtherCAT address of AKD Drive. Example: 1001 or AKD_1.
iRstFltHist	BOOL	FALSE, TRUE	N/A	No default	When input is TRUE, clears all Faults saved to drives history.

Outputs

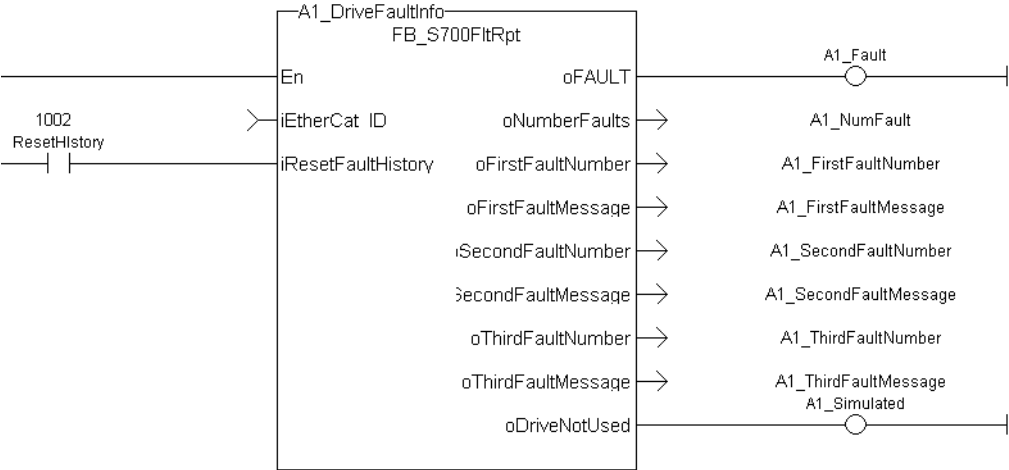
Output	Data Type	Range	Unit	Description
oFAULT	BOOL	FALSE, TRUE	N/A	TRUE if selected drive currently has a Fault.
oNumberFaults	DINT	0 to 10	N/A	Number of faults saved in the drive's history.
oFirstFaultNumber	DINT	100 to 999	N/A	Three-digit fault identifier.
oFirstFaultMessage	STRING	No range	N/A	Description of the fault.
oSecondFaultNumber	DINT	100 to 999	N/A	Three-digit fault identifier.

Output	Data Type	Range	Unit	Description
oSecondFaultMessage	STRING	No range	N/A	Description of the fault.
oThirdFaultNumber	DINT	100 to 999	N/A	Three-digit fault identifier.
oThirdFaultMessage	STRING	No range	N/A	Description of the fault.
oDriveNotUsed	BOOL	FALSE, TRUE	N/A	Is this drive: • 0 (zero) = Real. • 1 = Simulated.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Execute the Function Block
A1_FaultReporting (1001, 0);

//Read Function Block Outputs
A1_Fault:= A1_FaultReporting.oFault;
A1_NumFault:= A1_FaultReporting.oNumberFaults;
A1_FirstFaultNumber:= A1_FaultReporting.oFirstFaultNumber;
```

```
A1_FirstFaultMessage:= A1_FaultReporting.oFirstFaultMessage;  
A1_SecondFaultNumber:= A1_FaultReporting.oSecondFaultNumber;  
A1_SecondFaultMessage:= A1_FaultReporting.oSecondFaultMessage;  
A1_ThirdFaultNumber:= A1_FaultReporting.oThirdFaultNumber;  
A1_ThirdFaultMessage:= A1_FaultReporting.oThirdFaultMessage;  
A1_Simulated:= A1_FaultReporting.oDriveNotUsed;
```

NOTE

A1_FaultReporting is an instance of the **FB_S700FltRpt** function block.

See Also

- "MC_ReadStatus" (→ p. 652) (PLCopen Motion Engine)
- "MLAxisStatus" (→ p. 386) (Pipe Network Motion Engine)

2.4 UDFB - Motion / Pipe Network

These are the Kollmorgen UDFB Motion/Pipe Network function and function blocks.

Name	Description
MLFB_DriveFault	Returns the fault status, number, and description of the requested axis mapped to a Kollmorgen drive.
MLFB_ECATRestart	Reinitializes the EtherCAT network and the motion engine.
MLFB_HomeFindHomeFastInput	Fast homing to a home switch.
MLFB_HomeFindHomeFastInputModulo	Fast homing to a home switch: Modulo mode.
MLFB_HomeFindHomeInput	Homing to a home switch.
MLFB_HomeFindHomeInputThenZeroAngle	Homing to a home switch plus zero angle.
MLFB_HomeFindLimitFastInput	Fast homing to a limit switch.
MLFB_HomeFindLimitFastInputModulo	Fast homing to a limit switch: Modulo mode.
MLFB_HomeFindLimitInput	Homing to a limit switch.
MLFB_HomeFindLimitInputThenZeroAngle	Homing to a limit switch plus zero angle.
MLFB_HomeFindZeroAngle	Homing to zero angle reference.
MLFB_HomeMoveUntilPosErrExceeded	Homing to the position error is exceeded.
MLFB_HomeMoveUntilPosErrExceededThenZeroAngle	Homing to the position error is exceeded plus zero angle.
MLFB_HomeUsingCurrentPosition	Homing using current position.
MLFB_Jog	Define to jog an axis in the selected direction at a defined speed.
MLFB_PlsPosFw	Forward position range indicator.
MLFB_PlsPosFwBw	Forward/backward position range indicator.
MLFB_PlsTimeFw	Forward/backward Position/Time range indicator.

2.4.1 MLFB_DriveFault



 **Function Block** - Returns the fault status, number, and description of the requested axis mapped to a Kollmorgen drive.

ⓘIMPORTANT

This function block requires "FB_S700FitRpt" (→ p. 134), "MCFB_AKDFault" (→ p. 196), and "MCFB_AKDFaultLookup" (→ p. 198) subprograms imported to project to compile and function.

- The FAULT output returns TRUE when the selected drive goes into a fault state.
- The fault number and description depend on the drive type mapped to the axis.
 - If the drive is an:
 - AKD or AKD2G, the fault number is the same number as reported on the display of the drive.
 - AKT2G Stepper, the fault number represents the drive status word.
 - This word is a bitmask that represents the various error conditions.

2.4.1.0.1 Usage

- Provide drive fault information the application program uses to determine next steps.
 - Example: Perform a machine-controlled stop or perform an immediate disable of the servo drives.
- In the application program, sends output fault information from this UDFB to the HMI for review by the machine operator.

Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables the Kollmorgen UDFB. Used in the FFLD editor only.
AxisID	DINT	-2147483648 to 2147483647	N/A	No default	ID of the axis block of the Pipe Network.

Outputs

Output	Data Type	Range	Unit	Description
FAULT	BOOL	FALSE, TRUE	N/A	TRUE if the selected drive currently has a Fault.

Output	Data Type	Range	Unit	Description
FaultNumber	DINT	No range	N/A	If the axis is: - S300/S700 - Three-digit fault identifier. - See S300 & S700 Errors and Warnings on KDN for a list of fault codes. - AKD - Three-digit fault identifier. - See Fault and Warning Messages for a list of fault codes. - AKD2G - Four-digit fault identifier. - See Faults and Warnings View for a list of fault codes. - AKT2G Stepper - Drive Status word (bitmask). - See "Bitmask Causes" (→ p. 139).
FaultDescription	STRING	N/A	N/A	Description of the fault.

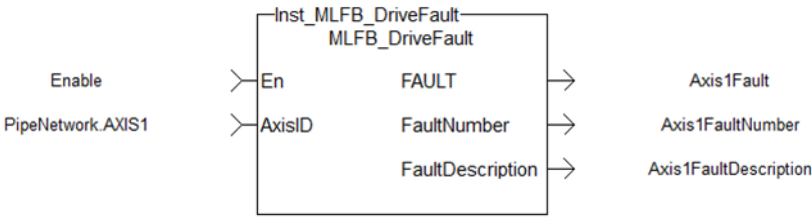
2.4.1.0.2 Bitmask Causes

Bit	Description	Cause
0	Saturated	Drive stage operates with maximum duty cycle.
1	Over temperature.	Internal temperature is higher than 80 °C.
2	Torque overload.	Motor current is higher than the rated current.
3	Under voltage.	Motor supply voltage is either: 20% lower than the configured nominal voltage (warning). - Less than 8V.
4	Over voltage.	Motor supply voltage is 10% higher than the configured nominal voltage.
5	Short circuit A.	Short circuit in motor coil A.
6	Short circuit B.	Short circuit in motor coil B.
7	No control power.	Control voltage at the power contacts is less than 12V.
8	Misc. error.	Either: - The terminal initialization failed. - Supply voltage is less than 8V. - Internal terminal temperature is higher than 100 °C.
9	Configuration error.	CoE change has not yet been adopted into the current configuration.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Execute and Read the Function Block
Inst_MLFB_DriveFault(PipeNetwork.AXIS1);
Axis1Fault := Inst_MLFB_DriveFault.FAULT;
Axis1FaultNumber := Inst_MLFB_DriveFault.FaultNumber;
Axis1FaultDescription := Inst_MLFB_DriveFault.FaultDescription;
```

See Also

- "FB_S700FltRpt" (→ p. 134)
- "MCFB_AKDFaultLookup" (→ p. 198)

2.4.2 MLFB_ECATRestart



 **Function Block** - Reinitializes the EtherCAT network and the motion engine.

- This function block clears motion engine errors, motion bus driver errors and EtherCAT network errors before reinitializing the motion engine, if requested to do so.
- This UDFB allows the EtherCAT and motion engines to restart without having to restart the entire project.

2.4.2.0.1 Examples

- EtherCAT network wire is replaced or accidentally disconnected.
- Axis setup parameters defined by CreateAxis and/or InitAxis function need to be changed while the application is running.

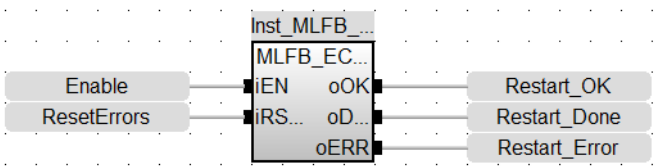
Inputs

Input	Data Type	Range	Unit	Default	Description
iEN	BOOL	FALSE, TRUE	N/A	No default	Enables the Kollmorgen UDFB. Used in the FFLD editor only.
IRSTERR	BOOL	1 to 256	0 to 1	No default	Clears the motion engine and EtherCAT network errors in case of any faults.

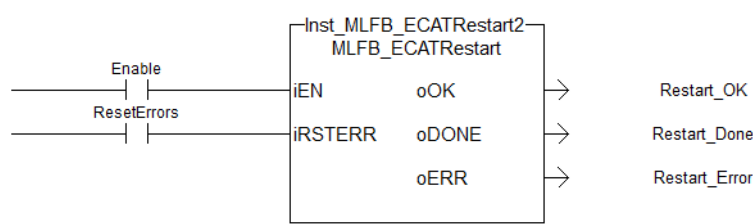
Outputs

Output	Data Type	Range	Unit	Description
oOK	BOOL	FALSE, TRUE	N/A	Function block activated status.
oDONE	BOOL	FALSE, TRUE	N/A	Execution complete.
oERR	BOOL	FALSE, TRUE	N/A	TRUE if the system initialization fails.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MLFB_ECATRstart( Restart, ResetErrors );  
IF Inst_MLFB_ECATRstart.oDONE THEN  
    RestartComplte:=1;  
End_IF;
```

See Also

- "ClearCtrlErrors" (→ p. 901)
- "MLMotionInit" (→ p. 867)
- "MLMotionRstErr" (→ p. 870)
- "MLMotionStart" (→ p. 872)

2.4.3 Fast Homing - UDFB - Motion/Pipe Network

These are the UDFB Motion/Pipe Network Fast Homing function blocks.

Name	Description
"MLFB_HomeFindHomeFastInput" (→ p. 144)	Performs a single-axis home to a limit switch connected to a High Speed Input.
"MLFB_HomeFindHomeFastInputModulo" (→ p. 148)	Performs a single-axis home to a limit switch connected to a High Speed Input.
"MLFB_HomeFindLimitFastInput" (→ p. 152)	Performs a single-axis home to a limit switch connected to a High Speed Input.
"MLFB_HomeFindLimitFastInputModulo" (→ p. 156)	Performs a single-axis home to a limit switch connected to a High Speed Input.

2.4.3.1 MLFB_HomeFindHomeFastInput

Pipe Network



Function Block - Fast homing to a home switch.

- Performs a single-axis home to a limit switch connected to a High Speed Input.
- The motor starts to move according to the direction setting.
 - The home position has been found as soon as the fast input selected is triggered on the edge selected.
- An absolute move is made to the triggered position and the position value is set.
 - The hardware limit switches are monitored during the homing procedure.
- The drive behaves this way in case a hardware limit switch is active before the home-switch has been activated:
The motor changes the direction until the home switch is crossed.

2.4.3.1.1.1 Usage

- This procedure performs a homing function searching for a sensor using only High Speed Input Switches.
 - A High Speed Limit Switch has 1 Off (or On) area.
- Home is commanded by the user in the designated homing direction at the selected or programmed Velocity.
- If **LimitSwitch** is found On with a rising Execute, the process is started in the opposite direction as specified.
- If **LimitSwitch** is found Off Edge (released), the process is restarted again in original direction. This can be On, depending on **LimitSwitchMode** setting.
 - This ensures the end conditions are always the same.
- The Timeout can cause an error if exceeded.

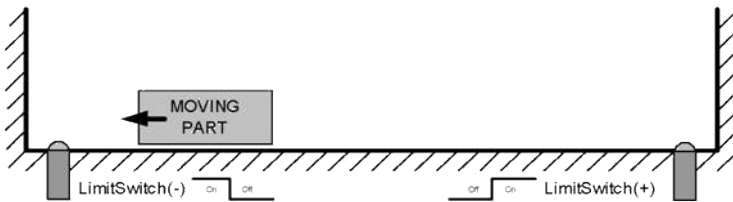


Figure 3-3: MLFB_HomeFindHomeFastInput Usage 1

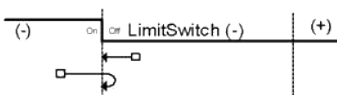


Figure 3-4: MLFB_HomeFindHomeFastInput Usage 2

Inputs

Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL	FALSE, TRUE	N/A	No default	Request the homing step procedure at the rising edge.
iAxisID	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
iPosition	LREAL	No range	User units	No default	Offset position applied after home switch is found.

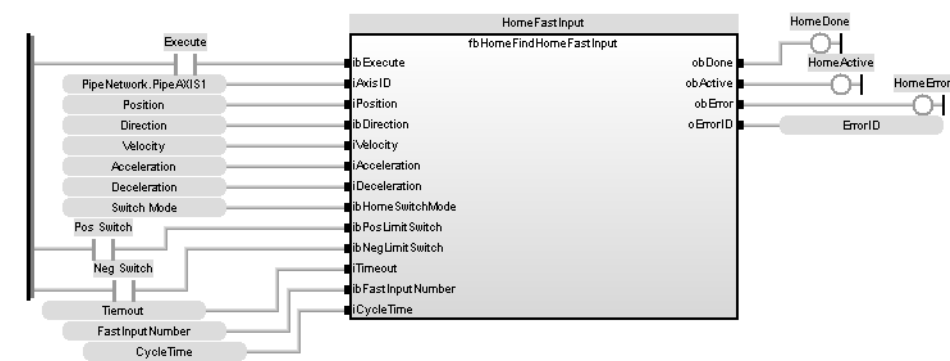
Input	Data Type	Range	Unit	Default	Description
ibDirection	BOOL	0 to 1	N/A	No default	Defines the axis homing direction. 0 (zero) = Clockwise rotation. 1 = Counterclockwise rotation.
iVelocity	LREAL	No range	User units / sec	No default	Commanded velocity for the homing move.
iAcceleration	LREAL	No range	User units / sec ²	No default	Commanded acceleration for the homing move.
iDeceleration	LREAL	No range	User units / sec ²	No default	Commanded deceleration for the homing move.
ibHomeSwitchMode	BOOL	0 to 1	N/A	No default	Limit switch state to complete homing. 0 (zero) = Rising edge of switch. 1 = Falling edge of switch.
ibPosLimitSwitch	BOOL	0 to 1	N/A	No default	The positive direction limit switch input I/O point.
ibNegLimitSwitch	BOOL	0 to 1	N/A	No default	The negative direction limit switch input I/O point.
iTimeout	TIME	No range	Sec	No default	Maximum time for homing move to complete. - If exceeded, the homing procedure errors out. 0 (zero) = no time limit.
ibFastInputNumber	BOOL	0 to 1	N/A	No default	Limit switch state to complete homing. 0 (zero) = Fast Input Number 1. 1 = Fast Input Number 2.
iCycleTime	LREAL	No range	Microseconds	No default	EtherCAT cycle time. Either 250, 500, or 1000.

Outputs

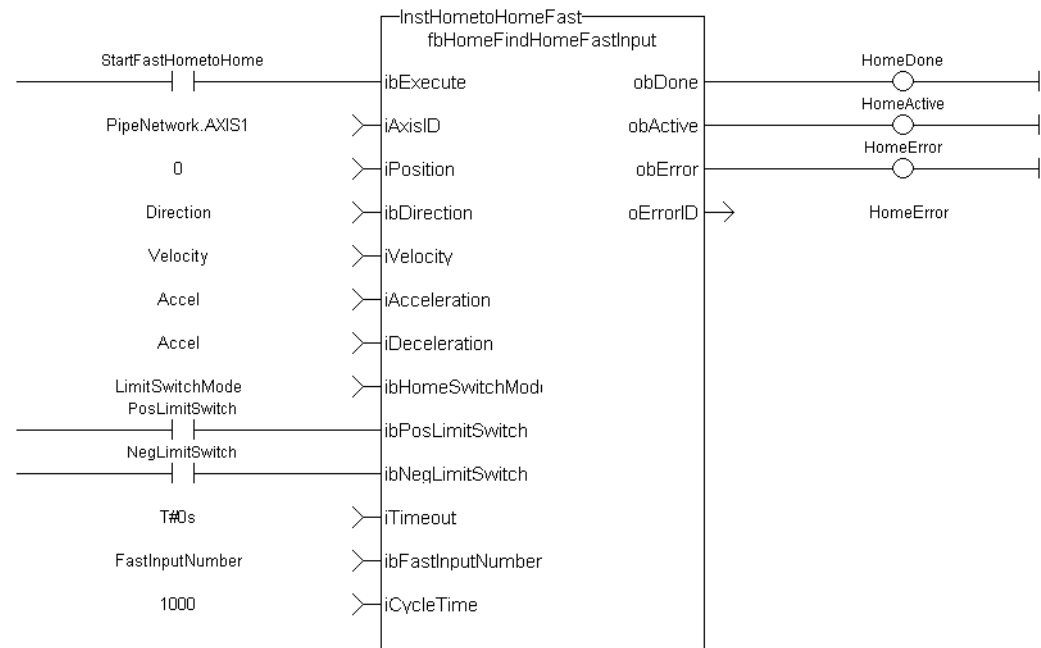
Output	Data Type	Range	Unit	Description
obDone	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.
obActive	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.
obError	BOOL	FALSE, TRUE	N/A	Indicates either: - An invalid input was specified. - The move was terminated due to an error.

Output	Data Type	Range	Unit	Description
oErrorID	DINT		N/A	Indicates the error if Error output is TRUE. Error identifier:
ErrorID	Description			
1	Axis in error.			
2	Axis is not enabled.			
3	Timeout exceeded or expired.			
4	SDO read / write error.			
5	Input parameter out of range.			

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```

Direction:= 0;
Position:=1000;
Velocity:=1000;
Acceleration:=10000;
Deceleration:=10000;
SwitchMode:=0;
Timeout:=T#100;
FastInputNumber:=0;
CycleTime:=1000;

inst_fbHomeFindHomeFastInput(True, Axis1, Position, Direction, Velocity,
Acceleration, Deceleration, HomeSwitchMode, PosLimitSwitch, NegLimitSwitch,
Timeout, FastInputNumber, CycleTime);

HomeComplete :=inst_fbHomeFindHomeFastInput.Done;
HomeActive :=inst_fbHomeFindHomeFastInput.Active;
HomeError :=inst_fbHomeFindHomeFastInput.Error;
HomeErrorID :=inst_fbHomeFindHomeFastInput.ErrorID;

(* PosLimitSwitch and NegLimitSwtch are declared I/O points *)

```

See Also

- "MLFB_HomeFindHomeFastInputModulo" (→ p. 148)
- "MLFB_HomeFindLimitFastInput" (→ p. 152)
- "MLFB_HomeFindLimitFastInputModulo" (→ p. 156)

2.4.3.2 MLFB_HomeFindHomeFastInputModulo

Pipe Network



Function Block - Fast homing to a home switch: Modulo mode.

- Performs a single-axis home to a limit switch connected to a High Speed Input.
- This function is used when the axis is set-up in Modulo mode.
- The motor starts to move according to the direction setting.
 - The home position has been found as soon as the fast input selected is triggered on the edge selected.
- An absolute move is made to the triggered position and the position value is set.
 - The hardware limit switches are monitored during the homing procedure.
- The drive behaves this way in case a hardware limit switch is active before the home-switch has been activated:
The motor changes the direction until the home switch is crossed.

2.4.3.2.1.1 Usage

- This procedure performs a homing function searching for a sensor using only High Speed Input Switches.
 - A High Speed Limit Switch has 1 Off (or On) area.
- Home is commanded by the user in the designated homing direction at the selected or programmed Velocity.
- If **LimitSwitch** is found On with a rising Execute, the process is started in the opposite direction as specified.
- If **LimitSwitch** is found Off Edge (released), the process is restarted again in original direction. This can be On, depending on **LimitSwitchMode** setting.
 - This ensures the end conditions are always the same.
- The Timeout can cause an error if exceeded.

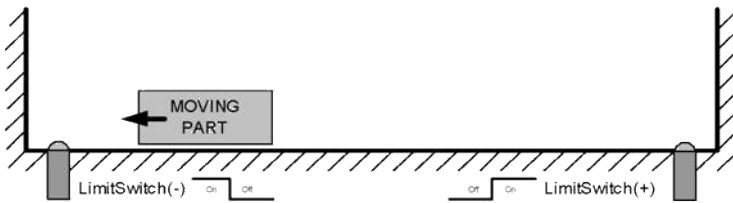


Figure 3-5: MLFB_HomeFindHomeFastInputModulo Usage 1

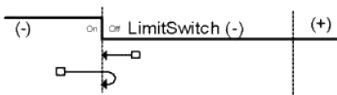


Figure 3-6: MLFB_HomeFindHomeFastInputModulo Usage 2

Inputs

Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL	0 to 1	N/A	No default	Request the homing step procedure at the rising edge.
iAxisID	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
iPosition	LREAL	No range	User units	No default	Offset position applied after home switch is found.

Input	Data Type	Range	Unit	Default	Description
ibDirection	BOOL	0 to 1	N/A	No default	Defines the axis homing direction. 0 (zero) = Clockwise rotation. 1 = Counterclockwise rotation.
iVelocity	LREAL	No range	User units / sec	No default	Commanded velocity for the homing move.
iAcceleration	LREAL	No range	User units / sec ²	No default	Commanded acceleration for the homing move.
iDeceleration	LREAL	No range	User units / sec ²	No default	Commanded deceleration for the homing move.
ibLimitSwitchMode	BOOL	0 to 1	N/A	No default	Limit switch state to complete homing. 0 (zero) = Rising edge of switch. 1 = Falling edge of switch.
ibPosLimitSwitch	BOOL	0 to 1	N/A	No default	The positive direction limit switch input I/O point.
ibNegLimitSwitch	BOOL	0 to 1	N/A	No default	The negative direction limit switch input I/O point.
iTimeout	TIME	No range	Sec	No default	Maximum time for homing move to complete. - If exceeded, the homing procedure errors out. 0 (zero) = no time limit.
ibFastInputNumber	BOOL	0 to 1	N/A	No default	Limit switch state to complete homing. 0 (zero) = Fast Input Number 1. 1 = Fast Input Number 2.
iCycleTime	LREAL	No range	Microseconds	No default	EtherCAT cycle time. Either 250, 500, or 1000.

Outputs

Output	Data Type	Range	Unit	Description
obDone	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.
obActive	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.
obError	BOOL	FALSE, TRUE	N/A	Indicates either: - An invalid input was specified. - The move was terminated due to an error.

Not available.

ST Language Example

```

Direction:= 0;
Position:=1000;
Velocity:=1000;
Acceleration:=10000;
Deceleration:=10000;
SwitchMode:=0;
Timeout:=T#100;
FastInputNumber:=0;
CycleTime:=1000;

inst_fbHomeFindHomeFastInputModuleulo(True, Axis1, Position, Direction,
Velocity, Acceleration, Deceleration, PosLimitSwitch, NegLimitSwitch,
Timeout, FastInputNumber, CycleTime);

HomeComplete :=inst_fbHomeFindHomeFastInputModuleulo.Done;
HomeActive :=inst_fbHomeFindHomeFastInputModuleulo.Active;
HomeError :=inst_fbHomeFindHomeFastInputModuleulo.Error;
HomeErrorID :=inst_fbHomeFindHomeFastInputModuleulo.ErrorID;

(* PosLimitSwitch and NegLimitSwtch are declared I/O points *)

```

See Also

- "MLFB_HomeFindHomeFastInput" (→ p. 144)
- "MLFB_HomeFindLimitFastInput" (→ p. 152)
- "MLFB_HomeFindLimitFastInputModuleulo" (→ p. 156)

2.4.3.3 MLFB_HomeFindLimitFastInput

Pipe Network



Function Block - Fast homing to a limit switch.

- Performs a single-axis home to a limit switch connected to a High Speed Input.
- The motor starts to move according to the direction setting.
 - The home position has been found as soon as the fast input selected is triggered on the edge selected.
- An absolute move is made to the triggered position and the position value is set.

2.4.3.3.1.1 Usage

- This procedure performs a homing function searching for a sensor using only High Speed Input Switches.
 - A High Speed Limit Switch has 1 Off (or On) area.
- Home is commanded by the user in the designated homing direction at the selected or programmed Velocity.
- The Timeout can cause an error if exceeded.

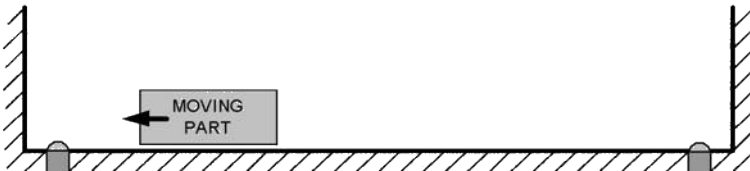


Figure 3-7: MLFB_HomeFindLimitFastInput Usage 1

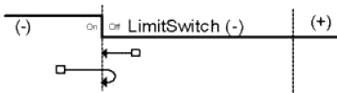


Figure 3-8: MLFB_HomeFindLimitFastInput Usage 2

Inputs

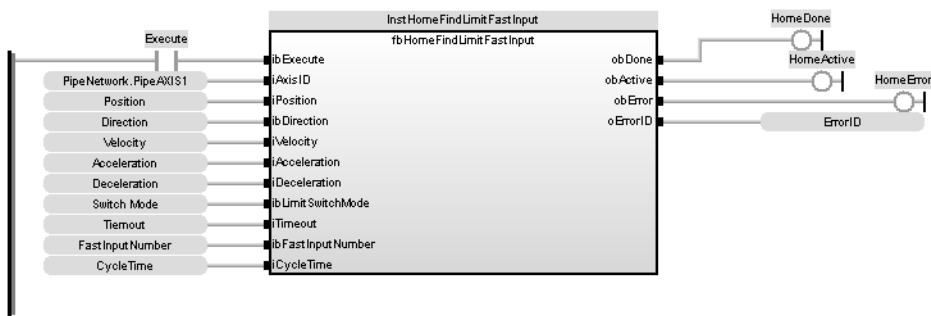
Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL	0 to 1	N/A	No default	Request the homing step procedure at the rising edge.
iAxisID	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
iPosition	LREAL	No range	User units	No default	Offset position applied after home switch is found.
ibDirection	BOOL	0 to 1	N/A	No default	Defines the axis homing direction. • 0 (zero) = Clockwise rotation. • 1 = Counterclockwise rotation.
iVelocity	LREAL	No range	User units / sec	No default	Commanded velocity for the homing move.
iAcceleration	LREAL	No range	User units / sec ²	No default	Commanded acceleration for the homing move.

Input	Data Type	Range	Unit	Default	Description
iDeceleration	LREAL	No range	User units / sec ²	No default	Commanded deceleration for the homing move.
ibLimitSwitchMode	BOOL	0 to 1	N/A	No default	Limit switch state to complete homing. 0 (zero) = Rising edge of switch. 1 = Falling edge of switch.
ibFastInputNumber	BOOL	0 to 1	N/A	No default	Limit switch state to complete homing. 0 (zero) = Fast Input Number 1. 1 = Fast Input Number 2.
iCycleTime	LREAL	No range	Microseconds	No default	EtherCAT cycle time. Either 250, 500, or 1000.

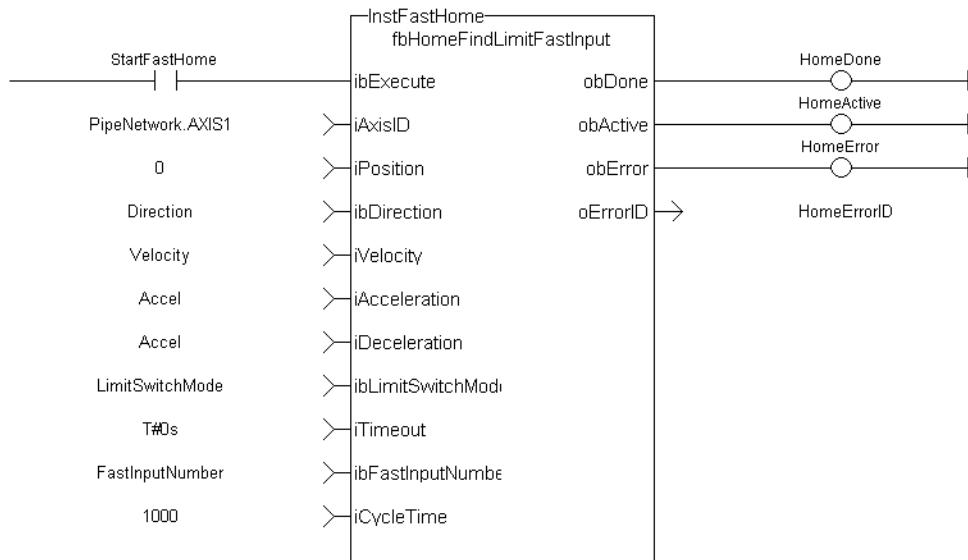
Outputs

Output	Data Type	Range	Unit	Description												
obDone	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.												
obActive	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.												
obError	BOOL	FALSE, TRUE	N/A	Indicates either: • An invalid input was specified. • The move was terminated due to an error.												
oErrorID	DINT		N/A	Indicates the error if Error output is TRUE. Error identifier: <table><tr><th>ErrorID</th><th>Description</th></tr><tr><td>1</td><td>Axis in error.</td></tr><tr><td>2</td><td>Axis is not enabled.</td></tr><tr><td>3</td><td>Timeout exceeded or expired.</td></tr><tr><td>4</td><td>SDO read / write error.</td></tr><tr><td>5</td><td>Input parameter out of range.</td></tr></table>	ErrorID	Description	1	Axis in error.	2	Axis is not enabled.	3	Timeout exceeded or expired.	4	SDO read / write error.	5	Input parameter out of range.
ErrorID	Description															
1	Axis in error.															
2	Axis is not enabled.															
3	Timeout exceeded or expired.															
4	SDO read / write error.															
5	Input parameter out of range.															

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Direction:= 0;
Position:=1000;
Velocity:=1000;
Acceleration:=10000;
Deceleration:=10000;
SwitchMode:=0;
Timeout:=T#100;
FastInputNumber:=0;
CycleTime:=1000;

inst_fbHomeFindLimitFastInput(True, Axis1, Position, Direction, Velocity,
Acceleration, Deceleration, LimitSwitchMode, Timeout, FastInputNumber,
CycleTime);
```

```
HomeComplete :=inst_fbHomeFindLimitFastInput.Done;  
HomeActive :=inst_fbHomeFindLimitFastInput.Active;  
HomeError :=inst_fbHomeFindLimitFastInput.Error;  
HomeErrorID :=inst_fbHomeFindLimitFastInput.ErrorID;
```

See Also

- "MLFB_HomeFindHomeFastInput" (→ p. 144)
- "MLFB_HomeFindHomeFastInputModulo" (→ p. 148)
- "MLFB_HomeFindLimitFastInputModulo" (→ p. 156)

2.4.3.4 MLFB_HomeFindLimitFastInputModulo

Pipe Network



Function Block - Fast homing to a limit switch: Modulo mode.

- Performs a single-axis home to a limit switch connected to a High Speed Input.
- This function is used when the axis is set-up in Modulo mode.
- The motor starts to move according to the direction setting.
 - The home position has been found as soon as the fast input selected is triggered on the edge selected.
- An absolute move is made to the triggered position and the position value is set.

2.4.3.4.1.1 Usage

- This procedure performs a homing function searching for a sensor using only High Speed Input Switches.
 - A High Speed Limit Switch has 1 Off (or On) area.
- Home is commanded by the user in the designated homing direction at the selected or programmed Velocity.
- The Timeout can cause an error if exceeded.

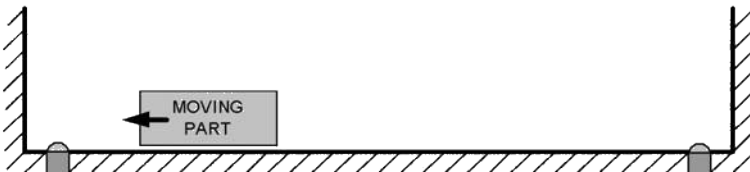


Figure 3-9: MLFB_HomeFindLimitFastInputModulo Usage 1

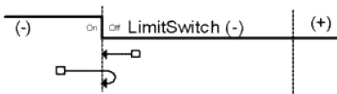


Figure 3-10: MLFB_HomeFindLimitFastInputModulo Usage 2

Inputs

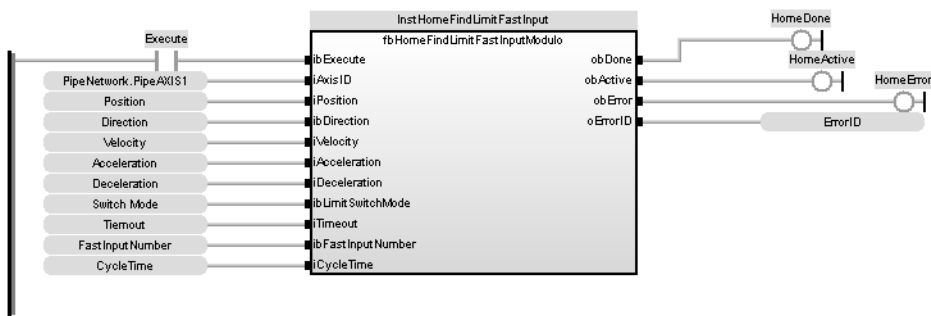
Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL	0 to 1	N/A	No default	Request the homing step procedure at the rising edge.
iAxisID	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
iPosition	LREAL	No range	User units	No default	Offset position applied after home switch is found.
ibDirection	BOOL	0 to 1	N/A	No default	Defines the axis homing direction. • 0 (zero) = Clockwise rotation. • 1 = Counterclockwise rotation.
iVelocity	LREAL	No range	User units / sec	No default	Commanded velocity for the homing move.
iAcceleration	LREAL	No range	User units / sec ²	No default	Commanded acceleration for the homing move.

Input	Data Type	Range	Unit	Default	Description
iDeceleration	LREAL	No range	User units / sec ²	No default	Commanded deceleration for the homing move.
ibLimitSwitchMode	BOOL	0 to 1	N/A	No default	Limit switch state to complete homing. 0 (zero) = Rising edge of switch. 1 = Falling edge of switch.
iTimeout	TIME	No range	Sec	No default	Maximum time for homing move to complete. - If exceeded, the homing procedure errors out. 0 (zero) = no time limit.
ibFastInputNumber	BOOL	0 to 1	N/A	No default	Limit switch state to complete homing. 0 (zero) = Fast Input Number 1. 1 = Fast Input Number 2.
iCycleTime	LREAL	No range	Microseconds	No default	EtherCAT cycle time. Either 250, 500, or 1000.

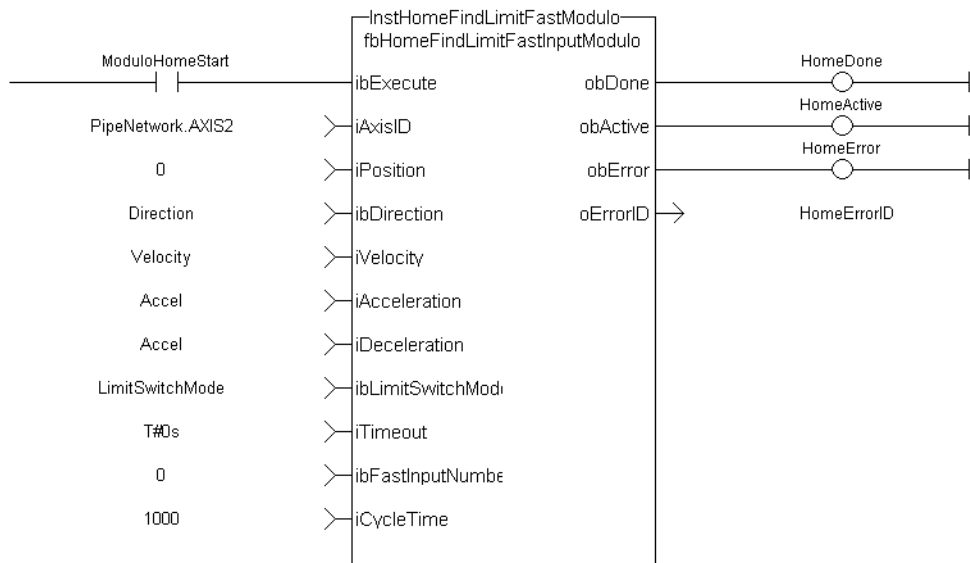
Outputs

Output	Data Type	Range	Unit	Description												
obDone	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.												
obActive	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.												
obError	BOOL	FALSE, TRUE	N/A	Indicates either: • An invalid input was specified. • The move was terminated due to an error.												
oErrorID	DINT		N/A	Indicates the error if Error output is TRUE. Error identifier: <table><tr><th>ErrorID</th><th>Description</th></tr><tr><td>1</td><td>Axis in error.</td></tr><tr><td>2</td><td>Axis is not enabled.</td></tr><tr><td>3</td><td>Timeout exceeded or expired.</td></tr><tr><td>4</td><td>SDO read / write error.</td></tr><tr><td>5</td><td>Input parameter out of range.</td></tr></table>	ErrorID	Description	1	Axis in error.	2	Axis is not enabled.	3	Timeout exceeded or expired.	4	SDO read / write error.	5	Input parameter out of range.
ErrorID	Description															
1	Axis in error.															
2	Axis is not enabled.															
3	Timeout exceeded or expired.															
4	SDO read / write error.															
5	Input parameter out of range.															

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Direction:= 0;
Position:=1000;
Velocity:=1000;
Acceleration:=10000;
Deceleration:=10000;
SwitchMode:=0;
Timeout:=T#100;
FastInputNumber:=0;
CycleTime:=1000;

inst_fbHomeFindLimitFastInputModule(True, Axis1, Position, Direction,
Velocity, Acceleration, Deceleration, LimitSwitchMode, Timeout,
FastInputNumber, CycleTime);
```

```
HomeComplete :=inst_fbHomeFindLimitFastInputModuleulo.Done;  
HomeActive :=inst_fbHomeFindLimitFastInputModuleulo.Active;  
HomeError :=inst_fbHomeFindLimitFastInputModuleulo.Error;  
HomeErrorID :=inst_fbHomeFindLimitFastInputModuleulo.ErrorID;
```

See Also

- "MLFB_HomeFindHomeFastInput" (→ p. 144)
- "MLFB_HomeFindHomeFastInputModuleulo" (→ p. 148)
- "MLFB_HomeFindLimitFastInput" (→ p. 152)

2.4.4 Homing - UDFB - Motion/Pipe Network

These are the UDFB Motion/Pipe Network Homing function blocks.

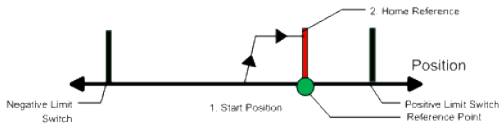
Name	Description
"MLFB_HomeFindHomeInput" (→ p. 161)	Homing to a home switch.
"MLFB_HomeFindHomeInputThenZeroAngle" (→ p. 164)	Homing to a home switch plus zero angle.
"MLFB_HomeFindLimitInput" (→ p. 167)	Homing to a limit switch.
"MLFB_HomeFindLimitInputThenZeroAngle" (→ p. 170)	Homing to a limit switch plus zero angle.
"MLFB_HomeFindZeroAngle" (→ p. 173)	Homing to zero angle reference.
"MLFB_HomeMoveUntilPosErrExceeded" (→ p. 176)	Homing to the position error is exceeded.
"MLFB_HomeMoveUntilPosErrExceededThenZeroAngle" (→ p. 179)	Homing to the position error is exceeded plus zero angle.
"MLFB_HomeUsingCurrentPosition" (→ p. 182)	Homing using current position.

2.4.4.1 MLFB_HomeFindHomeInput

Pipe Network



Function Block - Homing to a home switch.



- The motor starts to move according to the direction setting.
 - The home position has been found as soon as the home-switch becomes active during a motion in direction of the direction setting.
- The hardware limit switches are monitored during the homing procedure.
- The drive behaves this way in case a hardware limit switch is active before the home-switch has been activated:
The motor changes the direction until the home switch is crossed.
- The motor ramps down to 0 (zero) velocity and reverses direction again after crossing the home-switch.
 - The home-switch is now activated according to the direction setting and the home-position has been found.
- The command position of the drive is immediately set to the position value and the motor ramps down to 0 (zero) velocity.

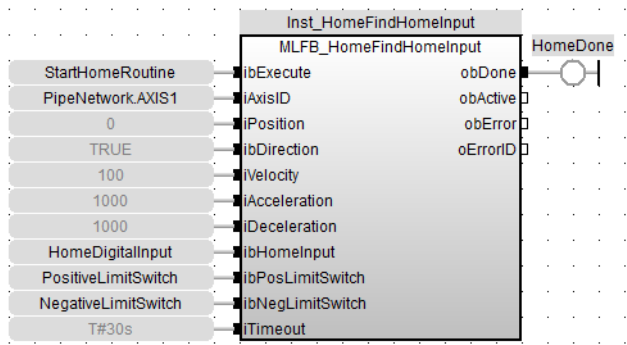
Inputs

Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL	FALSE, TRUE	N/A	No default	Start homing, edge-triggered.
iAxisID	DINT				ID of the axis block of the Pipe Network.
iPosition	LREAL				Reference position.
ibDirection	BOOL	0 to 1	N/A	No default	Defines the axis homing direction. 0 (zero) = Positive. 1 = Negative.
iVelocity	LREAL				Reference speed.
iAcceleration	LREAL				Reference acceleration.
iDeceleration	LREAL				Reference deceleration.
ibHomeInput	BOOL		N/A	No default	Home input, high-active.
ibPosLimitSwitch	BOOL		N/A	No default	Positive limit switch, high-active.
ibNegLimitSwitch	BOOL		N/A	No default	Negative limit switch, high-active.
iTimeout	TIME				Time monitoring (T#0ms: off).

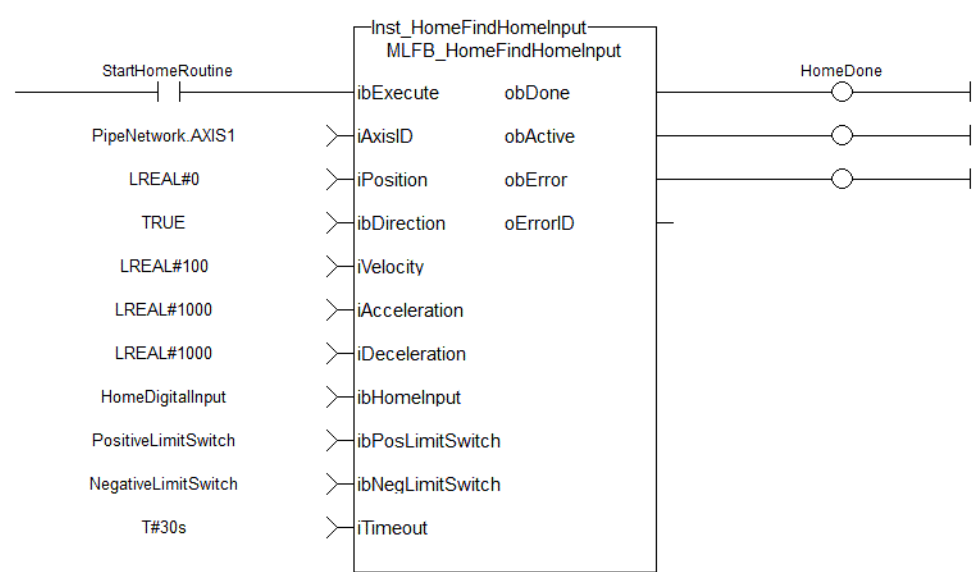
Outputs

Output	Data Type	Range	Unit	Description	
obDone	BOOL	FALSE, TRUE	N/A	Done bit.	
obActive	BOOL	FALSE, TRUE	N/A	Active bit.	
obError	BOOL	FALSE, TRUE	N/A	Error bit.	
oErrorID	DINT		N/A	Error identifier:	
				ErrorID	Description
				1	Axis in error.
				2	Axis is not enabled.
				3	Timeout exceeded or expired.
				4	SDO read / write error.
				5	Input parameter out of range.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Call homing function on Axis1 with preset velocity, accel, and decel values
//Start in negative direction, change if limit switch seen before home switch
//after seeing home switch, set axis position to zero
Inst_MLFB_HomeFindHomeInput( StartHomeRoutine,
    PipeNetwork.AXIS1,
    0,
    TRUE,
    100,
    1000,
    1000,
    HomeDigitalInput,
    PositiveLimitSwitch,
    NegativeLimitSwitch,
    T#30s );

HomeDone := Inst_MLFB_HomeFindHomeInput.obDone;
```

See Also

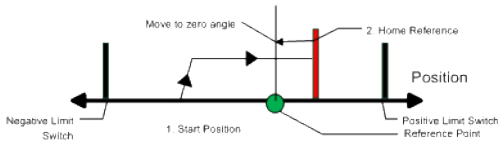
- "MLFB_HomeFindHomeInputThenZeroAngle" (→ p. 164)
- "MLFB_HomeFindLimitInput" (→ p. 167)
- "MLFB_HomeFindLimitInputThenZeroAngle" (→ p. 170)
- "MLFB_HomeFindZeroAngle" (→ p. 173)
- "MLFB_HomeMoveUntilPosErrExceeded" (→ p. 176)
- "MLFB_HomeMoveUntilPosErrExceededThenZeroAngle" (→ p. 179)
- "MLFB_HomeUsingCurrentPosition" (→ p. 182)

2.4.4.2 MLFB_HomeFindHomeInputThenZeroAngle

Pipe Network



Function Block - Homing to a home switch plus zero angle.



- Similar to the Find Home Limit method.
 - The find input home then find zero angle.
 - Mode follows the same steps, but upon completion of the move, it continues to move to find the zero angle reference of the motor.

Inputs

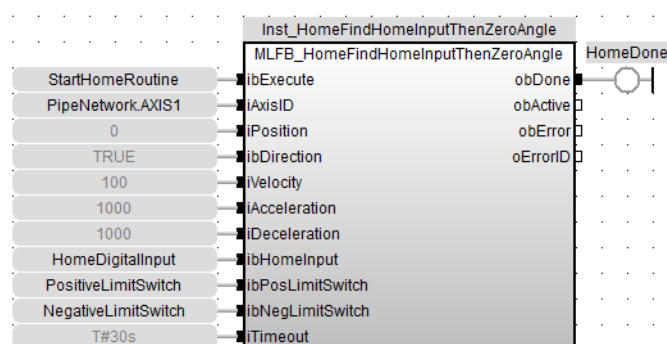
Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL	FALSE, TRUE	N/A	No default	Start homing, edge-triggered.
iAxisID	DINT				ID of the axis block of the Pipe Network.
iPosition	LREAL				Reference position.
ibDirection	BOOL	0 to 1	N/A	No default	Defines the axis homing direction. • 0 (zero) = Positive. • 1 = Negative.
iVelocity	LREAL				Reference speed.
iAcceleration	LREAL				Reference acceleration.
iDeceleration	LREAL				Reference deceleration.
ibHomeInput	BOOL		N/A	No default	Home input, high-active.
ibPosLimitSwitch	BOOL		N/A	No default	Positive limit switch, high-active.
ibNegLimitSwitch	BOOL		N/A	No default	Negative limit switch, high-active.
iTimeout	TIME				Time monitoring (T#0ms: off).

Outputs

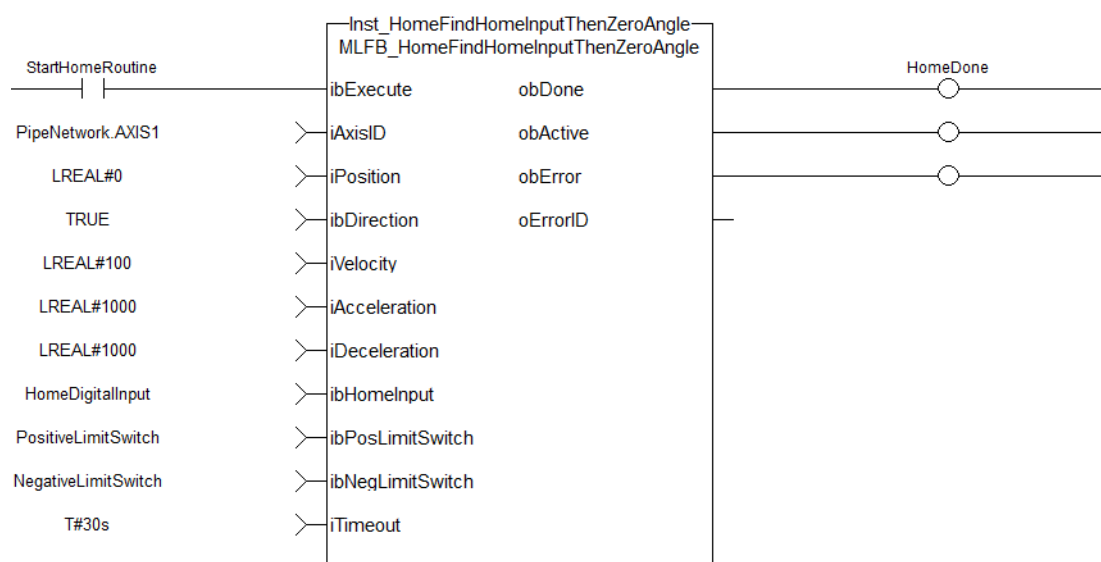
Output	Data Type	Range	Unit	Description
obDone	BOOL	FALSE, TRUE	N/A	Done bit.
obActive	BOOL	FALSE, TRUE	N/A	Active bit.
obError	BOOL	FALSE, TRUE	N/A	Error bit.

Output	Data Type	Range	Unit	Description	
oErrorID	DINT		N/A	Error identifier:	
				ErrorID	Description
				1	Axis in error.
				2	Axis is not enabled.
				3	Timeout exceeded or expired.
				4	SDO read / write error.
				5	Input parameter out of range.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```

//Call homing function on Axis1 with preset velocity, accel, and decel values
//Start in negative direction, change if limit switch seen before home switch
//after seeing home switch and moving to zero angle, set axis position to
zero
Inst_MLFB_HomeFindHomeInputThenZeroAngle( StartHomeRoutine,
    PipeNetwork.AXIS1,
    0,
    TRUE,
    100,
    1000,
    1000,
    HomeDigitalInput,
    PositiveLimitSwitch,
    NegativeLimitSwitch,
    T#30s );

HomeDone := Inst_MLFB_HomeFindHomeInputThenZeroAngle.obDone;

```

See Also

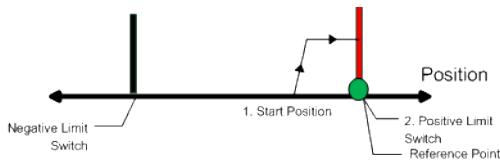
- "MLFB_HomeFindHomeInput" (→ p. 161)
- "MLFB_HomeFindLimitInput" (→ p. 167)
- "MLFB_HomeFindLimitInputThenZeroAngle" (→ p. 170)
- "MLFB_HomeFindZeroAngle" (→ p. 173)
- "MLFB_HomeMoveUntilPosErrExceeded" (→ p. 176)
- "MLFB_HomeMoveUntilPosErrExceededThenZeroAngle" (→ p. 179)
- "MLFB_HomeUsingCurrentPosition" (→ p. 182)

2.4.4.3 MLFB_HomeFindLimitInput

Pipe Network



Function Block - Homing to a limit switch.



- The find limit input mode moves to a limit input.
- This method is used if there is a positive or negative limit switch available to establish as a home reference point.

Inputs

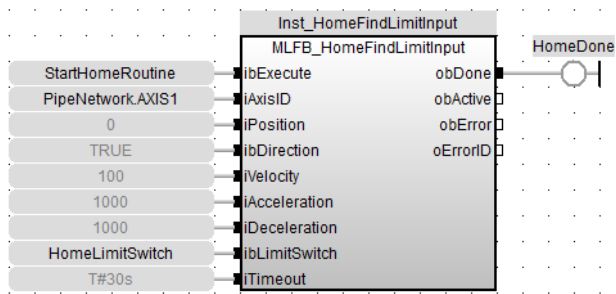
Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL	FALSE, TRUE	N/A	No default	Start homing, edge-triggered.
iAxisID	DINT	No range	N/A	No default	ID of the axis block of the Pipe Network.
iPosition	LREAL	No range	User units	No default	Reference position.
ibDirection	BOOL	FALSE, TRUE	N/A	No default	Defines the axis homing direction. • FALSE = Positive. • TRUE = Negative.
iVelocity	LREAL	iVelocity > 0	User units / sec	No default	Reference speed.
iAcceleration	LREAL	iAcceleration > 0	User units / sec ²	No default	Reference acceleration.
iDeceleration	LREAL	iDeceleration > 0	User units / sec ²	No default	Reference deceleration.
ibLimitSwitch	BOOL	FALSE, TRUE	N/A	No default	Positive or negative limit switch, high-active. Depends on the ibDirection.
iTimeout	TIME	No range	See TIME in Constant Expressions.	No default	Time monitoring (T#0ms: off).

Outputs

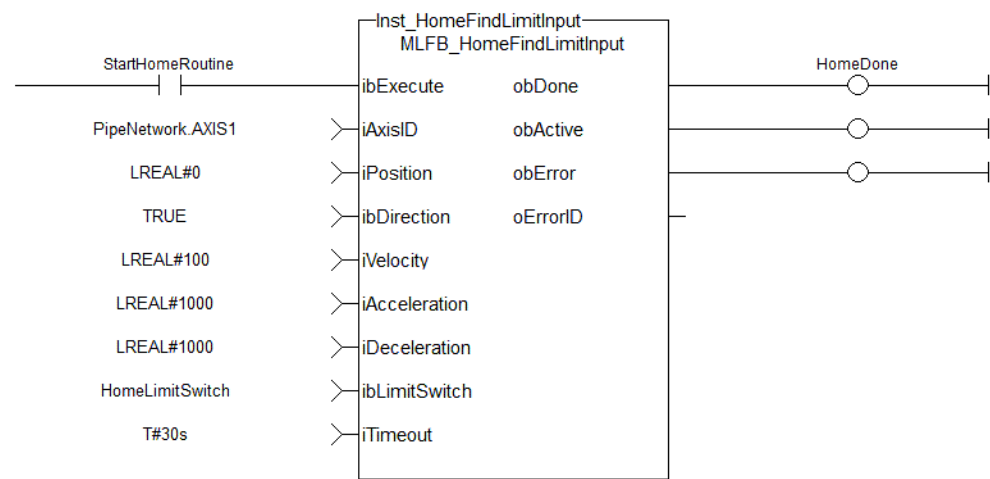
Output	Data Type	Range	Unit	Description
obDone	BOOL	FALSE, TRUE	N/A	Done bit.
obActive	BOOL	FALSE, TRUE	N/A	Active bit.

Output	Data Type	Range	Unit	Description												
obError	BOOL	FALSE, TRUE	N/A	Error bit.												
oErrorID	DINT	Enumerated	N/A	Error identifier:												
				<table><tr><th>ErrorID</th><th>Description</th></tr><tr><td>1</td><td>Axis in error.</td></tr><tr><td>2</td><td>Axis is not enabled.</td></tr><tr><td>3</td><td>Timeout exceeded or expired.</td></tr><tr><td>4</td><td>SDO read / write error.</td></tr><tr><td>5</td><td>Input parameter out of range.</td></tr></table>	ErrorID	Description	1	Axis in error.	2	Axis is not enabled.	3	Timeout exceeded or expired.	4	SDO read / write error.	5	Input parameter out of range.
ErrorID	Description															
1	Axis in error.															
2	Axis is not enabled.															
3	Timeout exceeded or expired.															
4	SDO read / write error.															
5	Input parameter out of range.															

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```

//Call homing function on Axis1 with preset velocity, accel, and decel values
//Start in negative direction and stop when axis hits limit switch or times
out
//after seeing limit switch, set axis position to zero
Inst_MLFB_HomeFindLimitInput( StartHomeRoutine,
    PipeNetwork.AXIS1,
    0,
    TRUE,
    100,
    1000,
    1000,
    HomeDigitalInput,
    T#30s );

HomeDone := Inst_MLFB_HomeFindLimitInput.obDone;

```

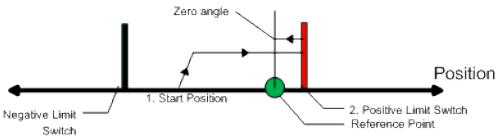
See Also

- "MLFB_HomeFindHomeInput" (→ p. 161)
- "MLFB_HomeFindHomeInputThenZeroAngle" (→ p. 164)
- "MLFB_HomeFindLimitInputThenZeroAngle" (→ p. 170)
- "MLFB_HomeFindZeroAngle" (→ p. 173)
- "MLFB_HomeMoveUntilPosErrExceeded" (→ p. 176)
- "MLFB_HomeMoveUntilPosErrExceededThenZeroAngle" (→ p. 179)
- "MLFB_HomeUsingCurrentPosition" (→ p. 182)

2.4.4.4 MLFB_HomeFindLimitInputThenZeroAngle



 **Function Block** - Homing to a limit switch plus zero angle.



- Similar to the Find Input Limit method, the find input limit then find zero angle.
- Mode follows the same steps, but upon completion of the move, it continues to move to find the zero angle reference of the motor.

Inputs

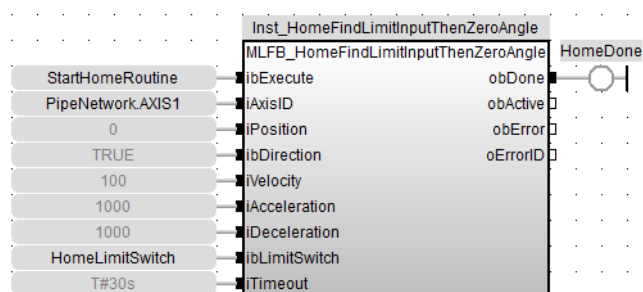
Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL				Start homing, edge-triggered.
iAxisID	BOOL				ID of the axis block of the Pipe Network.
iPosition	BOOL				Reference position.
ibDirection	BOOL				• 0 (zero) = Positive. • 1 = Negative.
iVelocity	BOOL				Reference speed.
iAcceleration	BOOL				Reference acceleration.
iDeceleration	BOOL				Reference deceleration.
ibLimitSwitch	BOOL				Positive or negative limit switch, high-active. Depends on the ibDirection.
iTimeout	BOOL				Time monitoring (T#0ms: off).

Outputs

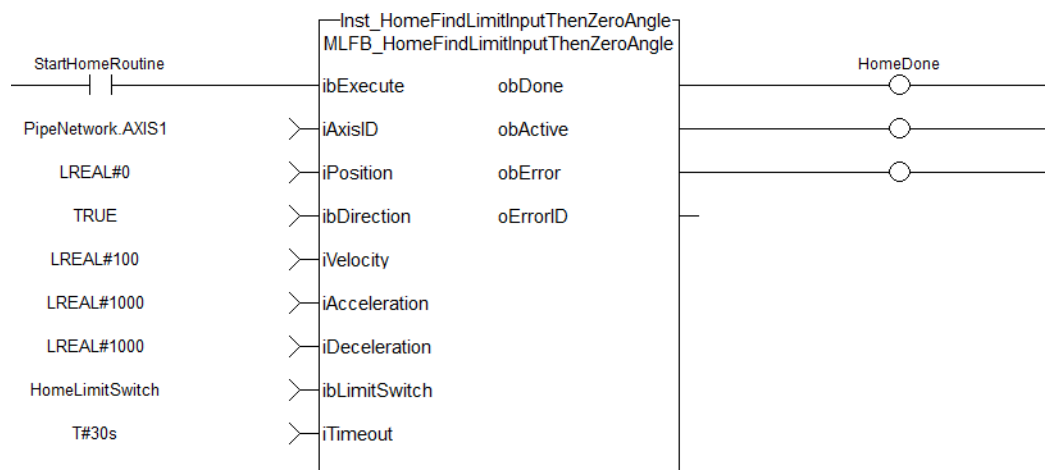
Output	Data Type	Range	Unit	Description
obDone	BOOL	FALSE, TRUE	N/A	Done bit.
obActive	BOOL	FALSE, TRUE	N/A	Active bit.
obError	BOOL	FALSE, TRUE	N/A	Error bit.

Output	Data Type	Range	Unit	Description	
oErrorID	DINT		N/A	Error identifier:	
				ErrorID	Description
				1	Axis in error.
				2	Axis is not enabled.
				3	Timeout exceeded or expired.
				4	SDO read / write error.
				5	Input parameter out of range.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Call homing function on Axis1 with preset velocity, accel, and decel values
//Start in negative direction and stop when axis hits limit switch or times
```

```

out
//after seeing limit switch, moves to zero angle and set axis position to
zero
Inst_MLFB_HomeFindLimitInputThenZeroAngle( StartHomeRoutine,
PipeNetwork.AXIS1,
0,
TRUE,
100,
1000,
1000,
HomeDigitalInput,
T#30s );

HomeDone := Inst_MLFB_HomeFindLimitInputThenZeroAngle.obDone;

```

See Also

- "MLFB_HomeFindHomeInput" (→ p. 161)
- "MLFB_HomeFindHomeInputThenZeroAngle" (→ p. 164)
- "MLFB_HomeFindLimitInput" (→ p. 167)
- "MLFB_HomeFindZeroAngle" (→ p. 173)
- "MLFB_HomeMoveUntilPosErrExceeded" (→ p. 176)
- "MLFB_HomeMoveUntilPosErrExceededThenZeroAngle" (→ p. 179)
- "MLFB_HomeUsingCurrentPosition" (→ p. 182)

2.4.4.5 MLFB_HomeFindZeroAngle



Function Block - Homing to zero angle reference.

NOTE

This function block is only applicable to motors with Resolver or SFD feedback.

1. Start Position.
2. End Position (Zero degrees).

Figure 3-11: Mode to find the zero angle reference of the motor.

Inputs

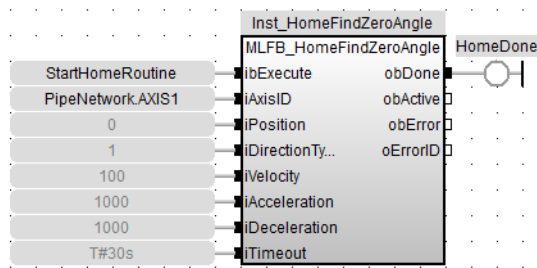
Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL	FALSE, TRUE	N/A	No default	Start homing, edge-triggered.
iAxisID	DINT				ID of the axis block of the Pipe Network.
iPosition	LREAL				Reference position.
iDirectionType	DINT				0 (zero) = Positive. 1 = Negative. 2 = Shortest.
iVelocity	LREAL				Reference speed.
iAcceleration	LREAL				Reference acceleration.
iDeceleration	LREAL				Reference deceleration.
iTimeout	TIME				Time monitoring (T#0ms: off).

Outputs

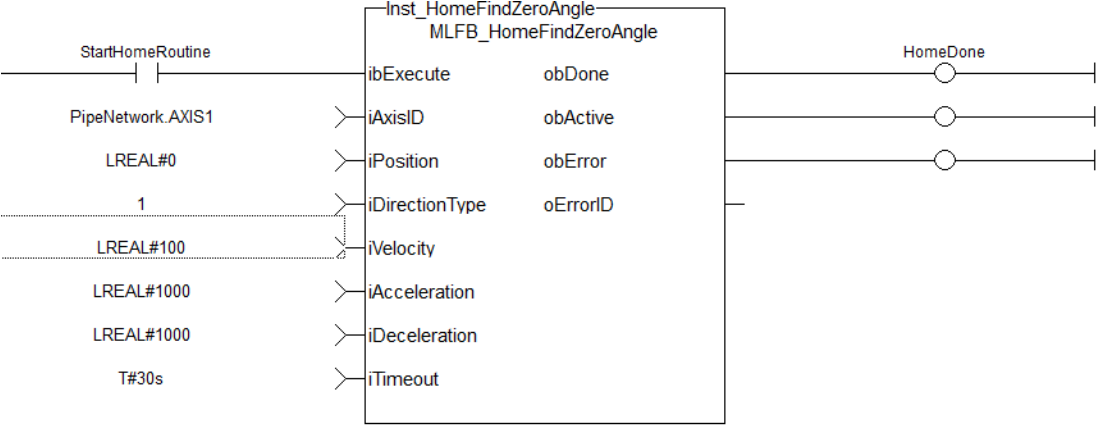
Output	Data Type	Range	Unit	Description
obDone	BOOL	FALSE, TRUE	N/A	Done bit.
obActive	BOOL	FALSE, TRUE	N/A	Active bit.
obError	BOOL	FALSE, TRUE	N/A	Error bit.

Output	Data Type	Range	Unit	Description	
oErrorID	DINT		N/A	Error identifier:	
				ErrorID	Description
				1	Axis in error.
				2	Axis is not enabled.
				3	Timeout exceeded or expired.
				4	SDO read / write error.
				5	Input parameter out of range.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Call homing function on Axis1 with preset velocity, accel, and decel values
//Start in negative direction and go to zero angle or time out
//after reaching zero angle set axis position to zero
Inst_MLFB_HomeFindZeroAngle( StartHomeRoutine,
```

```

PipeNetwork.AXIS1,
0,
1,
100,
1000,
1000,
T#30s );

HomeDone := Inst_MLFB_HomeFindZeroAngle.obDone;

```

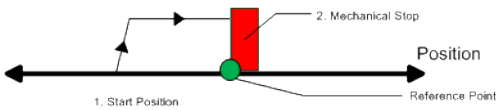
See Also

- "MLFB_HomeFindHomeInput" (→ p. 161)
- "MLFB_HomeFindHomeInputThenZeroAngle" (→ p. 164)
- "MLFB_HomeFindLimitInput" (→ p. 167)
- "MLFB_HomeFindLimitInputThenZeroAngle" (→ p. 170)
- "MLFB_HomeMoveUntilPosErrExceeded" (→ p. 176)
- "MLFB_HomeMoveUntilPosErrExceededThenZeroAngle" (→ p. 179)
- "MLFB_HomeUsingCurrentPosition" (→ p. 182)

2.4.4.6 MLFB_HomeMoveUntilPosErrExceeded



 **Function Block** - Homing to the position error is exceeded.



- When executed, the motor moves to the hard stop with a definable peak current.
- When the position error exceeds, the home Position is set.

Inputs

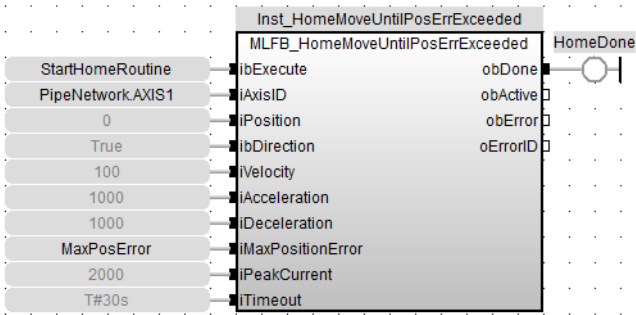
Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL	FALSE, TRUE	N/A	No default	Start homing, edge-triggered.
iAxisID	DINT				ID of the axis block of the Pipe Network.
iPosition	LREAL				Reference position.
ibDirection	BOOL	0 to 1	N/A	No default	Defines the axis homing direction. • 0 (zero) = Positive. • 1 = Negative.
iVelocity	LREAL				Reference speed.
iAcceleration	LREAL				Reference acceleration.
iDeceleration	LREAL				Reference deceleration.
iMaxPositionError	LREAL				Maximum position error.
iPeakCurrent	DINT				Peak current in mA.
iTimeout	TIME				Time monitoring (T#0ms: off).

Outputs

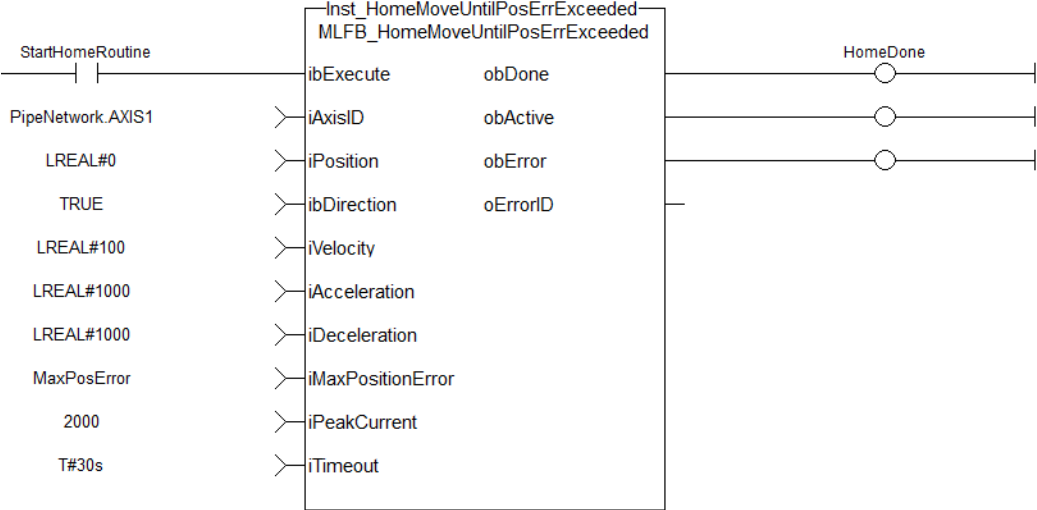
Output	Data Type	Range	Unit	Description
obDone	BOOL	FALSE, TRUE	N/A	Done bit.
obActive	BOOL	FALSE, TRUE	N/A	Active bit.

Output	Data Type	Range	Unit	Description	
obError	BOOL	FALSE, TRUE	N/A	Error bit.	
oErrorID	DINT		N/A	Error identifier:	
				ErrorID	Description
				1	Axis in error.
				2	Axis is not enabled.
				3	Timeout exceeded or expired.
				4	SDO read / write error.
5	Input parameter out of range.				

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example


```

//Call homing function on Axis1 with preset velocity, accel, and decel values
//Start in negative direction and go until position error exceeds input value
or time out
//afterterwards set axis position to zero
//function block temporarily writes new max current value to 2 Amp while home
routine active
Inst_MLFB_HomeMoveUntilPosErrExceeded( StartHomeRoutine,
    PipeNetwork.AXIS1,
    0,
    1,
    100,
    1000,
    1000,
    MaxPosError,
    2000,
    T#30s );

HomeDone := Inst_MLFB_HomeMoveUntilPosErrExceeded.obDone;

```

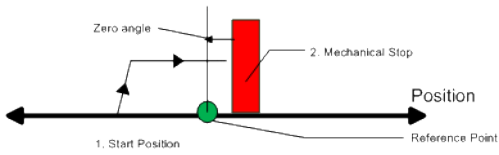
See Also

- "MLFB_HomeFindHomeInput" (→ p. 161)
- "MLFB_HomeFindHomeInputThenZeroAngle" (→ p. 164)
- "MLFB_HomeFindLimitInput" (→ p. 167)
- "MLFB_HomeFindLimitInputThenZeroAngle" (→ p. 170)
- "MLFB_HomeFindZeroAngle" (→ p. 173)
- "MLFB_HomeMoveUntilPosErrExceededThenZeroAngle" (→ p. 179)
- "MLFB_HomeUsingCurrentPosition" (→ p. 182)

2.4.4.7 MLFB_HomeMoveUntilPosErrExceededThenZeroAngle



Function Block - Homing to the position error is exceeded plus zero angle.



- Similar to the Move Until Position Error Exceeded method, the move until position error exceeded then find zero angle.
- Mode follows the same steps, but upon completion of the move, it continues to move to find the zero angle reference of the motor.

Inputs

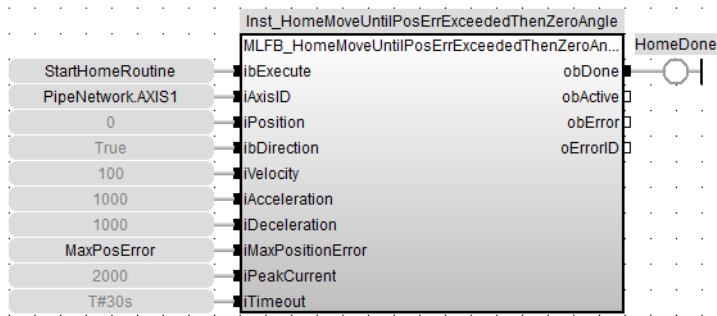
Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL	FALSE, TRUE	N/A	No default	Start homing, edge-triggered.
iAxisID	DINT				ID of the axis block of the Pipe Network.
iPosition	LREAL				Reference position.
ibDirection	BOOL	0 to 1	N/A	No default	Defines the axis homing direction. • 0 (zero) = Positive. • 1 = Negative.
iVelocity	LREAL				Reference speed.
iAcceleration	LREAL				Reference acceleration.
iDeceleration	LREAL				Reference deceleration.
iMaxPositionError	LREAL				Maximum position error.
iPeakCurrent	DINT				Peak current in mA.
iTimeout	TIME				Time monitoring (T#0ms: off).

Outputs

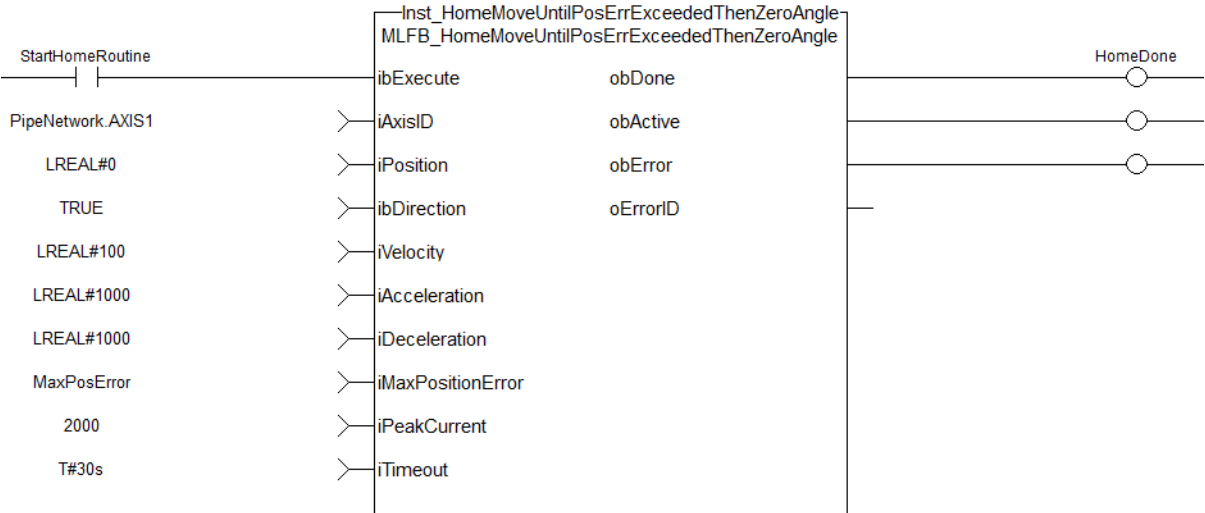
Output	Data Type	Range	Unit	Description
obDone	BOOL	FALSE, TRUE	N/A	Done bit.

Output	Data Type	Range	Unit	Description	
obActive	BOOL	FALSE, TRUE	N/A	Active bit.	
obError	BOOL	FALSE, TRUE	N/A	Error bit.	
oErrorID	DINT		N/A	Error identifier:	
				ErrorID	Description
				1	Axis in error.
				2	Axis is not enabled.
				3	Timeout exceeded or expired.
				4	SDO read / write error.
5	Input parameter out of range.				

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Call homing function on Axis1 with preset velocity, accel, and decel values
//Start in negative direction and go until position error exceeds input value
//or time out
//afterterwards moves to zero angle and sets axis position to zero
//function block temporarily writes new max current value to 2 Amp while home
//routine active
Inst_MLFB_HomeMoveUntilPosErrExceededThenZeroAngle( StartHomeRoutine,
    PipeNetwork.AXIS1,
    0,
    1,
    100,
    1000,
    1000,
    MaxPosError,
    2000,
    T#30s );

HomeDone := Inst_MLFB_HomeMoveUntilPosErrExceededThenZeroAngle.obDone;
```

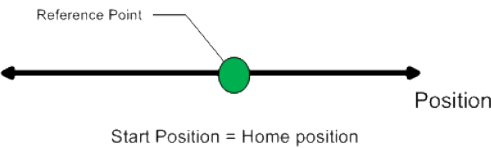
See Also

- "MLFB_HomeFindHomeInput" (→ p. 161)
- "MLFB_HomeFindHomeInputThenZeroAngle" (→ p. 164)
- "MLFB_HomeFindLimitInput" (→ p. 167)
- "MLFB_HomeFindLimitInputThenZeroAngle" (→ p. 170)
- "MLFB_HomeFindZeroAngle" (→ p. 173)
- "MLFB_HomeMoveUntilPosErrExceeded" (→ p. 176)
- "MLFB_HomeUsingCurrentPosition" (→ p. 182)

2.4.4.8 MLFB_HomeUsingCurrentPosition



 **Function Block** - Homing using current position.



- Use this parameter to set the value of the home position other than zero.
 - This allows you to offset your home reference away from zero.
- Using the current position is the most basic homing method.
 - This method uses the current position of the motor as the home point reference.

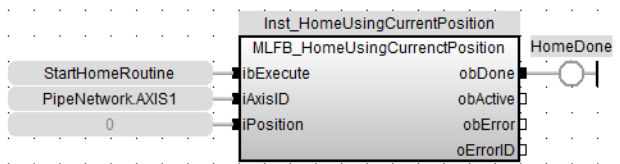
Inputs

Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL	FALSE, TRUE	N/A	No default	Start homing, edge-triggered.
iAxisID	DINT				ID of the axis block of the Pipe Network.
iPosition	LREAL				Reference position.

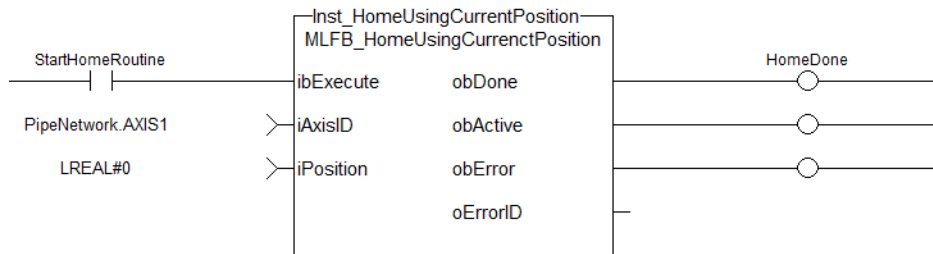
Outputs

Output	Data Type	Range	Unit	Description	
obDone	BOOL	FALSE, TRUE	N/A	Done bit.	
obActive	BOOL	FALSE, TRUE	N/A	Active bit.	
obError	BOOL	FALSE, TRUE	N/A	Error bit.	
oErrorID	DINT		N/A	Error identifier:	
				ErrorID	Description
				1	Axis in error.
				2	Axis is not enabled.
				3	Timeout exceeded or expired.
				4	SDO read / write error.
				5	Input parameter out of range.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//No movement, set current axis position to position input in this case zero
Inst_MLFB_HomeUsingCurrentPosition( StartHomeRoutine, PipeNetwork.AXIS1, 0
);
HomeDone := Inst_MLFB_HomeUsingCurrentPosition.obDone;
```

See Also

- "MLFB_HomeFindHomeInput" (→ p. 161)
- "MLFB_HomeFindHomeInputThenZeroAngle" (→ p. 164)
- "MLFB_HomeFindLimitInput" (→ p. 167)
- "MLFB_HomeFindLimitInputThenZeroAngle" (→ p. 170)
- "MLFB_HomeFindZeroAngle" (→ p. 173)
- "MLFB_HomeMoveUntilPosErrExceeded" (→ p. 176)
- "MLFB_HomeMoveUntilPosErrExceededThenZeroAngle" (→ p. 179)

2.4.5 MLFB_Jog



 **Function** - Define to jog an axis in the selected direction at a defined speed.

- This function or function block can be used with PLCopen Motion engines.
- This function or function block is used to command motion in a designated direction at a defined rate.
 - This may be used where continuous motion is required (e.g., a conveyor system or in a setup mode for manually jogging the axis).
- The **JogPlus** and **JogMinus** inputs select the direction the motion occurs in.
 - Motion starts when the **JogPlus** or **JogMinus** input is TRUE.
 - Motion stops when the input is FALSE.
 - Only one of these inputs should be enabled at a time.
 - If both are selected, the motion stops.
- If another motion is active when the jog is requested, that motion is aborted and the jog starts.
- The En input (FFLD editor only) must be high.
 - Typically wired to the rail.

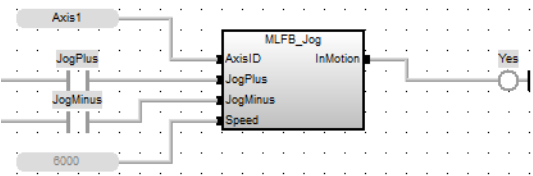
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	Enables execution. Used in the FFLD editor only.
AxisID	DINT	No range	N/A	No default	ID of the Axis block. The AxisID selects the axis to jog.
JogPlus	BOOL	FALSE, TRUE	N/A	No default	Enables a jog in the plus direction.
JogMinus	BOOL	FALSE, TRUE	N/A	No default	Enables a jog in the minus direction.
Speed	LREAL	No range	User units / sec	No default	Rate the axis moves.

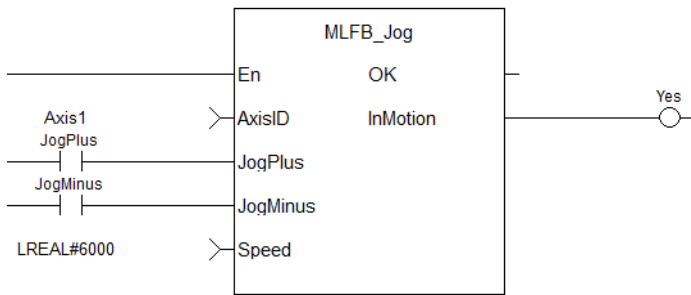
Outputs

Output	Data Type	Range	Unit	Description
InMotion	BOOL	FALSE, TRUE	N/A	Jogging is active when TRUE.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```

//Jog Axis1 at 6000 user units a second when JogPlus or JogMinus variables
are TRUE
//Stop motion on falling edge of either variable
MLFB_Jog( PipeNetwork.AXIS1, JogPlus, JogMinus, 6000 );

```

See Also

"MLAxisMoveVel" (→ p. 352)

2.4.6 PLS - UDFB - Motion/Pipe Network

These are the UDFB Motion/Pipe Network PLS function and function blocks.

Name	Description
"MLFB_PlsPosFw" (→ p. 187)	Forward position range indicator.
"MLFB_PlsPosFwBw" (→ p. 189)	Forward/backward position range indicator.
"MLFB_PlsTimeFw" (→ p. 192)	Forward/backward Position/Time range indicator.

2.4.6.1 MLFB_PlsPosFw

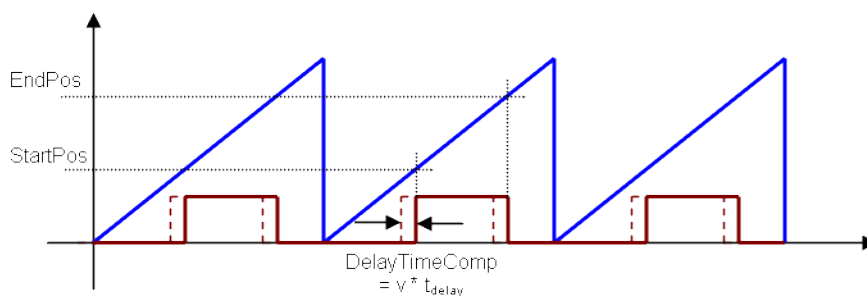
Pipe Network



Function Block - Forward position range indicator.

- Used in the command position path with ascending position.
- A dedicated comparator pipe block is needed.
- The Boolean output **oPLS** is set to:
 - TRUE if the position has crossed the start position.
 - FALSE if the position has crossed the end position.
- The function block is executed cyclically.
 - The modulo position is considered.
- The function block has the possibility to compensate a delay time of the connected device (e.g., glue nozzles).

2.4.6.1.1.1 Timing



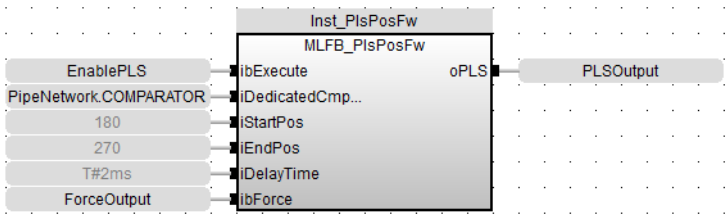
Inputs

Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL				Enable PLS.
iDedicatedCmpID	DINT				ID of the dedicated comparator.
iStartPos	LREAL				Start position of PLS.
iEndPos	LREAL				End position of PLS.
iDelayTime	TIME				Delay time for compensation.
ibForce	BOOL				Force PLS.

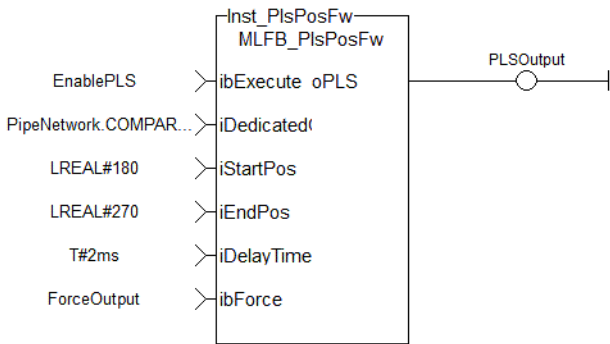
Outputs

Output	Data Type	Range	Unit	Description
oPLS	BOOL	FALSE, TRUE	N/A	Position limit switch.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//PLSOutput is True when chosen comparator is between 180 and 270 with a  
T#2ms delay  
//Can also force the output to be true with ForceOuput variable  
Inst_MLFB_PlsPosFw( EnablePLS, PipeNetwork.COMPARATOR, 180, 270, T#2ms,  
ForceOutput );  
PLSOutput := Inst_MLFB_PlsPosFw.oPLS;
```

2.4.6.2 MLFB_PlsPosFwBw

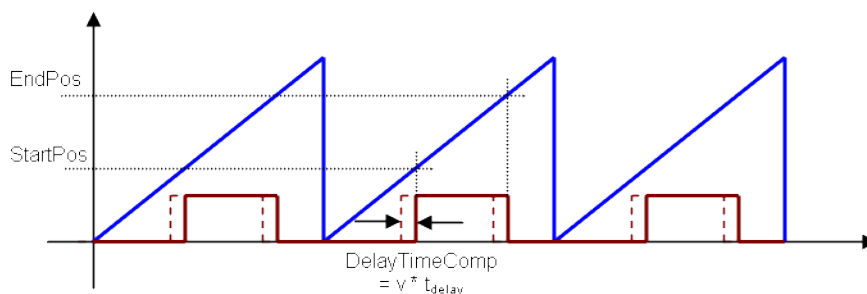
Pipe Network ✓



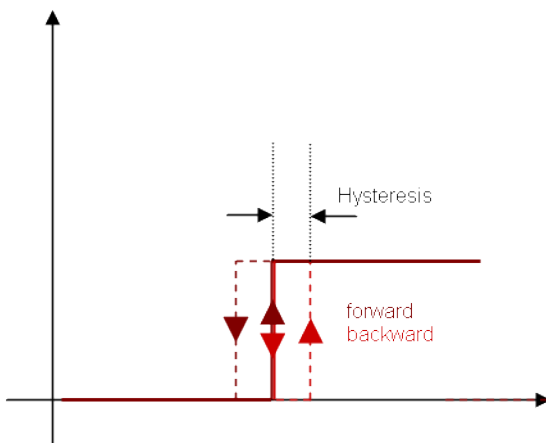
Function Block - Forward/backward position range indicator.

- Used in the command or actual position path, e.g., sampler pipe with noisy position, in both directions.
- Any modulo pipe block is needed, which can also be used for another instance of this UDFB.
- It is also possible to define a hysteresis for switching on and off of the PLS.
- The Boolean output **oPLS** is set to:
 - TRUE if the position has crossed the start position.
 - FALSE if the position has crossed the end position.
- The function block is executed cyclically.
 - The modulo position is considered.
- The function block has the possibility to compensate a delay time of the connected device (e.g., glue nozzles).

2.4.6.2.1.1 Timing



2.4.6.2.2.2 Hysteresis



Inputs

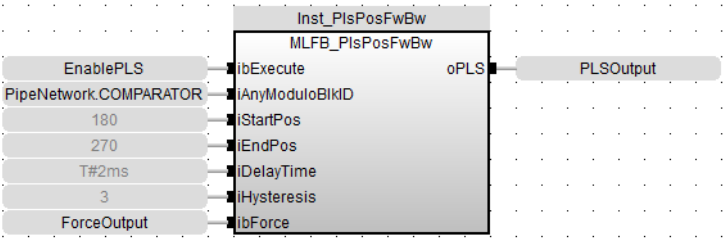
Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL				Enable PLS.

Input	Data Type	Range	Unit	Default	Description
iAnyModuloBlkID	DINT				Any modulo pipe network block ID.
iStartPos	LREAL				Start position of PLS.
iEndPos	LREAL				End position of PLS.
iDelayTime	TIME				Delay time for compensation.
iHysteresis	LREAL				Hysteresis.
ibForce	BOOL				Force PLS.

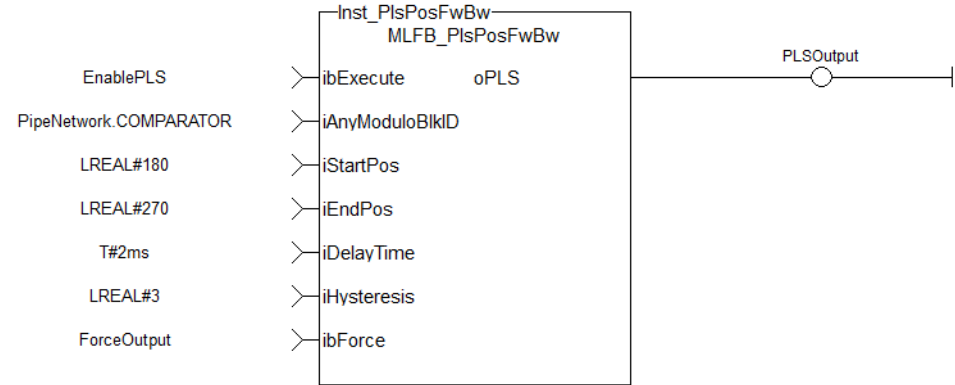
Outputs

Output	Data Type	Range	Unit	Description
oPLS	BOOL	FALSE, TRUE	N/A	Position limit switch.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//PLSOutput is True when chosen comparator is between 180 and 270 with a  
T#2ms delay  
//Can also force the output to be true with ForceOutput variable  
//Hysteresis is on for 3 user units in case direction changes around start  
point  
Inst_MLFB_PlsPosFwBw( EnablePLS, PipeNetwork.COMPARATOR, 180, 270, T#2ms, 3,  
ForceOutput );  
PLSOutput := Inst_MLFB_PlsPosFwBW.oPLS;
```

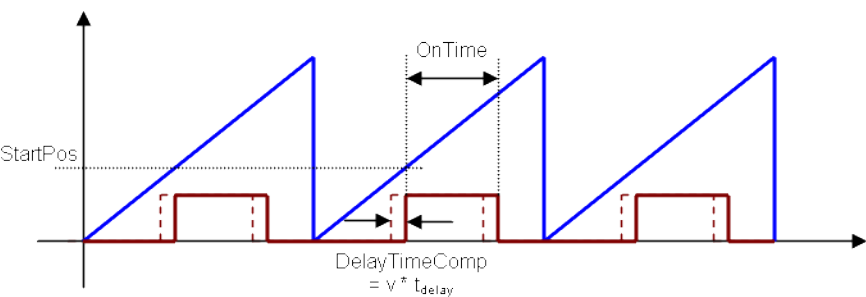
2.4.6.3 MLFB_PlsTimeFw



 **Function Block** - Forward/backward Position/Time range indicator.

- Used in the command position path with ascending position.
- A dedicated comparator pipe block is needed.
- The Boolean output **oPLS** is set to:
 - TRUE if the position has crossed the start position.
 - A timer with **iOnTime** must be started.
 - When the timer has expired the output is set to FALSE.
- The function block is executed cyclically.
 - The modulo position is considered.
- The function block has the possibility to compensate a delay time of the connected device (e.g., glue nozzles).

2.4.6.3.1.1 Timing



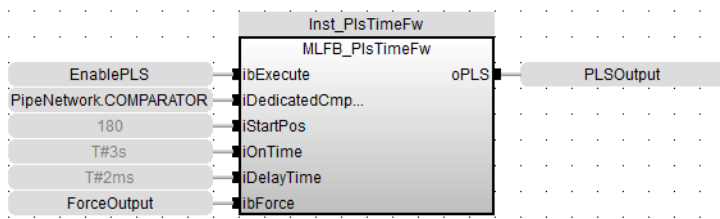
Inputs

Input	Data Type	Range	Unit	Default	Description
ibExecute	BOOL				Enable PLS.
iDedicatedCmplID	DINT				ID of the dedicated comparator.
iStartPos	LREAL				Start position of PLS.
iOnTime	TIME				Time the PLS is on.
iDelayTime	TIME				Delay time for compensation.
ibForce	BOOL				Force PLS.

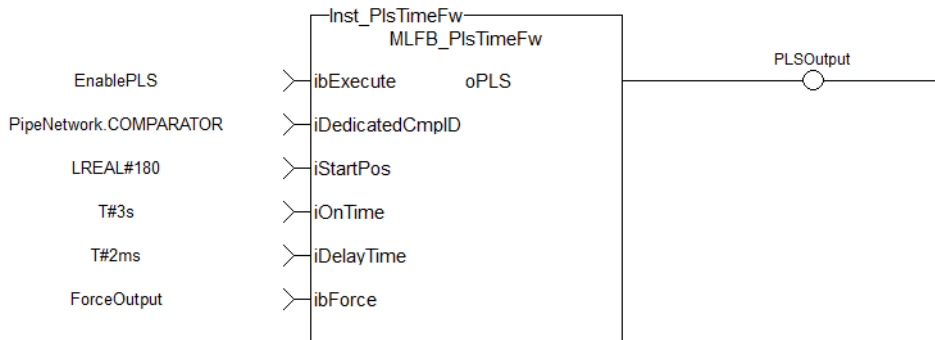
Outputs

Output	Data Type	Range	Unit	Description
oPLS	BOOL	FALSE, TRUE	N/A	Position limit switch.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//PLSOutput is True when chosen comparator passes 180 for 3 seconds a T#2ms
delay
//Can also force the output to be true with ForceOuput variable
Inst_MLFB_PlsTimeFw( EnablePLS, PipeNetwork.COMPARATOR, 180, T#3s, T#2ms,
ForceOutput );
PLSOutput := Inst_MLFB_PlsTimeFw.oPLS;
```


2.5 UDFB - Motion / PLCopen

These are the Kollmorgen UDFB Motion/PLCopen function and function blocks.

Name	Description
MCFB_AKDFault	Outputs AKD drive fault information.
MCFB_AKDFaultLookup	String message of the corresponding AKD drive fault number.
MCFB_DriveFault	Returns the fault status, number, and description of the requested axis mapped to a Kollmorgen drive.
MCFB_ECATRstart	Reinitializes the EtherCAT network and the motion engine.
MCFB_GearedWebTension	Facilitates dancer and tension control in an electronic geared master/slave machine design.
MCFB_Jog	Define to jog an axis in the selected direction at a defined speed.
MCFB_StepAbsolutes	Homing by setting Actual Position to the position of the feedback.
MCFB_StepAbsSwitch	Homing to a home switch.
MCFB_StepAbsSwitchFastInput	Fast homing to a home switch.
MCFB_StepBlock	Homing to a physical object, mechanically blocking the movement.
MCFB_StepLimitSwitch	Homing to a limit switch.
MCFB_StepLimitSwitchFastInput	Fast homing to a limit switch.
MCFB_StepRefPulse	Homing to a zero angle reference.

2.5.1 Drive Fault Reporting - UDFB - Motion/PLCopen

These are the Kollmorgen UDFB Motion/PLCopen Drive Fault Reporting function blocks.

Name	Description
"MCFB_AKDFault" (→ p. 196)	Outputs AKD drive fault information.
"MCFB_AKDFaultLookup" (→ p. 198)	String message of the corresponding AKD drive fault number.
"MCFB_DriveFault" (→ p. 200)	Returns the fault status, number, and description of the requested axis mapped to a Kollmorgen drive.

2.5.1.1 MCFB_AKDFault



Function Block - Outputs AKD drive fault information.

TIP

- This function block lists the **highest priority** fault as displayed on the AKD.
- The "FB_AKDFltRpt" (→ p. 122) function block lists faults in the order they occur.
- This function or function block can be used with PLCopen Motion engines.
- The FAULT output returns TRUE when the selected drive goes into a fault state.
- The **FaultNumber** is the same number as reported on the display of the AKD drive.

2.5.1.1.1.1 Usage

- Provide drive fault information the application program uses to determine next steps.
 - Example: Perform a machine-controlled stop or perform an immediate disable of the servo drives.
- In the application program, sends output fault information from this UDFB to the HMI for review by the machine operator.

Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables the Kollmorgen UDFB. Used in the FFLD editor only.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function. • See AXIS_REF Structure. • See Axis Name and Number.

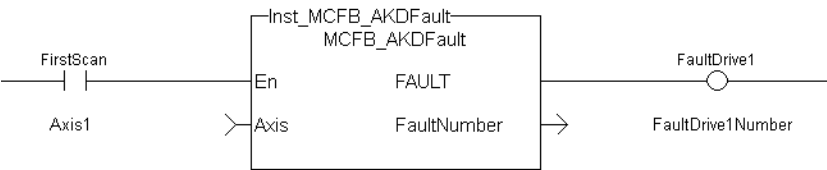
Outputs

Output	Data Type	Range	Unit	Description
FAULT	BOOL	FALSE, TRUE	N/A	TRUE if the selected drive currently has a Fault.
FaultNumber	DINT	100 to 999	N/A	The AKD drive fault number.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Execute and Read the Function Block

Inst_MCFB_AKDFault( Axis1(*lib:AXIS_REF*) );
FaultDrive1 := Inst_MCFB_AKDFault.FAULT;
FaultDrive1Number := Inst_MCFB_AKDFault.FaultNumber;

FaultDrive1Description := MCFB_AKDFaultLookup( FaultDrive1Number(*DINT*) );
```

See Also

- "FB_AKDFltRpt" (→ p. 122)
- "MC_ReadStatus" (→ p. 652) (PLCopen Motion Engine)
- "MCFB_AKDFaultLookup" (→ p. 198)

2.5.1.2 MCFB_AKDFaultLookup



 **Function Block** - String message of the corresponding AKD drive fault number.

- This function or function block can be used with PLCopen Motion engines.
- The **FaultNumber** is the same number as reported on the display of the AKD drive.
- OK output turns TRUE when the there is a match for the **FaultNumber**.
- The **FaultDescription** displays the corresponding text string.

2.5.1.2.1.1 Usage

- Provide drive fault information the application program uses to determine next steps.
 - Example: Perform a machine-controlled stop or perform an immediate disable of the servo drives.
- In the application program, sends output fault information from this UDFB to the HMI for review by the machine operator.

Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables the Kollmorgen UDFB. Used in the FFLD editor only.
FaultNumber	DINT	No range	N/A	No default	The AKD drive fault number.

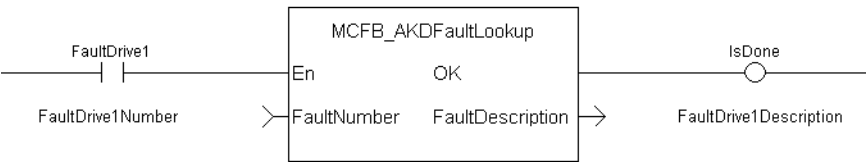
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	TRUE if there is a match for the FaultNumber .
FaultDescription	STRING	N/A	N/A	Description of the fault.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Execute and Read the Function Block
```

```
Inst_MCFB_AKDFault( Axis1(*lib:AXIS_REF*) );  
FaultDrive1 := Inst_MCFB_AKDFault.FAULT;  
FaultDrive1Number := Inst_MCFB_AKDFault.FaultNumber;  
  
FaultDrive1Description := MCFB_AKDFaultLookup( FaultDrive1Number(*DINT*) );
```

See Also

- "FB_AKDFltRpt" (→ p. 122)
- "MC_ReadStatus" (→ p. 652) (PLCopen Motion Engine)
- "MCFB_AKDFault" (→ p. 196)

2.5.1.3 MCFB_DriveFault



 **Function Block** - Returns the fault status, number, and description of the requested axis mapped to a Kollmorgen drive.

IMPORTANT

This function block requires "FB_S700FltRpt" (→ p. 134), "MCFB_AKDFault" (→ p. 196), and "MCFB_AKDFaultLookup" (→ p. 198) subprograms imported to project to compile and function.

- The FAULT output returns TRUE when the selected drive goes into a fault state.
- The fault number and description depend on the drive type mapped to the axis.
 - If the drive is an:
 - AKD or AKD2G, the fault number is the same number as reported on the display of the drive.
 - AKT2G Stepper, the fault number represents the drive status word.
 - This word is a bitmask that represents the various error conditions.

2.5.1.3.1.1 Usage

- Provide drive fault information the application program uses to determine next steps.
 - Example: Perform a machine-controlled stop or perform an immediate disable of the servo drives.
- In the application program, sends output fault information from this UDFB to the HMI for review by the machine operator.

Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables the Kollmorgen UDFB. Used in the FFLD editor only.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function. • See AXIS_REF Structure. • See Axis Name and Number.

Outputs

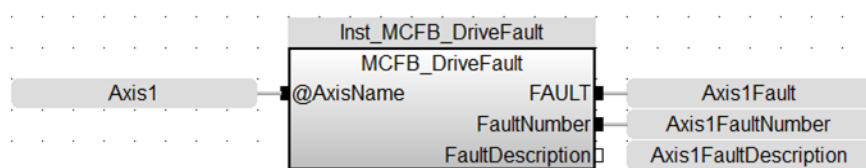
Output	Data Type	Range	Unit	Description
FAULT	BOOL	FALSE, TRUE	N/A	TRUE if the selected drive currently has a Fault.

Output	Data Type	Range	Unit	Description
FaultNumber	DINT	No range	N/A	If the axis is: - S300/S700 - Three-digit fault identifier. - See S300 & S700 Errors and Warnings on KDN for a list of fault codes. - AKD - Three-digit fault identifier. - See Fault and Warning Messages for a list of fault codes. - AKD2G - Four-digit fault identifier. - See Faults and Warnings View for a list of fault codes. - AKT2G Stepper - Drive Status word (bitmask). - See "Bitmask Causes" (→ p. 201).
FaultDescription	STRING	N/A	N/A	Description of the fault.

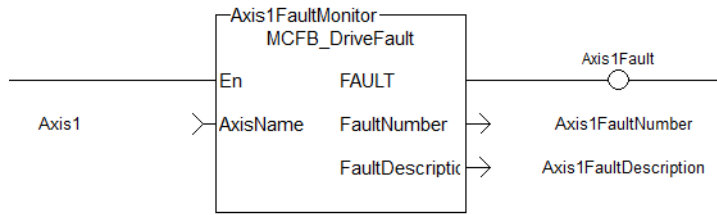
2.5.1.3.2.2 Bitmask Causes

Bit	Description	Cause
0	Saturated	Drive stage operates with maximum duty cycle.
1	Over temperature.	Internal temperature is higher than 80 °C.
2	Torque overload.	Motor current is higher than the rated current.
3	Under voltage.	Motor supply voltage is either: 20% lower than the configured nominal voltage (warning). - Less than 8V.
4	Over voltage.	Motor supply voltage is 10% higher than the configured nominal voltage.
5	Short circuit A.	Short circuit in motor coil A.
6	Short circuit B.	Short circuit in motor coil B.
7	No control power.	Control voltage at the power contacts is less than 12V.
8	Misc. error.	Either: - The terminal initialization failed. - Supply voltage is less than 8V. - Internal terminal temperature is higher than 100 °C.
9	Configuration error.	CoE change has not yet been adopted into the current configuration.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Execute and Read the Function Block
Inst_MCFB_DriveFault( Axis1 );
Axis1Fault := Inst_MCFB_DriveFault.FAULT;
Axis1FaultNumber := Inst_MCFB_DriveFault.FaultNumber;
Axis1FaultDescription := Inst_MCFB_DriveFault.FaultDescription;
```

Related Functions

- "FB_S700FltRpt" (→ p. 134)
- "MC_ReadStatus" (→ p. 652) (PLCopen Motion Engine)
- "MCFB_AKDFault" (→ p. 196)
- "MCFB_AKDFaultLookup" (→ p. 198)

2.5.2 MCFB_ECATRestart



Function Block - Reinitializes the EtherCAT network and the motion engine.

- This function block clears motion engine errors, motion bus driver errors and EtherCAT network errors before reinitializing the motion engine, if requested to do so.
- This UDFB allows the EtherCAT and motion engines to restart without having to restart the entire project.

Examples:

- EtherCAT network wire is replaced or accidentally disconnected.
- Axis setup parameters defined by CreateAxis and/or InitAxis function need to be changed while the application is running.

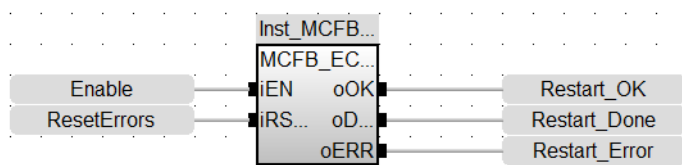
Inputs

Input	Data Type	Range	Unit	Default	Description
iEN	BOOL	FALSE, TRUE	N/A	No default	• Enables the Kollmorgen UDFB. • Used in the FFLD editor only.
IRSTERR	BOOL	FALSE, TRUE	0 to 1	No default	Clears the motion engine and EtherCAT network errors in case of any faults.

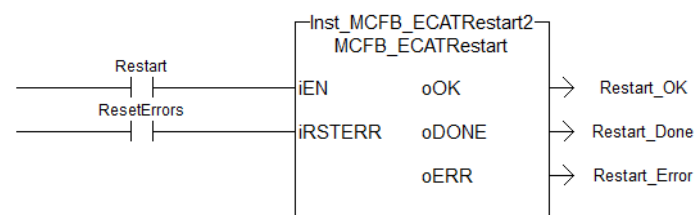
Outputs

Output	Data Type	Range	Unit	Description
oOK	BOOL	FALSE, TRUE	N/A	Function block activated status.
oDONE	BOOL	FALSE, TRUE	N/A	Execution complete.
oERR	BOOL	FALSE, TRUE	N/A	TRUE if the system initialization fails.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MCFB_ECATRstart( Restart, ResetErrors );  
IF Inst_MCFB_ECATRstart.oDONE THEN  
    RestartComplte:=1;  
End_IF;
```

See Also

- "ClearCtrlErrors" (→ p. 901)
- "MLMotionInit" (→ p. 867)
- "MLMotionRstErr" (→ p. 870)
- "MLMotionStart" (→ p. 872)

2.5.3 MCFB_StepAbsSwitchFastInput



Function Block - Fast homing to a home switch.

- Performs a homing function by searching for an absolute positioned external physical switch.
 - The switch must be connected to one of the two fast inputs on the Axis' AKD drive.
- An Absolute Switch has two Off (or On) areas.

2.5.3.0.1 Usage

- The homing is commanded in the most likely direction where the sensor can be found.
- If any LimitSwitch is found during Homing (any of them), a special process is started in the opposite direction.
 - The AbsSwitch is searched to switch Off (or On, depending on SwitchMode setting).
- The Edge (passed by) and homing process is restarted in the original direction and with the same conditions.
 - This ensures the end conditions are always the same.

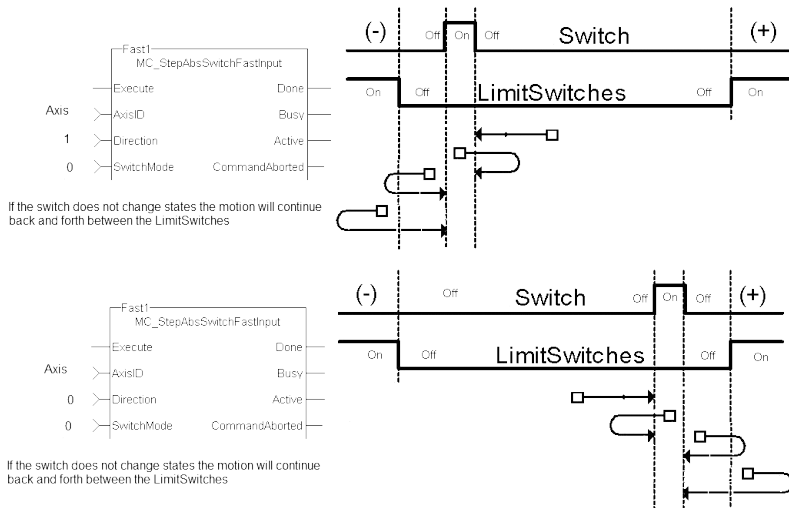


Figure 3-12: MCFB_StepAbsSwitchFastInput Usage 1

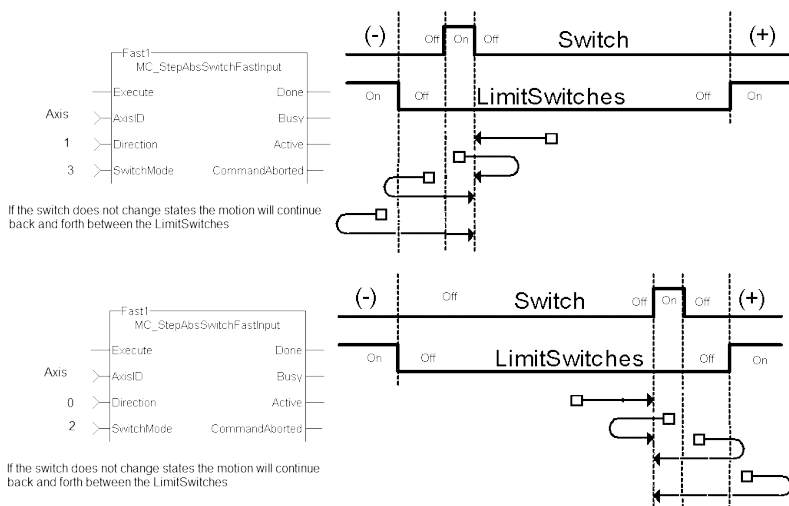


Figure 3-13: MCFB_StepAbsSwitchFastInput Usage 2

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	- Request the homing step procedure at the rising edge. - Outputs are reset when execute input is FALSE.
AxisID	AXIS_REF	1 to 256	N/A	No default	- Structure for specified Axis designated to home. - See AXIS_REF Structure.
Direction	BOOL	FALSE, TRUE	N/A	No default	Defines the axis homing direction. 0 (zero) = Clockwise rotation. 1 = Counterclockwise rotation.
SwitchMode	DINT	0 to 3	N/A	No default	Switch state to complete homing. 0 (zero) = Rising edge of sensor. 1 = Falling edge of sensor. 2 = Rising edge when traveling in positive direction but falling edge in negative direction. 3 = Falling edge when traveling in negative direction but rising edge in positive direction.
Velocity	LREAL	No range	User units / sec	No default	Commanded velocity for the homing move.
Acceleration	LREAL	No range	User units / sec ²	No default	Commanded acceleration for the homing move.
Deceleration	LREAL	No range	User units / sec ²	No default	Commanded deceleration for the homing move.
Jerk	LREAL	No range	User units / sec ³	No default	- Commanded jerk for the homing move. - If 0 (zero), trapezoidal acc/dec is used.
TorqueLimit	LREAL	No range	User units	No default	Maximum torque applied for the homing move. - Entered in thousandths of maximum torque. - Example: 250 is 250/1000, or 25%.
TimeLimit	TIME	No range	Sec	No default	Maximum time for homing move to complete. - If exceeded, the homing procedure errors out. 0 (zero) = no time limit.

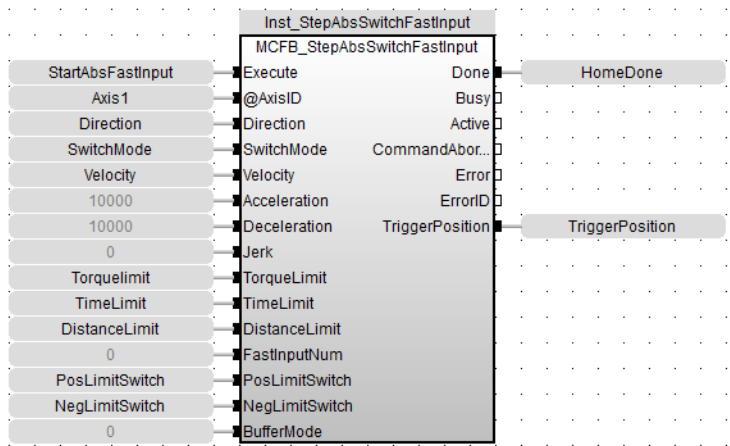
Input	Data Type	Range	Unit	Default	Description
DistanceLimit	LREAL	No range	User units	No default	Maximum distance for homing move to complete. - If exceeded, the homing procedure errors out. 0 (zero) = no distance limit.
FastInputNum	BOOL	FALSE, TRUE	N/A	No default	0 (zero) = first fast input (X7 pin 10). 1 = second fast input (X7 pin 9).
PosLimitSwitch	BOOL	FALSE, TRUE	N/A	No default	The positive direction limit switch input I/O point.
NegLimitSwitch	BOOL	FALSE, TRUE	N/A	No default	The negative direction limit switch input I/O point.
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. - BM_ABORTING - BM_BUFFERED - BM_BLENDING_PREVIOUS - BM_BLENDING_NEXT - BM_BLENDING_LOW - BM_BLENDING_HIGH - See Buffer Modes.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input goes high until the time the move is ended.
Active	BOOL	FALSE, TRUE	N/A	Set when the function block is active.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	Signals an error has occurred within the function block.

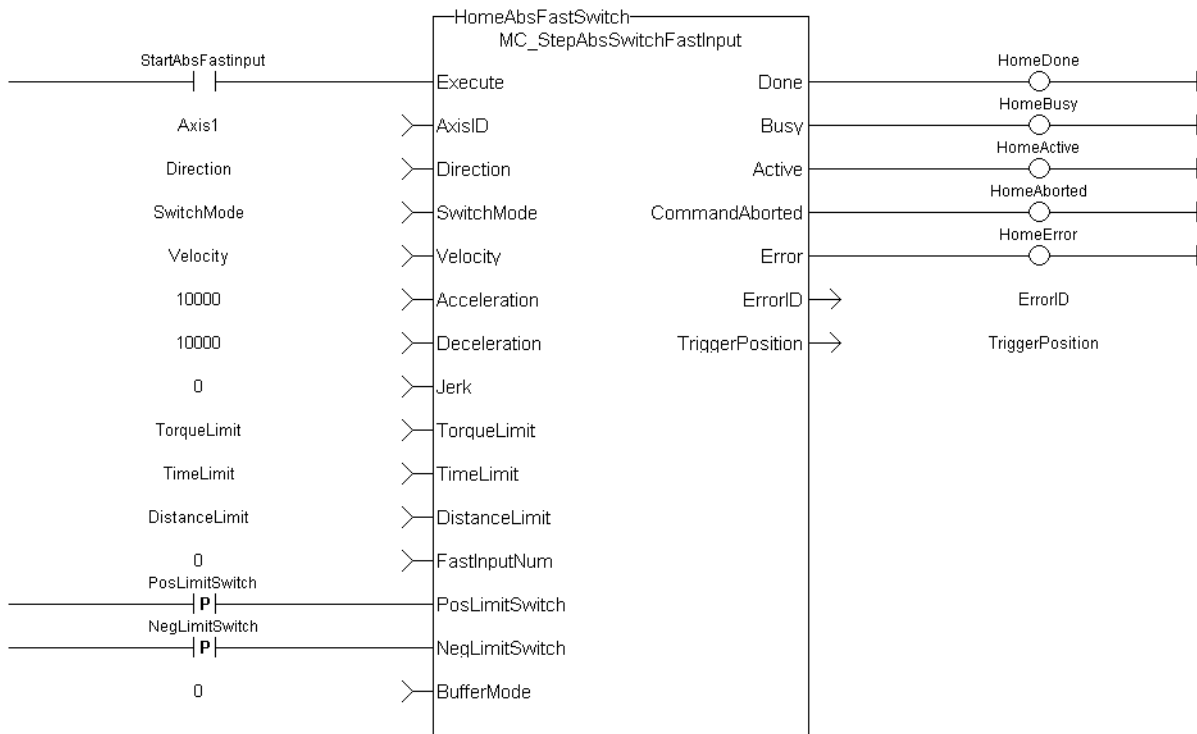
Output	Data Type	Range	Unit	Description
ErrorID	INT	Enumerated	N/A	Indicates the error if Error output is TRUE. Error identifier:
TriggerPosition	LREAL	No range	User units	

FBD Language Example



FFLD Language Example

PosLimitSwitch, NegLimitSwitch are declared I/O points.



IL Language Example

Not available.

ST Language Example

```
Execute_1 := 1;
(*Positive_Switch and Negative_Switch are physical hardware in Dictionary. *)

Inst_MC_StepAbsSwitchFastInput( Execute_1, Axis1, 0, 0,
10000.0,Acceleration:=10000.0, 10000.0, 0, 0, 0, 0, 0, Positive_Switch ,
Negative_Switch , 0)

HomeComplete := Inst_MC_StepAbsSwitchFastInput.Done;
HomeBusy := Inst_MC_StepAbsSwitchFastInput.Busy;
HomeActive := Inst_MC_StepAbsSwitchFastInput.Active;
HomeAborted := Inst_MC_StepAbsSwitchFastInput.CommandAborted;
HomeError := Inst_MC_StepAbsSwitchFastInput.Error;
HomeErrorID := Inst_MC_StepAbsSwitchFastInput.ErrorID;
HomeTriggerPosition := Inst_MC_StepAbsSwitchFastInput.TriggerPosition;
```

See Also

"MCFB_StepLimitSwitchFastInput" (→ p. 210)

2.5.4 MCFB_StepLimitSwitchFastInput



Function Block - Fast homing to a limit switch.

- Performs a homing function by searching for an external physical switch.
- The switch must be connected to one of the two fast inputs on the Axis' AKD drive.
- The Axis moves and, when a fast input is triggered, the triggered axis then performs an absolute move to the latched position.

2.5.4.0.1 Usage

The homing is commanded in the most likely direction where the sensor can be found.

In this example (-).

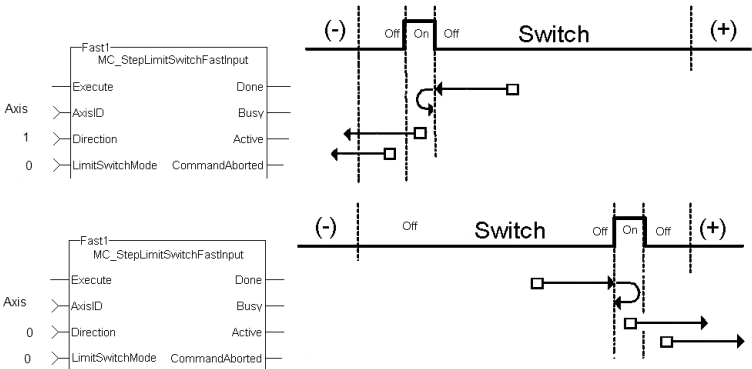


Figure 3-14: MCFB_StepLimitSwitchFastInput Usage

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	• Request the homing step procedure at the rising edge. • Outputs are reset when execute input is FALSE.
AxisID	AXIS_REF	1 to 256	N/A	No default	• Structure for specified Axis designated to home. • See AXIS_REF Structure.
Direction	BOOL	FALSE, TRUE	N/A	No default	Defines the axis homing direction. • 0 (zero) = Clockwise rotation. • 1 = Counterclockwise rotation.
LimitSwitchMode	DINT	0 to 1	N/A	No default	Limit switch state to complete homing. • 0 (zero) = Rising edge of sensor. • 1 = Falling edge of sensor.
Velocity	LREAL	No range	User units / sec	No default	Commanded velocity for the homing move.

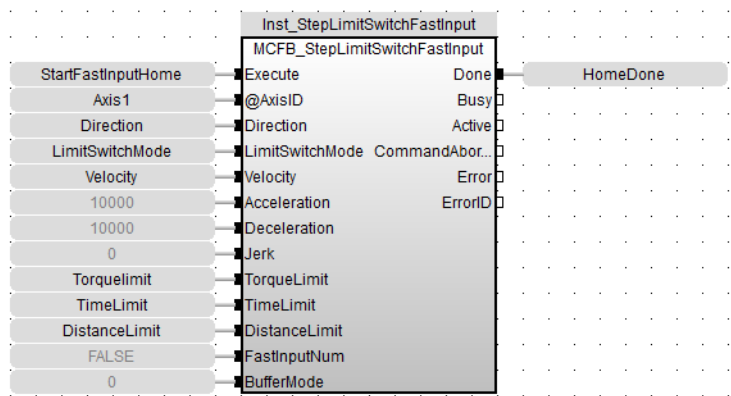
Input	Data Type	Range	Unit	Default	Description
Acceleration	LREAL	No range	User units / sec ²	No default	Commanded acceleration for the homing move.
Deceleration	LREAL	No range	User units / sec ²	No default	Commanded deceleration for the homing move.
Jerk	LREAL	No range	User units / sec ³	No default	- Commanded jerk for the homing move. - If 0 (zero), trapezoidal acc/dec is used.
TorqueLimit	LREAL	No range	User units	No default	Maximum torque applied for the homing move. - Entered in thousandths of maximum torque. - Example: 250 is 250/1000, or 25%.
TimeLimit	TIME	No range	Sec	No default	Maximum time for homing move to complete. - If exceeded, the homing procedure errors out. 0 (zero) = no time limit.
DistanceLimit	LREAL	No range	User units	No default	Maximum distance for homing move to complete. - If exceeded, the homing procedure errors out. 0 (zero) = no distance limit.
FastInputNum	BOOL	FALSE, TRUE	N/A	No default	0 (zero) = first fast input (X7 pin 10). 1 = second fast input (X7 pin 9).
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. - BM_ABORTING - BM_BUFFERED - BM_BLENDING_PREVIOUS - BM_BLENDING_NEXT - BM_BLENDING_LOW - BM_BLENDING_HIGH - See Buffer Modes.

Outputs

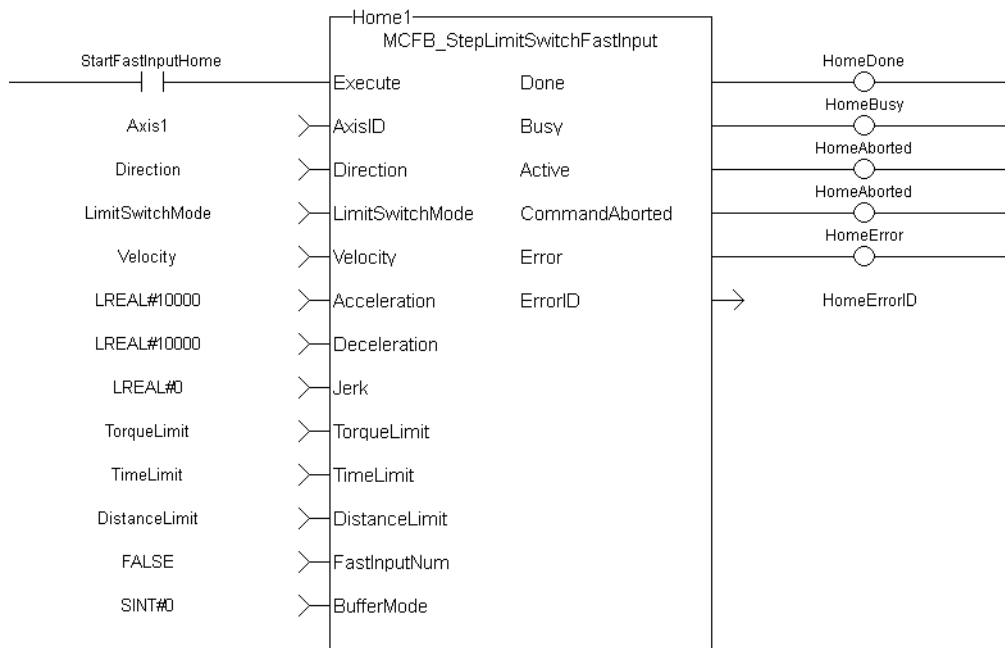
Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input goes high until the time the move is ended.

Output	Data Type	Range	Unit	Description														
Active	BOOL	FALSE, TRUE	N/A	Set when the function block is active.														
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.														
Error	BOOL	FALSE, TRUE	N/A	Signals an error has occurred within the function block.														
ErrorID	INT	Enumerated	N/A	Indicates the error if Error output is TRUE. Error identifier: <table><tr><th>ErrorID</th><th>Description</th></tr><tr><td>1</td><td>TimeLimit exceeded.</td></tr><tr><td>2</td><td>DistanceLimit exceeded.</td></tr><tr><td>3</td><td>TorqueLimit exceeded.</td></tr><tr><td>4</td><td>Axis error stop state.</td></tr><tr><td>5</td><td>Axis not enabled.</td></tr><tr><td>6</td><td>Invalid inputs for Velocity-Acceleration-Deceleration.</td></tr></table>	ErrorID	Description	1	TimeLimit exceeded.	2	DistanceLimit exceeded.	3	TorqueLimit exceeded.	4	Axis error stop state.	5	Axis not enabled.	6	Invalid inputs for Velocity-Acceleration-Deceleration.
ErrorID	Description																	
1	TimeLimit exceeded.																	
2	DistanceLimit exceeded.																	
3	TorqueLimit exceeded.																	
4	Axis error stop state.																	
5	Axis not enabled.																	
6	Invalid inputs for Velocity-Acceleration-Deceleration.																	

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Execute_1 := 1;

Inst_MCFB_StepLimitSwitchFastInput( Execute_1, Axis1, 0, 0, 10000.0, 10000.0,
10000.0, 0, 0, 0, 0, 0, 0);

HomeComplete := Inst_MCFB_StepLimitSwitchFastInput.Done;
HomeBusy := Inst_MCFB_StepLimitSwitchFastInput.Busy;
HomeActive := Inst_MCFB_StepLimitSwitchFastInput.Active;
HomeAborted := Inst_MCFB_StepLimitSwitchFastInput.CommandAborted;
HomeError := Inst_MCFB_StepLimitSwitchFastInput.Error;
HomeErrorID := Inst_MCFB_StepLimitSwitchFastInput.ErrorID;
```

See Also

"MCFB_StepAbsSwitchFastInput" (→ p. 205)

2.5.5 Homing - UDFB - Motion/PLCopen

These are the UDFB Motion/PLCopen Homing function blocks.

Name	Description
"MCFB_StepAbsolutes" (→ p. 215)	Homing by setting Actual Position to the position of the feedback.
"MCFB_StepAbsSwitch" (→ p. 217)	Homing to a home switch.
"MCFB_StepBlock" (→ p. 222)	Homing to a physical object, mechanically blocking the movement.
"MCFB_StepLimitSwitch" (→ p. 226)	Homing to a limit switch.
"MCFB_StepRefPulse" (→ p. 231)	Homing to a zero angle reference.

2.5.5.1 MCFB_StepAbsolutes



Function Block - Homing by setting Actual Position to the position of the feedback.

- Performs a static homing function by setting Actual Position to the position of an absolute encoder.
- No physical motion is performed in this mode.
- Equivalent to MC_SetPosition is performed with SetPosition coming from absolute encoder reading but with the option of using the once per rev feedback value.

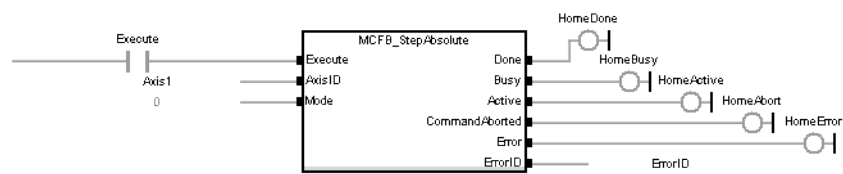
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	• Enables execution. • Used in the FFLD editor only.
Execute	BOOL	FALSE, TRUE	N/A	No default	Request the homing step procedure at the rising edge.
AxisID	AXIS_REF	1 to 256	N/A	No default	• Name of a declared instance of the AXIS_REF library function. • See AXIS_REF Structure.
Mode	BOOL	FALSE, TRUE	N/A	No default	Define the actual position assignment source. • 0 (zero) = Use drive feedback position for actual position. • 1 = Use once per rev feedback position.

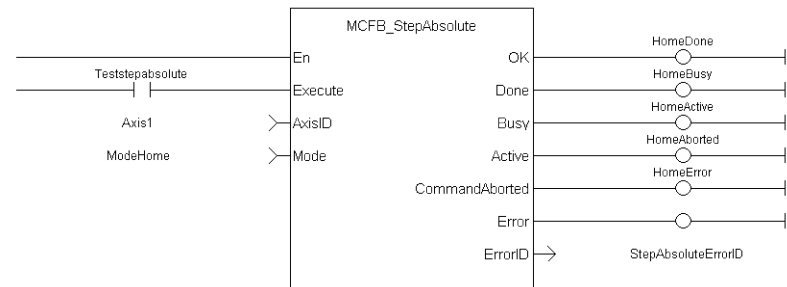
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input goes high until the time the move is ended.
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	Indicates either: • An invalid input was specified. • The move was terminated due to an error.
ErrorID	INT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • 1 = Desired SetPosition is outside of Rollover period.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Write current once per rev feedback position to overall axis position
MCFB_StepAbsolute( ExecuteHome, Axis1, ModeHome );
```

See Also

- "MCFB_StepAbsSwitch" (→ p. 217)
- "MCFB_StepBlock" (→ p. 222)
- "MCFB_StepLimitSwitch" (→ p. 226)
- "MCFB_StepRefPulse" (→ p. 231)

2.5.5.2 MCFB_StepAbsSwitch



Function Block - Homing to a home switch.

- Performs a homing function by searching for an absolute positioned external physical switch.
- An Absolute Switch has two Off (or On) areas.

2.5.5.2.1.1 Usage

This physical layout has the risk that homing is started in the wrong direction (escaping the switch).

To support such case, it implements a special behavior when Limit Switches are found (or the AbsSwitch itself is On at Execute).

- This function block does not modify the actual position.
- The homing is commanded in the most likely direction where the sensor can be found.
- The Velocity is defined by the input.
- The Torque is limited.
- Time and Distance Limits can cause an error if exceeded.
- If any LimitSwitch is found during Homing (any of them), a special process is started in the opposite direction.
 - The AbsSwitch is searched to switch Off (or On, depending on SwitchMode setting).
- The Edge (passed by) and homing process is restarted in the original direction and with the same conditions.
 - This ensures the end conditions are always the same.
- If the SwitchMode is either MC_SwitchNegative or MC_SwitchPositive, then the special process is also started in opposite direction depending from the switch state at Execute.
- The direction changes only when the specified Velocity is reached.

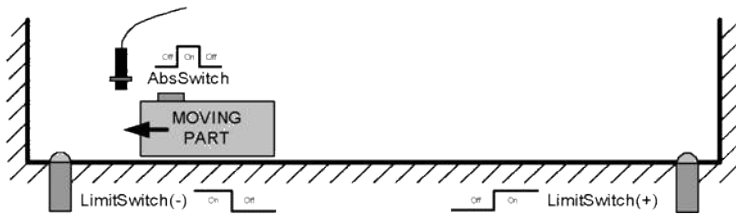


Figure 3-15: MCFB_StepAbsSwitch Usage 1

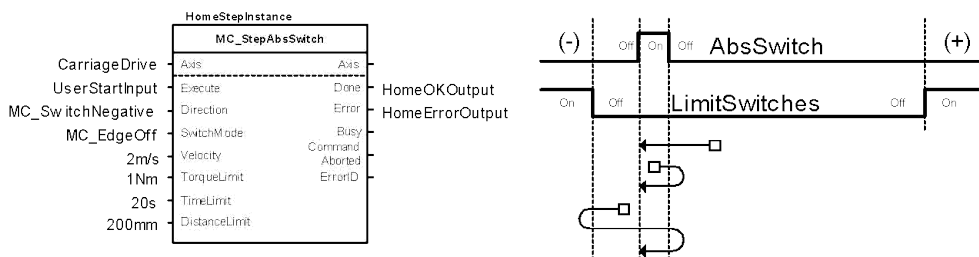


Figure 3-16: MCFB_StepAbsSwitch Usage 2

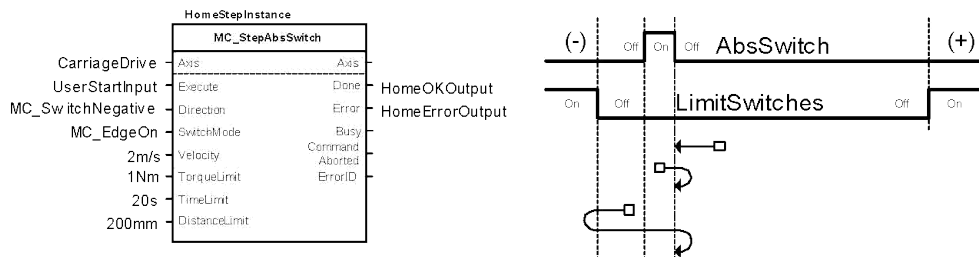


Figure 3-17: MCFB_StepAbsSwitch Usage 3

An overlapping switch configuration is also possible.
This has same the behavior as working on the limit switches:

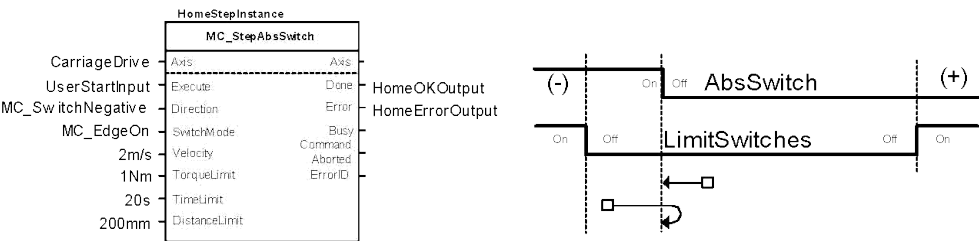


Figure 3-18: MCFB_StepAbsSwitch Usage 4

If the input Direction is set to a fixed direction (MC_Positive or MC_Negative), then the initial switch state is ignored.
Used for example in rotary axis where only one sense of rotation is allowed:

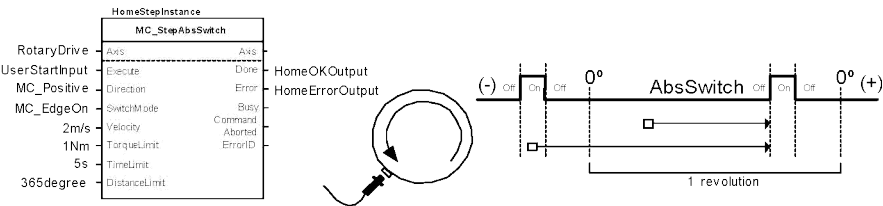


Figure 3-19: MCFB_StepAbsSwitch Usage 5

Inputs

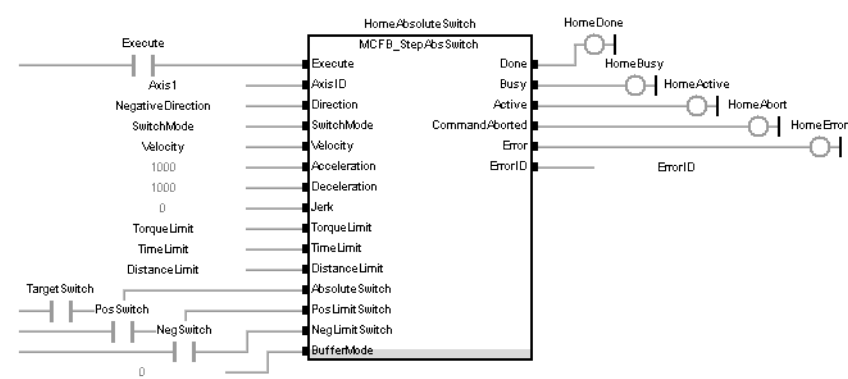
Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	- Request the homing step procedure at the rising edge. - Outputs are reset when execute input is FALSE.
AxisID	AXIS_REF	1 to 256	N/A	No default	- Name of a declared instance of the AXIS_REF library function. - See AXIS_REF Structure.
Direction	DINT	0 to 3	N/A	No default	Defines the axis homing direction. 0 (zero) = Clockwise rotation. 1 = Counterclockwise rotation. 2 = Clockwise if AbsoluteSwitch starts Off and negative if switch starts On. 3 = Counter-clockwise if AbsoluteSwitch starts On and positive if switch starts Off.
SwitchMode	DINT	0 to 3	N/A	No default	Switch state to complete homing. 0 (zero) = Switch is on. 1 = Switch is off. 2 = Rising edge of switch. 3 = Falling edge of switch.

Input	Data Type	Range	Unit	Default	Description
Velocity	LREAL	No range	User units / sec	No default	Commanded velocity for the homing move.
Acceleration	LREAL	No range	User units / sec ²	No default	Commanded acceleration for the homing move.
Deceleration	LREAL	No range	User units / sec ²	No default	Commanded deceleration for the homing move.
Jerk	LREAL	No range	User units / sec ³	No default	• Commanded jerk for the homing move. • If 0 (zero), trapezoidal acc/dec is used.
TorqueLimit	LREAL	No range	User units	No default	Maximum torque applied for the homing move. • Entered in thousandths of maximum torque. • Example: 250 is 250/1000, or 25%.
TimeLimit	TIME	No range	Sec	No default	Maximum time for homing move to complete. • If exceeded, the homing procedure errors out. • 0 (zero) = no time limit.
DistanceLimit	LREAL	No range	User units	No default	Maximum distance for homing move to complete. • If exceeded, the homing procedure errors out. • 0 (zero) = no distance limit.
AbsoluteSwitch	BOOL	FALSE, TRUE	N/A	No default	The absolute switch input I/O point.
PosLimitSwitch	BOOL	FALSE, TRUE	N/A	No default	The positive direction limit switch input I/O point.
NegLimitSwitch	BOOL	FALSE, TRUE	N/A	No default	The negative direction limit switch input I/O point.
SwitchMode	SINT	0 to 5	N/A	No default	Switch state to complete homing. • 0 = Abort. • 1 = Buffer. • 2 = Blend to active. • 3 = Blend to next. • 4 = Blend to low velocity. • 5 = Blend to high velocity.

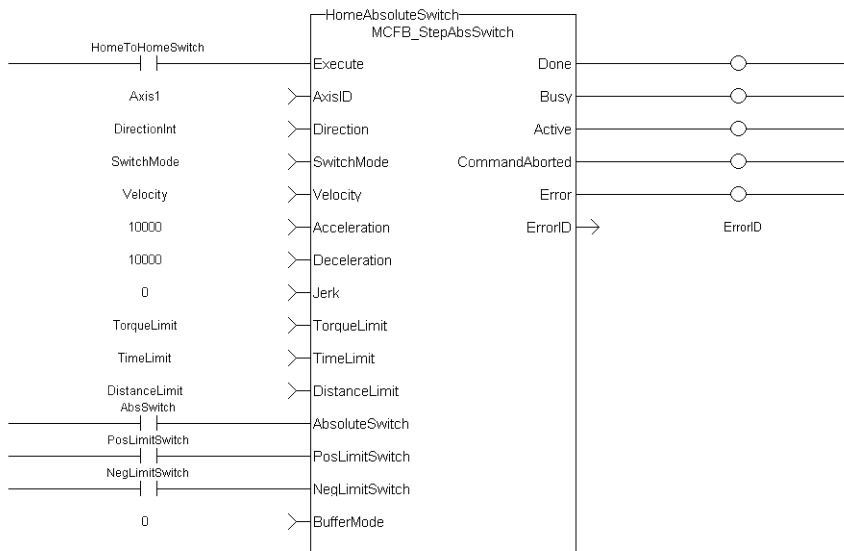
Outputs

Output	Data Type	Range	Unit	Description														
Done	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.														
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input goes high until the time the move is ended.														
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.														
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.														
Error	BOOL	FALSE, TRUE	N/A	Indicates either: • An invalid input was specified. • The move was terminated due to an error.														
ErrorID	INT	Enumerated	N/A	Indicates the error if Error output is TRUE. Error identifier: <table><tr><th>ErrorID</th><th>Description</th></tr><tr><td>1</td><td>TimeLimit exceeded.</td></tr><tr><td>2</td><td>DistanceLimit exceeded.</td></tr><tr><td>3</td><td>TorqueLimit exceeded.</td></tr><tr><td>4</td><td>Axis error stop state.</td></tr><tr><td>5</td><td>Axis not enabled.</td></tr><tr><td>6</td><td>Invalid inputs for Velocity-Acceleration-Deceleration.</td></tr></table>	ErrorID	Description	1	TimeLimit exceeded.	2	DistanceLimit exceeded.	3	TorqueLimit exceeded.	4	Axis error stop state.	5	Axis not enabled.	6	Invalid inputs for Velocity-Acceleration-Deceleration.
ErrorID	Description																	
1	TimeLimit exceeded.																	
2	DistanceLimit exceeded.																	
3	TorqueLimit exceeded.																	
4	Axis error stop state.																	
5	Axis not enabled.																	
6	Invalid inputs for Velocity-Acceleration-Deceleration.																	

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```

NegativeDirection :=1;
RisingEdge :=2;
Velocity :=10000.0;
TorqueLimit :=50.0;
TimeLimit :=T#10s;
DistanceLimit :=10000.0;

Inst_MCFB_StepAbsSwitch( True, Axis1, NegativeDirection, RisingEdge,
Velocity, 1000, 1000, 0, TorqueLimit, TimeLimit, DistanceLimit,
AbsoluteSwitch, PosLimitSwitch, NegLimitSwitch, 0 );

HomeComplete :=Inst_MCFB_StepAbsSwitch.Done;
HomeBusy :=Inst_MCFB_StepAbsSwitch.Busy;
HomeActive :=Inst_MCFB_StepAbsSwitch.Active;
HomeAborted :=Inst_MCFB_StepAbsSwitch.CommandAborted;
HomeError :=Inst_MCFB_StepAbsSwitch.Error;
HomeErrorID :=Inst_MCFB_StepAbsSwitch.ErrorID;
(* AbsoluteSwitch, PosLimitSwitch, NegLimitSwitch are declared I/O points *)

```

See Also

- "MCFB_StepAbsolutes" (→ p. 215)
- "MCFB_StepBlock" (→ p. 222)
- "MCFB_StepLimitSwitch" (→ p. 226)
- "MCFB_StepRefPulse" (→ p. 231)

2.5.5.3 MCFB_StepBlock



Function Block - Homing to a physical object, mechanically blocking the movement.

- Performs homing against a physical object, mechanically blocking the movement.
- In this mode, there is no limit switch or Reference Pulse.
- Adequate torque limits are required for not damaging mechanics during homing process.
- The StepBlock condition is that we have reached the torque limit and real velocity falls below 5% of demanded.

2.5.5.3.1.1 Usage

Homing against a physical object, mechanically blocking the movement require adequate torque limits for not damaging mechanics during homing process.

- This function block does not modify the actual position.
- Home is commanded by the user in the designated homing direction at the selected or programmed Velocity.
- The Torque is limited.
- Time and Distance Limits can cause an error if exceeded.
- Process is finished when Torque is in limit condition and real velocity is below 5% of selected Velocity.

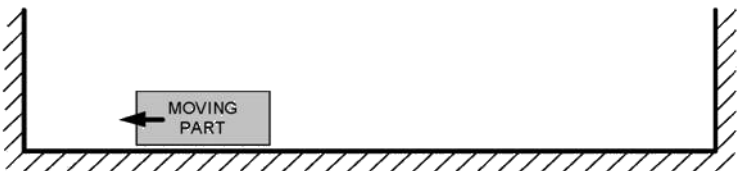


Figure 3-20: MCFB_StepBlock Usage 1

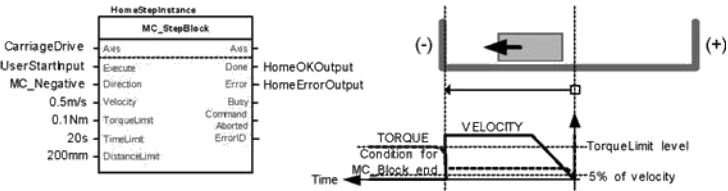


Figure 3-21: MCFB_StepBlock Usage 2

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	• Request the homing step procedure at the rising edge. • Outputs are reset when execute input is FALSE.
AxisID	AXIS_REF	1 to 256	N/A	No default	• Name of a declared instance of the AXIS_REF library function. • See AXIS_REF Structure.
Direction	DINT	0 to 1	N/A	No default	Defines the axis homing direction. • 0 (zero) = Clockwise rotation. • 1 = Counterclockwise rotation.

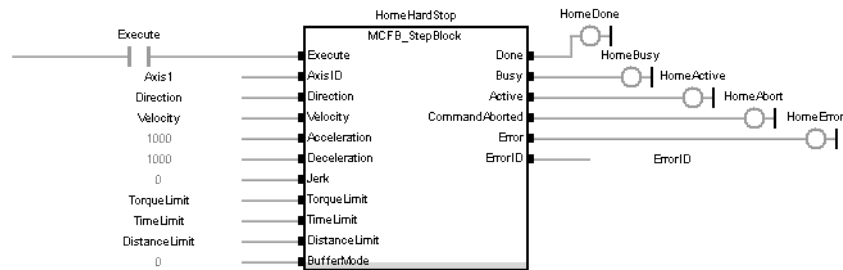
Input	Data Type	Range	Unit	Default	Description
Velocity	LREAL	No range	User units / sec	No default	Commanded velocity for the homing move.
Acceleration	LREAL	No range	User units / sec ²	No default	Commanded acceleration for the homing move.
Deceleration	LREAL	No range	User units / sec ²	No default	Commanded deceleration for the homing move.
Jerk	LREAL	No range	User units / sec ³	No default	- Commanded jerk for the homing move. - If 0 (zero), trapezoidal acc/dec is used.
TorqueLimit	LREAL	No range	User units	No default	Maximum torque applied for the homing move. - Entered in thousandths of maximum torque. - Example: 250 is 250/1000, or 25%.
TimeLimit	TIME	No range	Sec	No default	Maximum time for homing move to complete. - If exceeded, the homing procedure errors out. 0 (zero) = no time limit.
DistanceLimit	LREAL	No range	User units	No default	Maximum distance for homing move to complete. - If exceeded, the homing procedure errors out. 0 (zero) = no distance limit.
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. - BM_ABORTING - BM_BUFFERED - BM_BLENDING_PREVIOUS - BM_BLENDING_NEXT - BM_BLENDING_LOW - BM_BLENDING_HIGH - See Buffer Modes.

Outputs

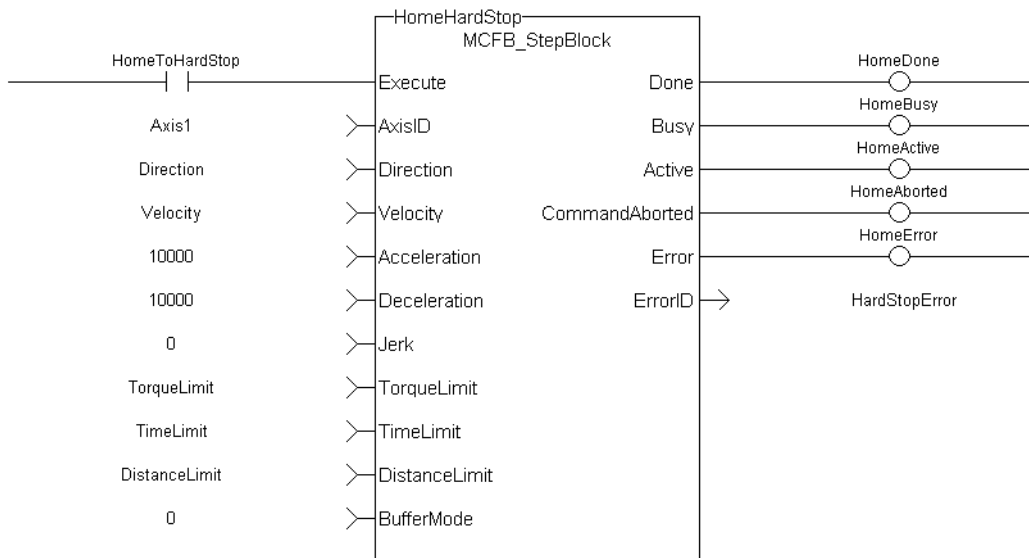
Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input goes high until the time the move is ended.

Output	Data Type	Range	Unit	Description														
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.														
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.														
Error	BOOL	FALSE, TRUE	N/A	Indicates either: • An invalid input was specified. • The move was terminated due to an error.														
ErrorID	INT	Enumerated	N/A	Indicates the error if Error output is TRUE. Error identifier: <table><tr><th>ErrorID</th><th>Description</th></tr><tr><td>1</td><td>TimeLimit exceeded.</td></tr><tr><td>2</td><td>DistanceLimit exceeded.</td></tr><tr><td>3</td><td>TorqueLimit exceeded.</td></tr><tr><td>4</td><td>Axis error stop state.</td></tr><tr><td>5</td><td>Axis not enabled.</td></tr><tr><td>6</td><td>Invalid inputs for Velocity-Acceleration-Deceleration.</td></tr></table>	ErrorID	Description	1	TimeLimit exceeded.	2	DistanceLimit exceeded.	3	TorqueLimit exceeded.	4	Axis error stop state.	5	Axis not enabled.	6	Invalid inputs for Velocity-Acceleration-Deceleration.
ErrorID	Description																	
1	TimeLimit exceeded.																	
2	DistanceLimit exceeded.																	
3	TorqueLimit exceeded.																	
4	Axis error stop state.																	
5	Axis not enabled.																	
6	Invalid inputs for Velocity-Acceleration-Deceleration.																	

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```

PositiveDirection :=0;
Velocity :=10000.0;
TorqueLimit :=50.0;
TimeLimit :=T#10s;
DistanceLimit :=10000.0;

Inst_MCFB_StepBlock( True, Axis1, PositiveDirection, Velocity, 1000, 1000, 0,
TorqueLimit, TimeLimit, DistanceLimit, 0 );

HomeComplete :=Inst_MCFB_StepBlock.Done;
HomeBusy :=Inst_MCFB_StepBlock.Busy;
HomeActive :=Inst_MCFB_StepBlock.Active;
HomeAborted :=Inst_MCFB_StepBlock.CommandAborted;
HomeError :=Inst_MCFB_StepBlock.Error;
HomeErrorID :=Inst_MCFB_StepBlock.ErrorID;

```

See Also

- "MCFB_StepAbsolutes" (→ p. 215)
- "MCFB_StepAbsSwitch" (→ p. 217)
- "MCFB_StepLimitSwitch" (→ p. 226)
- "MCFB_StepRefPulse" (→ p. 231)

2.5.5.4 MCFB_StepLimitSwitch



 **Function Block** - Homing to a limit switch.

- Performs a single-axis home to a limit switch.
- The limit switches are used for homing procedure.
 - They are always active once moving part working area has been surpassed.

2.5.5.4.1.1 Usage

- This function block does not modify the actual position.
- Home is commanded by the user in the designated homing direction at the selected or programmed Velocity.
- This procedure performs a homing function searching for sensor using only LimitSwitches.
 - A LimitSwitch has one Off (or On) area.
- If **LimitSwitch** is found On with a rising Execute, the process is started in the opposite direction as specified.
- LimitSwitch is searched for Off Edge (released) and process is restarted again in original direction.
 - Or On, depending on LimitSwitchMode setting.
 - This ensures the end conditions are always the same.
- The Torque is limited.
- Time and Distance Limits can cause an error if exceeded.
- The direction changes only when the specified Velocity is reached.
 - This ensures acceleration and deceleration spaces are fixed.

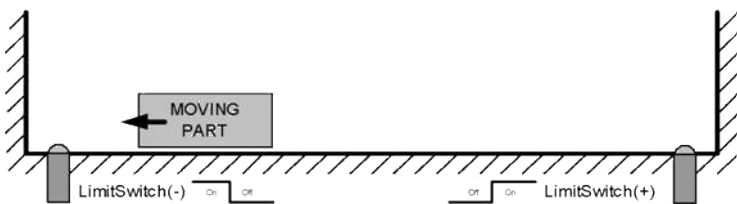


Figure 3-22: MCFB_StepLimitSwitch Usage 1

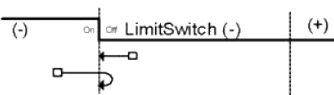


Figure 3-23: MCFB_StepLimitSwitch Usage 2

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	• Request the homing step procedure at the rising edge. • Outputs are reset when execute input is FALSE.
AxisID	AXIS_REF	1 to 256	N/A	No default	• Name of a declared instance of the AXIS_REF library function. • See AXIS_REF Structure.

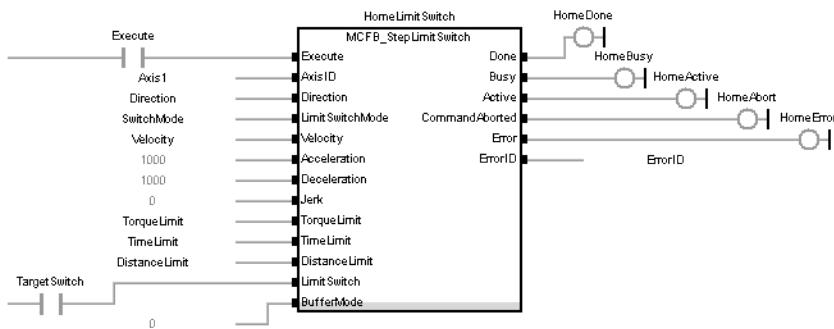
Input	Data Type	Range	Unit	Default	Description
Direction	BOOL	FALSE, TRUE	N/A	No default	Defines the axis homing direction. 0 (zero) = Clockwise rotation. 1 = Counterclockwise rotation.
LimitSwitchMode	DINT	0 to 3	N/A	No default	Limit switch state to complete homing. 0 (zero) = Switch is on. 1 = Switch is off. 2 = Rising edge of switch. 3 = Falling edge of switch.
Velocity	LREAL	No range	User units / sec	No default	Commanded velocity for the homing move.
Acceleration	LREAL	No range	User units / sec ²	No default	Commanded acceleration for the homing move.
Deceleration	LREAL	No range	User units / sec ²	No default	Commanded deceleration for the homing move.
Jerk	LREAL	No range	User units / sec ³	No default	- Commanded jerk for the homing move. - If 0 (zero), trapezoidal acc/dec is used.
TorqueLimit	LREAL	No range	User units	No default	Maximum torque applied for the homing move. - Entered in thousandths of maximum torque. - Example: 250 is 250/1000, or 25%.
TimeLimit	TIME	No range	Sec	No default	Maximum time for homing move to complete. - If exceeded, the homing procedure errors out. 0 (zero) = no time limit.
DistanceLimit	LREAL	No range	User units	No default	Maximum distance for homing move to complete. - If exceeded, the homing procedure errors out. 0 (zero) = no distance limit.
LimitSwitch	BOOL	FALSE, TRUE	N/A	No default	The limit switch input I/O point.

Input	Data Type	Range	Unit	Default	Description
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. - BM_ABORTING - BM_BUFFERED - BM_BLENDING_PREVIOUS - BM_BLENDING_NEXT - BM_BLENDING_LOW - BM_BLENDING_HIGH - See Buffer Modes.

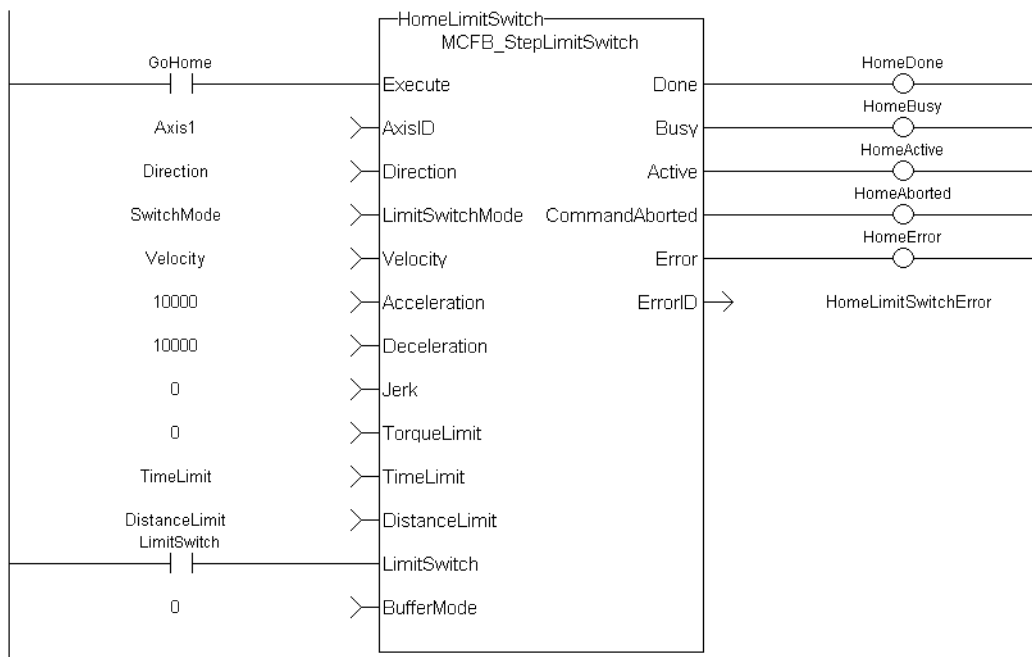
Outputs

Output	Data Type	Range	Unit	Description														
Done	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.														
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input goes high until the time the move is ended.														
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.														
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.														
Error	BOOL	FALSE, TRUE	N/A	Indicates either: • An invalid input was specified. • The move was terminated due to an error.														
ErrorID	INT	Enumerated	N/A	Indicates the error if Error output is TRUE. Error identifier: <table><tr><th>ErrorID</th><th>Description</th></tr><tr><td>1</td><td>TimeLimit exceeded.</td></tr><tr><td>2</td><td>DistanceLimit exceeded.</td></tr><tr><td>3</td><td>TorqueLimit exceeded.</td></tr><tr><td>4</td><td>Axis error stop state.</td></tr><tr><td>5</td><td>Axis not enabled.</td></tr><tr><td>6</td><td>Invalid inputs for Velocity-Acceleration-Deceleration.</td></tr></table>	ErrorID	Description	1	TimeLimit exceeded.	2	DistanceLimit exceeded.	3	TorqueLimit exceeded.	4	Axis error stop state.	5	Axis not enabled.	6	Invalid inputs for Velocity-Acceleration-Deceleration.
ErrorID	Description																	
1	TimeLimit exceeded.																	
2	DistanceLimit exceeded.																	
3	TorqueLimit exceeded.																	
4	Axis error stop state.																	
5	Axis not enabled.																	
6	Invalid inputs for Velocity-Acceleration-Deceleration.																	

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
PositiveDirection :=0;
RisingEdge :=2;
Velocity :=10000.0;
TorqueLimit :=50.0;
TimeLimit :=T#10s;
DistanceLimit :=10000.0;

Inst_MCFB_StepLimitSwitch( True, Axis1, PositiveDirection, RisingEdge,
Velocity, 1000, 1000, 0, TorqueLimit, TimeLimit, DistanceLimit, LimitSwitch,
0 );
```

```
HomeComplete :=Inst_MCFB_StepLimitSwitch.Done;  
HomeBusy :=Inst_MCFB_StepLimitSwitch.Busy;  
HomeActive :=Inst_MCFB_StepLimitSwitch.Active;  
HomeAborted :=Inst_MCFB_StepLimitSwitch.CommandAborted;  
HomeError :=Inst_MCFB_StepLimitSwitch.Error;  
HomeErrorID :=Inst_MCFB_StepLimitSwitch.ErrorID;  
  
(* LimitSwitch is a declared I/O point *)
```

See Also

- "MCFB_StepAbsolutes" (→ p. 215)
- "MCFB_StepAbsSwitch" (→ p. 217)
- "MCFB_StepBlock" (→ p. 222)
- "MCFB_StepRefPulse" (→ p. 231)

2.5.5.5 MCFB_StepRefPulse



Function Block - Homing to a zero angle reference.

- This function block modifies actual position and sets to the SetPosition input value at the end.
- Performs homing by searching for Zero pulse in encoder.
 - Also called Marker or Reference Pulse.
- The pulse appears once per encoder revolution.
- The advantage in using Reference Pulse for homing is the higher accuracy and precision achieved compared to traditional optical, mechanical, or magnetic sensors.

2.5.5.5.1.1 Usage

- Home is commanded by the user in the designated homing direction at the selected or programmed Velocity.
- First occurrence of the Reference Pulse, Homing is finished.
- The Torque is limited.
- Time and Distance Limits can cause an error if exceeded.

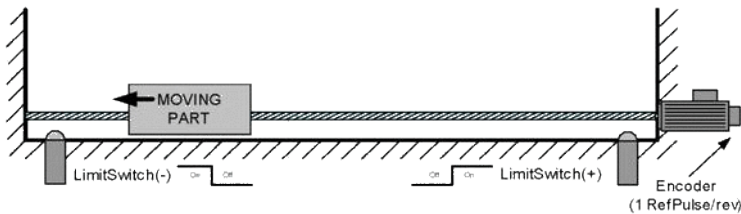


Figure 3-24: MCFB_StepRefPulse Usage 1

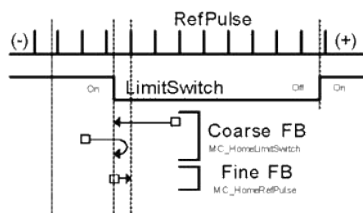


Figure 3-25: MCFB_StepRefPulse Usage 2

- It is common that a first approach is performed against a mechanical sensor at higher velocity, and after a Reference Pulse, at a lower velocity.
- This is a traditional 2-Step homing (Coarse by external Switch in reverse and Fine by Reference Pulse in forward).

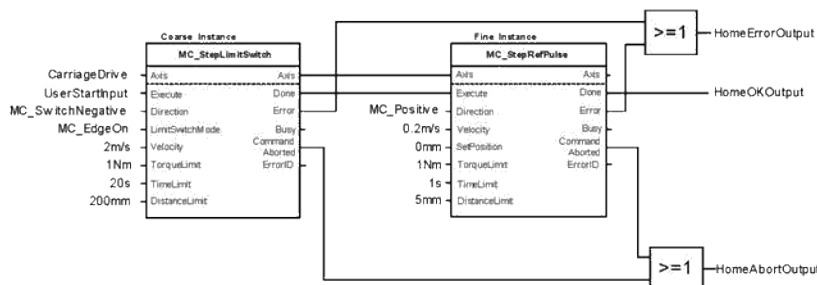


Figure 3-26: MCFB_StepRefPulse Usage 3

Inputs

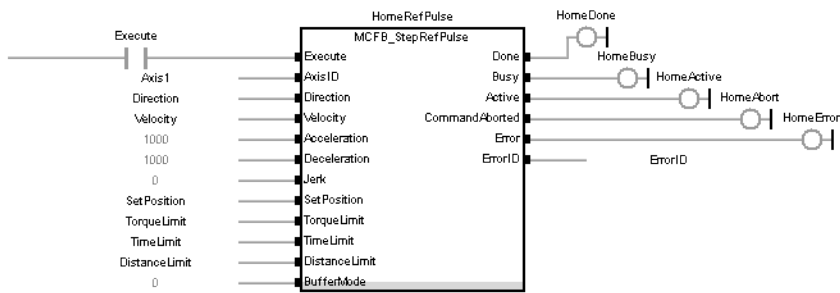
Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	Request the homing step procedure at the rising edge.
AxisID	AXIS_REF	1 to 256	N/A	No default	- Name of a declared instance of the AXIS_REF library function. - See AXIS_REF Structure.
Direction	BOOL	FALSE, TRUE	N/A	No default	Defines the axis homing direction. 0 (zero) = Clockwise rotation. 1 = Counterclockwise rotation.
Velocity	LREAL	No range	User units / sec	No default	Commanded velocity for the homing move.
Acceleration	LREAL	No range	User units / sec ²	No default	Commanded acceleration for the homing move.
Deceleration	LREAL	No range	User units / sec ²	No default	Commanded deceleration for the homing move.
Jerk	LREAL	No range	User units / sec ³	No default	- Commanded jerk for the homing move. - If 0 (zero), trapezoidal acc/dec is used.
SetPosition	LREAL	No range	User units	No default	Value of the absolute position to be set when the homing move is done.
TorqueLimit	LREAL	No range	User units	No default	Maximum torque applied for the homing move. - Entered in thousandths of maximum torque. - Example: 250 is 250/1000, or 25%.
TimeLimit	TIME	No range	Sec	No default	Maximum time for homing move to complete. - If exceeded, the homing procedure errors out. 0 (zero) = no time limit.
DistanceLimit	LREAL	No range	User units	No default	Maximum distance for homing move to complete. - If exceeded, the homing procedure errors out. 0 (zero) = no distance limit.

Input	Data Type	Range	Unit	Default	Description
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. • BM_ABORTING • BM_BUFFERED • BM_BLENDING_PREVIOUS • BM_BLENDING_NEXT • BM_BLENDING_LOW • BM_BLENDING_HIGH • See Buffer Modes.

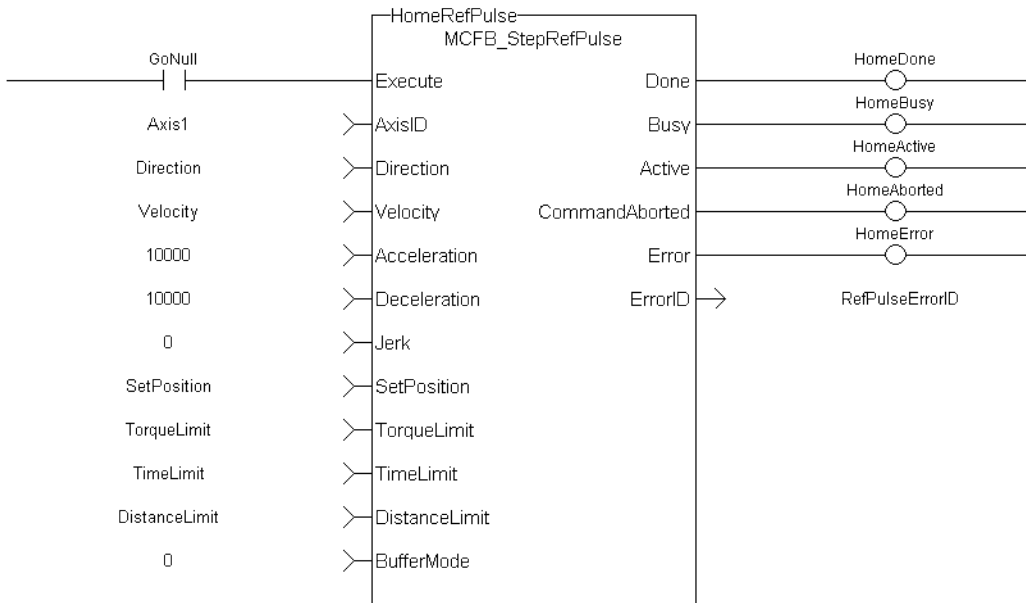
Outputs

Output	Data Type	Range	Unit	Description														
Done	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.														
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input goes high until the time the move is ended.														
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.														
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.														
Error	BOOL	FALSE, TRUE	N/A	Indicates either: • An invalid input was specified. • The move was terminated due to an error.														
ErrorID	INT	Enumerated	N/A	Indicates the error if Error output is TRUE. Error identifier: <table><tr><th>ErrorID</th><th>Description</th></tr><tr><td>1</td><td>TimeLimit exceeded.</td></tr><tr><td>2</td><td>DistanceLimit exceeded.</td></tr><tr><td>3</td><td>TorqueLimit exceeded.</td></tr><tr><td>4</td><td>Axis error stop state.</td></tr><tr><td>5</td><td>Axis not enabled.</td></tr><tr><td>6</td><td>Invalid inputs for Velocity-Acceleration-Deceleration.</td></tr></table>	ErrorID	Description	1	TimeLimit exceeded.	2	DistanceLimit exceeded.	3	TorqueLimit exceeded.	4	Axis error stop state.	5	Axis not enabled.	6	Invalid inputs for Velocity-Acceleration-Deceleration.
ErrorID	Description																	
1	TimeLimit exceeded.																	
2	DistanceLimit exceeded.																	
3	TorqueLimit exceeded.																	
4	Axis error stop state.																	
5	Axis not enabled.																	
6	Invalid inputs for Velocity-Acceleration-Deceleration.																	

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
PositiveDirection :=0;
Velocity :=10000.0;
SetPosition :=0.0;
TorqueLimit :=50.0;
TimeLimit :=T#10s;
DistanceLimit :=10000.0;
```

```
Inst_MCFB_StepRefPulse( True, Axis1, PositiveDirection, Velocity, 1000, 1000,
0, SetPosition, TorqueLimit, TimeLimit, DistanceLimit, 0 );
```

```
HomeComplete :=Inst_MCFB_StepRefPulse.Done;
HomeBusy :=Inst_MCFB_StepRefPulse.Busy;
HomeActive :=Inst_MCFB_StepRefPulse.Active;
```

```
HomeAborted :=Inst_MCFB_StepRefPulse.CommandAborted;  
HomeError :=Inst_MCFB_StepRefPulse.Error;  
HomeErrorID :=Inst_MCFB_StepRefPulse.ErrorID;
```

See Also

- "MCFB_StepAbsolutes" (→ p. 215)
- "MCFB_StepAbsSwitch" (→ p. 217)
- "MCFB_StepBlock" (→ p. 222)
- "MCFB_StepLimitSwitch" (→ p. 226)

2.5.6 MCFB_Jog



 **Function** - Define to jog an axis in the selected direction at a defined speed.

- This function or function block can be used with PLCopen Motion engines.
- This function or function block is used to command motion in a designated direction at a defined rate.
 - This may be used where continuous motion is required (e.g., a conveyor system or in a setup mode for manually jogging the axis).
- The **JogPlus** and **JogMinus** inputs select the direction the motion occurs in.
 - Motion starts when the **JogPlus** or **JogMinus** input is TRUE.
 - Motion stops when the input is FALSE.
 - Only one of these inputs should be enabled at a time.
 - If both are selected, the motion stops.
- If another motion is active when the jog is requested, that motion is aborted and the jog starts.
- The En input (FFLD editor only) must be high.
 - Typically wired to the rail.

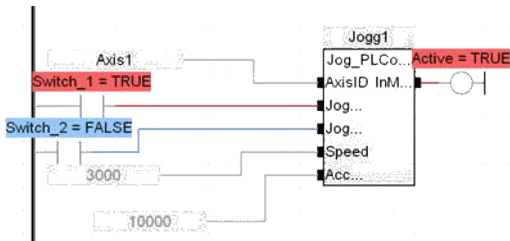
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	• Enables execution. • Used in the FFLD editor only.
AxisID	DINT	No range	N/A	No default	• ID of the Axis block. • The AxisID selects the axis to jog.
JogPlus	BOOL	FALSE, TRUE	N/A	No default	Enables a jog in the plus direction.
JogMinus	BOOL	FALSE, TRUE	N/A	No default	Enables a jog in the minus direction.
Speed	LREAL	No range	User units / sec	No default	Rate the axis moves.
Accel Decel	LREAL	No range	User units / sec ²	No default	Linear Acc/Dec rate.

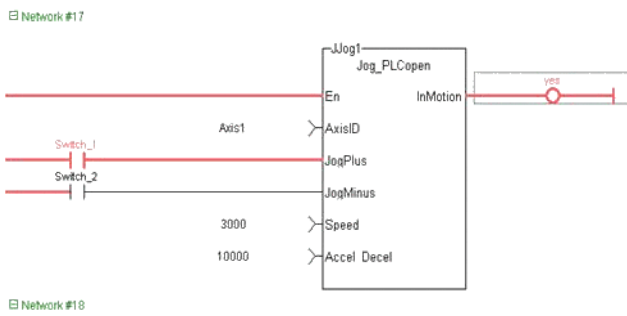
Outputs

Output	Data Type	Range	Unit	Description
InMotion	BOOL	FALSE, TRUE	N/A	Jogging is active when TRUE.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
InMotion := Inst_Jog_PLCOpen(Axis1, Switch_1, Switch_2, 600, 10000);
```

See Also

"MLAxisMoveVel" (→ p. 352)

2.5.7 MCFB_GearedWebTension



 **Function Block** - Facilitates dancer and tension control in an electronic geared master/slave machine design.

- This facilitation is done by using the analog feedback from an LVDT, tension transducer, potentiometer, encoder, resolver or some other similar device.
 - The analog feedback value is compared to a pre-determined analog set-point.
- The difference or error is used in a PID algorithm with the summed output driving changes to the master/slave gearing relationship.
 - This results in the slave axis either speeding up or slowing down to maintain desired tension.

2.5.7.0.1 Usage

- This UDFB is used in conjunction with the main ladder "MC_GearIn" (→ p. 710) function.
 - It is assumed that the master/slave move is active.
- Internal to the UDFB is another call to the **MC_GearIn** function.
 - Therefore, the MasterID, SlaveID, RatioNumerator, RatioDenominator, Acceleration, Deceleration, and Jerk inputs are the same values as the main ladder **MC_GearIn** function input values, both with the Buffer input of 0 (zero).
 - This assures that the initial starting master/slave ratio will transition to the new KollmorgenUDFB ratio smoothly.
- This UDFB changes the master/slave ratio defined by the **MC_GearIn** function based on the error between the analog input and the analog set-point.
 - The magnitude of the ratio and the rate of the ratio change is defined by the Kp, Ti, Td PID gain values.
 - The new ratio calculated is output at the NewRatio output.
- The **RatioLimitPercent** input is the maximum and minimum theoretical new ratio that can be changed.
 - This provides a +/- window limit around the running ratio to prevent unwanted motion in the event of a web break or analog feedback failure.

[Click to view Usage Example 1](#)

NOTE

This example assumes the analog feedback device is located **after** (or downstream in the process) the feedroll axis.

```
RatioNumerator = 1
RatioDenominator = 2      Therefore the master/slave starting ratio is
0.5000000
ErrorCaclMode = 0
DeviceFBValue = 6
DeviceSetPoint = 4      Therefore error 6 - 4 = 2
Kp = 0.005
Ti = 0
Td= 0
```

From the equation:

$$\text{New RatioDenominator} = (\text{RatioDenominator} - Kp * \text{error})$$

- The new RatioDenominator = $(2 - 0.005 * 2) = 1.99$.
- The new master/slave running ratio is $1 / 1.99 = 0.502512562$.

- Since the master/slave ratio is greater than the previous ratio, the slave axis is going faster and the tension is reduced.

[Click to view Usage Example 2](#)

NOTE

This example assumes the analog feedback device is located **before** (or upstream in the process) the feedroll axis.

```
RatioNumerator = 1
RatioDenominator = 2      Therefore the master/slave starting ratio is
0.5000000
ErrorCaclMode = 1
DeviceFBValue = 6
DeviceSetPoint = 4      Therefore error is 4 - 6 = -2
Kp = 0.005
Ti = 0
Td= 0
```

From the equation:

New RatioDenominator = (RatioDenominator - (Kp * error))

- The new RatioDenominator = $(2 + 0.005 * 2) = 2.01$.
- The new master/slave running ratio is $1 / 2.01 = 0.497512437$.
- Since the master/slave ratio is less than the previous ratio, the slave axis is going slower and the tension is reduced.

2.5.7.0.2 PID Function in KAS

There is a PID function in KAS that could be used for the PID control section in the UDFB.

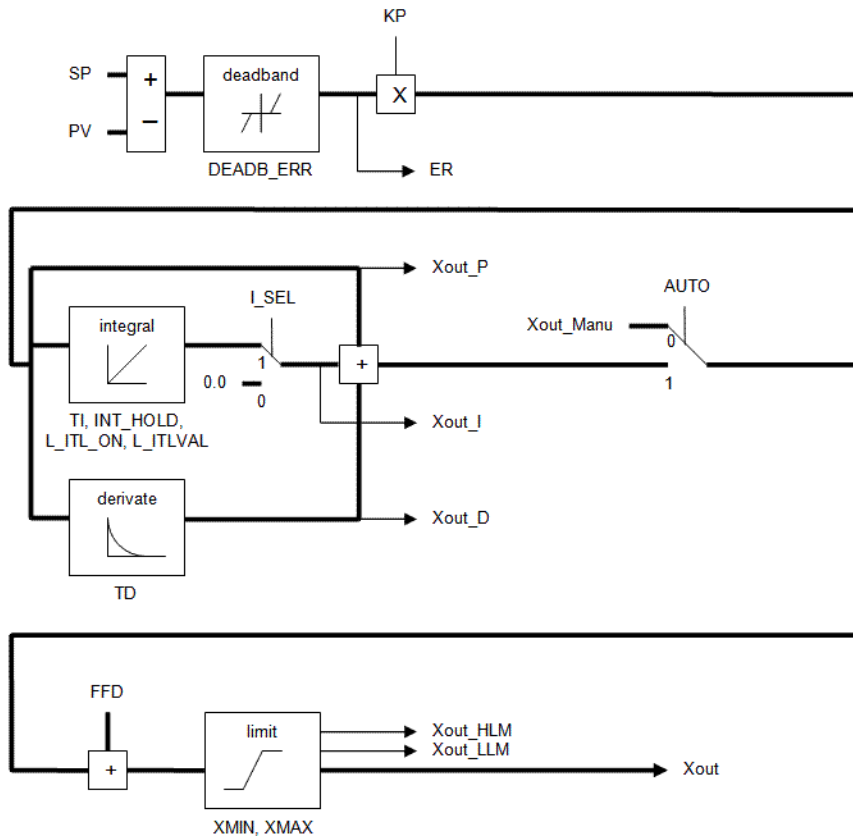


Figure 3-27: PID Function

2.5.7.0.3 Programming Tips

- The "FB_FirstOrderDigitalFilter" (→ p. 248)UDFB can be used to decrease excess dither on the analog input.
 - The filtered analog value is then used at the DeviceFBValue input of the MCFB_GearedWebTension UDFB.
- The assumption is an **MC_GearIn** function block is first called in the main ladder and these initial values are used at the inputs for the UDFB.
 - The resolution of the initial **MC_GearIn** the **RatioNumerator** and **RatioDenominator** inputs are directly related to the resolution of the calculated master/slave ratio (from the UDFB inputs) and may need to be scaled accordingly.

[Click to view PID Example 1](#)

- No scaling
Initial MC_GearIn input RatioNumerator = 2.
Initial MC_GearIn input RatioDenominator = 1 then initial Master/Slave ratio = 2.
- UDFB input RatioNumerator = 2.
UDFB input RatioDenominator = 1 then UDFB Master/Slave ratio = 2.
- UDFB input DeviceFBValue = 4.
UDFB input DeviceFBSetpoint = 3 then Device PID error = 1 assume KP = 1, Ti and Td =0.
- New UDFB RatioNumerator = Current RatioNumerator - PID error = 2 - 1 = 1 then new UDFB Master/Slave ratio = 1.

Resolution = Master/Slave ratio:PID Error ratio = 1:1.

- The resolution is so coarse that a change of 1 for the error output of the PID creates a Master/Slave ratio change of 1.

- This result is a significant change to the slave velocity that will probably cause excess slack or web breakage.

[Click to view PID Example 2](#)

- Scaling value = 1000
Initial MC_GearIn input RatioNumerator = 2.
Initial MC_GearIn input RatioDenominator = 1 then initial Master/Slave ratio = 2.
- UDFB input RatioNumerator = 2000
UDFB input RatioDenominator = 1000 then UDFB Master/Slave ratio = 2.
- UDFB input DeviceFBValue = 4.
UDFB input DeviceFBSetpoint = 3 then Device PID error = 1 assume KP = 1, Ti and Td = 0.
- New UDFB RatioNumerator = Current RatioNumerator - PID error = 2000 - 1 = 1999 then new UDFB Master/Slave ratio = 1999.

Resolution = Master/Slave ratio: PID Error ratio = 2000:1.

- This resolution is much finer than Example 1.
- For a change of 1 for the error output of the PID this creates a Master/Slave ratio change of 1999.
- This results in a slower rate of change to the slave velocity that is more suited to good tension in a machine process.

Inputs

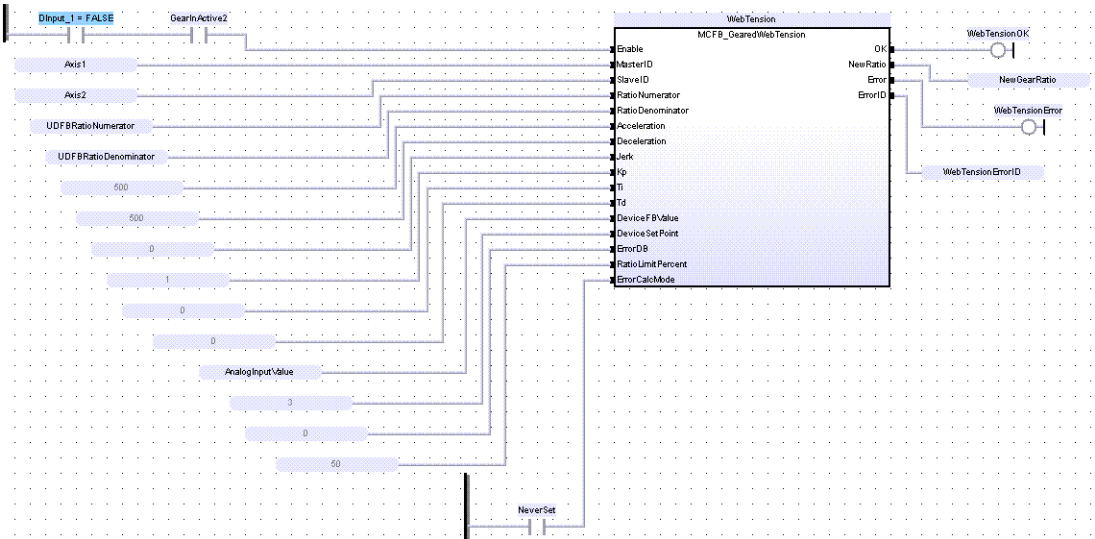
Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	Enables execution.
MasterID	AXIS_REF	N/A	N/A	No default	Master axis identifier.
SlaveID	AXIS_REF	N/A	N/A	No default	Slave axis identifier.
RatioNumerator	DINT	-2147483648 to 2147483647	N/A	No default	Numerator of slave/master ratio.
RatioDenominator	DINT	-2147483648 to 2147483647	N/A	No default	Denominator of slave/master ratio.
Acceleration	LREAL	-1.7E308 to 1.7E308	N/A	No default	Trapezoidal: Acceleration rate. • S-curve: Maximum acceleration. • 14 to 15 significant digits of accuracy.
Deceleration	LREAL	-1.7E308 to 1.7E308	N/A	No default	Trapezoidal: Deceleration rate. • S-curve: Unused. • 14 to 15 significant digits of accuracy.
Jerk	LREAL	-1.7E308 to 1.7E308	N/A	No default	Trapezoidal: 0 (zero). • S-curve: Constant jerk. • 14 to 15 significant digits of accuracy.

Input	Data Type	Range	Unit	Default	Description
Kp	LREAL	-1.7E308 to 1.7E308	N/A	No default	- Proportional gain. 14 to 15 significant digits of accuracy.
Ti	LREAL	-1.7E308 to 1.7E308	N/A	No default	- Integral gain. 14 to 15 significant digits of accuracy.
Td	LREAL	-1.7E308 to 1.7E308	N/A	No default	- Derivative gain. 14 to 15 significant digits of accuracy.
DeviceFBValue	DINT	-2147483648 to 2147483647	N/A	No default	Analog input.
DeviceSetPoint	DINT	-2147483648 to 2147483647	N/A	No default	Analog set point.
ErrorDB	LREAL	-1.7E308 to 1.7E308	N/A	No default	- Maximum or minimum error between DeviceFBValue and DeviceSetPoint before a change takes place. 14 to 15 significant digits of accuracy.
RatioLimitPercent	LREAL	-1.7E308 to 1.7E308	N/A	No default	- Maximum and minimum master/slave ratio window. 14 to 15 significant digits of accuracy.
ErrorCalcMode	BOOL	FALSE, TRUE	N/A	No default	- Not set: - DeviceFBValue-DeviceSetPoint - Set: - DeviceSetPoint-DeviceFBValue

Outputs

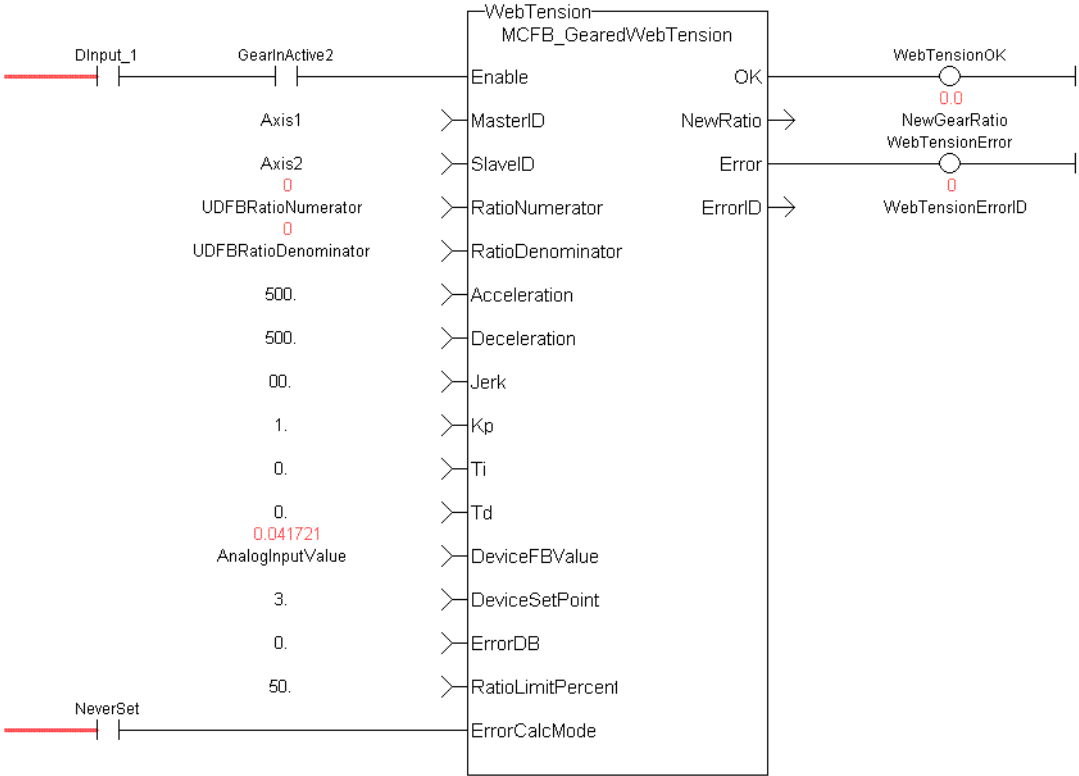
Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	The output has power flow after the enable input has been energized.
NewRatio	REAL	-3.4E38 to 3.4E38	N/A	New master/slave ratio. 6 to 7 significant digits of accuracy.
Error	BOOL	FALSE, TRUE	N/A	Function block error.
ErrorID	INT	-32768 to +32767	N/A	Function block error value.

FBD Language Example



FFLD Language Example

Network #17



IL Language Example

Not available.

ST Language Example

```
Inst_MCFB_GearedWebTension( DInput_1, Axis1, Axis2,  
    ratioNumerator, ratioDenominator, 500.0, 500.0, 0.0, 1.0, 0.0, 0.0,  
    AnalogInputValue, 3.0, 0.0, 50.0, NeverSet);  
WebTensionOK          := Inst_MCFB_GearedWebTension.OK;  
NewGearRatio          := Inst_MCFB_GearedWebTension.NewRatio;  
WebTensionHasError    := Inst_MCFB_GearedWebTension.Error;  
WebTensionErrorID     := Inst_MCFB_GearedWebTension.ErrorID;
```

See Also

"FB_FirstOrderDigitalFilter" (→ p. 248)

2.6 UDFB - PLC

These are the Kollmorgen UDFB PLC function and function blocks.

Name	Description
FB_ElapseTime	Keeps track of the time (oTotalOnTime) a Boolean input variable is on.
FB_FirstOrderDigitalFilter	Defined to filter an Analog signal.
FB_PWDutyOutput	Accepts an input value between the minimum and maximum input range and converts this to a duty cycle percentage.
FB_ScaleInput - Scaling Analog IO	Converts un-scaled DINT values from Analog Inputs into user units of type LREAL.
FB_ScaleOutput - Scaling Analog IO	Converts un-scaled LREAL values from a PLC program into units of type DINT.
FB_TemperaturePID	Provides PID temperature control with auto tuning.
PipeNetwork_FFLD - Special Function	Calls the PNCode function block in FFLD POU's.
ProfilesCode_FFLD - Special Function	Calls the Profiles Code Function Block in FFLD POU's.

2.6.1 FB_ElapseTime

 **Function Block** - Keeps track of the time (oTotalOnTime) a Boolean input variable is on.

- Once the iEN00 enable input is high, the Kollmorgen UDFB keeps track of the total time iVariable is on.
- If iVariable changes to an off state while iEN00 is on, the oTotalOnTime stops.
 - oTotalOnTime will start to add again once iVariable changes state to high.
 - As long as the iEN00 input is on, iVariable can change states many times.
 - The oTotalOnTime will reflect only the total time that iVariable has been on.
- While iVariable is still TRUE, oInProcess is also TRUE and oDone is FALSE.
 - Once iVariable is FALSE, oInProcess is FALSE and oDone is TRUE.
- If the iEN00 input goes off, oTotalOnTime stops counting and the Kollmorgen UDFB execution stops.
 - To restart the timer turn iEN00 on again.
 - This resets oTotalOnTime to zero and counting begins once iVariable is also on.

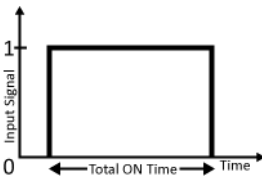


Figure 3-28: MFB_ElapseTime - Time Diagram

Usage

- Enable the block by setting iEN00 to TRUE
- Either manually set iVariable to TRUE or have the application set this variable to TRUE
- Once oDone returns TRUE, read the oTotalOnTime to find out how long iVariable was on.

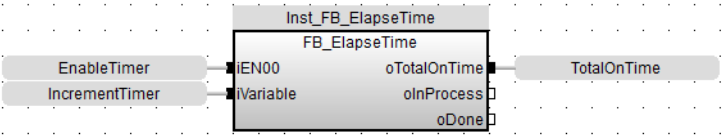
Inputs

Input	Data Type	Range	Unit	Default	Description
iEN00	BOOL	FALSE, TRUE	N/A	FALSE	Enable for the block.
iVariable	BOOL	FALSE, TRUE	N/A	FALSE	The variable to be tracked.

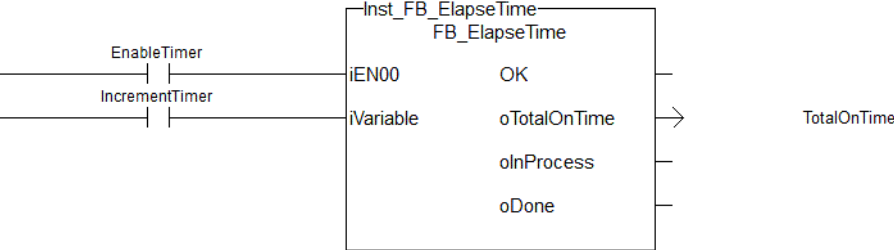
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Function block is OK. • This output follows the state on iEN00 input.
oTotalOnTime	TIME	0ms - 24hr	ms	The amount of time the iVariable is turned on.
oInProcess	BOOL	FALSE, TRUE	N/A	The state of the block's execution whether or not it is still keeping track of time.
oDone	BOOL	FALSE, TRUE	N/A	The state of the block's execution whether or not it is completed.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Keep track of total time that IncrementTimer variable is TRUE while
EnableTimer variable is true
//Timer will be reset when EnableTimer variable is false
Inst_FB_ElapseTime( EnableTimer, IncrementTimer );
TotalOnTime := Inst_FB_ElapseTime.oTotalOnTime;
```

2.6.2 FB_FirstOrderDigitalFilter



Function Block - Defined to filter an Analog signal.

In any control system with an analog feedback signal present, there is the risk of unwanted noise and jitter that can compromise the signal integrity yielding a less desirable system.

- This Kollmorgen UDFB provides a digital first order filter of an analog feedback signal from an LVDT, tension transducer, potentiometer, encoder, resolver, or some other like device.
- The amount of filtering is based on a gain value and can provide no filter to full filter conditioning.

Usage

When using this UDFB, the Enable (EN) input should always be energized to provide the desired filtering.

- The **AnalogInput** input is the unfiltered analog feedback signal from an LVDT, tension transducer, potentiometer, or some other like device.
- The **FilterGain** defines the amount of filtering to be used.
 - The range of the gain is from 1.0 or no filtering to 0.05 or the maximum filtering.
- The **FilterOutput** is the filtered analog input.
 - It is used as an input to some other function block or UDFB that has an analog input (e.g., the MCFB_GearedWebTension UDFB).
- The implementation of the digital first order filter is for PLCopen.
- The equation is defined as: $\text{Input} * \text{Gain} + \text{Output} * (1 - \text{Gain}) = \text{Output}$.
- The steady state filter delay with a gain of 0.8 is shown in the "Filter Input Delay" (→ p. 248) table.

2.6.2.0.1 Filter Input Delay

Example: Filter Input Delay

FilterGain	FilterInput	FilterOutput
0.8	0	0
	100	80
	100	96
	100	99.2
	100	99.84
	100	99.968
	100	99.9936
	100	99.99872
	100	99.999744
	100	99.9999488
	100	99.99998976
	100	99.99999795
	100	99.99999959
	100	99.99999992
	100	99.99999998
	100	100
	100	100

FilterGain	FilterInput	FilterOutput
	100	100
	100	100
	100	100
	100	100
	100	100
	100	100

2.6.2.0.2 Filter Delay Tn

Example Table: filter Delay T_{Yn}

In this table:

- The numbers of filter delays for a steady state analog input at a given gain are listed.
- The range of the filter gain is between 1.00 and 0.05.
- For a filter gain of 0.8 there is a delay of 15 time constants with a time constant defined as the rate the UDFB is scanned or executed in the application.
 - Example: If the UDFB was executed every millisecond a gain of 0.8 would provide a filter delay of 15ms.
 - Conversely a gain of 1.00 provides zero filtering and the output signal follows the input signal, and a gain of 0.05 provides the most filtering for 463 ms.

Gain	Filter Delay Tn	Graph
1.00	0	
0.95	8	
.90	11	
.85	13	
.80	15	
.75	18	
.70	20	
.65	23	
.60	26	
.55	30	
.50	35	
.45	40	
.40	47	
.35	56	
.30	66	
.25	83	
.20	107	
.15	146	
.10	226	
.05	463	

2.6.2.0.3 Example: Filter Input Lag - Random Input

Example Table: Filter Input Lag - Random Input

A real world analog input is most always a varying feedback signal.
In this table, an initial input of 100, a gain of 0.8, and a random variability of 10%.Filter Input

Filter Input	Filter Current Output	Amount of Input Filtering	Random Filter % Variation
0	0	0	10%
100	80	-20	
97.38903813	93.9112305	-3.477807626	
92.67638093	92.92335084	0.246969915	
94.12988912	93.88858146	-0.241307655	
103.0835564	101.2445614	-1.838994993	
91.16845433	93.18367575	2.015221422	
93.23936976	93.22823096	-0.011138803	
94.90272089	94.56782291	-0.334897986	
103.3070737	101.5592235	-1.747850153	

Filter Input	Filter Current Output	Amount of Input Filtering	Random Filter % Variation
96.83149418	97.77704005	0.945545867	
96.35024002	96.63560002	0.285360007	
99.82417525	99.1864602	-0.637715045	
105.0792636	103.9007029	-1.178560685	
97.36988208	98.67604626	1.306164172	
107.82502	105.9952253	-1.829794752	
97.7886524	99.42996698	1.641314572	
108.2038024	106.4490353	-1.754767081	
91.58527607	94.55802792	2.972751845	
93.6783421	93.85427926	0.175937164	
102.8695349	101.0664838	-1.803051129	
93.95916817	95.3806313	1.421463121	
108.6579707	106.0025028	-2.655467871	
109.3425748	108.6745604	-0.668014397	
103.9066	104.8601921	0.953592077	
92.30112142	94.81293555	2.511814127	
109.4460726	106.5194452	-2.926627416	
94.88799896	97.21428821	2.326289251	
105.4738635	103.8219484	-1.651915057	
102.988167	103.1549233	0.166756284	
92.92925408	94.97438792	2.045133846	
95.58185568	95.46036213	-0.121493552	
109.414248	106.6234708	-2.790777178	
106.5661311	106.577599	0.011467953	
99.85857253	101.2023778	1.343805301	
107.865421	106.5328124	-1.332608643	
92.19683177	95.0640279	2.867196126	
104.8558146	102.8974573	-1.958357346	
104.5140236	104.1907104	-0.323313268	
104.3675014	104.3321432	-0.035358206	
109.2704266	108.2827699	-0.987656683	
101.4962729	102.8535723	1.35729941	
92.19199163	94.32430776	2.132316128	
99.13065312	98.16938405	-0.961269073	
103.5068114	102.4393259	-1.067485466	
109.502983	108.0902516	-1.412731426	

Filter Input	Filter Current Output	Amount of Input Filtering	Random Filter % Variation
99.05504822	100.8620889	1.80704068	
94.97711299	96.15410817	1.176995182	
107.1063597	104.9159094	-2.190450308	
91.12245188	93.88114339	2.758691504	
108.130314	105.2804799	-2.849834129	
104.2923832	104.4900025	0.197619344	
101.3775072	102.0000062	0.62249907	
100.5303014	100.0399168	-0.490384645	Averages

FBD Language Example

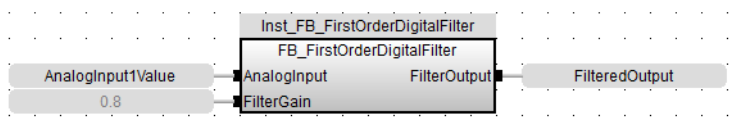
Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	No range	N/A	No default	- Enables execution. - Used in the FFLD editor only.
AnalogInput	INT	No range	N/A	No default	Analog Input from transducer.
FilterGain	REAL	1 to 0.05	N/A	No default	Filter gain.

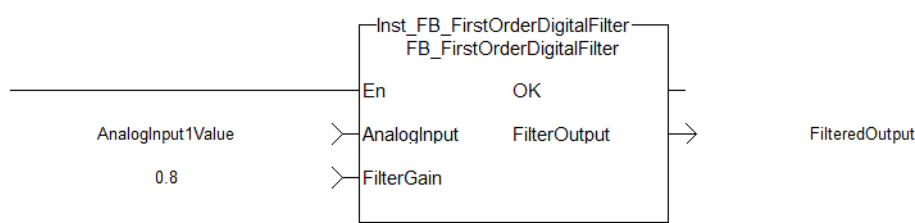
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	0 to 1	N/A	Execution successful.
FilterOutput	REAL	0 to 1	N/A	Filtered analog input value.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Filter analog input signal with a gain of 0.8 to remove noise  
FilteredOutput:= Inst_FB_FirstOrderDigitalFilter( AnalogInput1Value, 0.8 );
```

2.6.3 FB_PWDutyOutput

 **Function Block** - Accepts an input value between the minimum and maximum input range and converts this to a duty cycle percentage.

The output is then cycled on and off over the period of the duty cycle at the duty cycle percentage.

- To have the output ON time range from 0 (zero) to the duty cycle period, the minimum should be set to 0 (zero) and the maximum to the duty cycle period.
- If the calculated duty cycle is:
 - Based on the input and the range values are less than the minimum ON time (MinTime), the output does not come on.
 - Between or equal to the range values the output is cycled by the duty cycle.
 - Greater than the maximum ON time (MaxTime) the output will remain on.

Usage

Flash a warning light for operators.

2.6.3.0.1 Function Block Calculations

```
IF (uInputCommand - uMinInput) < 0 then      //If Command less than MinInput
turn out put off
    DutyOutput := 0;
ELSIF (uInputCommand - uMaxInput) > 0 then    //If Command greater than
MaxInput turn out put on
    DutyOutput := 1;
ELSE
    DutyCycle := (uInputCommand - uMinInput)/(uMaxInput -
uMinInput);      //Calculate Duty Cycle
    ONTimeFromInput := DutyCycle * any_to_REAL(uPeriod) ;      //Calculate
Ontime

    IF any_to_TIME(ONTimeFromInput) < uMinTime then
        DutyOutput := 0;
    ELSIF any_to_TIME(ONTimeFromInput) > uMaxTime then
        DutyOutput := 1;
    ELSE
        TM1 := any_to_TIME(ONTimeFromInput) ;
        TM0 := uPERIOD - TM1;      //Calculate offtime
        DutyOutput := Inst_blinkA( 1 , TM0 , TM1 );      //Use BlinkA function
to set PW output
    END_IF ;
END_IF ;
```

Inputs

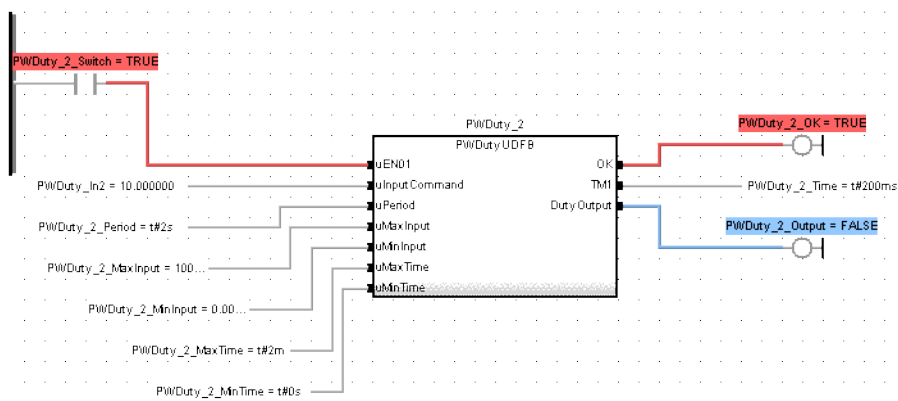
Input	Data Type	Range	Unit	Default	Description
uEN01	BOOL	0 to 1	N/A	No default	Enable for the block.

Input	Data Type	Range	Unit	Default	Description
uInputCommand	REAL	0 to --	N/A	No default	- Signal Input. - Sometimes the output of a PID block.
uPeriod	TIME	0 to --	N/A	No default	Period of the duty cycle.
uMaxInput	REAL	uMinInput to --	N/A	No default	uInputCommand at or above this number that sets DutyOutput = 1.
uMinInput	REAL	0 (zero) to uMaxInput	N/A	No default	uInputCommand at or below this number set DutyOutput = 0 (zero).
uMaxTime	TIME	uMinTime to uPeriod	N/A	No default	Maximum on time for the output.
uMinTime	TIME	0 (zero) to uMaxTime	N/A	No default	Minimum on-time for the PW Duty Output.

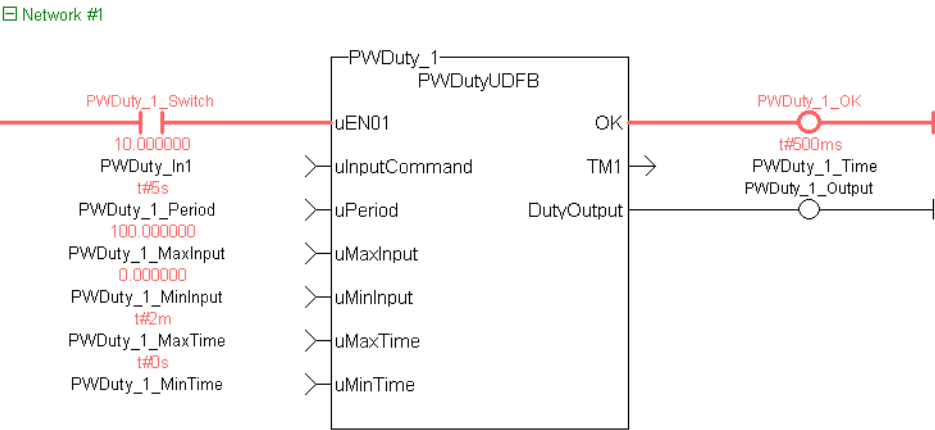
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	0 to 1	N/A	Function block is operational.
TM1	TIME	0 (zero) to uPeriod	N/A	On-time of the Output.
DutyOutput	BOOL	0 to 1	N/A	PW signal (switching between 0 and 1). DutyOutput is set to 0 (zero) when the function block is not active (not eabled by the first input).

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_FB_PWDutyOutput( PWDuty_3_Switch, PWDuty_In3, PWDuty_3_Period, PWDuty_3_MaxInput, PWDuty_3_MinInput, PWDuty_3_MaxTime, PWDuty_3_MinTime);
PWDuty_3_OK:=Inst_FB_PWDutyOutput.OK;
PWDuty_3_Time:=Inst_FB_PWDutyOutput.TM1;
PWDuty_3_Output:=Inst_FB_PWDutyOutput.DutyOutput;
```

See Also

Timers

2.6.4 FB_ScaleInput - Scaling Analog IO



Function Block - Converts un-scaled DINT values from Analog Inputs into user units of type LREAL.

- Scales DINT to LREAL.
- The input signal is converted based on a linear mapping automatically calculated by two points entered.
- InputMin is mapped to OutputMin, InputMax is mapped to OutputMax, and all values in between are scaled automatically.
- If an input value is not between the selected Min/Max, the Boolean output OutsideRange turns TRUE.
 - The OutputSignal is set to the corresponding OutputMin or OutputMax value.

Usage

Scale an analog signal from a drive.

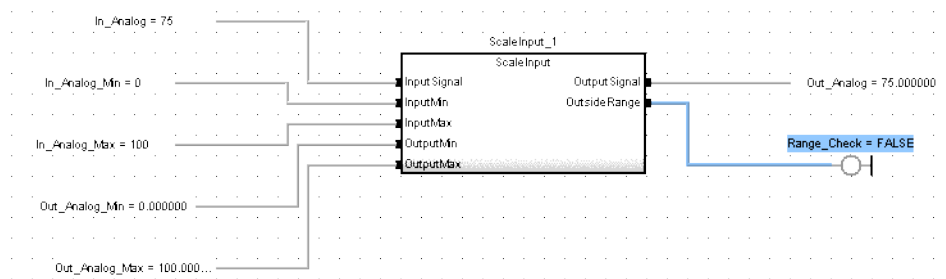
Inputs

Input	Data Type	Range	Unit	Default	Description
InputSignal	DINT	0 to 4	N/A	No default	Un-scaled input signal.
InputMin	DINT	0 to 4	N/A	No default	Minimum value of accepted input signal range.
InputMax	DINT	0 to 4	N/A	No default	Maximum value of accepted input signal range.
OutputMin	LREAL	0 to 4	N/A	No default	Output value mapped to the InputMin.
OutputMax	LREAL	0 to 4	N/A	No default	Output value mapped to the InputMax.

Outputs

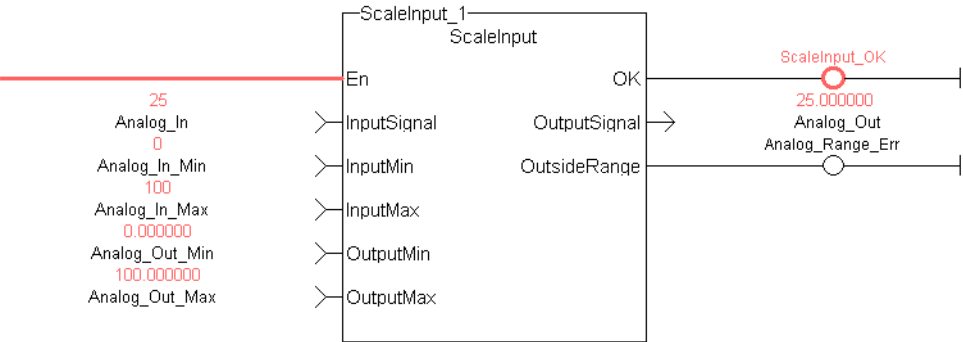
Output	Data Type	Range	Unit	Description
OutputSignal	LREAL		N/A	Scaled value of the Input Signal with type converted to LREAL. Stays within specified Min/Max output values.
OutsideRange	BOOL	FALSE, TRUE	N/A	TRUE if the InputSignal is outside the range setup by min/max values, otherwise FALSE.

FBD Language Example



FFLD Language Example

Network #3



IL Language Example

Not available.

ST Language Example

```
//Scale an integer based analog input signal into floating point LREAL
variable
ScaleInput_1( Analog_In, Analog_In_Min, Analog_In_Max, LREAL_Out_Min, LREAL_
Out_Max );
LREAL_OutputSignal:= ScaleInput_1.OutputSignal;
Analog_Range_Err:= ScaleInput_1.OutsideRange;
```

See Also

"FB_ScaleOutput - Scaling Analog IO" (→ p. 259)

2.6.5 FB_ScaleOutput - Scaling Analog IO



Function Block - Converts un-scaled LREAL values from a PLC program into units of type DINT.

- The scale LREAL to DINT can be mapped to an analog output.
- The input signal is converted based on a linear mapping automatically calculated by two points entered.
- InputMin is mapped to OutputMin, InputMax is mapped to OutputMax, and all values in between are scaled automatically.
- If an input value is not between the selected Min/Max, the Boolean output OutsideRange turns TRUE.
 - The OutputSignal is set to the corresponding OutputMin or OutputMax value.

Usage

Scale an analog signal from a drive.

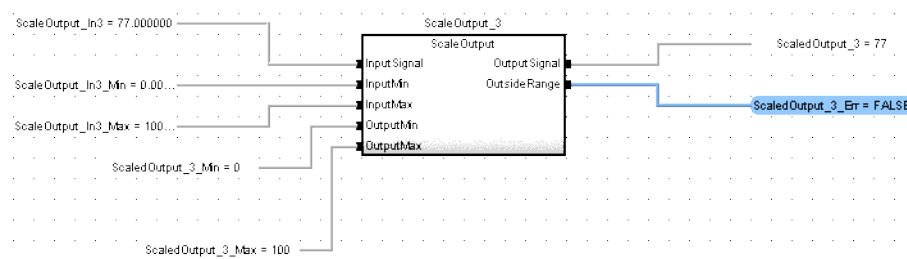
Inputs

Input	Data Type	Range	Unit	Default	Description
InputSignal	LREAL	0 to 4	N/A	No default	Un-scaled input signal.
InputMin	LREAL	0 to 4	N/A	No default	Minimum value of accepted input signal range.
InputMax	LREAL	0 to 4	N/A	No default	Maximum value of accepted input signal range.
OutputMin	DINT	0 to 4	N/A	No default	Output value mapped to the InputMin.
OutputMax	DINT	0 to 4	N/A	No default	Output value mapped to the InputMax.

Outputs

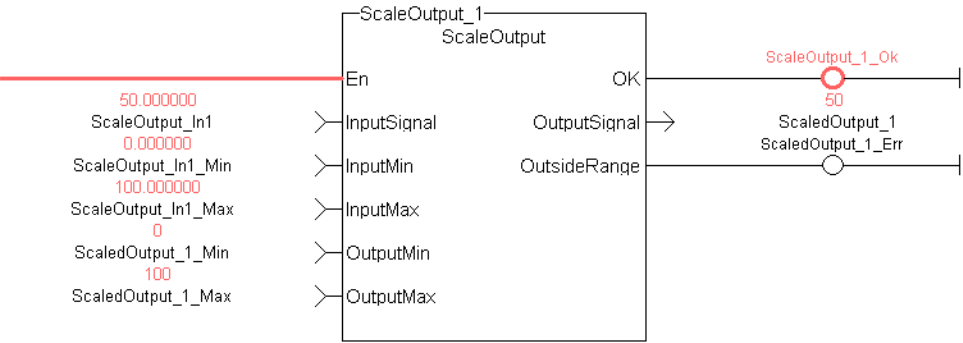
Output	Data Type	Range	Unit	Description
OutputSignal	DINT		N/A	Scaled value of the Input Signal with type converted to DINT. Stays within specified Min/Max output values.
OutsideRange	BOOL	FALSE, TRUE	N/A	TRUE if the InputSignal is outside the range setup by min/max values, otherwise FALSE.

FBD Language Example



FFLD Language Example

Network #4



IL Language Example

Not available.

ST Language Example

```
Inst_ScaleOutput1( ScaleOutput_In2, ScaleOutput_In2_Min,ScaleOutput_In2_Max,  
ScaledOutput_2_Min, ScaledOutput_2_Max );  
ScaledOutput_2:=Inst_ScaleOutput1.OutputSignal;  
ScaledOutput_2_Err:=Inst_ScaleOutput1.OutsideRange;
```

See Also

"FB_ScaleInput - Scaling Analog IO" (→ p. 257)

2.6.6 FB_TemperaturePID



Function Block - Provides PID temperature control with auto tuning.

Usage

2.6.6.0.1 Tuning Process

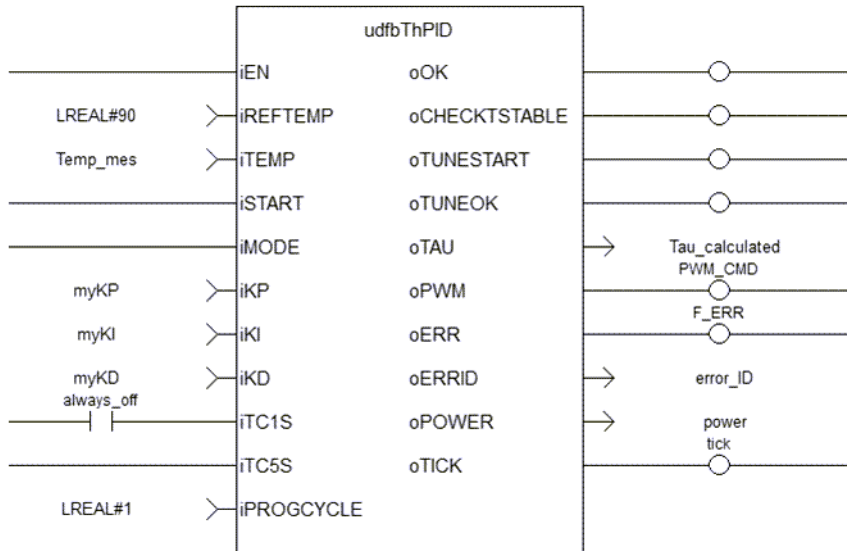


Figure 3-29: FB_TemperaturePID Usage 1

2.6.6.0.2 Tuning Procedure

[Click for the procedure](#)

1. Verify the ambient temperature is stable.
The measured $\Delta \text{temp} = T_{\text{max}} - T_{\text{min}}$ must be lower than $0.1 * T_{\text{max}}$.
This step takes 10 cycles ($10 * iTC5s$ or $10 * iTC1s$).
The tuning fails ($oERR = \text{TRUE}$, $oERRID = 1$) if the ambient temperature is greater than $0.1 * T_{\text{max}}$, otherwise $T_{\text{amb}} = (T_{\text{max}} + T_{\text{min}}) / 2$.
2. Start tuning Phase1: output **oPWM** is kept TRUE until the final measured temperature **iTEMP** gets over $iREFTEMP / 2$.
After that, **oPWM** is kept LOW.
3. Start tuning Phase2.
With **oPWM** kept LOW, the temperature gets down until the final value is lower than $[(iREFTEMP / 2 - T_{\text{amb}}) * 0.368 + T_{\text{amb}}]$.

After, PID gains are calculated as:

```
Kp=10
Ki=0.14
delta_time = time to complete Phase2

Kd=SQRT(delta_time)*7
```

The tuning is completed.



TIP
oTAU may be useful for setting the proper sampling time (1s or 5s).

2.6.6.0.3 Start PID Controller

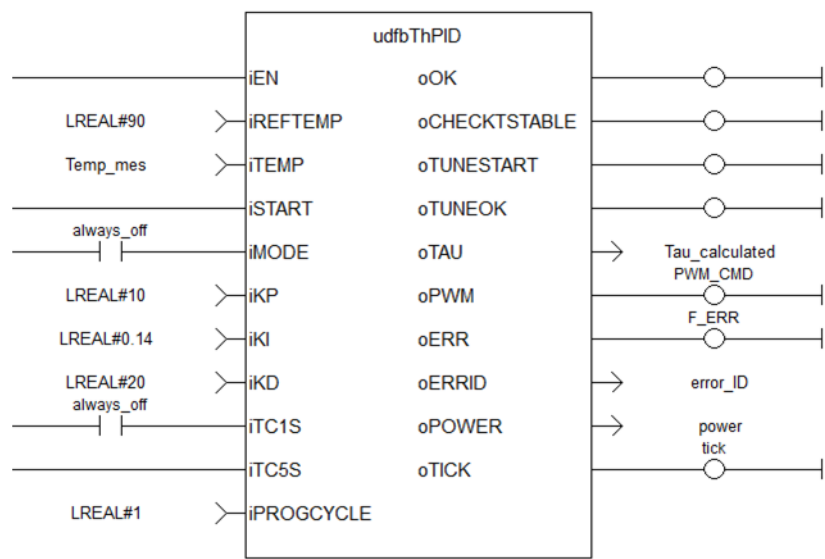


Figure 3-30: FB_TemperaturePID Usage 2

[Click for procedure](#)

- 1. When starting the PID controller:
 - The output **oPWM** is modulated five times within the sampling time (blue line is **oTICK**, green line is **oPWM**).
 - Each pulse length depends on output **oPOWER** (100%=full length).

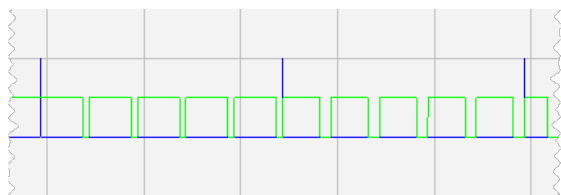


Figure 3-31: FB_TemperaturePID Usage 3

Inputs

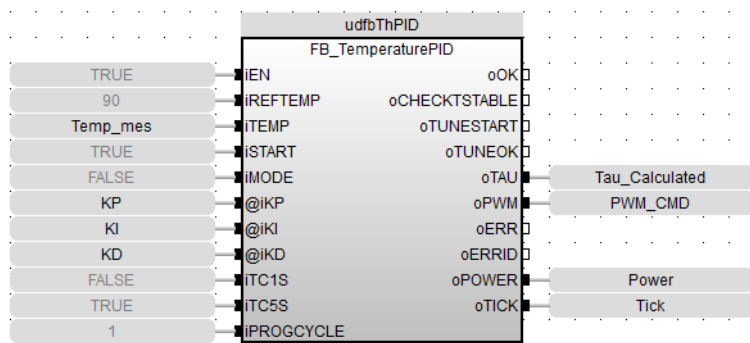
Input	Data Type	Range	Unit	Default	Description
iEN	BOOL				Enable the function.
iREFTEMP	LREAL				Reference temperature [°C].
ITEMP	LREAL				Actual temperature [°C].
iSTART	BOOL				Start PID or auto tuning.
IMODE	BOOL				• FALSE-automatic. • TRUE-tuning.
iKP	LREAL				PID Proportional Gain.
iKI	LREAL				PID Integral Gain.
iKD	LREAL				PID Derivative Gain.
ITC1S	BOOL				Sampling Time is 1s.

Input	Data Type	Range	Unit	Default	Description
iTC5S	BOOL				Sampling Time is 5s.
iPROGCYCLE	LREAL				Execution time of the function [ms].

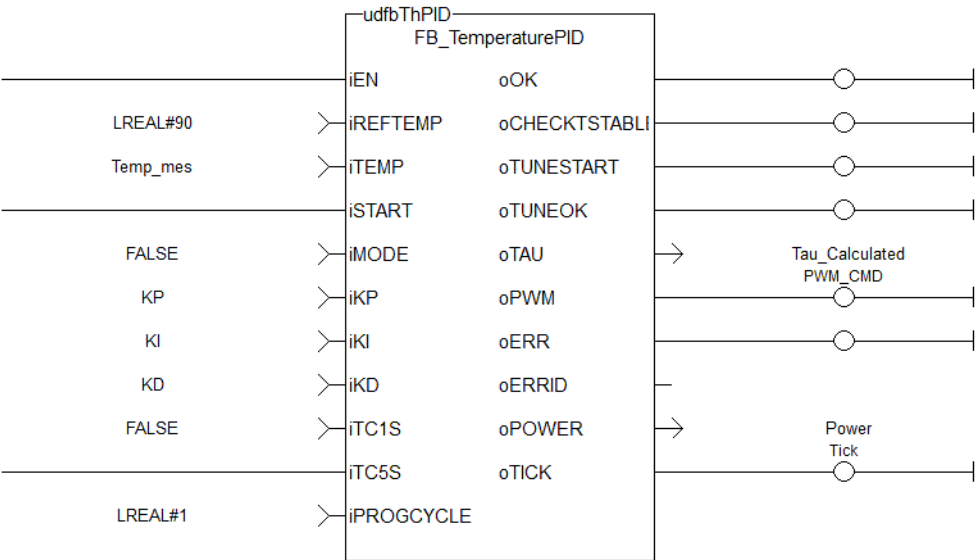
Outputs

Output	Data Type	Range	Unit	Description
oOK	BOOL			Function enabled.
oCHECKSTABLE	BOOL			TRUE when checking if ambient temperature is stable.
oTUNESTART	BOOL			Tuning is started.
oTUNEOK	BOOL			Tuning is completed.
oTAU	LREAL			System Time Constants.
oPWM	BOOL			PWM command for heater.
oERR	BOOL			Function error.
oERRID	INT			Function ID error (in case of oERR=TRUE).
oPOWER	LREAL			% of power requested from heater (100%=full power).
oTICK	BOOL			Pulse every sampling time.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Run PID function with determined proportional, integral, and derivative gains
//send PWM output to command heater
udfbThPID( TRUE, 90, Temp_mes, TRUE, FALSE, KP, KI, KD, FALSE, TRUE, 1);
Tau_Calculated := udfbThPID.oTAU;
PWM_CMD := udfbThPID.oPWM;
Power := udfbThPID.oPOWER;
Tick := udfbThPID.oTICK;
```

2.6.7 PipeNetwork_FFLD - Special FFLD Function

Pipe Network



Function - Calls the PNCode function block in FFLD POU's.

- It starts and initializes the Pipe Network, based on the command specified by `cmdID`.
 - Internally this function calls the Function Block PNCode.
- This is a special function.
 - It should only be used in Pipe Network applications that contain FFLD POU's that call PNCode.
 - To use this function, Pipe Network must be declared as a global variable in the dictionary.
- Calling this function instead of PNCode in FFLD POU's eliminates this compile error that occurs after modifying the Pipe Network using the Pipe Network editor:

```
Controller:PLC:Main: NW1(1,14): PNCode: Invalid block height
```

- The compile error is generated because the number of outputs on PNCode can vary.
 - This occurs after modifying the original Pipe Network using the Pipe Network editor.
- The new PNCode Function Block is not automatically updated in any FFLD POU's, reflecting the new outputs.
 - Manually update each PNCode function block call in any FFLD POU's to correct this problem.

TIP

- The Pipe NetworkFFLD Application Template is a good example of how to use this Function.
- See Pipe Network 2-Axes Template - FFLD.

Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	0 to 1	N/A	No default	Request to initialize the Pipe Network.
cmdID	DINT	N/A	N/A	No default	Commands used to start and initialize the Pipe Network. • MLPN_CREATE_OBJECTS - Create Pipe Network • MLPN_POWER_ON - Power on all axes • MLPN_POWER_OFF - Power off all axes • MLPN_ACTIVATE - Activate the pipes • MLPN_CONNECT - Connect the axes to the pipes • MLPN_DEACTIVATE - Deactivate the pipes

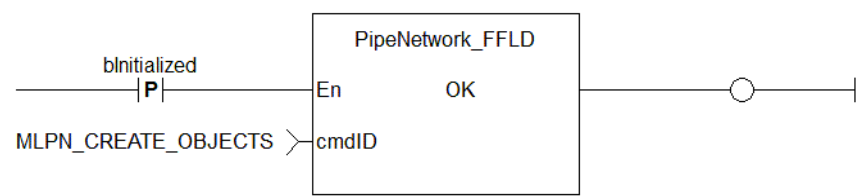
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

Not available.

See Also

- Design Motion with Pipe Network
- Initialize and Start a Pipe Network
- PLCopen 2-Axes Template - FFLD

2.6.8 ProfilesCode_FFLD - Special FFLD Function

Pipe Network



Function - Calls the Profiles Code Function Block in FFLD POU's.

- Internally, this function calls the Function Block ProfilesCode.
- This is a special function.
 - It should only be used in applications that contain FFLD POU's that call ProfilesCode.
 - To use this function, Profiles must be declared as a global variable in the dictionary.
- Calling this function instead of ProfilesCode in FFLD POU's eliminates this compile error that occurs after adding a new Profile to the project tree.

```
Controller:PLC:Main: NW1(1,14): ProfilesCode: Invalid block height
```

- The compile error is generated because the number of outputs on ProfilesCode can vary.
 - This occurs after adding a new profile to the project tree.
- The ProfilesCode Function Block is not automatically updated in any FFLD POU's, reflecting the new outputs.
 - You must manually update each ProfilesCode Function Block call in any FFLD POU's to correct this problem.
 - If you use this function instead, you no longer need to manually update each ProfilesCode Function Block in FFLD.

TIP

- The Pipe Network and PLCopen 2 Axis FFLD Application Templates are two examples of how to use this function.
- See Pipe Network 2-Axes Template - FFLD or PLCopen 2-Axes Template - FFLD.

Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	0 to 1	N/A	No default	Request to initialize the Pipe Network.
cmdID	DINT	N/A	N/A	No default	- Commands used to start and initialize the Pipe Network. MLPR_CREATE_PROFILES - Creation and initialization of profiles.

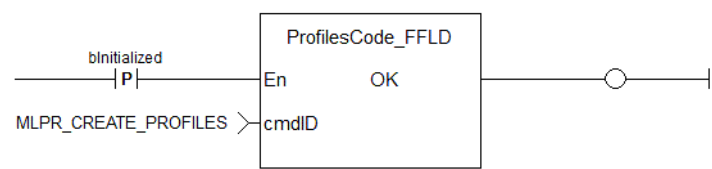
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

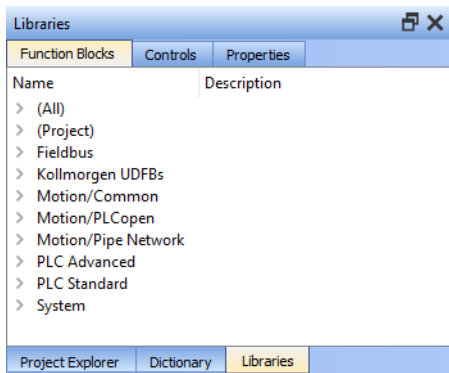
Not available.

ST Language Example

Not available.

3 Motion Library

The Motion Library (ML) (for Pipe Network Concept and PLCopen) is in the Function Blocks tab of the Library toolbox.



- The KAS function library contains ML function blocks used to integrate motion in a PLC program.
- ML function blocks can be used in four of the IEC 61131-3 languages: ST, FBD, FFLD and IL.
- For Sequential Function Chart (SFC) programs, ML function blocks are used as part of SFC Steps or SFC Transitions defined with FBD, FFLD, IL, or ST languages.

3.1 Motion / Pipe Network

The KAS-IDE function library contains Motion Library (ML) function blocks (FBs) used to integrate motion from a Pipe Network in a PLC program.

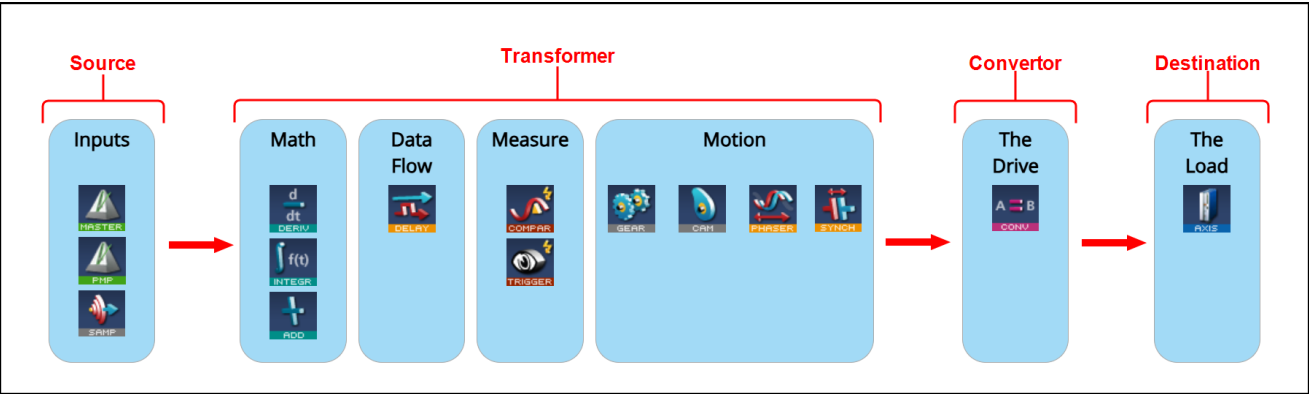
These are the ML function blocks:

Function	Description
Block	Manage the blocks: Create / activate.
Motion	Prepare the physical motion part: init, reset, start, stop.
Pipe Block	Manage each specific Pipe Block: Read / write parameters.
Pipe Network	Manage the Pipe Network: Create / activate.

ⓘ IMPORTANT

Pipe Network code is generated automatically by the compiler.
Do not try to modify it.

3.1.1 Example: Pipe Block Description



3.1.2 Adder

Name	Description
MAddInit	Initializes an Adder Pipe Block for use in a PLC Program with user-defined settings.
MAddReadOff1	Returns the offset value of the first entry of an Adder block.
MAddReadOff2	Returns the offset value of the second entry of an Adder block.
MAddReadRatio1	Returns the ratio value of the first entry of an Adder block.
MAddReadRatio2	Returns the ratio value of the second entry of an Adder block.
MAddWriteInput	Sets the source of an input of an Adder Pipe Block.
MAddWriteOff1	Set the offset value of the first entry of the Adder block.
MAddWriteOff2	Set the offset value of the second entry of the Adder block.
MAddWriteRat1	Set the ratio value of the first entry of the Adder block.
MAddWriteRat2	Set the ratio value of the second entry of the Adder block.

3.1.2.1 Rules

3.1.2.1.1 Adder - Rule 1

The pipe blocks connected to the Adder inputs (e.g., a Cam and a Gear) must have the same output modulo positions.

3.1.2.1.2 Adder - Rule 2

The modulo position of the pipe blocks connected to the Adder inputs must have the same value (or a multiple) as the modulo position of the pipe block connected to the output of the Adder.

3.1.2.2 MLAddInit



Function - Initializes an Adder Pipe Block for use in a PLC Program with user-defined settings.

NOTE

- Adder objects are normally created in the Pipe Network using the graphical engine.
- You do not have to add **MLAddInit** functions to their programs.
- Parameters are entered directly in pop-up windows; the code is automatically added to the current project.

- Function block is automatically called if an Adder Block is added to the Pipe Network.
 - User-defined settings are entered in the **Pipe Blocks Properties** screen.
- The Pipe Block is assigned ratios and offsets for both inputs.
 - After an Adder block is initialized, the inputs need to be selected using the "MLAddWriteInput" (→ p. 281) function block or graphically using the Pipe Network.

Inputs

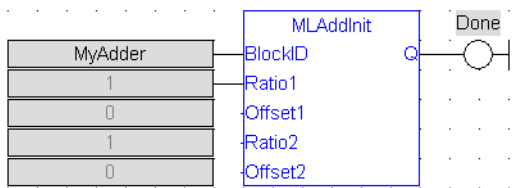
Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of a created Pipe Block.
Ratio1	LREAL	No range	N/A	No default	Sets the Ratio value of the first entry of an Adder object.
Offset1	LREAL	No range	N/A	No default	Sets the Offset value of the first entry of an Adder object.
Ratio2	LREAL	No range	N/A	No default	Sets the Ratio value of the second entry of an Adder object.
Offset2	LREAL	No range	N/A	No default	Sets the Offset value of the second entry of an Adder object.

Outputs

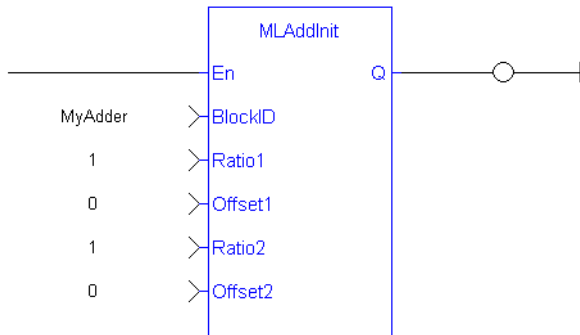
Adder Block Output = Ratio1*Input1 + Offset1 + Ratio2*Input2 + Offset2

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Adder Pipe Block is initialized. •

3.1.2.2.1 FBD Language Example



3.1.2.2.2 FFLD Language Example



IL Language Example

Not available.

3.1.2.2.3 ST Language Example

```
//Create and Initiate a Trigger object
MyAdder := MLBlkCreate( 'MyAdder', 'ADDER' );
MLAddInit( MyAdder, 1.0, 0.0, 1.0, 0.0 );
```

See Also

- "MLAddReadOff1" (→ p. 274)
- "MLAddReadRatio1" (→ p. 277)
- "MLAddWriteInput" (→ p. 281)
- "MLBlkCreate" (→ p. 292)

3.1.2.3 MLAddReadOff1



Function - Returns the offset value of the first entry of an Adder block.

- Offset1 shifts the value of the first input to the block before its added to the second input.
- Can change the offset value with "MLAddWriteOff1" (→ p. 283) function.

Inputs

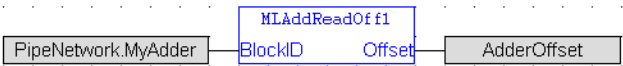
Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Adder object.

Outputs

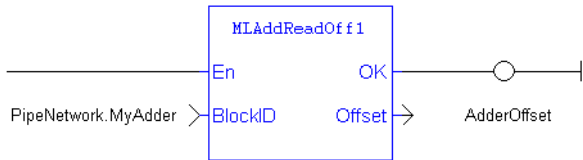
Adder Block Output = Ratio1*Input1 + Offset1 + Ratio2*Input2 + Offset2

Output	Data Type	Range	Unit	Description
Offset	LREAL	No range	User units	Returns the offset value of the first entry of an Adder block.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

3.1.2.3.1 ST Language Example

```
//Save the offset value of first entry to the Adder block
AdderOffset := MLAddReadOff1( PipeNetwork.MyAdder );
```

See Also

- "MLAddReadOff2" (→ p. 275)
- "MLAddReadRatio1" (→ p. 277)
- "MLAddWriteOff1" (→ p. 283)
- "MLAddWriteRat1" (→ p. 287)

3.1.2.4 MLAddReadOff2

Pipe Network ✓



Function - Returns the offset value of the second entry of an Adder block.

- Offset2 shifts the value of the second input to the block before its added to the first input.
- Can change the offset value with "MLAddWriteOff2" (→ p. 285) function.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Adder object.

Outputs

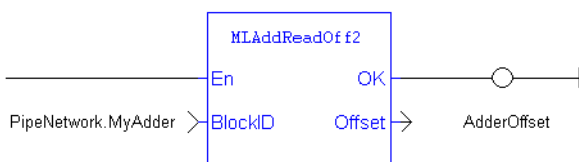
Adder Block Output = Ratio1*Input1 + Offset1 + Ratio2*Input2 + Offset2

Output	Data Type	Range	Unit	Description
Offset	LREAL	No range	User units	Returns the offset value of the second entry of an Adder block.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Save the offset value of second entry to the Adder block
AdderOffset := MLAddReadOff2( PipeNetwork.MyAdder );
```

See Also

- "MLAddReadOff1" (→ p. 274)
- "MLAddReadRatio2" (→ p. 279)

- "MLAddWriteOff2" (→ p. 285)
- "MLAddWriteRat2" (→ p. 289)

3.1.2.5 MAddReadRatio1

Pipe Network ✓



Function - Returns the ratio value of the first entry of an Adder block.

- Ratio1 amplifies the value of the first input to the block before its added to the second input.
- Can change the ratio value with "MLAddWriteRat1" (→ p. 287) function.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Adder object.

Outputs

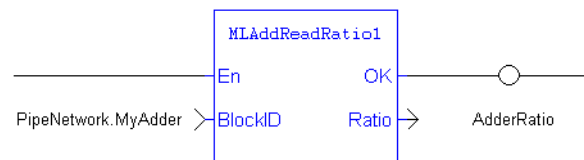
Adder Block Output = Ratio1*Input1 + Offset1 + Ratio2*Input2 + Offset2

Output	Data Type	Range	Unit	Description
Ratio	LREAL	No range	No units	Returns the ratio value of the first entry of an Adder block.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Save the ratio value of first entry to the Adder block
AdderRatio := MAddReadRatio1( PipeNetwork.MyAdder );
```

See Also

- "MLAddReadOff1" (→ p. 274)
- "MLAddReadOff2" (→ p. 275)

- "MLAddReadRatio2" (→ p. 279)
- "MLAddWriteRat1" (→ p. 287)

3.1.2.6 MAddReadRatio2

Pipe Network ✓



Function - Returns the ratio value of the second entry of an Adder block.

- Ratio2 amplifies the value of the second input to the block before its added to the first input.
- Can change the ratio value with "MLAddWriteRat2" (→ p. 289) function.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Adder object.

Outputs

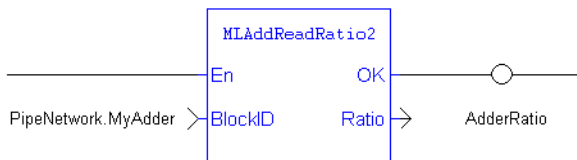
Adder Block Output = Ratio1*Input1 + Offset1 + Ratio2*Input2 + Offset2

Output	Data Type	Range	Unit	Description
Ratio	LREAL	No range	N/A	Returns the ratio value of the second entry of an Adder block.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Save the ratio value of second entry to the Adder block
AdderRatio := MAddReadRatio2( PipeNetwork.MyAdder );
```

See Also

- "MLAddReadOff1" (→ p. 274)
- "MLAddReadOff2" (→ p. 275)

- "MLAddReadRatio1" (→ p. 277)
- "MLAddWriteRat2" (→ p. 289)

3.1.2.7 MLAddWriteInput

Pipe Network



Function - Sets the source of an input of an Adder Pipe Block.

- Function block is automatically called if an Adder Block is connected to other blocks in the Pipe Network.

NOTE

- Adder objects are normally created in the Pipe Network using the graphical engine.
- You do not have to add **MLAddWriteInput** functions to their programs.
 - They are connected with lines in the Pipe Network.
- The code is automatically added to the current project.

Inputs

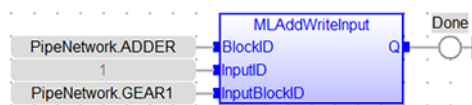
Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Adder object.
InputID	DINT	1 to 2	N/A	No default	Select first or second input to the Adder object.
InputBlockID	DINT	1 to 1024	N/A	No default	- ID number of an initiated Pipe Block. - This is an input to the Adder object.

Outputs

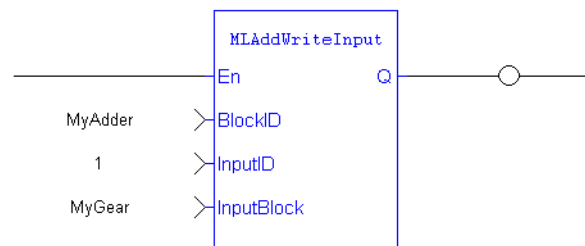
Adder Block Output = Ratio1*Input1 + Offset1 + Ratio2*Input2 + Offset2

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE if the input to the Adder object is set.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Set the first input of an Adder pipeblock to be connected to the output of  
GEAR1 pipeblock  
MLAddWriteInput( PipeNetwork.ADDER, 1, PipeNetwork.GEAR1 );
```

See Also

- "MLAddInit" (→ p. 272)
- "MLAddReadOff1" (→ p. 274)
- "MLAddReadRatio1" (→ p. 277)
- "MLBlkCreate" (→ p. 292)

3.1.2.8 MLAddWriteOff1

Pipe Network 



Function - Set the offset value of the first entry of the Adder block.

ⓘ IMPORTANT

Changes made to the Offset of an Adder block are executed immediately and can cause an axis position to jump.

- Offset1 shifts the value of the first input to the block before its added to the second input.

Inputs

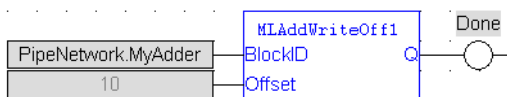
Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Adder object.
Offset	LREAL	No range	N/A	No default	Desired new value for the Adder Object's Offset1.

Outputs

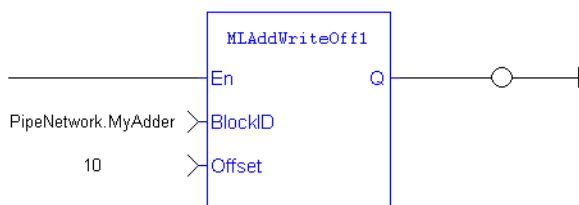
Adder Block Output = Ratio1*Input1 + Offset1 + Ratio2*Input2 + Offset2

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Offset value for input one is set. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Change the offset value of first entry to the Adder block to 10  
MLAddWriteOff1( PipeNetwork.MyAdder, 10 );
```

See Also

- "MLAddReadOff1" (→ p. 274)
- "MLAddReadRatio1" (→ p. 277)
- "MLAddWriteOff2" (→ p. 285)
- "MLAddWriteRat1" (→ p. 287)

3.1.2.9 MAddWriteOff2

Pipe Network 



Function - Set the offset value of the second entry of the Adder block.

ⓘ IMPORTANT

Changes made to the Offset of an Adder block are executed immediately and can cause an axis position to jump.

- Offset2 shifts the value of the second input to the block before its added to the first input.

Inputs

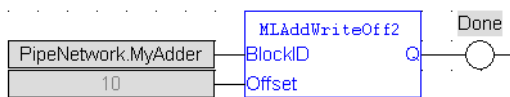
Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Adder object.
Offset	LREAL	No range	N/A	No default	Desired new value for the Adder Object's Offset2.

Outputs

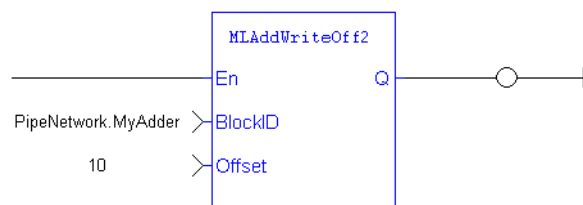
Adder Block Output = Ratio1*Input1 + Offset1 + Ratio2*Input2 + Offset2

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Offset value for input two is set. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Change the offset value of second entry to the Adder block to 10  
MLAddWriteOff2( PipeNetwork.MyAdder, 10 );
```

See Also

- "MLAddReadOff2" (→ p. 275)
- "MLAddReadRatio2" (→ p. 279)
- "MLAddWriteOff1" (→ p. 283)
- "MLAddWriteRat2" (→ p. 289)

3.1.2.10 MAddWriteRat1

Pipe Network ✓



Function - Set the ratio value of the first entry of the Adder block.

ⓘ IMPORTANT

Changes made to the Ratio of an Adder block are executed immediately and can cause an axis position to jump.

- Ratio1 amplifies the value of the first input to the block before its added to the second input.

Inputs

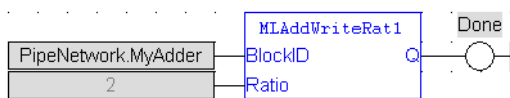
Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Adder object.
Ratio	LREAL	No range	N/A	No default	Designated new value for the Adder Object's Ratio1.

Outputs

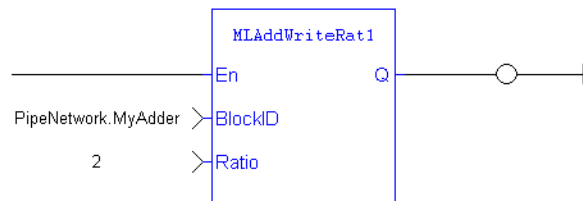
Adder Block Output = Ratio1*Input1 + Offset1 + Ratio2*Input2 + Offset2

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Returns TRUE if the Ratio value for input one is set.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Change the ratio value of first entry to the Adder block to 2  
MLAddWriteRat1( PipeNetwork.MyAdder, 2 );
```

See Also

- "MLAddReadOff1" (→ p. 274)
- "MLAddReadRatio1" (→ p. 277)
- "MLAddWriteOff1" (→ p. 283)
- "MLAddWriteRat2" (→ p. 289)

3.1.2.11 MLAddWriteRat2

Pipe Network 



Function - Set the ratio value of the second entry of the Adder block.

ⓘ IMPORTANT

Changes made to the Ratio of an Adder block are executed immediately and can cause an axis position to jump.

- Ratio2 amplifies the value of the second input to the block before its added to the first input.

Inputs

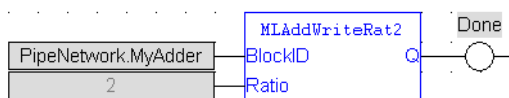
Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Adder object.
Ratio	LREAL	No range	N/A	No default	Designated new value for the Adder Object's Ratio2.

Outputs

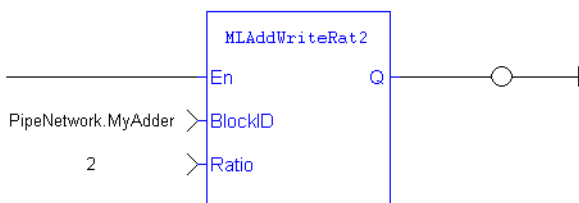
Adder Block Output = Ratio1*Input1 + Offset1 + Ratio2*Input2 + Offset2

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE if the Ratio value for input two is set.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example


```
//Change the ratio value of second entry to the Adder block to 2  
MLAddWriteRat2 ( PipeNetwork.MyAdder, 2 );
```

See Also

- "MLAddReadOff2" (→ p. 275)
- "MLAddReadRatio2" (→ p. 279)
- "MLAddWriteOff2" (→ p. 285)
- "MLAddWriteRat1" (→ p. 287)

3.1.3 Block

Name	Description
MLBikCreate	Creates a new Pipe Block object.
MLBikIsReady	Verifies if a Pipe Block currently has a function running.
MLBikReadModPos	Gets the value of the period of a block in user units.
MLBikReadOutVal	Gets the output value of a selected Pipe Block.
MLBikWriteModPos	Sets the value of the period of a block in user units.

3.1.3.1 MLBlkCreate



Function - Creates a new Pipe Block object.

- Before a Pipe Block is Initialized, the function needs to be created and assigned an ID number.
- The **MLBlkCreate** function is automatically called if a Block is added to the Pipe Network.

NOTE

- Pipe Blocks are normally created in the Pipe Network using the graphical engine.
- You do not have to add **MLBlkCreate** function blocks to their programs.
- The code with **MLBlkCreate** commands are automatically generated and called in a program with **Pipe Network(MLPN_CREATE_OBJECTS)**.

TIP

This function should be called after "MLMotionInit" (→ p. 867) is called and before "MLMotionStart" (→ p. 872) is called.

Inputs

Input	Data Type	Range	Unit	Default	Description
Name	STRING	No range	N/A	No default	Designated name for the newly created Pipe Block.
Type	STRING	No range	N/A	No default	• The type of Pipe Block to create. • Examples: MASTER, GEAR, PHASER, etc.

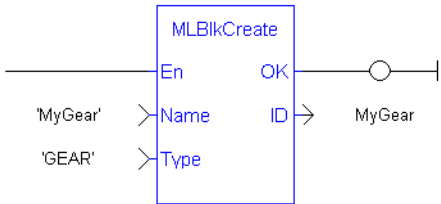
Outputs

Output	Data Type	Range	Unit	Description
ID	DINT	No range	N/A	Assigned ID number of the created Block.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Create a new GEAR Pipe Block named "MyGear"  
MyGear := MLBlkCreate( 'MyGear', 'GEAR' );
```

See Also

- "MLAxisInit" (→ p. 345)
- See Step 3 in Initialize and Start a Pipe Network.

3.1.3.2 MLBlkIsReady



Function - Verifies if a Pipe Block currently has a function running.

- Returns FALSE if the selected Pipe Block has a function running.
- Same return value as the .Q output of a specific function itself.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

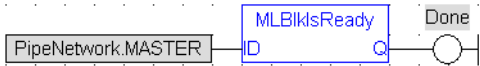
Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID number of a created Pipe Block.

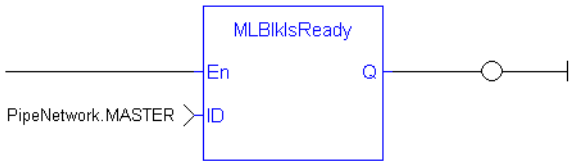
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if no function of a specified Pipe Block is running. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Check if the MST Pipe Block named "MASTER" has a function running
IsReady := MLBlkIsReady( PipeNetwork.MASTER );
```

See Also

- "MLBlkReadModPos" (→ p. 296)
- "MLBlkReadOutVal" (→ p. 298)

3.1.3.3 MLBlkReadModPos



Function - Gets the value of the period of a block in user units.

- The output value of a block is reset each time it reaches its period value.

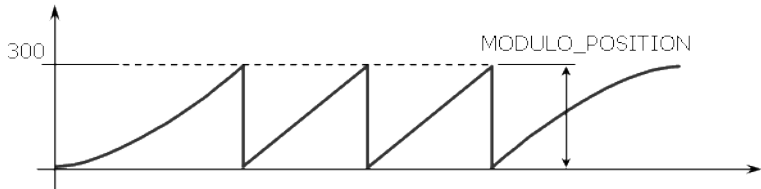


Figure 4-1: Modulo position of the axis.

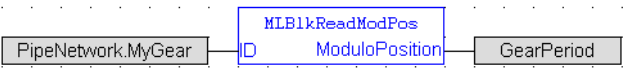
Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID number of a created Pipe Block.

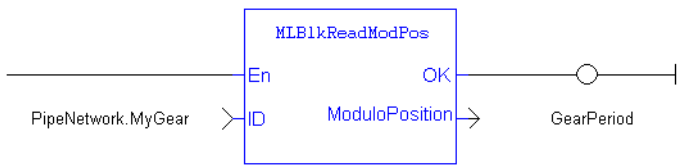
Outputs

Output	Data Type	Range	Unit	Description
ModuloPosition	LREAL	≥ 0 (zero)	User units	Current period value for selected Pipe Block.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Return and save the Period of a Pipe Block
GearPeriod := MLBlkReadModPos( PipeNetwork.MyGear );
```

See Also

- "MLBlkCreate" (→ p. 292)
- "MLBlkReadOutVal" (→ p. 298)
- "MLBlkWriteModPos" (→ p. 299)

3.1.3.4 MLBlkReadOutVal



Function - Gets the output value of a selected Pipe Block.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

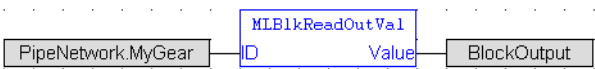
Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID number of a created Pipe Block.

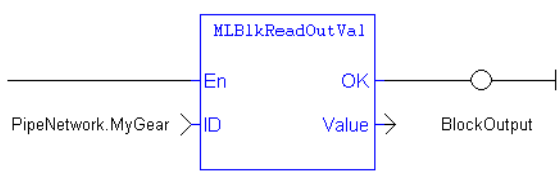
Outputs

Output	Data Type	Range	Unit	Description
Value	LREAL	No range	User units	Current output value of the selected Pipe Block.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Save the output of a Gear Pipe Block
BlockOutput := MLBlkReadOutVal( PipeNetwork.MyGear );
```

See Also

- "MLBlkCreate" (→ p. 292)
- "MLBlkReadModPos" (→ p. 296)

3.1.3.5 MLBlkWriteModPos



Function - Sets the value of the period of a block in user units.

- The output value of a block is reset each time it reaches its period value.

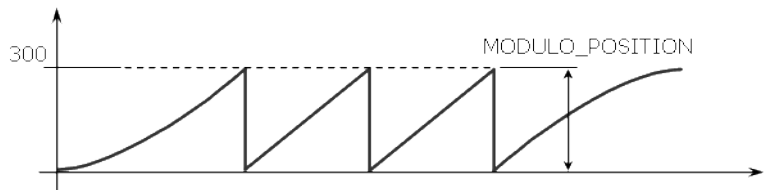


Figure 4-2: Modulo position of the axis.

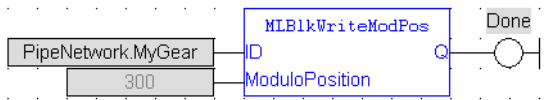
Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID number of a created Pipe Block.
ModuloPosition	LREAL	No range	User units	No default	Designated new Period Value for selected Pipe Block.

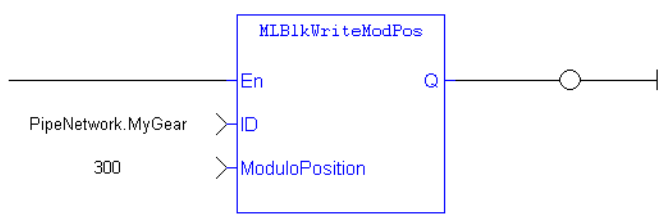
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the function block is executed.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Set the Period of a Pipe Block to 300  
MLBlkWriteModPos( PipeNetwork.MyGear, 300 );
```

See Also

- "MLBlkCreate" (→ p. 292)
- "MLBlkReadModPos" (→ p. 296)
- "MLBlkReadOutVal" (→ p. 298)

3.1.4 Pipe

Name	Description
MLPipeAct	Activates a pipe.
MLPipeAddBlock	Adds a Pipe Block to a pipe.
MLPipeCreate	Create a new pipe object.
MLPipeDeact	Deactivates a pipe.

3.1.4.1 MLPipeAct

Pipe Network ✓

 **Function** - Activates a pipe.

A Pipe contains an Input Pipe Block (Master, PMP, or Sampler), a Converter Output Pipe Block, and any Transformation Pipe Block that can be in between.

- The "MLPipeAct" (→ p. 302) image shows two Pipes, both with the same Master Input Pipe Block.
 - The first ends with the first converter and has a Gear Pipe Block to transform the input values from the Master.
 - The second ends with the second converter and has a CAM Pipe Block to modify the input values from the Master.
- Once a Pipe is activated, history on the values in the Pipe's Blocks are saved and updated each program cycle.
 - A Converter object connected to a destination Axis object cannot send updated position values unless its Pipe is activated.

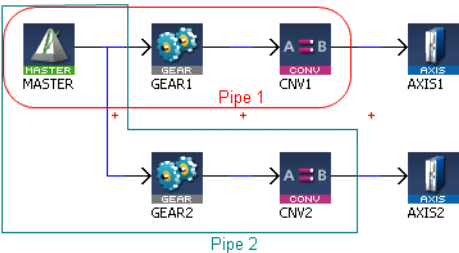


Figure 4-3: MLPipeAct

NOTE

- All Pipes in the Pipe Network can be activated at once with the command PipeNetwork(MLPN_ACTIVATE).
- This calls automatically generated code with MLPipeAct commands for each Pipe object.
- In a multi-pipe program, only one command can be used to activate Pipes instead of writing code for each Pipe separately.

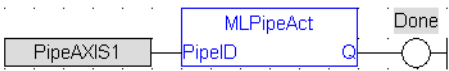
Inputs

Input	Data Type	Range	Unit	Default	Description
PipeID	DINT	1 to 1024	N/A	No default	ID number of a created Pipe object.

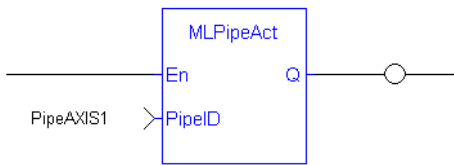
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Pipe is activated. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Activate a Pipe
MLPipeAct( PipeAXIS1 );
```

See Also

- "MLCNVConnect" (→ p. 445)
- "MLPipeAddBlock" (→ p. 304)
- "MLPipeDeact" (→ p. 308)
- Pipe Network Functions for the PLC - MLPN_ACTIVATE

3.1.4.2 MLPipeAddBlock



Function - Adds a Pipe Block to a pipe.

A Pipe contains an Input Pipe Block (Master, PMP, or Sampler), a Converter Output Pipe Block, and any Transformation Pipe Block that can be in between.

- The "MLPipeAddBlock" (→ p. 304) image shows two Pipes, both with the same Master Input Pipe Block.
- If Pipe 1 is created without using the Graphical Engine, these commands are used once a Pipe and the Pipe Blocks have been created:
 - `MLPipeAddBlock( PipeAXIS1, MASTER );`
 - `MLPipeAddBlock( PipeAXIS1, MyGear );`
 - `MLPipeAddBlock( PipeAXIS1, CNV1 );`

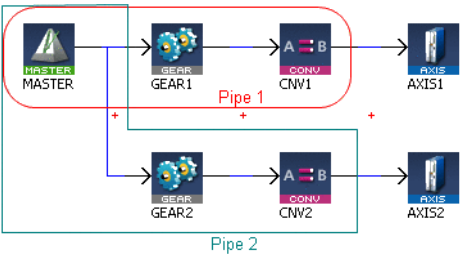


Figure 4-4: MLPipeAddBlock

NOTE

- All Blocks in the Pipe Network are added to a Pipe automatically.
- Code with **MLPipeAddBlock** commands are automatically generated and called in a program with `PipeNetwork(MLPN_CREATE_OBJECTS)`.
- When using the Pipe Network graphical engine to create Pipe Blocks, the user does not have to manually add MLPipeAddBlock commands to the Project.

Inputs

Input	Data Type	Range	Unit	Default	Description
PipeID	DINT	1 to 1024	N/A	No default	ID number of a created Pipe object.
BlockID	DINT	1 to 1024	N/A	No default	ID number of a created Pipe object to add to the selected Pipe.

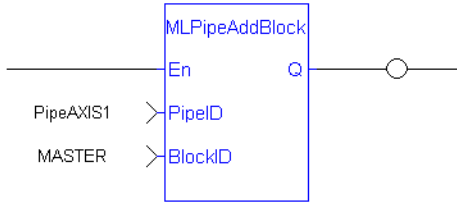
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Pipe Block is added to the Pipe. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Add a block to a pipe
MLPipeAddBlock( PipeAXIS1, MyGear );
```

See Also

- "MLCNVConnect" (→ p. 445)
- "MLPipeAct" (→ p. 302)
- "MLPipeCreate" (→ p. 306)
- "MLPipeDeact" (→ p. 308)
- Pipe Network Functions for the PLC - MLPN_CREATE_OBJECTS

3.1.4.3 MLPipeCreate



 **Function** - Create a new pipe object.

A Pipe contains an Input Pipe Block (Master, PMP, or Sampler), a Converter Output Pipe Block, and any Transformation Pipe Block that can be in between.

NOTE

- Pipes are created in the Pipe Network using the graphical engine.
- You do not have to add MLPipeCreate function blocks to their programs.
- The code with **MLPipeCreate** commands are automatically generated and called in a program with PipeNetwork(MLPN_CREATE_OBJECTS).

Inputs

Input	Data Type	Range	Unit	Default	Description
Name	STRING	No range	N/A	No default	Name for the newly created Pipe.

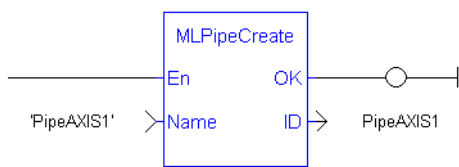
Outputs

Output	Data Type	Range	Unit	Description
ID	DINT			Assigned ID number of the created Pipe.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Create a new pipe
PipeAxis1 := MLPipeCreate( 'PipeAxis1' );
```

See Also

- "MLPipeAct" (→ p. 302)
- "MLPipeAddBlock" (→ p. 304)
- "MLPipeDeact" (→ p. 308)
- Pipe Network Functions for the PLC - MLPN_CREATE_OBJECTS

3.1.4.4 MLPipeDeact

Pipe Network ✓

 **Function** - Deactivates a pipe.

A Pipe contains an Input Pipe Block (Master, PMP, or Sampler), a Converter Output Pipe Block, and any Transformation Pipe Block that can be in between.

- The "MLPipeDeact" (→ p. 308) image shows two Pipes, both with the same Master Input Pipe Block.
 - The first ends with the first converter and has a Gear Pipe Block to transform the input values from the Master.
 - The second ends with the second converter and has a CAM Pipe Block to modify the input values from the Master.
- Once a Pipe is activated, history on the values in the Pipe's Blocks are lost and no longer updated.
 - A Converter object connected to a destination Axis object cannot send updated position values once its Pipe is deactivated.

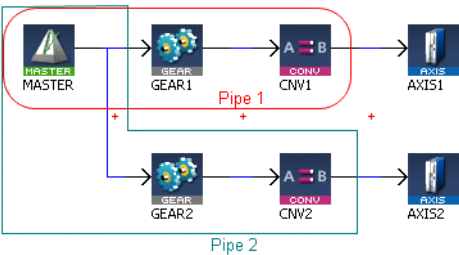


Figure 4-5: MLPipeDeact

NOTE

- All Pipes in the Pipe Network can be deactivated at once with the command PipeNetwork(MLPN_DEACTIVATE).
- This calls automatically generated code with **MLPipeDeact** commands for each Pipe object.
- In a multi-pipe program, only one command can be used to deactivate Pipes instead of writing code for each Pipe separately.

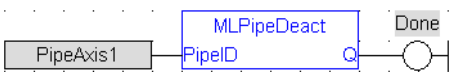
Inputs

Input	Data Type	Range	Unit	Default	Description
PipeID	DINT	1 to 1024	N/A	No default	ID number of a created Pipe object.

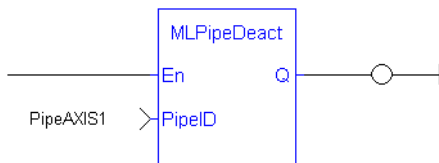
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Pipe is deactivated. •

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Deactivate a Pipe
MLPipeDeact( PipeAXIS1 );
```

See Also

- "MLCNVDisconnect" (→ p. 450)
- "MLPipeAct" (→ p. 302)
- "MLPipeAddBlock" (→ p. 304)
- "MLPipeCreate" (→ p. 306)
- Pipe Network Functions for the PLC - MLPN_DEACTIVATE

3.1.5 Axis

See "Axis Function Examples" (→ p. 407).

Name	Description
MLAxisAbs	Performs a move to an absolute position.
MLAxisAdd	A selected Axis performs a move for a specified distance relative to the endpoint of the previous move.
MLAxisAddress	Returns the motion bus address of the axis.
MLAxisAddTq	Set the additive torque value to the drive output (torque feed-forward).
MLAxisCfgFastIn	Configures the Fast Input for the axis by writing the expected settings in the Latch Control Word.
MLAxisCmdPos	Returns the reference position of the axis.
MLAxisDriveNumber	Returns the drive number associated with the axis or -1 if the function block failed.
MLAxisFBackPos	Returns the feedback position of the axis.
MLAxisGenEN	Enables or disables the internal TMP generator of the axis.
MLAxisGenIsEN	Verifies if the internal TMP generator of the axis is enabled.
MLAxisGenIsRdy	Verifies if an axis is ready.
MLAxisGenPos	Returns the generator position of the axis.
MLAxisGenReadAcc	Gets the acceleration of the internal generator of an axis.
MLAxisGenReadDec	Gets the deceleration of the internal generator of an axis.
MLAxisGenReadSpd	Gets the speed of the internal generator of an axis.
MLAxisGenWriteAcc	Sets the acceleration of the internal generator of an axis.
MLAxisGenWriteDec	Sets the deceleration of the internal generator of an axis.
MLAxisGenWriteSpd	Sets the speed of the internal generator of an axis.
MLAxisInit	Initializes an axis object.
MLAxisIsCnctd	Verifies if a pipe is currently connected to the axis.
MLAxisIsTriggered	Verifies if the axis received a trigger event.
MLAxisMoveVel	Jog at the specified speed.
MLAxisPipePos	Returns the pipe position of the axis.
MLAxisPower	Powers up or down the axis. Enables or disables a Servo or Stepper drive mapped to the axis.
MLAxisPowerDOff - Deprecated	This function has been deprecated.
MLAxisRatedTq	Allows conversion of drive torque values from rated torque units (1000 = rated motor continuous torque) to N.m (Newton meter).
MLAxisReadActPos	Returns the actual position of the axis.
MLAxisReadBoolParam	Read an Axis Boolean property of a Pipe Network Axis.
MLAxisReadFBUnit	Gets the feedback units per revolution value of the axis.

Name	Description
MLAxisReadFEUU	Returns the difference between the reference position and the actual position of the drive mapped to the specified axis.
MLAxisReadGenStatus	Returns the status of the internal generator of the axis.
MLAxisReadModPos	Gets the value period of the axis.
MLAxisReadParam	Returns the value of the specified axis parameter.
MLAxisReadTq	Returns the actual torque applied by the drive mapped to the specified axis.
MLAxisReadUUnits	Gets the User units per revolution value of the axis.
MLAxisReadVel	Returns the actual velocity of the axis based on the data provided by the drive's feedback device.
MLAxisReAlignRdy	Verifies if an axis is ready.
MLAxisReAlign	Realigns the actual position with the reference position by moving the axis by the specified delta position.
MLAxisRel	A selected Axis performs a move for a specified distance relative to the current position.
MLAxisResetErrors	Clears errors of the specified axis.
MLAxisRstFastIn	Writes in the Latch Control Word to reset the Fast Input.
MLAxisStatus	Returns the status of the axis.
MLAxisStop	Stop with the specified deceleration. This does not remove the input source, but does stop the drive from continuing to move.
MLAxisTimeStamp	Returns the timestamp of the triggered axis.
MLAxisWriteBoolParam	Write an Axis Boolean property of a Pipe Network Axis.
MLAxisWriteModPos	Sets the value period of the axis.
MLAxisWriteParam	Write an axis property for a Pipe Network Axis.
MLAxisWritePipPos	Forces the pipe position internal value.
	ⓘ IMPORTANT This function works only when no pipe is connected.
MLAxisWritePos	Sets a new value to an axis' current location.
MLAxisWriteUUnits	Set the user units per revolution value of the axis.
MLPNAxisCreate	Creates a new axis object.

3.1.5.1 Associated Data on Positions

[Click to view Associated Data for Axis](#)

This data are illustrated here:

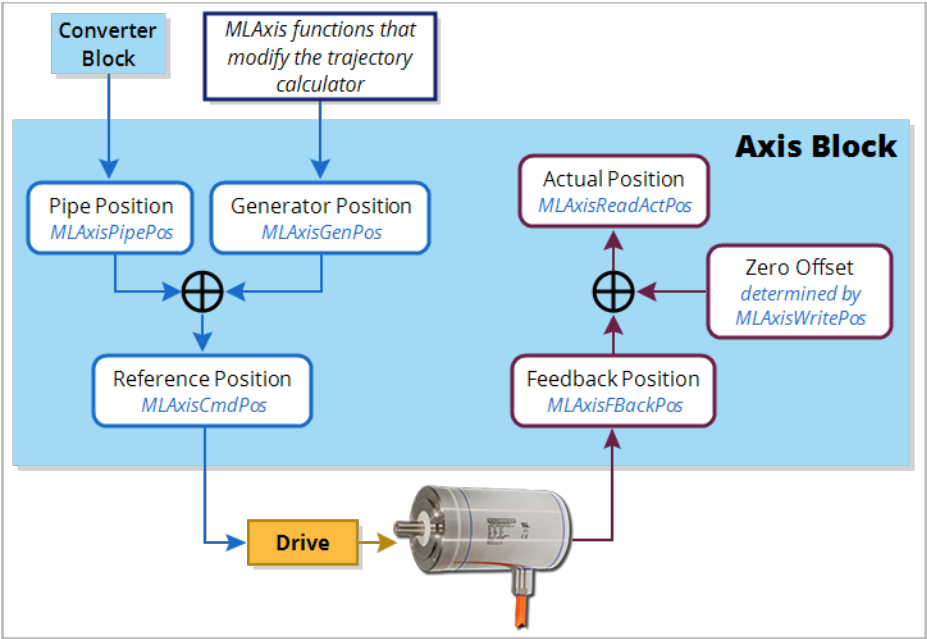


Figure 4-6: Axis Block Positions Diagram

NOTE	
All positions are in user units with modulo applied if active, unless specified.	
Position / Offset	Description
Actual Position	This is the actual position of the underlying axis as reported by the drive. - It is the sum of the feedback value (Position actual value) returned from the communication link to the drive and any zero-offset due to an MLWritePos function block. - See either "MLAxisWritePipPos" (→ p. 399) or "MLAxisWritePos" (→ p. 401). ActualPos := FeedbackPos + ZeroOffset
Feedback Position	- This is the current position the drive reports for an axis, scaled to user units. - It does not take into account the value of the Zero Offset or axis modulo.
Generator Position	This is the summation of all previous commands (i.e., calls to functions which perform motion) to the Axis internal motion generator. See either such as "MLAxisAbs" (→ p. 313), "MLAxisMoveVel" (→ p. 352), or "MLAxisRel" (→ p. 381). - It is modified by "MLAxisWritePos" (→ p. 401) to insure no jumps in the Reference Position command. - It accumulates changes in pipe position due to activate and deactivation of the pipe the Axis block is associated with.
Pipe Position	- This is the output of the convertor block is written into the Pipe Position value whenever the Convertor block is connected to the axis and the pipe is active. - See "Convertor" (→ p. 442).
Reference Position	- This is the commanded axis position sent to the drive. - It is the summation of Pipe Position and Generator Position. - ReferencePosition = Pipe Position + Generator Position
Zero Offset	This adjusts the coordinate system so the Actual Position reports correct values after homing or using "MLAxisWritePos" (→ p. 401).

3.1.5.2 MLAxisAbs

Pipe Network



Function - Performs a move to an absolute position.

- This function sets a new target Generator Position.
- Returns TRUE if the function succeeded.
- See "Axis Function Examples" (→ p. 407).

3.1.5.2.1.1 Position with Modulo On

NOTE

This information applies to both "MLAxisAbs" (→ p. 313) and "MLMstAbs" (→ p. 490).
For simplicity, the term Axis Block also refers to Master Block.

When the Modulo is turned on, the Axis Block moves to the targeted position during the corresponding period and is calculated as:

- If the Position input is between 0 (zero) and the Modulo position, the Axis Block moves within the **current** period (no position rollover).
- If the Position input is greater than the Modulo position, the Axis Block moves during one of the **next** period (positive position rollover).

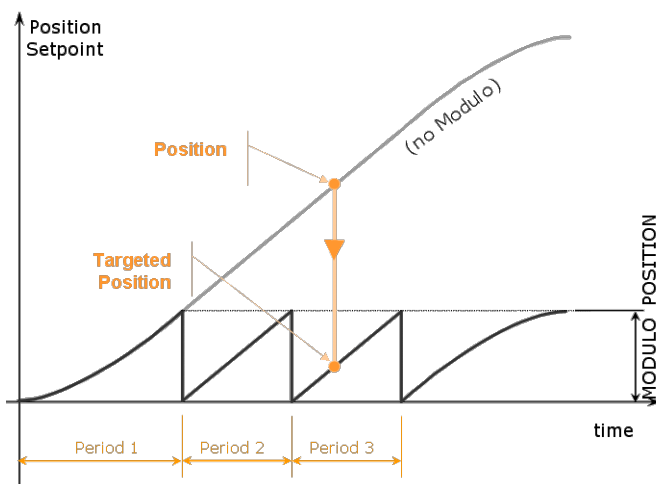


Figure 4-7: MLAxisAbs or MLMstAbs Modulo

The Axis Block works similarly for negative positions.

If the Position input is less than 0 (zero), the Axis Block moves during one of the **previous** period (negative position rollover).

3.1.5.3 Forcing the Direction of Rotation

In some applications, the direction of rotation for the axis is forced in one direction only.

As a consequence, the motor movement goes to the next or previous modulo in these situations:

3.1.5.3.1.1 End Position is Less Than Start Position

If the **End Position** is less than the **Start Position** and the direction of rotation for the axis is forced to be clockwise, the **red point** shows when the Modulo position is reached.

See Row 2 of the "MLAxisAbs" (→ p. 313).

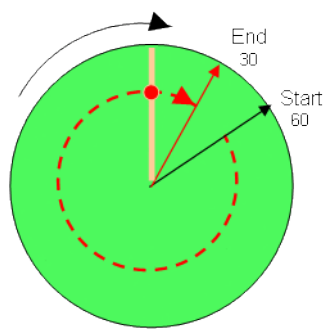


Figure 4-8: MAxisAbs or MLMstAbs - End Position Less Than Start Position

3.1.5.3.2.2 End Position is Greater Than Start Position

If the **End Position** is greater than the **Start Position** and the direction of rotation for the axis is forced to be counter clockwise.

See Row 4 of the "MAxisAbs" (→ p. 313).

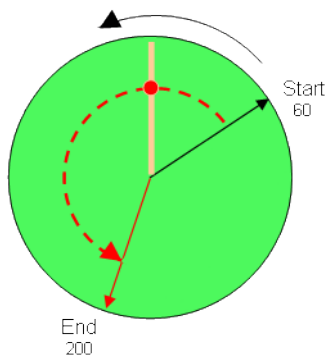


Figure 4-9: MAxisAbs or MLMstAbs - End Position Greater Than Start Position

Example Table

Start Position	End Position	Direction of Rotation	Cross Modulo	Position Input to MAxisAbs (1)	RelativeDistance Moved (2)	
60	200	Clockwise	No	200	140	(i.e., 200 - 60 + 0)
60	30	Clockwise	Yes	390	330	(i.e., 30 - 60 + 360)
60	30	Counter clockwise	No	30	-30	(i.e., 30 - 60 - 0)
60	200	Counter clockwise	Yes	-160	-220	(i.e., 200 - 60 - 360)

With:

- (1) **Position Input** = End Position (+ Modulo * Direction of rotation)
- (2) **Relative Distance Moved** = End Position - Start Position (+ Modulo * Direction of rotation)

Where:

Direction of rotation = 1 when clockwise and -1 when anti-clockwise.

3.1.5.3.3.3 Travel Speed Update with MAxisAbs

The travel speed of the generator can be updated using the function block "MAxisGenWriteSpd" (→ p. 343).

Depending on the state of the generator, this speed is directly reflected on the current move or a future move.

- If **MLAxisAbs** is not currently being executed, the new travel speed is applied for the trajectory calculation for a future **MLAxisAbs** command.
- If **MLAxisAbs** is currently being executed, and a new **MLAxisAbs** with the **same** target position is called, the new travel speed is taken into account only if the current state of the TMP profile is the constant velocity or acceleration.
 - If the axis was decelerating to stop at the goal position the new travel speed is not taken into account.
- If a **MLAxisAbs** is currently being executed, and a new **MLAxisAbs** with a **different** target position is called, the new travel speed is taken into account.

Example 1



1. First **MLAxisAbs**.
2. Second **MLAxisAbs**.
3. Initial Speed.
4. New Speed.

Figure 4-10: Initial speed is smaller than the new speed.

Example 2



1. First **MLAxisAbs**.
2. Second **MLAxisAbs**.
3. Initial Speed.
4. New Speed.

Figure 4-11: Initial speed is bigger than the new speed.

Example 3



1. Acceleration.
2. Constant Velocity.
3. Deceleration.

Figure 4-12: The speed update is taken into account only if the second **MLAxisAbs** is triggered during acceleration or constant velocity.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
Position	LREAL	No range	User units	No default	Sets the value of the absolute destination position. • See "MLAxisAbs" (→ p. 313). • See Set the Axis Block Position Units.

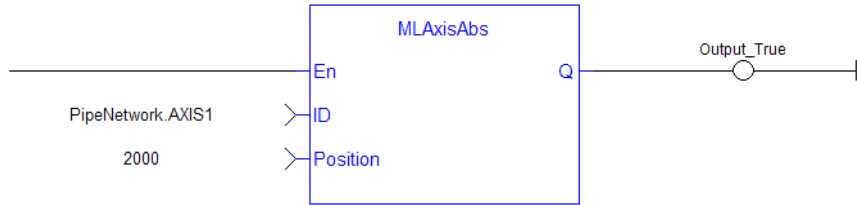
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLAxisAbs( PipeNetwork.Axis1, 2000 ) ;
```

See Also

- "MLAxisGenWriteAcc" (→ p. 339)
- "MLAxisGenWriteDec" (→ p. 341)
- "MLAxisGenWriteSpd" (→ p. 343)

3.1.5.4 MLAxisAdd

Pipe Network



Function - A selected Axis performs a move for a specified distance relative to the endpoint of the previous move.

- This function adds a relative distance to the current target Generator Position.
- See "Axis Function Examples" (→ p. 407).

3.1.5.4.1.1 DeltaPosition Axis

- The **DeltaPosition** input is signed so:
 - The move can be in the positive or negative direction.
 - The Axis moves this distance in User units.
- The travel speed, acceleration, deceleration, and User units of the move are values inherited from the selected Axis.
- The default settings are entered when an Axis is created and initiated, and can be changed with other MLAxis commands.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
DeltaPosition	LREAL	No range	User units	No default	• Sets the Axis Delta Position to add to the endpoint of the previous move. • See Set the Axis Block Position Units.

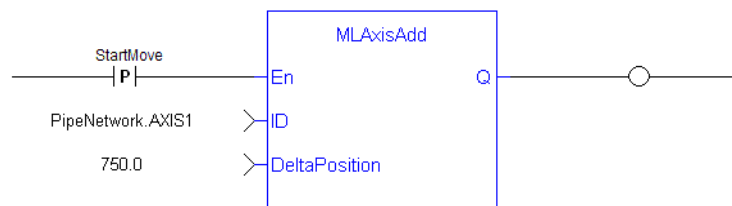
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes. • TRUE is after the motion profile is complete. •

FBD Language Example



FPLD Language Example



NOTE

- A positive transition contact is required to start the FB.
- See FFLD Contacts.

IL Language Example

Not available.

ST Language Example

```
MLAxisAdd(PipeNetwork.Axis1, LREAL#750.0 ) ;
```

See Also

- "MLAxisGenWriteAcc" (→ p. 339)
- "MLAxisGenWriteDec" (→ p. 341)
- "MLAxisGenWriteSpd" (→ p. 343)
- "MLAxisWriteUUnits" (→ p. 404)

3.1.5.5 MLAxisAddress

Pipe Network ✓



Function - Returns the motion bus address of the axis.

- Axes have the same address when they are mapped to the same multi-axis drive (e.g., the dual-axis AKD2G drive).
 - Example: If Axis1 is mapped to an AKD2G's Drive Axis Number 1 and Axis2 is mapped to the AKD2G's Drive Axis Number 2, both axes return the same address. (Figure 4-13)

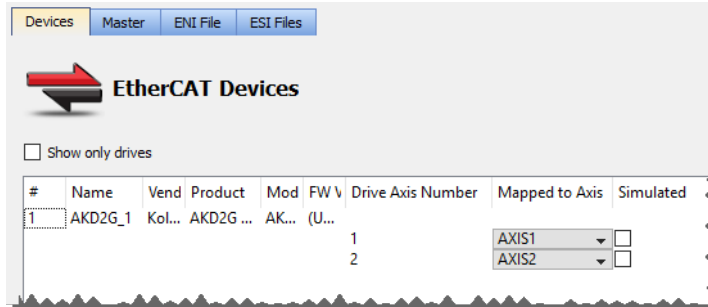


Figure 4-13: Example of Axis 1 and Axis 2

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

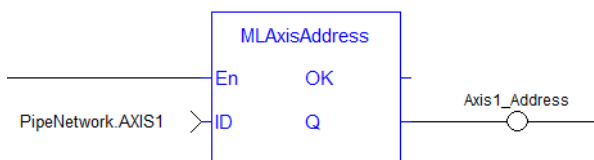
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes.
Default (.Q)	DINT	No range	N/A	Returns the motion bus address of the axis.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Axis1_Address := MlAxisAddress(PipeNetwork.Axis1);
```

3.1.5.6 MAxisAddTq

Pipe Network ✓



Function - Set the additive torque value to the drive output (torque feed-forward).

- This function is only active after the "MLAxisRatedTq" (→ p. 359) function has been invoked.
- Using the PDO, it also requires the IL.KBUSFF value to be set to 1 in the drive.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

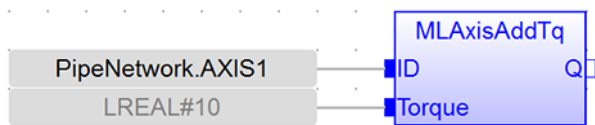
Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	Pipe Network identifier of the axis block.
Torque	LREAL	No range	• Rated torque units as used in the drive. • Example: Rated motor continuous torque X the Torque factor.	No default	Requested additive torque value in N.m (Newton meter).

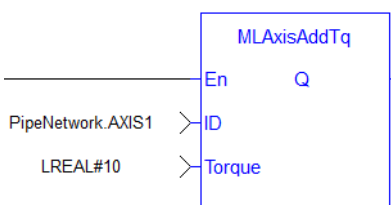
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLAxisAddTq(PipeNetwork.Axis1, LREAL#10 ) ;
```

See Also

"MLAxisRatedTq" (→ p. 359)

3.1.5.7 MAxisCfgFastIn

Pipe Network



Function - Configures the Fast Input for the axis by writing the expected settings in the Latch Control Word.

Fast Input can be armed on falling or rising edge.

3.1.5.7.1.1 Fast Homing - Inputs

See these topics:

- Example: Fast Homing with the Pipe Network Motion Engine Axis Pipe Block
- Example: Fast Homing with the PLCopen Motion Engine
- Fast Inputs with Pipe Network Motion
- H1-Pipe Network Registration and Fast Homing
- Example: Registration Position Capture with Pipe Network Trigger Block

Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	Enables execution.
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
InputID	DINT	See Description.	N/A	No default	ID of the FastInput of an axis (e.g., IN1 and IN2 on S300). Valid values of InputID are: • 0 = Touch Probe 1 / Capture Engine 0 • 1 = Touch Probe 2 / Capture Engine 1
Mode	DINT	0 to 2	N/A	No default	Configures the Fast Inputs as: • 0= Disabled • 1=Rising edge • 2=Falling edge

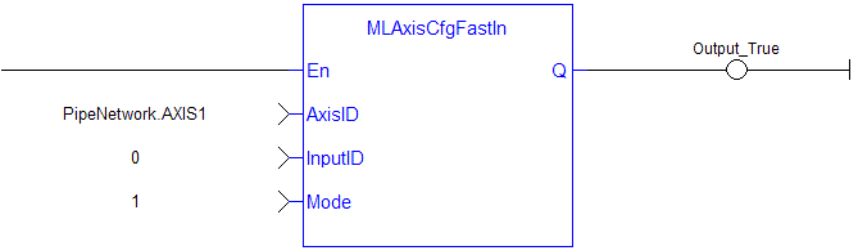
3.1.5.7.2 Outputs

Output	Data Type	Range	Unit	Description
Q	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. • Returns FALSE if the fast input could not be configured because of an invalid PDO mapping in the .XML file.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MAxisCfgFastIn( PipeNetwork.Axis1, 0, 1 ) ;
```

See Also

- "MLAxisIsTriggered" (→ p. 350)
- "MLAxisRstFastIn" (→ p. 384)

3.1.5.8 MAxisCmdPos

Pipe Network 



Function - Returns the reference position of the axis.

- The commanded position to an axis.
- Previous function name: MAxisRefPos

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

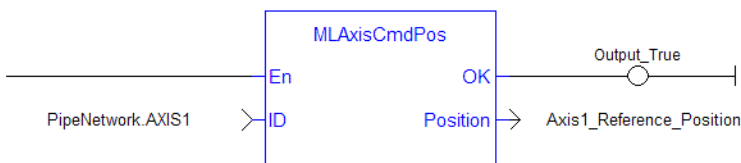
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes.
Position	LREAL	No range	User units	• Returns the Axis reference position. • See Set the Axis Block Position Units.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Axis1_ReferencePosition := MAxisCmdPos(PipeNetwork.AXIS1);
```

See Also

- "MLAxisFBackPos" (→ p. 328)
- "MLAxisGenPos" (→ p. 334)
- "MLAxisPipePos" (→ p. 354)
- "MLAxisReadActPos" (→ p. 361)
- "MLAxisWritePipPos" (→ p. 399)

3.1.5.9 MAxisDriveNumber

Pipe Network ✓



Function Block - Returns the drive number associated with the axis or -1 if the function block failed.



"MLPNAxisCreate" (→ p. 405) assigns the drive axis number.

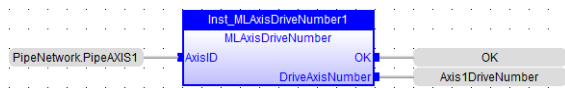
Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

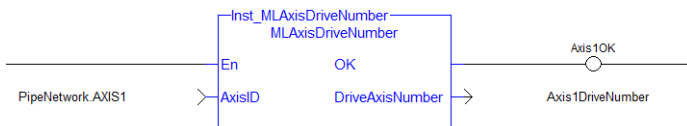
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes.
DriveAxisNumber	INT	-1 or 1,32767	N/A	Either: - The drive number associated with the axis. -1 if the function block failed.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```

Inst_MLAxisDriveNumber( AxisID)
IF Inst_MLAxisDriveNumber.OK Then
    Axis1DriveNumber := Inst_MLAxisDriveNumber.DriveAxisNumber
End_IF;

```

3.1.5.10 MLAxisFBackPos



Function - Returns the feedback position of the axis.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

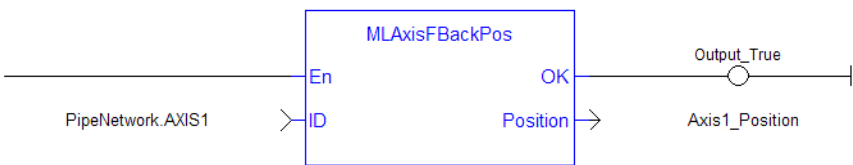
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes.
Position	LREAL	No range	User units	• Returns the feedback position of the axis. • See Set the Axis Block Position Units.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Axis1_Position := MLAxisFBackPos( PipeNetwork.Axis1 ) ;
```

See Also

- "MLAxisCmdPos" (→ p. 325)
- "MLAxisGenPos" (→ p. 334)
- "MLAxisPipePos" (→ p. 354)

- "MLAxisReadActPos" (→ p. 361)
- "MLAxisWritePipPos" (→ p. 399)

3.1.5.11 MLAxisGenEN



Function - Enables or disables the internal TMP generator of the axis.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
Enable	BOOL	FALSE, TRUE	N/A	No default	Boolean switch to activate the generator.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLAxisGenEN( PipeNetwork.Axis1, true) ;
```

See Also

"MLAxisGenIsEN" (→ p. 331)

3.1.5.12 MLAxisGenIsEN

Pipe Network



Function - Verifies if the internal TMP generator of the axis is enabled.

- Returns TRUE if the internal generator is enabled.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

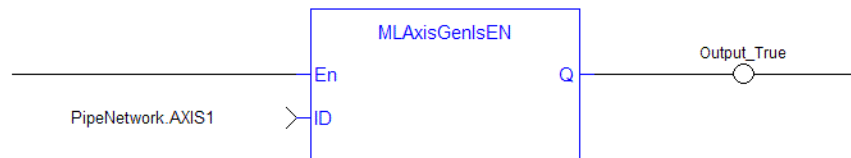
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLAxisGenIsEN(PipeNetwork.Axis1 ) ;
```

See Also

"MLAxisGenIsRdy" (→ p. 332)

3.1.5.13 MLAxisGenIsRdy



Function - Verifies if an axis is ready.

Specifies whether the Axis generator is ready.

- Returns TRUE if the internal generator axis is ready.
- See "Axis Function Examples" (→ p. 407).

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

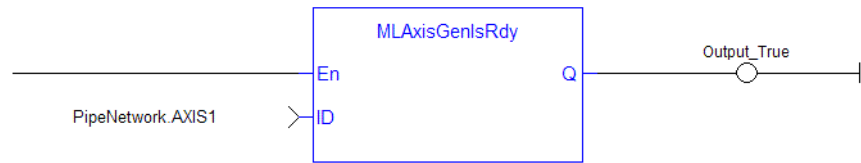
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLAxisGenIsRdy(PipeNetwork.Axis1 );
```

See Also

- "MLAxisGenIsEN" (→ p. 331)
- "MLAxisStatus" (→ p. 386)

3.1.5.14 MLAxisGenPos



Function - Returns the generator position of the axis.

- Returns TRUE if the internal generator axis is ready.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

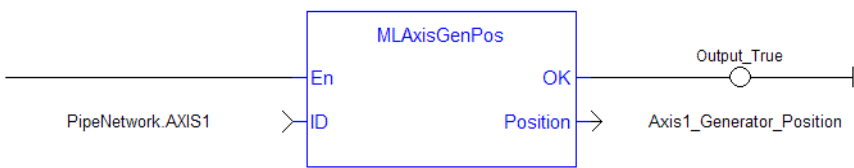
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •
Position	LREAL	No range	User units	• Returns the Axis generator position value. • See Set the Axis Block Position Units.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Axis1_Generator_Position := MLAxisGenPos(PipeNetwork.Axis1 ) ;
```

See Also

- "MLAxisCmdPos" (→ p. 325)
- "MLAxisFBackPos" (→ p. 328)
- "MLAxisPipePos" (→ p. 354)
- "MLAxisReadActPos" (→ p. 361)
- "MLAxisWritePipPos" (→ p. 399)

3.1.5.15 MAxisGenReadAcc



Function - Gets the acceleration of the internal generator of an axis.

Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

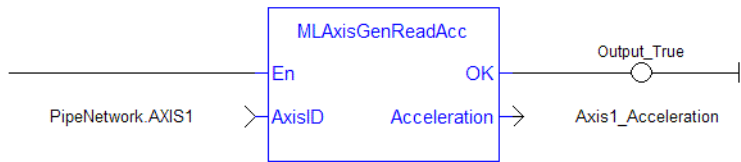
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •
Acceleration	LREAL	No range	User units / sec ²	• Returns the Axis Acceleration value. • See Set the Axis Block Position Units.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Axis1_Acceleration := MAxisGenReadAcc( PipeNetwork.Axis1 );
```

See Also

- "MAxisGenReadDec" (→ p. 337)
- "MAxisGenReadSpd" (→ p. 338)

3.1.5.16 MAxisGenReadDec

Pipe Network ✓



Function - Gets the deceleration of the internal generator of an axis.

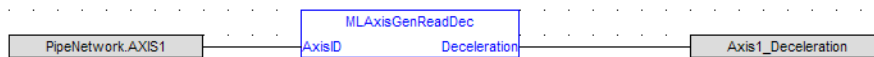
Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.
Deceleration	LREAL	No range	User units / sec ²	- Returns the Axis Deceleration value. - See Set the Axis Block Position Units.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Axis1_Deceleration := MAxisGenReadDec( PipeNetwork.Axis1 );
```

See Also

- "MAxisGenReadAcc" (→ p. 336)
- "MAxisGenReadSpd" (→ p. 338)

3.1.5.17 MAxisGenReadSpd



Function - Gets the speed of the internal generator of an axis.

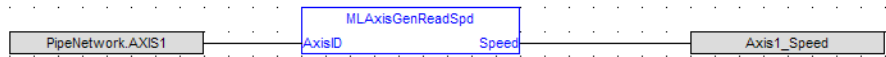
Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

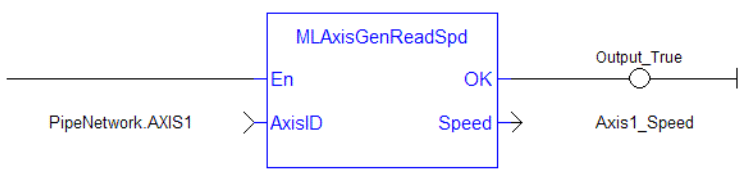
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •
Speed	LREAL	No range	User units / sec	• Returns the Axis Speed value. • See Set the Axis Block Position Units.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Axis1_Speed := MAxisGenReadSpd( PipeNetwork.Axis1 ) ;
```

See Also

- "MAxisGenReadAcc" (→ p. 336)
- "MAxisGenReadDec" (→ p. 337)

3.1.5.18 MAxisGenWriteAcc

Pipe Network ✓



Function - Sets the acceleration of the internal generator of an axis.

- Returns TRUE if the internal generator axis is ready.

Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
Acceleration	LREAL	No range	User units / sec ²	No default	• Sets the generator Acceleration value. • See Set the Axis Block Position Units.

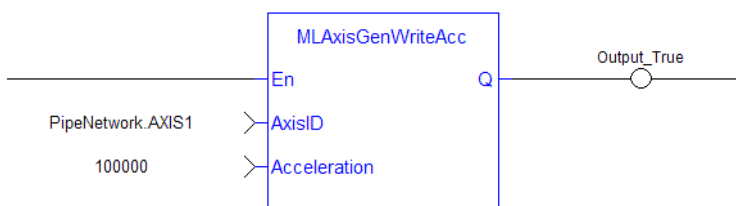
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MAxisGenWriteAcc(PipeNetwork.Axis1, 100000 ) ;
```

See Also

- "MLAxisGenWriteDec" (→ p. 341)
- "MLAxisGenWriteSpd" (→ p. 343)

3.1.5.19 MLAxisGenWriteDec

Pipe Network ✓



Function - Sets the deceleration of the internal generator of an axis.

Returns TRUE if the internal generator axis is ready.

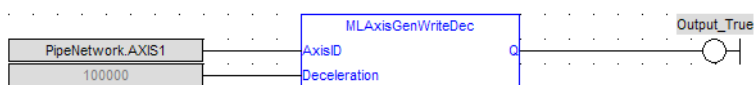
Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
Deceleration	LREAL	No range	User units / sec ²	No default	Sets the generator Deceleration value. - The axis deceleration rate is limited so the velocity cannot change by more than the value of the declared velocity limit in a single iteration. - The Pipe Network Axis block uses the TRAVEL_SPEED parameter to scale this limit. - The maximum deceleration is affected by the Pipe Network Axis Block parameter TRAVEL_SPEED and the axis update rate. - See Set the Axis Block Position Units.

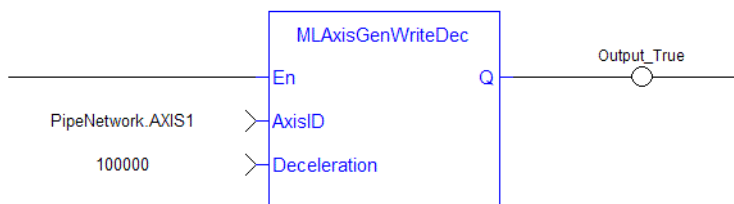
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLAxisGenWriteDec(PipeNetwork.Axis1, 100000 ) ;
```

See Also

- "MLAxisGenWriteAcc" (→ p. 339)
- "MLAxisGenWriteSpd" (→ p. 343)

3.1.5.20 MAxisGenWriteSpd

Pipe Network ✓



Function - Sets the speed of the internal generator of an axis.

This function or function block does not generate any motion.

Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
Speed	LREAL	No range	User units / sec	No default	- Sets the generator Speed value. - See Set the Axis Block Position Units.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MAxisGenWriteSpd(PipeNetwork.Axis1, 500 ) ;
```

See Also

- "MLAxisGenWriteAcc" (→ p. 339)
- "MLAxisGenWriteDec" (→ p. 341)

3.1.5.21 MLAxisInit


 **Function** - Initializes an axis object.

- Returns TRUE if the function succeeded.
- The axis object can be mapped to servo or stepper drives.

Inputs

Input	Data Type	Range	Unit	Default	Description
Acceleration	LREAL	Acceleration > 0 (zero)	User units / sec ²	No default	• Sets the Axis Acceleration value. • See Set the Axis Block Position Units.
AxisID	DINT	No range	N/A	No default	ID of the Axis block.
Deceleration	LREAL	Deceleration > 0 (zero)	User units / sec ²	No default	• Sets the Axis Deceleration value. • See Set the Axis Block Position Units.
FeedbackUnitsPerRev	DINT	No range	User units	No default	• Number of units associated to the Drive for one revolution of the physical motor. • This is only available when using an imported ENI file. • This parameter is only configurable through "MLAxisInit" (→ p. 345).
InitialPosition	LREAL	No range	User units	No default	Initial position value expressed in user logical units. • Used only at the pipe activation to initialize the position starting point. • This parameter is only configurable through "MLAxisInit" (→ p. 345). • See Set the Axis Block Position Units.

Input	Data Type	Range	Unit	Default	Description
Modulo	BOOL	FALSE, TRUE	N/A	No default	The available modes are Modulo (TRUE) and No Modulo (FALSE). Depending on the type of the moving object the axis acts on, you can define the MODULO_POSITION parameter or not. Modulo - Moving objects, performing a never ending cyclical motion are called periodic (e.g., printing cylinder, cutting wheel). - Example: If a user unit = 0.1mm has been chosen, a Modulo Position = 2000 Units could be selected for this transportation system.  Figure 4-14: Mode - Modulo No Modulo - Objects always moving within a certain position range (forward/backwards) can be called linear or range axes (e.g., lift axis, moving tables). - Example: If a user unit = 0.1mm has been chosen, a position range = 0 to 10,000 Units could be selected for this moving table.  Figure 4-15: Mode - No Modulo
ModuloPosition	LREAL	No range	User units	No default	- The value of the period of a cyclic system expressed in User units. - The parameter is defined to correctly manage the periodicity (modulo) of the input values.

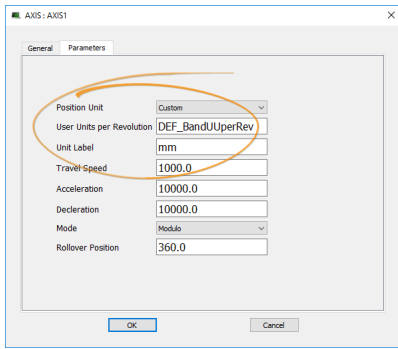
Input	Data Type	Range	Unit	Default	Description
Speed / Travel Speed	LREAL	No range	User units	No default	- The speed value is used to set the constant speed part of the trapezoidal motion profile. - See Set the Axis Block Position Units.
UserUnitsPerRev / User Units per Revolution	LREAL	No range	User units	No default	To divide the current axis into graduations adapted to your project, you must define the unit that is equivalent to one revolution of the physical motor (e.g., 3600 means that you define the user unit to be tenth of a degree). You can rely on expressions to define values. Example with Expressions Gear factor 1:3 and 1000.0 User Units per one gear shaft revolution. <pre>// user units per revolution calculation example #define DEF_BandGear 3.0 // gearbox ratio #define DEF_BandUnit 1000.0 // user units for 1 mechanical turn #define DEF_ BandUUpperRev DEF_ BandUnit/DEF_ BandGear</pre> 

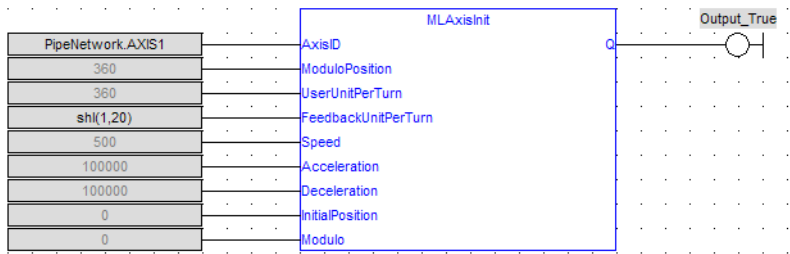
Figure 4-16: Define Value with Expressions

See Use the Defines List.

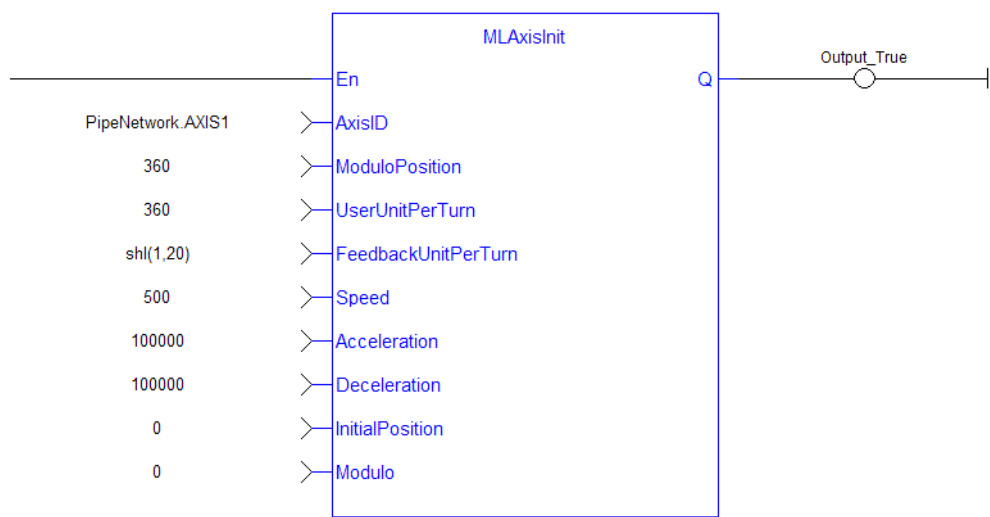
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
MAxisInit( PipeNetwork.Axis1, 360.0, 360.0, SHL(1,20), 500.0, 100000.0, 100000.0, 0.0, true ) ;
```

3.1.5.22 MAxisIsCnctd

Pipe Network



Function - Verifies if a pipe is currently connected to the axis.

Returns TRUE if a pipe is connected.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

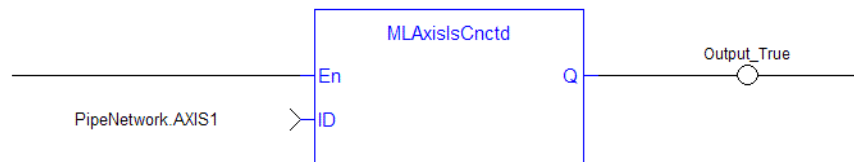
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MAxisIsCnctd(PipeNetwork.Axis1 ) ;
```

3.1.5.23 MLAxisIsTriggered



Function - Verifies if the axis received a trigger event.
Returns TRUE if the Fast Input event has been **triggered** and not yet been reset.

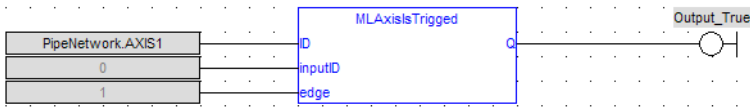
Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
InputID	DINT	See Description.	N/A	No default	ID of the triggered Fast input of an axis (i.e., IN1 and IN2 on S300). Valid values of InputID are: 0 = Touch Probe 1 / Capture Engine 0 1 = Touch Probe 2 / Capture Engine 1
edge	DINT	See Description.	N/A	No default	Configures the Inputs as: 0 = Disabled. 1 = Rising edge. 2 = Falling edge.

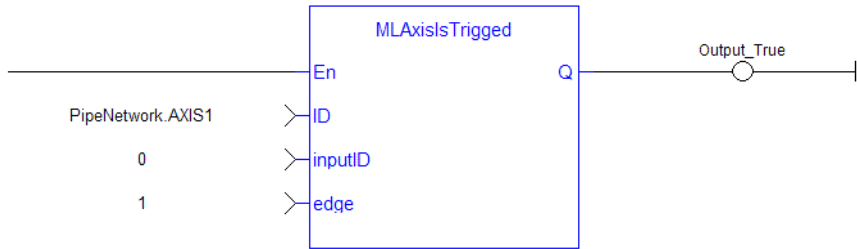
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLAxisIsTriggered (PipeNetwork.Axis1, 0,1 ) ;
```

See Also

"MLAxisRstFastIn" (→ p. 384)

3.1.5.24 MAxisMoveVel



Function - Jog at the specified speed.

- This function sets the generator position moving at a programmed speed.
- Previous function name: MAxisRun
- Returns TRUE if the function succeeded.
- See "Axis Function Examples" (→ p. 407).

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
Speed	LREAL	No range	User units / sec	No default	• Sets the axis speed. • See Set the Axis Block Position Units.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. • This is TRUE after the motion has reached jog speed. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MAxisMoveVel(PipeNetwork.Axis1, 500 ) ;
```

See Also

- "MLAxisGenWriteAcc" (→ p. 339)
- "MLAxisGenWriteDec" (→ p. 341)
- "MLAxisGenWriteSpd" (→ p. 343)

3.1.5.25 MLAxisPipePos



Function - Returns the pipe position of the axis.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
Speed	LREAL	No range	User units / sec	No default	• Sets the axis speed. • See Set the Axis Block Position Units.

Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •
Position	LREAL	No range	User units	See Set the Axis Block Position Units.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Axis1_Pipe_Position := MLAxisPipePos (PipeNetwork.Axis1 ) ;
```

See Also

- "MLAxisCmdPos" (→ p. 325)
- "MLAxisFBackPos" (→ p. 328)
- "MLAxisGenPos" (→ p. 334)
- "MLAxisReadActPos" (→ p. 361)
- "MLAxisWritePipPos" (→ p. 399)

3.1.5.26 MAxisPower



Function - Powers up or down the axis.

IMPORTANT

- Powering on an axis affects the position and motion state of an axis.
 - Do now call **MAxisPower** with the **On** input flag set to **True** while the axis is in motion.
-
- Enables or disables a Servo or Stepper drive mapped to the axis.
 - Previous Function Names:
 - MAxisPowerOn
 - MAxisPowerOff
 - When the axis is powered up:
 - The **ReferencePosition** is modified to equal the **ActualPosition**.
 - KAS updates the **GeneratorPosition**.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
On	BOOL	FALSE, TRUE	N/A	No default	Flag to: • Power up (TRUE) the axis. • Power down (FALSE) the axis.

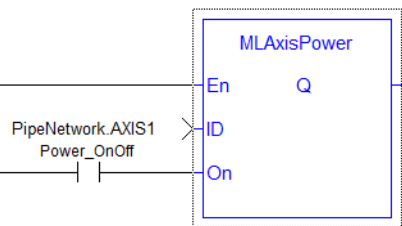
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* If Power_OnOff is TRUE then power in ON, otherwise OFF*)  
MLAxisPower( PipeNetwork.Axis1, Power_OnOff) ;
```

See Also

"MLAxisPowerDOff - Deprecated" (→ p. 358)

3.1.5.27 MLAxisPowerDOff - Deprecated

Pipe Network ✓



Function - This function has been deprecated.

3.1.5.28 MAxisRatedTq

Pipe Network



Function - Allows conversion of drive torque values from rated torque units (1000 = rated motor continuous torque) to N.m (Newton meter).

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	Pipe network identifier of the axis block.
Torque	LREAL	See Description.	N.m (Newton meter)	No default	Actual torque applied by the drive associated to the axis. Rated torque = Nominal Drive Current * Torque factor = DRV.ICONT * MOTOR.KT. About SDO DRV.ICONT is obtained by SDO parameter: index 5083h (sub-index 0). MOTOR.KT is obtained by SDO parameter: index 3593h (sub-index 0). See: • Communication SDOs. • Manufacturer specific SDOs. • Profile specific SDOs. To read/write an SDO object with an index greater than 16#7FFF (32767), the value must be entered in this form: <code>any_to_int(index # in hex format).</code> Example: <code>any_to_int(16#8321).</code> • The actual units of DRV.ICONT and MOTOR.KT are 1/1000 of the actual values if obtained by SDO. • The formula, if using the SDO values, is: • Rated Torque = Torque = (SDO (DRV.ICONT)/1000) * (SDO (MOTOR.KT)/1000).

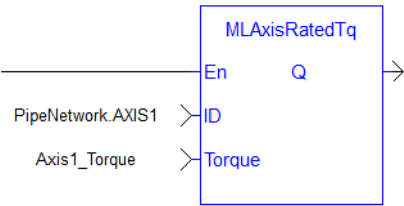
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MAxisRatedTq(PipeNetwork.Axis1, Axis1_Torque ) ;
```

See Also

"MAxisReadTq" (→ p. 374)

3.1.5.29 MAxisReadActPos

Pipe Network ✓



Function - Returns the actual position of the axis.

- Previous function name: MAxisActualPos

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

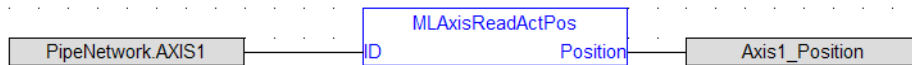
Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

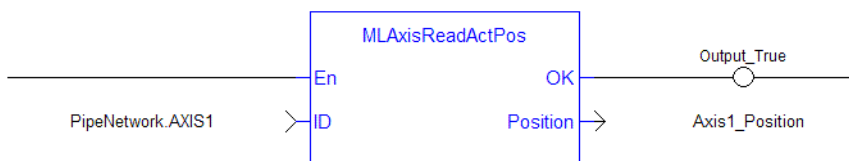
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •
Position	LREAL	No range	User units	• Returns the absolute position of the axis. • See Set the Axis Block Position Units.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Axis1_Position := MAxisReadActPos ( PipeNetwork.Axis1 ) ;
```

See Also

- "MLAxisCmdPos" (→ p. 325)
- "MLAxisFBackPos" (→ p. 328)
- "MLAxisGenPos" (→ p. 334)
- "MLAxisPipePos" (→ p. 354)
- "MLAxisWritePipPos" (→ p. 399)

3.1.5.30 MLAxisReadBoolParam



This function/function block was added in KAS v4.02.



Function Block - Read an Axis Boolean property of a Pipe Network Axis.

Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
ParameterNumber	INT	-32768 to +32767	N/A	No default	See "Axis Parameters" (→ p. 363).

Outputs

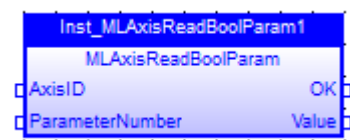
Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.
Value	BOOL	FALSE, TRUE	N/A	Value read.

Axis Parameters

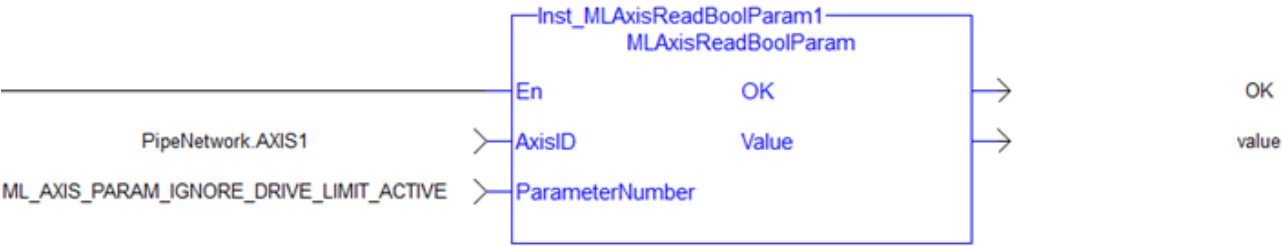
ID	Parameter	Name	R/W	Update Rate Type	Description
1038	ML_AXIS_PARAM_IGNORE_DRIVE_LIM_STATUS	Ignore Drive Limit Active	Read / Write	Static	- The default value is FALSE. - When TRUE, the Pipe Network motion engine ignores the DS-402 Status word bit #11 (Internal Limit Active). - See Overtravel Conditions.

ID	Parameter	Name	R/W	Update Rate Type	Description
1039	ML_AXIS_PARAM_DRIVE_OVERRIDE_ON_LIMIT_ACTIVE	Drive Override on Limit Active	Read / Write	Static	- Set to TRUE if the drive ignores controller demand position when DS-402 Status word bit #11 (Internal Limit Active) is on. - Set to FALSE if the Pipe Network is responsible for stopping an axis when DS-402 Status word bit #11 is on. - The default value for axes connected to AKD-family drives is TRUE. - For all other axes, the default value is FALSE.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
(*Check if Axis Drive Limit Active is ignored*)
Inst_MLAxisReadBoolParam(PipeNetwork.AXIS1, ML_AXIS_PARAM_IGNORE_DRIVE_
LIM_STATUS);
IF Inst_MLAxisReadBoolParam.OK THEN
```

```
value := Inst_MLAxisReadBoolParam.Value;  
END_IF;
```

See Also

"MLAxisWriteBoolParam" (→ p. 394)

3.1.5.31 MAxisReadFBUnit



Function - Gets the feedback units per revolution value of the axis.

Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •
FBUnitsPerRev	LREAL	> 0 (zero)	Feedback units / revolution	Returns the Axis Feedback Units per revolution.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Axis1_Feedback_Units := MAxisReadFBUnit(PipeNetwork.Axis1 ) ;
```

3.1.5.32 MAxisReadFEUU

Pipe Network ✓



Function - Returns the difference between the reference position and the actual position of the drive mapped to the specified axis.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	Pipe network identifier of the axis block.

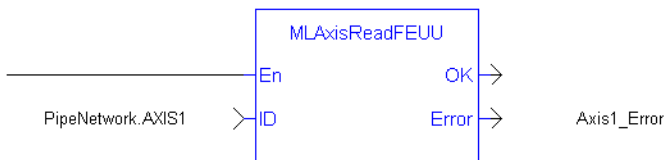
Outputs

Output	Data Type	Range	Unit	Description
Error	LREAL	No range	User units	Returns the difference between the reference position and the actual position of the drive mapped to the specified axis.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Axis1_Error := MAxisReadFEUU(PipeNetwork.Axis1) ;
```

See Also

"MAxisReadActPos" (→ p. 361)

3.1.5.33 MAxisReadGenStatus



Function - Returns the status of the internal generator of the axis.

The status of the Axis generator: acceleration, run, deceleration, change designation point, single step.

- Previous function name: MAxisGenStatus

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •
Default (.Q)	DINT	Enumerated	N/A	• Shows the status of the internal generator based on the "Internal Generator Status" (→ p. 368) table. •

Remarks

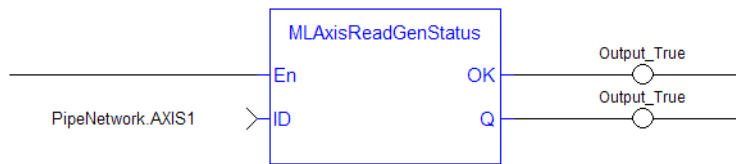
3.1.5.33.1.1 Internal Generator Status

Status	Description
0	RUN mode (acceleration).
1	RUNNING or STOPPED.
2	MOVE: Changing move destination.
3	MOVE: Changing move destination.
4	MOVE: Acceleration.
5	MOVE: Constant speed (travel speed).
6	MOVE: Deceleration.
7	MOVE: Single step (micro movement).

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLAxisReadGenStatus (PipeNetwork.Axis1 ) ;
```

See Also

- "MLAxisGenIsRdy" (→ p. 332)
- "MLAxisStatus" (→ p. 386)

3.1.5.34 MLAxisReadModPos



Function - Gets the value period of the axis.

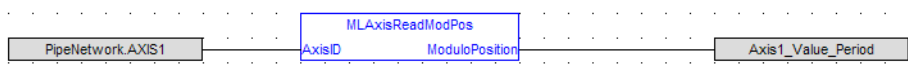
Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.
ModuloPosition	LREAL	No range	User units	Returns the Axis Value Period.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Axis1_Value_Period := MLAxisReadModPos (PipeNetwork.Axis1 ) ;
```

3.1.5.35 MLAxisReadParam



This function/function block was added in KAS v4.02.



Function Block - Returns the value of the specified axis parameter.

Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
ParameterNumber	INT	Enumerated	N/A	No default	See "Axis Parameters" (→ p. 371).

Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.
Value	LREAL	No range	N/A	Value of the axis parameter.

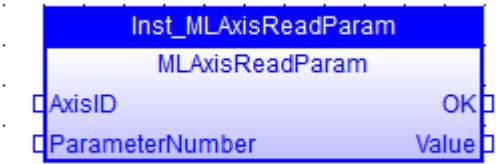
Remarks

Axis Parameters

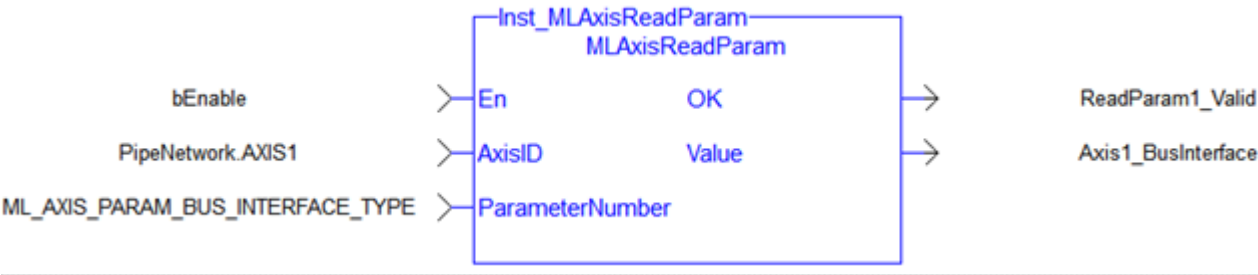
ID	Parameter	Name	R/W	Update Rate Type	Description
1035	ML_AXIS_PARAM_BUS_INTERFACE_TYPE	Bus Interface Type	Read-only	Static	The parameter value is: • DRIVE_BUS_INTERFACE_ECAT when the axis is associated with an EtherCAT device. • DRIVE_BUS_INTERFACE_SIMULATOR when the axis is simulated.
1500	ML_AXIS_PARAM_BUS_POSITION_DEMAND	Position demand value	Read-only	EtherCAT Cyclic	Drive Position Demand Value 32-bits. NOTE There may be a 1 cycle delay.

ID	Parameter	Name	R/W	Update Rate Type	Description
1501	ML_AXIS_PARAM_BUS_STATUS_WORD	Drive Status Word	Read-only	EtherCAT Cyclic	Drive Axis Status Word 16-bits. - DS-402 compatible drives: - This is the same as the CANopen Status Word. - Stepper Drives: - This is the same as the STM Inputs (CoE object 0x6010). - See the AKT2G-SM-Lx Object Description. - Simulated Drives: N/A NOTE There may be a 1 cycle delay.
1502	ML_AXIS_PARAM_BUS_CONTROL_WORD	Drive Control Word	Read-only	EtherCAT Cyclic	Drive Axis Control Word 16-bits. - DS-402 compatible drives: - This is the same as the CANopen Control Word. - Stepper Drives: - This is the same as the STM Outputs (CoE object 0x7010). - See the AKT2G-SM-Lx Object Description. - Simulated Drives: N/A NOTE There may be a 1 cycle delay.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* Check if Pipe axis is associated with an EtherCAT device or simulated *)
Inst_MLAxisReadParam(PipeNetwork.AXIS1, ML_AXIS_PARAM_BUS_INTERFACE_TYPE);
IF Inst_MLAxisReadParam.OK THEN
    Axis1_Simulated := Inst_MLAxisReadParam.Value = DRIVE_BUS_INTERFACE_
SIMULATOR;
END_IF;
```

See Also

"MLAxisReadBoolParam" (→ p. 363)

3.1.5.36 MLAxisReadTq



Function - Returns the actual torque applied by the drive mapped to the specified axis.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	Pipe network identifier of the axis block.

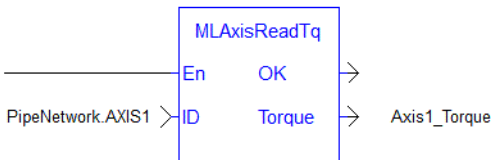
Outputs

Output	Data Type	Range	Unit	Description
Torque	LREAL	No range	N.m (Newton meter)	• Actual torque applied by the drive associated to the axis. • If the "MLAxisRatedTq" (→ p. 359) function is not invoked, the Output value is rated motor continuous torque (where 1000.0 = rated torque).

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Axis1_Torque := MLAxisReadTq(PipeNetwork.Axis1 ) ;
```

See Also

- "MLAxisRatedTq" (→ p. 359)
- "MLAxisReadActPos" (→ p. 361)
- "MLAxisReadVel" (→ p. 376)

3.1.5.37 MLAxisReadUUnits

Pipe Network ✓



Function - Gets the User units per revolution value of the axis.

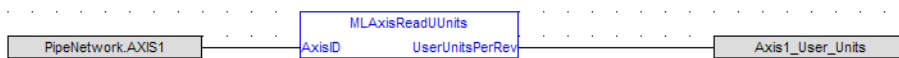
Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

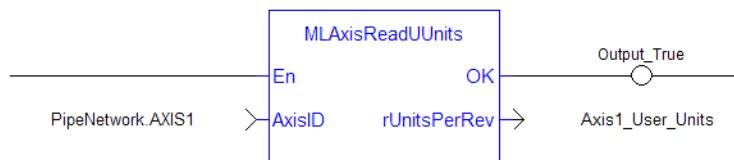
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.
UserUnitsPerRev	LREAL	> 0 (zero)	UserUnitsPerRev / User Units per Revolution	Returns the Axis User Units per revolution.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
Axis1_User_Units := MLAxisReadUUnits(PipeNetwork.Axis1) ;
```

3.1.5.38 MLAxisReadVel



Function - Returns the actual velocity of the axis based on the data provided by the drive's feedback device.
In AKD, S300, and S700 drives:

- The actual velocity is calculated internally by the drive.
- The **Velocity Actual Value** object (CoE object 0x606C, subindex 0) must be included in the drive's Input (Tx) PDO data for the controller to read the axis actual velocity from the drives.
 - This is added using the PDO Editor tab.
- By default, the **Velocity Actual Value** object is included in the AKD PDOs 0x1B20, 0x1B22, and 0x1B23.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	Pipe network identifier of the axis block.

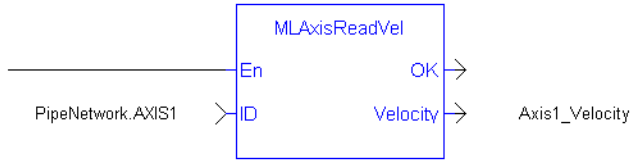
Outputs

Output	Data Type	Range	Unit	Description
Velocity	LREAL	No range	User units / sec	The actual velocity of the axis.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Axis1_Velocity := MLAxisReadVel(PipeNetwork.Axis1 ) ;
```

See Also

- "MLAxisReadActPos" (→ p. 361)
- "MLAxisReadTq" (→ p. 374)

3.1.5.39 MAxisReAlignRdy



Function - Verifies if an axis is ready.

- Returns TRUE if the internal realignment axis is ready.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

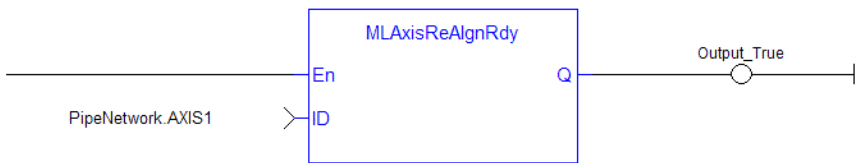
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MAxisReAlignRdy(PipeNetwork.Axis1 ) ;
```

See Also

"MAxisReAlign" (→ p. 379)

3.1.5.40 MAxisReAlign



Function - Realigns the actual position with the reference position by moving the axis by the specified delta position.

- Causes the Axis to move by a programmed amount without changing the Reference Position following an MAxisStop.
- Allows the Pipe Position to be used following an MAxisStop.

NOTE

The realign function does not work properly if the "MAxisStop" (→ p. 389) function is continuously executed using its Start input.

- When stopping the drive a motion profile is applied to decelerate.
- During the deceleration, the Reference position changes.
- Calling **MAxisReAlign** realigns the actual position with the reference position by moving the axis by the specified delta position, typically calculated by the application code.
- After a "MAxisStop" (→ p. 389) is executed, **MAxisReAlign** is required for the Pipe Position to be used again.

Inputs

Input	Data Type	Range	Unit	Default	Description
Acceleration	LREAL	No range	User units / sec ²	No default	• Sets the Realign Acceleration. • See Set the Axis Block Position Units.
Deceleration	LREAL	No range	User units / sec ²	No default	• Sets the Realign Deceleration rate. • See Set the Axis Block Position Units.
DeltaPos	LREAL	No range	User units	No default	• Sets the Axis Delta Position or the relative distance to be moved. • See Set the Axis Block Position Units.
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
Speed	LREAL	No range	User units / sec	No default	• Sets the Axis Speed. • See Set the Axis Block Position Units.

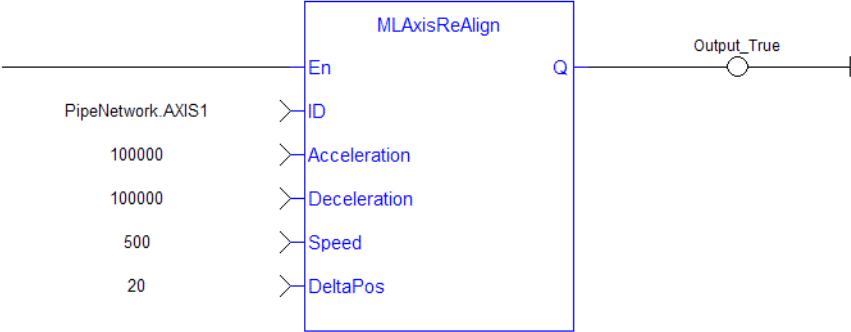
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLAxisReAlign(PipeNetwork.Axis1, 100000, 100000, 500, 20 ) ;
```

See Also

"MLAxisReAlignRdy" (→ p. 378)

3.1.5.41 MAxisRel

Pipe Network



Function - A selected Axis performs a move for a specified distance relative to the current position.

- This function adds a relative distance to the current Generator Position.
- Use "MAxisGenIsRdy" (→ p. 332) to know when a move has completed.
 - The output of **MAxisRel** can occur before moves have finished.
- See "Axis Function Examples" (→ p. 407).

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
DeltaPosition	LREAL	No range	User units	No default	• Sets the Axis Delta Position or the relative distance to be moved. • See Set the Axis Block Position Units.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes. • This occurs immediately after the function is called. • The function does not wait for the motion profile to be completed. •

Remarks

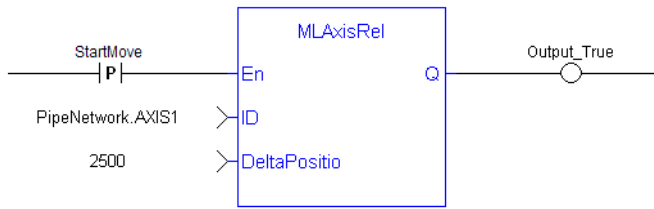
3.1.5.41.1.1 DeltaPosition Axis

- The **DeltaPosition** input is signed so:
 - The move can be in the positive or negative direction.
 - The Axis moves this distance in User units.
- The travel speed, acceleration, deceleration, and User units of the move are values inherited from the selected Axis.
- The default settings are entered when an Axis is created and initiated, and can be changed with other MAxis commands.

FBD Language Example



FPLD Language Example

**NOTE**

- A positive transition contact is required to start the FB.
- See FFLD Contacts.

IL Language Example

Not available.

ST Language Example

```
MLAxisRel(PipeNetwork.Axis1, 2500 ) ;
```

See Also

- "MLAxisGenWriteAcc" (→ p. 339)
- "MLAxisGenWriteDec" (→ p. 341)
- "MLAxisGenWriteSpd" (→ p. 343)
- "MLAxisWriteUUnits" (→ p. 404)

3.1.5.42 MAxisResetErrors

Pipe Network ✓



Function - Clears errors of the specified axis.

- Previous function name: MAxisClrErrors

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

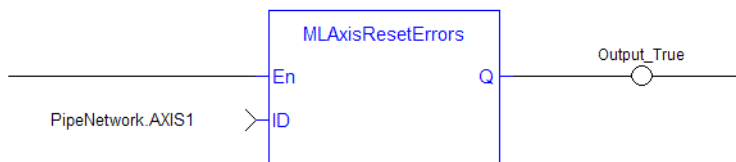
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MAxisResetErrors( PipeNetwork.Axis1 ) ;
```

3.1.5.43 MAxisRstFastIn



Function - Writes in the Latch Control Word to reset the Fast Input.

Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
InputID	DINT	See Description.	N/A	No default	ID name of the Fast input of an axis (i.e., IN1 and IN2 on S300). Valid values of InputID are: • 0 = Touch Probe 1 / Capture Engine 0 • 1 = Touch Probe 2 / Capture Engine 1

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MAxisRstFastIn (PipeNetwork.Axis1, 0 ) ;
```

See Also

- "MLAxisCfgFastIn" (→ p. 323)
- "MLAxisIsTriggered" (→ p. 350)

3.1.5.44 MLAxisStatus



 **Function** - Returns the status of the axis.

This includes information on the:

- Enabled/disabled state.
- Bus connection.
- Pipe Network connection.
- Drive executing an axis stop function.
- Other status information.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes.
Default (.Q)	DINT	No range	N/A	• Returns the status of the axis. • See "Bit Descriptions" (→ p. 386).

Remarks

3.1.5.44.1.1 Bit Descriptions

[Click to see Bit Descriptions](#)

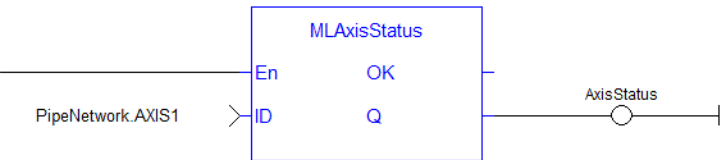
Bit	Description
0	Initialized (1 if initialized).
1	• Power (1 if power is on). • Is linked to Bit 1 (Switched on) of the Status Word.
2	• Enabled (1 if enabled). • Is linked to Bit 0 (Ready to switch on) of the Status Word.
3	• Found (1 if found on the network). • EtherCAT state is Pre-Operational.
4	• Configured (1 if configured) EtherCAT state is Safe-Operational.

Bit	Description
5	Running (1 if running) EtherCAT state is Operational.
6	Error (1 if in error).
7	Simulated (1 if working with a simulated axis).
8	Connected (1 if a pipe is connected).
9	Warning (1 if the drive signals a warning).
10	Stopping (1 if the drive is performing a Stop).
11	Stopped (1 if the drive has finished the Stop).
12	Axis is turning and has not reached target location.
13	Drive Internal limit active rising edge. - This bit is only set on the rising edge of bit #11 in the CANopen Status Word. - It remains set for only one motion cycle. NOTE The drive internal limit state changes are monitored only when the axis parameter MC_AXIS_PARAM_IGNORE_DRIVE_LIM_STATUS is set to FALSE. See Overtravel Conditions.
14	Drive Control Override. This bit is equivalent to bit #11 in the CANopen Status Word. NOTE The drive internal limit state changes are monitored only when the axis parameter MC_AXIS_PARAM_IGNORE_DRIVE_LIM_STATUS is set to FALSE. See Overtravel Conditions.
15 to 31	Reserved.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
AxisStatus := MLAxisStatus(PipeNetwork.AXI_A1_Axis) ;
IF AxisStatus.11 THEN
MLAxisStop(PipeNetwork.AXI_A1_Axis,FALSE,DEF_A1_StopDec) ;
END_IF;
```

```
AxisStatus := MLAxisStatus(PipeNetwork.AXIS1);
If AxisStatus.0 Then
    (*Axis is initialized*)
ElsIf AxisStatus.1 Then
    (*Axis' power is ON*)
ElsIf AxisStatus.2 Then
    (*Axis is READY to be enabled*)
End_If;
```

3.1.5.45 MAxisStop



 **Function Block** - Stop with the specified deceleration.

This does **not** remove the input source, but does stop the drive from continuing to move.

- This function stops any current Generator Motion.
- It causes the axis to start ignoring any changes in Pipe position to be added into the reference position.
- It decelerates, if moving, at a programmed rate.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.
- After stopping the drive, the motion must be restarted by realigning the **Actual Position** with the **Reference Position**.
- When the stop occurs, the master keeps moving and the axis starts ignoring the **Pipe Position** value and begins a controlled stop based on the input parameters.
See Axis Pipe Block for more information about Pipe Position.
 - At this point, any Axis Block level profile (issued from an FB like MAxisAbs, MAxisRel...) are aborted.
 - When the stop is complete, it is up to the application to decide how to move the axis, master, or both to a position where they can be realigned, and the master restarted.
- The "MAxisReAlign" (→ p. 379) function is used to move the axis to a restart position in order to enable synchronized machine motion to start again.
 - Once the realign function is successfully completed, the **Pipe Position** is again summed with the **Generator Position** to create the **Reference Position**.

Inputs

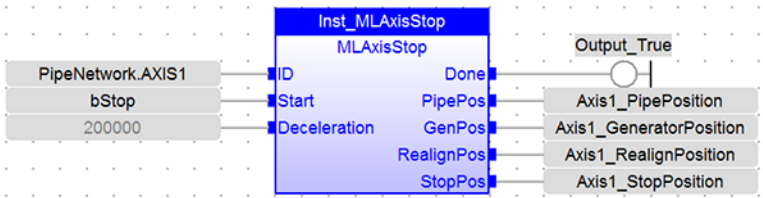
Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
Start	BOOL	FALSE, TRUE	N/A	No default	Initiates the stop action for the Axis block.
Deceleration	LREAL	No range	User units / sec ²	No default	The declaration used to bring the axis to a stop. See Set the Axis Block Position Units.

Outputs

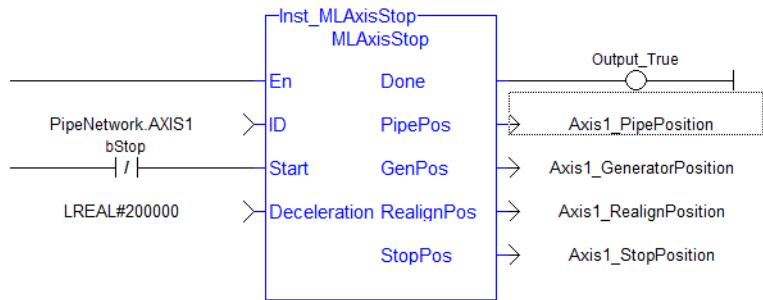
Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Becomes TRUE when the Axis is completely stopped.
GenPos	LREAL	No range	User units	• Corresponds to the Generator Position input to the axis at the time the stop is triggered. • See Axis Pipe Block about positions.

Output	Data Type	Range	Unit	Description
PipePos	LREAL	No range	User units	Corresponds to the Pipe Position input to the axis at the time the stop is triggered.
RealignPos	LREAL	No range	User units	Realign Position is the Reference Position at which the stop is triggered. - The Realign Position is: - Obtained by converting the last value sent to the drive from drive interface units into user units. - Useful to return to the point at which the trajectory was abandoned or to realign the master to the slave.
StopPos	LREAL	No range	User units	Corresponds to the last Reference Position sent to the drive at the time when the Axis is completely stopped. - It is functionally different than the Actual Position because that position is the drive position converted to user units. - The correct delta for the realign move to get in sync with the trajectory in order to realign the slave to the master is the current Reference Position minus the Stop Position for the realign move. - After stopping, if the axis is disabled and the motor position is manually altered, this distance must be considered when performing the realign.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MLAxisStop(PipeNetwork.AXIS1, bStop, 200000);  
If Inst_MLAxisStop.Done Then  
    Axis1_PipePosition      := Inst_MLAxisStop.PipePos;  
    Axis1_GeneratorPosition := Inst_MLAxisStop.GenPos;  
    Axis1_RealignPosition   := Inst_MLAxisStop.RealignPos;  
    Axis1_StopPosition      := Inst_MLAxisStop.StopPos;  
End_if;
```

See Also

"MLAxisReAlign" (→ p. 379)

3.1.5.46 MLAxisTimeStamp



 **Function** - Returns the timestamp of the triggered axis.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	Enables execution.
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
InputID	DINT	See Description.	N/A	No default	ID of the triggered Fast input of an axis. • 0 = First. • 1 = Second (e.g., IN1 and IN2 on S300).
edge	DINT	See Description.	N/A	No default	Configures the Inputs as: • 0 = Disabled. • 1 = Rising edge. • 2 = Falling edge.

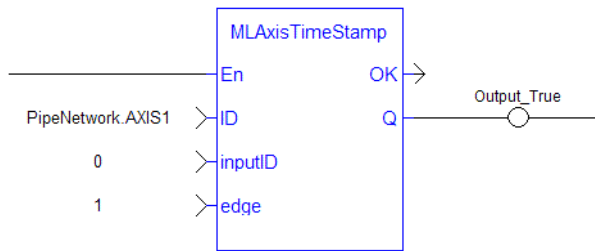
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes.
Q	DINT	No range	Microseconds	• Returns the time stamp value. • See Interpret a Timestamp.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLAxisTimeStamp(PipeNetwork.Axis1, 0, 1 ) ;
```

See Also

- "MLAxisCfgFastIn" (→ p. 323)
- "MLAxisIsTriggered" (→ p. 350)
- "MLAxisRstFastIn" (→ p. 384)

3.1.5.47 MLAxisWriteBoolParam



This function/function block was added in KAS v4.02.



Function Block - Write an Axis Boolean property of a Pipe Network Axis.

Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
ParameterNumber	INT	-32768 to +32767	N/A	No default	See "Axis Parameters" (→ p. 394).
Value	BOOL	FALSE, TRUE	N/A	No default	State to write.

Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.

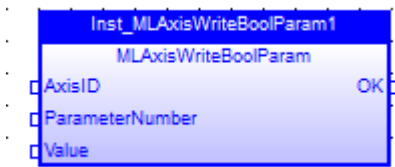
Remarks

Axis Parameters

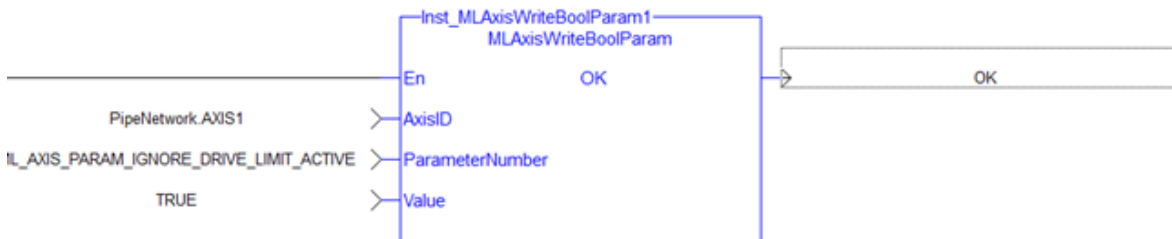
ID	Parameter	Name	R/W	Update Rate Type	Description
1038	ML_AXIS_PARAM_IGNORE_DRIVE_LIM_STATUS	Ignore Drive Limit Active	Read / Write	Static	• The default value is FALSE. • When TRUE, the Pipe Network motion engine ignores the DS-402 Status word bit #11 (Internal Limit Active). • See Overtravel Conditions.

ID	Parameter	Name	R/W	Update Rate Type	Description
1039	ML_AXIS_PARAM_DRIVE_OVERRIDE_ON_LIMIT_ACTIVE	Drive Override on Limit Active	Read / Write	Static	- Set to TRUE if the drive ignores controller demand position when DS-402 Status word bit #11 (Internal Limit Active) is on. - Set to FALSE if the Pipe Network is responsible for stopping an axis when DS-402 Status word bit #11 is on. - The default value for axes connected to AKD-family drives is TRUE. - For all other axes, the default value is FALSE.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
(*Ignore axis drive limit active bit *)
Inst_MLAxisWriteBoolParam(
PipeNetwork.AXIS1,ML_AXIS_PARAM_IGNORE_DRIVE_LIM_STATUS,ignoreDriveActiveW );
```

See Also

"MLAxisReadBoolParam" (→ p. 363)

3.1.5.48 MLAxisWriteModPos



Function - Sets the value period of the axis.

Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
ModuloPosition	LREAL	No range	User units	No default	Sets the Axis Period Value when Mode is set to Modulo.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLAxisWriteModPos (PipeNetwork.Axis1, 360) ) ;
```

3.1.5.49 MLAxisWriteParam



This function/function block was added in KAS v4.05.



Function Block - Write an axis property for a Pipe Network Axis.

Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
ParameterNumber	INT	Enumerated	N/A	No default	See "Axis Parameters" (→ p. 397).
Value	LREAL	No range	N/A	No default	State to write.

Outputs

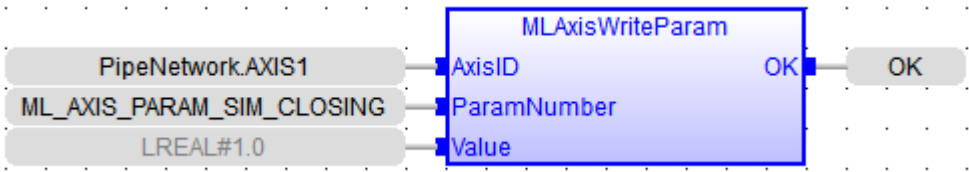
Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.

Remarks

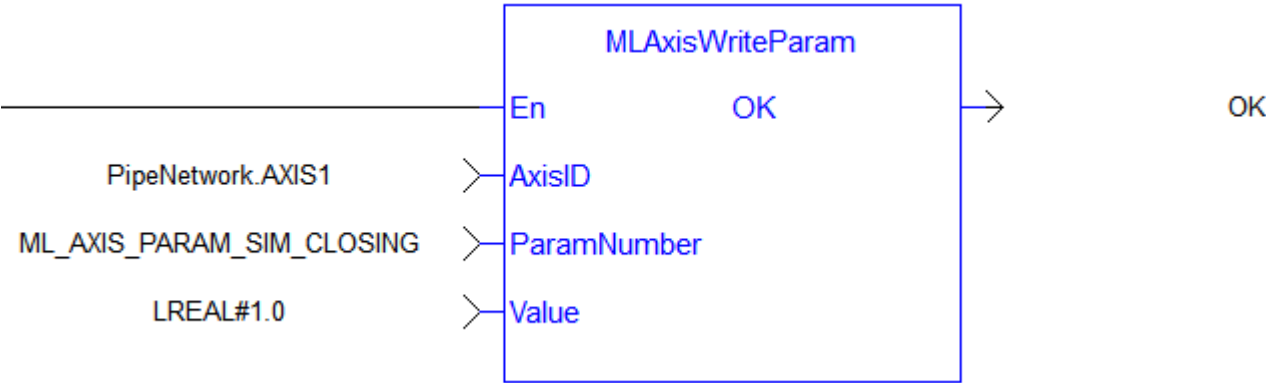
Axis Parameters

ID	Parameter	Name	R/W	Update Rate Type	Description
1036	ML_AXIS_PARAM_SIM_CLOSING	Simulator Distance Closing Fraction	Read / Write	Controller	This setting affects how responsive a simulated axis is to changing command positions. - The valid range of this parameter is (0.0, 1.5). - The default value is 0.7. Values use these ranges: (0.0, 1.0): Simulate following error. - Value 0.1 generates a large following error. - Value 0.9 generates a small following error. 1.0: No following error. (1.0, 1.5): Simulate leading error. TIP Value 1.0 is recommended when using an axis as a virtual master to command motion through camming and/or gearing.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* Set perfect following for simulated axes *)  
MAxisWriteParam(PipeNetwork.AXIS1, ML_AXIS_PARAM_SIM_CLOSING, LREAL#1.0);
```

See Also

"MAxisReadParam" (→ p. 371)

3.1.5.50 MLAxisWritePipPos

Pipe Network



Function - Forces the pipe position internal value.

- Changes the pipe position to be the new value.
- May cause a jump in motion.

ⓘ IMPORTANT

This function works only when no pipe is connected.

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
PipePosition	LREAL	No range	User units	No default	Sets the Axis Pipe Position. See Set the Axis Block Position Units.

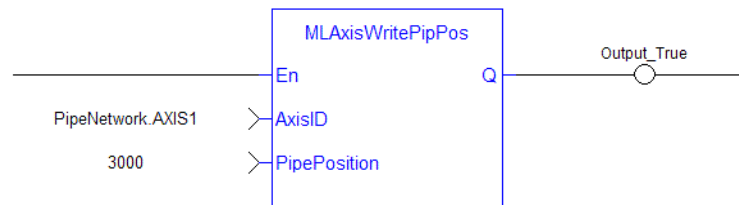
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLAxisWritePipPos(PipeNetwork.Axis1, 3000 ) ;
```

See Also

- "MLAxisCmdPos" (→ p. 325)
- "MLAxisFBackPos" (→ p. 328)
- "MLAxisGenPos" (→ p. 334)
- "MLAxisPipePos" (→ p. 354)
- "MLAxisReadActPos" (→ p. 361)

3.1.5.51 MAxisWritePos



 **Function** - Sets a new value to an axis' current location.

- If a convertor is **not** connected, zero Pipe Position and Pipe Offset.
- If a convertor is connected (pipe active also), the pipe position and offset are left alone.
 - The actual position is then set equal to the target position, and the Zero Offset is adjusted for no motion.
 - The Reference position and Generator Positions are then realigned so that the new reference position creates no step in motion.
 - The lag between reference position and actual position is absorbed in the generator position.
- Previous function name: MAxisSetZero
- After this function is called, the axis' current location has a value equal to the **Position** argument.

Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID of the Axis block.
Position	LREAL	No range	User units	No default	The new value for the axis' current location.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

Remarks

Associated Data on Positions

This data are illustrated here:

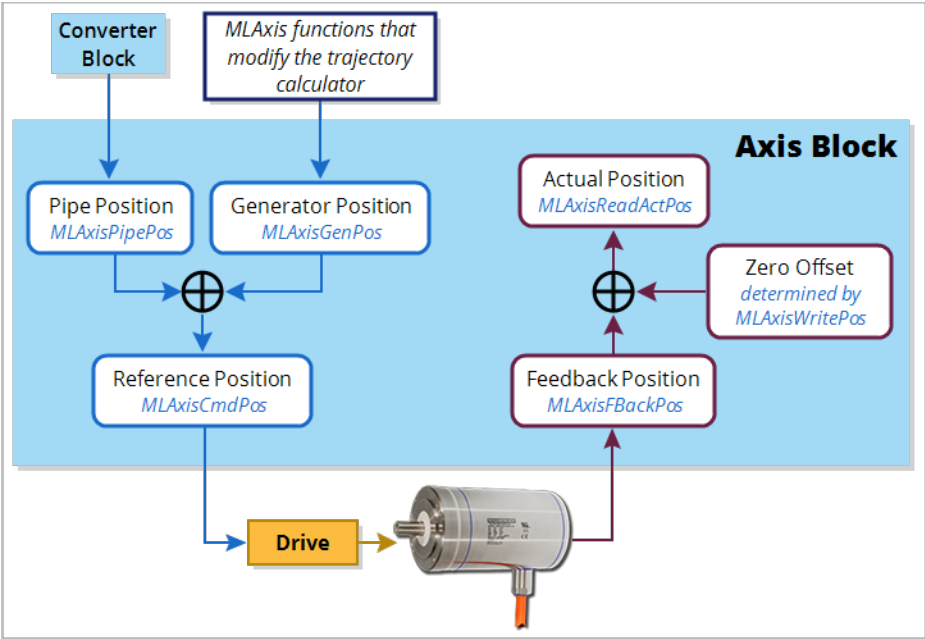


Figure 4-17: Axis Block Positions Diagram

NOTE	
All positions are in user units with modulo applied if active, unless specified.	
Position / Offset	Description
Actual Position	This is the actual position of the underlying axis as reported by the drive. - It is the sum of the feedback value (Position actual value) returned from the communication link to the drive and any zero-offset due to an MLWritePos function block. - See either "MLAxisWritePipPos" (→ p. 399) or "MLAxisWritePos" (→ p. 401). ActualPos := FeedbackPos + ZeroOffset
Feedback Position	- This is the current position the drive reports for an axis, scaled to user units. - It does not take into account the value of the Zero Offset or axis modulo.
Generator Position	This is the summation of all previous commands (i.e., calls to functions which perform motion) to the Axis internal motion generator. See either such as "MLAxisAbs" (→ p. 313), "MLAxisMoveVel" (→ p. 352), or "MLAxisRel" (→ p. 381). - It is modified by "MLAxisWritePos" (→ p. 401) to insure no jumps in the Reference Position command. - It accumulates changes in pipe position due to activate and deactivation of the pipe the Axis block is associated with.
Pipe Position	- This is the output of the convertor block is written into the Pipe Position value whenever the Convertor block is connected to the axis and the pipe is active. - See "Convertor" (→ p. 442).
Reference Position	- This is the commanded axis position sent to the drive. - It is the summation of Pipe Position and Generator Position. - ReferencePosition = Pipe Position + Generator Position
Zero Offset	This adjusts the coordinate system so the Actual Position reports correct values after homing or using "MLAxisWritePos" (→ p. 401).

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLAxisWritePos(PipeNetwork.Axis1, 0) ;
```

3.1.5.52 MAxisWriteUUnits



Function - Set the user units per revolution value of the axis.

- User units are user-defined position units used in the KAS application.
- Selected units must be as natural as possible and must make sense for the machine.
 - It must be related to the final moving object (e.g., the driven belt rather than the axis shaft).
 - The same unit must be used for all related axes for simplicity reasons.
- Speeds are defined in [User units / sec] and accelerations in [User units / sec²].

Inputs

Input	Data Type	Range	Unit	Default	Description
AxisID	DINT	1 to 1024	N/A	No default	ID of the Axis block.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •
UserUnitsPerRev	LREAL	No range	N/A	Sets the Axis User Units per revolution.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MAxisWriteUUnits(PipeNetwork.Axis1, 360 ) ;
```

3.1.5.53 MLPNAxisCreate



 **Function** - Creates a new axis object.

Inputs

Input	Data Type	Range	Unit	Default	Description
Name	STRING	No range	N/A	No default	Name of the created Axis.
DriverName	STRING	No range	N/A	No default	Is the Motion bus driver name or Simulated.
DriveAxisNumber	UINT	1 to 256	N/A	No default	- This one-based number specifies the axis on the drive. - For a single-axis drive, this number should be 1.
Address	DINT	No range	N/A	No default	Axis motion bus address.

Outputs

Output	Data Type	Range	Unit	Description
ID	DINT	0 to 1024	N/A	Returns the ID of the newly created axis object or 0 if the function failed.

Remarks

- The ID of the axis object returns 0 (zero) when the number of axes created in the application has reached the maximum permitted for the controller model.
 - In this function, the ID is set to 0 (zero) when the number of axes created in the application has already reached the maximum permitted for the controller model.
 - When this axis limit is exceeded, the controller generates error E45.
 - This error is shown on the 7-segment display and on the Diagnostics tab of the Web server.
 - See E45 for an explanation and recommended corrective actions.
 - The maximum axis count specified for each controller model applies only to physical axes.
 - This includes both PLCopen and Pipe Network axes.
 - This limit **does not** apply to virtual axis or digitizing axis.

TIP

This function should be called after "MLMotionInit" (→ p. 867) is called and before "MLMotionStart" (→ p. 872) is called.

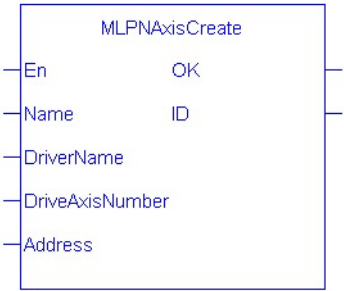
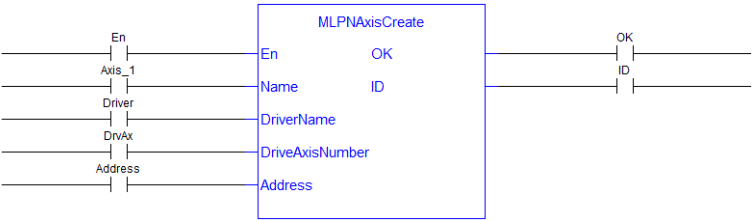


Figure 4-18: MLPNAxisCreate

FBD Language Example



FFLD Language Example



IL Language Example

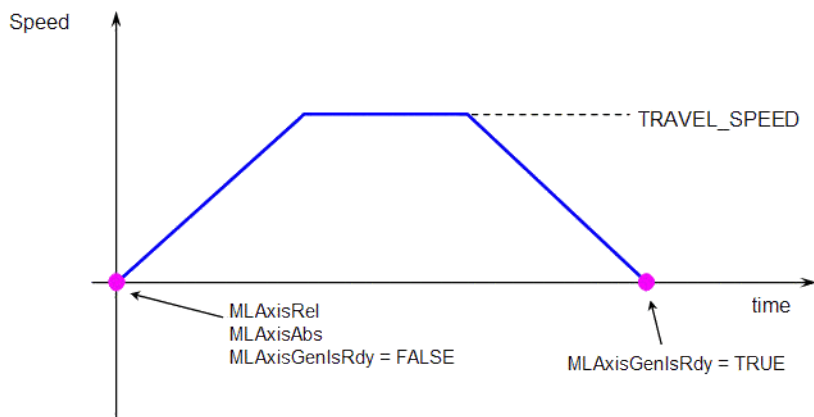
Not available.

ST Language Example

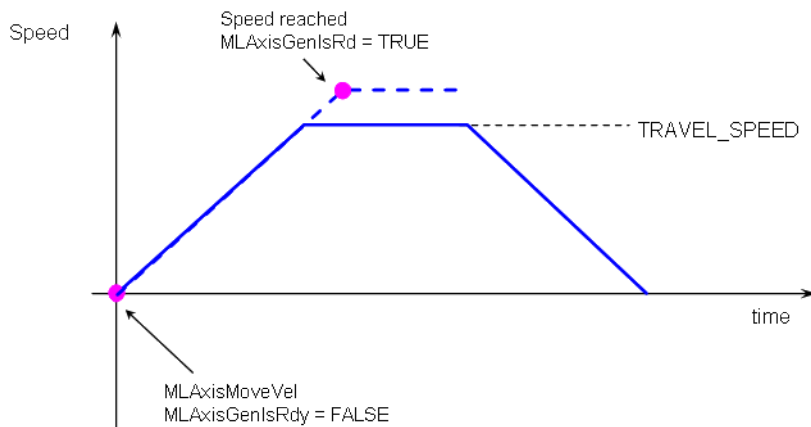
```
PipeNetwork.AXIS1 := MLPNAxisCreate('AXIS1','SercosDriver',0,1001);
```

3.1.5.54 Axis Function Examples

3.1.5.54.1 MLAxisMoveVel(Speed)



- **MLAxisMoveVel(Speed)** starts to run the axis.
- **MLAxisGenIsRdy** returns TRUE when the Speed is reached.



3.1.5.54.2 MLAxisMoveVel(0.0)

- **MLAxisMoveVel(0.0)** reduces the speed down to 0 (zero).
- **MLAxisGenIsRdy** returns TRUE once the axis is ready.

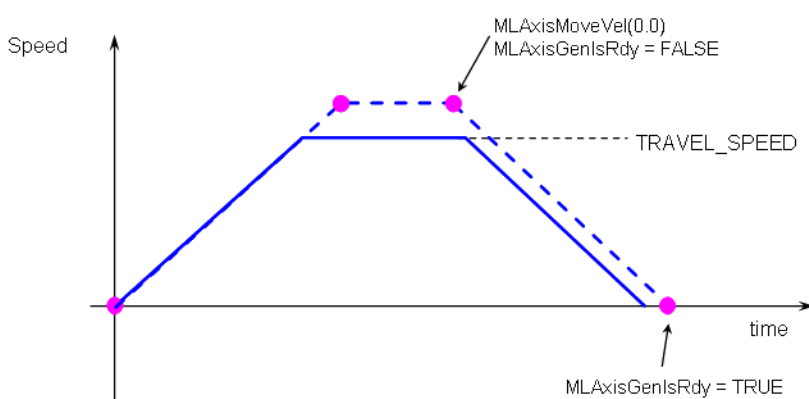


Figure 4-19: Axis Functions Usage

3.1.6 CAM Profile

Name	Description
MLCamInit	Initializes a CAM Pipe Block for use in a PLC Program.
MLCamSwitch	Switches the CAM Profile in a selected CAM object.
MLPrfReadIOffset	Returns the Input Offset value of a selected CAM Profile.
MLPrfReadIScale	Returns the Input Ratio value of a selected CAM Profile.
MLPrfReadOOffset	Returns the Output Offset value of a selected CAM Profile.
MLPrfReadOScale	Returns the Output Ratio value of a selected CAM Profile.
MLPrfWriteIOffset	Sets the Input Offset value of a selected CAM Profile.
MLPrfWriteIScale	Sets the Input Ratio value of a selected CAM Profile.
MLPrfWriteOOffset	Sets the Output Offset value of a selected CAM Profile.
MLPrfWriteOScale	Sets the Output Ratio value of a selected CAM Profile.
MLProfileBuild	Creates a CAM Profile from application data that can be executed by a CAM block in Pipe Network or PLCopen.
MLProfileCreate	Creates a new CAM Profile Object for use in a PLC Program or Pipe Network CAM block.
MLProfileInit	Initializes a previously created CAM Profile object for use in a PLC Program or Pipe Network CAM block.
MLProfileRelease	Removes a Profile so the ProfileID can be used by a different or new Profile.

3.1.6.1 Shape Specification

The shape of the cam profile must be processed by the Cam Profile Editor utility before it is usable by the Pipe Network Editor.

- The shape of the profile is represented by a table of numerical values.
 - These values can be generated using software tools such as spreadsheets or specialized cam software.
- The KAS Cam Editor software tool provides the capability to visualize, analyze, edit and smooth profiles.
- Cam blocks have gain as well as offset adjustment capabilities.
 - Axis position is usually the profile variable; however, velocity or torque profiles may also be generated.

3.1.6.1.1 Cam's INPUT-OUTPUT Transfer Function

The mathematical relationship of the cam output as a function of the input and the cam parameters is:

$$\text{If } O_{in} \leq X_i \leq O_{in} + A_{in} \text{ then} \\ Y_i = O_{out} + (fct((X_i - O_{in})/A_{in}) * A_{out})$$

Within the stated limits, these functions apply:

```

If  $X_i < O_{in}$  then  $Y_i = O_{out} + (fct(0.0) * A_{out})$ 

If  $X_i > O_{in} + A_{in}$  then  $Y_i = O_{out} + (fct(1.0) * A_{out})$ 

```

With:

Parameter	Definition
X_i	Input value.
O_i	Input offset.
A_{in}	Input amplitude.
fct	the function defining the shape.
Y_i	Output value.
O_{out}	Output offset.
A_{out}	Output amplitude.

Example: Cam Pipe Block

This image illustrates the use of the Cam blocks in a three-axes container filler mechanism.

- The cam profile for:
 - Axis 1 controls the volume of liquid dispensed and the fill rate.
 - Axis 2 raises and lowers the container.
 - Axis 3 indexes containers under the filling mechanism.
- All three axes track the main machine motion profile produced by a TMP Generator. (Figure 4-20)

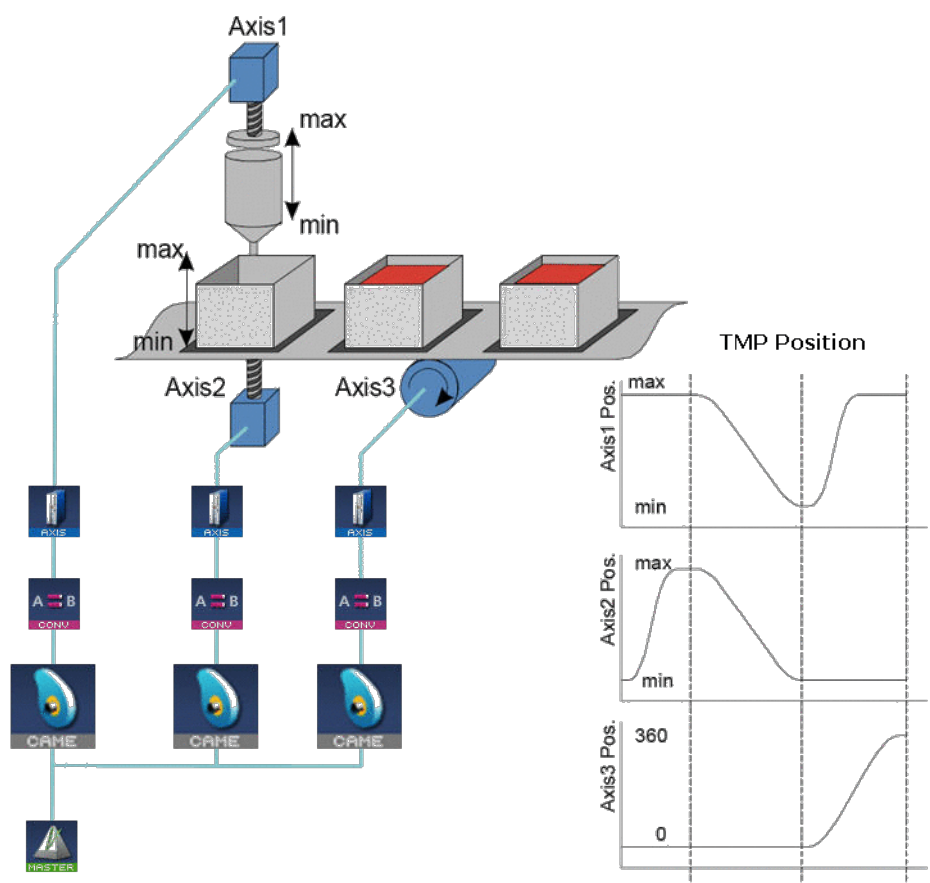


Figure 4-20: Example: Cam Blocks Control Operation of a Three Axis Filling Mechanism

3.1.6.2 MLCamInit

Pipe Network



Function - Initializes a CAM Pipe Block for use in a PLC Program.

This function is automatically called if a CAM Block is added to the Pipe Network, with user-defined settings then entered in the Pipe Blocks Properties screen.

- The CAM Pipe Block is used to generate motion profiles of any shape.
 - These profiles are created and initiated separately and the shape is modified with the CAM Editor.
 - With the Editor profiles can be changed graphically or by manually changing values in a numeric table relating input and output values with specific slopes.
 - The CAM Editor software tool provides the capability to visualize, analyze, edit, and smooth profiles.

With the Pipe Network (PN) CAM block:

- The CAM block's profile is in reference to the input positions coming into the PN CAM block (Master Absolute)
- The PN CAM block output positions are in reference to PN CAM block's output position at the end of the last CAM cycle (Slave Relative).
- Profile switching can be done on the fly, without losing synchronization and without dead time.
 - In addition, the offsets and ratios of CAM Profiles can be changed on the fly.
 - See CAM Profile Switching for more information.

NOTE

- CAM objects are normally created in the Pipe Network using the graphical engine.
- You do not have to add **MLCamInit** functions to their programs.
- Parameters are entered directly in pop-up windows; the code is automatically added to the current project.

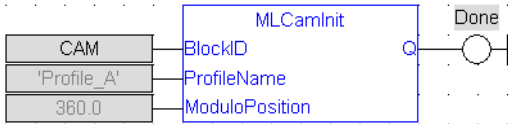
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of a created Pipe Block.
ProfileName	STRING	No range	N/A	No default	• Name of the current profile assigned to the cam. • It must be a declared profile object.
ModuloPosition	LREAL	0 to 1.0x10 ³⁰⁰	User units	No default	• Value of the period of the cam output values expressed in User units. • This is for a cyclic system. • 0 (zero) indicates the output signal does not use the modulo operation.

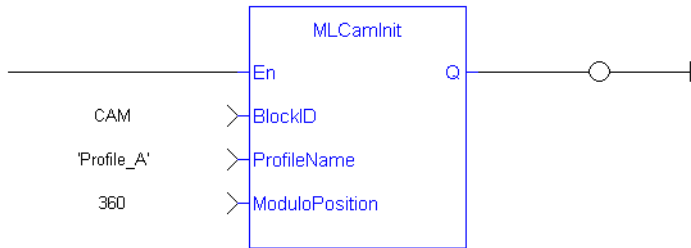
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the CAM Pipe Block is initialized. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Initialize a Pipe Network block named "CAM" with a profile named "Profile_
A", set the cam modulo position to 360
CAM := MLBlkCreate( 'CAM', 'CAM' );
MLCamInit( CAM, 'Profile_A', 360.0 );
```

See Also

- "MLProfileCreate" (→ p. 890)
- "MLProfileInit" (→ p. 892)

3.1.6.3 MLCamSwitch

Pipe Network



Function - Switches the CAM Profile in a selected CAM object.

- Can be used in combination with a comparator to verify profiles are switched at a time where the input and output values of both the old and new profiles are equal.
 - This is so an Axis receives continuous position values and does not jump.
- These profiles are created and initiated separately and the shape is created with the CAM Editor.
 - With the Editor, profiles can be changed graphically or by manually changing values in a numeric table relating input and output values with specific slopes.
 - The CAM Editor is used to visualize, analyze, edit, and smooth profiles.
- See CAM Profile Switching.

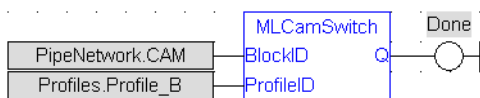
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initialized CAM Pipe Block.
ProfileID	DINT	-2147483648 to 2147483647	N/A	No default	• Name of the new CAM Profile which is assigned to the CAM Pipe Block. • It must be a declared profile object.

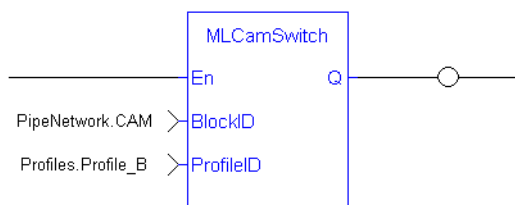
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the CAM Profile is changed. •

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Switch CAM Profile  
MLCamSwitch(PipeNetwork.CAM, Profiles.Profile_B);
```

See Also

- "MLPrfWriteOffset" (→ p. 423)
- "MLPrfWriteOScale" (→ p. 429)
- "MLProfileCreate" (→ p. 890)
- "MLProfileInit" (→ p. 892)

3.1.6.4 MLPrfReadIOffset



Function - Returns the Input Offset value of a selected CAM Profile.

- Offsets can be changed on the fly to modify the CAM Profile while maintaining its shape.
- A change in input offset is equivalent to shifting the CAM Profile on the X or Input Axis.

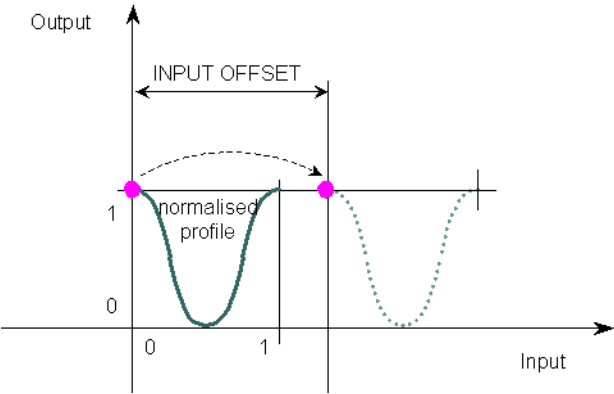


Figure 4-21: MLPrfReadIOffset

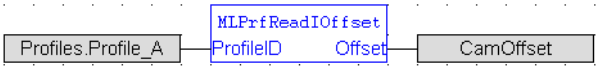
Inputs

Input	Data Type	Range	Unit	Default	Description
ProfileID	DINT	1 to 1024	N/A	No default	ID number of a created CAM Profile.

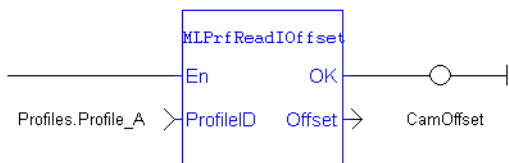
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	No range	N/A	• Returns TRUE when the function successfully executes. •
Offset	LREAL	No range	N/A	Returns the Input Offset of the selected CAM Profile.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Save value of input offset  
CamOffset := MLPrfReadIOffset( Profiles.Profile_A );
```

See Also

- "MLPrfWriteIOffset" (→ p. 423)
- "MLProfileCreate" (→ p. 890)
- "MLProfileInit" (→ p. 892)

3.1.6.5 MLPrfReadIScale



 **Function** - Returns the Input Ratio value of a selected CAM Profile.

- Previous function name: MLPrfGetIRatio
- Ratios can be changed on the fly to modify the CAM Profile while maintaining its basic shape.
- A change in input ratio is equivalent to stretching the CAM Profile on the X (or Input) Axis.
- A negative value is not allowed.

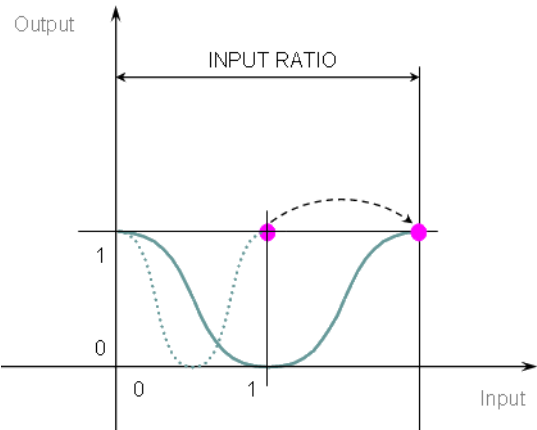


Figure 4-22: MLPrfReadIScale

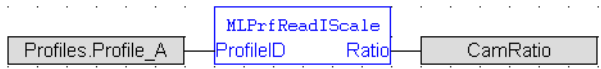
Inputs

Input	Data Type	Range	Unit	Default	Description
ProfileID	DINT	1 to 1024	N/A	No default	ID number of an initialized CAM Profile.

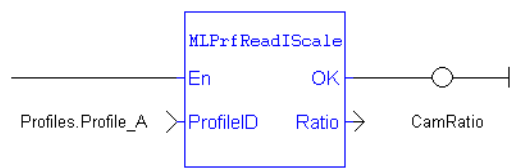
Outputs

Output	Data Type	Range	Unit	Description
Ratio	LREAL	≥ 0 (zero)	N/A	Returns the Input Ratio value of a selected CAM Profile.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Save value of input ratio  
CamRatio := MLPrfReadIScale( Profiles.Profile_A );
```

See Also

- "MLPrfWriteIScale" (→ p. 425)
- "MLProfileCreate" (→ p. 890)
- "MLProfileInit" (→ p. 892)

3.1.6.6 MLPrfReadOOffset

Pipe Network ✓



Function - Returns the Output Offset value of a selected CAM Profile.

- Offsets can be changed on the fly to modify the CAM Profile while maintaining its shape.
- A change in output offset is equivalent to shifting the CAM Profile on the Y (or Output) Axis.

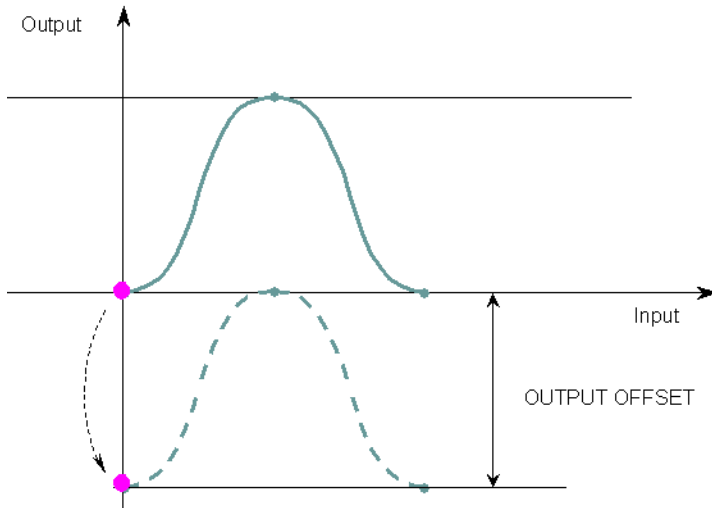


Figure 4-23: MLPrfReadOOffset

Inputs

Input	Data Type	Range	Unit	Default	Description
ProfileID	DINT	1 to 1024	N/A	No default	ID number of an initialized CAM Profile.

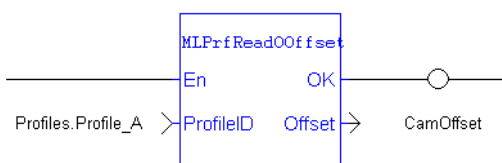
Outputs

Output	Data Type	Range	Unit	Description
Offset	LREAL	No range	User units	Returns the Output Offset of the selected CAM Profile.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Save value of output offset  
CamOffset := MLPrfReadOOffset( Profiles.Profile_A );
```

See Also

- "MLPrfWriteOOffset" (→ p. 427)
- "MLProfileCreate" (→ p. 890)
- "MLProfileInit" (→ p. 892)

3.1.6.7 MLPrfReadOScale



Function - Returns the Output Ratio value of a selected CAM Profile.

- Previous function name: MLPrfGetORatio
- Ratios can be changed on the fly to modify the CAM Profile while maintaining its basic shape.
- A change in output ratio is equivalent to stretching, and flipping if negative, the CAM Profile on the Y (or Output) Axis.

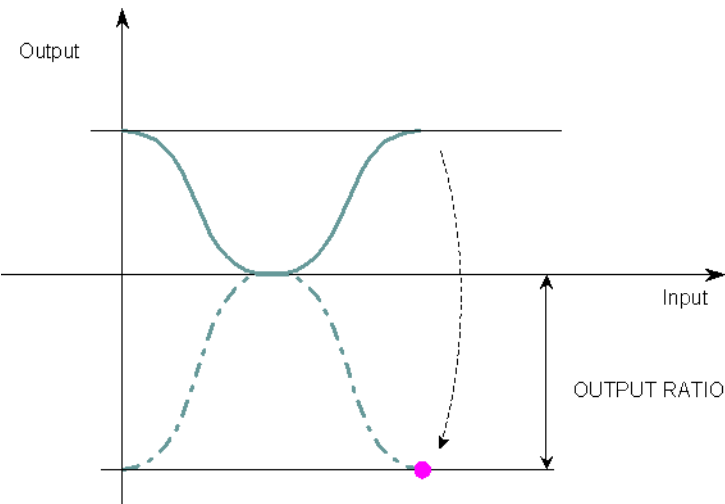


Figure 4-24: MLPrfReadOScale

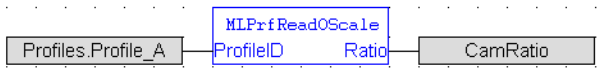
Inputs

Input	Data Type	Range	Unit	Default	Description
ProfileID	DINT	1 to 1024	N/A	No default	ID number of an initialized CAM Profile.

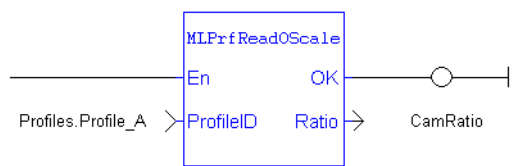
Outputs

Output	Data Type	Range	Unit	Description
Ratio	LREAL	No range	N/A	Returns the Output Ratio value of a selected CAM Profile.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Save value of output ratio  
CamRatio := MLPrfReadOScale( Profiles.Profile_A );
```

See Also

- "MLPrfWriteOScale" (→ p. 429)
- "MLProfileCreate" (→ p. 890)
- "MLProfileInit" (→ p. 892)

3.1.6.8 MLPrfWriteOffset

Pipe Network



Function - Sets the Input Offset value of a selected CAM Profile.

- Offsets can be changed on the fly to modify the CAM Profile while maintaining its shape.
- A change in input offset is equivalent to shifting the CAM Profile on the X (or Input) Axis.

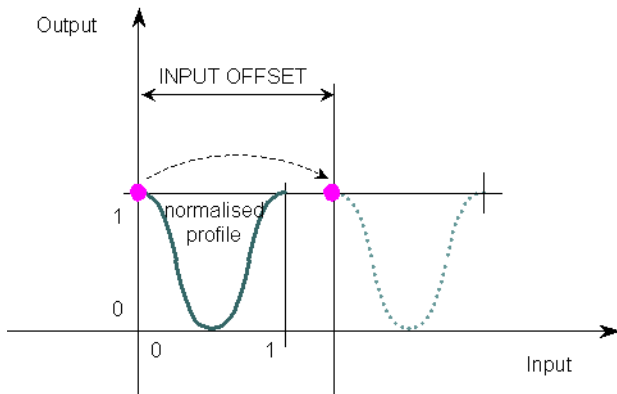


Figure 4-25: MLPrfWriteOffset

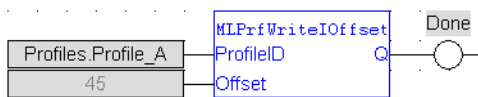
Inputs

Input	Data Type	Range	Unit	Default	Description
ProfileID	DINT	1 to 1024	N/A	No default	ID number of an initialized CAM Profile.
Offset	LREAL	No range	N/A	No default	Desired new value of Input Offset.

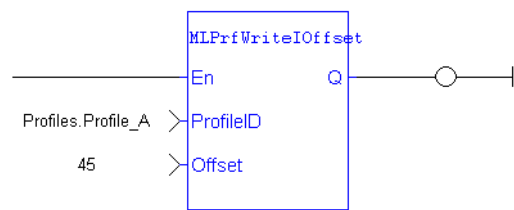
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Input Offset is changed to the new value. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Change the value of input offset
MLPrfWriteIOffset( Profiles.Profile_A , 45 );
```

See Also

- "MLPrfReadIOffset" (→ p. 415)
- "MLProfileCreate" (→ p. 890)
- "MLProfileInit" (→ p. 892)

3.1.6.9 MLPrfWriteIScale



Function - Sets the Input Ratio value of a selected CAM Profile.

- Previous function name: MLPrfSetIRatio
- Ratios can be changed on the fly to modify the CAM Profile while maintaining its basic shape.
- A change in input ratio is equivalent to stretching the CAM Profile on the X (or Input) Axis.

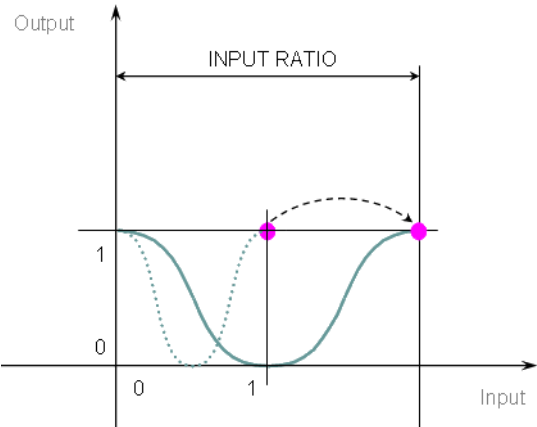


Figure 4-26: MLPrfWriteIScale

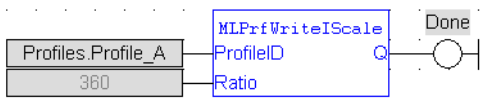
Inputs

Input	Data Type	Range	Unit	Default	Description
ProfileID	DINT	1 to 1024	N/A	No default	ID number of an initialized CAM Profile.
Ratio	LREAL	Positive	N/A	No default	Desired new value of Input Ratio.

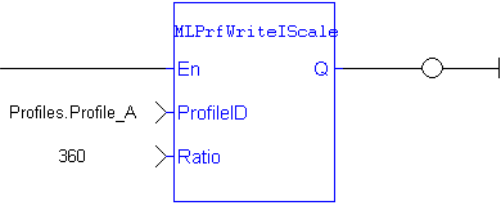
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Input Ratio is changed to the new value. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Change value of input ratio
MLPrfWriteIScale( Profiles.Profile_A, 360 );
```

See Also

- "MLPrfReadIScale" (→ p. 417)
- "MLProfileCreate" (→ p. 890)
- "MLProfileInit" (→ p. 892)

3.1.6.10 MLPrfWriteOOffset

Pipe Network



Function - Sets the Output Offset value of a selected CAM Profile.

- Offsets can be changed on the fly to modify the CAM Profile while maintaining its shape.
- A change in output offset is equivalent to shifting the CAM Profile on the Y (or Output) Axis.

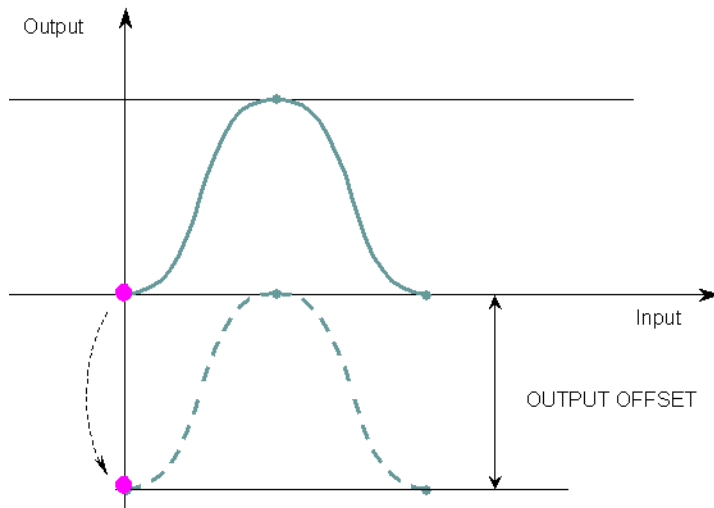


Figure 4-27: MLPrfWriteOOffset

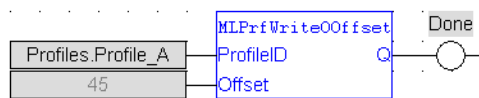
Inputs

Input	Data Type	Range	Unit	Default	Description
ProfileID	DINT	1 to 1024	N/A	No default	ID number of an initialized CAM Profile.
Offset	LREAL	No range	N/A	No default	Desired new value of Output Offset.

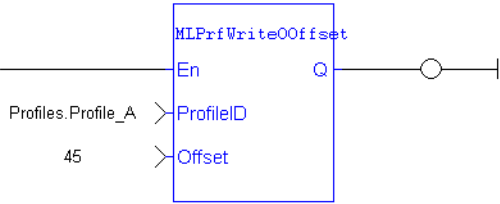
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Output Offset is changed to the new value. •

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Change value of output offset
MLPrfWriteOOffset( Profiles.Profile_A , 45 );
```

See Also

- "MLPrfReadOOffset" (→ p. 419)
- "MLProfileCreate" (→ p. 890)
- "MLProfileInit" (→ p. 892)

3.1.6.11 MLPrfWriteOScale

Pipe Network ✓



Function - Sets the Output Ratio value of a selected CAM Profile.

- Previous function name: MLPrfSetORatio
- Ratios can be changed on the fly to modify the CAM Profile while maintaining its basic shape.
- A change in output ratio is equivalent to stretching (flipping if negative - see "MLPrfWriteOScale" (→ p. 429)) the CAM Profile on the Y (or Output) Axis.

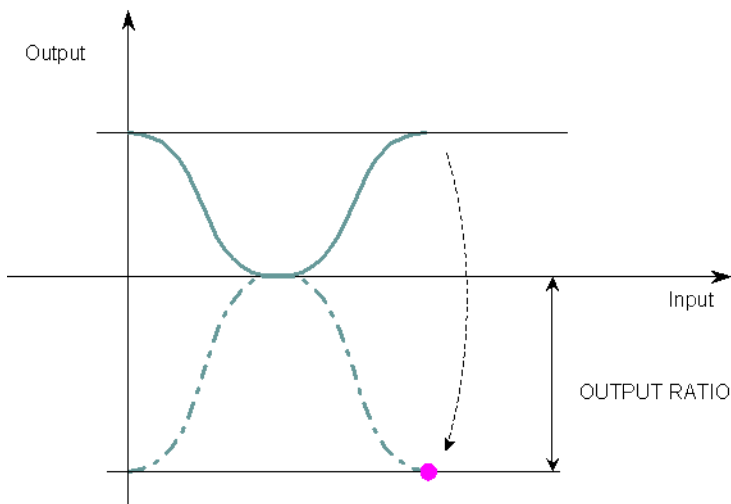


Figure 4-28: MLPrfWriteOScale

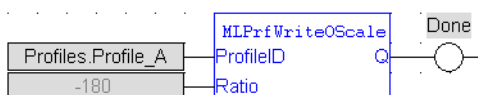
Inputs

Input	Data Type	Range	Unit	Default	Description
ProfileID	DINT	1 to 1024	N/A	No default	ID number of an initialized CAM Profile.
Ratio	LREAL	No range	N/A	No default	Desired new value of Output Ratio.

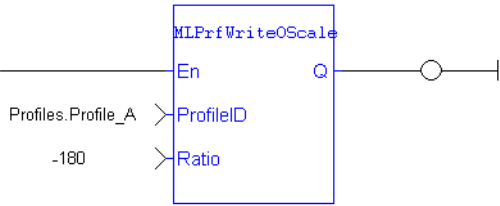
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Output Ratio is changed to the new value. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Change value of output ratio
MLPrfWriteOScale( Profiles.Profile_A , -180 );
```

See Also

- "MLPrfReadOScale" (→ p. 421)
- "MLProfileCreate" (→ p. 890)
- "MLProfileInit" (→ p. 892)

3.1.7 Comparator

Name	Description
MLCompCheck	Verifies if the reference of a comparator Pipe Block has been crossed.
MLCompInit	Initializes a comparator Pipe Block with user-defined settings for use in a PLC Program.
MLCompReadRef	Returns the reference position of a comparator block.
MLCompReset	Clears the Transition Flag of a comparator Pipe Block.
MLCompWriteRef	Sets the reference position of a comparator block.

3.1.7.1 Examples: Comparator Functions

When you call the "MLCompWriteRef" (→ p. 440) function (Figure 4-29), the output for "MLCompCheck" (→ p. 433) becomes TRUE (Figure 4-30) as soon as the input value reaches the reference.

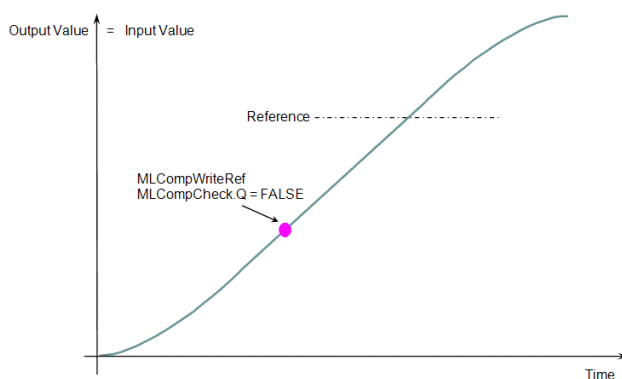


Figure 4-29: MLCompCheck = FALSE

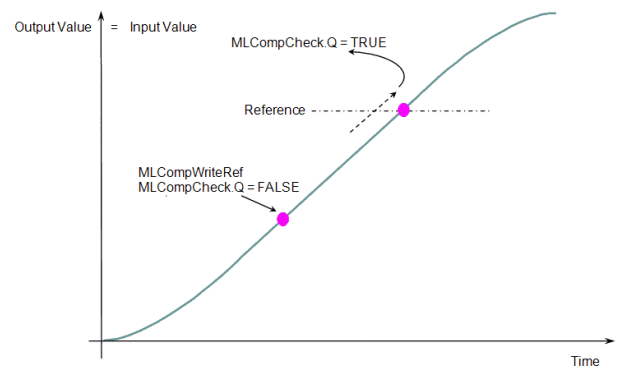


Figure 4-30: MLCompCheck = TRUE

The same function can also be called for a cyclic input value. (Figure 4-31 and Figure 4-32)

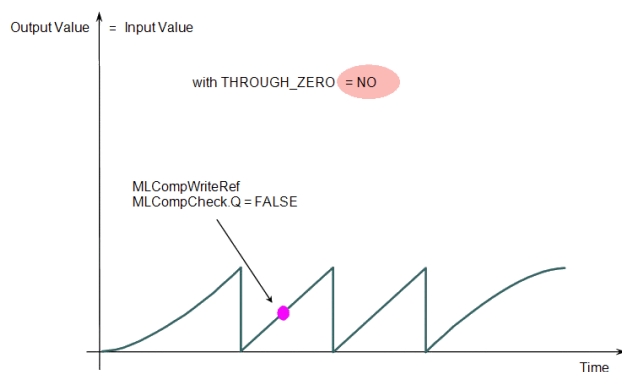


Figure 4-31: MLCompCheck = FALSE

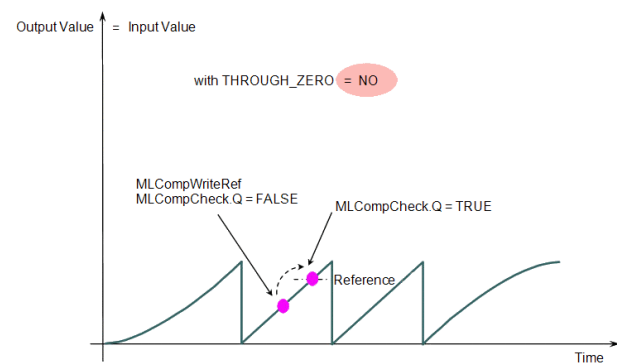


Figure 4-32: MLCompCheck = TRUE

When the THROUGH_ZERO parameter is set to YES (Figure 4-33), the output for "MLCompCheck" (→ p. 433) becomes TRUE (Figure 4-34) as soon as the input value reaches the reference, but not before it has passed through zero.

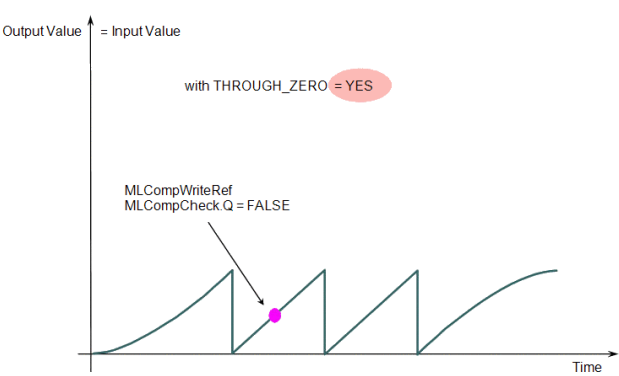


Figure 4-33: MLCompCheck = FALSE

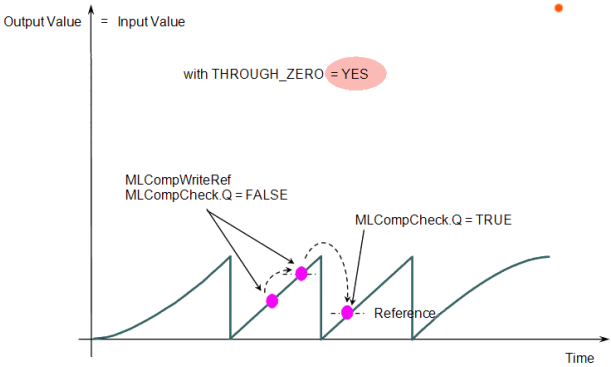


Figure 4-34: MLCompCheck = TRUE

3.1.7.2 MLCompCheck



Function - Verifies if the reference of a comparator Pipe Block has been crossed.

- Returns the Transition Flag of a comparator object, which turns TRUE if the input position to the comparator is greater or equal to the reference.
 - The Comparator Transition Flag stays TRUE until it is reset.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

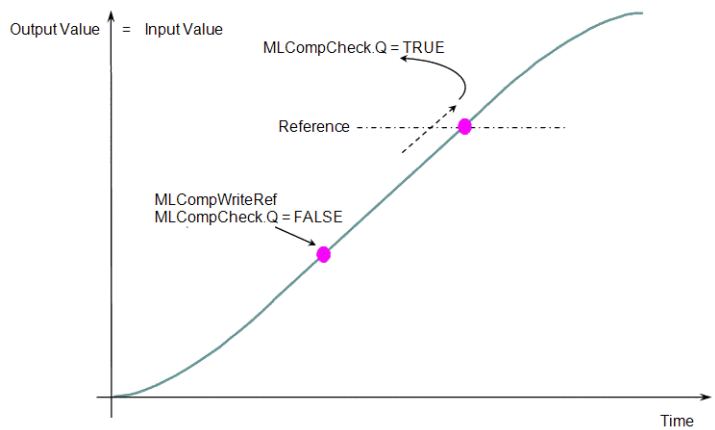


Figure 4-35: MLCompCheck

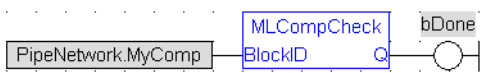
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Comparator object.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if reference position of the Comparator object has been crossed. •

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Check if Comparator Reference has been reached
bCrossed := MLCompCheck( PipeNetwork.MyComp );
```

See Also

- "MLCompReadRef" (→ p. 438)
- "MLCompReset" (→ p. 439)
- "MLCompWriteRef" (→ p. 440)

3.1.7.3 MLComplnit

Pipe Network



Function - Initializes a comparator Pipe Block with user-defined settings for use in a PLC Program.

- Function block is automatically called if a Comparator Block is added to the Pipe Network.
 - User-defined settings are entered in the Pipe Blocks Properties screen.
- The Transition Flag of a comparator object turns TRUE if the input position to the comparator is greater or equal to the reference.
- The Comparator Transition Flag stays TRUE until it is reset.
- If the input **ThroughZero** is set to TRUE, the system must cross zero and then the reference position before the Transition Flag is set.
- If **ThroughZero** is FALSE, the Transition Flag is set immediately if the input pipe position is greater or equal to the Reference value.

NOTE

- Comparator objects are normally created in the Pipe Network using the graphical engine.
- You do not have to add **MLComplnit** functions to their programs.
- Parameters are entered directly in pop-up windows; the code is automatically added to the current project.

Inputs

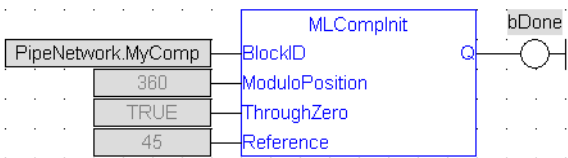
Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of a created Comparator Pipe Block.
ModuloPosition	LREAL	0 to 1.0x10 ³⁰⁰	User units	No default	• Value of the period of a cyclic system expressed in User units. • The parameter MODULO_POSITION is defined to correctly manage the periodicity (modulo) of the input values. • 0 (zero) indicates the input signal does not use the modulo operation.

Input	Data Type	Range	Unit	Default	Description
ThroughZero	BOOL	FALSE, TRUE	N/A	No default	Through zero reference mode can be set or not: - TRUE: Used to properly detect a periodic threshold crossing of motions on periodic axis where the flow values are always greater than or equal to zero but lower than the Modulo Position. - In this mode, the flow values must first cross one period limit and then, as soon as a value is greater than or equal to the reference, the ready flag becomes true. - FALSE: Applies mainly to bounded motions. - The Comparator's ready flag is false as long as the flow value is less than the reference and becomes true as soon as the flow value is greater than or equal to the reference.
Reference	LREAL	No range	User units	No default	- The reference position in the new Comparator object. - The Comparator checks if the input value of the Comparator is greater or equal to this reference value.

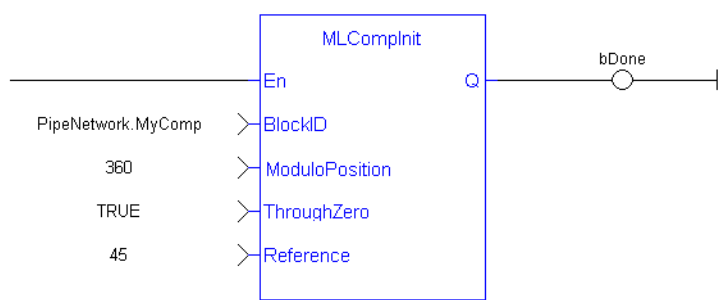
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE when function starts to execute.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Initiate a created Comparator Block named "MyComp" to:  
//  Modulo of 360  
//  Require the input position to first cross 0 before the  
//    MLCompCheck output is triggered  
//  Input compared position to 45  
  
MyComp := MLBlkCreate( 'MyComp', 'COMPARATOR' );  
MLCompInit( MyComp, 360.0, TRUE, 45.0 );
```

See Also

- "MLBlkCreate" (→ p. 292)
- "MLCompCheck" (→ p. 433)
- "MLCompReset" (→ p. 439)
- "MLCompWriteRef" (→ p. 440)

3.1.7.4 MLCompReadRef



Function - Returns the reference position of a comparator block.

- The Transition Flag of a comparator object turns TRUE if the input position to the comparator is greater or equal to the reference.
- The Comparator Transition Flag stays TRUE until it is reset.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Comparator object.

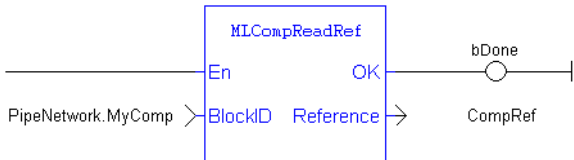
Outputs

Output	Data Type	Range	Unit	Description
Reference	LREAL	No range	User units	Returns the current reference position of the Comparator object.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Return the Comparator Reference value
CompRef := MLCompReadRef( PipeNetwork.MyComp );
```

See Also

- "MLCompCheck" (→ p. 433)
- "MLCompReset" (→ p. 439)
- "MLCompWriteRef" (→ p. 440)

3.1.7.5 MLCompReset

Pipe Network



Function - Clears the Transition Flag of a comparator Pipe Block.

- The Transition Flag of a comparator object turns TRUE if the input position to the comparator is greater or equal to the reference.
- The Comparator Transition Flag stays TRUE until it is reset.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Comparator object.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when function starts to execute. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Clear the Transition Flag of a Comparator object
MLCompReset( PipeNetwork.MyComp );
```

See Also

- "MLCompCheck" (→ p. 433)
- "MLCompReadRef" (→ p. 438)
- "MLCompWriteRef" (→ p. 440)

3.1.7.6 MLCompWriteRef



Function - Sets the reference position of a comparator block.

- The Transition Flag of a comparator object turns TRUE if the input position to the comparator is greater or equal to the reference.
- The Comparator Transition Flag stays TRUE until it is reset.
- If the input ThroughZero is set to TRUE, system must cross zero and then the reference position before the Transition Flag is set.
- If ThroughZero is FALSE, Transition Flag is set immediately if the input pipe position is greater then or equal to the Reference value.

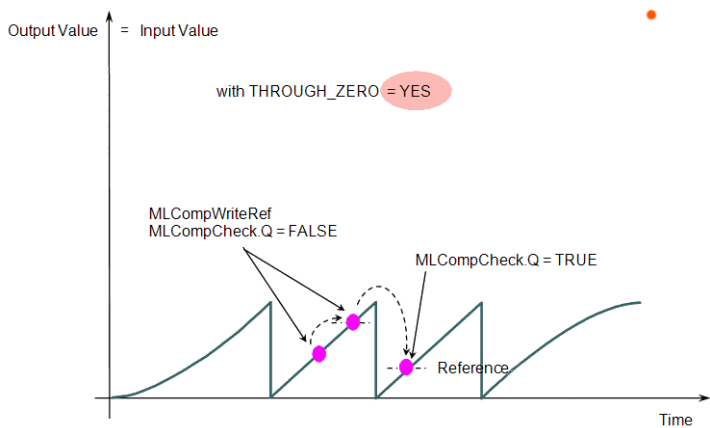


Figure 4-36: MLCompWriteRef

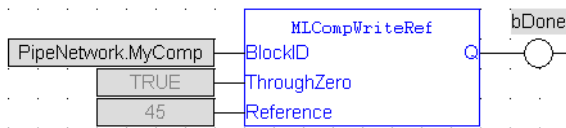
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Comparator object.
ThroughZero	BOOL	FALSE, TRUE	N/A	No default	• When TRUE, the system must cross zero and then the reference position before the Transition Flag is set. • When FALSE, the Transition Flag is set immediately if the input pipe position is greater then or equal to the Reference value.
Reference	LREAL	No range	User units	No default	New reference position to set in the selected Comparator object.

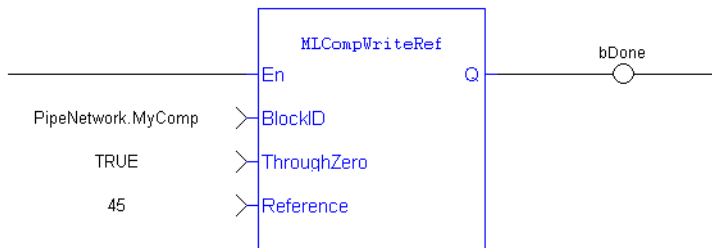
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when function starts to execute. •

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Set the Comparator Reference value
MLCompWriteRef( PipeNetwork.MyComp , TRUE , 45 );
```

See Also

- "MLCompCheck" (→ p. 433)
- "MLCompReadRef" (→ p. 438)
- "MLCompReset" (→ p. 439)

3.1.8 Convertor

Name	Description
MLCNVConECAT	Connects the output of a pipe convertor block to an EtherCAT Output (Rx) PDO object.
MLCNVConnect	Connects a converter Pipe Block to the specified axis.
MLCNVConnectEx	Connects the output of a pipe to an axis data other than the control position. Connects an extra converter Pipe Block to the specified axis.
MLCNVDisconnect	Disconnect a converter Pipe Block from its associated axis.
MLCNVInit	Initializes a converter Pipe Block.

3.1.8.1 MLCNVConECAT



Function - Connects the output of a pipe convertor block to an EtherCAT Output (Rx) PDO object.

- The output value of the convertor block is written to the PDO object every update of the convertor block.
- The pipe block is specified by the BlockID input.
- The PDO object is specified by the DeviceAddr, Index, and SubIndex inputs.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	The convertor block whose output value is written to the PDO object. Example: PipeNetwork:CNV1.
DeviceAddr	INT	No range	N/A	No default	The device address of the PDO object to be written. • EtherCAT devices are numbered in order. • The first device is 1001, the second 1002, etc.
Index	UINT	No range	N/A	No default	The index of the PDO object to be written. • The index can be determined from the table located in the PDO Selection/Mapping tab of the EtherCAT device page. • In Project Explorer, under EtherCAT, select the device, then select the PDO Selection/Mapping tab.
SubIndex	USINT	No range	N/A	No default	The sub-index of the PDO object to be written. • The sub-index can be determined from the table located in the PDO Selection/Mapping tab of the EtherCAT device page. • In Project Explorer, under EtherCAT, select the device, then select the PDO Selection/Mapping tab.

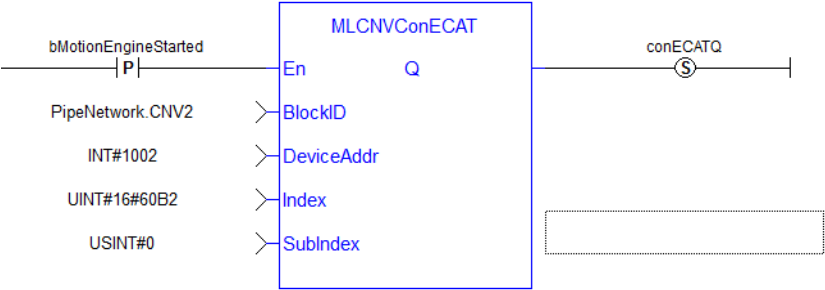
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if this function has successfully connected the output of the pipe convertor block to the EtherCAT Output (Rx) PDO Object. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Connect a converter Pipe Block named "CNV2" to PDO 16#60B2 (Accel FF) on
ECAT address 1002.
MLCNVConECAT( PipeNetwork.CNV2, 1002, 16#60B2, 0 );
```

See Also

- "MLCNVDisconnect" (→ p. 450)
- "MLCNVInit" (→ p. 452)

3.1.8.2 MLCNVConnect

 **Function** - Connects a converter Pipe Block to the specified axis.

- Initializes the pipe position to the Converter block output value.
- Adjusts the axis Pipe Offset so that no jump in motion is generated.
- When using the Pipe Network for coordinated motion, Pipe Blocks have to be Activated, Connected, and then Powered On before move commands work.
- The Converter block changes the incoming flow of values to continuous position output with no periodicity.
 - If a converter block is not connected to an Axis, it does not send position output values to its assigned Axis.
 - Every pipe branch must end in a converter, whether or not it is connected to a destination Axis object.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

NOTE

- All converters in the Pipe Network can be connected at once with the command Pipe Network(MLPN_Connect).
- This calls automatically generated code with **MLCNVConnect** commands for each Converter block.
- In a multi-axis program only one command can be used to connect Pipe Blocks instead of writing code for each Axis separately.

TIP

The converter block can control the analog output on the AKD.

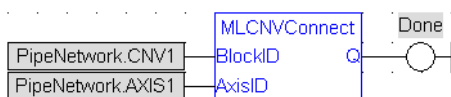
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Converter object.
AxisID	DINT	1 to 256	N/A	No default	ID number of an initiated Axis object.

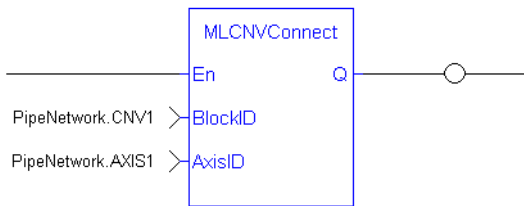
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the converter is connected to the Axis object. •

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Connect a converter Pipe Block named "CNV1" to Pipe Block AXIS1.
MLCNVConnect( PipeNetwork.CNV1, AXIS1 );
```

See Also

- "MLCNVConnectEx" (→ p. 447)
- "MLCNVDisconnect" (→ p. 450)
- "MLCNVInit" (→ p. 452)

3.1.8.3 MLCNVConnectEx



Function - Connects the output of a pipe to an axis data other than the control position.

- This function:
 - Connects an extra converter Pipe Block to the specified axis.
 - Connects several converter Pipe Blocks to the same axis and acts on different data.
- Normally a Converter block sends position values to an Axis.
 - However, some cases exist that require additional information (e.g., torque feed-forward (IDN 3056)) that needs to be provided by a second converter.
- The PDO values are overwritten by Mapped PLC variables including either:
 - A possible link to the mapping of variables.
 - The section on **MLParamWrite()** warning indicating that the function block write of Analog output is overwritten by the **MLCnvConnectEx** function.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

NOTE

- This FB does NOT work if you have Simulated the device.
- In this instance, the FB continuously generates error messages in the Controller log window.

❗IMPORTANT

Add 16#8000 to the designated IDN number for ValueID input. 8000 in hexadecimal signals a vendor-specific IDN value.

3.1.8.3.1.1 Precedence Rules

1. A PLC variable mapped to Analog Output takes precedence.
2. If **MLCNVConnect** assigns a Pipe output to an Analog Output, it takes precedence over a **DriveParamWrite** function call.
3. **DriveParamWrite** modifies the Analog Output but is overwritten by the higher precedent options if they are present.

Inputs

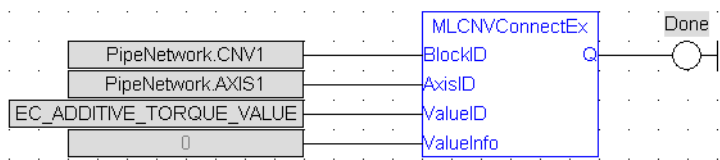
Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Converter object.
AxisID	DINT	1 to 256	N/A	No default	ID number of an initiated Axis object.

Input	Data Type	Range	Unit	Default	Description
ValueID	DINT	Enumerated	N/A	No default	Specify this constant: - EC_ADDITIVE_TORQUE_VALUE - For torque feed-forward. - EC_ANALOG_OUTPUT - For control of Analog Output: AKD parameter: AOUT.VALUEU. - See Example: Pre-defined Constants for more information. - If the Analog Output is mapped to a PLC variable, the connection to the analog output by EC_ANALOG_OUTPUT does not work because the output value is overwritten by the PLC mapped variable data. - To function properly, the AOUT.MODE must be set to User Mode (mode = 0).
ValueInfo	DINT	-2147483648 to 2147483647	N/A	No default	- This value is ignored. - It must be set to 0 (zero).

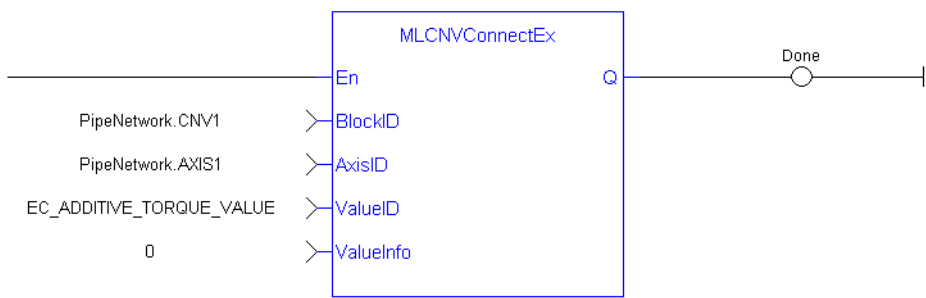
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE if the converter is connected to the Axis object.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Connect a converter block named "CNV1" to the AXIS1 block.  
//Send feed-forward (EC_ADDITIVE_TORQUE_VALUE) to the drive.  
MLCNVConnectEx( PipeNetwork.CNV1, PipeNetwork.AXIS1, EC_ADDITIVE_TORQUE_  
VALUE, 0 );
```

See Also

- "MLCNVConnect" (→ p. 445)
- "MLCNVDisconnect" (→ p. 450)
- "MLCNVInit" (→ p. 452)

3.1.8.4 MLCNVDisconnect

 **Function** - Disconnect a converter Pipe Block from its associated axis.

- Stops sending the convertor output to the Pipe Position.
- Disconnects the convertor from the axis.
- If a converter block is not connected to an Axis, it does not send position output values to its assigned Axis.
 - Can disconnect one or multiple Axis from the Pipe Network and still send single-axis motion commands.
- Axis can be disconnected while the Pipe Positions are reset to different values or if coordinated motion is only not needed with every axis in the project in a certain state.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Converter object.

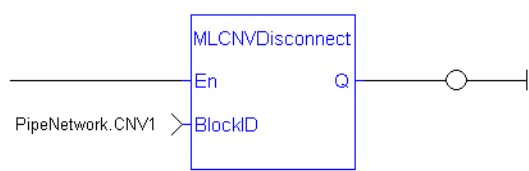
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the converter is disconnected from the Axis object. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Disconnect a converter block name "CNV1".  
MLCNVDisconnect( PipeNetwork.CNV1);
```

See Also

- "MLCNVConnect" (→ p. 445)
- "MLCNVInit" (→ p. 452)

3.1.8.5 MLCNVInit



Function - Initializes a converter Pipe Block.

- **MLCNVInit** is automatically called if a Converter Block is added to the Pipe Network, with the input mode (position or speed) entered in the Pipe Blocks Properties screen.
- The Converter block changes the incoming flow of speed or position values to continuous position output with no periodicity.

NOTE

- Converter objects are normally created in the Pipe Network using the graphical engine.
- You do not have to add MLCNVInit function blocks to their programs.
- Parameters are entered directly in pop-up windows; the code is automatically added to the current project.

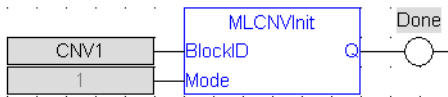
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of a created Pipe Block.
Mode	DINT	1 to 2	N/A	No default	Determines the type of input to the Converter Object. The available modes are: Position • 1 is for Position mode. • The values drive the position of the motor. • At pipe activation, the current (axis) position is set to the first value given by the pipe by moving the motor. • Speed and acceleration are derivatives of position. • The torque is set according to the regulator needs. • Units are the axis physical units. Speed • 2 is for Speed mode. • The values drive the speed of the motor. • At pipe activation, the current position is not affected. • Position is the integral of speed and acceleration is the derivative of speed. • The torque is set according to the regulator needs. • Units are the axis physical units per second.

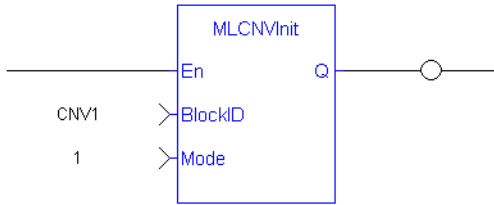
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Converter Pipe Block is initialized. •

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
// Initiate a created convertor block named "CNV1".
CNV1 := MLBlkCreate( 'CNV1', 'CONVERTOR' );
MLCNVInit( CNV1, 1 );
```

See Also

- "MLBlkCreate" (→ p. 292)
- "MLCNVConnect" (→ p. 445)

3.1.9 Delay

Name	Description
MLDelayInit	Initializes a delay object.

3.1.9.1 MLDelayInit

Pipe Network ✓



Function - Initializes a delay object.

- This is automatically created in the compiled code of a Pipe Network.
 - It is included in the MLPN_CREATE_OBJECT (created in ST) which is typically executed in a project as part of the startup sequence of the Pipe Network.
- Returns TRUE if the function succeeded.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of a created Pipe Block.
CycleDelay	DINT	0 to 9	Cycle	No default	Number of cycles for postponement.

Outputs

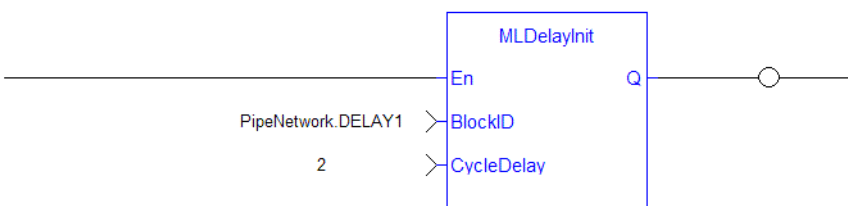
Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Function or function block executed successfully.

FBD Language Example



FFLD Language Example

Network #1



End of Module

IL Language Example

Not available.

ST Language Example

```
MLDelayInit(PipeNetwork.DELAY1, 2 );
```

3.1.10 Derivator

Name	Description
MLDerInit	Initializes an derivator object.
MLDerReadInModPos	Returns the Input ModuloPosition of the derivator block.
MLDerWriteInModPos	Sets the Input ModuloPosition of the Derivator block.

3.1.10.1 MLDerInit

Pipe Network



Function - Initializes an derivator object.

- Function is automatically called if a Derivator Block is added to the Pipe Network with user-defined settings entered in the Pipe Blocks Properties screen.
- Input ModuloPosition is defined to manage the periodicity (modulo) of the input values.

NOTE

- Derivator objects are normally created in the Pipe Network using the graphical engine.
- You do not have to add **MLDerInit** functions to their programs.
- Parameters are entered directly in pop-up windows; the code is automatically added to the current project.

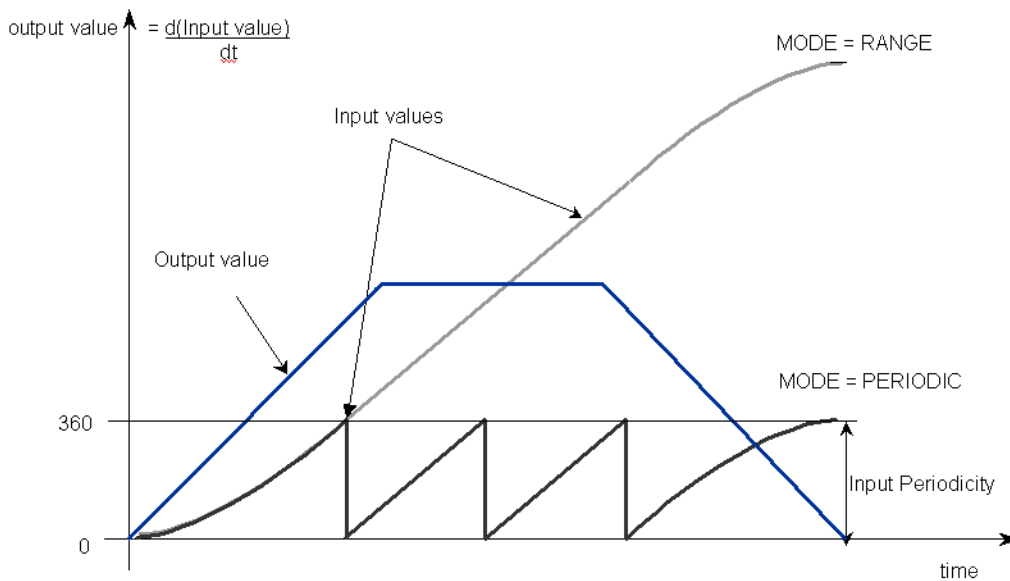


Figure 4-37: MLDerInit

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of a created Pipe Block.

Input	Data Type	Range	Unit	Default	Description
ModuloPosition	LREAL	0 to 1.0x10 ³⁰⁰	User units	No default	Value of the period of a cyclic system expressed in User units. - The parameter INPUT_MODULO_POSITION is defined to correctly manage the periodicity (modulo) of the input values. 0 (zero) indicates the input signal does not use the modulo operation. Example: If the input value increases each millisecond by one (degree), then the output value is a thousand (degrees per second). - If the input value skips suddenly from 359 to 0: - If VALUE PERIOD = 360, the output continues to indicate 1000 (degrees per second). - This indicates that the roll-over into the next period has been properly handled. - If VALUE PERIOD = 1000, the output indicates -359,000 (degrees per second). - This indicates that the input has incorrectly interpreted roll-over as a 359 degree change in input in one millisecond.

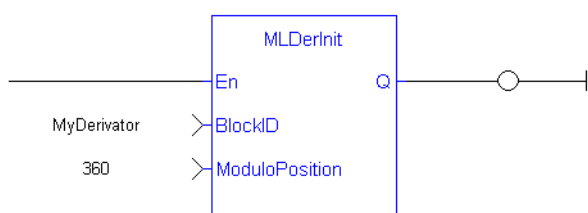
Outputs

Input	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE if the Derivator object is initialized.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Create and Initiate a Derivator object  
MyDerivator := MLBlkCreate( 'MyDerivator', 'DERIVATOR' );  
MLDerInit( MyDerivator, 360.0 );
```

See Also

- "MLBlkCreate" (→ p. 292)
- "MLDerReadInModPos" (→ p. 460)
- "MLDerReadInModPos" (→ p. 460)

3.1.10.2 MLDerReadInModPos



Function - Returns the Input ModuloPosition of the derivator block.

Input ModuloPosition is defined to manage the periodicity (modulo) of the input values.

Example

If the input value increases each millisecond by one degree then the output value is 1000 degrees per second. Suddenly, the input value skips from 359 to 0 (zero).

- Input ModuloPosition = 360, the output continues to indicate 1000 degrees per second.
 - This indicates that rollover into the next period has been properly handled
- Input ModuloPosition = 1000, the output then indicates 359,000 degrees per second.
 - This indicates that the input has incorrectly interpreted roll-over as a 359 degree move in one millisecond.

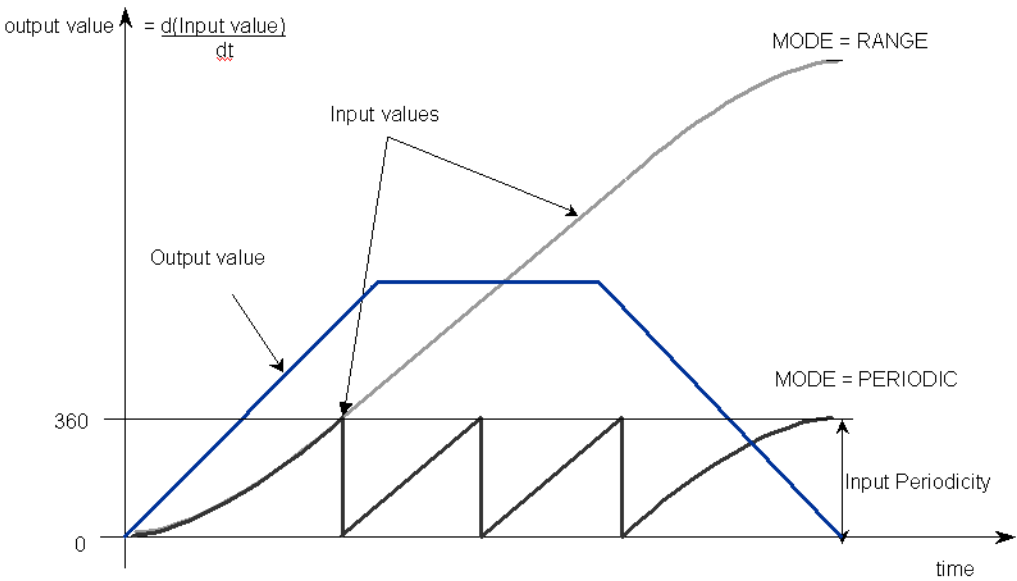


Figure 4-38: MLDerReadInModPos

NOTE

The first calculation of a Derivator Pipe Block just after the pipe installation indicates 0 (zero) regardless of the initial input value.

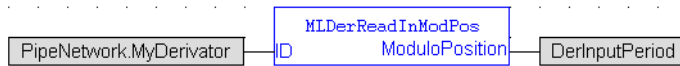
Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID number of an initiated Derivator object.

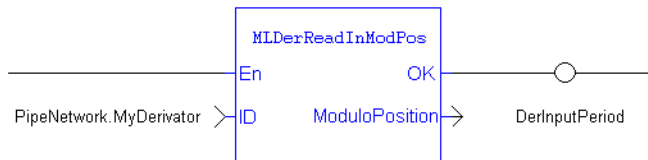
Outputs

Output	Data Type	Range	Unit	Description
ModuloPosition	LREAL	No range	User units	Current Input ModuloPosition of the selected Derivator object.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//save the current input MODULO_POSITION of a Derivator object
DerInputPeriod := MLDerReadInModPos ( PipeNetwork.MyDerivator );
```

See Also

- "MLDerInit" (→ p. 457)
- "MLDerWriteInModPos" (→ p. 462)

3.1.10.3 MLDerWriteInModPos



Function - Sets the Input ModuloPosition of the Derivator block.

Input ModuloPosition is defined to manage the periodicity (modulo) of the input values.

Example

If the input value increases each millisecond by one degree then the output value is 1000 degrees per second. Suddenly, the input value skips from 359 to 0 (zero).

- Input ModuloPosition = 360, the output continues to indicate 1000 degrees per second.
 - This indicates that rollover into the next period has been properly handled
- Input ModuloPosition = 1000, the output then indicates 359,000 degrees per second.
 - This indicates that the input has incorrectly interpreted roll-over as a 359 degree move in one millisecond.

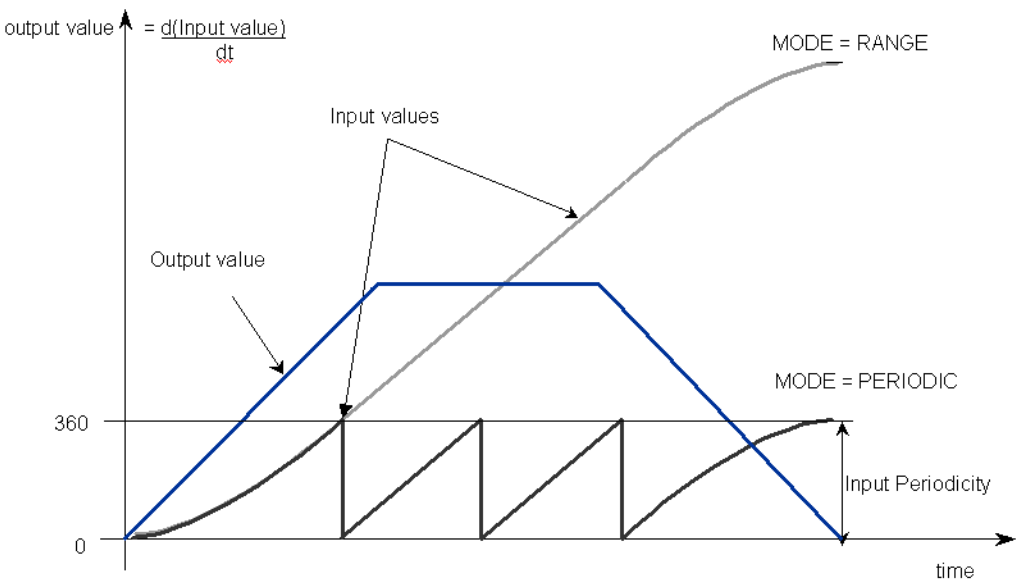


Figure 4-39: MLDerWriteInModPos

NOTE

The first calculation of a Derivator Pipe Block just after the pipe installation indicates 0 (zero) regardless of the initial input value.

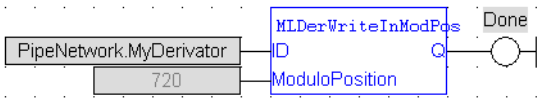
Inputs

Input	Data Type	Range	Unit	Default	Description
ID	DINT	1 to 1024	N/A	No default	ID number of an initiated Derivator object.
ModuloPosition	LREAL	No range	User units	No default	Designated new value of Input ModuloPosition of the selected Derivator object.

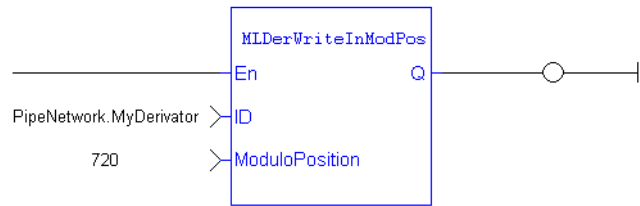
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Input ModuloPosition value is changed. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//change the input MODULO_POSITION of a Derivator object to 720
MLDerWriteInModPos ( PipeNetwork.MyDerivator, 720 );
```

See Also

- "MLDerInit" (→ p. 457)
- "MLDerReadInModPos" (→ p. 460)

3.1.11 Gear

Name	Description
MLGearInit	Initializes a Gear Pipe Block with user-defined settings and for use in a PLC program.
MLGearReadOffset	Returns the Offset value of a selected Gear Block from the Pipe Network.
MLGearReadOffSlp	Returns the Offset Slope value of a selected Gear Block from the Pipe Network.
MLGearReadRatio	Returns the Ratio value of a selected Gear Block from the Pipe Network.
MLGearReadRatSlp	Returns the Ratio Slope value of a selected Gear Block from the Pipe Network.
MLGearWriteOff	Sets the Offset value of a selected Gear Pipe Block.
MLGearWriteOSlp	Sets the Offset Slope value of a selected Gear Pipe Block.
MLGearWriteRatio	Sets the Ratio value of a selected Gear Pipe Block.
MLGearWriteRatSlp	Sets the Ratio Slope value of a selected Gear Pipe Block.

3.1.11.1 Examples: Gear Functions

The output value starts with offset = 0 and gain = 1 (blue line). (Figure 4-40)

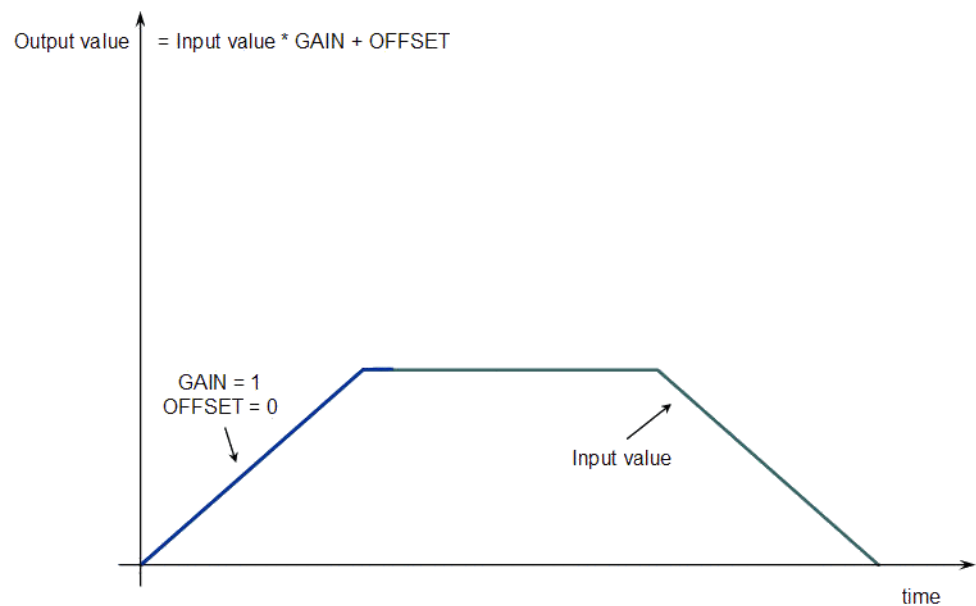


Figure 4-40: Offset = 0 and Gain = 1

Call the **MLGearWriteOff** function to modify the Offset.

This is where Offset_Slope is set with the **MLGearWriteOSlp** function. (Figure 4-41)

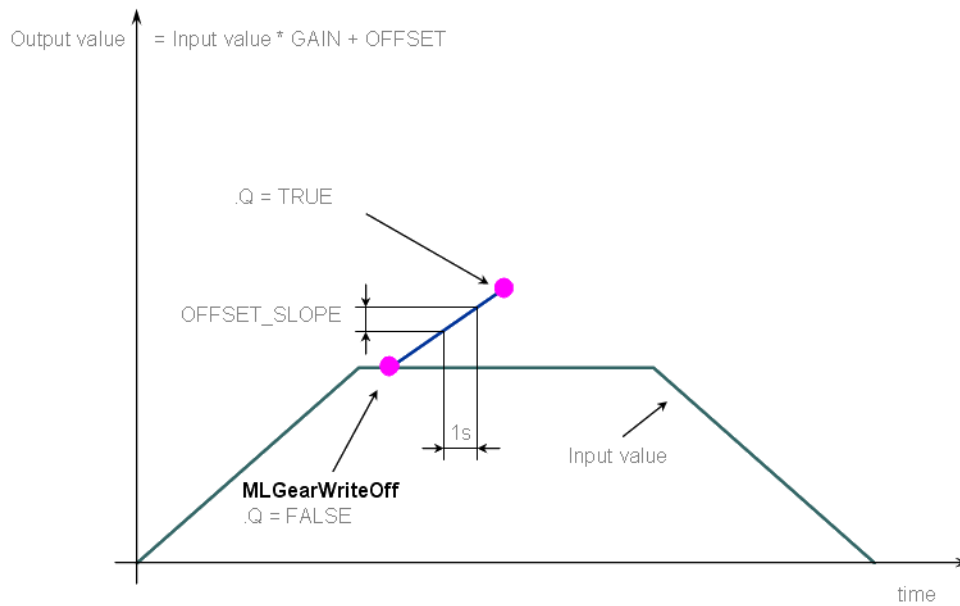


Figure 4-41: Offset_Slope set with MLGearWriteOSlp

After setting the Offset (Q=TRUE in the previous figure), call the **MLGearWriteRatio** function to modify the gear Ratio.

This is where Gain_Slope is set with the **MLGearWriteRatSlp** function. (Figure 4-42)

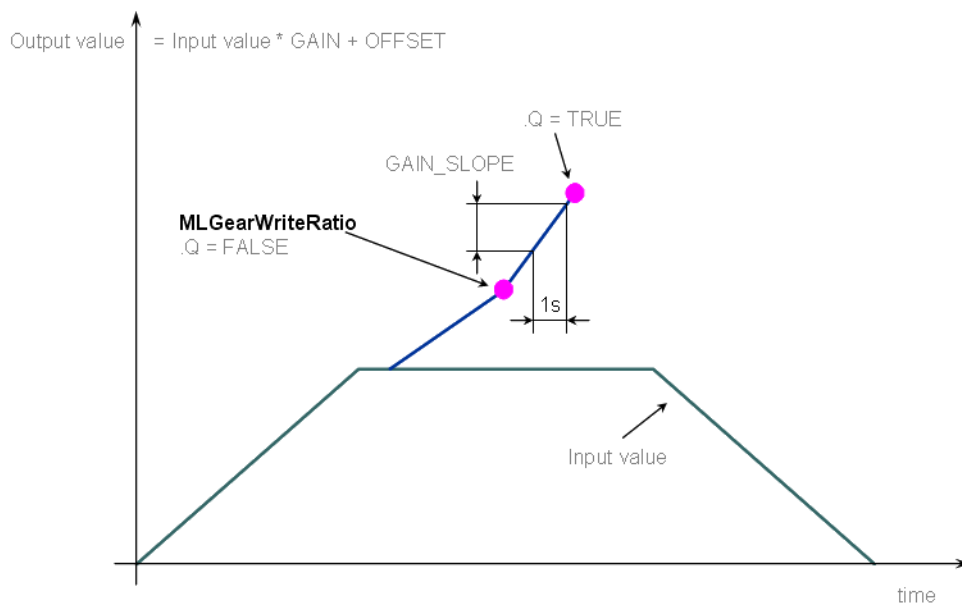


Figure 4-42: Gain_Slope set with MLGearWriteRatSlp

The output value is adapted with the gear offset and ratio (blue line). (Figure 4-43)

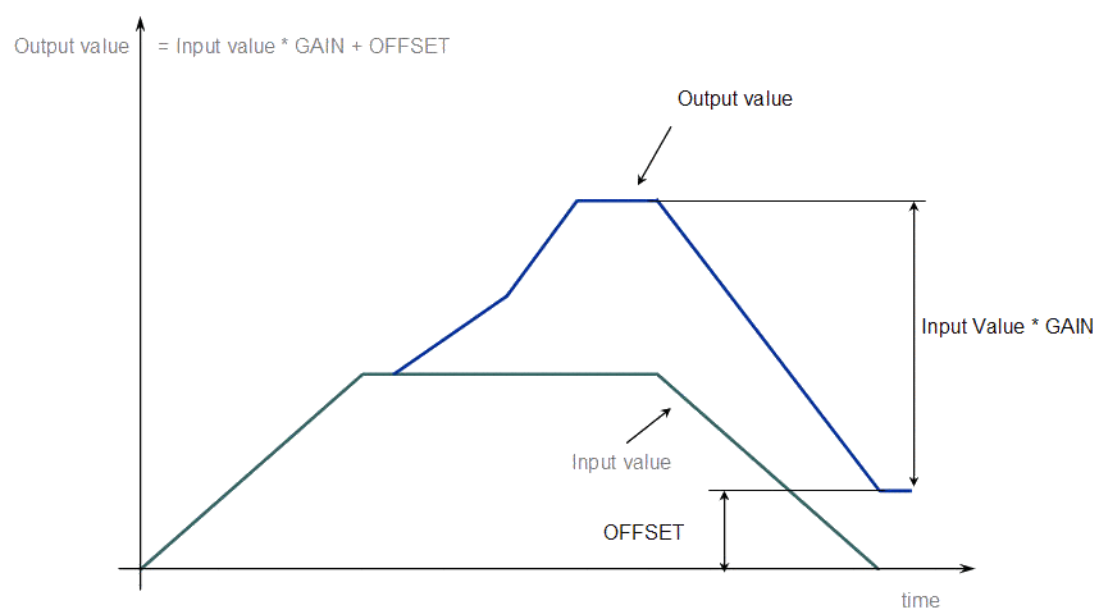


Figure 4-43: Output value adapted with gear offset and ratio

3.1.11.2 MLGearInit

Pipe Network



Function - Initializes a Gear Pipe Block with user-defined settings and for use in a PLC program.

- This function is automatically called if a Gear Block is added to the Pipe Network, with user-defined settings entered in the **Pipe Blocks Properties** screen.
- The Pipe Block is assigned a **Name**, **Ratio**, **Offset**, and **Slopes** for changes in Ratio and Offset values.
 - Choose between Modulo or Not modulo mode.
 - Slopes set the limit at which step changes in Ratio and Offset are implemented.

ⓘ IMPORTANT

- Set the $\text{RatioSlope} < (\text{Ratio} * \text{EtherCAT Update Rate})$.
- The Gear block makes a jump (without a ramp) from one gear to the next when the RatioSlope is greater than the Ratio change factor multiplied by the update rate scale factor.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of a created Pipe Block.
Ratio	LREAL	No range	N/A	No default	Ratio coefficient of a new Gear Pipe Block. • Values lower than 1.0 can be entered, but require a leading 0 (zero). • Example: 0.8 instead of .8.
Offset	LREAL	No range	N/A	No default	Input offset of a Gear Pipe Block. • Values lower than 1.0 can be entered, but require a leading 0 (zero). • Example: 0.8 instead of .8.
UseUserRatioSlope	BOOL	FALSE, TRUE	N/A	No default	FALSE to use the maximum Slope. • FALSE causes an instantaneous gear change within one cycle. • Use TRUE to use user-defined RatioSlope.

Input	Data Type	Range	Unit	Default	Description
RatioSlope	LREAL	No range	1/sec	No default	User-defined limit at which step changes in Ratio are implemented. - This sets the maximum rate of change at the pipe block output resulting from changes in Ratio parameters. - When set to the MAX (the default setting), the slope is infinite. - Units are 1/second for RatioSlope. - Values lower than 1.0 can be entered, but require a leading 0 (zero). - Example: 0.8 instead of .8.
UseUserOffsetSlope	BOOL	FALSE, TRUE	N/A	No default	FALSE to use the maximum Slope. - FALSE causes an instantaneous gear change within one cycle. - Use TRUE to use user-defined OffsetSlope.
OffsetSlope	LREAL	No range	User units / sec	No default	User-defined limit at which step changes in Offset are implemented. - This sets the maximum rate of change at the pipe block output resulting from changes in Offset parameters. - When set to the MAX (the default setting), the slope is infinite. - Units are user units per second for OffsetSlope. - Values lower than 1.0 can be entered, but require a leading 0 (zero). - Example: 0.8 instead of .8.
Modulo	BOOL	FALSE, TRUE	N/A	No default	Modulo mode adapts the output values according to the ModuloPosition (modulo). - TRUE when the mode is modulo. - Modulo mode adapts the output values according to the ModuloPosition (modulo).

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Returns TRUE if the Gear Pipe Block is initialized. - The output of a Gear Block = Input value * Ratio + Offset.

Remarks

3.1.11.2.1.1 Modulo Value

If the Gear block's input is a modulo value and the position delta is greater than $\frac{1}{2}$, the modulo value within one sample period in the opposite direction, the Gear block cannot detect the change in the direction of motion.

Example

- The sample period is: 1 msec.
- The Master is configured for a 360 degree modulo.
- The Master position is changed by >180 degrees within 1 msec.
 - In this case, the Gear block cannot determine whether the direction is in the same or opposite direction.

3.1.11.3 Avoid Modulo Calculation Problems

- Either deactivate and reactivate the PipeNetwork when forcing the Master position with "MLMstForcePos" (→ p. 497).
- Use a "MLMstAbs" (→ p. 490) or "MLMstRel" (→ p. 508) move to force the Master's position value.

Example

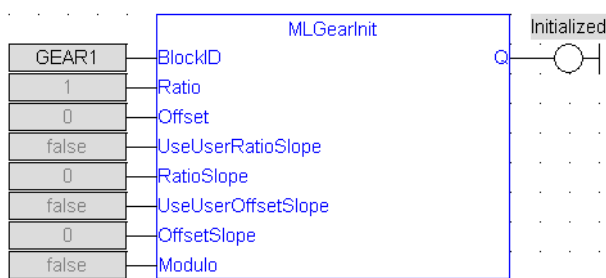
To force the Master position to 0 (zero), use this code:

```
PipeNetwork(MLPN_DEACTIVATE);
MLMstForcePos(PipeNetwork.MASTER, 0);
PipeNetwork(MLPN_ACTIVATE);
```

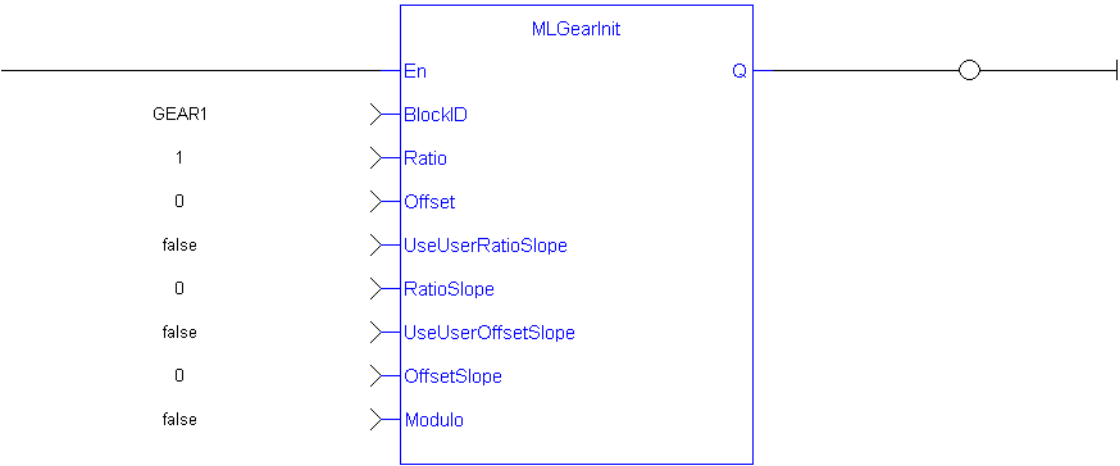
TIP

- Gear objects are normally created in the Pipe Network using the graphical engine.
- You do not have to add **MLGearInit** functions to their programs.
- Parameters are entered directly in pop-up windows; the code is automatically added to the current project.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Initialize a Gear Pipe Block named GEAR1 with:  
//  Ratio = 1,Offset = 0, User Ratio Slope OFF, User Ratio  
//  Slope = 0, Offset Slope = 0, and no Modulo  
GEAR1 := MLBlkCreate( 'GEAR1', 'GEAR' );  
MLGearInit( GEAR1, 1.0, 0.0, false, 0.0, false, 0.0, false);
```

See Also

- "MLBlkCreate" (→ p. 292)
- "MLGearWriteRatio" (→ p. 479)

3.1.11.4 MLGearReadOffset

Pipe Network ✓



Function - Returns the Offset value of a selected Gear Block from the Pipe Network.

The output of a Gear Block = Input value * Ratio + Offset.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initialized Gear object.

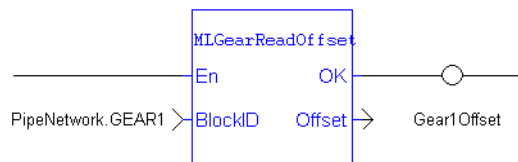
Outputs

Output	Data Type	Range	Unit	Description
Offset	LREAL	No range	User units	The offset value currently assigned to the selected Gear Pipe Block.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Find the Offset value of Gear1 Pipe Block
Gear1Offset := MLGearReadOffset( PipeNetwork.GEAR1 );
```

See Also

- "MLGearInit" (→ p. 467)
- "MLGearWriteOff" (→ p. 475)

3.1.11.5 MLGearReadOffSlp



Function - Returns the Offset Slope value of a selected Gear Block from the Pipe Network.

- Offset Slope sets the limit in User Units per Second at which step changes in offset are implemented.
- The default value when creating a Gear Block is OFFSET_SLOPE_MAX or infinite.

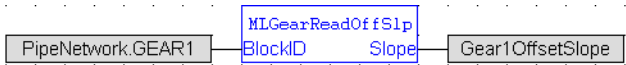
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initialized Gear object.

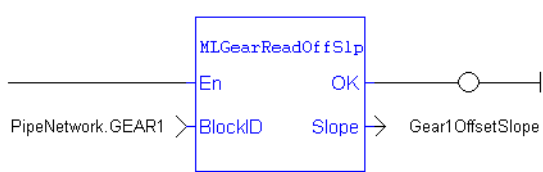
Outputs

Output	Data Type	Range	Unit	Description
Slope	LREAL	≥ 0 (zero)	User units / sec	• The Offset Slope value currently assigned to the selected Gear Pipe Block. • This can be a different sign than what is programmed with MLGearWriteOSlp.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Find the Offset Slope value of Gear1 Pipe Block
Gear1OffsetSlope := MLGearReadOffSlp(PipeNetwork.GEAR1);
```

See Also

- "MLGearInit" (→ p. 467)
- "MLGearWriteOSlp" (→ p. 477)

3.1.11.6 MLGearReadRatio

Pipe Network ✓



Function - Returns the Ratio value of a selected Gear Block from the Pipe Network.

The output of a Gear Block = Input value * Ratio + Offset.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initialized Gear object.

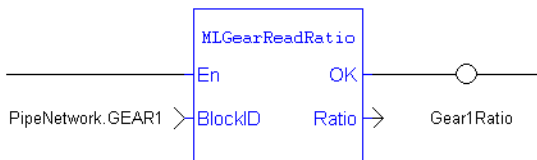
Outputs

Output	Data Type	Range	Unit	Description
Ratio	LREAL	No range	N/A	The Ratio value currently assigned to the selected Gear Pipe Block.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Find the Ratio value of Gear1 Pipe Block
Gear1Ratio := MLGearReadRatio(PipeNetwork.GEAR1);
```

See Also

- "MLGearInit" (→ p. 467)
- "MLGearWriteRatio" (→ p. 479)

3.1.11.7 MLGearReadRatSlp



Function - Returns the Ratio Slope value of a selected Gear Block from the Pipe Network.
The Ratio Slope sets the limit in 1/Seconds (or s⁻¹) at which step changes in Ratio are implemented.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initialized Gear object.

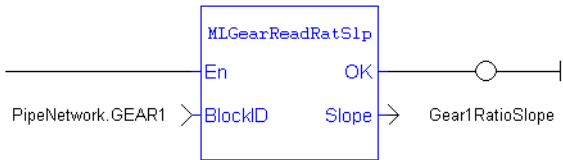
Outputs

Output	Data Type	Range	Unit	Description
Slope	LREAL	≥ 0 (zero)	1/sec (or s ⁻¹)	- The Ratio Slope value currently assigned to the selected Gear Pipe Block. - This can be a different sign than what is programmed with MLGearWriteRatSlp.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Find the Ratio Slope value of Gear1 Pipe Block
Gear1RatioSlope := MLGearReadRatSlp(PipeNetwork.GEAR1);
```

See Also

- "MLGearInit" (→ p. 467)
- "MLGearWriteRatSlp" (→ p. 481)

3.1.11.8 MLGearWriteOff

Pipe Network 



Function - Sets the Offset value of a selected Gear Pipe Block.

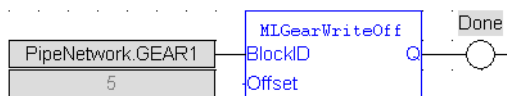
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initialized Gear Pipe Block.
Offset	LREAL	No range	User units	No default	Input offset of a Gear Pipe Block. - Values lower than 1.0 can be entered, but require a leading 0 (zero). - Example: 0.8 instead of .8.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Returns TRUE if the Offset value is changed in the selected Gear Pipe Block. The output of a Gear Block = Input value * Ratio + Offset.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Set the Offset value of Gear1 Pipe Block to 5 User Units  
MLGearWriteOff(PipeNetwork.GEAR1, 5.0);
```

See Also

- "MLGearInit" (→ p. 467)
- "MLGearReadOffset" (→ p. 471)

3.1.11.9 MLGearWriteOSlp

Pipe Network



Function - Sets the Offset Slope value of a selected Gear Pipe Block.

- Offset Slope sets the limit in User Units per Second at which step changes in offset are implemented.
- The default value when creating a Gear Block is OFFSET_SLOPE_MAX or infinite.

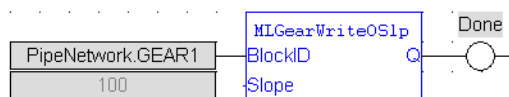
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initialized Gear Pipe Block.
Slope	LREAL	No range	User units / sec	No default	Input offset of a Gear Pipe Block. • Values lower than 1.0 can be entered, but require a leading 0 (zero). • Example: 0.8 instead of .8.

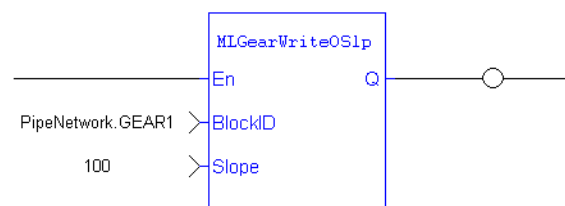
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Offset Slope value is changed in the selected Gear Pipe Block. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Set the Offset Slope value of Gear1 Pipe Block to 100  
MLGearWriteOSlp(PipeNetwork.GEAR1, 100.0);
```

See Also

- "MLGearInit" (→ p. 467)
- "MLGearReadOffSlp" (→ p. 472)

3.1.11.10 MLGearWriteRatio

Pipe Network ✓



Function - Sets the Ratio value of a selected Gear Pipe Block.

The output of a Gear Block = Input value * Ratio + Offset.

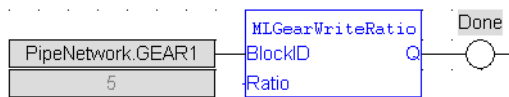
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initialized Gear Pipe Block.
Ratio	LREAL	No range	User units / sec	No default	Input offset of a Gear Pipe Block. - Values lower than 1.0 can be entered, but require a leading 0 (zero). - Example: 0.8 instead of .8.

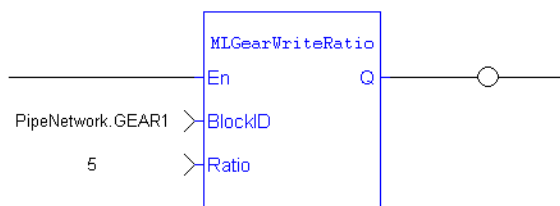
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE if the Ratio value is changed in the selected Gear Pipe Block.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Set the Ratio value of Gear1 Pipe Block to 5  
MLGearWriteRatio(PipeNetwork.GEAR1, 5.0);
```

See Also

- "MLGearInit" (→ p. 467)
- "MLGearReadRatio" (→ p. 473)

3.1.11.11 MLGearWriteRatSlp

Pipe Network



Function - Sets the Ratio Slope value of a selected Gear Pipe Block.

- Ratio Slope sets the limit at which step changes in ratio are implemented.
- The GEAR block output adds a position offset to the GEAR block input when using a RatioSlope.
- See "RatioSlope Offset" (→ p. 481) for examples.

ⓘ IMPORTANT

- Set the `RatioSlope < (Ratio * EtherCAT Update Rate)`.
- The Gear block makes a jump (without a ramp) from one gear to the next when the RatioSlope is greater than the Ratio change factor multiplied by the update rate scale factor.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initialized Gear Pipe Block.
Slope	LREAL	No range	1/sec	No default	Input offset of a Gear Pipe Block. • Values lower than 1.0 can be entered, but require a leading 0 (zero). • Example: 0.8 instead of .8.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Ratio Slope value is changed in the selected Gear Pipe Block. •

Remarks

3.1.11.11.1 RatioSlope Offset

3.1.11.12 With Offset

- If **MLGearWriteRatSlp** is set as **MLGearWriteRatSlp** (PipeNetwork.GEAR1 12, Gear1RatioSlope 500.0) ; to generate a ramp (instead of a step) when going from a gear ratio of 1 to 2, there is a position offset when the gear ratio settles as 2.
- In this image:
 - The ratio goes from 1.0 to 2.0.
 - Green is PN Gear Block Output.
 - Red is Gearbox Input.

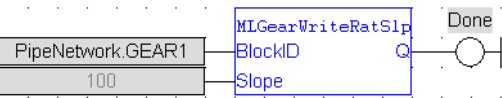
1. Green line: PN Gear Block Output.
2. Red line: PN Gearbox Input.
3. When the ratio changes.
4. Phase difference.

3.1.11.13 Without Offset

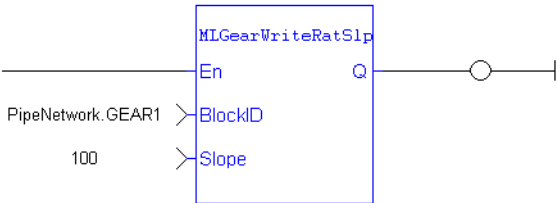
If `MLGearWriteRatSlp` is set without a ramp, `MLGearWriteRatSlp (PipeNetwork.GEAR1 12, Gear1RatioSlope 1e+301) ;`, there is no offset.

1. Green line: PN Gear Block Output.
2. Red line: PN Gearbox Input.
3. When the ratio changes.
4. Synched.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Set the Ratio Slope value of Gear1 Pipe Block to 100
MLGearWriteRatSlp(PipeNetwork.GEAR1, 100.0);
```

See Also

- "MLGearInit" (→ p. 467)
- "MLGearReadOffSlp" (→ p. 472)

3.1.12 Integrator

Name	Description
MLIntInit	Initializes an integrator object.
MLIntWriteOutVal	Sets the output value of an integrator object.

3.1.12.1 MLIntWriteOutVal



Function - Sets the output value of an integrator object.

- This function can force the output to an entered value not dependent on the input value from the Pipe Network.
- Output value can jump to another value instantly after the function is executed if the Pipe Network is running.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

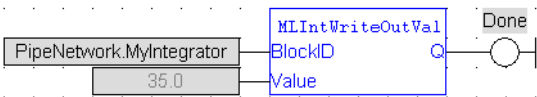
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Integrator object.
Value	LREAL	No range	User units	No default	Designated new output value of the selected Integrator object.

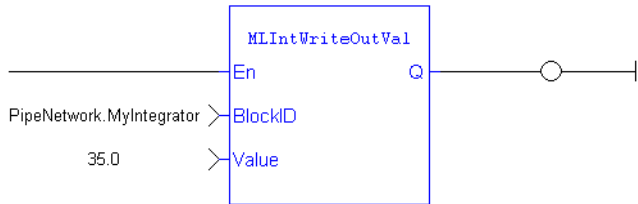
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the output value if the Integrator object is changed. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//change the output value of an integrator object to 35  
MLIntWriteOutVal ( PipeNetwork.MyIntegrator, 35.0 );
```

See Also

"MLIntInit" (→ p. 486)

3.1.12.2 MLIntInit



Function - Initializes an integrator object.

- The function is automatically called if an Integrator Block is added to the Pipe Network.
 - User-defined settings are entered in the **Pipe Blocks Properties** screen.
- The Integrator object can operate in modulo or not modulo mode.
 - While in modulo mode, the output values are adapted according to the entered **ModuloPosition** value.

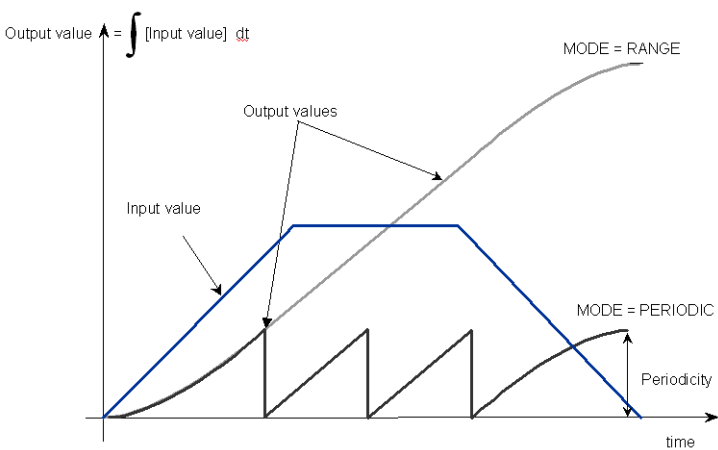


Figure 4-44: MLIntInit

NOTE

- Integrator objects are normally created in the Pipe Network using the graphical engine.
- You do not have to add **MLIntInit** functions to their programs.
- Parameters are entered directly in pop-up windows; the code is automatically added to the current project.

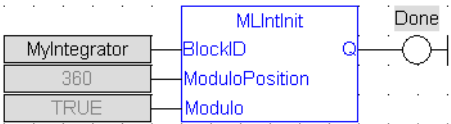
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of a created Pipe Block.
ModuloPosition	LREAL	No range	User units	No default	• When mode is set to Modulo, integrate the input values with respect to time. • The OUTPUT_MODULO_POSITION is defined to correctly manage the periodicity (modulo) of the output values.
Modulo	BOOL	FALSE, TRUE	N/A	No default	• TRUE when mode is modulo. • Modulo mode adapts the output values according to the ModuloPosition (modulo).

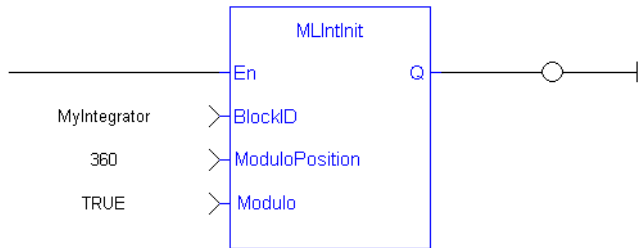
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Integrator object is initialized. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Initiate an Integrator Pipe Block named "MyIntegrator" with a Modulo of 360
MLIntInit(MyIntegrator, 360.0, true );
```

See Also

- "MLBlkCreate" (→ p. 292)
- "MLIntWriteOutVal" (→ p. 484)

3.1.13 Master

Name	Description
MLMstAbs	Performs a move to an absolute position.
MLMstAdd	Performs a move for a specified distance relative to the endpoint of the previous move.
MLMstForcePos	Forces the position of a Master Block to a specified position.
MLMstInit	Initializes a Master TMP generator block.
MLMstReadAccel	Gets the present acceleration value of a master block.
MLMstReadDecel	Gets the present deceleration value of a master block.
MLMstReadInitPos	Gets the initial position of a master block.
MLMstReadSpeed	Gets the speed of a master block.
MLMstRel	Performs a relative move for a specified distance from the current position.
MLMstRun	Jogs at the specified speed.
MLMstStatus	Returns the status of the generator.
MLMstWriteAccel	Sets the acceleration of a master block.
MLMstWriteDecel	Sets the deceleration of a master block.
MLMstWriteInitPos	Sets the initial position of a master block.
MLMstWriteSpeed	Sets the speed of a master block.

3.1.13.1 Examples: Master Function

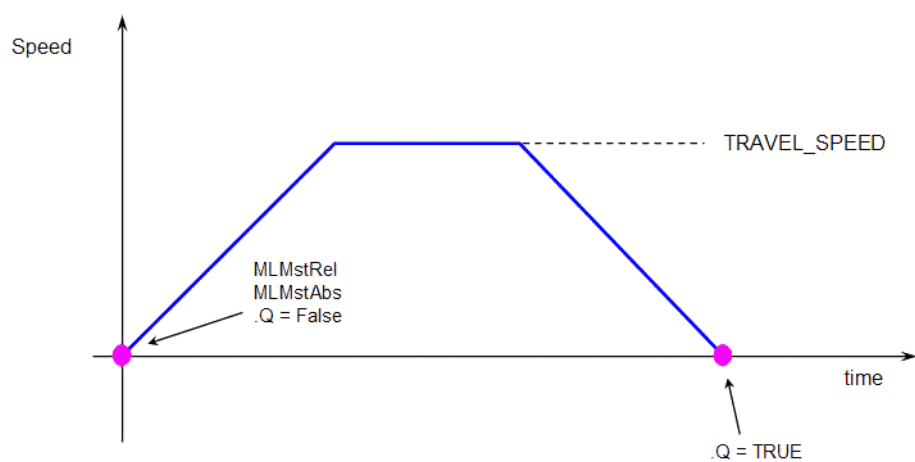


Figure 4-45: MLMstRun(0.0) reduce the speed down to 0.

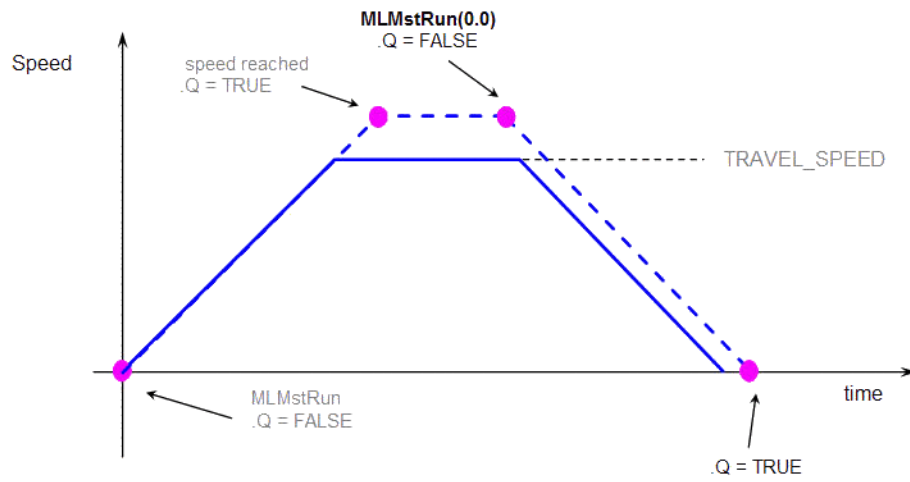


Figure 4-46: Master Functions Usage

3.1.13.2 MLMstAbs

Pipe Network ✓



Function - Performs a move to an absolute position.

NOTE

Perform one of these actions to prevent undesired axis movement if modulo is turned on:

- Verify the **Position** value is within the modulo range.
- If the **Position** value is outside of the modulo range, call **MLMstAbs** function only once. See the **TIP** for an example of how to do this.

TIP

To reduce the load on the CPU, call **MLMstAbs** only once for each move. This is achieved by adding a control variable, shown here:

```
// Master: modulo is on and modulo range is 0 - 360 with rollover
at 360

If Not MoveStarted Then
    Position := 500;
    MLMstAbs( PipeNetwork.MASTER, Position);
    MovesStarted := TRUE;
End_if;
```

3.1.13.2.1.1 Position with Modulo On

Position with Modulo On

NOTE

This information applies to both "MLAxisAbs" (→ p. 313) and "MLMstAbs" (→ p. 490). For simplicity, the term Axis Block also refers to Master Block.

When the Modulo is turned on, the Axis Block moves to the targeted position during the corresponding period and is calculated as:

- If the Position input is between 0 (zero) and the Modulo position, the Axis Block moves within the **current** period (no position rollover).
- If the Position input is greater than the Modulo position, the Axis Block moves during one of the **next** period (positive position rollover).

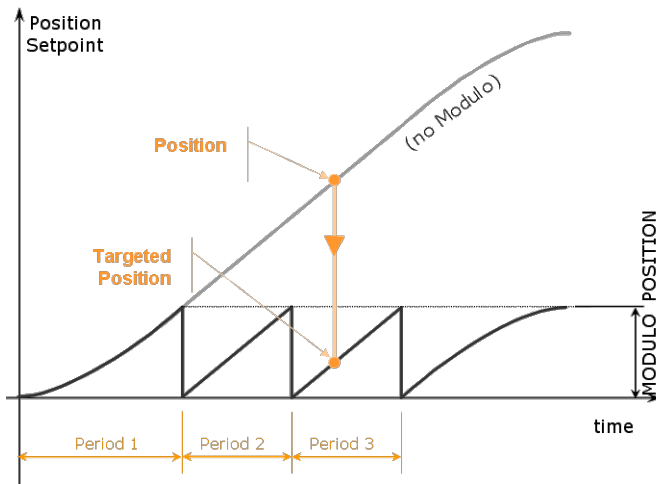


Figure 4-47: MLAxisAbs or MLMstAbs Modulo

The Axis Block works similarly for negative positions.

If the Position input is less than 0 (zero), the Axis Block moves during one of the **previous** period (negative position rollover).

3.1.13.3 Forcing the Direction of Rotation

In some applications, the direction of rotation for the axis is forced in one direction only.

As a consequence, the motor movement goes to the next or previous modulo in these situations:

3.1.13.3.1.1 End Position is Less Than Start Position

If the **End Position** is less than the **Start Position** and the direction of rotation for the axis is forced to be clockwise, the **red point** shows when the Modulo position is reached.

See Row 2 of the "MLMstAbs" (→ p. 490).

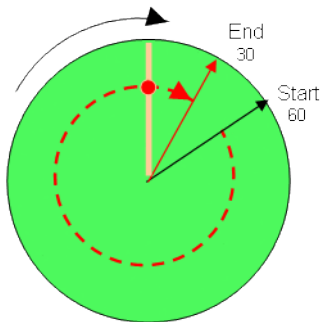


Figure 4-48: MLAxisAbs or MLMstAbs - End Position Less Than Start Position

3.1.13.3.2.2 End Position is Greater Than Start Position

If the **End Position** is greater than the **Start Position** and the direction of rotation for the axis is forced to be counter clockwise.

See Row 4 of the "MLMstAbs" (→ p. 490).

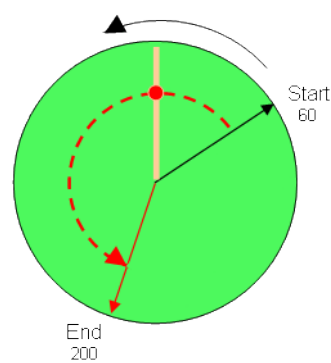


Figure 4-49: MLAxisAbs or MLMstAbs - End Position Greater Than Start Position

Example Table

Start Position	End Position	Direction of Rotation	Cross Modulo	Position Input to MLAxisAbs (1)	RelativeDistance Moved (2)	
60	200	Clockwise	No	200	140	(i.e., 200 - 60 + 0)
60	30	Clockwise	Yes	390	330	(i.e., 30 - 60 + 360)
60	30	Counter clockwise	No	30	-30	(i.e., 30 - 60 - 0)
60	200	Counter clockwise	Yes	-160	-220	(i.e., 200 - 60 - 360)

With:

- (1) **Position Input** = End Position (+ Modulo * Direction of rotation)
- (2) **Relative Distance Moved** = End Position - Start Position (+ Modulo * Direction of rotation)

Where:

Direction of rotation = 1 when clockwise and -1 when anti-clockwise.

3.1.13.3.3.3 Travel Speed Update with MLAxisAbs

The travel speed of the generator can be updated using the function block "MLAxisGenWriteSpd" (→ p. 343). Depending on the state of the generator, this speed is directly reflected on the current move or a future move.

- If **MLAxisAbs** is not currently being executed, the new travel speed is applied for the trajectory calculation for a future **MLAxisAbs** command.
- If **MLAxisAbs** is currently being executed, and a new MLAxisAbs with the **same** target position is called, the new travel speed is taken into account only if the current state of the TMP profile is the constant velocity or acceleration.
 - If the axis was decelerating to stop at the goal position the new travel speed is not taken into account.
- If a **MLAxisAbs** is currently being executed, and a new MLAxisAbs with a **different** target position is called, the new travel speed is taken into account.

Example 1

-
- 1. First MLAxisAbs.
 - 2. Second MLAxisAbs.
 - 3. Initial Speed.
 - 4. New Speed.

Figure 4-50: Initial speed is smaller than the new speed.**Example 2**

1. First MLAxisAbs.
2. Second MLAxisAbs.
3. Initial Speed.
4. New Speed.

Figure 4-51: Initial speed is bigger than the new speed.**Example 3**

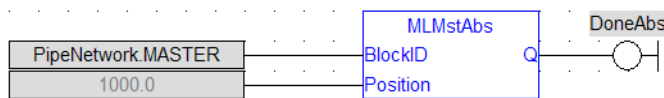
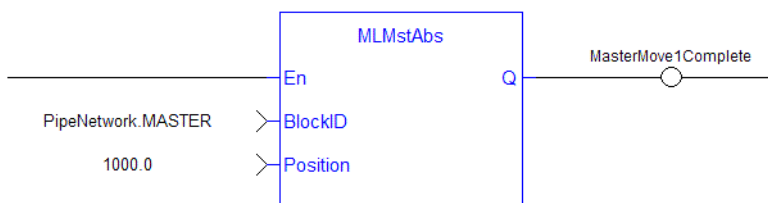
1. Acceleration.
2. Constant Velocity.
3. Deceleration.

Figure 4-52: The speed update is taken into account only if the second MLAxisAbs is triggered during acceleration or constant velocity.**Inputs**

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Master Block.
Position	LREAL	No range	User units	No default	Sets the value of the absolute destination position. • See "Position with Modulo On" (→ p. 490). • See Set the Axis Block Position Units.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example**FPLD Language Example**

IL Language Example

Not available.

ST Language Example

```
//Make an absolute positon move with a Master block called "MASTER" to  
position 1000.0  
MLMstAbs( PipeNetwork.MASTER, 1000.0 );
```

See Also

- "MLMstWriteSpeed" (→ p. 519)
- "MLMstReadDecel" (→ p. 504)

3.1.13.4 MLMstAdd

Pipe Network ✓



Function - Performs a move for a specified distance relative to the endpoint of the previous move.

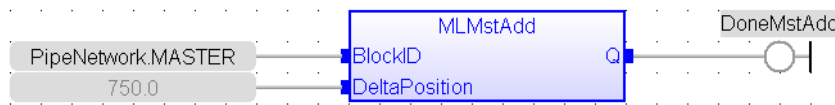
Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	0 to 1	N/A	No default	Enables the function or function block to be executed.
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Master Block.
DeltaPos	LREAL	No range	N/A	No default	Relative distance to move.

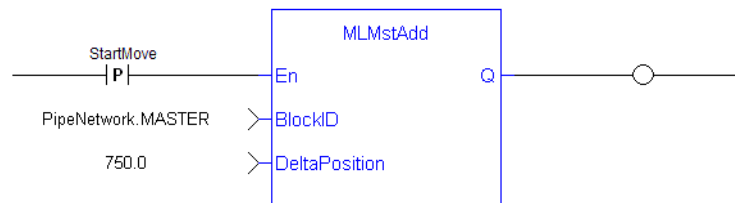
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.

FBD Language Example



FFLD Language Example



NOTE

- A positive transition contact is required to start the FB.
- See FFLD Contacts.

IL Language Example

Not available.

ST Language Example

```
//At the endpoint of the previous move, with the Master pipe block named  
"MASTER", make a move of 750.0  
MLMstAdd( PipeNetwork.MASTER, 750.0 );
```

See Also

- "MLMstWriteDecel" (→ p. 516)
- "MLMstWriteSpeed" (→ p. 519)

3.1.13.5 MLMstForcePos

Pipe Network



Function - Forces the position of a Master Block to a specified position.

- This block can only be executed when motion is **not** occurring.
- It can be used to force the master starting position to the desired values from which to start motion.

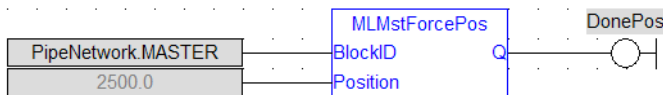
Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	0 to 1	N/A	No default	Enables the function or function block to be executed.
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Master Block.
Position	LREAL	No range	User units	No default	Defines the Master starting position when the motion starts.

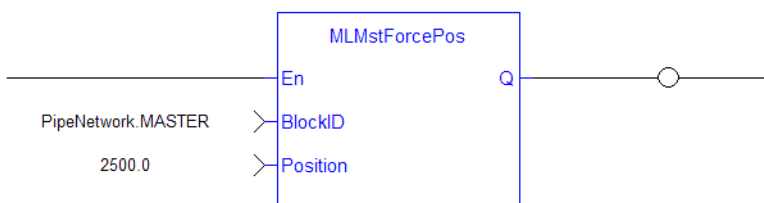
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



NOTE

- A positive transition contact is required to start the FB.
- See FFLD Contacts.

IL Language Example

Not available.

ST Language Example

```
//Reset the output position of the Master Pipe Block named "Master" to 2500.0  
MLMstForcePos(PipeNetwork.Master, 2500.0);
```

See Also

"MLMstReadInitPos" (→ p. 505)

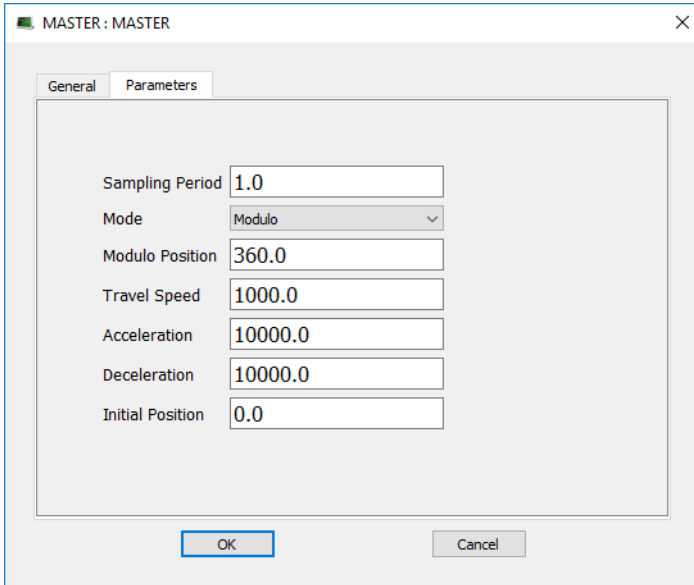
3.1.13.6 MLMstInit

Pipe Network



Function - Initializes a Master TMP generator block.

- This function is automatically created when the MLMaster Block is included in the Pipe Network Editor.
- Based on the parameters defined in the Master pipe block, the Inputs for this function are initialized by default.
 - See example in the Figure 4-53.



The screenshot shows a dialog box titled "MASTER: MASTER" with a close button (X) in the top right corner. It has two tabs: "General" and "Parameters". The "Parameters" tab is active, displaying several input fields with their current values: Sampling Period (1.0), Mode (Module), Modulo Position (360.0), Travel Speed (1000.0), Acceleration (10000.0), Deceleration (10000.0), and Initial Position (0.0). At the bottom of the dialog are "OK" and "Cancel" buttons.

Figure 4-53: TMP Initialization

Inputs

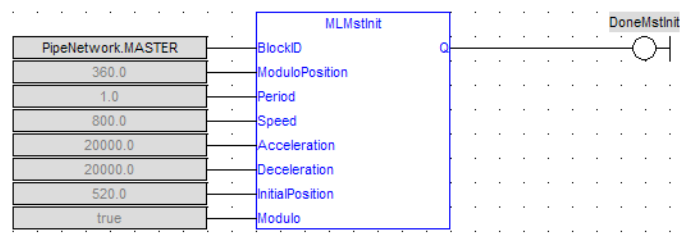
Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Master Block.
ModuloPosition	LREAL	No range	User units	No default	Modulo Position for cyclic motion systems expressed in user logical units (Position Rollover Value).
SamplingPeriod	LREAL	No range	Cycles	No default	• Sampling period of the generator expressed according to the update cycle. • Example: 2.0 means the sampling is done once every 2 cycles.
TravelSpeed	LREAL	No range	User units	No default	• Travel speed value expressed in user position units per second. • The travel speed value is used to set the constant speed part of the trapezoidal motion profile (TMP). • See Set the Axis Block Position Units.

Input	Data Type	Range	Unit	Default	Description
Acceleration	LREAL	No range	User units / sec ²	No default	- Acceleration value expressed in user position units per second squared. - The acceleration value is always used to generate the first part of the trapezoidal motion profile (TMP). - See Set the Axis Block Position Units.
Deceleration	LREAL	No range	User units / sec ²	No default	- Deceleration value expressed in user position units per second squared. - The deceleration value is always used to generate the last part of the trapezoidal motion profile (TMP). - See Set the Axis Block Position Units.
InitialPosition	LREAL	No range	User units	No default	- Initial position value expressed in user position units. - Used only at the pipe activation to initialize the position starting point.
Modulo	BOOL	FALSE, TRUE	N/A	No default	The available modes are: - TRUE - Modulo - FALSE - No modulo

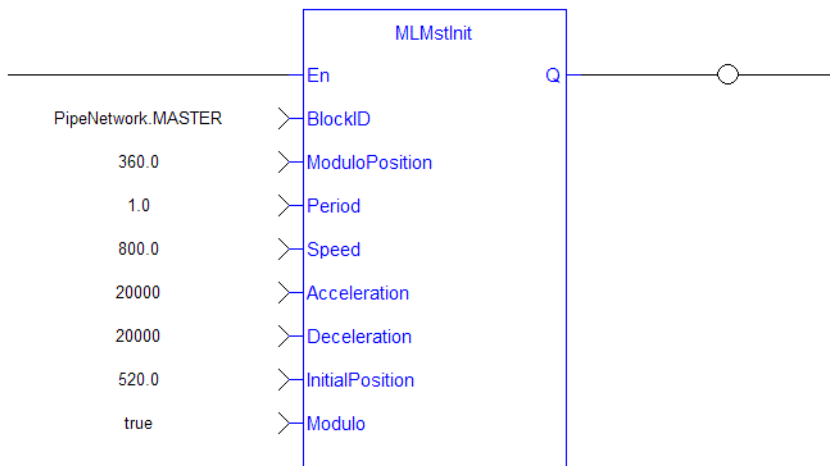
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Initialize a Master Pipe Block named "MASTER" to a Modulo of 360, motion
generator sample period of 1, Speed of 1000.0, Accel and Decel of 10000.0,
Initial position of 0.0,
MLMstInit( PipeNetwork.MASTER, 360.0, 1.0, 1000.0, 10000.0, 10000.0, 0.0,
true );
```

3.1.13.7 MLMstReadAccel



Function - Gets the present acceleration value of a master block.

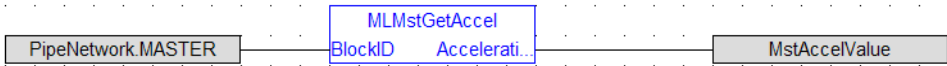
Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	0 to 1	N/A	No default	Enables the function or function block to be executed.
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Master Block.

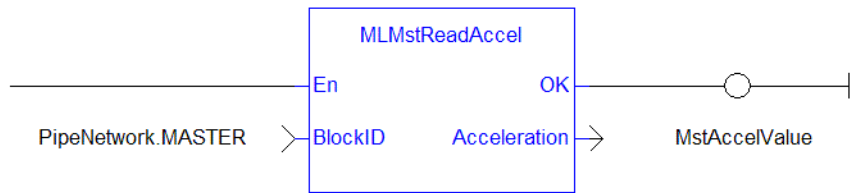
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes.
Acceleration	LREAL		User units / sec ²	- Returns the acceleration value. - See Set the Axis Block Position Units.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
// Read the present acceleration of a Pipe Block named "MASTER"
MLMstReadAccel( PipeNetwork.MASTER );
```

See Also

- "MLMstReadDecel" (→ p. 504)
- "MLMstReadSpeed" (→ p. 506)

3.1.13.8 MLMstReadDecel



Function - Gets the present deceleration value of a master block.

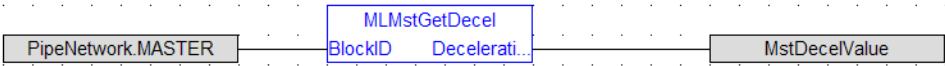
Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	0 to 1	N/A	No default	Enables the function or function block to be executed.
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Master Block.

Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •
Deceleration	LREAL		User units / sec ²	• Returns the deceleration value. • See Set the Axis Block Position Units.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
// Read the present deceleration of a Pipe Block named "MASTER"
MLMstReadDecel( PipeNetwork.MASTER );
```

3.1.13.9 MLMstReadInitPos

Pipe Network ✓



Function - Gets the initial position of a master block.

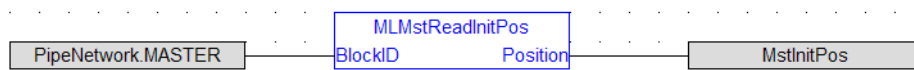
Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables the function or function block to be executed.
BlockID	DINT	1 to 1024	N/A	No default	Pipe Network Block.

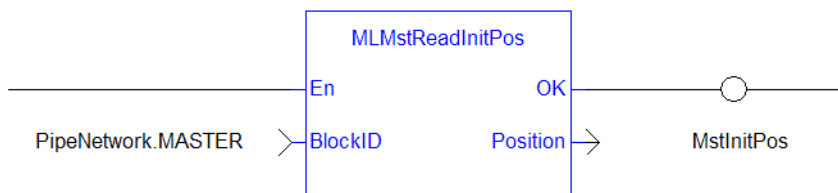
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes.
Position	LREAL		User units	- Returns Initial Position. - See Set the Axis Block Position Units.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MstInitPos := MLMstReadInitPos( PipeNetwork.MASTER );
```


3.1.13.10 MLMstReadSpeed



Function - Gets the speed of a master block.

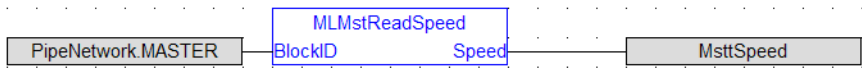
Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables the function or function block to be executed.
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Master Block.

Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •
Speed	LREAL		User units / sec	• Returns the current speed. • See Set the Axis Block Position Units.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MstSpeed := MLMstReadSpeed( PipeNetwork.MASTER );
```

See Also

- "MLMstReadAccel" (→ p. 502)
- "MLMstReadDecel" (→ p. 504)

3.1.13.11 MLMstRel



 **Function** - Performs a relative move for a specified distance from the current position.

Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables the function or function block to be executed.
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Master Block.
DeltaPos	LREAL	No range	User units	No default	Relative distance to the move.

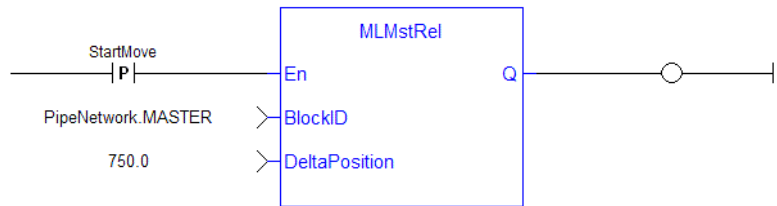
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.

FBD Language Example



FFLD Language Example



NOTE

- A positive transition contact is required to start the FB.
- See FFLD Contacts.

IL Language Example

Not available.

ST Language Example

```
MLMstRel( PipeNetwork.MASTER, 750.0 );
```

See Also

- "MLMstWriteDecel" (→ p. 516)
- "MLMstWriteSpeed" (→ p. 519)

3.1.13.12 MLMstRun



Function - Jogs at the specified speed.

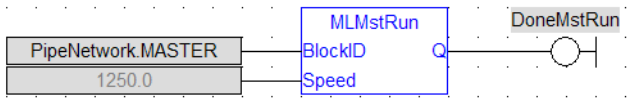
Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables the function or function block to be executed.
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Master Block.
Speed	LREAL	No range	User units / sec ²	No default	- Defines the Master starting position when the motion starts. - See Set the Axis Block Position Units.

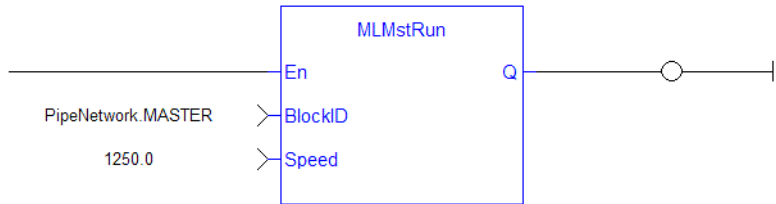
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLMstRun( PipeNetwork.MASTER, 1250.0 );
```

See Also

- "MLMstWriteDecel" (→ p. 516)
- "MLMstWriteSpeed" (→ p. 519)

3.1.13.13 MLMstStatus



 **Function** - Returns the status of the generator.

- The value returned is the state being executed by the TMP generator as it processes the various motion commands.
- Some states are transitory, others are stable until the next event takes place.

These terms are relevant to the returned values:

Term	Definition
Running	Speed is non-zero.
Stopped	Speed is 0 (zero).
Positioning	A target position has been programmed with a relative, additive, or absolute command.

3.1.13.13.1.1 Status Definitions

Status Definitions Table

Status	Definition
0	New speed programmed is entered when both: • A jog move (MLMstRun) is commanded. • The current speed is not at the commanded speed.
1	Stable state Running or Stopped is entered when both: • A jog move (MLMstRun) is commanded. • The current speed is at the commanded speed. Stable state Running or Stopped is entered when both: • A position move is programmed. • Motion is completed.
2	Speed change is entered when the current speed is greater than the commanded speed.
3	Speed reversal while positioning is entered when both: • A position move is programmed. • The distance to go requires a speed reversal.
4	Acceleration while positioning current speed is below the travel speed.
5	Constant Speed while positioning is entered when both: • A positioning move is commanded. • The current speed is at the commanded speed.
6	Deceleration while positioning is entered when both: • A positioning move is commanded. • The current speed is changing to achieve the target position at 0 (zero) speed.
7	Micro step is entered when both: • A small change in position is required. • The current speed is 0 (zero).

Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables the function or function block to be executed.
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Master Block.

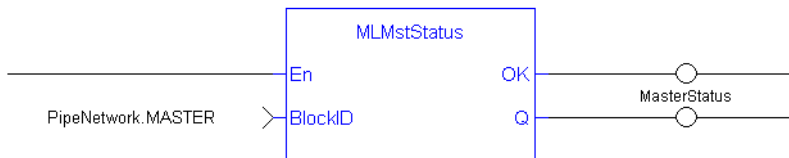
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.
Default (.Q)	DINT			Returns the status of the generator.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
MasterStatus := MLMstStatus( PipeNetwork.MASTER );
```


3.1.13.14 MLMstWriteAccel



 **Function** - Sets the acceleration of a master block.

Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables the function or function block to be executed.
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Master Block.
Acceleration	LREAL	No range	User units / sec ²	No default	- Acceleration value expressed in user position units per second squared. - See Set the Axis Block Position Units.

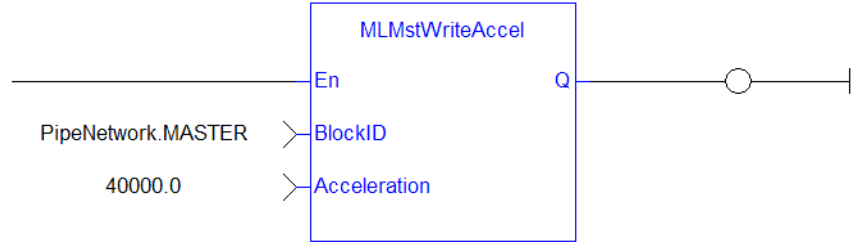
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLMstWriteAccel( PipeNetwork.MASTER, 40000.0 );
```

See Also

- "MLMstAbs" (→ p. 490)
- "MLMstRel" (→ p. 508)
- "MLMstWriteDecel" (→ p. 516)
- "MLMstWriteSpeed" (→ p. 519)

3.1.13.15 MLMstWriteDecel



 **Function** - Sets the deceleration of a master block.

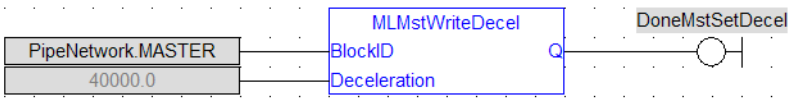
Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables the function or function block to be executed.
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Master Block.
Deceleration	LREAL	No range	User units / sec ²	No default	- Deceleration value expressed in user position units per second squared. - See Set the Axis Block Position Units.

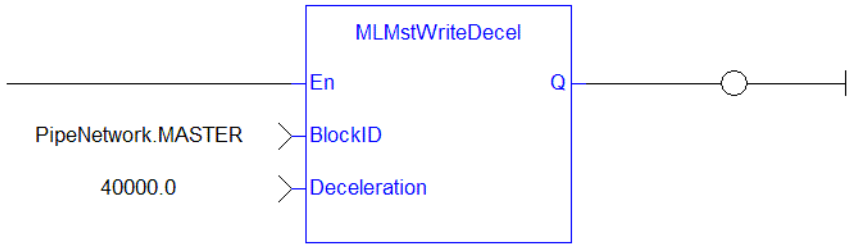
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLMstWriteDecel( PipeNetwork.MASTER, 40000.0 );
```

See Also

"MLMstWriteSpeed" (→ p. 519)

3.1.13.16 MLMstWriteInitPos



Function - Sets the initial position of a master block.

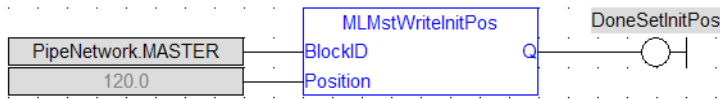
Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables the function or function block to be executed.
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Master Block.
Position	LREAL	No range	User units	No default	Initial position.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLMstWriteInitPos( PipeNetwork.MASTER, 120.0 );
```

3.1.13.17 MLMstWriteSpeed

Pipe Network



Function - Sets the speed of a master block.

- Set the speed of motion for the "MLMstAbs" (→ p. 490) and "MLMstRel" (→ p. 508) blocks.
- This function or function block does not generate any motion.

Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables the function or function block to be executed.
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Master Block.
Speed	LREAL	No range	User units / sec	No default	• Speed of the motion. • See Set the Axis Block Position Units.

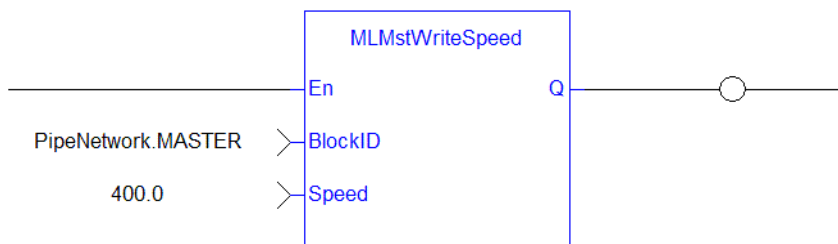
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLMstWriteSpeed( PipeNetwork.MASTER, 400.0 );
```

See Also

"MLMstWriteDecel" (→ p. 516)

3.1.14 Phaser

Names	Description
MLPhaInit	Initializes a phaser Pipe Block.
MLPhaReadActPhase	Gets the actual phase value of a phaser block.
MLPhaReadPhase	Gets the phase value of a phaser block.
MLPhaReadSlope	Gets the phase slope value of a phaser block.
MLPhaWritePhase	Sets the phase value of a phaser block.
MLPhaWriteSlope	Sets the phase slope value of a phaser block.

TIP

There is a delay when using an external encoder.

The delay is five cycles:

- 2 cycles to read the encoder from the AKD via EtherCAT,
- 1 cycle for computing,
- 2 cycles for sending the new position set point to the AKD).

This lag error is speed proportional (5 cycles * speed).

A Phaser block can be used to compensate for this lag.

3.1.14.1 Examples: Phaser Function

- Call "MLPhaWritePhase" (→ p. 529) function to modify the Phase value. (Figure 4-54)
- Call "MLPhaWriteSlope" (→ p. 531) to modify the rate of change of phase or slope.
 - This is applied when the Phase value is changed. (Figure 4-55)

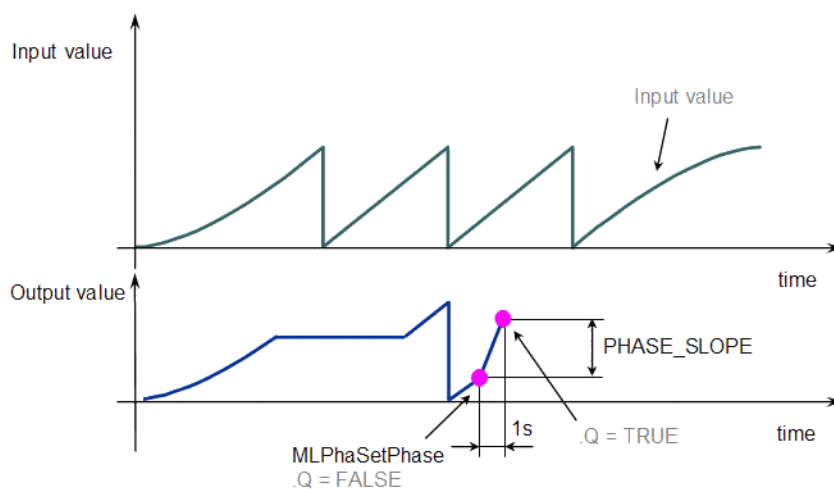


Figure 4-54: Changed Phase value

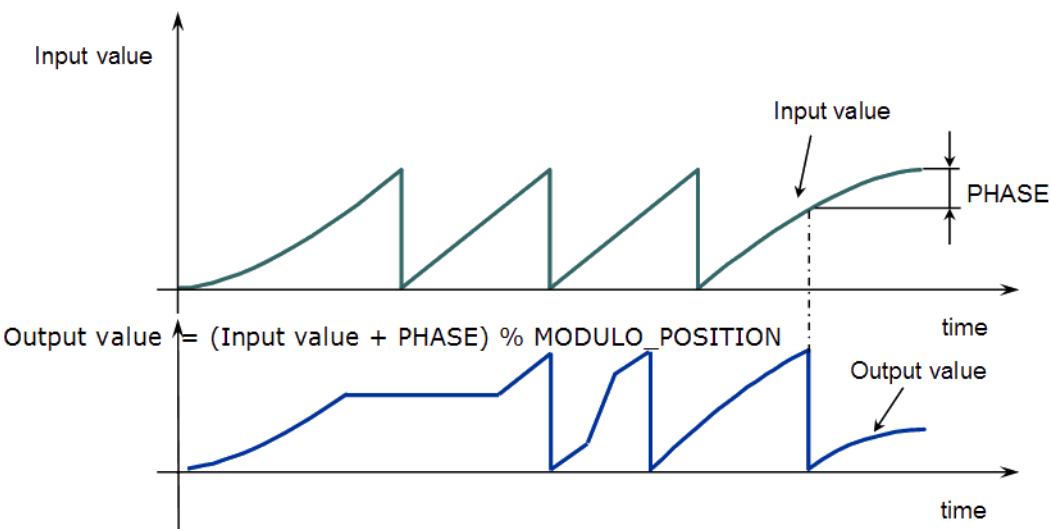


Figure 4-55: Changed Slope value

NOTE

% MODULO_POSITION is in the equation to take into account the modulo (periodicity) of the value.

3.1.14.2 MLPhalnit

Pipe Network



Function - Initializes a phaser Pipe Block.

- Returns TRUE if the function succeeded.
- This function block is automatically called by the Pipe Network(MLPN_CREATE_OBJECTS) if a Phaser block is added to the Pipe Network.
 - User-defined settings are entered in the Pipe Blocks Properties screen.
- The Phaser Pipe Block is assigned a Name, OUTPUT_PERIOD, PHASE, PHASE_SLOPE_TYPE, and STANDBY_VALUE.

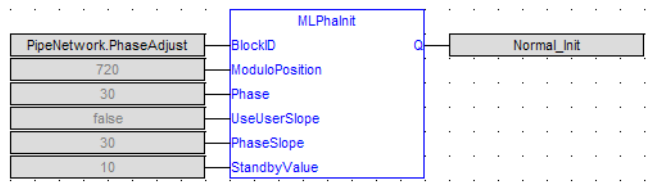
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a phaser function block in the Pipe Network.
ModuloPosition	LREAL	0 to 1.0x10 ³⁰⁰	User units	No default	• Rollover position of the phaser block output signal. • 0 (zero) indicates the output signal does not use the modulo operation.
Phase	LREAL	No range	N/A	No default	The amount of phase adjustment. • The phase value can be negative. • A negative phase value is subtracted from the input value. • Phase is expressed in User units.
UseUserSlope	BOOL	FALSE, TRUE	N/A	No default	• Determines if the maximum slope or user-defined slope is used. • FALSE indicates the phaser block should use maximum slope. • TRUE indicates the phaser block should use user-defined slope.
PhaseSlope	LREAL	No range	User units / sec	No default	• User-defined rate at which phase changes are implemented. • Expressed in user logical units per second. • A slow rate parameter is used to limit the implementation of step changes of phase.
StandbyValue	LREAL	No range	N/A	No default	• The output signal value assumed by the phaser output when the phaser is in a stopped condition. • Expressed in user logical units. • This is the output from the Phaser Block until the "MLPhaWritePhase" (→ p. 529) function is executed.

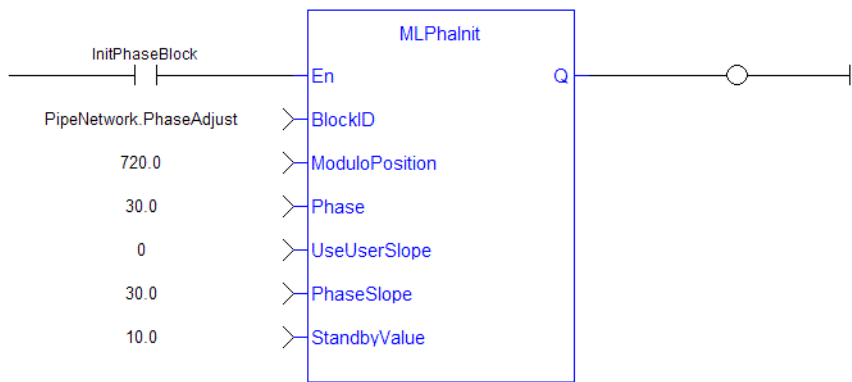
Outputs

Input	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the function or function block is successfully executing. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Initialize a Phaser Pipe Block named "PhaseAdjust" to a Modulo of 720,  
phase offset value of 30, use the Max Slope  
MLPhaInit( PipeNetwork.PhaseAdjust , 720, 30, false, 30 , 10 );
```

See Also

- "MLPhaReadActPhase" (→ p. 525)
- "MLPhaReadPhase" (→ p. 527)
- "MLPhaReadSlope" (→ p. 528)
- "MLPhaWritePhase" (→ p. 529)
- "MLPhaWriteSlope" (→ p. 531)

3.1.14.3 MLPhaReadActPhase



Function - Gets the actual phase value of a phaser block.

- If a PHASE_SLOPE_USER value is used, the new phase isn't set immediately.
 - See "MLPhaWritePhase" (→ p. 529).
 - The phase is ramped with the slope value from the old phase value to the new phase value.
 - MLPhaReadActPhase returns this ramping value.
 - See "MLPhaReadSlope" (→ p. 528) and "MLPhaWriteSlope" (→ p. 531).
- "MLPhaReadPhase" (→ p. 527) returns the new value and this also when the phaser is still ramping.
 - If using Max Slope means no ramping.
 - **MLPhaReadActPhase** and **MLPhaReadPhase** always returns the same value.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable					
BlockID	DINT	1 to 1024	N/A	No default	ID name of a phaser function block in the Pipe Network.

Outputs

Input	Data Type	Range	Unit	Description
OK				
Phase	LREAL			

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

Not available.

See Also

- "MLPhalnit" (→ p. 523)
- "MLPhaReadPhase" (→ p. 527)
- "MLPhaReadSlope" (→ p. 528)
- "MLPhaWritePhase" (→ p. 529)
- "MLPhaWriteSlope" (→ p. 531)

3.1.14.4 MLPhaReadPhase

Pipe Network ✓



Function - Gets the phase value of a phaser block.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a phaser function block in the Pipe Network.

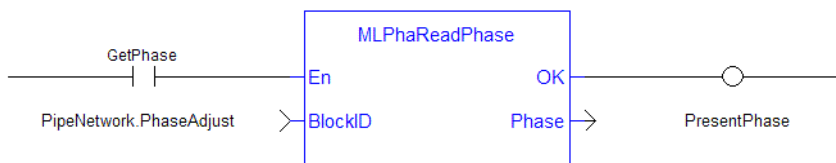
Outputs

Input	Data Type	Range	Unit	Description
Phase	LREAL			

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
PresentPhase := MLPhaReadPhase( PipeNetwork.PhaseAdjust );
```

See Also

- "MLPhaInit" (→ p. 523)
- "MLPhaReadActPhase" (→ p. 525)
- "MLPhaReadSlope" (→ p. 528)
- "MLPhaWritePhase" (→ p. 529)
- "MLPhaWriteSlope" (→ p. 531)

3.1.14.5 MLPhaReadSlope



Function - Gets the phase slope value of a phaser block.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a phaser function block in the Pipe Network.

Outputs

Input	Data Type	Range	Unit	Description
Slope	LREAL		User units / sec	Present slope value. - Value defined in the setup of a phaser block in a Pipe Network. - Depending on the phase slope type setting, it is the VALUE in PHASE_SLOPE_USER, PHASE, or the Max Slope.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
PresentSlope :=MLPhaReadSlope( PipeNetwork.PhaseAdjust );
```

See Also

- "MLPhaInit" (→ p. 523)
- "MLPhaReadActPhase" (→ p. 525)
- "MLPhaReadPhase" (→ p. 527)
- "MLPhaWritePhase" (→ p. 529)
- "MLPhaWriteSlope" (→ p. 531)

3.1.14.6 MLPhaWritePhase



Function - Sets the phase value of a phaser block.

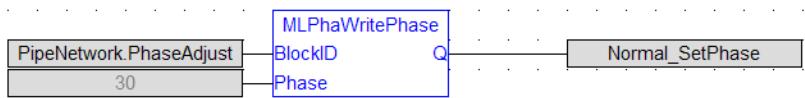
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a phaser function block in the Pipe Network.
Phase	LREAL	No range	User units / sec		Phase value. - Value defined in the setup of a phaser block in a Pipe Network. - It is in the PHASE field in the parameter tab.

Outputs

Input	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE if the function or function block is successfully executing.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLPhaWritePhase( PipeNetwork.PhaseAdjust , 30 );
```


See Also

- "MLPhaInit" (→ p. 523)
- "MLPhaReadActPhase" (→ p. 525)
- "MLPhaReadPhase" (→ p. 527)
- "MLPhaReadSlope" (→ p. 528)
- "MLPhaWriteSlope" (→ p. 531)

3.1.14.7 MLPhaWriteSlope

Pipe Network ✓



Function - Sets the phase slope value of a phaser block.

Returns TRUE if the function succeeded.

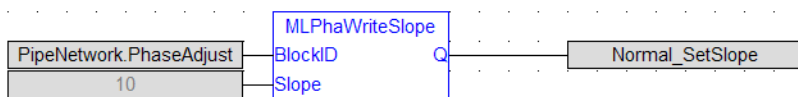
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a phaser function block in the Pipe Network.
Slope	LREAL	No range	User units / sec		Sets the slope of the phase adjustment. - Value defined in the setup of a phaser block in a Pipe Network. - Depending on the phase slope type setting, it is the VALUE in PHASE_SLOPE_USER, PHASE, or the Max Slope.

Outputs

Input	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE if the function or function block is successfully executing.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLPhaWriteSlope( PipeNetwork.PhaseAdjust , 10 );
```

See Also

- "MLPhaInit" (→ p. 523)
- "MLPhaReadActPhase" (→ p. 525)
- "MLPhaReadPhase" (→ p. 527)
- "MLPhaReadSlope" (→ p. 528)
- "MLPhaWritePhase" (→ p. 529)

3.1.15 PMP

Name	Description
MLPmpAbs	Moves to an absolute position using a parabolic acceleration profile.
MLPmpForcePos	Forces the position of a PMP block to a specified position.
MLPmpInit	Initializes a PMP block for use in a PLC program.
MLPmpReadAccel	Gets the Acceleration parameter of a PMP block.
MLPmpReadFstSpd	Gets the FirstTravelSpeed parameter of a PMP block.
MLPmpReadInitPos	Gets the Initial Position parameter of a PMP block.
MLPmpReadJerk	Gets the Jerk parameter of a PMP block.
MLPmpReadLstSpd	Gets the LastTravelSpeed parameter of a PMP block.
MLPmpRel	Used to perform two subsequent relative moves.
MLPmpRun	Jog the generator at the specified speed.
MLPmpStatus	Returns the status of the PMP block generator.
MLPmpWriteAccel	Sets the Acceleration parameter of a PMP block.
MLPmpWriteFstSpd	Sets the FirstTravelSpeed parameter of a PMP block.
MLPmpWriteJerk	Sets the Jerk parameter of a PMP block.
MLPmpWriteLstSpd	Sets the LastTravelSpeed parameter of a PMP block.

3.1.15.1 MLPmpAbs

Pipe Network ✓

 **Function** - Moves to an absolute position using a parabolic acceleration profile.

- The FIRST_TRAVEL_SPEED is used as the velocity for the motion.
- JERK determines the level of parabolic acceleration.
- Returns TRUE if the function succeeded.

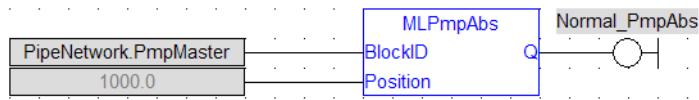
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a PMP function block in the Pipe Network.
Position	LREAL	No range	User units	No default	Absolute position of the motor/load to be at after this function block is complete.

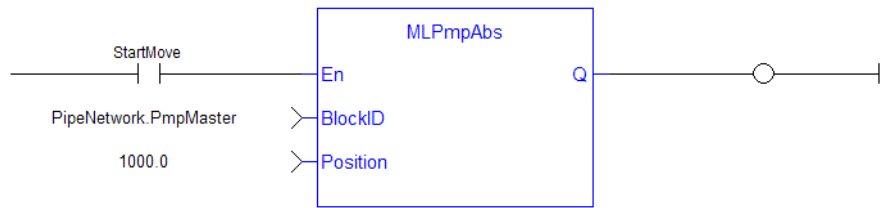
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the function or function block is successfully executing. •

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLPmpAbs ( PipeNetwork.PmpMaster, 1000.0 ) ;
```

See Also

- "MLPmpWriteAccel" (→ p. 555)
- "MLPmpWriteJerk" (→ p. 559)
- "MLPmpWriteFstSpd" (→ p. 557)

3.1.15.2 MLPmpForcePos



Function - Forces the position of a PMP block to a specified position.

- This block can only be executed when motion is **NOT** occurring.
- It can be used to force the PMP starting position to the values to start motion from.

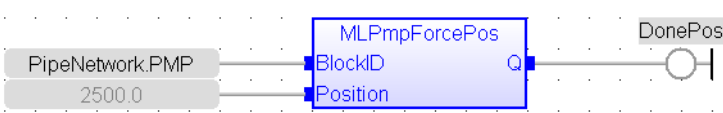
Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables execution.
BlockID	DINT	1 to 1024	N/A	No default	ID name of a PMP function block in the Pipe Network.
Position	LREAL	No range	User units	No default	Defines the PMP starting position when the motion starts.

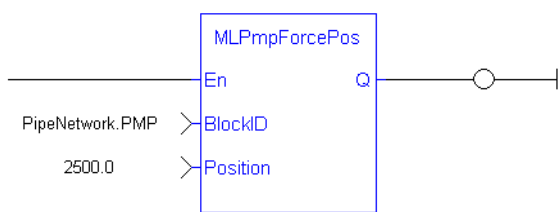
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



NOTE

- A positive transition contact is required to start the FB.
- See FFLD Contacts.

IL Language Example

Not available.

ST Language Example

```
MLPmpForcePos( PipeNetwork.PMP, 2500.0 );
```

See Also

"MLPmpReadInitPos" (→ p. 545)

3.1.15.3 MLPmplnit

Pipe Network



Function - Initializes a PMP block for use in a PLC program.

- Initializes a PMP object (Parabolic Motion Profile generator) with user-defined settings.
- This function block is automatically called by the function Pipe Network(MLPN_CREATE_OBJECTS) if a PMP block is added to the Pipe Network.
 - User-defined settings are entered in the Pipe Blocks Properties screen.
- The PMP pipe block is assigned a Name, SAMPLING_PERIOD, MODULO_POSITION, FIRST_TRAVEL_SPEED, LAST_TRAVEL_SPEED, ACCELERATION, JERK, and INITIAL Position.
- Some of these parameters can be changes in an application program using other MLPmp function blocks.
- An **MLPmpRel** function block is used to make a bi-directional motion.
 - First movement in one direction, then a return motion back to the initial position.
- An **MLPmpAbs** function block is used to move one direction to an absolute position.

NOTE

- PMP objects are normally created in the Pipe Network using the graphical engine.
- You do not have to add **MLPmplnit** functions to their programs.
- Parameters are entered directly in pop-up windows; the code is automatically added to the current project.

Inputs

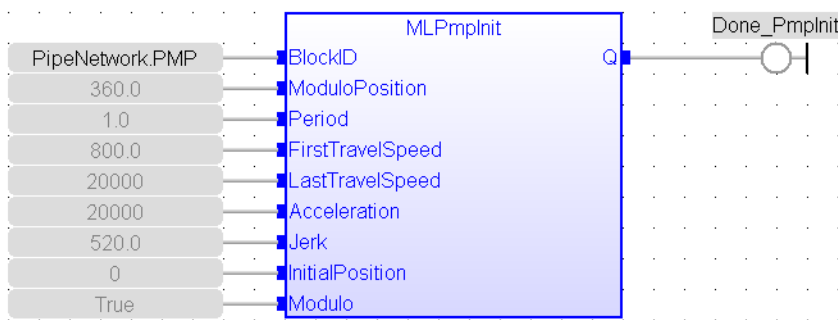
Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a PMP function block in the Pipe Network.
ModuloPosition	LREAL	No range	User units	No default	• Modulo Position for cyclic motion systems expressed in user logical units. • Position Rollover Value.
Period	LREAL	No range	User units	No default	• Sampling period of the generator expressed according to the update cycle. • Example: 2.0 means the sampling is done once every 2 cycles.
FirstTravelSpeed	LREAL	No range	User units / sec	No default	First travel speed of the motion.
LastTravelSpeed	LREAL	No range	User units / sec	No default	Last travel speed of the motion.
Acceleration	LREAL	No range	User units / sec ²	No default	Acceleration of the PMP block motion.
Jerk	LREAL	No range	User units / sec ³	No default	Jerk.

Input	Data Type	Range	Unit	Default	Description
InitialPosition	LREAL	No range	User units	No default	Initial position of the PMP block when the Pipe Network is started.
Modulo	BOOL	FALSE, TRUE	N/A	No default	The available modes are: • TRUE = Modulo. • FALSE = No modulo.

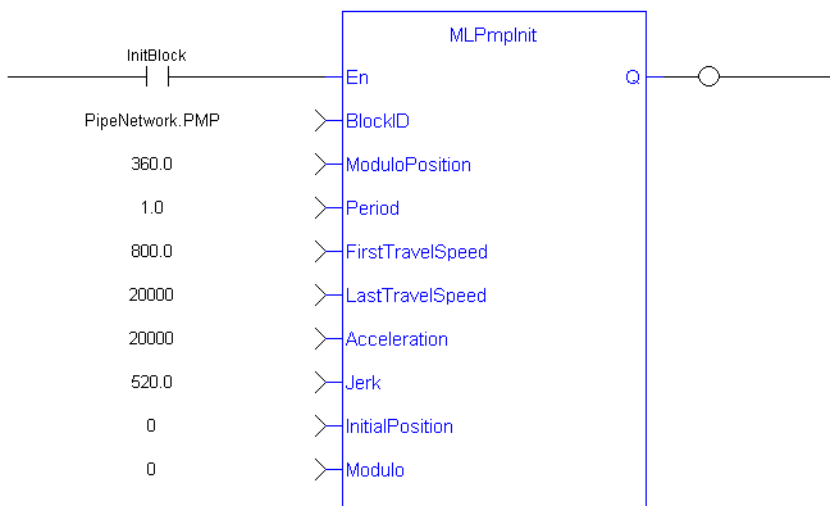
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the function or function block is successfully executing. •

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Initialize a PMP Pipe Block named "PMP" to a modulo roll over of 360,  
motion generator sample period of 1,First Travel Speed of 800.0, Second  
Travel Speed of 20000.0, Accel of 20000.0,Jerk of 520.0, Initial position of  
0.0  
MLPmpInit( PipeNetwork.Pmp , 360.0, 1.0, 800.0, 20000.0, 20000.0, 520.0, 0,  
true ) ;
```

See Also

- "MLPmpReadAccel" (→ p. 541)
- "MLPmpReadFstSpd" (→ p. 543)
- "MLPmpReadInitPos" (→ p. 545)
- "MLPmpReadJerk" (→ p. 546)
- "MLPmpReadLstSpd" (→ p. 547)

3.1.15.4 MLPmpReadAccel

Pipe Network ✓



Function - Gets the Acceleration parameter of a PMP block.

Used in both the **MLPmpAbs** and **MLPmpRel** function blocks.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a PMP function block in the Pipe Network.

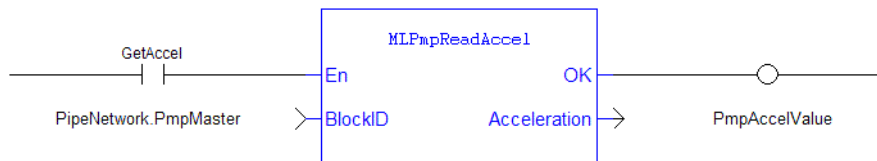
Outputs

Output	Data Type	Range	Unit	Description
Acceleration	LREAL		User units / sec^2	Present Acceleration of the PMP Pipe Network function block. - Value defined PMP block when creating a Pipe Network. - The value is entered in the ACCELERATION field in the Parameter tab.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
PmpAccelValue := MLPmpReadAccel ( PipeNetwork.PmpMaster ) ;
```

See Also

- "MLPmpReadFstSpd" (→ p. 543)
- "MLPmpReadInitPos" (→ p. 545)
- "MLPmpReadJerk" (→ p. 546)
- "MLPmpReadLstSpd" (→ p. 547)

3.1.15.5 MLPmpReadFstSpd

Pipe Network ✓



Function - Gets the FirstTravelSpeed parameter of a PMP block.

- This parameter is used as:
 - The first of two speeds in an **MLPmpRel** function block.
 - The speed in an **MLPmpAbs** function block.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a PMP function block in the Pipe Network.

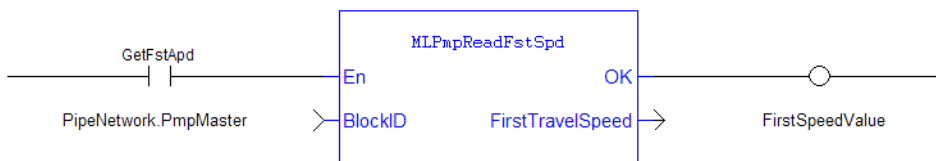
Outputs

Output	Data Type	Range	Unit	Description
FirstTravelSpeed	LREAL		User units / sec	Present first travel speed of the PMP Pipe Network function block. - Value defined in the setup of a PMP block in a Pipe Network. - The value is entered in the FIRST_TRAVEL_SPEED field in the Parameter tab.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
FirstSpeedValue := MLPmpReadFstSpd( PipeNetwork.PmpMaster ) ;
```

See Also

- "MLPmpReadAccel" (→ p. 541)
- MLPmpReadFstSpd
- "MLPmpReadInitPos" (→ p. 545)
- "MLPmpReadJerk" (→ p. 546)
- "MLPmpReadLstSpd" (→ p. 547)
- "MLPmpWriteLstSpd" (→ p. 561)

3.1.15.6 MLPmpReadInitPos

Pipe Network



Function - Gets the Initial Position parameter of a PMP block.

- This value is the position the PMP block starts at when the Pipe Network is enabled.
- Can be set when adding a PMP block to a Pipe Network and defining the parameters for that block.

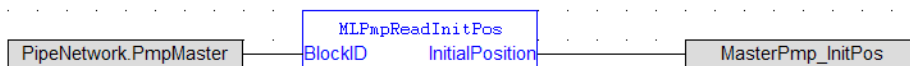
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a PMP function block in the Pipe Network.

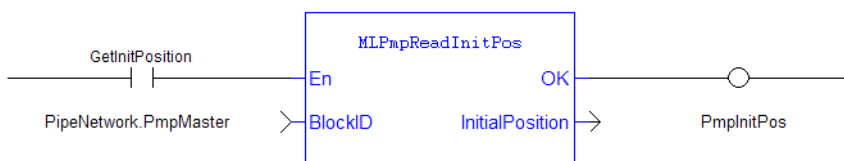
Outputs

Output	Data Type	Range	Unit	Description
InitialPosition	LREAL		User units	Present Initial Position of the PMP Pipe Network function block. • Value defined in the setup of a PMP block in a Pipe Network. • The value is entered in the INITIAL_POSITION field in the Parameter tab.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
PmpInitPos := MLPmpReadInitPos( PipeNetwork.PmpMaster ) ;
```

See Also

"MLPmpInit" (→ p. 538)

3.1.15.7 MLPmpReadJerk



Function - Gets the Jerk parameter of a PMP block.
Used in both the **MLPmpAbs** and **MLPmpRel** function blocks.

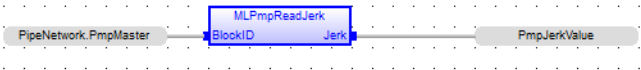
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a PMP function block in the Pipe Network.

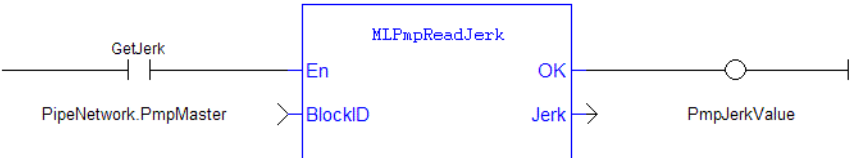
Outputs

Output	Data Type	Range	Unit	Description
Jerk	LREAL		User units / sec ³	Jerk of the PMP Pipe Network function block. - Value defined in the setup of a PMP block in a Pipe Network. - The value is entered in the JERK field in the Parameter tab.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
PmpJerkValue := MLPmpReadJerk( PipeNetwork.PmpMaster ) ;
```

3.1.15.8 MLPmpReadLstSpd

Pipe Network ✓



Function - Gets the LastTravelSpeed parameter of a PMP block.

Used in the **MLPmpRel** function block.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a PMP function block in the Pipe Network.

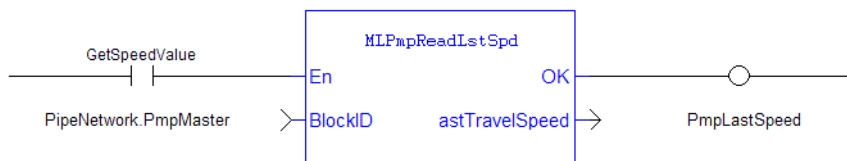
Outputs

Output	Data Type	Range	Unit	Description
LastTravelSpeed	LREAL		User units / sec	Last Travel Speed of the PMP Pipe Network function block. - Value defined in the setup of a PMP block in a Pipe Network. - The value is entered in the LAST_TRAVEL_SPEED field in the Parameter tab.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
PmpLastSpeed := MLPmpReadLstSpd( PipeNetwork.PmpMaster ) ;
```

See Also

- "MLPmpReadAccel" (→ p. 541)
- "MLPmpReadFstSpd" (→ p. 543)
- "MLPmpReadInitPos" (→ p. 545)
- "MLPmpReadJerk" (→ p. 546)

3.1.15.9 MLPmpRel

Pipe Network ✓



Function - Used to perform two subsequent relative moves.

- Using the **MLPmpRel** function block, the PMP Generator produces forward-backward motions with a non-stop, jerk-free transition through zero speed.
 - See Figure 4-56.
 - This feature is useful for linear axes which must move back and forward without any pause at one end.
- This function is used to do a single relative move, ending in zero speed, by setting the **DeltaSecond** argument to zero (0.0).
 - If it is done, for the controlling speed to be the first move, the **Last_Travel_Speed** parameter must be set equal to or greater than the **First_Travel_Speed** parameter.
- The slower of the two speeds is utilized to optimize the S-curve behavior for the move whether it is a 2 or 1 delta move.
- If the **DeltaSecond** argument is non-zero, it must have the opposite sign than the sign of the **DeltaFirst** argument.

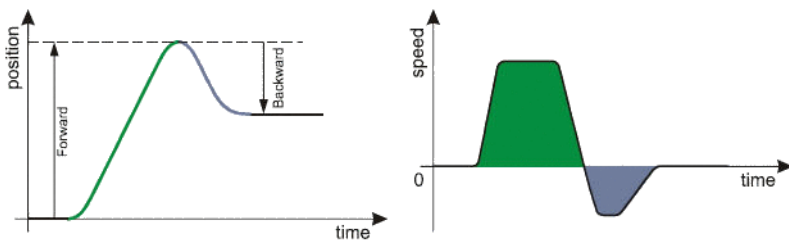


Figure 4-56: PMP Generator Forward and Backward Motion Profile

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a PMP function block in the Pipe Network.
DeltaFirst	LREAL	No range	User units	No default	Length of the first move. - Value defined in the setup of a PMP block in a Pipe Network. - The value is entered in the FIRST_TRAVEL_SPEED field in the Parameter tab.
DeltaSecond	LREAL	No range	User units	No default	Length of the second (return) move. - Value defined in the setup of a PMP block in a Pipe Network. - The value is entered in the LAST_TRAVEL_SPEED field in the Parameter tab.

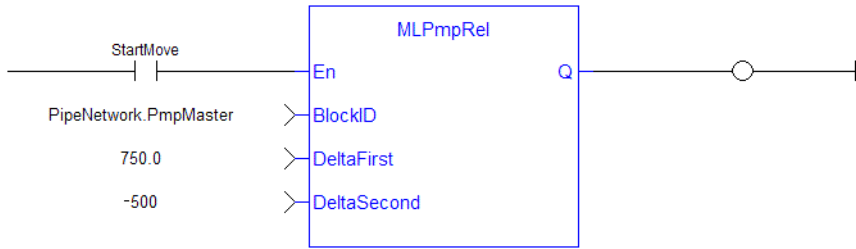
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the function or function block successfully completed. •

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Execute a Relative move on a PMP Block name "PmpMaster" with a First Travel
Speed of 750.0, Second Travel Speed of -500,
MLPmpRel( PipeNetwork.PmpMaster, 750 , -500);
```

See Also

- "MLPmpWriteAccel" (→ p. 555)
- "MLPmpWriteFstSpd" (→ p. 557)
- "MLPmpWriteJerk" (→ p. 559)
- "MLPmpWriteLstSpd" (→ p. 561)

3.1.15.10 MLPmpRun

Pipe Network



Function - Jog the generator at the specified speed.

Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	- Enables execution. - Is only recognized if the PMP generator is Idle or at constant velocity as determined from the "MLPmpStatus" (→ p. 553) function.
BlockID	DINT	1 to 1024	N/A	No default	ID name of a PMP function block in the Pipe Network.
Speed	LREAL	No range	User units / sec	No default	The designated rate to Jog at. - If the speed is 0.0 User units / sec, the PMP block decelerates to zero speed and switches to the Idle state (0). - See Set the Axis Block Position Units.

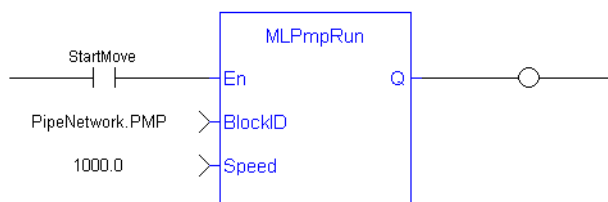
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE if the function or function block successfully completed.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Execute a Relative move on a PMP Block name "PmpMaster" with a Jog speed of  
1000.0  
MLPmpRun( PipeNetwork.PmpMaster, 1000.0 ) ;
```

See Also

"MLPmpStatus" (→ p. 553)

3.1.15.11 MLPmpStatus

Pipe Network ✓



Function - Returns the status of the PMP block generator.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enables execution.
BlockID	DINT	1 to 1024	N/A	No default	ID name of a PMP function block in the Pipe Network.

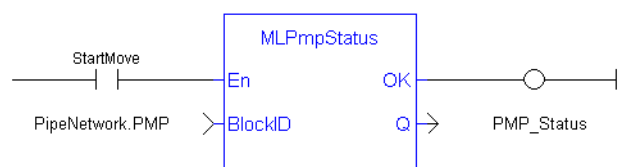
Outputs

Output	Data Type	Range	Unit	Description	
OK	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •	
Default (.Q)	DINT		N/A	Returns the status of the PMP block generator.	
				Value	Description
				0	• Indicates the PMP block is idle. • No command is currently running in the generator. It can be used to determine that a previous move is complete.
				1	Indicates the PMP block is either accelerating to a position or speed, or is decelerating to a position or speed.
				2	Indicates the PMP block is running at a constant speed.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
PMP_Status := MLPmpStatus ( PipeNetwork.PmpMaster ) ;
Done :=TRUE;
```

3.1.15.12 MLPmpWriteAccel

Pipe Network



Function - Sets the Acceleration parameter of a PMP block.

- Can be set when adding a PMP block to a Pipe Network and defining the parameters for that block.
- Returns TRUE if the function succeeded.

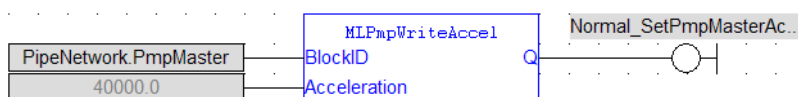
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a PMP function block in the Pipe Network.
Acceleration	LREAL	> 0 (zero)	User units / sec ²	No default	• Acceleration value. • The value is entered in the ACCELERATION field in the Parameter tab.

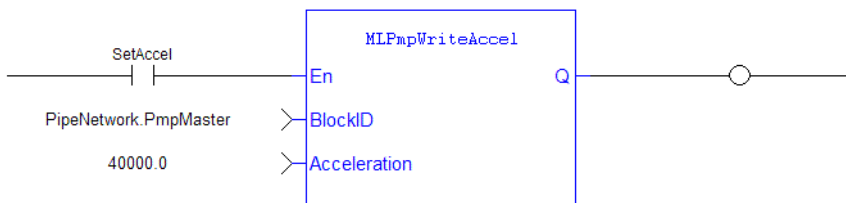
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the function or function block is successfully executing. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLPmpWriteAccel( PipeNetwork.PmpMaster, 40000.0 ) ;
```

See Also

- "MLPmpAbs" (→ p. 534)
- "MLPmpRel" (→ p. 549)
- "MLPmpWriteFstSpd" (→ p. 557)
- "MLPmpWriteJerk" (→ p. 559)
- "MLPmpWriteLstSpd" (→ p. 561)

3.1.15.13 MLPmpWriteFstSpd

Pipe Network



Function - Sets the FirstTravelSpeed parameter of a PMP block.

- Can be set when adding a PMP block to a Pipe Network and defining the parameters for that block.
- Returns TRUE if the function succeeded.

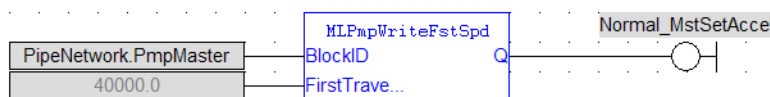
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a PMP function block in the Pipe Network.
FirstTravelSpeed	LREAL	> 0 (zero)	User units / sec	No default	First Travel Speed value. • Value defined in the setup of a PMP block in a Pipe Network. • The value is entered in the FIRST_TRAVEL_SPEED field in the Parameter tab.

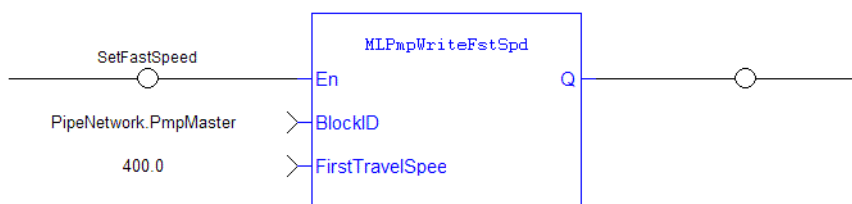
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the function or function block is successfully executing. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLPmpWriteFstSpd( PipeNetwork.PmpMaster, 300.0 ) ;
```

See Also

- "MLPmpAbs" (→ p. 534)
- "MLPmpRel" (→ p. 549)
- "MLPmpWriteAccel" (→ p. 555)
- "MLPmpWriteJerk" (→ p. 559)
- "MLPmpWriteLstSpd" (→ p. 561)

3.1.15.14 MLPmpWriteJerk

Pipe Network



Function - Sets the Jerk parameter of a PMP block.

- Can be set when adding a PMP block to a Pipe Network and defining the parameters for that block.
- Returns TRUE if the function succeeded.

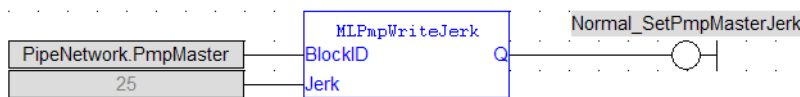
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a PMP function block in the Pipe Network.
Jerk	LREAL	> 0 (zero)	User units / sec ³	No default	Jerk value. • Value defined in the setup of a PMP block in a Pipe Network. • The value is entered in the JERK field in the Parameter tab.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the function or function block is successfully executing. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLPmpWriteJerk( PipeNetwork.PmpMaster, 15.0 ) ;
```

See Also

- "MLPmpAbs" (→ p. 534)
- "MLPmpReadJerk" (→ p. 546)
- "MLPmpRel" (→ p. 549)
- "MLPmpWriteAccel" (→ p. 555)
- "MLPmpWriteFstSpd" (→ p. 557)
- "MLPmpWriteLstSpd" (→ p. 561)

3.1.15.15 MLPmpWriteLstSpd

Pipe Network



Function - Sets the LastTravelSpeed parameter of a PMP block.

- Can be set when adding a PMP block to a Pipe Network and defining the parameters for that block.
- Returns TRUE if the function succeeded.

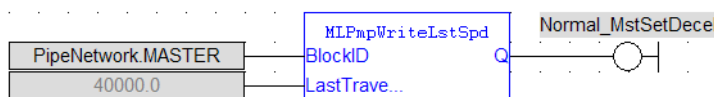
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of a PMP function block in the Pipe Network.
LastTravelSpeed	LREAL	> 0 (zero)	User units / sec	No default	Last Travel Speed value. • Value defined in the setup of a PMP block in a Pipe Network. • The value is entered in the LAST_TRAVEL_SPEED field in the Parameter tab.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the function or function block is successfully executing. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLPmpWriteLstSpd( PipeNetwork.PmpMaster, 650 ) ;
```

See Also

- "MLPmpAbs" (→ p. 534)
- "MLPmpReadLstSpd" (→ p. 547)
- "MLPmpRel" (→ p. 549)
- "MLPmpWriteAccel" (→ p. 555)
- "MLPmpWriteFstSpd" (→ p. 557)
- "MLPmpWriteJerk" (→ p. 559)

3.1.16 Sampler

Name	Description
MLSmpConECAT	Connects a sampler block to the specified CoE object in a PDO.
MLSmpConnect	Connects a sampler block to a pipe network axis or pipe block.
MLSmpConPLCAxis	Connects a sampler block to a specific PLCopen axis variable.
MLSmpConPNAxis	Connects a sampler block to a specific Pipe Network axis variable.
MLSmpInit	Initializes a sampler object.

TIP

There is a delay when using an external encoder.
 The delay is five cycles:
 2 cycles to read the encoder from the AKD via EtherCAT,
 1 cycle for computing,
 2 cycles for sending the new position set point to the AKD).
 This lag error is speed proportional (5 cycles * speed).
 A Phaser block can be used to compensate for this lag.

3.1.16.1 MLSmpConECAT



Function - Connects a sampler block to the specified CoE object in a PDO.

- The CoE object must be included in a PDO for the specified EtherCAT device.
 - Use any EtherCAT data source as input for the specified sampler block.
 - If the device providing the CoE Object for the Sampler block is removed:
 - The CoE object value becomes 0 (zero) if the EtherCAT device providing the CoE object for the Sampler block is removed using the "ECATDeviceAction" (→ p. 42) function block.
 - The CoE object automatically connects to the sampler block when the device is reconnected to the EtherCAT network.
 - This function can be programmed from within the Pipe Network block.
- Right-click the block and click **Properties**.

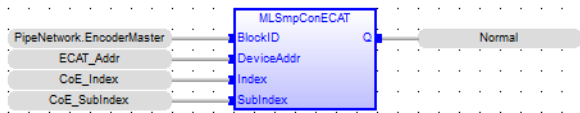
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID Name of the SMP function in the Pipe Network.
DeviceAddr	INT	No range	N/A	No default	The EtherCAT address of the slave device. • The first node usually has the value 1001. • Alternately, use the members of the EtherCAT structure to specify a device's address.
Index	UINT	No range	N/A	No default	The CoE index of the object to be connected with the Sampler block.
SubIndex	USINT	No range	N/A	No default	The CoE sub-index of the object to be connected with the Sampler block.

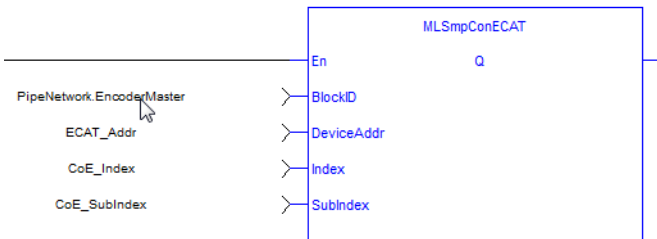
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Function block is operational. •

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Connect a Sampler pipe block named "EncoderMaster" to an ECAT Index Object
defined by variable "CoE_SubIndex" with the SubIndex defined by variable
"CoE_SubIndex", from a device with Ethercat Address defined by "ECAT_Addr"
MLSmpConECAT(PipeNetwork.EncoderMaster, ECAT_Addr, CoE_Index, CoE_SubIndex );
```

See Also

- "MLSmpConnect" (→ p. 566)
- "MLSmpInit" (→ p. 572)

3.1.16.2 MLSmpConnect



 **Function** - Connects a sampler block to a pipe network axis or pipe block.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID Name of the SMP function in the Pipe Network.
PipeBlockID	DINT	1 to 1024	N/A	No default	ID Name of the pipe block the sampler is connected to.

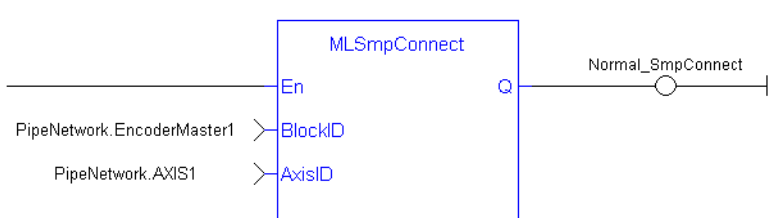
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the Sampler is connected. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Connect a Sampler pipe block named "EncoderMaster1" to a PipeNetwork Axis
block named AXIS1
MLSmpConnect( PipeNetwork.EncoderMaster1, PipeNetwork.AXIS1 ) ;
```

See Also

- "MLSmpConECAT" (→ p. 564)
- "MLSmpConPLCAxis" (→ p. 568)
- "MLSmpConPNAxis" (→ p. 570)
- "MLSmpInit" (→ p. 572)

3.1.16.3 MLSmpConPLCAxis



 **Function** - Connects a sampler block to a specific PLCopen axis variable.

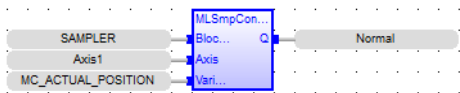
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID Name of the SMP function in the Pipe Network.
AxisID	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function. See Modify PLCopen Axis for more information.
Variable	UINT	N/A Use available macros.	N/A	No default	Variable to be connected to. Use one of these Internal Defines: - MC_ACTUAL_POSITION - MC_COMMAND_POSITION - MC_NORMAL_COMMAND_POSITION - MC_PHASE_COMMAND_POSITION - MC_SUPERIMPOSED_COMMAND_POSITION

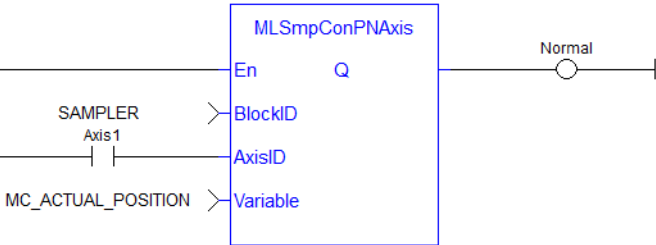
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Function block is operational.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Connect a Sampler pipe block named "SAMPLER" to a variable named "MC_
ACTUAL_POSITION" from a PLCOpen Axis named Axis1.
MLSmpConPLCAxis( PipeNetwork.SAMPLER, Axis1, MC_ACTUAL_POSITION);
```


3.1.16.4 MLSmpConPNAxis



 **Function** - Connects a sampler block to a specific Pipe Network axis variable.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID Name of the SMP function in the Pipe Network.
AxisID	DINT	1 to 256	N/A	No default	ID Name of the axis the sampler is connected to.
Variable	UINT	N/A Use available macros.	N/A	No default	Variable to be connected to. Use one of these Internal Defines: - ML_ACTUAL_POSITION - ML_ACTUAL_TORQUE - ML_ACTUAL_VELOCITY - ML_CURRENT_POSITION - ML_FEEDBACK_POSITION - ML_FOLLOWING_ERROR - ML_GENERATOR_POSITION - ML_PIPE_POSITION - ML_REFERENCE_POSITION

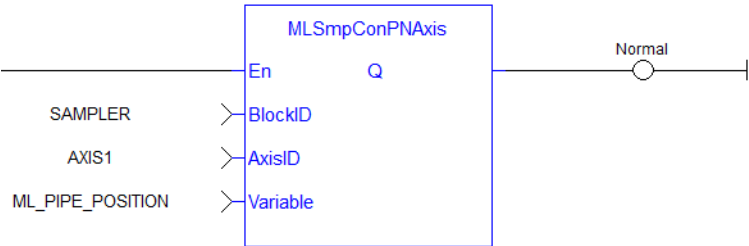
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Function block is operational.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

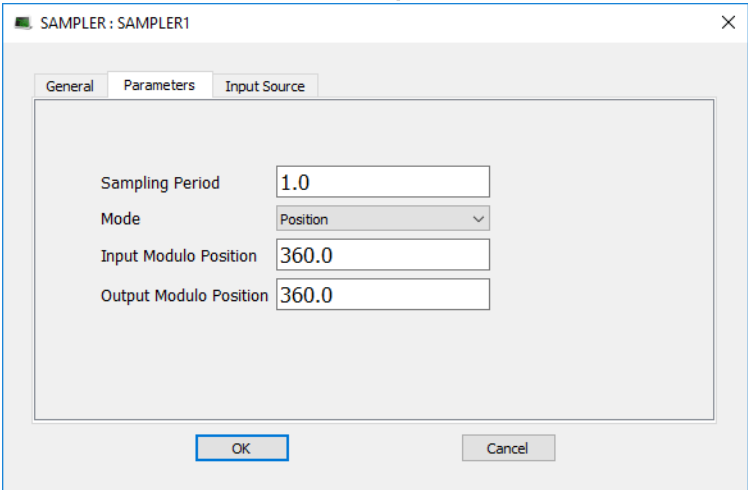
```
//Connect a sampler block named "SAMPLER" to a variable named ML_PIPE_
POSITION from a Pipe Network Axis named PipeNetwork.AXIS1
MLSmpConPNAxis( PipeNetwork.SAMPLER , PipeNetwork.AXIS1, ML_PIPE_POSITION );
```

3.1.16.5 MLsmpInit

Pipe Network ✓

 **Function** - Initializes a sampler object.

- This sampler block is used to periodically sample and place into a pipe some output of a source object.
 - The sampled output can be the POSITION or SPEED of a source object measured by a resolver, an encoder, or some other types of sensor.
 - The sampler implements the logical connection between:
 - An encoder on a physical master axis (the source object).
 - One or more pipes.
 - It performs the function of periodically sampling the source and placing the sampled values into the pipe.
 - This function block is automatically called by the Function PipeNetwork(MLPN_CREATE_OBJECTS) if an SMP Block is added to the Pipe Network, with user-defined settings entered in the Pipe Blocks Properties screen.
 - The SMP Pipe Block is assigned a Name, SAMPLING_PERIOD, MODE, INPUT_VALUE_PERIOD, and OUTPUT_VALUE_PERIOD.
 - This function can be programmed from within the Pipe Network block.
- Right-click the block and click **Properties**.

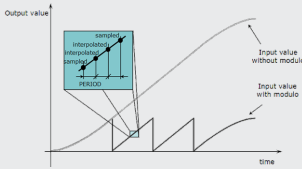
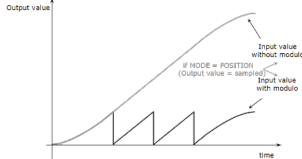
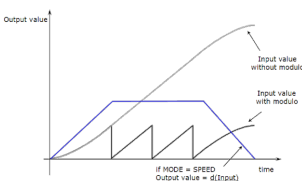


- To offset the Sampler Block Output Position in the Pipe Network either:
 - Place a Phaser Block (and write "MLPhaWritePhase" (→ p. 529) in the application code).
 - Place a Gear Block (and write "MLGearWriteOff" (→ p. 475)) after the Sampler Block.



Use AKD Secondary Feedback	Use AKD2G Additional Feedback
• The Sampler can be connected to the secondary feedback on the AKD using "MLSmpConPNAxis" (→ p. 570). • The scaling for the AKD Secondary Feedback is setup using AKD Parameters: DRV.HANDWHEEL and FB2.ENCRES . • The feedback signal comes through EtherCAT in object 0x2050. • The scaling for this position signal is 0 to 4294967296 = 0 to FB2.ENCRES. • Object 0x2050 rolls over to 0 when reaching 4294967296.	• The Sampler can be connected to the additional feedback (1-5) on the AKD2G using "MLSmpConPNAxis" (→ p. 570). • See the AKD2GFeedback Settings section for setting up the additional feedback type and resolution. • The default feed constant value in the KAS-IDE is scaled to 65536. • This feed constant value can be changed using EtherCAT object 0x60E9 subindex 1-5. • The feedback signal comes through EtherCAT in object 0x60E4 subindex 1-5.

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID Name of the SMP function in the Pipe Network.
SamplingPeriod	LREAL	0.25 to 31,557,600,000.0	Millisecond	No default	Period that the device is sampled, expressed in milliseconds.  Figure 4-57: Example: Sampler Period
Mode	DINT	See Description.	N/A	No default	Sampled output can be either Position or Speed. • 1 = Position Mode. • 2 = Speed Mode.  Figure 4-58: Sampler Mode Position  Figure 4-59: Sampler Mode Speed

Input	Data Type	Range	Unit	Default	Description
InputModuloPosition	LREAL	0 to 1.0x10 ³⁰⁰	User units	No default	Period of the input signal. - When used to track the feedback value of an EtherCAT device, the value of InputModuloPosition depends on the EtherCAT device used as an Input Source for this Pipe Block. AKD: This should be set equal to 2³² (4294967296.0). AKD2G: This should be set based on the feed constant value assigned in the EtherCAT object 0x60E4 subindex 1-5. 0 (zero) indicates the input signal does not use the modulo operation. See "Modulo Affects on the Signal Scaling" (→ p. 574).
OutputModuloPosition	LREAL	0 to 1.0x10 ³⁰⁰	User units	No default	Period of the output signal. 0 (zero) indicates the output signal does not use the modulo operation. See "Modulo Affects on the Signal Scaling" (→ p. 574).

Outputs

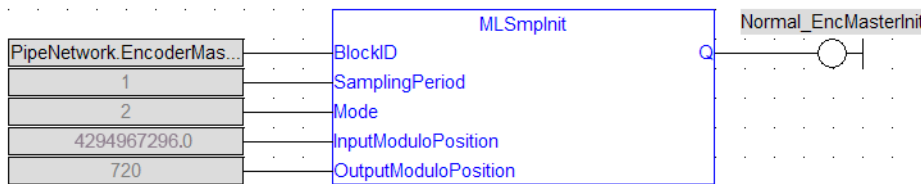
Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	SMP Block successfully initiated.

3.1.16.5.1.1 Modulo Affects on the Signal Scaling

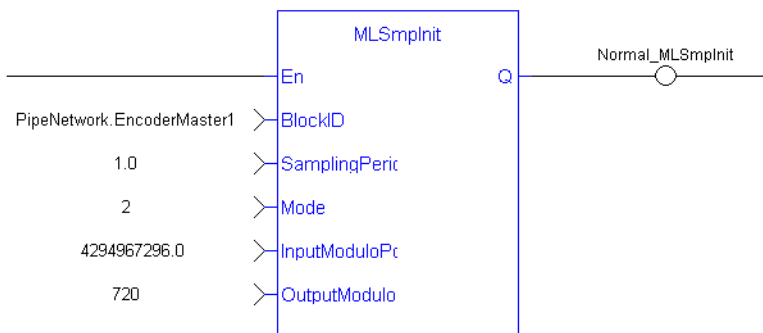
This table lists how the values of InputModuloPosition and OutputModuloPosition affects the scaling of the signal.

InputModuloPosition Value	OutputModuloPosition Value	Scaling Effect	Rollover Effect
Non-zero	Non-zero	Output signal = (Input signal) x (OutputModuloPosition / InputModuloPosition)	Output signal rolls over at OutputModuloPosition.
0 (zero)	0 (zero)	Output signal = Input signal	Output signal does not roll over.
Non-zero	0 (zero)	Output signal = Input signal	Output signal rolls over at OutputModuloPosition.
0 (zero)	Non-zero	Output signal = Input signal	Output signal rolls over at InputModuloPosition.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Initialize a Sampler Pipe Block named "EncoderMaster1" to a Sample Period
of 1 millisec, Mode of Operation to 2 (Velocity), Input Modulo of 4294967296,
and Output Modulo of 720
MLSmpInit( PipeNetwork.EncoderMaster1, 1.0,2,4294967296,720);
```

See Also

- "MLSmpConECAT" (→ p. 564)
- "MLSmpConnect" (→ p. 566)

- "MLSmpConPLCAxis" (→ p. 568)
- "MLSmpConPNAxis" (→ p. 570)

3.1.17 Synchronizer

Name	Description
MLSyncInit	Initializes a synchronizer Pipe Block.
MLSyncReadDeltaS	Gets the output phasing value of a synchronizer block.
MLSyncStart	Starts a synchronization of a synchronizer Pipe Block.
MLSyncStop	De-synchronizes a synchronizer Pipe Block.
MLSyncWriteDeltaS	Sets the output phasing value of a synchronizer block.

3.1.17.1 Example: Synchronizer Functions

- When **MLSyncStart** function is called, the output value is adapted to catch up with the input value.
- The OUTPUT_PHASING parameter is used to define a curve in order to smooth the output value. (Figure 4-60)

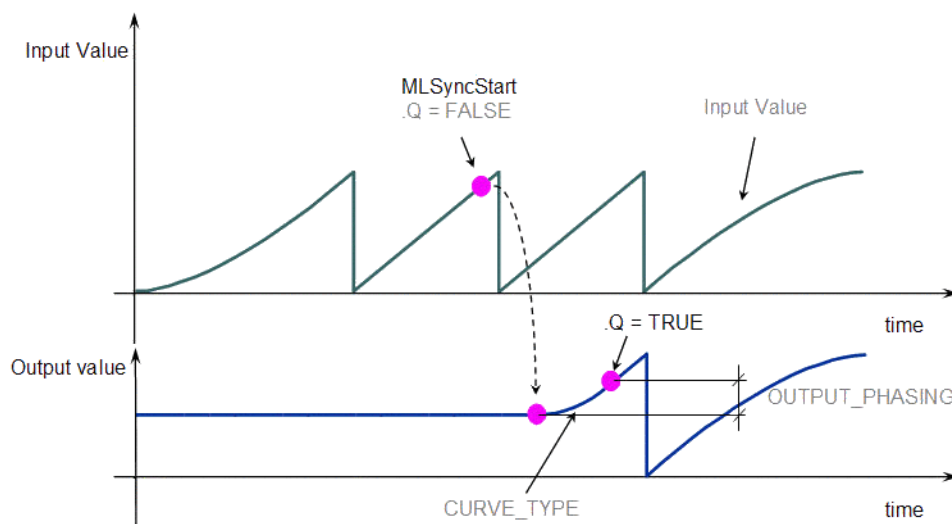


Figure 4-60: Get Output Phasing after **MLSyncStart**

- When the **MLSyncStop** function is called, the output value is adapted according to the specified Stop-Position (point B).
- The OUTPUT_PHASING parameter is used to define point A, where the flow follows a curve in order to smooth the output value. (Figure 4-61)

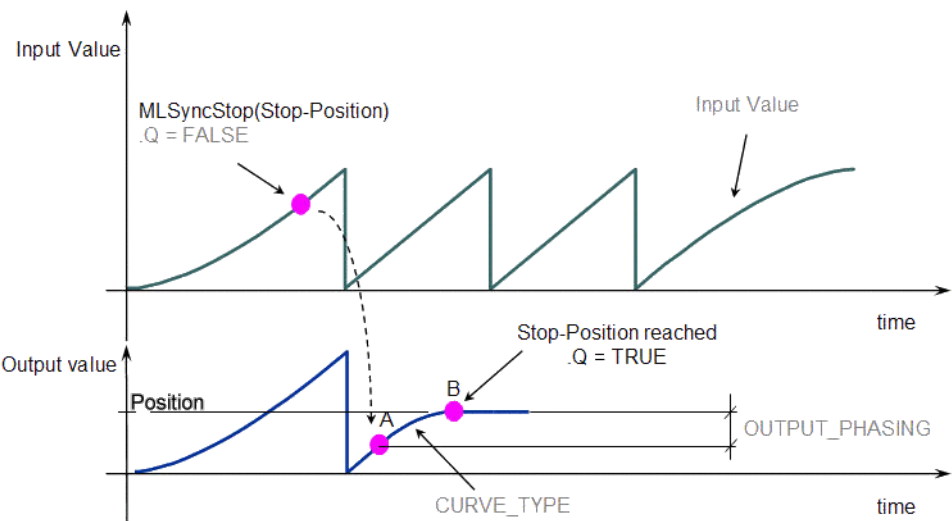


Figure 4-61: Get Output Phasing after MLSyncStop

3.1.17.2 MLSyncInit

Pipe Network



Function - Initializes a synchronizer Pipe Block.

- Returns TRUE if the function succeeded.
- This FB is automatically created in the compiled code of a Pipe Network.
- This function block is part of the MLPN_CREATE_OBJECT to initialize the Pipe Network.
 - It is called at the beginning of an application program with the function call:

```
PipeNetwork (MLPN_CREATE_OBJECTS) ;
```

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Pipe Network Block.
ModuloPosition	LREAL	0 to 1.0x10 ³⁰⁰	User units	No default	• Value of the cyclic period of the input signal. • Expressed in User units. • The parameter is defined to correctly manage the periodicity (modulo) of the input values. • 0 (zero) indicates the input signal does not use the modulo operation.
CurveType	DINT	1 to 2	N/A	No default	When synchronizing, specifies which type of curve (parabolic or polynomial) has to be implemented for merging with the master. The options are: • 1 = Parabolic • 2 = Polynomial
DeltaS	LREAL	No range	User units	No default	• The Distance to get in or out of synchronization. • This parameter is used in "MLSyncStart" (→ p. 583) and "MLSyncStop" (→ p. 585).

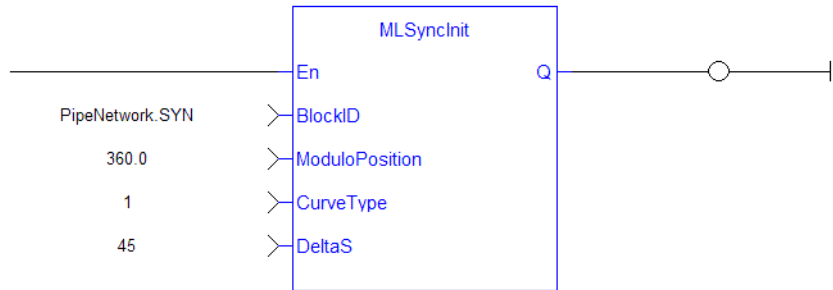
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Function or function block executed successfully.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Initialize a synchronizer Pipe Block named " SYN" with a modulo of 360,  
Curve Type of Parabolic, and a distance (DeltaS) of 30 to get in and out of  
synchronization  
MLSyncInit( PipeNetwork.SYN, 360, 1, 30 );
```

See Also

"MLSyncWriteDeltaS" (→ p. 587)

3.1.17.3 MLSyncReadDeltaS

Pipe Network



Function - Gets the output phasing value of a synchronizer block.

- Output phasing is the distance or the slope the output takes to synchronize with the input when "MLSyncStart" (→ p. 583) block is executed.
 - See Figure 4-62.
- It also affects the distance or the slope the output takes to de-synchronize with the input and come to a stop when "MLSyncStop" (→ p. 585) block is executed.
 - See Figure 4-63.

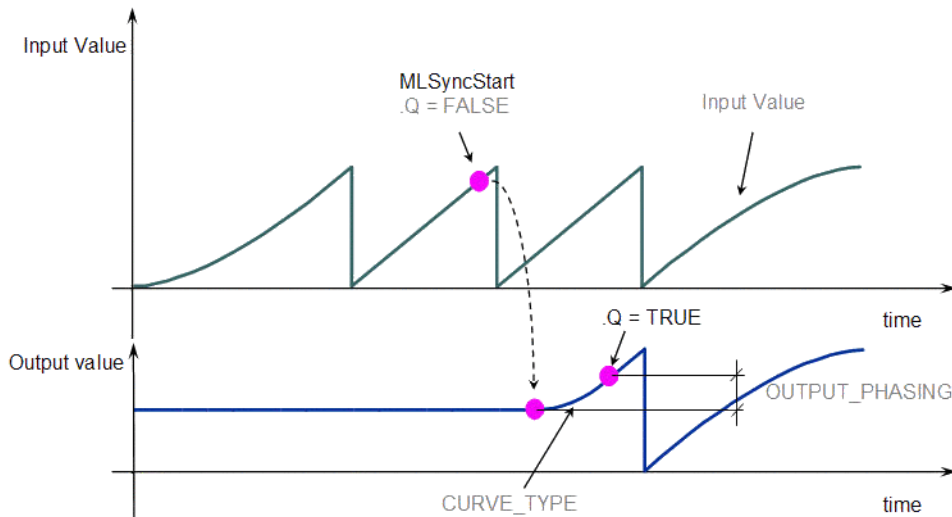


Figure 4-62: Get Output Phasing after **MLSyncStart**

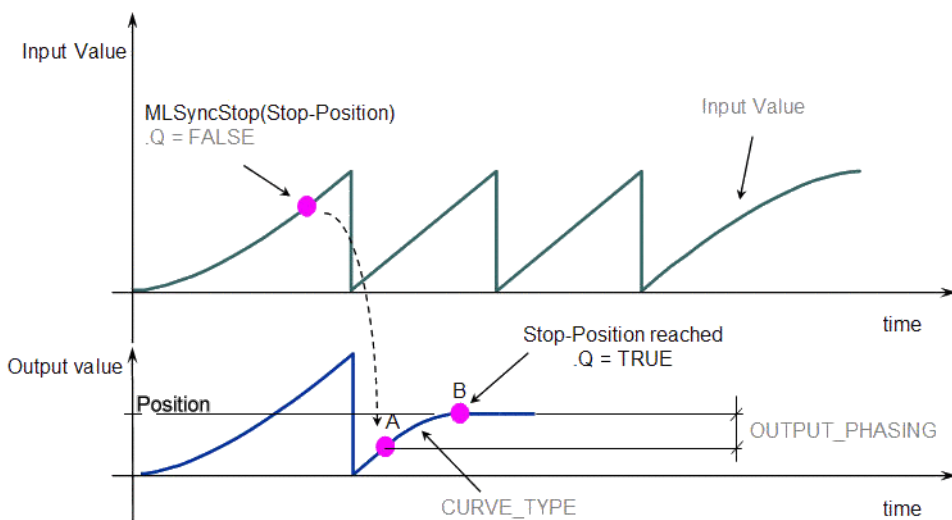


Figure 4-63: Get Output Phasing after **MLSyncStop**

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Pipe Network Block.

Outputs

Output	Data Type	Range	Unit	Description
DeltaS	LREAL		User units	Present Delta Slope value.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
ActScope := MLSyncReadDeltaS( PipeNetwork.SYN );
```

See Also

"MLSyncWriteDeltaS" (→ p. 587)

3.1.17.4 MLSyncStart

Pipe Network



Function - Starts a synchronization of a synchronizer Pipe Block.

Returns TRUE if the function succeeded.

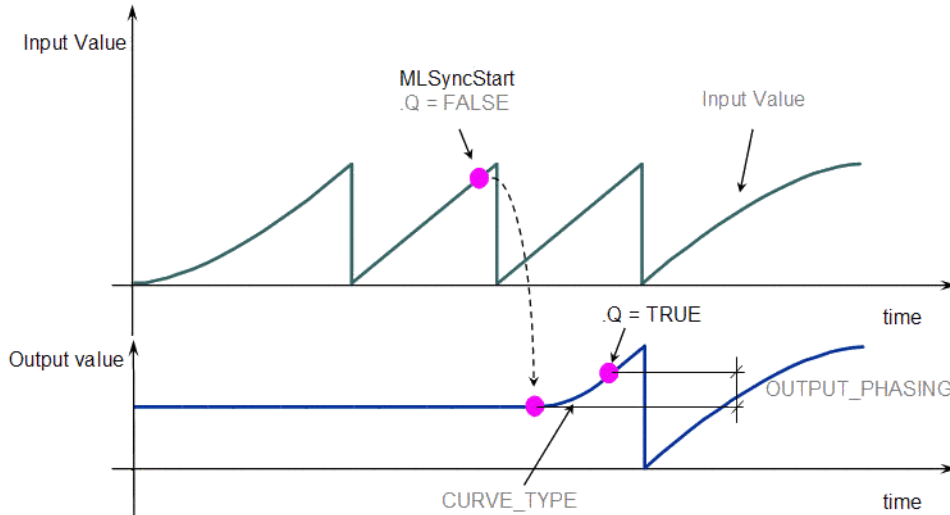


Figure 4-64: MLSyncStart

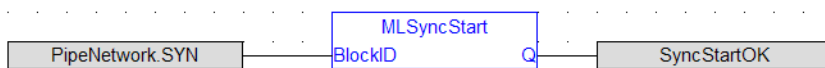
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Pipe Network Block.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Function or function block executed successfully.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLSyncStart( PipeNetwork.SYN );
```

3.1.17.5 MLSyncStop



Function - De-synchronizes a synchronizer Pipe Block.
Returns TRUE if the function succeeded.

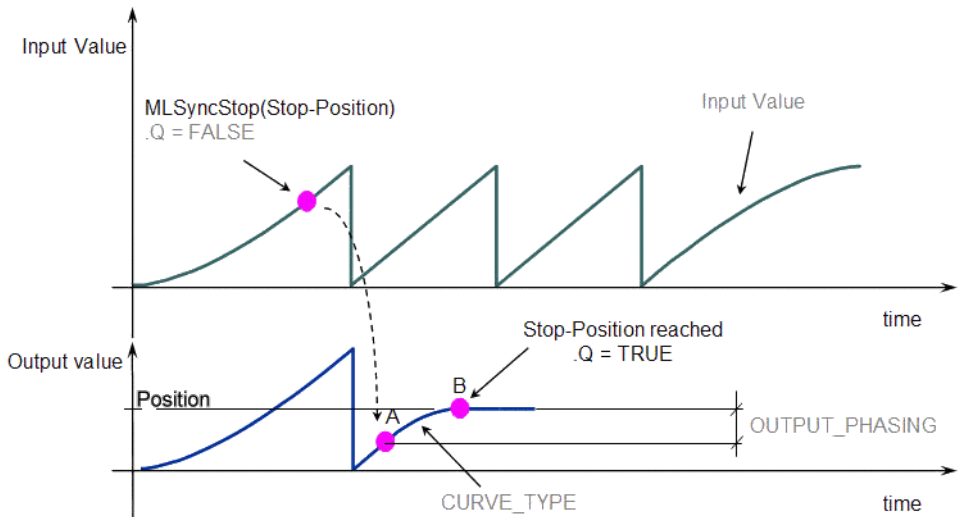


Figure 4-65: MLSyncStop

Inputs

Input	Data Type	Range	Unit	Default	Description
Position	LREAL	No range	User units	No default	Motion Stop Position.

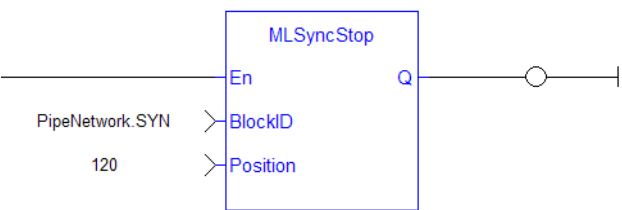
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Function or function block executed successfully.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLSyncStop( PipeNetwork.SYN , 120 );
```

3.1.17.6 MLSyncWriteDeltaS

Pipe Network



Function - Sets the output phasing value of a synchronizer block.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.
- Returns TRUE if the function succeeded.
- Output phasing is the distance or the slope the output takes to synchronize with the input when MLSyncStart block is executed.
- It also affects the distance or the slope the output takes to de-synchronize with the input and come to a stop when MLSyncStop block is executed.

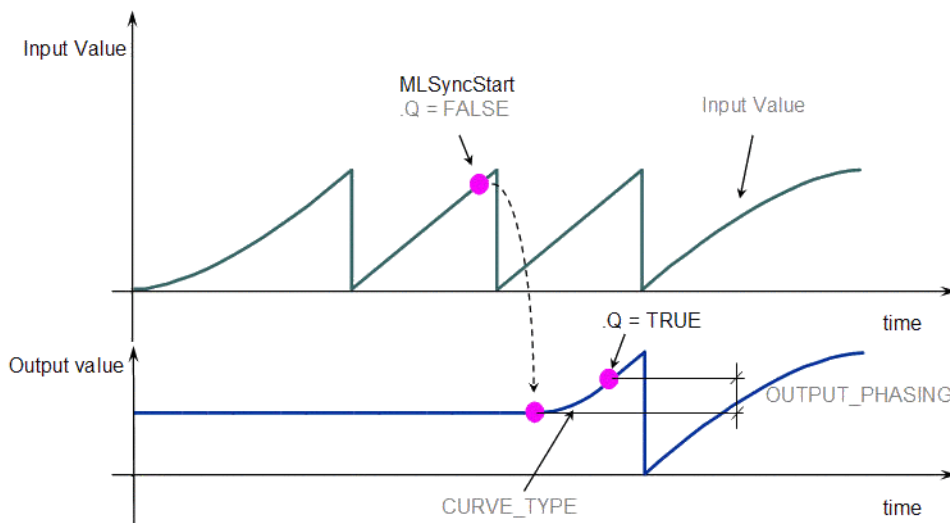


Figure 4-66: Set output phasing after MLSyncStart

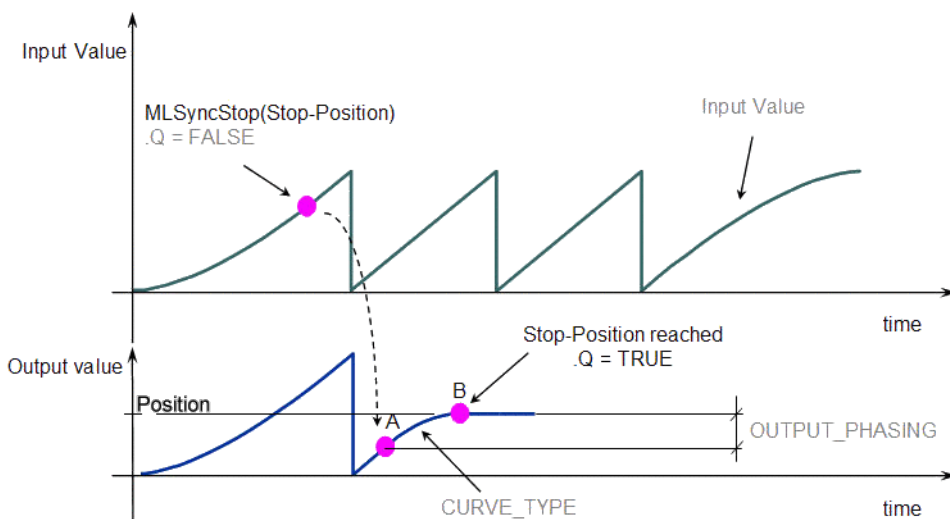


Figure 4-67: Set output phasing after MLSyncStop

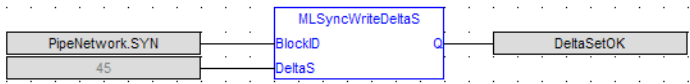
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID name of the Pipe Network Block.
DeltaS	LREAL	No range	User units	No default	- Slope to be used during Start and stop of Synchronization. - This parameter is used in "MLSyncStart" (→ p. 583) and "MLSyncStop" (→ p. 585).

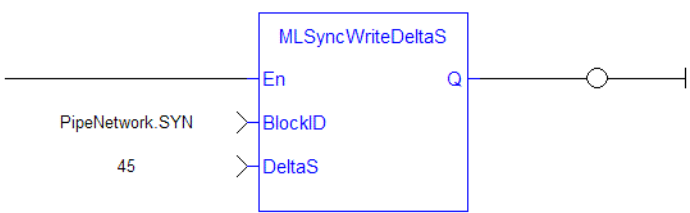
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Function or function block executed successfully.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
MLSyncWriteDeltaS( PipeNetwork.SYN, 45 );
```

3.1.18 Trigger

Name	Description
MLTrigClearFlag	Clears the flag of an initiated Trigger block.
MLTrigInit	Initializes a Trigger object for use in a PLC program.
MLTrigsTriggered	Checks if the selected block has been triggered.
MLTrigReadDelay	Returns the time the trigger block uses to compensate the delay of the sensor that captures the triggering signal.
MLTrigReadPos	Returns the position of the block at the moment when it was triggered.
MLTrigReadTime	Returns the time of the moment where the block was triggered in milliseconds.
MLTrigSetEdge	Sets the edge configuration for a trigger block (e.g., rising, falling, etc.)
MLTrigWriteDelay	Sets the time the trigger block uses to compensate for the delay introduced by the sensor that captures the triggering signal.

3.1.18.1 Example: Trigger Functions

① IMPORTANT

- The trigger delay must be calculated by **you** and set with "MLTrigWriteDelay" (→ p. 605).
- This delay belongs to the sensor.
- It is in addition to the "MLTrigReadTime" (→ p. 601) / "MLTrigReadPos" (→ p. 599).

When you call the **MLTrigClearFlag** function, the flag for trigger is reset to FALSE. (Figure 4-68)

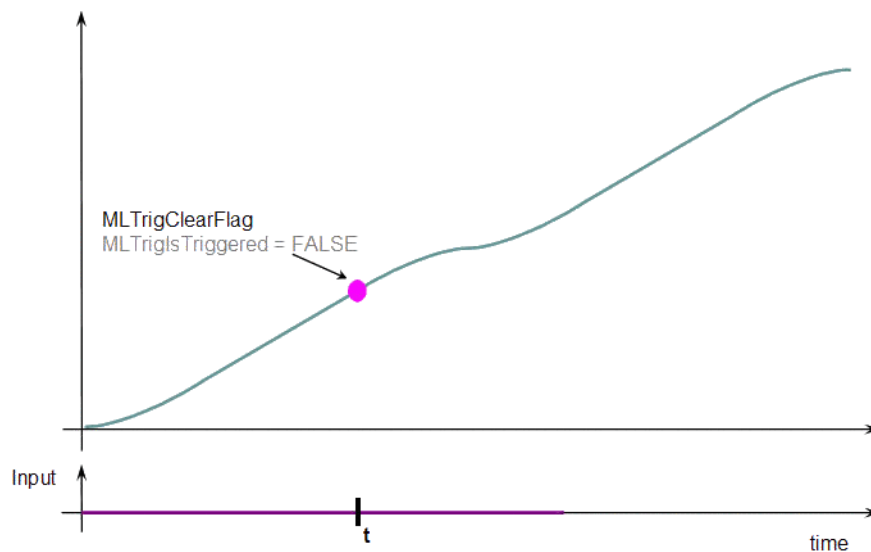


Figure 4-68: Example: MLTrigClearFlag = FALSE

When a Fast Input is set, the **MLTrigsTriggered** function returns TRUE. (Figure 4-69)

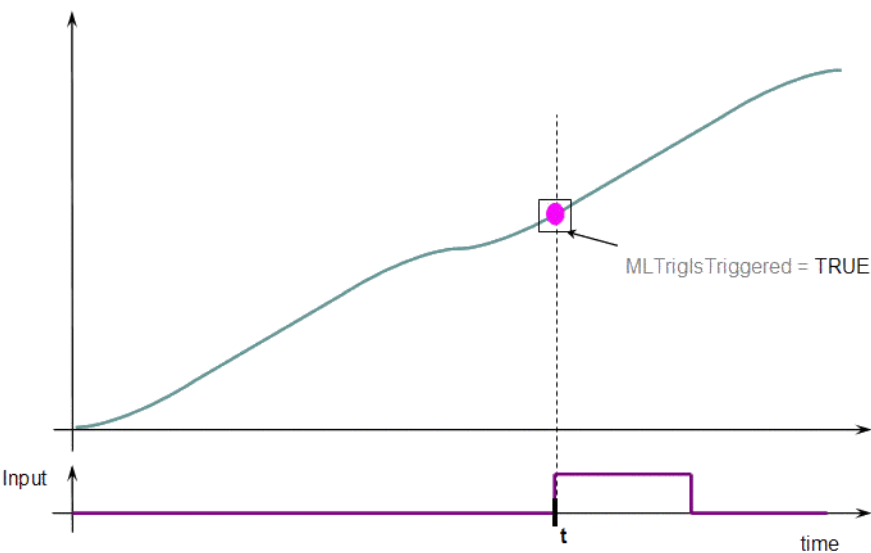


Figure 4-69: Example: MLTrigsTriggered = TRUE

Then call the **MLTrigReadPos** and **MLTrigReadTime** functions to get more details. (Figure 4-70)

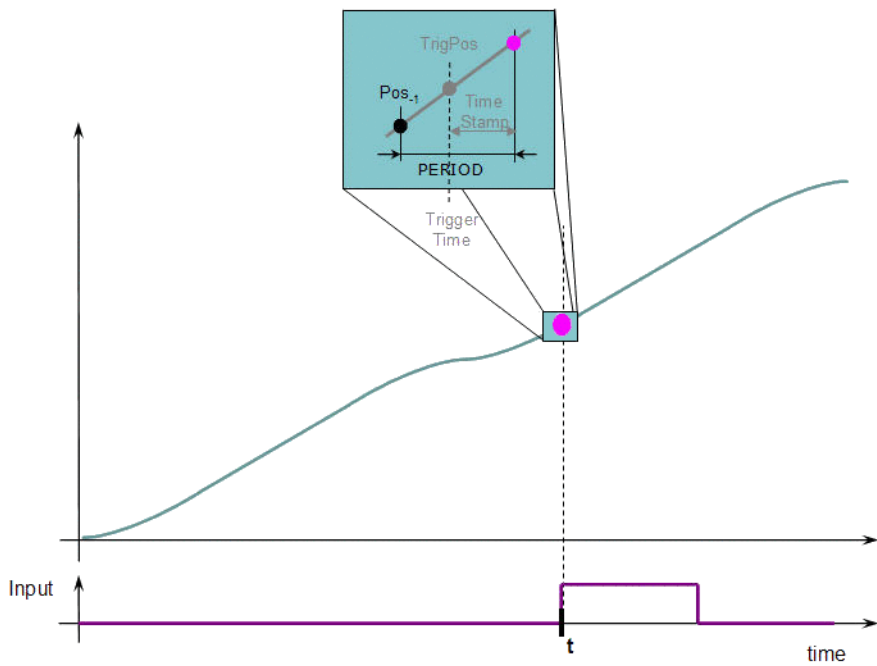


Figure 4-70: Example: MLTrigReadPos and MLTrigReadTime details

3.1.18.2 MLTrigClearFlag

Pipe Network



Function - Clears the flag of an initiated Trigger block.

- The block captures the position and time of the next event.
- Once triggered, a block has to be reset before it can be triggered again.
 - All events sent to a block while in a triggered state are ignored.
 - The position and time information is lost.

ⓘ IMPORTANT

- The Fast Input assigned to a Trigger block has to be reset before information on a new event can be captured.
- **MLAxisRstFastIn** is used at the same time as **MLTrigClearFlag**.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

3.1.18.2.1.1 Fast Homing - Inputs

See these topics:

- Example: Fast Homing with the Pipe Network Motion Engine Axis Pipe Block
- Example: Fast Homing with the PLCopen Motion Engine
- Fast Inputs with Pipe Network Motion
- H1-Pipe Network Registration and Fast Homing
- Example: Registration Position Capture with Pipe Network Trigger Block

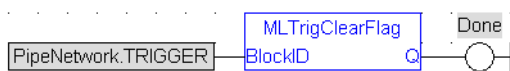
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Trigger object.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the function block is executed. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Clear Trigger Flag
MLTrigClearFlag( PipeNetwork.TRIGGER );
```

See Also

- "MLAxisRstFastIn" (→ p. 384)
- "MLTrigsTriggered" (→ p. 595)
- "MLTrigReadPos" (→ p. 599)
- "MLTrigReadTime" (→ p. 601)

3.1.18.3 MLTrigInit

Pipe Network



Function - Initializes a Trigger object for use in a PLC program.

- This function block is automatically called if a Trigger Block is added to the Pipe Network, with user-defined settings entered in the Pipe Blocks Properties screen.
- The Trigger object monitors a selected Fast Input and captures the time of a rising or falling edge event.
 - With the time and pipe position information, the Trigger object extrapolates the axis position when the Fast Input event occurred.
- Parameters to enter include:
 - The name of the Pipe Block.
 - The Axis where the Fast Input is located.
 - The number of the desired Fast Input.
 - Whether to trigger on the rising or falling edge of the input.

NOTE

- Trigger objects are normally created in the Pipe Network using the graphical engine.
- You do not have to add **MLTrigInit** functions to their programs.
- Parameters are entered directly in pop-up windows; the code is automatically added to the current project.

3.1.18.3.1.1 Fast Homing - Inputs

See these topics:

- Example: Fast Homing with the Pipe Network Motion Engine Axis Pipe Block
- Example: Fast Homing with the PLCopen Motion Engine
- Fast Inputs with Pipe Network Motion
- H1-Pipe Network Registration and Fast Homing
- Example: Registration Position Capture with Pipe Network Trigger Block

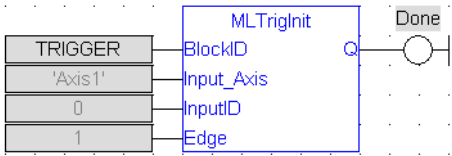
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Trigger object.
Input_Axis	STRING	No range	N/A	No default	Name of the axis where the Fast Input is located.
InputID	DINT	0 to 1	N/A	No default	ID number of the Fast Input. Valid values of InputID are: • 0 = Touch Probe 1 / Capture Engine 0 • 1 = Touch Probe 2 / Capture Engine 1
TriggerMode / Edge	DINT	See Description.	N/A	No default	The edge to trigger on. • 0 = Disable the Fast Input. • 1 = Rising edge. • 2 = Falling edge.

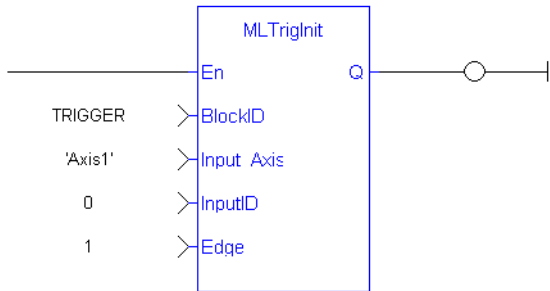
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the function block is executed. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Create and Initiate a Trigger Pipe Block named "Trigger" and set it up to
receive the trigger signal from Axis1, capture engine 0, and the rising edge
of the signal
TRIGGER := MLBlkCreate( 'TRIGGER', 'TRIGGER' );
MLTrigInit( TRIGGER, 'Axis1', 0, 1 );
```

See Also

- "MLAxisRstFastIn" (→ p. 384)
- "MLTrigClearFlag" (→ p. 591)
- "MLTrigsTriggered" (→ p. 595)
- "MLTrigReadPos" (→ p. 599)

3.1.18.4 MLTrigsTriggered

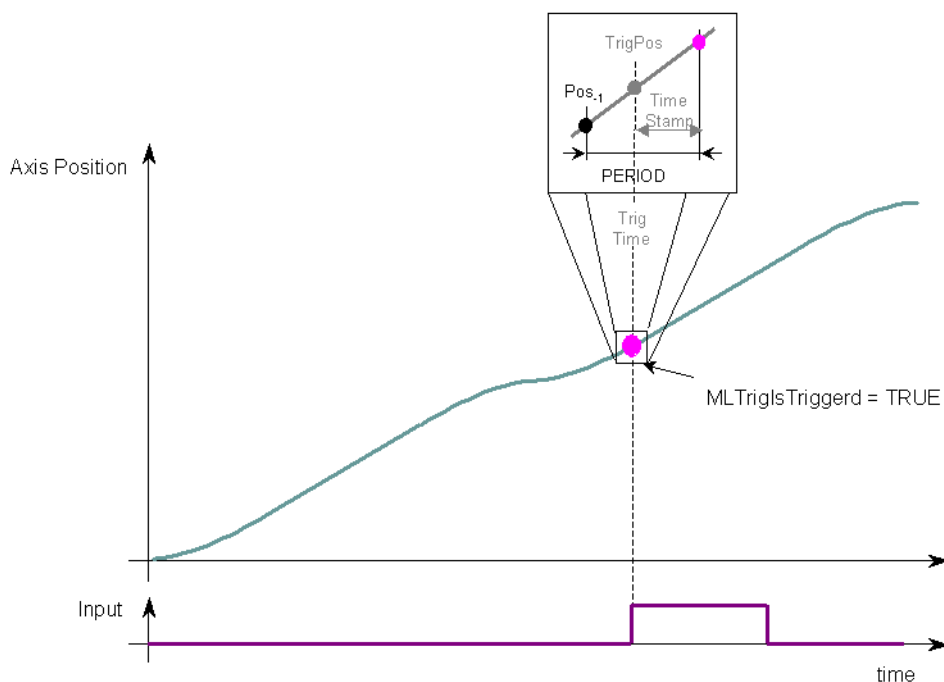
Pipe Network

 **Function** - Checks if the selected block has been triggered.

- When a block has been triggered, it contains the time and position when a Fast Input event occurred.
 - The application has to reset the block before the block can be triggered again.
- All trigger events that are sent to the block during its triggered state are lost.
- Once triggered, a block has to be reset before it can be triggered again.
 - All events sent to a block while in a triggered state are ignored.
 - The position and time information is lost.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.



3.1.18.4.1.1 Fast Homing - Inputs

See these topics:

- Example: Fast Homing with the Pipe Network Motion Engine Axis Pipe Block
- Example: Fast Homing with the PLCopen Motion Engine
- Fast Inputs with Pipe Network Motion
- H1-Pipe Network Registration and Fast Homing
- Example: Registration Position Capture with Pipe Network Trigger Block

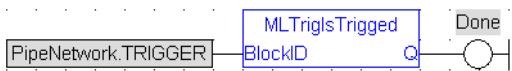
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Trigger object.

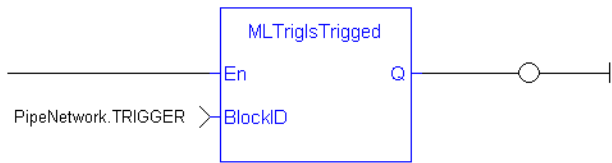
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the selected Trigger Object has triggered. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Check if a Trigger Block has been triggered, then save position
IF MLTrigIsTriggered( PipeNetwork.TRIGGER ) THEN
  Trig_Position := MLTrigReadPos( PipeNetwork.TRIGGER );
END_IF
```

See Also

- "MLTrigReadPos" (→ p. 599)
- "MLTrigReadTime" (→ p. 601)

3.1.18.5 MLTrigReadDelay



Function - Returns the time the trigger block uses to compensate the delay of the sensor that captures the triggering signal.

- Returns the delay programmed in a trigger block by the "MLTrigWriteDelay" (→ p. 605) function to compensate for the reaction time required by the sensor.
- Electronic sensors are not able to respond immediately to a signal.
- Sensors usually require a certain amount of time to process a change of state in their input signal.

3.1.18.5.1.1 Fast Homing - Inputs

See these topics:

- Example: Fast Homing with the Pipe Network Motion Engine Axis Pipe Block
- Example: Fast Homing with the PLCopen Motion Engine
- Fast Inputs with Pipe Network Motion
- H1-Pipe Network Registration and Fast Homing
- Example: Registration Position Capture with Pipe Network Trigger Block

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	Identifier of the trigger block.
En	BOOL	FALSE, TRUE	N/A	No default	Enables execution.

Outputs

Output	Data Type	Range	Unit	Description
Delay	LREAL	No range	Microseconds	Value of the delay compensation currently applied by the trigger block.
OK	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes.

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

Not available.

See Also

"MLTrigWriteDelay" (→ p. 605)

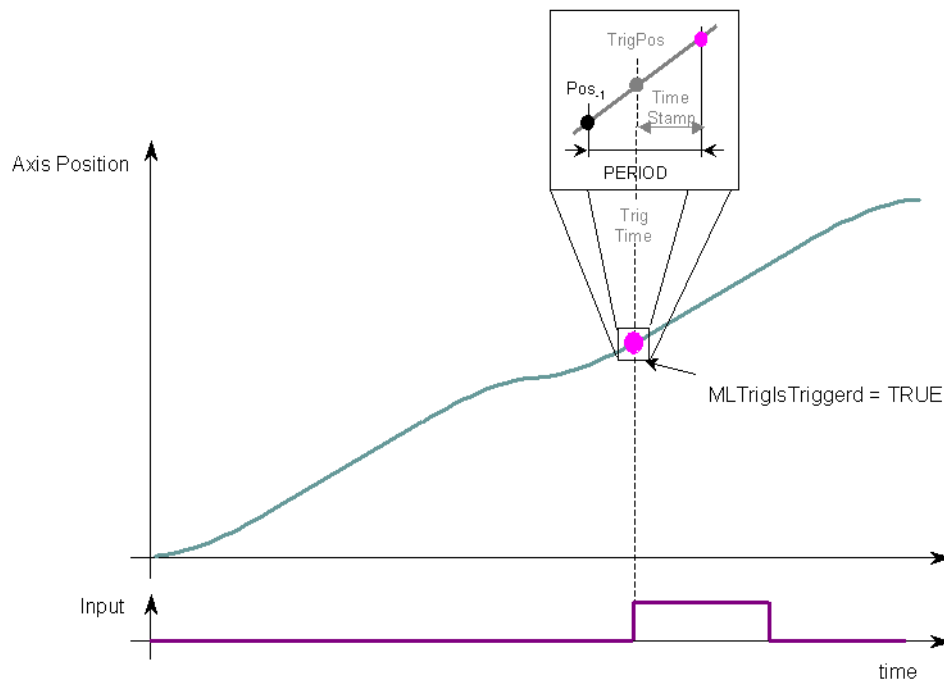
3.1.18.6 MLTrigReadPos

Pipe Network



Function - Returns the position of the block at the moment when it was triggered.

- Returns the modulo-applied position of the pipe at the moment it is triggered by the Trigger Block's selected Fast Input.
- This value is only valid when TrigsTriggered() returns TRUE.
 - The Trigger block extrapolates the output value based on the timestamp of the Fast Input event to provide an accurate position even if the event occurs in the middle of a program cycle.
- Once triggered, a block has to be reset before it can be triggered again.
 - All events sent to a block while in a triggered state are ignored.
 - The position and time information is lost.



- **Modulo Calculation:** MLTrigReadPos uses the Output Modulo Position value of the previous block in the pipe, even if the previous pipe is configured for No Modulo mode.
 - The previous block must specify a 0 (zero) value for Output Modulo Position before setting the Mode to No Modulo to prevent a modulo operation for MLTrigReadPosPipe.
- Previous Function Name: MLTrigGetPos

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

3.1.18.6.1.1 Fast Homing - Inputs

See these topics:

- Example: Fast Homing with the Pipe Network Motion Engine Axis Pipe Block
- Example: Fast Homing with the PLCopen Motion Engine
- Fast Inputs with Pipe Network Motion
- H1-Pipe Network Registration and Fast Homing
- Example: Registration Position Capture with Pipe Network Trigger Block

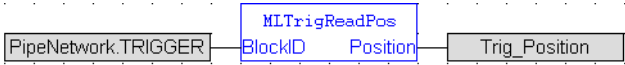
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Trigger object.

Outputs

Output	Data Type	Range	Unit	Description
Position	LREAL	No range	User units	Returns the position of the selected block's Axis at the moment when it was triggered.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Save position of Axis when Fast Input event occurs
Trig_Position := MLTrigReadPos( PipeNetwork.TRIGGER );
```

See Also

- "MLAxisRstFastIn" (→ p. 384)
- "MLTrigClearFlag" (→ p. 591)
- "MLTrigsTriggered" (→ p. 595)
- "MLTrigReadTime" (→ p. 601)

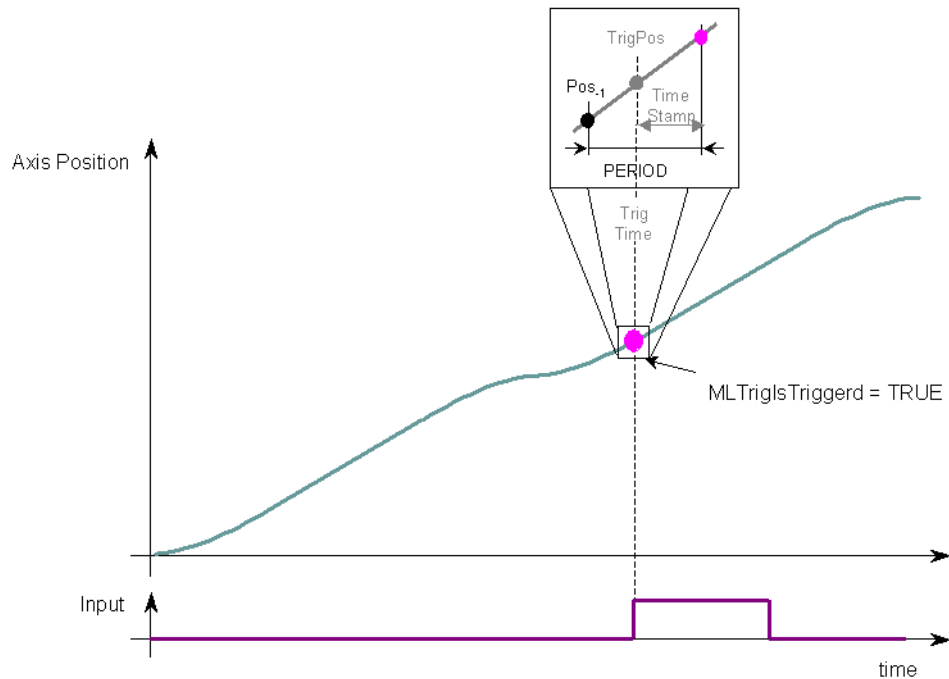
3.1.18.7 MLTrigReadTime

Pipe Network



Function - Returns the time of the moment where the block was triggered in milliseconds.

- This value is only valid when TrigsTriggered() returns TRUE.
 - The output is computed from the timestamp of a Fast Input time event.
- Once triggered, a block has to be reset before it can be triggered again.
 - All events sent to a block while in a triggered state are ignored.
 - The position and time information is lost.



- Previous Function Name: MLTrigGetTime

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

3.1.18.7.1.1 Fast Homing - Inputs

See these topics:

- Example: Fast Homing with the Pipe Network Motion Engine Axis Pipe Block
- Example: Fast Homing with the PLCopen Motion Engine
- Fast Inputs with Pipe Network Motion
- H1-Pipe Network Registration and Fast Homing
- Example: Registration Position Capture with Pipe Network Trigger Block

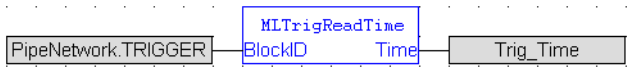
Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of an initiated Trigger object.

Outputs

Output	Data Type	Range	Unit	Description
Time	LREAL	No range	Microseconds	Returns the time the Trigger Block's selected Fast Input was triggered.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Save time when Fast Input event occurs
Trig_Time := MLTrigReadTime( PipeNetwork.TRIGGER );
```

See Also

- "MLAxisRstFastIn" (→ p. 384)
- "MLTrigClearFlag" (→ p. 591)
- "MLTrigsTriggered" (→ p. 595)
- "MLTrigReadPos" (→ p. 599)

3.1.18.8 MLTrigSetEdge



Function - Sets the edge configuration for a trigger block (e.g., rising, falling, etc.)

- This block should be called prior to calling "MLAxisCfgFastIn" (→ p. 323).
- The value at the Edge input must match the value at MLAxisCfgFastIn's Mode input.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

3.1.18.8.1.1 Fast Homing - Inputs

See these topics:

- Example: Fast Homing with the Pipe Network Motion Engine Axis Pipe Block
- Example: Fast Homing with the PLCopen Motion Engine
- Fast Inputs with Pipe Network Motion
- H1-Pipe Network Registration and Fast Homing
- Example: Registration Position Capture with Pipe Network Trigger Block

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	ID number of the trigger block.
Edge	DINT	0 to 2	N/A	No default	The edge to trigger on. • 0 = Disable the Fast Input. • 1 = Rising edge. • 2 = Falling edge.

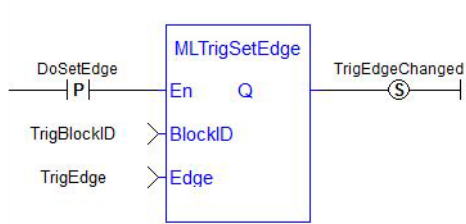
Outputs

Output	Data Type	Range	Unit	Description
Q	BOOL	FALSE, TRUE	N/A	• TRUE if block executed successfully. • FALSE if execution is not successful.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
TrigEdgeChanged := MLTrigSetEdge (TrigBlockID,TrigEdge);
```

3.1.18.9 MLTrigWriteDelay



Function - Sets the time the trigger block uses to compensate for the delay introduced by the sensor that captures the triggering signal.

- Allows the trigger block to calculate the exact moment at which a signal was triggered by the specified compensation.
- The delay compensation should include drive processing time, sensor delay, and the communication latency through the EtherCAT network.
- Electronic sensors are not able to respond immediately to a signal.
- Sensors usually require a certain amount of time to process a change of state in their input signal.

3.1.18.9.1.1 Fast Homing - Inputs

See these topics:

- Example: Fast Homing with the Pipe Network Motion Engine Axis Pipe Block
- Example: Fast Homing with the PLCopen Motion Engine
- Fast Inputs with Pipe Network Motion
- H1-Pipe Network Registration and Fast Homing
- Example: Registration Position Capture with Pipe Network Trigger Block

Inputs

Input	Data Type	Range	Unit	Default	Description
BlockID	DINT	1 to 1024	N/A	No default	Identifier of the trigger block.
Delay	LREAL	No range	Microseconds	No default	Reaction time of the sensor that the trigger block has to compensate.

Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if the delay is successfully set. . •

FBD Language Example

Not available.

FPLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

Not available.

See Also

"MLTrigReadDelay" (→ p. 597)

3.2 Motion / PLCopen Library

These are the Motion/PLCopen functions and function blocks.

- "Control" (→ p. 607)
- "Information" (→ p. 607)
- "I/O" (→ p. 607)
- "PLCopenMotion" (→ p. 608)
- "Profile" (→ p. 608)
- "Reference" (→ p. 609)
- "Registration" (→ p. 609)

3.2.1 Control

Name	Description
MC_ClearFaults	Sends a request to the drive to clear any drive faults.
MC_CreatePLCAxis	Creates a PLCopen Axis.
MC_EStop	Causes an emergency stop (E-stop).
MC_InitAxis	Initializes a PLCopen Axis' data.
MC_InitAxisFeedback	Initializes a PLCopen digitizing Axis' position data.
MC_Power	Requests to either enable the drive and close the loop, or disable the drive and open the loop.
MC_ResetError	Resets the errors of the specified axis.
MC_Stop	Aborts the active move, removes the next move from the queue, performs a controlled stop, and switches the axis to Stopping state.

3.2.2 Information

Name	Description
MC_ReadActPos	Reads the actual position of the axis.
MC_ReadActVel	Reads the actual velocity of the axis.
MC_ReadAxisErr	Returns the error status of the specified axis.
MC_ReadBoolPar	Returns the value of the specified Boolean axis parameter.
MC_ReadParam	Returns the value of the specified axis parameter.
MC_ReadStatus	Returns the state of the specified axis.
MC_WriteBoolPar	Writes the specified axis Boolean parameter.
MC_WriteParam	Writes the specified axis parameter.

3.2.3 I/O

Name	Description
MC_AbortTrigger	When the Execute input transitions from low to high, this function block aborts an MC_TouchProbe function block.
MC_TouchProbe	Arms a Fast Input and returns the latched position when the Fast Input event occurs.

3.2.4 PLCopenMotion

Name	Description
MC_AddSuperAxis	Adds a Superimposed Axis to the Axis's list of assigned superimposed axes.
MC_Halt	Decelerates an axis to zero velocity.
MC_MoveAbsolute	Performs a single-axis move to a specified endpoint position.
MC_MoveAdditive	Performs a single-axis move for a specified distance from the endpoint of the previous move.
MC_MoveContVel	Performs a single-axis non-ending move at a specified velocity with the option of continually updating the ongoing motion with the current input parameters.
MC_MoveRelative	Performs a single-axis move of a specified distance relative to the actual position at the time of the start of execution.
MC_MoveSuperimp	Performs a single-axis move superimposed upon the currently executing move.
MC_MoveVelocity	Performs a single-axis, non-ending move at a specified velocity.
MC_RemSuperAxis	Remove an axis from the axis's list of assigned, superimposed axes.
MC_SetOverride	Writes velocity and acceleration override factors.

3.2.5 Profile

Name	Description
MC_CamIn	Performs a slave axis move which follows the master axis based on the Cam Table specified by CamTableID.
MC_CamOut	Disengages the slave axis from an "MC_CamIn" (→ p. 693) move or the master axis immediately in a cam block.
MC_CamResumePos	Returns the slave axis position for resuming an "MC_CamIn" (→ p. 693) move.
MC_CamStartPos	Returns the slave axis position for starting an "MC_CamIn" (→ p. 693) move.
MC_CamTblSelect	Defined to read and initialize the specified profile. Selects the Cam tables by setting the pointers to the relevant tables.
MC_GearIn	Performs a slave axis move which follows the master axis using the ratio specified by RatioNumerator and RatioDenominator. Once engaged, the slave moves RatioNumerator units for every RatioDenominator units the master moves.
MC_GearInPos	Performs a slave axis move which synchronizes motion at the SlaveSyncPosition and MasterSyncPosition. Once synchronized, the slave moves RatioNumerator units for every RatioDenominator units the master moves.
MC_GearOut	Disengages the slave axis from a MC_GearIn or MC_GearInPos move or the master axis.
MC_Phasing	Performs a master position phase shift for the slave axis.
MC_SyncSlaves	Allows the application to specify what slave axes are to be synchronized and which master they follow.

3.2.6 Reference

Name	Description
MC_Reference	Used to execute a fast home to a switch.
MC_SetPos	Changes the present actual position of the axis (as reported by "MC_ReadActPos" (→ p. 642)) to the position specified by the Position and Mode inputs.
MC_SetPosition	Deprecated - See "MC_SetPos" (→ p. 736).

3.2.7 Registration

Name	Description
MC_MachRegist	Enables Mark-to-Machine registration.
MC_MarkRegist	Enables Mark-to-Mark registration.
MC_StopRegist	Turns off registration for the specified axis and disarms the specified fast input.

3.2.8 Control

These functions provide general controls to drives and axes.

Name	Description
MC_ClearFaults	Sends a request to the drive to clear any drive faults.
MC_CreatePLCAxis	Creates a PLCopen Axis.
MC_EStop	Causes an emergency stop (E-stop).
MC_InitAxis	Initializes a PLCopen Axis' data.
MC_InitAxisFeedback	Initializes a PLCopen digitizing Axis' position data.
MC_Power	Requests to either enable the drive and close the loop, or disable the drive and open the loop.
MC_ResetError	Resets the errors of the specified axis.
MC_Stop	Aborts the active move, removes the next move from the queue, performs a controlled stop, and switches the axis to Stopping state.

3.2.8.1 MC_ClearFaults

PLCopen 



Function - Sends a request to the drive to clear any drive faults.

- The condition causing the drive fault has to be corrected before calling this function.
- If the fault condition still exists when this function is called, this function sends a request to the drive but the drive faults remain.
- This function does **not** reset axis errors.
 - [MC_ResetError](#) is required to reset axis errors and possibly to re-enable or turn power on to the servo axis after the fault condition is cleared.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

3.2.8.1.1.1 Usage

When there is a positive transition of the EN input, this function requests a Fault Reset of the Drive for the Axis defined in the axis input of this function.

Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	When TRUE, requests to clear the drive faults.
Axis	AXIS_REF	1 to 256	N/A	No default	AXIS_REF.AXIS_NUM is the master axis number.

Outputs

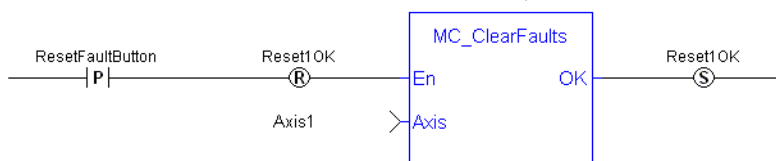
Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Boolean output to indicate successful request. • This output does not indicate the faults are cleared. • It does indicate the request was made. •

FBD Language Example



FFLD Language Example

Reset the Fault of the Drive for the current Axis when the button is pressed



IL Language Example

Not available.

ST Language Example

```
(* MC_ClearFaults ST example *)  
MC_ClearFaults( Axis1);    //clear drive faults for Axis 1
```

See Also

"MC_ResetError" (→ p. 625)

3.2.8.2 MC_CreatePLCAxis



Function - Creates a PLCopen Axis.

- A call to this function is automatically generated when the application is compiled, based on the data entered in the PLCopen Axis Data dialog.
- This function returns FALSE when the number of axes created in the application is greater than the maximum number permitted for the controller model.
 - When this axis limit is exceeded, the controller generates error E45.
 - This error is shown on the 7-segment display and on the Diagnostics tab of the Web server.
 - See E45 for an explanation and recommended corrective actions.
- The maximum axis count specified for each controller model applies only to physical axes.
 - This includes both PLCopen and Pipe Network axes.
- This limit **does not** apply to virtual axis or digitizing axis.
- **MC_CreateAxis** must be called between "MLMotionInit" (→ p. 867) and "MLMotionStart" (→ p. 872).

Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	When TRUE, requests to create a PLCopen axis.
AxisName	STRING	No range	N/A	No default	Axis name.
BusInterface	STRING	No range	N/A	No default	Bus interface identifier: • EtherCATDriver = EtherCAT interface. • MSBusDriver = KAS Simulator interface.
BusAddress	DINT	Bus dependent	N/A	No default	Address of the drive on the bus.
AxisNumber	UINT	1 to 256	N/A	No default	Axis number.

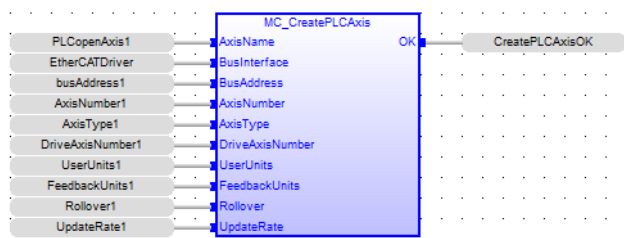
Input	Data Type	Range	Unit	Default	Description		
AxisType	USINT	Enumerated	N/A	No default	One of these enumeration values:		
					USINT	Type	Description
					0	MC_AXIS_TYPE_SERVO_STEPPER	A Servo or Stepper axis can be mapped to a physical or simulated drive. Either a servo or stepper drive is supported.
					1	MC_AXIS_TYPE_DIGITIZING	A digital position input from a drive or other device. It is useful as an input for gearing, camming, etc.
2	MC_AXIS_TYPE_VIRTUAL	A virtual axis cannot be mapped to a physical or simulated drive. It is useful for generating motion trajectory as an input for gearing, camming, etc.					
DriveAxisNumber	UINT	1 to 256	N/A	No default	This one-based number specifies the axis on the drive. For a single-axis drive this number should be 1.		
UserUnits	DINT	1 to 2147483647	User units	No default	User unit portion of the user unit/feedback unit ratio.		

Input	Data Type	Range	Unit	Default	Description
FeedbackUnits	DINT	1 to 2147483647	Feedback units	No default	Feedback unit portion of the user unit/feedback unit ratio.
Rollover	LREAL	0 to 4294967296	User units	No default	Rollover position. 0 (zero) = no rollover.
UpdateRate	UINT	3 to 9	N/A	No default	Servo update rate. 0, 1, and 2 are reserved. 3 = 125 µsec 4 = 250 µsec 5 = 500 µsec 6 = 1 msec 7 = 2 msec 8 = 4 msec 9 = 8 msec

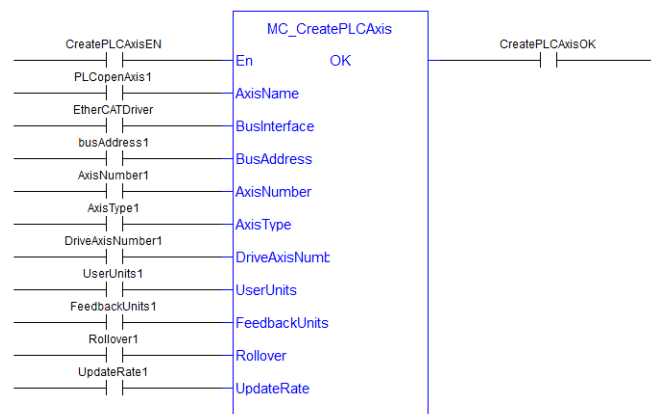
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Indicates the axis has been created.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_CreatePLCAxis ST Example *)
AxisName1      := 'PLCOpenAxis1';
BusName1       := 'EtherCATDriver';
BusAddress1    := 1001;
AxisNumber1    := 1;
AxisType1      := MC_AXIS_TYPE_SERVO_STEPPER;
DriveAxisNumber1 := 1;
UserUnits1     := 360;
FeedbackUnits1 := 1048576;
Rollover1      := 0;
UpdateRate1    := 3;
MC_CreateAxis(AxisName1, BusName1, BusAddress1, AxisNumber1, AxisType1,
DriveAxisNumber1, UserUnits1, FeedbackUnits1, Rollover1, UpdateRate1);
```

3.2.8.3 MC_EStop

PLCopen 



Function - Causes an emergency stop (E-stop).

- Call **MC_EStop** to generate an emergency stop for an axis.
- Call "MC_ResetError" (→ p. 625) to reset the emergency stop.
- Actions of an E-stop:
 - Stops motion interpolation.
 - Clears all moves from the queue (active and next).
 - Changes the axis state to ErrorStop.
 - See PLCopen State Machine for more information.
 - Request the drive to open the position loop and disable the drive.
- The E-stop remains in effect until the application calls "MC_ResetError" (→ p. 625) to reset the E-stop.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

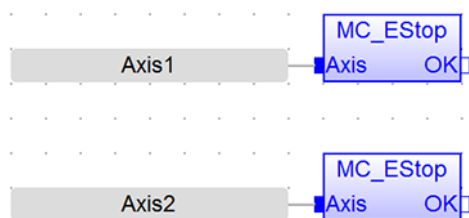
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	A positive transition of this input causes an E-stop on the specified axis.
Axis	AXIS_REF	1 to 256	N/A	No default	The AXIS_NUM element of the AXIS_REF structure.

Outputs

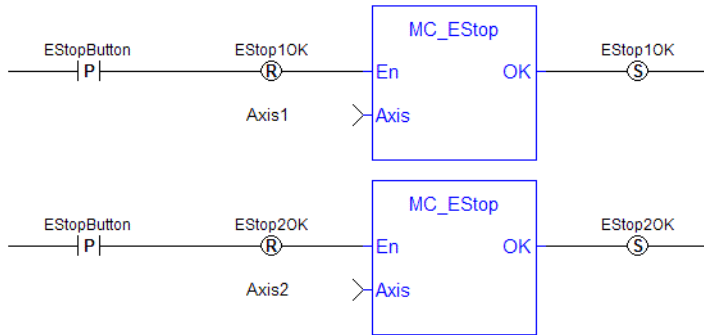
Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Indicates the E-stop was executed. • If an invalid Axis input was specified, this output is not energized and no E-stop is performed.

FBD Language Example



FFLD Language Example

E-stop both axes when the E-stop button is pressed



IL Language Example

Not available.

ST Language Example

```
(* MC_EStop ST example *)
ON EStopButton DO
  MC_EStop( Axis1 );
  MC_EStop( Axis2 );
END_DO;
```

See Also

"MC_ResetError" (→ p. 625)

3.2.8.4 MC_InitAxis



Function - Initializes a PLCopen Axis' data.

- A call to this function is automatically generated when the application is compiled.
- This is based on the data entered in the PLCopen Axis Data dialog.

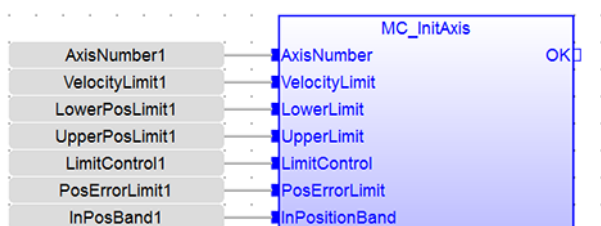
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	When TRUE, requests to create a PLCopen axis.
AxisNumber	UINT	1 to 256	N/A	No default	Axis number.
VelocityLimit	LREAL	No range	User units / sec	No default	Reserved.
LowerLimit	LREAL	No range	User units	No default	Reserved.
UpperLimit	LREAL	No range	User units	No default	Reserved.
LimitControl	UINT	0 to 2	N/A	No default	Reserved.
PosErrorLimit	LREAL	No range	User units	No default	Position error limit: When the Position Error (command position - actual position) exceeds this value, an E-stop is generated.
InPositionBand	LREAL	No range	User units	No default	In-position bandwidth: When the axis actual position is within this distance from its programmed endpoint, the axis is considered in position.

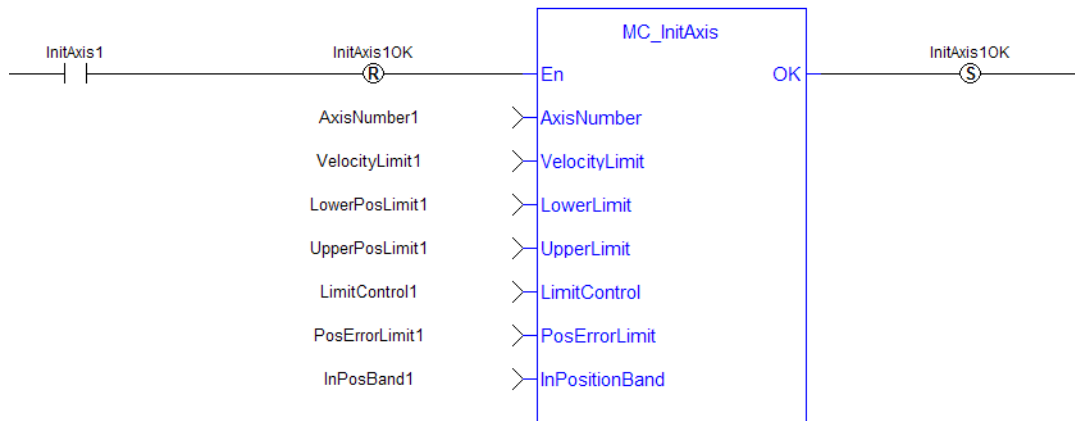
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Indicates the initialization is complete.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_InitAxis ST example *)
AxisNumber1      := 1;
VelocityLimit1   := 10000; (*User unit/second*)
LowerPosLimit1   := 0;
UpperPosLimit1   := 0;
LimitControl1    := 0; (* Ignore lower and upper pos limit*)
PosErrorLimit1   := 10; (*User unit*)
InPosBand1       := 0;
MC_InitAxis(AxisNumber1, VelocityLimit1, LowerPosLimit1, UpperPosLimit1,
LimitControl1, PosErrorLimit1, InPosBand1);
```

3.2.8.5 MC_InitAxisFeedback



Function - Initializes a PLCopen digitizing Axis' position data.

- A call to this function is automatically generated when the application is compiled.
- This call is based on the data entered in the PLCopen Axis Data dialog box.

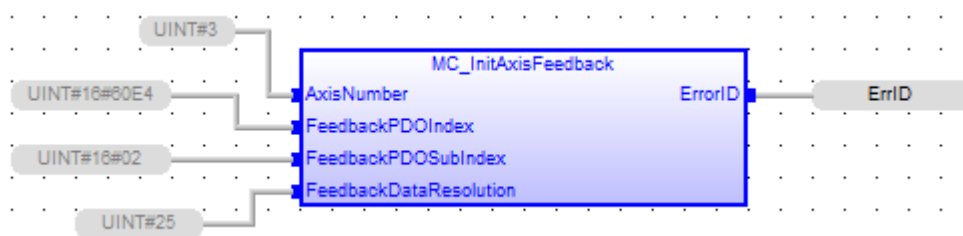
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	• Request to Initialize a PLCopen Digitizing Axis. • This input is only applicable to FFLD programs.
AxisNumber	UINT[] UINT	1 to 256	N/A	No default	Digitizing Axis number.
FeedbackPDOIndex	UINT	0 to 65536	N/A	No default	The object Index through which the feedback data is sent.
FeedbackPDOSubIndex	UINT	0 to 255	N/A	No default	The sub index of the object through which the feedback data is sent to the controller.
FeedbackDataResolution	UINT	1 to 32	N/A	No default	The resolution of the Encoder.

Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Indicates the initialization is complete. • This output is only applicable to FFLD programs. •
ErrorID	DINT	No range	N/A	Indicates if the call failed or succeeded. If the call succeeds, this is set to 0 (zero).

FBD Language Example



FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* MC_InitAxisFeedback ST example *)  
// Initialize the digitizing Axis (Axis #3) with the Feedback object 0x60E4  
subIndex 2.  
Encoder resolution is 25bits.  
ErrID := MC_InitAxisFeedback(3, 16#60E4, 16#02, 25);
```

3.2.8.6 MC_Power



 **Function Block** - Requests to either enable the drive and close the loop, or disable the drive and open the loop.

- The Status output indicates the state of the position loop.
 - If the position loop is open, the axis command position is set to the actual position of the axis and tracks the actual position.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

NOTE

- Be careful if there is more than one instance of **MC_Power** FB for the same drive, scanned in the same cycle.
- The problem arises when one instance requests the drive to enable and the other requests the same drive to disable.
- To avoid this trap, it is recommended to have only one instance of **MC_Power** for all active programs.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	When this transitions goes to high: • The control closes the servo loop. • Sends a command to the drive to enable. When this transitions go to low: • The control opens the servo loop. • Sends a command to the drive to disable.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
EnablePositive	BOOL	FALSE, TRUE	N/A	No default	Unused.
EnableNegative	BOOL	FALSE, TRUE	N/A	No default	Unused.
BufferMode	SINT	0 (zero)	N/A	No default	Unused.

Outputs

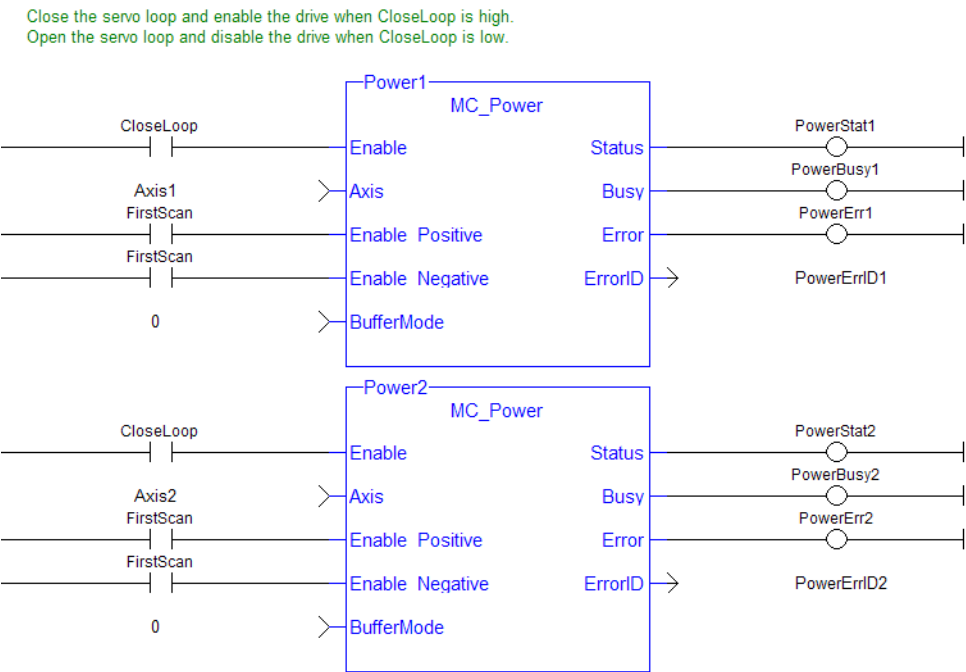
Output	Data Type	Range	Unit	Description
Status	BOOL	FALSE, TRUE	N/A	Indicates the enabled/disabled state of the drive.
Busy	BOOL	FALSE, TRUE	N/A	Always FALSE.
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified.

Output	Data Type	Range	Unit	Description
ErrorID	INT	Enumerated	N/A	• Indicates the error if Error output is TRUE.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_Power ST example *)
Inst_MC_Power( CloseLoopReq, Axis1, TRUE, TRUE, 0 );
//Inst_MC_Power is an instance of MC_Power function block
DriveIsOn := Inst_MC_Power.Status;      //store the Status output into a user
defined variable
```

3.2.8.7 MC_ResetError

PLCopen 



Function - Resets the errors of the specified axis.

- Makes the transition from the state ErrorStop to StandStill by resetting all internal axis-related errors and clearing pending commands.
- This does not affect the output of the FB instances.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

This function performs these tasks in sequence:

1. Sends a request to the drive to clear any drive faults that exists
2. Resets the axis errors

NOTE

- The condition causing the axis error has to be corrected before calling this function.
- The axis error still remains until the error condition exists when this function is called.
- See Transition Events and Actions, #15 in the status machine of the CANopen protocol.

Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	Requests to reset the axis errors.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.

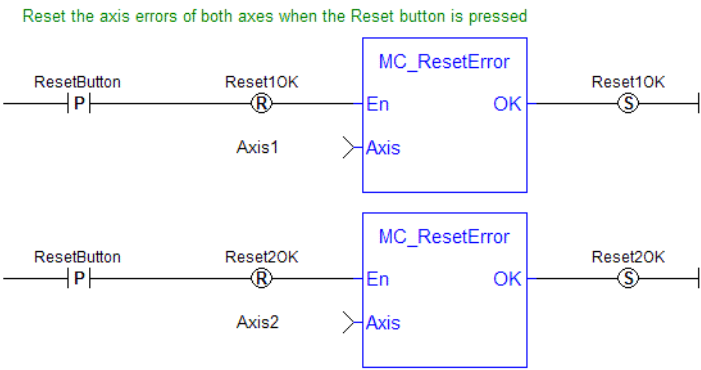
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Function or function block executed successfully.

FBD Language Example

Not available.

FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//reset the axis and drive errors for Axis 1  
  
MC_ResetError( Axis1 );
```

3.2.8.8 MC_Stop



 **Function Block** - Aborts the active move, removes the next move from the queue, performs a controlled stop, and switches the axis to Stopping state.

-
- An MC_Stop cannot be aborted.
 - This means that, while in Stopping state, no function block can command any motion on the axis.
 - The axis remains in Stopping state until it reaches zero velocity and the Execute input is low.
- The application program can hold the axis in Stopping state even after it reaches zero velocity by leaving the Execute input high.
- It aborts any ongoing function block execution.
- When the Done output is set, the state transfers to **StandStill**.
- While the axis is in **Stopping** state, no other FB can perform any motion on the same axis.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to stop the axis. It can be held high to prevent any other moves from being queued.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
Deceleration	LREAL	No range	User units / sec ²	No default	Trapezoidal: Deceleration rate. • S-curve: Unused.
Jerk	LREAL	No range	User units / sec ³	No default	Trapezoidal: 0 (zero). • S-curve: Constant jerk.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates both: • The axis has reached zero velocity. • The Execute input is low.
Busy	BOOL	FALSE, TRUE	N/A	High from the time both: • The Execute input goes high until the axis reaches zero velocity. • The Execute input is low.

Output	Data Type	Range	Unit	Description
Active	BOOL	FALSE, TRUE	N/A	High from the time both: • The MC_Stop move becomes the active move until the axis reaches zero velocity. • The Execute input is low.
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified.
ErrorID	INT			• Indicates the error if Error output is TRUE.

Remarks

Time Diagram

This example shows the behavior of the combination of a MC_Stop FB with a "MC_MoveVelocity" (→ p. 685).

- A rotating axis is ramped down with FB2 MC_Stop
- The axis rejects motion commands as long as MC_Stop parameter "Execute" = TRUE

FB1 MC_MoveVelocity reports an error indicating the busy MC_Stop command.

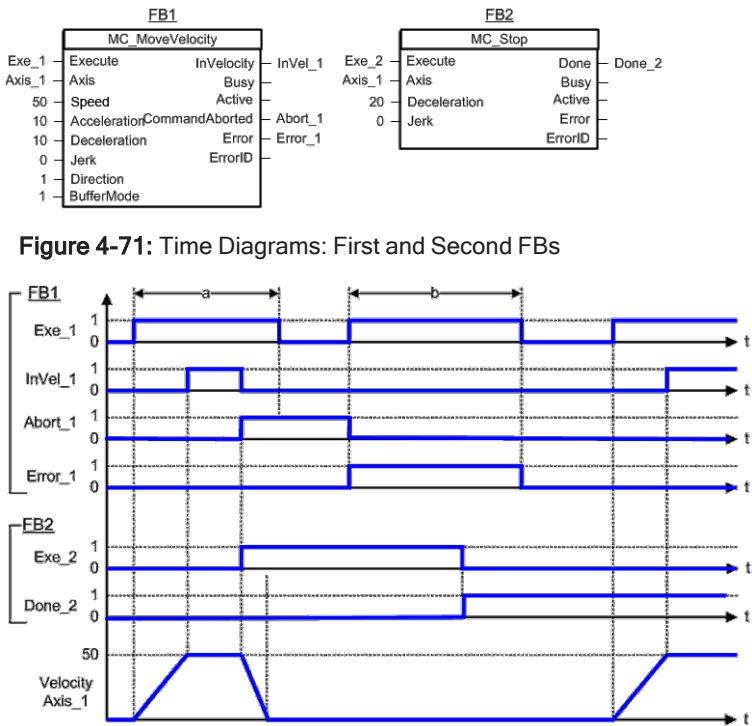


Figure 4-71: Time Diagrams: First and Second FBs

Figure 4-72: Time Diagram

FBD Language Example

Not available.

FFLD Language Example

Put Axis 1 into Stopping Mode



IL Language Example

Not available.

ST Language Example

```
(* MC_Stop S
T example *)

Inst_MC_Stop( StopRequest , Axis1, 100.0, 100.0 ); //Inst_MC_Stop is an
instance of MC_Stop function block

StopComplete := Inst_MC_Stop.Done;           //store the Done output into a user
defined variable

StopActive := Inst_MC_Stop.Active;           //store the Active output into a user
defined variable

StopError := Inst_MC_Stop.Error;             //store the Error output into a user
defined variable
```

3.2.9 I/O

These functions provides I/O control over TouchProbe functions.

Name	Description
MC_AbortTrigger	When the Execute input transitions from low to high, this function block aborts an MC_TouchProbe function block.
MC_TouchProbe	Arms a Fast Input and returns the latched position when the Fast Input event occurs.

3.2.9.1 MC_AbortTrigger



 **Function Block** - When the Execute input transitions from low to high, this function block aborts an MC_TouchProbe function block.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to abort the "MC_TouchProbe" (→ p. 633) function block.
Axis	AXIS_REF	1 to 256	N/A	No default	Identifies the axis specified in the MC_TouchProbe function block which is to be aborted.
TriggerInput	TRIGGER_REF	INT	N/A	No default	Identifies the fast input specified in the MC_TouchProbe function block which is to be aborted. The structure elements are: Direction INT Range = 1 to 2. • 1 = Rising edge. • 2 = Falling edge. Valid values of InputID are: • 0 = Touch Probe 1 / Capture Engine 0 • 1 = Touch Probe 2 / Capture Engine 1 Trigid INT Range = 0 to 256. • The axis number of the input. • 0 (zero) indicates the trigger axis is to be the same as Axis.AXIS_NUM.

NOTE

TrigMode INT (TriggerInput.TrigMode) is not presently supported by this function.

Outputs

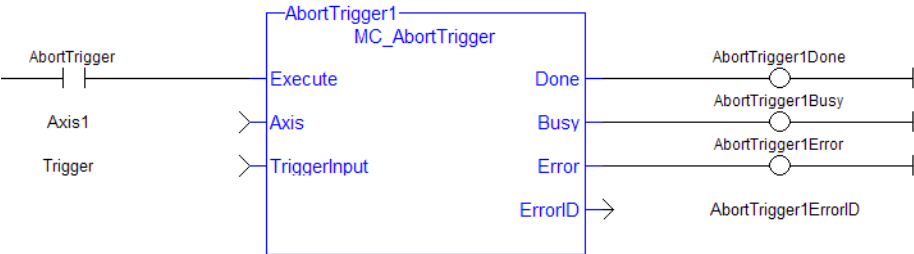
Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Function block has completed.

Output	Data Type	Range	Unit	Description
Busy	BOOL	FALSE, TRUE	N/A	Indicates this function block is executing.
Error	BOOL	FALSE, TRUE	N/A	Indicates the function block did not complete due to an error. The ErrorID output indicates the type of error when this output is high.
ErrorID	INT		N/A	When the Error output is: • High, this output indicates the type of error. • Low, this output is undefined.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_AbortTrigger ST example *)
Inst_MC_AbortTrigger( AbortReq, Axis1, TriggerInputRef );
//Inst_MC_AbortTrigger is an instance of MC_AbortTrigger
```

See Also

"MC_TouchProbe" (→ p. 633)

3.2.9.2 MC_TouchProbe



Function Block - Arms a Fast Input and returns the latched position when the Fast Input event occurs.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.
- This function or function block does not generate any motion.
- This function block captures an axis position
- When the Execute input transitions from low to high, the control requests the drive to arm its Fast Input to latch the axis position when a Fast Input occurs.
- The Axis input specifies which axis's position to latch and the TriggerInput input specifies which Fast Input to use and whether to trigger on the rising or falling edge of the Fast Input.
- When the Fast Input event occurs, the drive latches the axis's position.
 - This function block then returns the latched position at the RecordedPosition output and set the Done output high.
 - This process can be canceled with the AbortTrigger function block.
- If the WindowOnly input is high, the FirstPosition input and the LastPosition input define a window in which a Fast Input is accepted.
- Any Fast Input events that occur outside the window is ignored.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to arm the input to return the latched position.
Axis	AXIS_REF	1 to 256	N/A	N/A	Selects the axis the position is latched to.
TriggerInput	TRIGGER_REF	See Description.	N/A	No default	• Sets up the mechanism on the controller for the capture input signal. • An instance of the TRIGGER_REF function must first be setup in the Project Dictionary, as seen here: • See "Elements" (→ p. 635).
WindowOnly	BOOL	FALSE, TRUE	N/A	No default	Enables a position latching window. • When this input is set, a window is defined by the FirstPosition and LastPosition inputs. • Any Fast Input event that occurs outside the window is ignored. • The first Fast Input event that occurs within the window latches the axis position.

Input	Data Type	Range	Unit	Default	Description
FirstPosition	LREAL	No range	User units	No default	See the "WindowOnly" (→ p. 633) Description about how this input and the LastPosition input define the window. • This input is only applicable when the WindowOnly input is high. • If the WindowOnly input is low, this input is ignored.
LastPosition	LREAL	No range	User units	No default	See the "WindowOnly" (→ p. 633) Description about how this input and the FirstPosition input define the window. • This input is only applicable when the WindowOnly input is high. • If the WindowOnly input is low, this input is ignored.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	• Function block has completed. • The RecordedPosition output is valid.
Busy	BOOL	FALSE, TRUE	N/A	Indicates the specified input is: • Arming or is armed. • Waiting for the trigger and recording of the position to occur.
CommandAborted	BOOL	FALSE, TRUE	N/A	• If TRUE, the command was aborted by another function block. • A TriggerAbort function block has executed and canceled this function.
Error	BOOL	FALSE, TRUE	N/A	• The function block has not completed successfully due to an error. • The ErrorID output indicates the type of error.
ErrorID	INT			• When the Error output is high, this output indicates the type of error. • When the Error output is low, this output is undefined.
RecordedPosition	LREAL		User units	• When the Done output goes high, this output returns the latched position. • When the Done output is low, this output is undefined.

Remarks

❗IMPORTANT

There are differences in how **MC_TouchProbe** interacts with AKD and AKD2G drives. See these topics for drive-specific information:

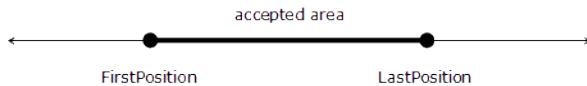
- "AKD Support with MC_TouchProbe" (→ p. 638)
- "AKD2G Support With MC_TouchProbe" (→ p. 639)

TIP

- The accuracy of captured position data depends on the travel velocity.
- See this article [MC_TouchProbe and Time-Based Capture](#) about how to correct for timing.

Example 1

If $\text{FirstPosition} \leq \text{LastPosition}$, the window in which a Fast Input is accepted is:

$$\text{FastInputPosition} \geq \text{FirstPosition} \text{ AND } \text{FastInputPosition} \leq \text{LastPosition}$$
**Example 2**

If $\text{FirstPosition} > \text{LastPosition}$, the window in which a Fast Input is accepted is:

$$\text{FastInputPosition} \geq \text{FirstPosition} \text{ OR } \text{FastInputPosition} \leq \text{LastPosition}$$
**3.2.9.2.1.1 Elements****Capture Engine**

This is the drive capture engine to be used.

INT TriggerInput.InputID

Valid values of **InputID** are:

- 0 = Touch Probe 1 / Capture Engine 0
- 1 = Touch Probe 2 / Capture Engine 1

Trigger Direction

This is the input signal's edge to capture.

INT TriggerInput.Direction

Range = 1 to 2

- 1 = Rising edge.
- 2 = Falling edge.

TIP

Trigger Direction is also sent to the servo drive and is shown in the WorkBench Position Capture screen **Edge** setup.

Axis Number

This is where the input comes from.

INT TriggerInput.TrigID

Range = 0 to 256.

0 = Trigger axis is to be the same as Axis.AXIS_NUM.

Trigger Mode

This is the capture method.

INT TriggerInput.TrigMode

Range = 0 to 1

- 0 = Time based capture.
- 1 = Position based capture.
 - For position based capture, the TrigID must be the same as the AXIS_REF.

NOTE

The Mode (either Time or Position) must be configured the same in the servo drive.
This can be done either in:

- COE Init commands and executed when the EtherCAT network is initialized (0x3460, subindex 3 and 4).
- The WorkBench Positon Capture screen.

3.2.9.2.2.2 Usage

This function block can be used to:

- Perform registration.
- Determine the position of a product.
- Measure product length.

3.2.9.2.3.3 Limitations

- Both high speed inputs cannot be used at the same time.
- The TrigMode option is only used by MC_TouchProbe.

3.2.9.2.4.4 Fast Homing - Inputs

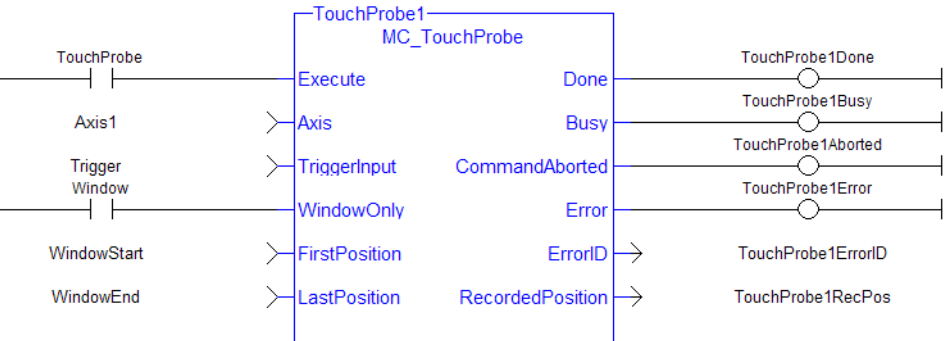
See these topics:

- Example: Fast Homing with the Pipe Network Motion Engine Axis Pipe Block
- Example: Fast Homing with the PLCopen Motion Engine
- Fast Inputs with Pipe Network Motion
- H1-Pipe Network Registration and Fast Homing
- Example: Registration Position Capture with Pipe Network Trigger Block

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_TouchProbe ST example *)
TriggerInputRef.InputID := 1;    //configure InputID
TriggerInputRef.Direction := 1;  //configure Direction
TriggerInputRef.TrigID := 0;     //configure TrigID
TriggerInputRef.TrigMode := 0;   //Capture trigger based on distributed
clock time
Inst_MC_TouchProbe( ArmProbe, Axis1, TriggerInputRef, FALSE, 0.0, 0.0 );
//Inst_MC_TouchProbe is an instance of MC_TouchProbe function block
ProbeIsDone := Inst_MC_TouchProbe.Done;           //store Done output into
a user defined variable
ProbeValue := Inst_MC_TouchProbe.RecordedPosition; //store RecordedPosition
output into a user defined variable
```

See Also

"MC_AbortTrigger" (→ p. 631)

3.2.9.2.5 AKD Support with MC_TouchProbe

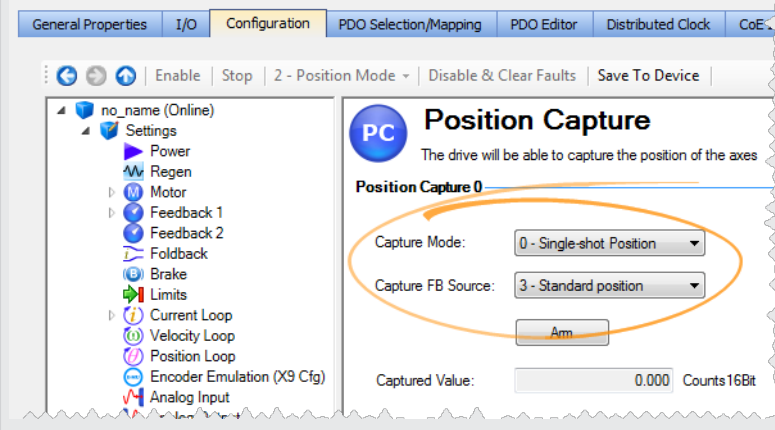
These are tips related to using MC_TouchProbe with AKD drives.

TIP

To use Capture Engine 1, modify the input PDOs that are used and add the Latch Position 1 parameter.

TIP

When using position-based capture, the proper Capture Mode and FB Source may need to be set up in the drive.
One place to do that is in the Position Capture Screen in the KAS-IDE embedded WorkBench:



TIP

When setting up Position Capture, check the CoE-Init Command settings.
This is to verify they do not overwrite the corresponding drive parameters with unwanted values when the EtherCAT network initializes.

Pre Operational -> Safe Operational						
Index	Subindex	Value	Comment	Direction	Source	
0x6060	0	7	Opmode	Write	ESI File	
0x60C2	1	1	Cycle time	Write	ESI File	
0x60C2	2	-3	Cycle exp	Write	ESI File	
0x3460	1	0	Latching engine 0 config to FI0, CAP0.TRIGGER=0, 0x3460-1:=0 (1 byte)	Write	ESI File	
0x3460	2	1	Latching engine 1 config to FI1, CAP1.TRIGGER=1, 0x3460-2:=1 (1 byte)	Write	ESI File	
0x60FE	2	196608	Digital Outputs Mask, 0x60fe:=0x30000 (4 bytes)	Write	ESI File	
0x36E6	0	1	Set FBUS.PARAM02 to activate synchronization with the interrupt	Write	ESI File	
0x36E8	0	1	Set FBUS.PARAM04 to disabled the drive on a motion error	Write	ESI File	
0x3506	0	1	Set DRV.HWENMODE to disabled the rising edge of the hardware enable from clearing the drive faults	Write	ESI File	
0x35CA	0	1048576	Set UNIT.PIN to set gear IN to the correct unit conversion.	Write	ESI File	
0x365F	0	0	Set UNIT.VROTARY to set the velocity units to RPM.	Write	ESI File	
0x3460	3	2	Capture mode to distributed clock time (DCT), CAP0.MODE=2, 0x3460-3:=0 (1 byte)	Write	ESI File	
0x3460	4	2	Capture mode to distributed clock time (DCT), CAP1.MODE=2, 0x3460-4:=0 (1 byte)	Write	ESI File	
0x50E2	0	1000	Set ILKBUSFF (1.0 = 1000) UINT32	Write	ESI File	
0x3498	0	1	Set FBUS.PROTECTION (available only since FW 01-07-03-000)	Write	ESI File	

3.2.9.2.6 AKD2G Support With MC_TouchProbe

These are tips related to using **MC_TouchProbe** with AKD2G drives.

NOTE

AKD2G only supports position capture.

- AKD2G does not use CAP1 or CAP2 to provide the EtherCAT touch probes.
 - AKD2G supports ETG6010 and DS402 for Touch Probe objects.

Two touch probes per axis are supported over EtherCAT with their own dedicated hardware in the drive.

Each touch probe can capture two positions, the position on the rising and the position of the falling edge of the trigger input.

These are the standards-compliant ETG6010 and DS402, EtherCAT / CANopen objects AKD2G supports.

Axis 1 Index	Axis 2 Index	Name	Note
60B8h	68B8h	Touch probe function / control.	
60B9h	68B9h	Touch probe status.	
60BAh	68BAh	Touch probe position 1 positive value.	AXIS#.PL.FB, Scaling same as axis
60BBh	68BBh	Touch probe position 1 negative value.	
60BCh	68BCh	Touch probe position 2 positive value.	
60BDh	68BDh	Touch probe position 2 negative value.	
60D0H	68D0h	Touch probe source.	

- The KAS-IDE pre-populates these PDOs with the required Touch probe objects by default.
 - Rx PDO 0x1602 with the required Touch Probe control objects.
 - Tx PDO 0x1A02 with the required Touch Probe status and position value objects.

Index	Subindex	Object Name	Size [bit]	PLC Variable
0x60B9	0	Touch probe status - Axis 1	16	
0x60BA	0	Touch probe 1 position positive value - Axis 1	32	
0x60BB	0	Touch probe 1 position negative value - Axis 1	32	
0x60BC	0	Touch probe 2 position positive value - Axis 1	32	
0x60BD	0	Touch probe 2 position negative value - Axis 1	32	

Index	Subindex	Object Name	Size [bit]	PLC Variable
0x60B8	0	Touch probe function - Axis 1	16	
0x60B8	0	Touch probe function - Axis 2	16	

- The Trigger input source is set by sending an SDO command.
 - Axis1:
 - 0x60D0 sub Index 1 for Touch Probe 1 Source.
 - 0x60D0 sub Index 2 for Touch Probe 2 Source.

- Axis2:
0x68D0 sub Index 1 for Touch Probe 1 Source.
0x68D0 sub Index 2 for Touch Probe 2 Source.
- 6#D0h, Touch Probe Source.
This table shows how AKD2G signals are mapped to the touch probe source entry in the object dictionary.
 - A few sources appear in both the standard and the manufacture ranges to provide some consistency.

DS402 & ETG6010 Values	Text from Standard	AKD2G Values for 6#D0h	Equivalent CAP#.TRIGGER	AKD2G Note
-32768 to -1	Manufacturer specific	-41 to -42	41 to 42	Z pulse for Axis 1 to 2
		-31 to -35	31 to 35	Z pulse for Feedback 1 to 5. FB1, 2, 4, and 5 do not support Z pulses; these are not shown. When we support SFA on FB 1 and 2 then Z pulse may be possible. X23 is optional so if not fitted then -33 is not valid.
		-21 to -26	21 to 26	DIO1 to DIO6 • When X22 is not fitted options -21 and -22 are not valid. • When X23 is not fitted options -23 to -26 are not valid.
		-1 to -12	1 to 12	DIN1 to DIN12 When X22 is not fitted options -9 to -12 are not valid.
0	Reserved			Not valid.
1	Digital Input 1 (Touch Probe input)	1	1	DIN1. Fast Opto
2	Digital Input 2 (Touch Probe input)	2	2	DIN2. Fast Opto
3	Digital Input 3 (Touch Probe input)			Not valid.
4	Digital Input 4 (Touch Probe input)			Not valid.
5	Hardware zero pulse signal of position encoder	5	41 for Axis 1 42 for Axis 2	Valid if PL.FBSOURCE is using a feedback that supports a Z pulse.
6	Software zero pulse encoder			Not valid.
7 to 32767	Reserved			Not valid.

3.2.10 Information

These functions provide feedback and are used to write parameters.

Name	Description
MC_ReadActPos	Reads the actual position of the axis.
MC_ReadActVel	Reads the actual velocity of the axis.
MC_ReadAxisErr	Returns the error status of the specified axis.
MC_ReadBoolPar	Returns the value of the specified Boolean axis parameter.
MC_ReadParam	Returns the value of the specified axis parameter.
MC_ReadStatus	Returns the state of the specified axis.
MC_WriteBoolPar	Writes the specified axis Boolean parameter.
MC_WriteParam	Writes the specified axis parameter.

3.2.10.1 MC_ReadActPos



Function Block - Reads the actual position of the axis.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	• Request to read the axis's actual position. • Keeps continuously reading the actual position every PLC cycle, as long as the Enable remains high.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.

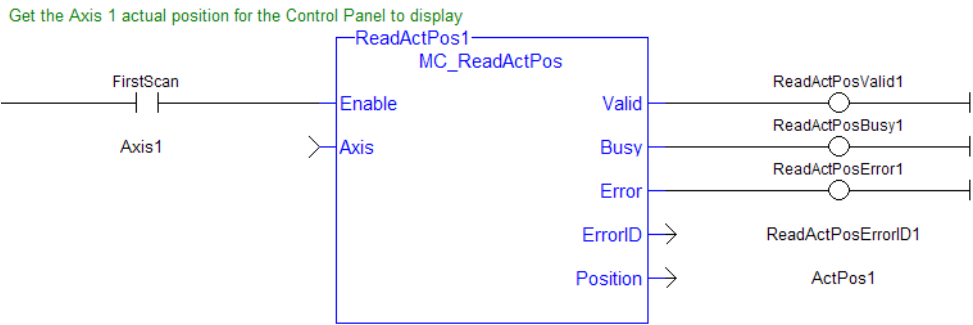
Outputs

Output	Data Type	Range	Unit	Description
Valid	BOOL	FALSE, TRUE	N/A	Indicates the value at the Position output is available.
Busy	BOOL	FALSE, TRUE	N/A	Indicates this function block is executing.
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified.
ErrorID	INT	-32768 to +32767		• Indicates the error if Error output is TRUE.
Position	LREAL		User units	Actual position of the axis.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_ReadActPos S
   T example *)
Inst_MC_ReadActPos( TRUE, Axis1 );
//Inst_MC_ReadActPos is an instance of MC_ReadActPos function block

ActualPos := Inst_MC_ReadActPos.Position;
//store Position output into a user defined variable
```

3.2.10.2 MC_ReadActVel



 **Function Block** - Reads the actual velocity of the axis.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	Request to read the axis's actual velocity.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.

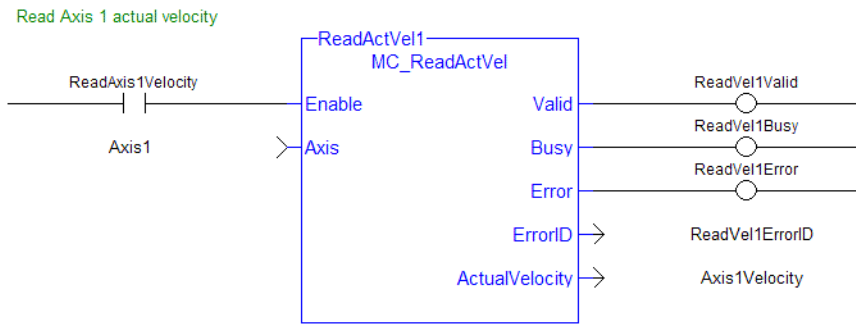
Outputs

Output	Data Type	Range	Unit	Description
Valid	BOOL	FALSE, TRUE	N/A	Indicates the value at the ActualVelocity output is available.
Busy	BOOL	FALSE, TRUE	N/A	Indicates this function block is executing.
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified.
ErrorID	INT	-32768 to +32767		• Indicates the error if Error output is TRUE.
ActualVelocity	LREAL		User units / sec	• Actual velocity of the axis. • Oscillations may be seen because this is an instant velocity, not an average velocity.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
* MC_ReadActVel S
T example *);
Inst_MC_ReadActVel( TRUE, Axis1 ); //Inst_MC_ReadActVel is an instance of MC_
ReadActVel function block

ActualVel := Inst_MC_ReadActVel.ActualVelocity; // store ActualVelocity
output into a user defined variable
```

3.2.10.3 MC_ReadAxisErr



 **Function Block** - Returns the error status of the specified axis.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	Request to read the error status of the axis.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.

Outputs

Output	Data Type	Range	Unit	Description
Valid	BOOL	FALSE, TRUE	N/A	Indicates the AxisErrorID output is valid.
Busy	BOOL	FALSE, TRUE	N/A	Indicates this function block is executing.
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified.
ErrorID	INT	-32768 to +32767		Indicates the error if Error output is TRUE.
AxisErrorID	INT	-32768 to +32767		Indicates the error status of the axis. • Each bit indicates a specific error. • Both emergency-stop (E-stop) and controlled-stop (C-stop) errors are indicated. • The "Output Bits" (→ p. 646) table defines the bits of this output.

3.2.10.3.1.1 Output Bits

Hexadecimal	Decimal	Description
0000H	0	No Error
0001H	1	User-set E-stop via MC_EStop, E-stop
0002H	2	Loss of Feedback, E-stop
0004H	4	Drive Fault, E-stop
0008H	8	Drive Communication Failure, E-stop

Hexadecimal	Decimal	Description
0400H	1024	Synchronization Error, C-stop
0700H	7192	- Drive Overtravel Limit Exceeded, Cstop. - See Overtravel Conditions.

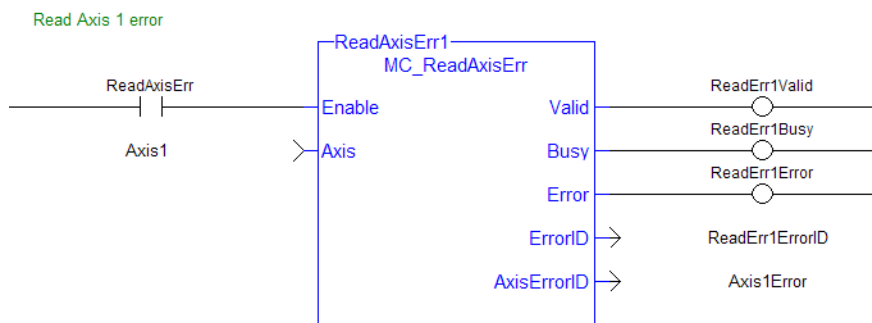
NOTE

- Multiple errors can be active at the same time.
- Example: If a User-set E-stop and an Excess Position Error E-stop are both active, the value is 00000011H (17 decimal).

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_ReadAxisErr ST example *)

Inst_MC_ReadAxisErr( TRUE, Axis1 );
//Inst_MC_ReadAxisErr is an instance of MC_ReadAxisErr function block
AxisErrorBits := Inst_MC_ReadAxisErr.AxisErrorID; //AxisErrorID contains the
error bits
```

3.2.10.4 MC_ReadBoolPar



 **Function Block** - Returns the value of the specified Boolean axis parameter.

Returns the value of a drive parameter.

- The returned value has to be converted to REAL if required.
- If not possible, the vendor must provide a supplier-dependent FB for it.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	Requests to read a Boolean axis parameter.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
ParameterNumber	INT	-32768 to +32767	N/A	No default	Parameter number. See Axis Parameters.

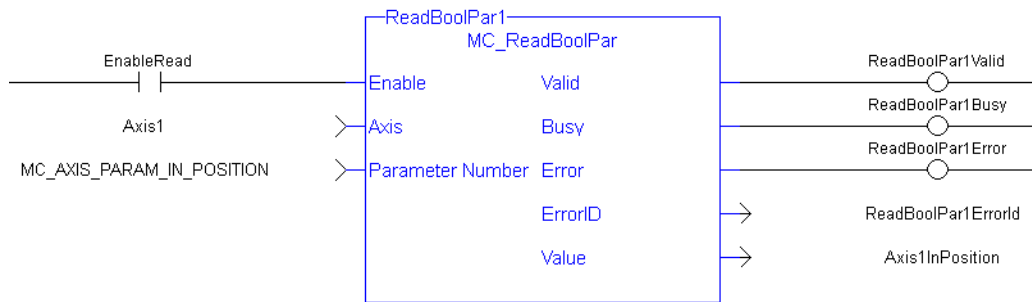
Outputs

Output	Data Type	Range	Unit	Description
Valid	BOOL	FALSE, TRUE	N/A	Indicates the Value output is valid.
Busy	BOOL	FALSE, TRUE	N/A	Indicates this function block is executing.
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified.
ErrorID	INT	-32768 to +32767		• Indicates the error if Error output is TRUE.
Value	BOOL	FALSE, TRUE	N/A	State of the Boolean parameter.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_ReadBoolPar ST example *)
Inst_MC_ReadBoolPar( EnableRead, Axis1, MC_AXIS_PARAM_IN_POSITION );
Axis1InPosition := Inst_MC_ReadBoolPar.Value;
```


3.2.10.5 MC_ReadParam



 **Function Block** - Returns the value of the specified axis parameter.

Returns the value of a drive parameter.

- The returned value has to be converted to REAL if required.
- If not possible, the vendor must provide a supplier-dependent FB for it.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	Request to read the axis parameter.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
ParameterNumber	INT	-32768 to +32767	N/A	No default	• Parameter number. • See Axis Parameters.

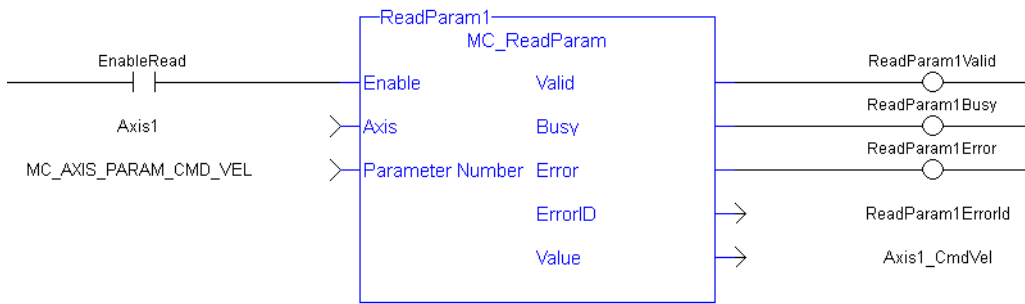
Outputs

Output	Data Type	Range	Unit	Description
Valid	BOOL	FALSE, TRUE	N/A	Indicates the Value output is valid.
Busy	BOOL	FALSE, TRUE	N/A	Indicates this function block is executing.
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified.
ErrorID	INT	-32768 to +32767		• Indicates the error if Error output is TRUE.
Value	LREAL			Value of the parameter.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_ReadParam ST example *)
Inst_MC_ReadParam( EnableRead, Axis1, MC_AXIS_PARAM_CMD_VEL );
Axis1_CmdVel := Inst_MC_ReadParam.Value;
```

3.2.10.6 MC_ReadStatus



Function Block - Returns the state of the specified axis.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	Request to read and return the axis status.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.

Outputs

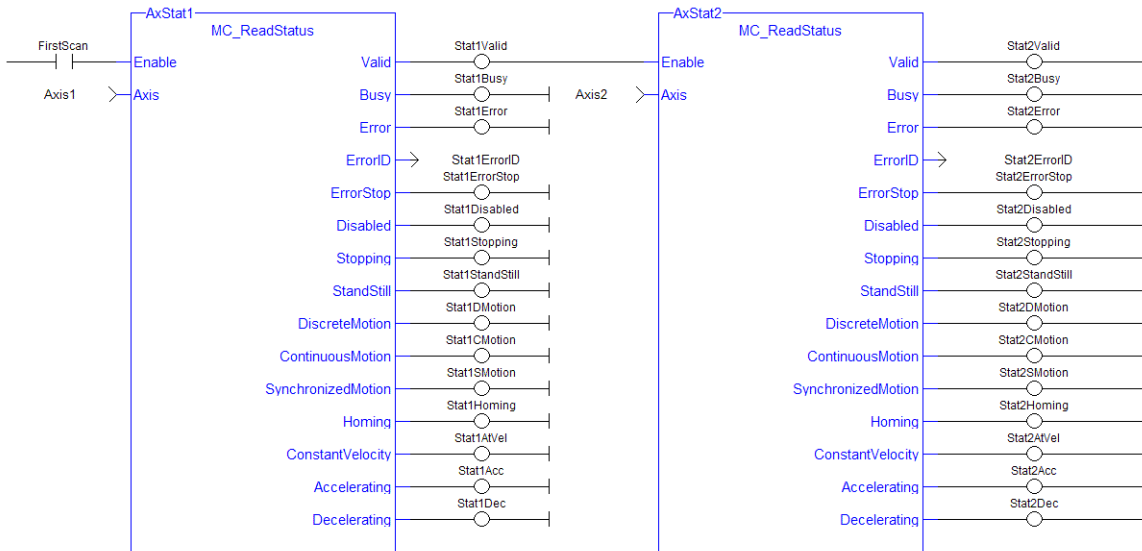
Output	Data Type	Range	Unit	Description
Valid	BOOL	FALSE, TRUE	N/A	Indicates the outputs are valid.
Busy	BOOL	FALSE, TRUE	N/A	Indicates this function block is executing.
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified.
ErrorID	INT	-32768 to +32767		• Indicates the error if Error output is TRUE.
ErrorStop	BOOL	FALSE, TRUE	N/A	• Indicates Error Stop state. • Either E-stop or C-stop.
Disabled	BOOL	FALSE, TRUE	N/A	• Indicates Disabled state. • Open loop and drive is disabled.
Stopping	BOOL	FALSE, TRUE	N/A	• Indicates Stopping state. • MC_Stop command.
StandStill	BOOL	FALSE, TRUE	N/A	• Indicates Stand Still state. • No move, closed loop, drive enabled.
DiscreteMotion	BOOL	FALSE, TRUE	N/A	• Indicates Discrete Motion state. • Pprogrammed endpoint move is active.
ContinuousMotion	BOOL	FALSE, TRUE	N/A	• Indicates Continuous Motion state. • Unending, single-axis move is active.
SynchronizedMotion	BOOL	FALSE, TRUE	N/A	• Indicates Synchronized Motion state • Slave move is active.
Homing	BOOL	FALSE, TRUE	N/A	• Indicates Homing state. • A homing cycle is currently executing.

Output	Data Type	Range	Unit	Description
ConstantVelocity	BOOL	FALSE, TRUE	N/A	Indicates the axis is moving at a constant velocity.
Accelerating	BOOL	FALSE, TRUE	N/A	Indicates the axis is accelerating.
Decelerating	BOOL	FALSE, TRUE	N/A	Indicates the axis is decelerating.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_ReadStatus ST example *)
```

```
Inst_MC_ReadStatus( EnableRead, Axis1 );
//Inst_MC_ReadStatus is an instance of MC_ReadStatus function block
AxisStopping := Inst_MC_ReadStatus.Stopping; // store Stopping output to a
user defined variable
AxisAccelerating := Inst_MC_ReadStatus.Accelerating; // store Accelerating
output to a user defined variable
```

3.2.10.7 MC_WriteBoolPar



 **Function Block** - Writes the specified axis Boolean parameter.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	Requests to write a Boolean axis parameter.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
ParameterNumber	INT	No range	N/A	No default	- Parameter number. - See Axis Parameters.
Value	BOOL	FALSE, TRUE	N/A	No default	State to write.

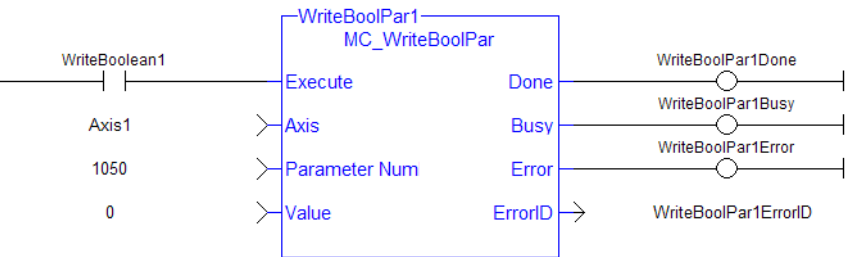
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the Boolean parameter has been written.
Busy	BOOL	FALSE, TRUE	N/A	Indicates this function block is executing.
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified.
ErrorID	INT			Indicates the error if Error output is TRUE.

FBD Language Example

Not available.

FFLD Language Example



NOTE

Currently, MC_WriteBoolPar does not support any parameters (1050 is an arbitrary number chosen for example).

IL Language Example

Not available.

ST Language Example

```
(* MC_WriteBoolPar ST example *)  
  
WriteBool := FALSE;  
Inst_MC_WriteBoolPar( WriteReq, Axis1, 1050, WriteBool );
```

3.2.10.8 MC_WriteParam



Function Block - Writes the specified axis parameter.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	Requests to write an axis parameter.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
ParameterNumber	INT	No range	N/A	No default	• Parameter number. • See Axis Parameters.
Value	LREAL	No range	N/A	No default	Value to write.

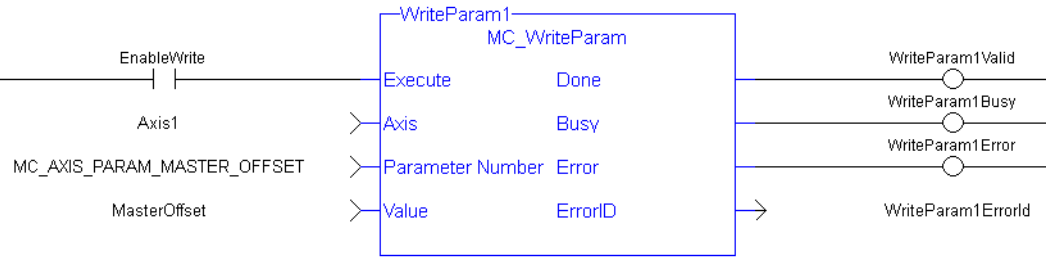
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the parameter has been written.
Busy	BOOL	FALSE, TRUE	N/A	Indicates this function block is executing.
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified.
ErrorID	INT	-32768 to +32767		• Indicates the error if Error output is TRUE.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_WriteParam ST example *)  
  
MasterOffset := 12.34;  
Inst_MC_WriteParam( EnableWrite, Axis1, MC_AXIS_PARAM_MASTER_OFFSET,  
MasterOffset);
```


3.2.11 PLCopen Motion

These functions provide control over an axis.

Name	Description
MC_AddSuperAxis	Adds a Superimposed Axis to the Axis's list of assigned superimposed axes.
MC_Halt	Decelerates an axis to zero velocity.
MC_MoveAbsolute	Performs a single-axis move to a specified endpoint position.
MC_MoveAdditive	Performs a single-axis move for a specified distance from the endpoint of the previous move.
MC_MoveContVel	Performs a single-axis non-ending move at a specified velocity with the option of continually updating the ongoing motion with the current input parameters.
MC_MoveRelative	Performs a single-axis move of a specified distance relative to the actual position at the time of the start of execution.
MC_MoveSuperimp	Performs a single-axis move superimposed upon the currently executing move.
MC_MoveVelocity	Performs a single-axis, non-ending move at a specified velocity.
MC_RemSuperAxis	Remove an axis from the axis's list of assigned, superimposed axes.
MC_SetOverride	Writes velocity and acceleration override factors.

3.2.11.1 MC_AddSuperAxis



 **Function Block** - Adds a Superimposed Axis to the Axis's list of assigned superimposed axes.

This feature allows the application program to superimpose the moves of multiple axes (Superimposed Axes) on top of the move of another axis (Receiving Axis).

- This is performed internally by adding the command deltas of the Superimposed Axes to the command delta of the Receiving Axis.
- A maximum of four different Superimposed Axes can be superimposed upon a Receiving Axis.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.
- While the Superimposed Axis is on this list, its command deltas are added to the Axis's command deltas.
- The **Axis** and the **SuperimposedAxis** must have the same update rate.
- The **OK** output goes high to indicate the function executed successfully.
- If the **OK** output **does not** go high, one of these errors is detected:
 - Axis and SuperimposedAxis do not have the same update rate.
 - Four different superimposed axes have already been assigned to Axis.
 - The axis is not a valid axis.
 - The axis is not a servo or virtual axis.
 - SuperimposedAxis is not:
 - A valid axis number.
 - A servo or virtual axis.
 - Axis could not acquire PLC motion engine lock.

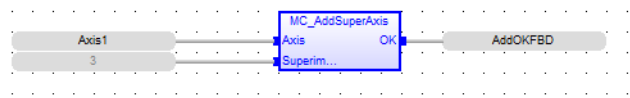
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	N/A	Enables execution.
Axis	AXIS_REF	AXIS_NUM 1 to 256	N/A	N/A	Axis to receive the additional superimposed axis's command delta.
SuperimposedAxis	UINT	1 to 256	N/A	N/A	Axis number of the superimposed axis whose command delta is added to delta of Axis.

Outputs

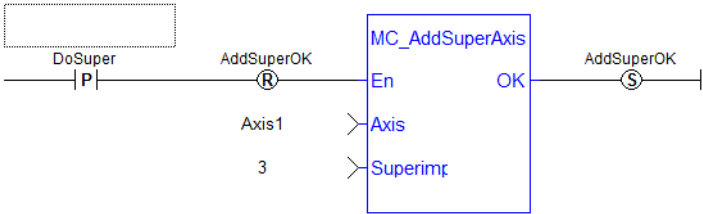
Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Execution successful.

FBD Language Example



FFLD Language Example

Network #10



IL Language Example

Not available.

ST Language Example

```
AddOKST := MC_AddSuperAxis( Axis1, 3);
```

See Also

"MC_RemSuperAxis" (→ p. 688)

3.2.11.2 MC_Halt



 **Function Block** - Decelerates an axis to zero velocity.

- It is a queued single-axis move.
 - The move is complete when the axis reaches zero velocity.
- It is typically used with Abort at the BufferMode input to terminate a move.
- See "MC_Stop" (→ p. 627) to execute a stop that cannot be aborted.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to queue the move.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
Deceleration	LREAL	No range	User units / sec ²	No default	• Trapezoidal: Deceleration rate. • S-curve: Unused.
Jerk	LREAL	No range	User units / sec ³	No default	• Trapezoidal: 0 (zero). • S-curve: Constant jerk.
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. • BM_ABORTING • BM_BUFFERED • BM_BLENDING_PREVIOUS • BM_BLENDING_NEXT • BM_BLENDING_LOW • BM_BLENDING_HIGH • See Buffer Modes.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input goes high until the time the move is ended.
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.

Output	Data Type	Range	Unit	Description
Error	BOOL	FALSE, TRUE	N/A	Indicates either: • An invalid input was specified. • The move was terminated due to an error.
ErrorID	INT	No range	N/A	• Indicates the error if Error output is TRUE.

Time Diagram

This example shows the behavior in combination with a "MC_MoveVelocity" (→ p. 685).

- A rotating axis is ramped down with FB2 MC_Halt.
- Another motion command overrides the MC_Halt command.
 - MC_Halt allows this, in contrast to MC_Stop.
- The axis can accelerate again without reaching standstill.

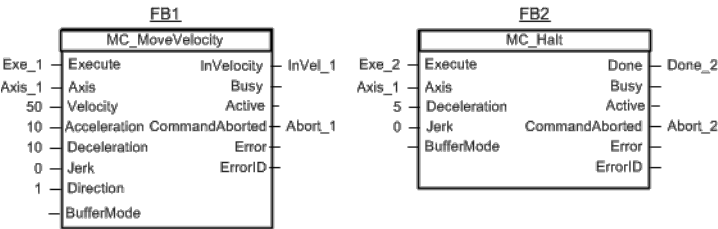


Figure 4-73: Time Diagrams: First and Second FBs

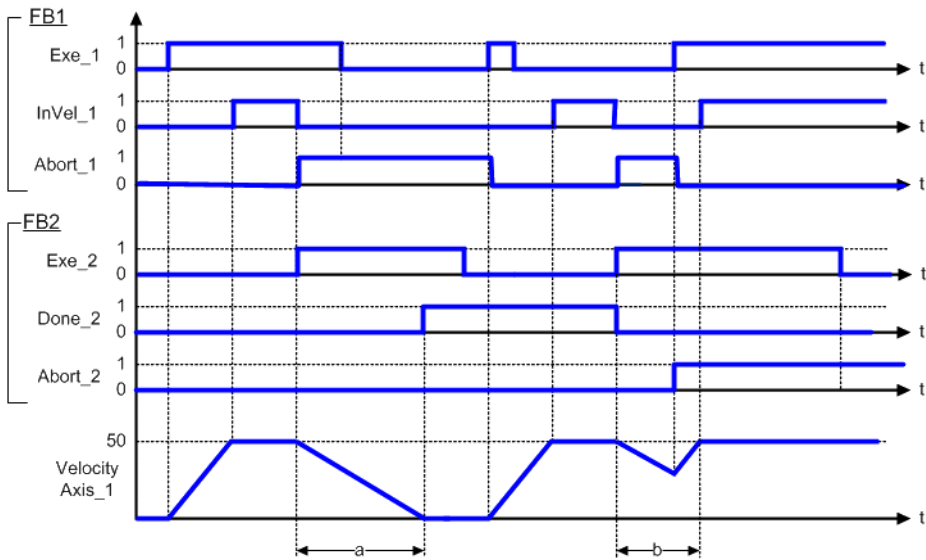


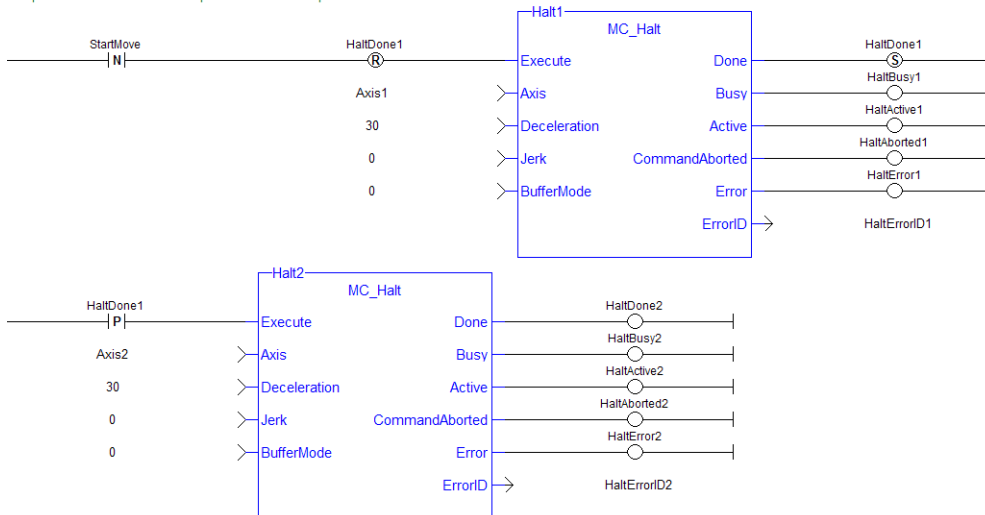
Figure 4-74: Time Diagram

FBD Language Example

Not available.

FFLD Language Example

Stop both axes when the Run/Stop switch is set to Stop



IL Language Example

Not available.

ST Language Example

```
(* MC_Halt ST example *)
Inst_MC_Halt( HaltReq, Axis1, 100.0, 100.0, 0 );
//Inst_MC_Halt is an instance of MC_halt function block
HaltComplete := Inst_MC_Halt.Done;           //store Done output into user
defined variable
```

See Also

- Call a Function Block
- Differences Between Functions and Function Blocks

3.2.11.3 MC_MoveAbsolute



Function Block - Performs a single-axis move to a specified endpoint position.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to queue the move.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
Position	LREAL	See Description.	User units	No default	Endpoint position. • If Rollover Position is nonzero, this value must be in the range $0 \leq \text{Position} < \text{Rollover Position}$. • When not in Rollover mode, the input accepts a 64-bit floating point value. • When converted to feedback units, the range is $[-2^{51}, 2^{51}-1]$. • The axis parameter MC_AXIS_PARAM_APPLY_SUPERIMPOSED_DISTANCE (1033) affects how superimposed moves affect MC_MoveAbsolute. • See "Effects of Superimposed Moves on MC_MoveAbsolute" (→ p. 666).
Velocity	LREAL	No range	User units / sec	No default	Velocity setpoint.
Acceleration	LREAL	No range	User units / sec ²	No default	• Trapezoidal: Acceleration rate. • S-curve: Maximum acceleration. • If Acceleration is not valid, the ErrorID is set to 21.
Deceleration	LREAL	No range	User units / sec ²	No default	• Trapezoidal: Deceleration rate. • S-curve: Maximum deceleration.
Jerk	LREAL	No range	User units / sec ³	No default	• Trapezoidal: 0 (zero). • S-curve: Constant jerk. • If Jerk is not valid, the ErrorID is set to 21.

Input	Data Type	Range	Unit	Default	Description												
Direction	SINT	0 to 4	N/A	No default	- When Rollover Position is zero, a value of 0 must be specified. - When Rollover Position is nonzero, a value of 1, 2, 3, or 4 must be specified.												
					<table><tr><th>Value</th><th>Description</th></tr><tr><td>0</td><td>No direction specification.</td></tr><tr><td>1</td><td>Positive direction. The axis travels in the positive direction to the endpoint.</td></tr><tr><td>2</td><td>Shortest distance. The axis travels in the direction that provides the shortest distance to the endpoint.</td></tr><tr><td>3</td><td>Negative direction. The axis travels in the negative direction to the endpoint.</td></tr><tr><td>4</td><td>Last direction. The axis travels to the endpoint in the same direction as its previous move.</td></tr></table>	Value	Description	0	No direction specification.	1	Positive direction. The axis travels in the positive direction to the endpoint.	2	Shortest distance. The axis travels in the direction that provides the shortest distance to the endpoint.	3	Negative direction. The axis travels in the negative direction to the endpoint.	4	Last direction. The axis travels to the endpoint in the same direction as its previous move.
					Value	Description											
					0	No direction specification.											
					1	Positive direction. The axis travels in the positive direction to the endpoint.											
					2	Shortest distance. The axis travels in the direction that provides the shortest distance to the endpoint.											
3	Negative direction. The axis travels in the negative direction to the endpoint.																
4	Last direction. The axis travels to the endpoint in the same direction as its previous move.																
If the Position input is the same as the axis's current position, then:																	
- If Direction = 2 (shortest distance): - The axis does not move. - The Done output goes high indicating that the move has been completed. - If Direction = 1, 3, or 4: - The axis travels in the specified direction, through one rollover cycle. - The axis arrives back at the same position.																	
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. - BM_ABORTING - BM_BUFFERED - BM_BLENDING_PREVIOUS - BM_BLENDING_NEXT - BM_BLENDING_LOW - BM_BLENDING_HIGH - See Buffer Modes.												

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input goes high until the time the move is ended.
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	Indicates either: - An invalid input was specified. - The move was terminated due to an error.
ErrorID	INT	No range	N/A	Indicates the error if Error output is TRUE.

Remarks

Effects of Superimposed Moves on MC_MoveAbsolute

The axis parameter MC_AXIS_PARAM_APPLY_SUPERIMPOSED_DISTANCE (1033) affects how superimposed moves affect MC_MoveAbsolute.

- By default, MC_MoveAbsolute does not take into account any added superimposed distance due to MC_MoveSuperimp.
 - When a MC_MoveAbsolute move completes, the final command position of the axis includes superimposed distances may not be equal to the Position input.
- To have MC_MoveAbsolute ignore superimposed distances and move the Position input, use "MC_WriteBoolPar" (→ p. 654) to set the value of the MC_AXIS_PARAM_APPLY_SUPERIMPOSED_DISTANCE axis parameter to TRUE.
- When MC_MoveAbsolute begins execution, the distance to move is calculated at that point in time.
 - If a superimposed move is started while MC_MoveAbsolute is in progress, the final position of the axis when MC_MoveAbsolute completes is not equal to the Position input, regardless of the value of the MC_AXIS_PARAM_APPLY_SUPERIMPOSED_DISTANCE parameter.

Time Diagram

These images show two examples of the combination of two absolute move Function Blocks (FBs):

- The left part of the diagram illustrates when the second FB is called **after** the first one.
 - If the first reaches the commanded position of 6000, and the velocity is 0 (zero), the output **Done** causes the second FB to move to the position 10000.
- The right part of the diagram illustrates when the second move FB starts the execution **while** the first FB is still executing.
 - The first motion is interrupted and aborted by the Test signal during the constant velocity of the first FB.
 - The second FB moves directly to the position 10000 although the position of 6000 is not yet reached.

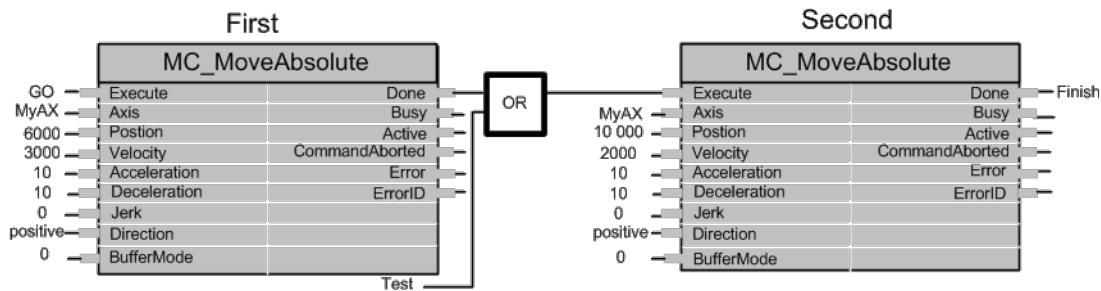


Figure 4-75: Time Diagrams: First and Second FBs

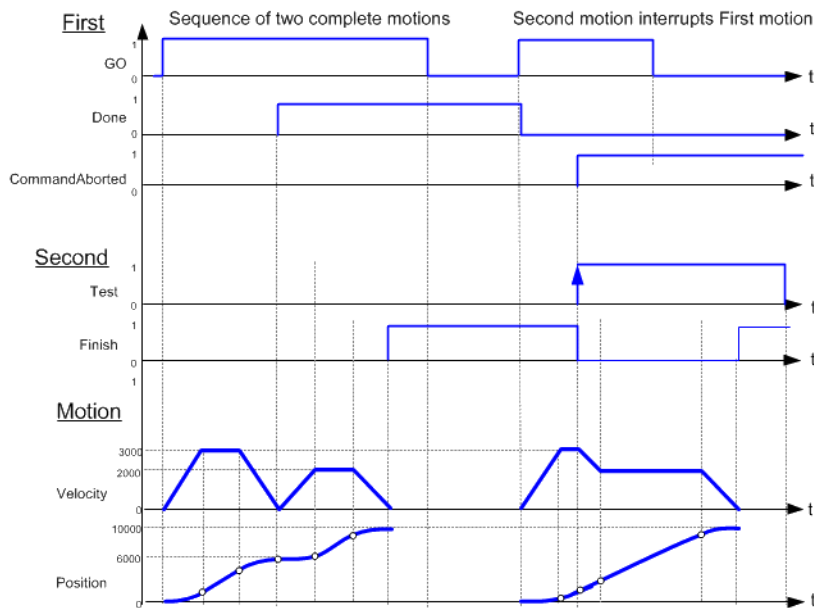


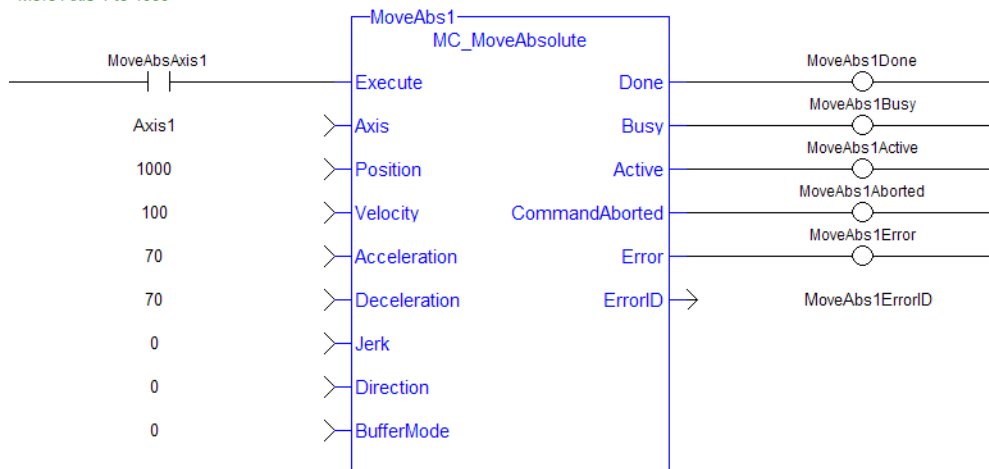
Figure 4-76: Time Diagram

FBD Language Example

Not available.

FFLD Language Example

Move Axis 1 to 1000



IL Language Example

Not available.

ST Language Example

```
(* MC_MoveAbsolute ST example *)  
Inst_MC_MoveAbsolute( MovAbsReq, Axis1, 1234.567, 100.0, 100.0, 100.0, 0, 0,  
0 ); //instance of MC_MoveAbsolute  
MovAbsDone := Inst_MC_MoveAbsolute.Done; //store done output into user  
defined variable  
MovAbsBusy := Inst_MC_MoveAbsolute.Busy;  
MovAbsActive := Inst_MC_MoveAbsolute.Active;  
MovAbsAborted := Inst_MC_MoveAbsolute.CommandAborted;  
MovAbsError := Inst_MC_MoveAbsolute.Error;  
MovAbsErrID := Inst_MC_MoveAbsolute.ErrorID;
```

See Also

- Call a Function Block
- Differences between Functions and Function Blocks

3.2.11.4 MC_MoveAdditive



 **Function Block** - Performs a single-axis move for a specified distance from the endpoint of the previous move.

- It is typically used with Abort specified at the BufferMode input.
 - If BufferMode is not Abort, this move is identical to an "MC_MoveRelative" (→ p. 677).

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to queue the move.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
Distance	LREAL	No range	User units	No default	Distance to add to the endpoint of the previous move.
Velocity	LREAL	No range	User units / sec	No default	Velocity setpoint.
Acceleration	LREAL	No range	User units / sec ²	No default	- Trapezoidal: Acceleration rate. - S-curve: Maximum acceleration.
Deceleration	LREAL	No range	User units / sec ²	No default	- Trapezoidal: Deceleration rate. - S-curve: Maximum deceleration.
Jerk	LREAL	No range	User units / sec ³	No default	- Trapezoidal: 0 (zero). - S-curve: Constant jerk.
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. - BM_ABORTING - BM_BUFFERED - BM_BLENDED_PREVIOUS - BM_BLENDED_NEXT - BM_BLENDED_LOW - BM_BLENDED_HIGH - See Buffer Modes.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.

Output	Data Type	Range	Unit	Description
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input goes high until the time the move is ended.
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	Indicates either: • An invalid input was specified. • The move was terminated due to an error.
ErrorID	INT	No range	N/A	• Indicates the error if Error output is TRUE.

Time Diagram

These images show two examples of the combination of two Function Blocks (FBs) while the axis is in Discrete Motion state:

- The left part of the diagram illustrates when the second FB is called **after** the first one.
 - If the first reaches the commanded distance 6000, and the velocity is 0 (zero), the output **Done** causes the second FB to move to the distance 10000.
- The right part of the diagram illustrates when the second move FB starts the execution **while** the first FB is still executing.
 - The first motion is interrupted and aborted by the Test signal during the constant velocity of the first FB.
 - The second FB:
 - Adds on the previous commanded position of 6000 the distance 4000.
 - Moves the axis to the resulting position of 10000.

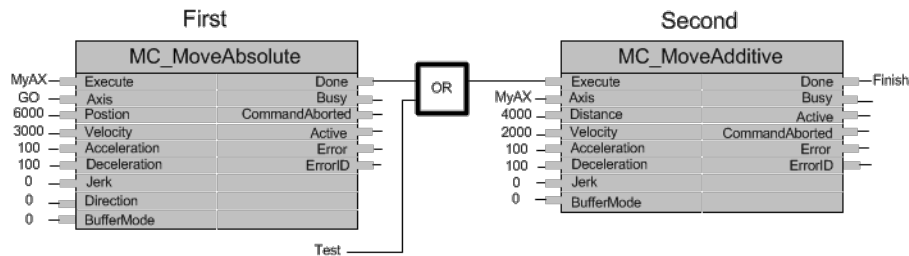


Figure 4-77: Time Diagrams: First and Second FBs

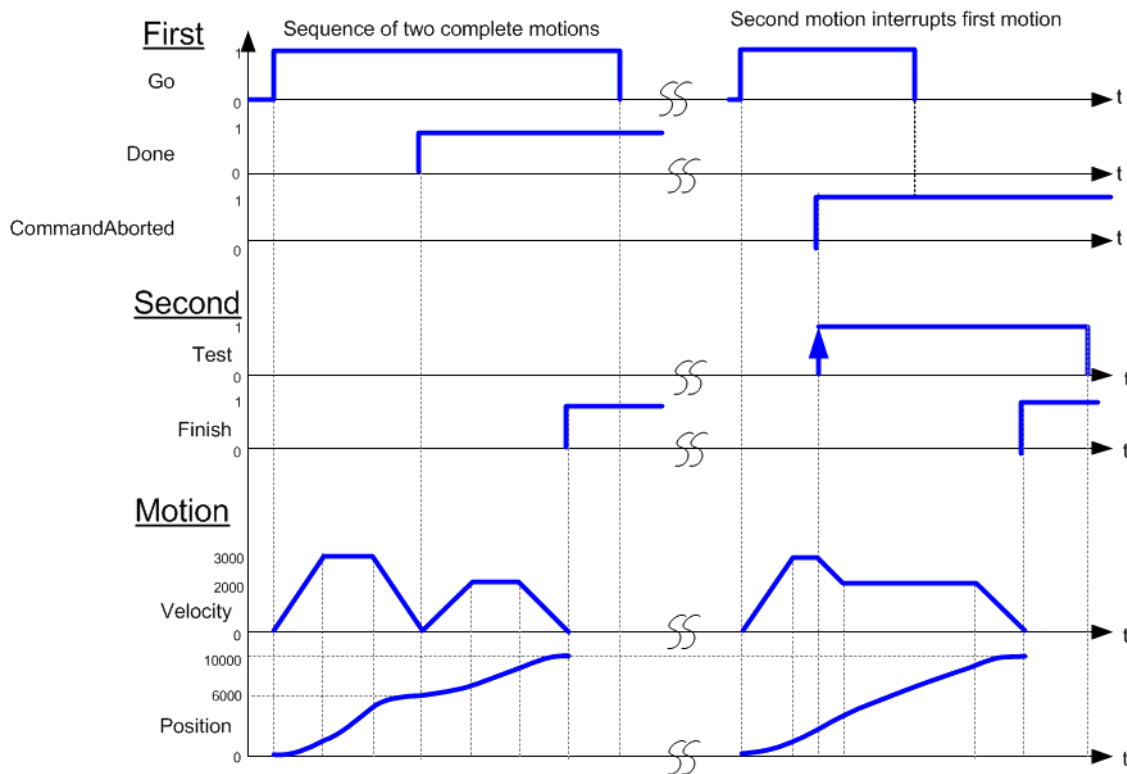


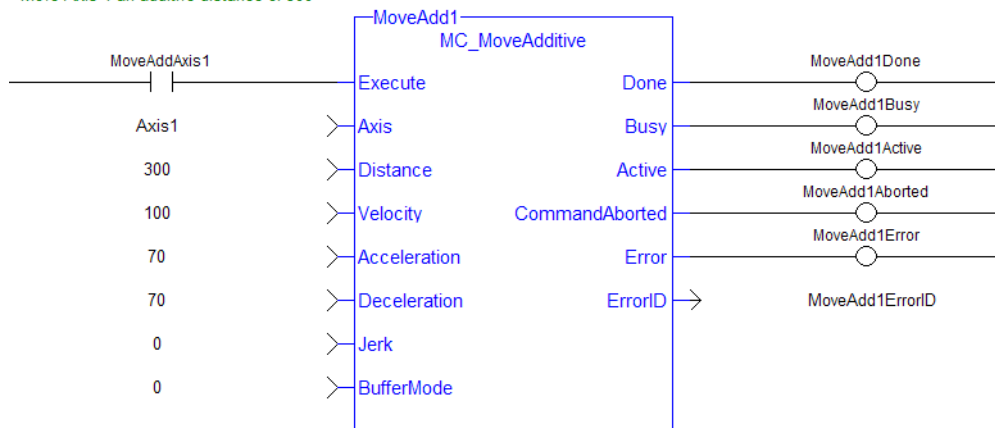
Figure 4-78: Time Diagram

FBD Language Example

Not available.

FFLD Language Example

Move Axis 1 an additive distance of 300



IL Language Example

Not available.

ST Language Example

```
(* MC_MoveAdditive ST example *)  
  
Inst_MC_MoveAdditive( MovAddReq, Axis1, 123.456, 100.0, 100.0, 100.0, 0, 0 );  
//Inst_MC_MoveAdditive is an instance of MC_MoveAdditive function block  
MovAddDone := Inst_MC_MoveAdditive.Done;  
//store Done output into user defined variable
```

See Also

- Call a Function Block
- Differences between Functions and Function Blocks

3.2.11.5 MC_MoveContVel



Function Block - Performs a single-axis non-ending move at a specified velocity with the option of continually updating the ongoing motion with the current input parameters.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

- After **MC_MoveContVel** execution begins (**Execute** input - low to high), follow up changes to input parameters immediately affect the ongoing motion, without requiring an additional low to high transition on the **Execute** input.
- This type of move can be terminated with the "MC_Halt" (→ p. 661) function block or by aborting it with another move.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to queue the move.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
Velocity	LREAL	All finite values.	User units / sec	No default	The target axis velocity. • Negative values of velocity move the axis in the negative direction. • Positive values move the axis in the positive direction. • A value of 0 (zero) is valid. • 0 (zero) indicates a deceleration to zero velocity.
Acceleration	LREAL	Positive values.	User units / sec ²	No default	• Trapezoidal: Acceleration rate. • S-curve: Maximum acceleration.
Deceleration	LREAL	Positive values.	User units / sec ²	No default	• Trapezoidal: Deceleration rate. • S-curve: Maximum deceleration.
Jerk	LREAL	No range	User units / sec ³	No default	• Trapezoidal: 0 (zero). • S-curve: Constant jerk.

Input	Data Type	Range	Unit	Default	Description
ContinuousUpdate	BOOL	FALSE, TRUE	N/A	No default	Determines if the inputs of the function block are re-evaluated every cycle or if they are only evaluated on the rising edge of Execute. - If TRUE, when the function block is triggered (on the rising edge of Execute), the function block uses the current updated values of the input variables and apply it to the ongoing movement of the axis. - This continues as long as ContinuousUpdate stays TRUE. - The impact of ContinuousUpdate ends as soon as the function block is no longer busy (Busy output is FALSE) or ContinuousUpdate is set to FALSE.
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. - BM_ABORTING - BM_BUFFERED - BM_BLENDING_PREVIOUS - BM_BLENDING_NEXT - BM_BLENDING_LOW - BM_BLENDING_HIGH - See Buffer Modes.

Outputs

Output	Data Type	Range	Unit	Description
InVelocity	BOOL	FALSE, TRUE	N/A	Indicates the command velocity has reached the programmed velocity.
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input goes high until the time the move is ended.
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	Indicates either: - An invalid input was specified. - The move was terminated due to an error.
ErrorID	INT	-32768 to +32767	N/A	Indicates the error if Error output is TRUE.

Time Diagram

This example shows the behavior of the combination of a "MC_Stop" (→ p. 627) function block with a MC_MoveContVel function block.

- A rotating axis is ramped down with FB2 **MC_Stop**.
- The axis rejects motion commands as long as **MC_Stop** parameter **Execute** = TRUE.
- FB1 MC_MoveContVel reports an error indicating the busy **MC_Stop** command.

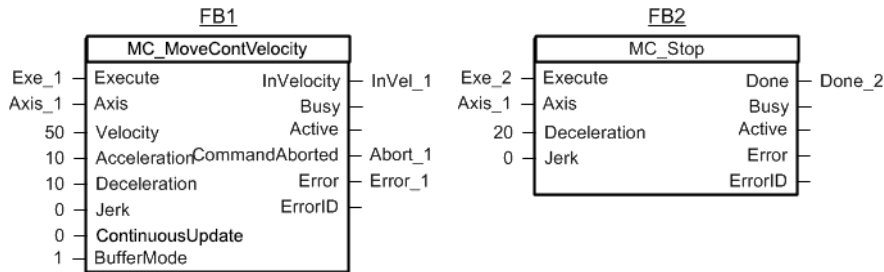


Figure 4-79: Time Diagrams: First and Second FBs

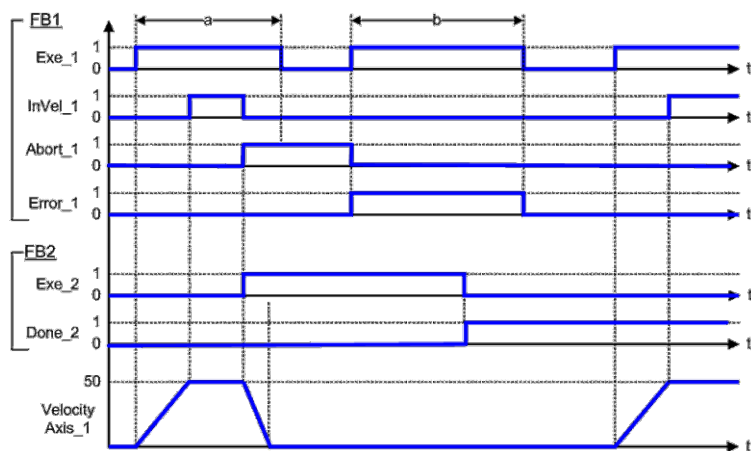


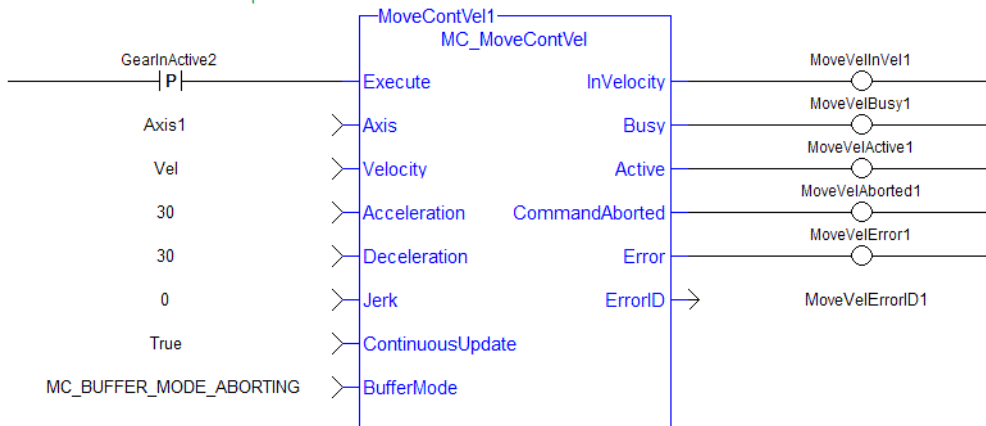
Figure 4-80: Time Diagram

FBD Language Example

Not available.

FFLD Language Example

Run Axis 1 when the Run/Stop switch is set to Run



IL Language Example

Not available.

ST Language Example

```
(* MC_MoveContVel ST example *)  
Inst_MC_MoveContVel( MovVelReq , Axis1, Vel, 100.0, 100.0, 0, True, MC_  
BUFFER_MODE_ABORTING );
```

See Also

- Call a Function Block
- Differences between Functions and Function Blocks
- "MC_MoveVelocity" (→ p. 685)

3.2.11.6 MC_MoveRelative



 **Function Block** - Performs a single-axis move of a specified distance relative to the actual position at the time of the start of execution.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to queue the move.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
Distance	LREAL	No range	User units	No default	Distance.
Velocity	LREAL	No range	User units / sec	No default	Velocity setpoint.
Acceleration	LREAL	No range	User units / sec ²	No default	• Trapezoidal: Acceleration rate. • S-curve: Maximum acceleration.
Deceleration	LREAL	No range	User units / sec ²	No default	• Trapezoidal: Deceleration rate. • S-curve: Maximum deceleration.
Jerk	LREAL	No range	User units / sec ³	No default	• Trapezoidal: 0 (zero). • S-curve: Constant jerk.
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. • BM_ABORTING • BM_BUFFERED • BM_BLENDING_PREVIOUS • BM_BLENDING_NEXT • BM_BLENDING_LOW • BM_BLENDING_HIGH • See Buffer Modes.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input goes high until the time the move is ended.

Output	Data Type	Range	Unit	Description
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	Indicates either: • An invalid input was specified. • The move was terminated due to an error.
ErrorID	INT	No range	N/A	• Indicates the error if Error output is TRUE.

Time Diagram

These images show two examples of the combination of two relative move Function Blocks (FBs).

- The left part of the diagram illustrates when the second FB is called **after** the first one.
 - If the first reaches the commanded position of 6000, and the velocity is 0 (zero), the output **Done** causes the second FB to move to the distance 10000.
- The right part of the diagram illustrates when the second move FB starts the execution **while** the first FB is still executing.
 - The first motion is interrupted and aborted by the Test signal during the constant velocity of the first FB.
 - The second FB **adds on the actual position** of 3250 the distance 4000 and moves the axis to the resulting position of 7250.

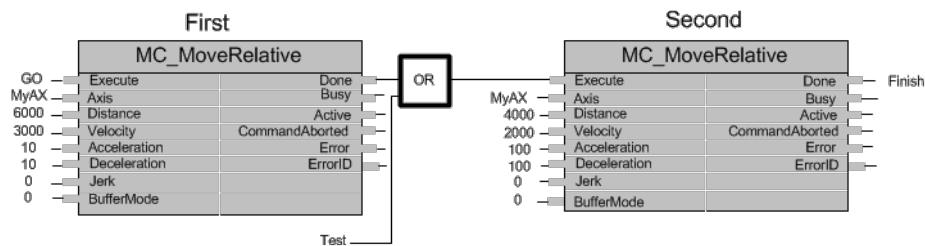


Figure 4-81: Time Diagrams: First and Second FBs

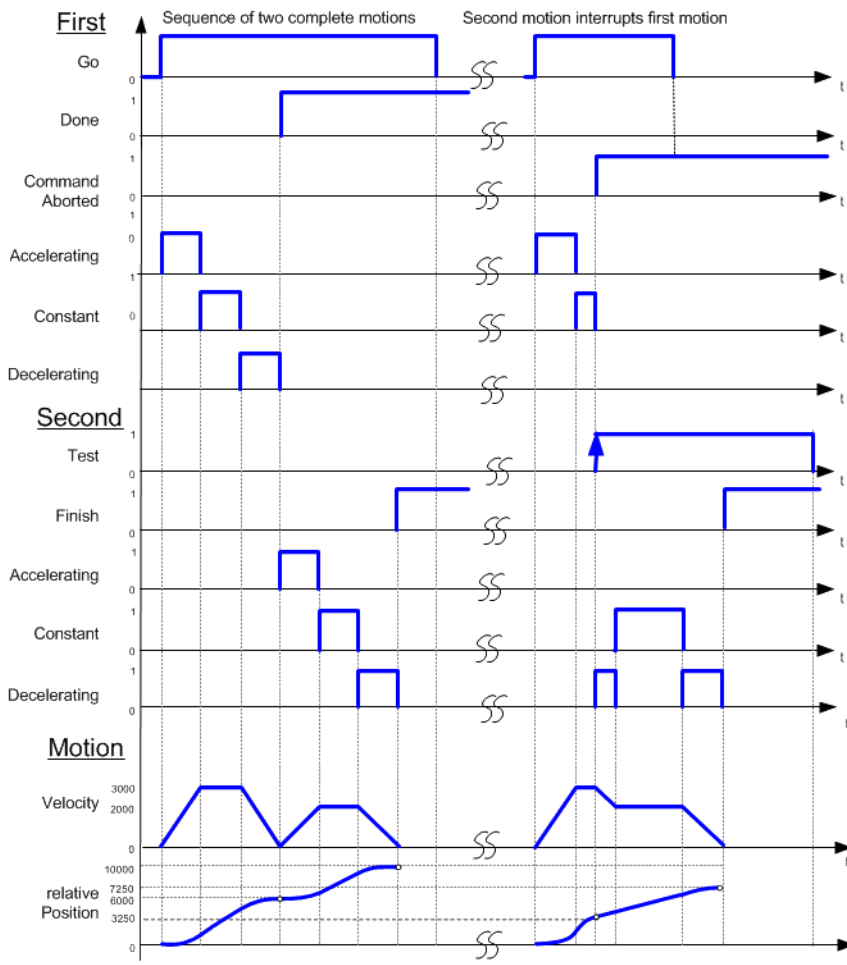


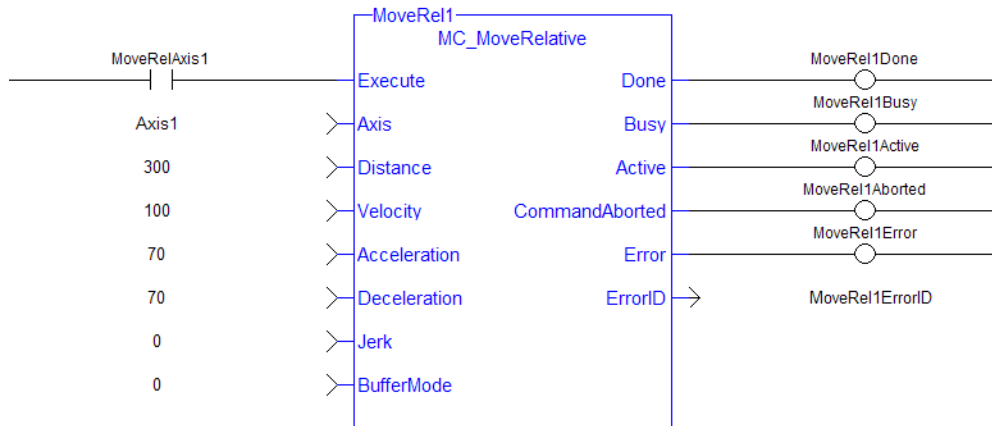
Figure 4-82: Time Diagram

FBD Language Example

Not available.

FFLD Language Example

Move Axis 1 a relative distance of 300



IL Language Example

Not available.

ST Language Example

```
(* MC_MoveRelative ST example *)  
Inst_MC_MoveRelative( MovRelReq, Axis1, 10.0, 200.0,150.0, 150.0, 0,0 );  
MovRelDone := Inst_MC_MoveRelative.Done; //store Done output into user  
defined variable
```

See Also

- Call a Function Block
- Differences between Functions and Function Blocks

3.2.11.7 MC_MoveSuperimp



 **Function Block** - Performs a single-axis move superimposed upon the currently executing move.

- Commands a controlled motion of a specified relative distance additional to an existing motion.
- The existing Motion is not interrupted, but is superimposed by the additional motion.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

3.2.11.7.1.1 Usage

This function block:

- Provides a way to smoothly apply a shift in axis position while it is executing a move.
- Is commonly used with "MC_TouchProbe" (→ p. 633) for performing position corrections on the slave axis in a **Mark-to-Machine** registration application.

TIP

MC_MoveSuperimp performs a similar function to the SlaveOffset input in the "MC_CamIn" (→ p. 693) function block.

It has the additional features of setting the velocity, acceleration, deceleration, and jerk motion parameters.

- A superimposed move is executed similar to a "MC_MoveRelative" (→ p. 677) move using the specified **Distance**, **Velocity**, **Acceleration**, **Deceleration**, and **Jerk** values.
 - The interpolated command generated by a superimposed move is added to the command of the currently executing move.
 - Subsequent calls to MC_MoveSuperimp can abort or blend to an executing MC_MoveSuperimp move.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to queue the move.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
Distance	LREAL	No range	User units	No default	Distance.
VelocityDiff	LREAL	No range	User units / sec	No default	Velocity rate.
Acceleration	LREAL	No range	User units / sec ²	No default	• Trapezoidal: Acceleration rate. • S-curve: Maximum acceleration.
Deceleration	LREAL	No range	User units / sec ²	No default	• Trapezoidal: Deceleration rate. • S-curve: Maximum deceleration.
Jerk	LREAL	No range	User units / sec ³	No default	• Trapezoidal: 0 (zero). • S-curve: Constant jerk.

Input	Data Type	Range	Unit	Default	Description
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. • BM_ABORTING • BM_BUFFERED • BM_BLENDING_PREVIOUS • BM_BLENDING_NEXT • BM_BLENDING_LOW • BM_BLENDING_HIGH • See Buffer Modes.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the move completed successfully. The Command Position has reached the endpoint.
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input goes high until the time the move is ended.
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	Indicates either: • An invalid input was specified. • The move was terminated due to an error.
ErrorID	INT	No range	N/A	• Indicates the error if Error output is TRUE.

Time Diagram

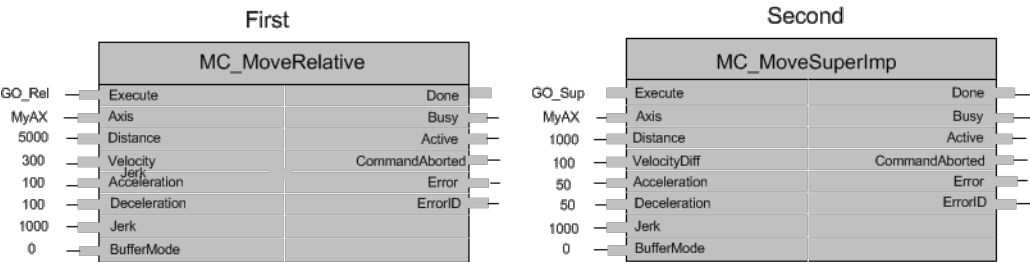


Figure 4-83: Time Diagrams: First and Second FBs

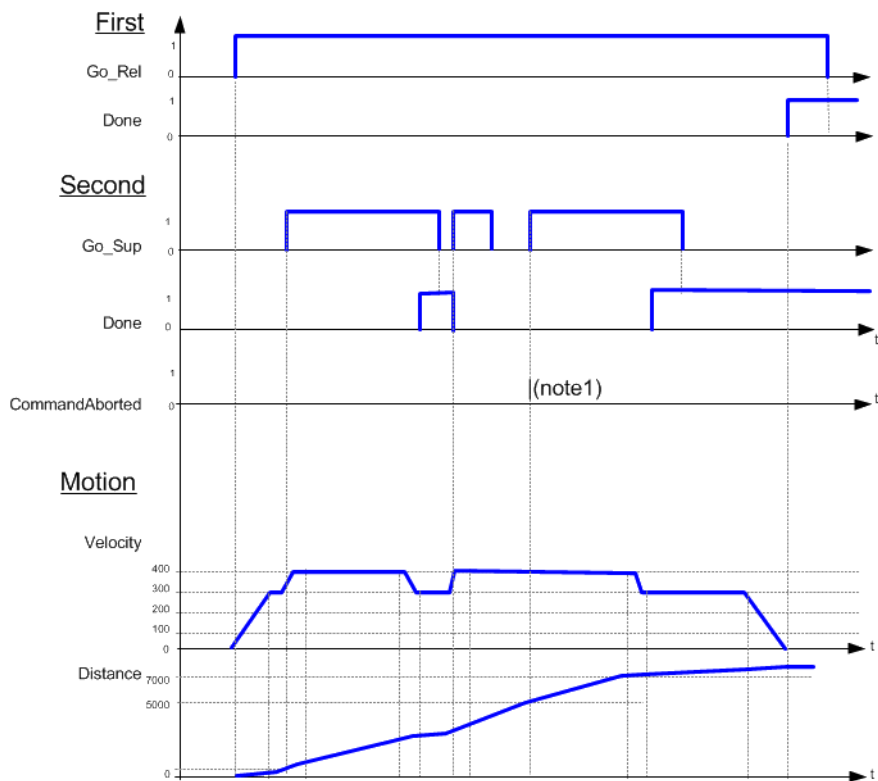


Figure 4-84: Time Diagram

NOTE

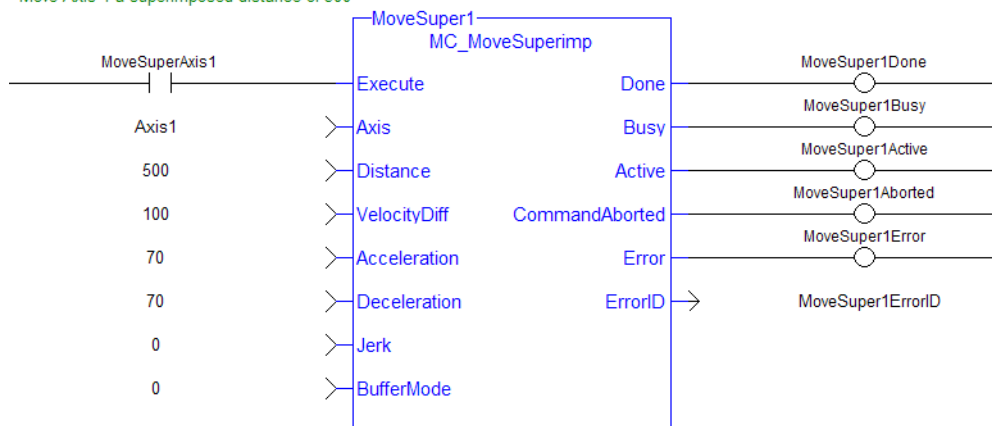
- The CommandAborted is not visible here because the new command works on the same instance.
- The end position is between 7000 and 8000, depending on the timing of the aborting of the second command set for the MC_MoveSuperimposed.

FBD Language Example

Not available.

FPLD Language Example

Move Axis 1 a superimposed distance of 500



IL Language Example

Not available.

ST Language Example

```
(* MC_MoveSuperimp ST example *)  
Inst_MC_MoveSuperimp( MovSupReq, Axis1, 123.555, 10.0, 100.0, 100.0, 0, 0 );  
MovSupDone := Inst_MC_MoveSuperimp.Done; //store Done output into user  
defined variable
```

See Also

- Call a Function Block
- Differences between Functions and Function Blocks

3.2.11.8 MC_MoveVelocity

PLCopen 



Function Block - Performs a single-axis, non-ending move at a specified velocity.

- This type of move can be terminated with the MC_Halt function block or by aborting it with another move.

TIP

- Consider using the "MC_MoveContVel" (→ p. 673) function block.
- It is more flexible and allows for the continuous update of motion parameters.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to queue the move.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
Speed	LREAL	Positive values.	User units / sec	No default	The target axis speed. Direction is specified by the Direction input parameter.
Acceleration	LREAL	Positive values.	User units / sec ²	No default	• Trapezoidal: Acceleration rate. • S-curve: Maximum acceleration.
Deceleration	LREAL	Positive values.	User units / sec ²	No default	• Trapezoidal: Deceleration rate. • S-curve: Maximum deceleration.
Jerk	LREAL	No range	User units / sec ³	No default	• Trapezoidal: 0 (zero). • S-curve: Constant jerk.
Direction	SINT	0 to 1	N/A	No default	• A 0 or False value specifies that the axis should move in the positive direction. • A 1 or True value specifies that the axis should move in the negative direction.
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. • BM_ABORTING • BM_BUFFERED • BM_BLENDING_PREVIOUS • BM_BLENDING_NEXT • BM_BLENDING_LOW • BM_BLENDING_HIGH • See Buffer Modes.

Outputs

Output	Data Type	Range	Unit	Description
InVelocity	BOOL	FALSE, TRUE	N/A	Indicates the command velocity has reached the programmed velocity.
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input is one-shot to the time the move is ended.
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	Indicates either: - An invalid input was specified. - The move was terminated due to an error.
ErrorID	INT	No range	N/A	Indicates the error if Error output is TRUE.

Time Diagram

These images show two examples of the combination behavior of an [MC_Stop](#) FB with a MC_MoveVelocity FB.

- A rotating axis is ramped down with FB2 MC_Stop.
- The axis rejects motion commands as long as MC_Stop parameter “Execute” = TRUE.
- FB1 MC_MoveVelocity reports an error indicating the busy MC_Stop command.

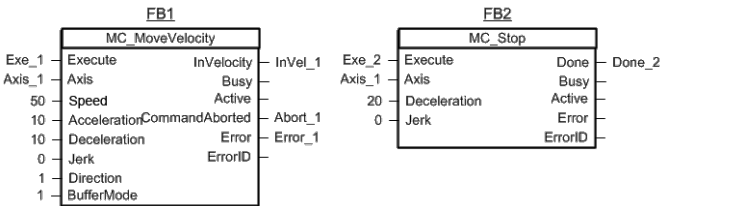


Figure 4-85: Time Diagrams: First and Second FBs

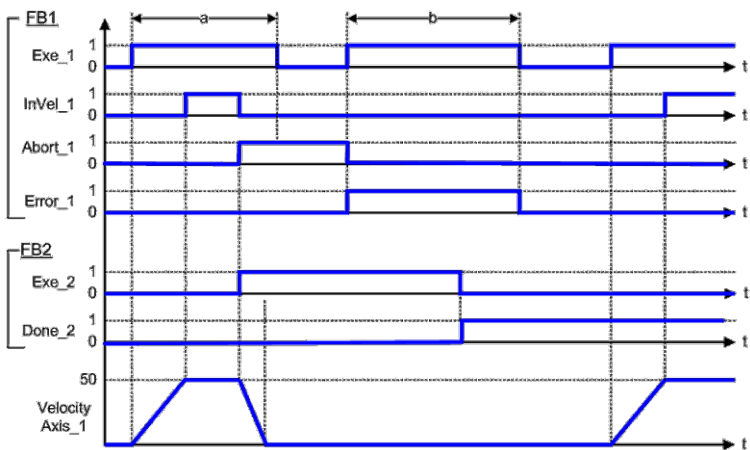
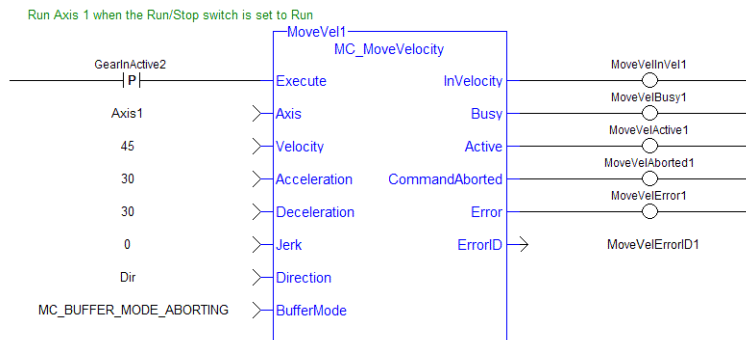


Figure 4-86: Time Diagram

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_MoveVelocity ST example *)

Inst_MC_MoveVelocity( MovVelReq , Axis1, 200.0, 100.0, 100.0, 0, True, MC_
BUFFER_MODE_ABORTING );
```

See Also

- Call a Function Block
- Differences between Functions and Function Blocks
- "MC_MoveContVel" (→ p. 673)

3.2.11.9 MC_RemSuperAxis



 **Function Block** - Remove an axis from the axis's list of assigned, superimposed axes.

This feature allows the application program to superimpose the moves of multiple axes (Superimposed Axes) on top of the move of another axis (Receiving Axis).

- This is performed internally by adding the command deltas of the Superimposed Axes to the command delta of the Receiving Axis.
- A maximum of four different Superimposed Axes can be superimposed upon a Receiving Axis.
- If the value at **SuperimposedAxis** is 0 (zero), all assigned superimposed axes are removed from Axis's list.
- The **OK** output goes high to indicate the function executed successfully.
- If the **OK** output **does not** go high, one of these errors is detected:
 - The axis is not a valid axis.
 - The axis is not a servo or virtual axis.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

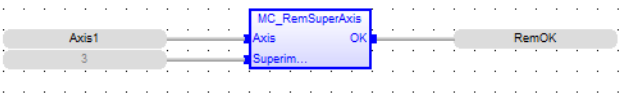
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	N/A	Enables execution.
Axis	AXIS_REF	AXIS_NUM1 to 256	N/A	N/A	Axis whose list of assigned superimposed axes is updated.
SuperimposedAxis	UINT	N/A	N/A	N/A	Axis number of the superimposed axis that is removed from Axis's list of assigned superimposed axes. A value of 0 (zero) removes all superimposed axes from Axis's list.

Outputs

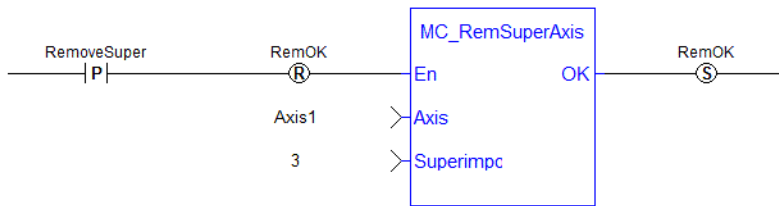
Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Execution successful.

FBD Language Example



FFLD Language Example

Network #19



IL Language Example

Not available.

ST Language Example

```
RemOK := MC_RemSuperAxis(Axis1, 3);
```

See Also

"MC_AddSuperAxis" (→ p. 659)

3.2.11.10 MC_SetOverride



Function Block - Writes velocity and acceleration override factors.

-
- A change in the velocity override factor takes effect immediately on the active move.
- The velocity override factor is applied to the programmed velocity (of a "MC_MoveAbsolute" (→ p. 664), "MC_MoveAdditive" (→ p. 669), "MC_MoveRelative" (→ p. 677), "MC_MoveSuperimp" (→ p. 681), or "MC_MoveVelocity" (→ p. 685) function block) to determine the command velocity:

```
command velocity = programmed velocity * VelFactor
```

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	Request to write the override factors.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
VelFactor	REAL	0.0 to 2.0	N/A	No default	Velocity override factor.

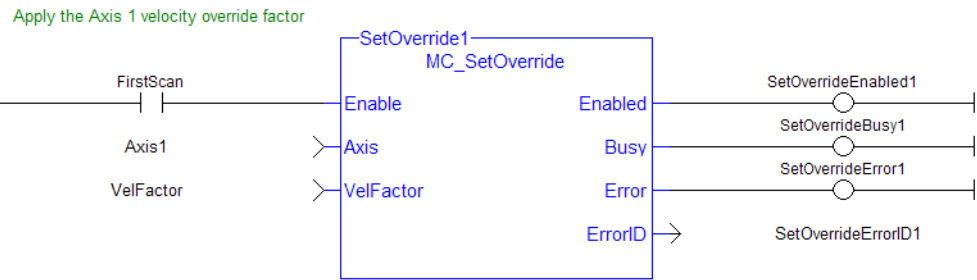
Outputs

Output	Data Type	Range	Unit	Description
Enabled	BOOL	FALSE, TRUE	N/A	Indicates the override values have been written.
Busy	BOOL	FALSE, TRUE	N/A	Indicates this function block is executing.
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified.
ErrorID	INT			• Indicates the error if Error output is TRUE.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_SetOverride ST example *)  
  
VelFactor := 1.25 ;    //set the velocity factor to 1.25 (125%)  
Inst_MC_SetOverride( TRUE , Axis1, VelFactor ); // Inst_MC_Setoverride is an  
instance of MC_SetOverride
```

3.2.12 Profile

These functions provide commands for slave axes, such as cams and gearing.

Name	Description
MC_CamIn	Performs a slave axis move which follows the master axis based on the Cam Table specified by CamTableID.
MC_CamOut	Disengages the slave axis from an "MC_CamIn" (→ p. 693) move or the master axis immediately in a cam block.
MC_CamResumePos	Returns the slave axis position for resuming an "MC_CamIn" (→ p. 693) move.
MC_CamStartPos	Returns the slave axis position for starting an "MC_CamIn" (→ p. 693) move.
MC_CamTblSelect	Defined to read and initialize the specified profile. Selects the Cam tables by setting the pointers to the relevant tables.
MC_GearIn	Performs a slave axis move which follows the master axis using the ratio specified by RatioNumerator and RatioDenominator. Once engaged, the slave moves RatioNumerator units for every RatioDenominator units the master moves.
MC_GearInPos	Performs a slave axis move which synchronizes motion at the SlaveSyncPosition and MasterSyncPosition. Once synchronized, the slave moves RatioNumerator units for every RatioDenominator units the master moves.
MC_GearOut	Disengages the slave axis from a MC_GearIn or MC_GearInPos move or the master axis.
MC_Phasing	Performs a master position phase shift for the slave axis.
MC_SyncSlaves	Allows the application to specify what slave axes are to be synchronized and which master they follow.

3.2.12.1 MC_CamIn



 **Function Block** - Performs a slave axis move which follows the master axis based on the Cam Table specified by CamTableID.

- This function block is used to either:
 - Initiate a new MC_CamIn move.
 - Resume a previously programmed MC_CamIn move.
- See "MC_CamStartPos" (→ p. 705) or "MC_CamResumePos" (→ p. 703) for more information about positioning the slave axis prior to calling MC_CamIn.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to queue the move.
Master	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
Slave	AXIS_REF	1 to 256	N/A	No default	• AXIS_REF.AXIS_NUM is the slave axis number.
MasterOffset	LREAL	No range	User units	No default	• Profile shift along the master axis. • This input is not used if the StartMode input is set to 1 for Resume Mode.
SlaveOffset	LREAL	No range	User units	No default	• Profile shift along the slave axis. • This input is not used if the StartMode input is set to 1 for Resume Mode.
MasterScaling	LREAL	No range	User units	No default	Master axis scale factor. • This input is not used if the StartMode input is set to 1 for Resume Mode. • Scaling must be a positive value greater than 0 (zero).
SlaveScaling	LREAL	No range	User units	No default	Slave axis scale factor. • This input is not used if the StartMode input is set to 1 for Resume Mode. • Scaling must be a positive value greater than 0 (zero).

Input	Data Type	Range	Unit	Default	Description
StartMode	INT	0 to 1	N/A	No default	Starting mode of the cam profile. The function block uses values that were in effect during the most recently programmed MC_CamIn move for the slave axis. 0 = Start Mode - Input indicates if the axis should start a MC_CamIn move as an initial cam start. 1 = Resume Mode - Input indicates if the axis should resume the most recently programmed MC_CamIn move. - The inputs MasterOffset, SlaveOffset, MasterScaling, and SlaveScaling are not used.
CamTableID	INT	No range	N/A	No default	Profile ID number.
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. - BM_ABORTING - BM_BUFFERED - See Buffer Modes.

Outputs

Output	Data Type	Range	Unit	Description
InSync	BOOL	FALSE, TRUE	N/A	Indicates the slave axis is in sync with the profile.
Busy	BOOL	FALSE, TRUE	N/A	Indicates this function block is executing.
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	Indicates either: - An invalid input was specified. - The move was terminated due to an error.
ErrorID	INT	No range	N/A	Indicates the error if the Error output is high.

Output	Data Type	Range	Unit	Description
EndOfProfile	BOOL	FALSE, TRUE	N/A	Indicates the end of profile has been reached. - If the profile is periodic, this output is set to ON for one ladder scan. - If the profile is not periodic, the output remains ON while outside the range of the profile. - See "Periodic or Not Periodic" (→ p. 695) for more information.

Remarks

3.2.12.1.1.1 Abort Camming

NOTE

Ending camming is also possible with other single axis function blocks (e.g., "MC_MoveRelative" (→ p. 677) or "MC_MoveAbsolute" (→ p. 664)).

There are two options to stop camming after MC_CamIn has been called:

- "MC_CamOut" (→ p. 700) continues motion at the instantaneous final actual velocity of the slave axis when it is called.
 - Axis motion continues at that final actual velocity.
- "MC_Halt" (→ p. 661) (with buffer mode input = 0) decelerates axis motion to 0 (zero) speed and stop motion.
- The master / slave relationship between the two axes is ended when MC_CamOut or MC_Halt is called.

3.2.12.1.2.2 Usage

The slave axis immediately locks on to the Cam Table profile.

- The **Master Offset** is used to shift the profile along the master axis.
- The **Master Scaling** defines the range of the profile along the master axis.
- The **Slave Offset** is used to shift the profile along the Slave axis.
- The **Slave Scaling** defines the range of the profile along the slave axis.

3.2.12.1.3.3 Periodic or Not Periodic

- If the profile is periodic, when the end of profile is reached, the profile continues at the start of the profile.
 - The EndOfProfile output is ON for 1 ladder scan.
- If the profile is not periodic, when the end of profile is reached, the slave axis stops and remains at the end of the profile until the master axis returns to within the profile range as defined by MasterScaling.
 - The EndOfProfile output remains ON anytime the master axis is outside of the profile range.

3.2.12.1.4.4 Adjustments Computation is Done

When cam is first started, offsets are adjusted if necessary.

- If the master is not absolute, then master offset = master offset + starting position.
- If the slave is not absolute, then slave offset = slave offset + starting position.

At run-time:

- Master position for profile = master position - master offset.
 - Use master position for profile table to obtain slave profile position.
- Slave commanded position = slave profile position + slave offset.

3.2.12.1.5.5 Dynamically Change a Cam Profile

MC_CamIn can be used to dynamically change from one cam profile to another.
Care must be taken when doing this to avoid unexpected motion.

TIP

- Tips to dynamically change cam profiles:
- Verify the first cam's last position and the replacement cam's first position are the same.
 - Offsets as set by "MC_CamTblSelect" (→ p. 707) affect actual cam position.
 - Verify the first cam's last velocity and the replacement cam's first velocity do not cause any unexpected motion.
 - Jumps can be eliminated by defining the present cam as **Cyclic** and defining the replacement cam as an **Absolute Master** and **Slave**, as set by the "MC_CamTblSelect" (→ p. 707) inputs.
 - This eliminates any possible small error accumulating when the cam is switched.

3.2.12.1.6 Examples

These examples use the "MC_CamIn" (→ p. 693) image showing the cam profile **MyProfile**:

- "Example 1" (→ p. 696)
- "Example 2" (→ p. 697)
- "Example 3" (→ p. 697)

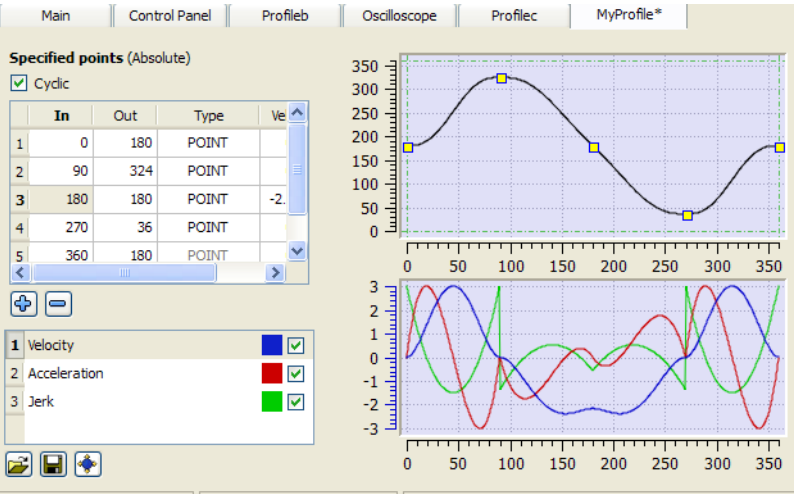
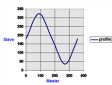


Figure 4-87: MC_CamIn Example

3.2.12.1.7.1 Example 1

Example 1 - MC_CamIn Cam Profile

Profile	MyProfile
Cycle	NO
MasterAbsolute	YES
SlaveAbsolute	YES
MasterOffset	0.0
SlaveOffset	0.0
MasterScaling	360.0
SlaveScaling	360.0
Initial Master position	0.0
Initial Slave position	180.0

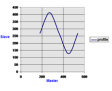


After `MC_CamTblSelect` and `MC_CamIn` are programmed with the parameters, the slave axis is locked on to the profile.

- Since both axes have zero offsets, the profile is not shifted in either axis.
- The initial condition of the master axis at position 0 (zero), yields a slave command position of 180.0.
 - As the master axis moves positive, the slave position follows the profile.
 - When the master position is at 90.0, the slave is commanded to 324.0.
 - See curve above where in = 90, out = 324.
 - The slave follows the profile as the master axis moves until the master axis reaches a position of 360.0.
 - At this time, the slave is commanded to 180.0.
- If the master were to continue to move past 360.0, the slave commanded position would remain at 180.0 since the Cyclic input is false.
- If the master moves negative, and its position returns to less than 360.0, the slave follows the profile again.

3.2.12.1.8.2 Example 2

Example 2 - MC_CamIn Cam Profile

Profile	MyProfile	
Cycle	YES	
MasterAbsolute	NO	
SlaveAbsolute	NO	
MasterOffset	0.0	
SlaveOffset	0.0	
MasterScaling	360.0	
SlaveScaling	360.0	
Initial Master position	180.0	
Initial Slave position	90.0	

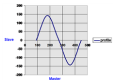
After `MC_CamTblSelect` and `MC_CamIn` are programmed with the parameters, the slave axis is locked on to the profile.

- Since both axes have zero offsets, the profile is not shifted in either axis.
- Neither the **MasterAbsolute** nor **SlaveAbsolute** input is on, so the profile is relative to the axes initial positions.
- Specifically, the initial condition of the master axis at position 180 would represent a master profile position of 0 (180-180).
 - This yields a slave command position of 270 (180 + 90).
- As the master axis moves positive, the slave position follows the profile.
 - When the master position is at 270, the slave is commanded to 414.0 (324 + 90).
 - The slave follows the profile as the master axis moves until the master axis reaches a position of 540.
 - At this time the slave is commanded to 270.0 (180 + 90).
- If the master continues to move past 540.0, the slave commanded position follows the profile from the beginning since the Cyclic input is TRUE.
- When the master reaches a position of 630, the slave is commanded to a position of 414.0 (324 + 90).

3.2.12.1.9.3 Example 3

Example 3 - MC_CamIn Cam Profile

Profile	MyProfile
Cycle	NO
MasterAbsolute	YES
SlaveAbsolute	YES
MasterOffset	90.0
SlaveOffset	-180
MasterScaling	360.0
SlaveScaling	360.0
Initial Master position	180.0
Initial Slave position	144.0



After MC_CamTblSelect and MC_CamIn are programmed with the parameters, the slave axis is locked on to the profile.

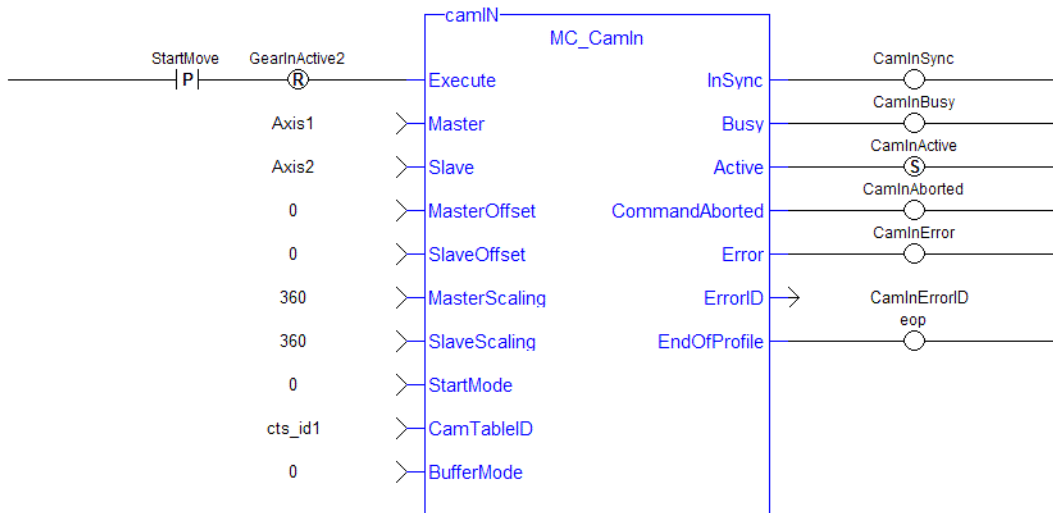
- Since the both axes **have offsets**, the profile **is shifted** along both axes.
- Specifically, the master axis is shifted 90, and the slave axis is shifted -180.
- Initially the master axis position of 180 yields a master position for the profile calculation of 90 (master position 180 - Master offset 90).
 - This yields a slave command position of 144 (slave profile command 324 + slave offset (-180)).
- As the master axis moves positive, the slave position follows the profile.
 - When the master axis position is at 270, the master position for profile calculation is 180 (270 - 90).
 - This yields a slave command position of 0 (zero) (180 + (-180)).
- The slave follows the profile as the master axis moves until the master axis reaches a position of 450.
- The master axis position of 450 yields a master position for profile calculation of 360 (450 - 90).
 - The slave command position is 0 (180 + (-180)).
- When the master reaches a position of 450, the slave commanded position remains at 0 (zero) since the Cyclic input is FALSE.

FBD Language Example

Not available.

FFLD Language Example

Network #3



IL Language Example

Not available.

ST Language Example

```
(* MC_CamIn ST example *) //Inst_MC_CamIn is an instance of MC_CamIn
Inst_MC_CamIn( CamStartBool, Axis1, Axis2, 0.0, 0.0, 360.0, 360.0, 0,
CamTableID, 0 );
```

See Also

- Call a Function Block
- Differences Between Functions and Function Blocks
- "MC_CamOut" (→ p. 700)
- "MC_CamResumePos" (→ p. 703)
- "MC_CamStartPos" (→ p. 705)
- "MC_CamTblSelect" (→ p. 707)

3.2.12.2 MC_CamOut



 **Function Block** - Disengages the slave axis from an "MC_CamIn" (→ p. 693) move or the master axis immediately in a cam block.

- If this function block is called and the active move is not a MC_CamIn move, this function block returns an error and the active move is not aborted.
- This function block:
 - Aborts the active MC_CamIn move.
 - Disengages the axis from its master.
 - Disengages the slave axis from a MC_CamIn move.
 - Commands the axis to continue at its current velocity.
 - The axis continues to run at this velocity until this MC_CamOut move is aborted.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

TIP

The current velocity is calculated by taking the average of the actual velocity during the previous 16 cycles.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to queue the move.
Slave	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
Acceleration	LREAL	No range	User units / sec ²	No default	• Trapezoidal: Acceleration rate. • S-curve: Maximum acceleration.
Deceleration	LREAL	No range	User units / sec ²	No default	• S-curve: Unused. • Trapezoidal: Deceleration rate.
Jerk	LREAL	No range	User units / sec ³	No default	• S-curve: Constant jerk. • Trapezoidal: 0 (zero).

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the axis is disengaged from its master.
Busy	BOOL	FALSE, TRUE	N/A	Indicates this function block is executing.
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.

Output	Data Type	Range	Unit	Description
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	Indicates either: - An invalid input was specified. - No MC_CamIn move was active.
ErrorID	INT	No range	N/A	When the Error output is: - High, this output indicates the type of error. - Low, this output is undefined.

Remarks

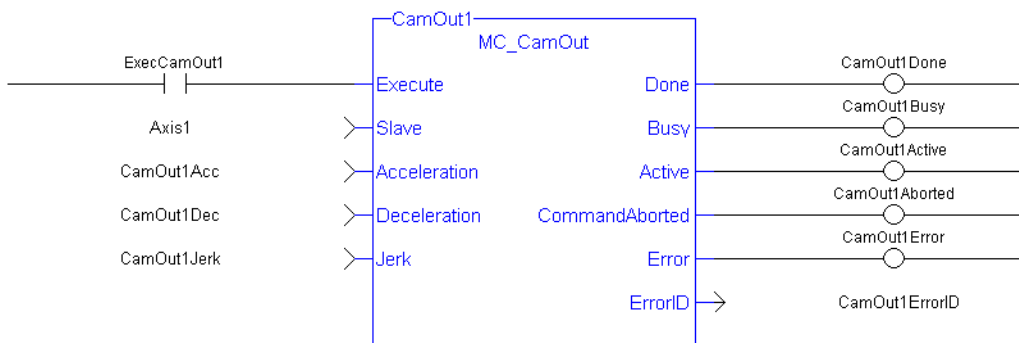
3.2.12.2.1.1 Cancel the Cam Motion

- An alternative method to cancel the cam motion is a single axis move ("MC_MoveAbsolute" (→ p. 664), "MC_MoveRelative" (→ p. 677), "MC_MoveAdditive" (→ p. 669), "MC_MoveVelocity" (→ p. 685), and "MC_Halt" (→ p. 661)) with the BufferMode input set to 0 (zero).
 - This cancels the "MC_CamIn" (→ p. 693) function and starts the new motion function on the slave axis.
- Many applications prefer calling MC_Halt instead of MC_CamOut because it does not send a velocity command to the slave axis.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_CamOut ST example *)
Inst_MC_CamOut (ExecCamOut1, Axis1, CamOut1Acc, CamOut1Dec, CamOut1Jerk);
//Inst_MC_CamOut is an instance of MC_CamOut
```

See Also

- Call a Function Block
- Differences between Functions and Function Blocks
- "MC_CamIn" (→ p. 693)
- "MC_CamTblSelect" (→ p. 707)

3.2.12.3 MC_CamResumePos



Function Block - Returns the slave axis position for resuming an "MC_CamIn" (→ p. 693) move.

- Returns the slave axis position for the most recently executed "MC_CamIn" (→ p. 693) profile, based on the current position of the master axis.
 - This slave axis position can be used to command the slave axis to return to the proper location prior to resuming a MC_CamIn function.
 - When calculating the slave axis position, MC_CamResumePos utilizes the master offset, slave offset, master scaling, and slave scaling of the most recently executed MC_CamIn function block for the slave axis.
- The typical application of MC_CamResumePos is to aid in returning a slave axis back to its profile position after an event (e.g., E-stop) caused the slave axis to go off path.
- See Resuming Camming After an E-Stop for instructions.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

Inputs

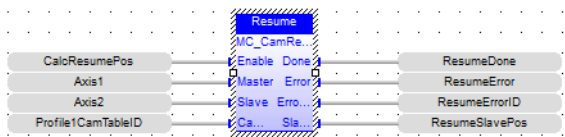
Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	Enables execution.
Master	AXIS_REF	AXIS_NUM - 1 to 256	N/A	No default	• Master axis. • This must be the same as the Master Axis specified for the most recently executed MC_CamIn function block.
Slave	AXIS_REF	AXIS_NUM - 1 to 256	N/A	No default	• Slave axis. • This must be the same as the Slave Axis specified for the most recently executed MC_CamIn function block.
CamTableID	INT	0 to 255	N/A	No default	Profile ID number. • This value is generated by "MC_CamTblSelect" (→ p. 707). • This must be the same as the CamTableID specified for the most recently executed MC_CamIn function block.

Outputs

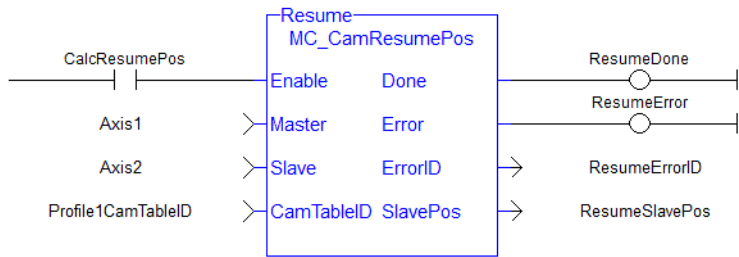
Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	• TRUE = the function block has successfully calculated the slave position. • The slave position is available at the SlavePos output.

Output	Data Type	Range	Unit	Description
Error	BOOL	FALSE, TRUE	N/A	- TRUE = an invalid input was specified or an error occurred in the calculations. - The value at the SlavePos output is undefined.
ErrorID	INT	No range	N/A	Indicates the error if Error output is TRUE.
SlavePos	LREAL	No range	User units	If the Done output is TRUE, this output returns the: - Position for the slave axis given the profile. - Current master axis position. - previously programmed master and slave offsets and scaling.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_CamResumePos( TRUE, Axis1, Axis2, Profile1CamTableID);
```

See Also

- "MC_CamIn" (→ p. 693)
- "MC_CamStartPos" (→ p. 705)

3.2.12.4 MC_CamStartPos



Function Block - Returns the slave axis position for starting an "MC_CamIn" (→ p. 693) move.

- This function block returns the slave axis position for the specified profile, based on the current position of the master axis.
 - This slave axis position can be used to command the slave axis to move to the proper location prior to commanding a "MC_CamIn" (→ p. 693) move with StartMode = 0 (Start mode).
- The typical application of MC_CamStartPos is to aid in positioning a slave axis to its starting position for a MC_CamIn move with a slave absolute profile.
- See Positioning an Axis Before Starting Camming for instructions.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

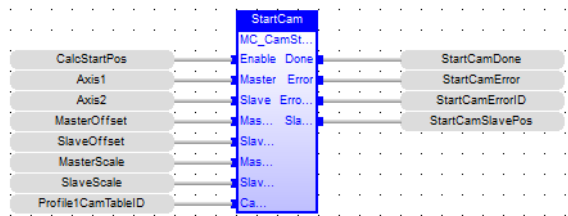
Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	Enables execution.
Master	AXIS_REF	AXIS_NUM - 1 to 256	N/A	No default	Master axis.
Slave	AXIS_REF	AXIS_NUM - 1 to 256	N/A	No default	Slave axis.
MasterOffset	LREAL	No range	User units	No default	• Profile shift along the master axis. • This input is not used if the StartMode input is set to 1 for Resume Mode.
SlaveOffset	LREAL	No range	User units	No default	• Profile shift along the slave axis. • This input is not used if the StartMode input is set to 1 for Resume Mode.
MasterScaling	LREAL	No range	User units	No default	• Master axis scale factor. • Scaling must be a positive value greater than 0 (zero).
SlaveScaling	LREAL	No range	User units	No default	• Slave axis scale factor. • Scaling must be a positive value greater than 0 (zero).
CamTableID	INT	0 to 255	N/A	No default	• Profile ID number. • This value is generated by "MC_CamTblSelect" (→ p. 707).

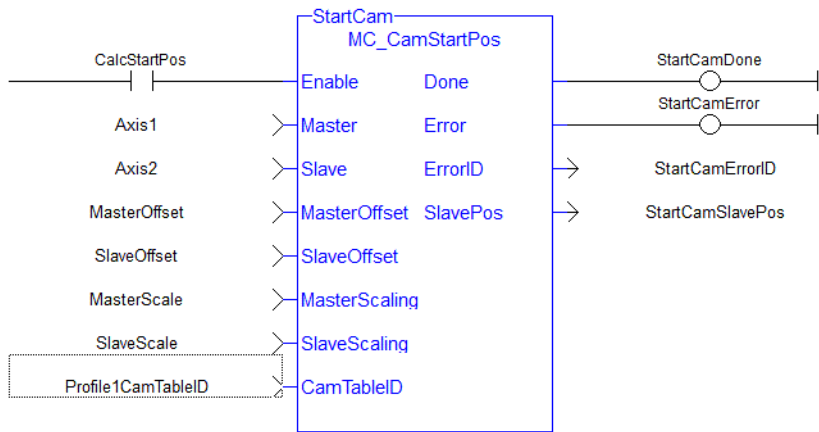
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	- TRUE = the function block has successfully calculated the slave position. - The slave position is available at the SlavePos output.
Error	BOOL	FALSE, TRUE	N/A	- TRUE = an invalid input was specified or an error occurred in the calculations. - The value at the SlavePos output is undefined.
ErrorID	INT	No range	N/A	Indicates the error if Error output is TRUE.
SlavePos	LREAL	No range	User units	If the Done output is TRUE, this output returns the: - Position for the slave axis given the profile. - Current master axis position.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_CamStartPos( TRUE, Axis1, Axis2, MasterOffset, SlaveOffset,
MasterScale, SlaveScale, Profile1CamTableID);
```

3.2.12.5 MC_CamTblSelect



 **Function Block** - Defined to read and initialize the specified profile.
Selects the Cam tables by setting the pointers to the relevant tables.

- Returns an ID to be used with "MC_CamIn" (→ p. 693).
- See Main for information about how this function block is used in the Hole punch project.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to initialize the CAM profile.
CamTable	STRING	No range	N/A	No default	Profile name as defined in the CAM Profile Properties dialog.
Periodic	BOOL	FALSE, TRUE	N/A	No default	• If TRUE, makes the profile is periodic. • See Usage.
MasterAbsolute	BOOL	FALSE, TRUE	N/A	No default	• If TRUE, use Absolute Master position. • If FALSE, use Relative Master position. • See Usage.
SlaveAbsolute	BOOL	FALSE, TRUE	N/A	No default	• If TRUE, use Absolute Slave position. • If FALSE, use Relative Slave position. • See Usage.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates whether this function block has completed without error.
Busy	BOOL	FALSE, TRUE	N/A	Indicates this function block is executing.
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified.
ErrorID	INT	No range	N/A	• Indicates the error if the Error output is high.
CamTableID	INT	0 to 255	N/A	CAM table identifier used with other CAM function blocks.

Remarks

3.2.12.5.1.1 Usage

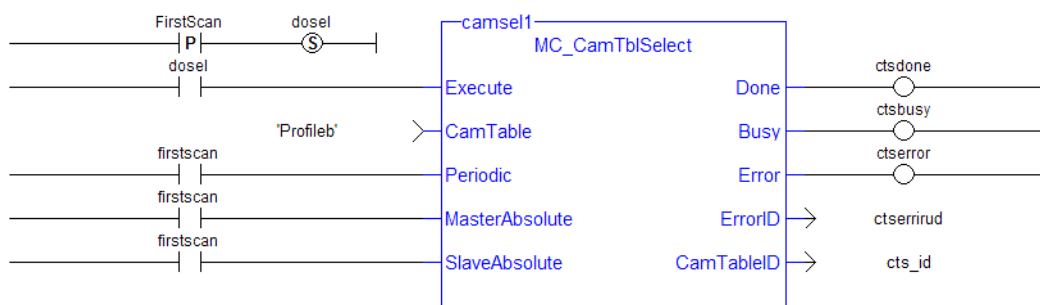
- Each positive transition of the **Enable** input creates a unique Cam ID and store the profile information in a table.

- The number of unique Cam IDs is limited to 256.
- If the application attempts to create more than 256 Cam IDs:
 - The **Error** output is TRUE.
 - The **ErrorID** output is 22 (Too Many Profiles).
- It is only necessary to call MC_CamTblSelect once for each Profile/Periodic/MasterAbsolute/SlaveAbsolute configuration to be used.
- The **Periodic** input specifies if the profile is to repeat each cycle.
 - If the profile is not periodic and the master axis moves beyond the profile range, the slave stops at the end of the profile.
- If the master axis moves back into the profile range, the slave resumes following the profile.
- If the **MasterAbsolute** input is ON, the profile is in reference to the Master axis position.
 - If the MasterAbsolute input is OFF, the profile is in reference to the Master axis position at the time the MC_CamIn function block is executed.
- The **SlaveAbsolute** input specifies if the slave positions are in reference to the Slave axis position or the Slave axis position at the time the MC_CamIn function block is executed.
 - If the **SlaveAbsolute** input is set to TRUE, the axis jumps back to the starting position.
 - If the **SlaveAbsolute** input is set to FALSE, the axis no longer jumps back.
 - Instead, as the profile repeats, the slave moves relative to the start of each period.

FBD Language Example

Not available.

FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_CamTblSelect ST example *) //call this function block every scan until
"Done"
Inst_MC_CamTblSelect(DoSelect, 'Profileb', TRUE, TRUE, TRUE ); //Inst_MC_
CamTblSelect is instance of MC_CamTblSelect
CamSelDone := Inst_MC_CamTblSelect.Done; //store Done output to user defined
variable
IF CamSelDone = TRUE THEN//when function block is "done" store
CamTableID := Inst_MC_CamTblSelect.CamTableID; //CamTableID in user defined
variable
END_IF;
```

See Also

- Call a Function Block
- Differences Between Functions and Function Blocks
- "MC_CamIn" (→ p. 693)
- "MC_CamOut" (→ p. 700)

3.2.12.6 MC_GearIn



Function Block - Performs a slave axis move which follows the master axis using the ratio specified by RatioNumerator and RatioDenominator.

Once engaged, the slave moves RatioNumerator units for every RatioDenominator units the master moves.

```
SlaveCommandPosition = MasterActualPosition * RatioNumerator /
RatioDenominator
```

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.
- When this command is executed, the slave axis accelerates or decelerates (using the Acceleration, Deceleration, and Jerk) to the target velocity determined by the master axis velocity and the ratio.
- When the slave axis reaches a velocity within the In Gear bandwidth around the target velocity, it locks on to the master, and the InGear output goes high.
 - See Axis Parameters.
- When the slave is locked to the master, the slave motion is no longer affected by the acceleration, deceleration, and jerk inputs.
 - Example: If the In Gear bandwidth is set to 0.1 User units / sec, the InGear output turns on if the slave velocity is within +/- 0.1 User units / sec of the target velocity.
 - The slave axis continues to follow the master axis until this move is aborted.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to queue the slave gear ratio move.
Master	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
Slave	AXIS_REF	1 to 256	N/A	No default	AXIS_REF.AXIS_NUM is the slave axis number.
RatioNumerator	DINT	-2147483648 to 2147483647	N/A	No default	• Numerator of slave/master ratio. • A positive ratio indicates the slave axis moves in the same direction as the master axis. • A negative ratio indicates the slave axis moves in the opposite direction as the master axis.
RatioDenominator	DINT	-2147483648 to 2147483647	N/A	No default	• Denominator of slave/master ratio. • A positive ratio indicates the slave axis moves in the same direction as the master axis. • A negative ratio indicates the slave axis moves in the opposite direction as the master axis.

Input	Data Type	Range	Unit	Default	Description
Acceleration	LREAL	No range	User units / sec ²	No default	- Trapezoidal: Acceleration rate. - S-curve: Maximum acceleration.
Deceleration	LREAL	No range	User units / sec ²	No default	- Trapezoidal: Deceleration rate. - S-curve: Unused.
Jerk	LREAL	No range	User units / sec ³	No default	- Trapezoidal: 0 (zero). - S-curve: Constant jerk.
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. - BM_ABORTING - BM_BUFFERED - See Buffer Modes.

Outputs

Output	Data Type	Range	Unit	Description
InGear	BOOL	FALSE, TRUE	N/A	Indicated the slave axis is locked on to the master axis.
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input goes high until the time the move is ended.
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	Indicates either: - An invalid input was specified. - The move was terminated due to an error.
ErrorID	INT	No range	N/A	Indicates the error if Error output is TRUE.

Remarks

3.2.12.6.1.1 Aborting Gearing

Gearing functions can generate large accelerations while following the master.

- If the aborting function block has small, non-zero Jerk, or small acceleration values, it can take a long time for an accelerating axis to reach the target velocity, or position of the aborting function block.
 - If the Jerk and/or acceleration of the aborting function cannot be increased to suitable values, it may be desirable to either:
 - Abort the gearing function with an "MC_GearOut" (→ p. 722) with higher accelerations and/or Jerk values (or zero jerk value).
 - Execute the next MC motion function block (e.g., "MC_Halt" (→ p. 661)).

3.2.12.6.2.2 Time to Reach the Target Velocity

While following the master, gearing functions can generate large accelerations.

- If the gearing function is aborted while the axis is currently accelerating, and the aborting function block has small non-zero Jerk or small acceleration values, it can take a long time to reach the target velocity, or position of the aborting function block.
 - If the Jerk and/or acceleration of the aborting function cannot be increased to suitable values, it may be desirable to either:
 - Abort the gearing function with an "MC_GearOut" (→ p. 722) with higher accelerations and/or Jerk values (or zero jerk value).
 - Execute the next MC motion function block (e.g., "MC_Halt" (→ p. 661)).

Time Diagram

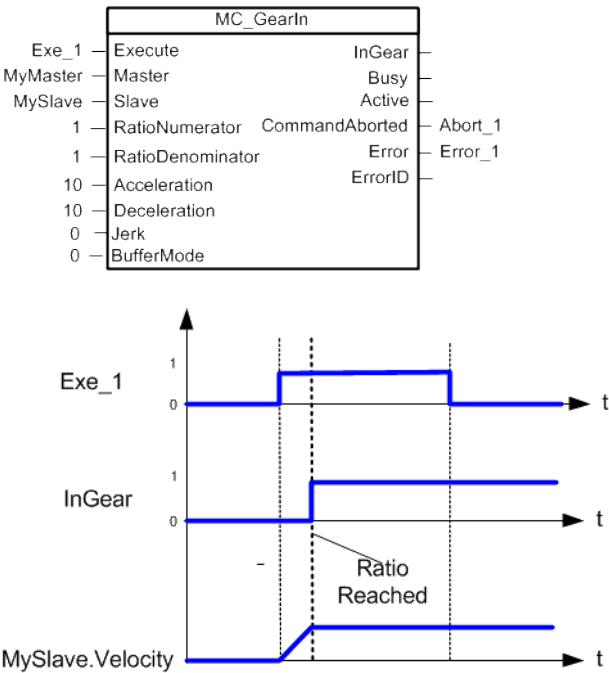
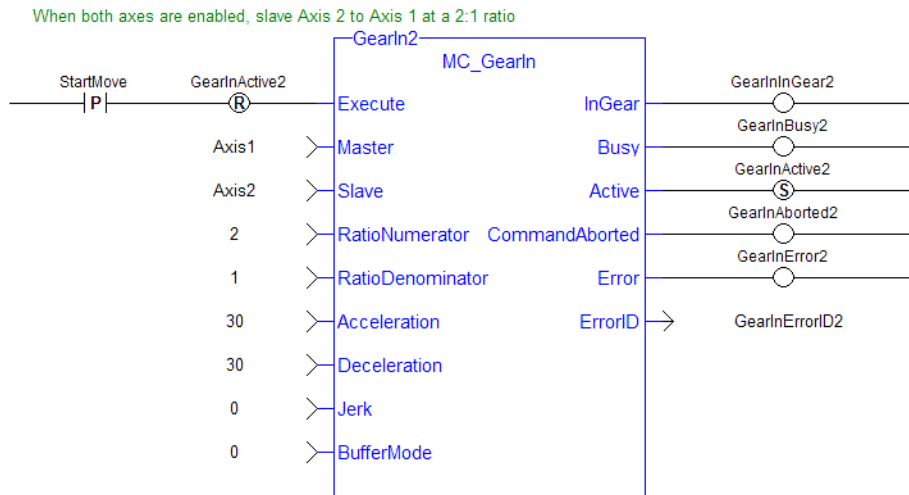


Figure 4-88: Time Diagram

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_GearIn ST example *)
Inst_MC_GearIn( GearInReq, Axis1, Axis2, 2, 1, 150.0, 150.0, 0, 0 );
//Inst_MC_GearIn is an instance of MC_GearIn
```


3.2.12.7 MC_GearInPos



Function Block - Performs a slave axis move which synchronizes motion at the SlaveSyncPosition and MasterSyncPosition.

Once synchronized, the slave moves RatioNumerator units for every RatioDenominator units the master moves.

```
SlaveCommandPosition = MasterActualPosition * RatioNumerator /
RatioDenominator
```

- This function block allows the application to specify sync positions for the master and slave axes.
 - It is the point in which the master and slave axes become engaged in synchronous motion.
- When the master axis reaches the MasterStartDistance from the MasterSyncPosition, the slave axis begins to accelerate to the target velocity determined by the master axis velocity and the ratio.
 - The slave axis arrives at the target velocity and the SlaveSyncPosition at the same time the master axis arrives at the MasterSyncPosition.
 - At that time, the slave is locked on to the master and follows the master at the ratio specified.
 - The slave axis continues to follow the master axis until this move is aborted.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to queue the slave gear ratio move.
Master	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
Slave	AXIS_REF	1 to 256	N/A	No default	• AXIS_REF.AXIS_NUM is the slave axis number.
RatioNumerator	DINT	-2147483648 to 2147483647	N/A	No default	• Numerator of slave/master ratio. • A positive ratio indicates the slave axis moves in the same direction as the master axis. • A negative ratio indicates the slave axis moves in the opposite direction as the master axis.

Input	Data Type	Range	Unit	Default	Description
RatioDenominator	DINT	-2147483648 to 2147483647	N/A	No default	- Denominator of slave/master ratio. - A positive ratio indicates the slave axis moves in the same direction as the master axis. - A negative ratio indicates the slave axis moves in the opposite direction as the master axis. - For a master moving in the: - Positive direction, use a positive RatioDenominator. - Negative direction, use a negative RatioDenominator.
MasterSyncPosition	LREAL	-1.7E308 to 1.7E308	N/A	No default	- Master axis sync position. 14 to 15 significant digits of accuracy.
SlaveSyncPosition	LREAL	-1.7E308 to 1.7E308	N/A	No default	- Slave axis sync position. 14 to 15 significant digits of accuracy.

Input	Data Type	Range	Unit	Default	Description
SyncMode	INT	0 to 1	N/A	No default	SyncMode determines the allowed conditions for synchronization. 0 = Normal Synchronization • Prior to executing the MC_GearInPos function block, the Master axis position must be before the MasterSyncPosition by a distance greater than the MasterStartDistance. • The Slave axis position must be before the SlaveSyncPosition. • Axes that have a non-zero rollover, the MC_GearInPos function block always assumes the axes meet these conditions by assuming the sync point is in the next occurrence of the sync position. • MasterStartDistance must be positive and greater than the distance the master axis is currently moving per axis update. • If the master start distance and the slave axis distance from the SlaveSyncPosition are sufficiently large enough, the slave axis ramps to the sync position. • If not sufficiently large enough, acceleration of the slave axis may be excessive. 1 = Immediate Synchronization Allowed • This mode is only allowed if both the master and slave axes have rollover = 0 (zero). • If the conditions of SyncMode = 0 (zero) are not met, Synchronization is allowed even though the axis positions may be beyond their respective Sync Positions. • The MasterStartDistance may be 0 (zero). • If the

Input	Data Type	Range	Unit	Default	Description
					MasterStartDistance is 0 (zero), the Slave axis synchronizes with the master the instant the master axis crosses the MasterSyncPosition. • If either the master or slave axis are beyond their respective sync start positions, the slave axis immediately synchronizes to the master axis. • If the master start distance and the slave axis distance from the SlaveSyncPosition are sufficiently large enough, the slave axis ramps to the sync position. • If not sufficiently large enough or immediate synchronization occurs, slave axis acceleration may be excessive. • Excessive slave acceleration may occur if the master axis velocity is large or the master and slave axes have disproportionally different distances to their respective sync positions. • If the slave axis is ahead of the master axis at the time of synchronization, the slave axis moves backwards.

Input	Data Type	Range	Unit	Default	Description
MasterStartDistance	LREAL	-1.7E308 to 1.7E308	User units	No default	When the master axis reaches this distance before MasterSyncPosition, the slave axis begins its lock-on process. 14 to 15 significant digits of accuracy. - The MasterStartDistance * (RatioNumerator/RatioDenominator) should be greater than (or equal to) the slave sync distance. - The slave sync distance is defined as the distance between the slave position when MC_GearInPos executes and the SlaveSyncPosition. - If the MasterStartDistance is too short, the MC_GearInPos may have excessive acceleration and a warning log message is generated.
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. - BM_BUFFERED - See Buffer Modes.

Outputs

Output	Data Type	Range	Unit	Description
StartSync	BOOL	FALSE, TRUE	N/A	Indicates the master axis has reached the MasterStartDistance from the MasterSyncPosition and the lock-on process has begun.
InSync	BOOL	FALSE, TRUE	N/A	Indicated the slave axis is locked on to the master axis.
Busy	BOOL	FALSE, TRUE	N/A	High from the moment the Execute input goes high until the time the move is ended.
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block. - If the abort arises because the inputs cause inconsistent motion, then this FB: - Performs no motion. - Sets an error flag. - Set the ErrorID to 13.

Output	Data Type	Range	Unit	Description
Error	BOOL	FALSE, TRUE	N/A	Indicates either: - An invalid input was specified. - The move was terminated due to an error.
ErrorID	INT	No range	N/A	Indicates the error if Error output is TRUE.

Remarks

3.2.12.7.1.1 Aborting Gearing

Gearing functions can generate large accelerations while following the master.

- If the aborting function block has small, non-zero Jerk, or small acceleration values, it can take a long time for an accelerating axis to reach the target velocity, or position of the aborting function block.
 - If the Jerk and/or acceleration of the aborting function cannot be increased to suitable values, it may be desirable to either:
 - Abort the gearing function with an "MC_GearOut" (→ p. 722) with higher accelerations and/or Jerk values (or zero jerk value).
 - Execute the next MC motion function block (e.g., "MC_Halt" (→ p. 661)).

3.2.12.7.2.2 Time to Reach the Target Velocity

While following the master, gearing functions can generate large accelerations.

- If the gearing function is aborted while the axis is currently accelerating, and the aborting function block has small non-zero Jerk or small acceleration values, it can take a long time to reach the target velocity, or position of the aborting function block.
 - If the Jerk and/or acceleration of the aborting function cannot be increased to suitable values, it may be desirable to either:
 - Abort the gearing function with an "MC_GearOut" (→ p. 722) with higher accelerations and/or Jerk values (or zero jerk value).
 - Execute the next MC motion function block (e.g., "MC_Halt" (→ p. 661)).

MC_GearInPos	
Execute	StartSync
Master	InSync
Slave	Busy
RatioNumerator	Active
RatioDenominator	CommandAborted
MasterSyncPosition	Error
SlaveSyncPosition	ErrorID
SyncMode	
MasterStartDistance	
BufferMode	

Figure 4-89: MC_GearInPos

Time Diagram

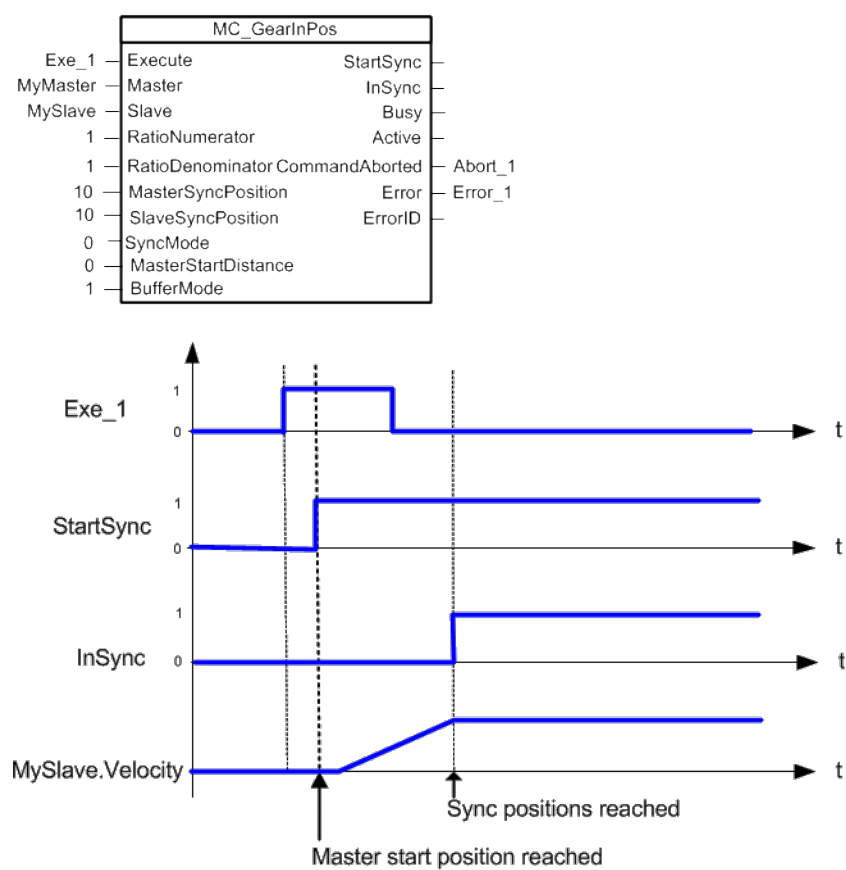
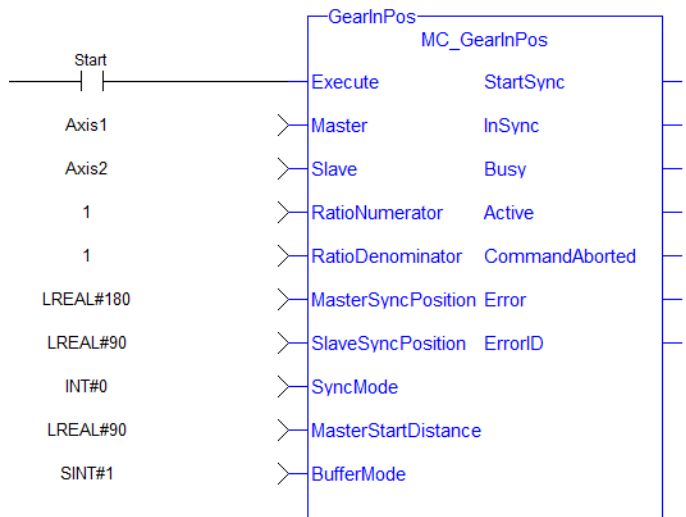


Figure 4-90: Time Diagram

3.2.12.7.3.3 Example

- Master and Slave are rotary axes with rollovers at 360 degrees.
- The Master initial position is 0 degrees and the slave initial position is 45 degrees.
- The GearInPos FB commands the slave to accelerate up to the geared ratio (1:1) during the master start distance (90 degrees) and be synchronized with the master at the master and slave sync positions.

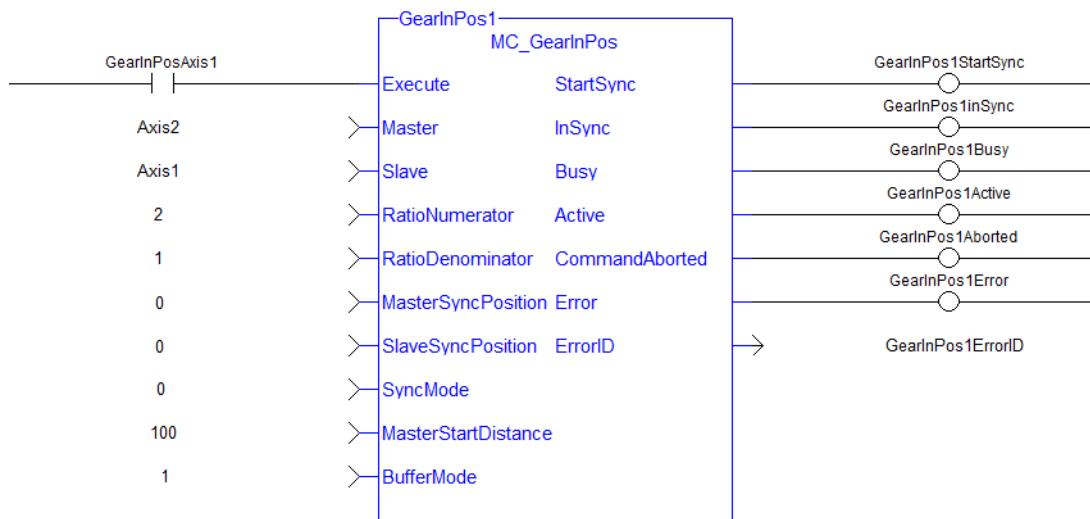


1. Master Axis
2. Slave Axis
3. Master Start Distance
4. Slave starts to accelerate
5. Master Sync Position
6. Slave Sync Position

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_GearInPos ST example *)

Inst_MC_GearInPos( GearInPosReq, Axis1, Axis2, 2, 1, 0, 0, 0, 100.0, 1 );
//Inst_MC_GearInPos is instance of MC_GearInPos
GearInPosSync:= Inst_MC_GearInPos.InSync;
//store InSync output into user defined variable
```


3.2.12.8 MC_GearOut



Function Block - Disengages the slave axis from a MC_GearIn or MC_GearInPos move or the master axis.

This function block:

- Aborts the active MC_GearIn or MC_GearInPos move.
- Disengages the axis from its master.
- Commands the axis to continue at its current velocity.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

TIP

The current velocity is calculated by taking the average of the actual velocity during the previous 16 cycles.

- The control continues to command the axis to move at this velocity until this MC_GearOut move is aborted.
- The Acceleration, Deceleration, and Jerk input parameters are applied if this command velocity is modified by the [MC_SetOverride](#) function block.
- If this function block is called, and the active move is not a [MC_GearIn](#) or [MC_GearInPos](#) move, this function block returns an error and the active move is not aborted.

NOTE

- The MC_GearOut is done when the slave axis is disengaged from the master axis.
 - Once done, the MC_GearOut remains busy and active until it is aborted by a different motion function block.
- This is different behavior than most other motion function blocks.
 - The MC_GearOut function block represents an exception to the exclusivity rule as the **Done** and **Active** outputs may be true at the same time.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to disengage the slave axis from a "MC_GearIn" (→ p. 710) or "MC_GearInPos" (→ p. 714) move.
Slave	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
Acceleration	LREAL	No range	User units / sec ²	No default	• Trapezoidal: Acceleration rate. • S-curve: Maximum acceleration.
Deceleration	LREAL	No range	User units / sec ²	No default	• Trapezoidal: Deceleration rate. • S-curve: Unused.
Jerk	LREAL	No range	User units / sec ³	No default	• Trapezoidal: 0 (zero). • S-curve: Constant jerk.

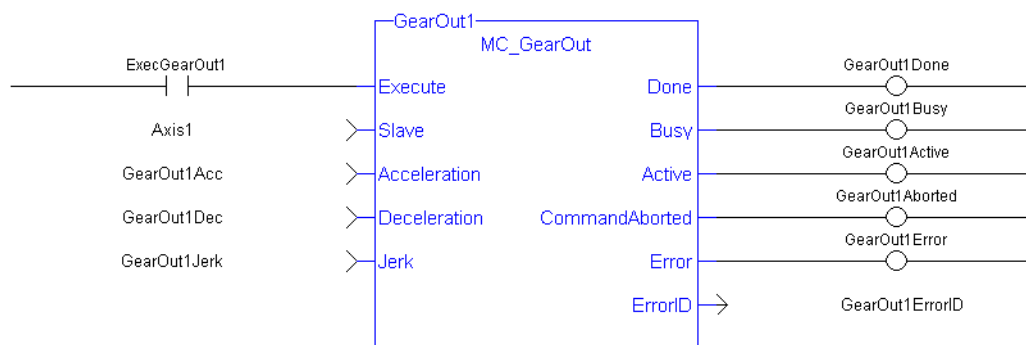
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the axis is disengaged from its master.
Busy	BOOL	FALSE, TRUE	N/A	Indicates the function is executing.
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	Indicates either: - An invalid input was specified. - No MC_GearIn or MC_GearInPos move is active.
ErrorID	INT	No range	N/A	Indicates the error if Error output is TRUE.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_GearOut ST example *)

Inst_MC_GearOut(ExecGearOut1,Axis1,GearOut1Acc,GearOut1Dec,GearOut1Jerk);
//Inst_MC_GearOut is instance of MC_GearOut
```

See Also

- Call a Function Block
- Differences Between Functions and Function Blocks

3.2.12.9 MC_Phasing



Function Block - Performs a master position phase shift for the slave axis.

This function block:

- Provides a way to smoothly apply a master offset instead of writing values directly to the Master Offset Parameter 1002.
- Is commonly used with "MC_TouchProbe" (→ p. 633) for performing position corrections on the slave axis in a **Mark-to-Mark** registration application.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

TIP

- **MC_Phasing** performs a similar function to adjusting the MasterOffset input in the "MC_CamIn" (→ p. 693) function block.
- It has the additional features of setting the velocity, acceleration, deceleration, and jerk motion parameters.
- The distance entered at the **PhaseShift** input is iterated into the Slave axis's Master Offset.
 - This distance is iterated like a "MC_MoveRelative" (→ p. 677) move using the specified **Velocity**, **Acceleration**, **Deceleration**, and **Jerk** values.
- The difference in the interpolated command delta is not commanded to the axis but is added to the Slave axis's Master Offset.
 - This shifts the Master axis's position as viewed by the Slave axis, causing a change in the Slave axis's physical position.
 - This only affects the Slave axis if it is executing a slave move.
- Subsequent calls to MC_Phasing can abort or blend to an executing MC_Phasing command.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to queue the phase shift.
Master	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
Slave	AXIS_REF	1 to 256	N/A	No default	• AXIS_REF.AXIS_NUM is the slave axis number.
PhaseShift	LREAL	No range	User units	No default	Amount of phase shift.
Velocity	LREAL	No range	User units / sec	No default	Velocity setpoint.
Acceleration	LREAL	No range	User units / sec ²	No default	• Trapezoidal: Acceleration rate. • S-curve: Maximum acceleration.

Input	Data Type	Range	Unit	Default	Description
Deceleration	LREAL	No range	User units / sec ²	No default	- Trapezoidal: Deceleration rate. - S-curve: Unused.
Jerk	LREAL	No range	User units / sec ³	No default	- Trapezoidal: 0 (zero). - S-curve: Constant jerk.
BufferMode	SINT	Enumerated	N/A	No default	The specified buffer mode. - BM_ABORTING - BM_BUFFERED - BM_BLENDING_PREVIOUS - BM_BLENDING_NEXT - BM_BLENDING_LOW - BM_BLENDING_HIGH - See Buffer Modes.

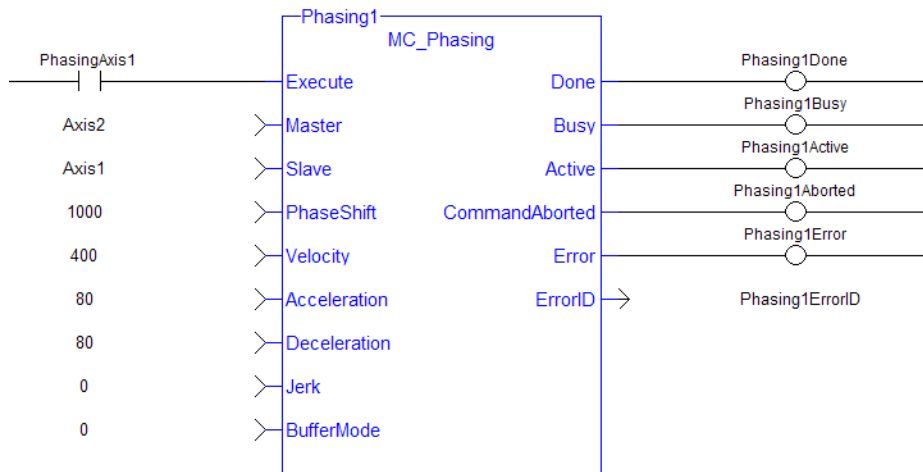
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the phase shift has been completely applied.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Active	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is controlling motion.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	No range	N/A	Indicates the error if Error output is TRUE.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_Phasing ST example *) //Inst_MC_Phasing is an instance of MC_Phasing
function block
Inst_MC_Phasing(PhasingAxis1, Axis2, Axis1, 1000.0,100.0, 200.0, 200.0, 0, 0
);
```

3.2.12.10 MC_SyncSlaves



 **Function Block** - Allows the application to specify what slave axes are to be synchronized and which master they follow.

- After this function block is executed successfully, all the slave axes specified at the **SlaveList** input start their slave moves (i.e., **MC_CamIn**, **MC_GearIn**, etc.) on the same servo interrupt for a synchronized slave start.
- When a slave move is commanded for one of the slave axes listed, the slave move is queued but the motion is held off until all of the listed slaves have queued their slave moves.
- Call MC_SyncSlaves to specify the slave axes to synchronize.
- Call each slave move (e.g., **MC_GearIn**) for each slave axis.
 - The motion is held off until all the slave moves have been queued.
- After all the slave moves have been queued, the interpolation for all the slave axes begin on the same servo interrupt, providing a synchronized start.
- The master axis can be in motion prior to this sequence or the master can be commanded after all the slave moves are queued.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to queue the synchronization of a list of slave axes.
Master	AXIS_REF	1 to 256	N/A	No default	Master axis identifier.
SlaveCount	AXIS_REF	1 to 256	N/A	No default	The number of slave axes listed in the SlaveList array input that are to be synchronized. • This number must not be greater than the declared size of the SlaveList array. • If this number is 0 (zero), the list of synchronized slaves for the specified Master axis is cleared. • The AXIS_NUM element of the AXIS_REF structure must be in the range. • See AXIS_REF Structure.
SlaveList	UINT	1 to 32	N/A	No default	The list of slave axes that are to be synchronized. • Each element of this array contains a unique axis number. • The axis number must not be the same as the Master axis number.

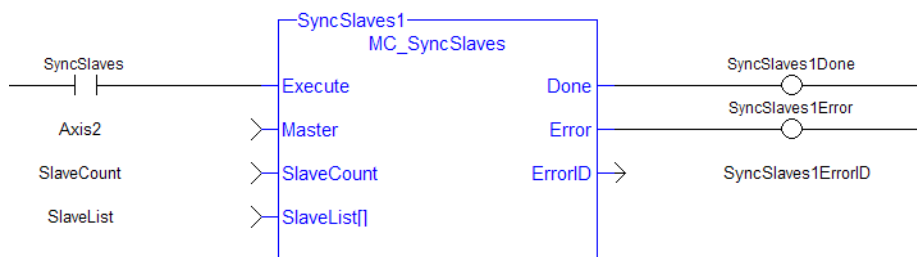
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the synchronized slave assignments were completed without error.
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified.
ErrorID	INT			Indicates the error if Error output is TRUE.

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* MC_SyncSlaves ST example *)
// Inst_MC_SyncSlaves is an instance of MC_SyncSlaves function block
Inst_MC_SyncSlaves( SyncSlaves, Axis1, SlaveCount, SlaveList );
```

See Also

- "MC_CamIn" (→ p. 693)
- "MC_GearIn" (→ p. 710)
- "MC_GearInPos" (→ p. 714)

3.2.13 Reference

These functions provide commands for reference points.

Name	Description
MC_Reference	Used to execute a fast home to a switch.
MC_SetPos	Changes the present actual position of the axis (as reported by "MC_ReadActPos" (→ p. 642)) to the position specified by the Position and Mode inputs.
MC_SetPosition	Deprecated - See "MC_SetPos" (→ p. 736).

3.2.13.1 MC_Reference



Function Block - Used to execute a fast home to a switch.

ⓘ IMPORTANT

- Position capture is not available for PLCopen axes assigned to the secondary feedback input (digitizing axes).
- MC_Reference cannot be used to home digitizing axes.
- If the application selects to reference to the index mark of an encoder, or the null of a resolver (which is typical), the new position value is assigned to the position of the index of the encoder (or the null of the resolver) and not the position of the switch.
- The **ECATWriteSDO** function block is used to setup the trigger event and any desired preconditions.
- This function block utilizes the Position Capture Mode of the AKD.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

⚡ TIP

There are differences between how an AKD and an AKD2G are used with this function block.

- "AKD Support with MC_TouchProbe" (→ p. 638)
- "AKD2G Support With MC_TouchProbe" (→ p. 639)

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to queue the move and arms reference trigger events.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.

Input	Data Type	Range	Unit	Default	Description
TriggerInput	TRIGGER_REF	See Description.	N/A	No default	TRIGGER_REF structure defines the trigger. DirectionINT Range = 1 to 2. 1 = Rising edge. 2 = Falling edge. Valid values of InputID are: 0 = Touch Probe 1 / Capture Engine 0 1 = Touch Probe 2 / Capture Engine 1 TrigidINT - Must be 0 (zero). NOTE TrigMode INT (TriggerInput.TrigMode) is not presently supported by this function.
Velocity	LREAL	No range	User units / sec	No default	Commanded velocity for the reference move.
Acceleration	LREAL	No range	User units / sec ²	No default	Commanded acceleration for the reference move.
Deceleration	LREAL	No range	User units / sec ²	No default	Commanded deceleration for the reference move.
Jerk	LREAL	No range	User units / sec ³	No default	- Commanded jerk for the reference move. - If 0 (zero), the trapezoidal acc/dec is used.
Direction	SINT	0 to 1	N/A	No default	Commanded direction of the reference.
Position	LREAL	No range	User units	No default	The position the axis is reset to when at the machine reference location.

Input	Data Type	Range	Unit	Default	Description
Option	SINT	0 to 15	N/A	No default	Option identifier for Resolvers/Modulo reference. 0 = Use latched position for reference. 1 = Use resolver position of nearest null for reference 2 pole resolver. 2 = Use resolver position of nearest null for reference 4 pole resolver. 3 = Use resolver position of nearest null for reference 6 pole resolver. 4 = Use resolver position of nearest null for reference 8 pole resolver. 5 = Use resolver position of nearest null for reference 10 pole resolver. ... 15 = Use resolver position of nearest null for reference 30 pole resolver.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the reference move and position adjustment is complete.
Busy	BOOL	FALSE, TRUE	N/A	Indicates this function block is executing.
Active	BOOL	FALSE, TRUE	N/A	Indicates this move is the Active move.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	Indicates either: - An invalid input was specified. - The move was terminated due to an error.
ErrorID	INT			Indicates the error if Error output is high.

Remarks

3.2.13.1.1.1 Usage

TIP

Once the **MC_Reference** block is queued, but before it is completed, the cycle can be aborted by either:

- Using a "MC_Halt" (→ p. 661) or "MC_Stop" (→ p. 627) function block.
- By queuing a new motion function block with the Abort selected for buffer mode.

These are steps for homing a PLCopen axis using the MC_Reference function block.

Not all of the steps are necessary depending on the configuration and the homing cycle design.

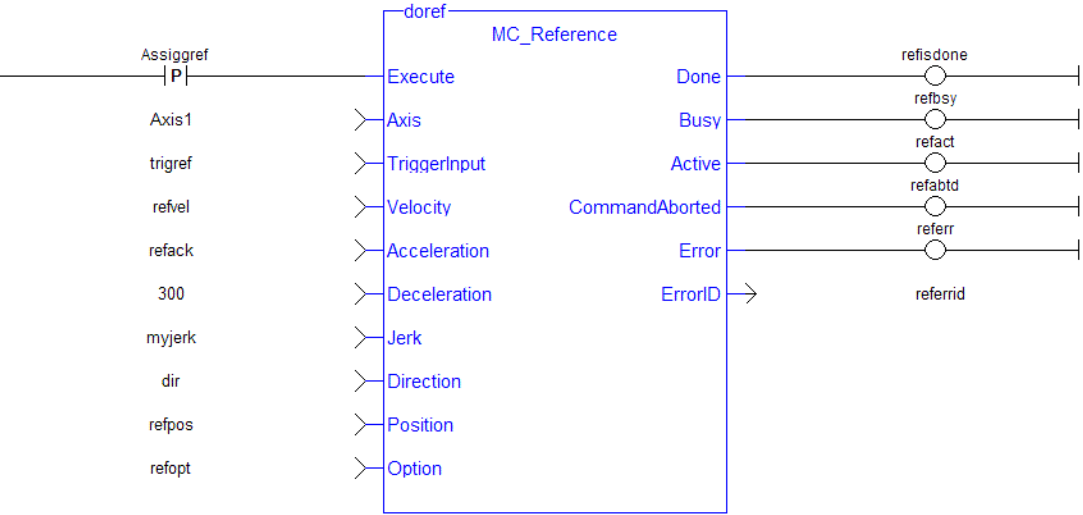
1. Verify the Axis is not on Reference switch.
 - If a switch is used in the homing cycle for the event or precondition to the event, verify the axis is not already tripping the switches that trigger the event and precondition.
 - If it is, move the axis off the switches.
2. Configure the AKD capture engine.
 - Configuration of the AKD capture engine is performed by writing drive CAN objects via SDO.
 - Use the **ECATWriteSdo** function for this action.
 - The AKD Capture mode must be set to **POSITION CAPTURE**.
 - See AKD Capture Engine Configuration for information about the available configurations.
 - See PLCopen Homing Methods for example AKD capture engine configurations and references.
3. Call the **MC_Reference** function to initiate optional homing motion and to arm the AKD capture engine.
 - **MC_Reference** selects the trigger edge (rising or falling edge) and arms the capture.
 - Then, it optionally moves the axis to the reference location as directed by inputs to this function.
 - When the AKD indicates the capture event has occurred, the coordinate system is shifted so the reference position input to this function block is set to the reference location and the reference motion is stopped.
4. Wait for the completion of the **MC_Reference** function block.
 - The application is notified by the completion, abort, or error of the homing by the **MC_Reference** function block.
5. When the **MC_Reference** function block is completed, the axis can be moved to the home position with a **MC_MoveAbsolute** function block.

FBD Language Example

Not available.

FFLD Language Example

Network #18
Command the Reference move, specifying the Trigger, Velocity, Acc/Dec, direction, Reference position and Options.



IL Language Example

Not available.

3.2.13.1.2.2 ST Language Example

```
(* MC_Reference ST example *)  
TriggerInput.InputID := 0;    //configure the reference InputID  
TriggerInput.DIRECTION := 1; //configure the reference direction  
Inst_MC_Reference( RefReq, Axis1, TriggerInput, 20.0, 100.0, 100.0, 100.0, 0,  
0.0, 0 );
```

See Also

- "ECATWriteSdo" (→ p. 89)
- "MC_MoveAbsolute" (→ p. 664)

3.2.13.2 MC_SetPos



 **Function Block** - Changes the present actual position of the axis (as reported by "MC_ReadActPos" (→ p. 642)) to the position specified by the **Position** and **Mode** inputs.

- If a motor is associated with the axis, it does not move when MC_SetPos is executed.
 - MC_SetPos does not cause any motion.
 - It applies an offset to the command and actual positions.
- MC_SetPos sets the accumulated Superimposed distance value for the input axis to 0 (zero).
 - See Axis Positions Data.
- This function block replaces **MC_SetPosition**.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to queue the change to the axis position.
Axis	AXIS_REF	1 to 256	N/A	No default	Name of a declared instance of the AXIS_REF library function.
Position	LREAL	No range	N/A	No default	• Absolute Mode: New Axis Position to replace the present position. • Relative Mode: Position offset to apply to present position. • Typically used with multi-turn absolute position feedback devices.
Mode	BOOL	FALSE, TRUE	N/A	No default	• LOW = Position input is an absolute position. • HIGH = Position input is a relative position

Outputs

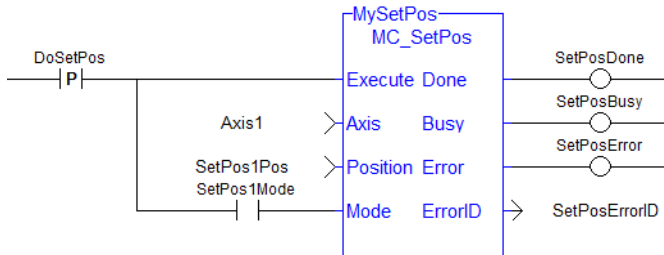
Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indicates the reference move and position adjustment is complete.
Busy	BOOL	FALSE, TRUE	N/A	Indicates this function block is executing.
Error	BOOL	FALSE, TRUE	N/A	Indicates either: • An invalid input was specified. • The move was terminated due to an error.

Output	Data Type	Range	Unit	Description
ErrorID	INT	-32768 to +32767		Indicates the error if Error output is high.

FBD Language Example

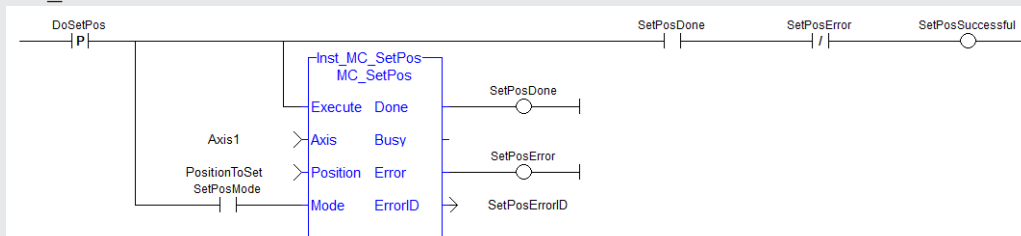
Not available.

FFLD Language Example



❗IMPORTANT

- This function block finishes immediately.
 - Because of this, the **Done** output does not get set to FALSE in a second call to the same MC_SetPos instance unless there is an error.
- If the application needs to look for a state change to determine if a particular call to MC_SetPos was successful, then use AND for the rising edge of the **Execute** input, the **Done** output, and the inverse of the **Error** output.
- This FFLD Language example shows how this can be done.
In the example, the SetPosSuccessful variable is set to TRUE for one cycle upon a successful call to the MC_SetPos instance.



IL Language Example

Not available.

ST Language Example

```
(* MC_SetPos ST example *)

Inst_MC_SetPos ( Axis1 , 0, 0 );
//Inst_MC_SetPos is an instance of MC_SetPos function

(* MC_SetPos absolute mode example: Set position value to zero. *)
Inst_MC_SetPos ( Axis1 , 0, 0 );
//Inst_MC_SetPos is an instance of MC_SetPos function
```



```
(* MC_SetPos relative mode example: Increase position value by 1000. *)  
Inst_MC_SetPos ( Axis1 , 1000, 1 );  
//Inst_MC_SetPos is an instance of MC_SetPos function
```

3.2.13.3 MC_SetPosition - Deprecated

 **Function** - Deprecated by "MC_SetPos" (→ p. 736).

3.2.14 Registration

These function blocks allow for Mark-to-Mark or Mark-to-Machine registration.
See Registration for techniques on setting up and using the registration function blocks.

Name	Description
MC_MachRegist	Enables Mark-to-Machine registration.
MC_MarkRegist	Enables Mark-to-Mark registration.
MC_StopRegist	Turns off registration for the specified axis and disarms the specified fast input.

3.2.14.1 MC_MachRegist



Function Block- Enables Mark-to-Machine registration.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

TIP

- Is this the right function block to use?
 - See Deciding Which Function Blocks to Use for Registration and Registration Application Guide.
- It is used on any servo or digitizing axis and with any move type.
- It is used most frequently in master/slave applications.

Used with ...	Effect
Non-slave moves	Resets the axis position when a good mark is captured by the fast input.
Slave moves	Applies a compensation offset to correct for the difference between the target position and the measured position. This provides the ability to compensate for product or process inconsistencies, provides a system that remains synchronized with no accumulated error, and maintains repeatable accuracy throughout the process.

Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to enable registration.
Axis	AXIS_REF	The range of .AXIS_NUM is [1, 256]	N/A	N/A	Axis whose position is used to determine a good mark.

Input	Data Type	Range	Unit	Default	Description
TriggerInput	TRIGGER_REF	No range	N/A	No default	Structure specifying the fast input. The structure elements are: Direction INT Range = 1, 5. 1 = Rising edge. 2 = Falling edge. 3 = N/A 4 = Toggle between both, falling edge first. 5 = Toggle between both, rising edge first. Valid values of InputID are: 0 = Touch Probe 1 / Capture Engine 0 1 = Touch Probe 2 / Capture Engine 1 TrigID INT Range = 0 to 256. - The axis number of the fast input. 0 (zero) indicates this trigger axis is to be the same as the Axis input. NOTE TrigMode INT (TriggerInput.TrigMode) is not presently supported by this function.
Distance	LREAL	- When converted to feedback units, the range is $[-2^{51}, 2^{51}-1]$. - This value must have the same sign as Ignore.	User units	N/A	- This is the expected distance between good marks. - Along with Tolerance and Ignore, this value is used to determine if the mark detected by the fast input is a good mark.
Tolerance	LREAL	When converted to feedback units, the range is $[0, 2^{51}-1]$.	User units	N/A	This value specifies the distance, plus or minus, about Distance to determine if the mark detected by the fast input is a good mark.

Input	Data Type	Range	Unit	Default	Description
Ignore	LREAL	- When converted to feedback units, the range is $[-2^{51}, 2^{51}-1]$. - This value must have the same sign as Distance.	User units	N/A	This value specifies the distance after the previous good mark in which any detected marks are ignored.
Target	LREAL	When converted to feedback units, the range is: $[-2^{51}, 2^{51}-1]$ if PosAxis' rollover value is zero. $[0, \text{PosAxis' Rollover Value}]$ if PosAxis' rollover value is non-zero (i.e., $\geq 0 < \text{PosAxis' Rollover Value}$).	User units	N/A	- This is the target position. - This position is compared to the actual position captured by the fast input to determine the amount of registration compensation to apply.
Position	LREAL	When converted to feedback units, the range is: $[-2^{51}, 2^{51}-1]$ if PosAxis' rollover value is zero. $[0, \text{PosAxis' Rollover Value}]$ if PosAxis' rollover value is non-zero (i.e., $\geq 0 < \text{PosAxis' Rollover Value}$).	User units	N/A	- The position the axis is set to when a good registration mark occurs. - If the Inhibit Reference on Good Mark option is specified for the Option argument, this argument is not used. - The position of the axis is not changed when a registration mark is encountered.
PosAxis	AXIS_REF	The range of .AXIS_NUM is [1, 256]	N/A	N/A	- The position of this axis at the time the fast input occurs is compared to the Target position. - This comparison determines the amount of registration compensation to apply.
CompAxis	AXIS_REF	The range of .AXIS_NUM is [1, 256]	N/A	N/A	The calculated registration compensation is applied to this axis.

Input	Data Type	Range	Unit	Default	Description
PosTolerance	LREAL	When converted to feedback units, the range is $[-2^{51}, 2^{51}-1]$.	User units	N/A	This value specifies the distance, plus or minus, about the Target position to determine if the position is accepted and the compensation value is calculated and applied.
RatioNumerator	DINT	When converted to feedback units, the range is $[1, 4294967295]$.	User units	N/A	- This value is typically the number of User Units of CompAxis motion for one product cycle. - This value is used with RatioDenominator to create a conversion factor for calculating the compensation value when PosAxis and CompAxis are different axes.
RatioDenominator	DINT	When converted to feedback units, the range is $[1, 4294967295]$.	User units	N/A	- This value is typically the number of User Units of PosAxis motion for one product cycle. - This value is used with RatioNumerator to create a conversion factor for calculating the compensation value when PosAxis and CompAxis are different axes.
Options	UINT	See the "MC_MachRegist Options Table" (→ p. 746).	N/A	N/A	- Each bit enables / disables an option. - The "MC_MachRegist Options Table" (→ p. 746) defines the bits. - Any bits not defined are reserved. - The third bit, 0004H, must be 0 (zero).

Outputs

Output	Data Type	Range	Unit	Description
RegistOn	BOOL	FALSE, TRUE	N/A	Indicates registration is activated.
Aborted	BOOL	FALSE, TRUE	N/A	Indicates registration has been terminated by "MC_StopRegist" (→ p. 755).

Output	Data Type	Range	Unit	Description
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified or the registration was terminated due to an error.
ErrorID	INT	No range	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

Remarks

3.2.14.1.1.1 Transition and Registration

Transition and Registration

- A positive transition of the **En** input starts registration.
 - The application may change the registration parameters while registration is active by changing the input values and causing another positive transition of the **En** input.
 - The function block then reads and applies the new values.
- The axis number at the **Axis** input indicates the axis whose position, at the fast input, is used to determine if the mark is a good mark.

3.2.14.1.2.2 Distance, Tolerance, Ignore, Target Inputs

Distance, Tolerance, and Ignore Inputs

- The **Distance**, **Tolerance**, and **Ignore** inputs are used to determine whether or not the registration mark is good.
 - For a mark to be recognized as good, it must be outside of the **Ignore** distance and the correct **Distance** from the previous mark +/- the **Tolerance** window.
 - A mark is bad if it occurs outside of the "good tolerance band" and is not ignored.
 - Both good marks and bad marks are recognized as marks, ignored marks are not recognized.
 - If all marks are to be recognized as good marks, enter 0 (zero) at both **Distance** and **Tolerance**.
- The **Distance** value defines the distance between good marks.
 - In Clear Lane and Product registration the **Distance** input value is typically the same as the **Target** input value.
 - In **Print** registration, the **Distance** is typically not the same as **Target**.
- The **Tolerance** value is the distance, plus and minus, about **Distance**.
 - Marks detected in this window are good marks and registration occurs.
 - Marks detected outside this window and outside the **Ignore** band, are bad marks and registration does not occur.
 - This window should be large enough to allow for the worst case error in the distance between the previous mark and the current mark.
- The **Ignore** value defines the distance from the previous mark where all marks detected by the fast input are ignored.
 - This is crucial when registering products that do not have Clear Lane registration marks.

Target Input

- The **Target** input is the expected target position that is used to calculate how much registration compensation is to be applied when a registration mark is considered good.
 - When a good mark is detected, the position of the **PosAxis** is compared to the **Target** position to calculate a correction.
 - The registration correction is only applied with master/slave move types.

3.2.14.1.3.3 Position and Option Inputs

Position Inputs

- The **Position** input is the position value the registration Axis position is reset to when a good registration mark is detected.
 - When a good mark occurs, the position of the **PosAxis** is compared to the **Target** position.
 - This comparison is used to calculate the amount of registration compensation to apply to the **CompAxis**.
 - Registration compensation is applied to the axis specified at the **CompAxis** input under these conditions:
 - If **CompAxis** is a master axis, the compensation is applied to the master offsets of all its slaves.
 - This shifts the master's position as seen by its slaves.
 - If **CompAxis** is executing a slave move (e.g., "MC_GearIn" (→ p. 710) or "MC_CamIn" (→ p. 693)), the compensation is applied directly to the axis.
 - The **PosTolerance** input is the distance, plus and minus, about the **Target** position used to determine if compensation is applied.
 - When a good mark occurs, the position of the **PosAxis** axis is checked to see if it lies in the window defined by **PosTolerance**.
 - If it is in the window, compensation is applied.
 - If it is outside the window, compensation is not applied even though a good mark was found.
 - If **PosAxis** and **CompAxis** are different axes, the **RatioNumerator** and **RatioDenominator** inputs define the conversion factor for calculating the compensation value.
 - This is needed because the amount of error between actual and target positions is determined by the **PosAxis's** position and the compensation is applied to the **CompAxis**.
 - The **RatioNumerator** should typically be the number of User Units of **CompAxis** motion for one registration cycle.
 - The **RatioDenominator** should typically be the number of User Units of **PosAxis** motion for one registration cycle.
 - If **PosAxis** and **CompAxis** are the same, **RatioNumerator** and **RatioDenominator** should be the same value, thus resulting in a 1:1 ratio.

Option Inputs

- The **Option** input defines various modes of operation for registration.
 - The first bit, 0001H, selects Absolute or Resetting.
 - This refers to the way the second mark and all subsequent marks are determined to be good marks.
 - With both registration schemes, the very first detected mark is the starting point.
 - With **Resetting** registration, when the next mark is detected, the position of that mark becomes the starting point for the next good mark detection calculation and so on.
 - The starting point is reset with each good or bad mark.
 - This allows the product to re-synchronize, if necessary, due to process issues (e.g., product shift) etc.
 - **Absolute** registration determines all good marks based on the very first mark.
 - The position of the second and each subsequent mark is compared to an integer multiple of **Distance** from the very first mark.
 - This method insures the product always registers to a known fixed distance.
 - The third bit, 0004H, must be 0 (zero).
 - Mark-to-machine registration requires time-based capture.

3.2.14.1.4.4 MC_MachRegist Options Table

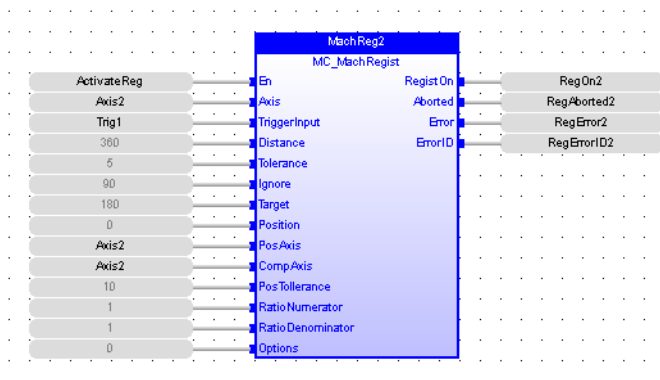
Hexadecimal	Decimal	Option	Description
0001 H	1	Absolute/Resetting	0 = Resetting, 1 = Absolute
0002 H	2	Reserved	0
0004 H	4	Time/position based capture	0 = time based capture, 1 = position based capture

Hexadecimal	Decimal	Option	Description
0008 H	8	Inhibit Reference on Good Mark	0 = Perform reference, 1 = inhibit reference When this bit is set, the Position function block argument is unused and the axis position is not changed when a registration mark is encountered.
0010H	16	Inhibit Master Compensation	0 = Perform Master Compensation, 1 = Inhibit Master Compensation
0020H	32	Inhibit Slave Compensation	0 = Perform Slave Compensation, 1 = Inhibit Slave Compensation.

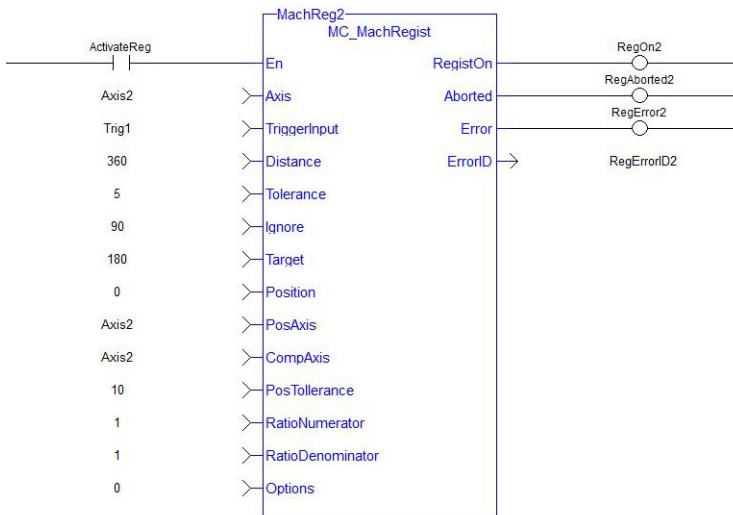
TIP

To use Capture Engine 1, modify the input PDOs that are used and add the Latch Position 1 parameter.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_MachRegist( ActivateReg, Axist2, Trig1, 360, 5, 90, 180, 0, Axis2,  
Axis2, 10, 1, 1, 0 );
```

See Also

- "MC_ReadParam" (→ p. 650)
- "MC_StopRegist" (→ p. 755)
- "MC_WriteParam" (→ p. 656)

3.2.14.2 MC_MarkRegist



Function Block - Enables Mark-to-Mark registration.

Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to enable registration.
Axis	AXIS_REF	The range of .AXIS_NUM is [1, 256]	N/A	N/A	Axis to apply registration to.
TriggerInput	TRIGGER_REF	No range	N/A	No default	Structure specifying the fast input. The structure elements are: DirectionINT Range = 1, 5. • 1 = Rising edge. • 2 = Falling edge. • 3 = N/A • 4 = Toggle between both, falling edge first. • 5 = Toggle between both, rising edge first. Valid values of InputID are: • 0 = Touch Probe 1 / Capture Engine 0 • 1 = Touch Probe 2 / Capture Engine 1 TrigIDINT Range = 0 to 256. • The axis number of the fast input. • 0 (zero) indicates this trigger axis is to be the same as the Axis input.

NOTE

TrigMode INT
(TriggerInput.TrigMode) is not presently supported by this function.

Input	Data Type	Range	Unit	Default	Description
Distance	LREAL	When converted to feedback units, the range is $[-2^{51}, 2^{51}-1]$.	User units	N/A	- This value must have the same sign as Ignore. - This is the expected distance between good marks. - Along with Tolerance and Ignore, this value is used to determine if the mark detected by the fast input is a good mark.
Tolerance	LREAL	When converted to feedback units, the range is $[0, 2^{51}-1]$.	User units	N/A	This value specifies the distance, plus or minus, about Distance to determine if the mark detected by the fast input is a good mark.
Ignore	LREAL	When converted to feedback units, the range is $[-2^{51}, 2^{51}-1]$.	User units	N/A	- This value must have the same sign as Distance. - This value specifies the distance after the previous good mark in which any detected marks are ignored.
Target	LREAL	When converted to feedback units, the range is $[-2^{51}, 2^{51}-1]$.	User units	N/A	- This value must have the same sign as Distance. - This is the target distance between good marks. - This distance is compared to the actual distance measured by the fast input to determine the amount of registration compensation to apply.
Position	LREAL	When converted to feedback units, the range is: $[-2^{51}, 2^{51}-1]$ if PosAxis' rollover value is zero. $[0, \text{PosAxis' Rollover Value}]$ if PosAxis' rollover value is non-zero (i.e., $\geq 0 < \text{PosAxis' Rollover Value}$).	User units	N/A	- The position the axis is set to when a good registration mark occurs. - If the Inhibit Reference on Good Mark option is specified for the Option argument, this argument is not used. - The position of the axis is not changed when a registration mark is encountered.

Input	Data Type	Range	Unit	Default	Description
Options	UINT	See the "MC_MachRegist Options Table" (→ p. 753).	N/A	N/A	- Each bit enables / disables an option. - The "MC_MachRegist Options Table" (→ p. 753) defines the bits. - Any bits not defined are reserved. - The third bit, 0004H, must be 0 (zero).

Outputs

Output	Data Type	Range	Unit	Description
RegistOn	BOOL	FALSE, TRUE	N/A	Indicates registration is activated.
Aborted	BOOL	FALSE, TRUE	N/A	Indicates registration has been terminated by "MC_StopRegist" (→ p. 755).
Error	BOOL	FALSE, TRUE	N/A	Indicates an invalid input was specified or the registration was terminated due to an error.
ErrorID	INT	No range	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

Remarks

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

TIP

- Is this the right function block to use?
 - See Deciding Which Function Blocks to Use for Registration and Registration Application Guide.
- It is used on any servo or digitizing axis and with any move type.
- It is used most frequently in master/slave applications.

Used with ...	Effect
Non-slave moves	Resets the axis position when a good mark is captured by the fast input.
Slave moves	In addition to resetting the axis position, applies a compensation offset to correct for the difference between the target mark-to-mark distance and the measured mark-to-mark distance. This provides the ability to compensate for product or process inconsistencies, provides a system that remains synchronized with no accumulated error, and maintains repeatable accuracy throughout the process.

3.2.14.2.1.1 Transition and Registration

Transition and Registration

- A positive transition of the **En** input starts registration.
 - The application may change the registration parameters while registration is active by changing the input values and causing another positive transition of the **En** input.
 - The function block then reads and applies the new values.
- The axis number at the **Axis** input identifies the axis of registration.
 - If **Axis** is a master axis for another axis's slave move, Master Registration is activated.
 - Master Registration calculates a compensation that is added to the master offset of its slaves.
 - This offset shifts the position of the master axis as seen by its slaves.
 - The compensation is not applied to the master axis, but to its slaves.
 - If **Axis** is a slave axis, Slave Registration is activated.
 - Slave Registration calculates a compensation that is added to the slave offset of the axis.
 - This compensation value is applied directly to the slave axis.

3.2.14.2.2.2 Distance, Tolerance, Ignore, Target Inputs

Distance, Tolerance, and Ignore Inputs

- The **Distance**, **Tolerance**, and **Ignore** inputs are used to determine whether or not the registration mark is good.
 - For a mark to be recognized as good, it must be outside of the **Ignore** distance and the correct **Distance** from the previous mark +/- the **Tolerance** window.
 - A mark is bad if it occurs outside of the "good tolerance band" and is not ignored.
 - Both good marks and bad marks are recognized as marks, ignored marks are not recognized.
 - If all marks are to be recognized as good marks, enter 0 (zero) at both **Distance** and **Tolerance**.
- The **Distance** value defines the distance between good marks.
 - In Clear Lane and Product registration the **Distance** input value is typically the same as the **Target** input value.
 - In Print registration, the **Distance** is typically not the same as **Target**.
- The **Tolerance** value is the distance, plus and minus, about **Distance**.
 - Marks detected in this window are good marks and registration occurs.
 - Marks detected outside this window and outside the **Ignore** band, are bad marks and registration does not occur.
 - This window should be large enough to allow for the worst case error in the distance between the previous mark and the current mark.
- The **Ignore** value defines the distance from the previous mark where all marks detected by the fast input are ignored.
 - This is crucial when registering products that do not have Clear Lane registration marks.

Target Input

- The **Target** input is the expected distance between good registration marks.
 - It is used to calculate how much registration compensation is applied when a registration mark is considered good.
 - In many applications, this is equivalent to the product length or the cycle length.
 - When a good mark is detected, the actual distance between the good mark and the previous mark is determined and compared to the **Target** distance to calculate a correction.
 - The registration correction is only applied with master/slave move types and always affects the slave axis.

3.2.14.2.3.3 Position and Option Inputs

Position and Option Inputs

- **Position Input**
 - The **Position** input is the position value the registration Axis position is reset to when a good registration mark is detected.

- **Option Inputs**

- The **Option** input defines various modes of operation for registration.
 - The first bit, 0001H, selects Absolute or Resetting.
 - This refers to the way the second mark and all subsequent marks are determined to be good marks.
 - With both registration schemes, the very first detected mark is the starting point.
 - With **Resetting** registration, when the next mark is detected, the position of that mark becomes the starting point for the next good mark detection calculation and so on.
 - The starting point is reset with each good or bad mark.
 - This allows the product to re-synchronize, if necessary, due to process issues (e.g., product shift) etc.
 - **Absolute** registration determines all good marks based on the very first mark.
 - The position of the second and each subsequent mark is compared to an integer multiple of **Distance** from the very first mark.
 - This method insures the product always registers to a known fixed distance.

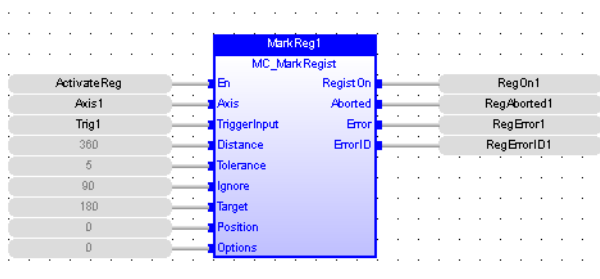
3.2.14.2.4 MC_MachRegist Options Table

Hexadecimal	Decimal	Option	Description
0001 H	1	Absolute/Resetting	0 = Resetting, 1 = Absolute
0002 H	2	Reserved	0
0004 H	4	Time/position based capture	0 = time based capture, 1 = position based capture
0008 H	8	Inhibit Reference on Good Mark	0 = Perform reference, 1 = inhibit reference When this bit is set, the Position function block argument is unused and the axis position is not changed when a registration mark is encountered.
0010H	16	Inhibit Master Compensation	0 = Perform Master Compensation, 1 = Inhibit Master Compensation
0020H	32	Inhibit Slave Compensation	0 = Perform Slave Compensation, 1 = Inhibit Slave Compensation.

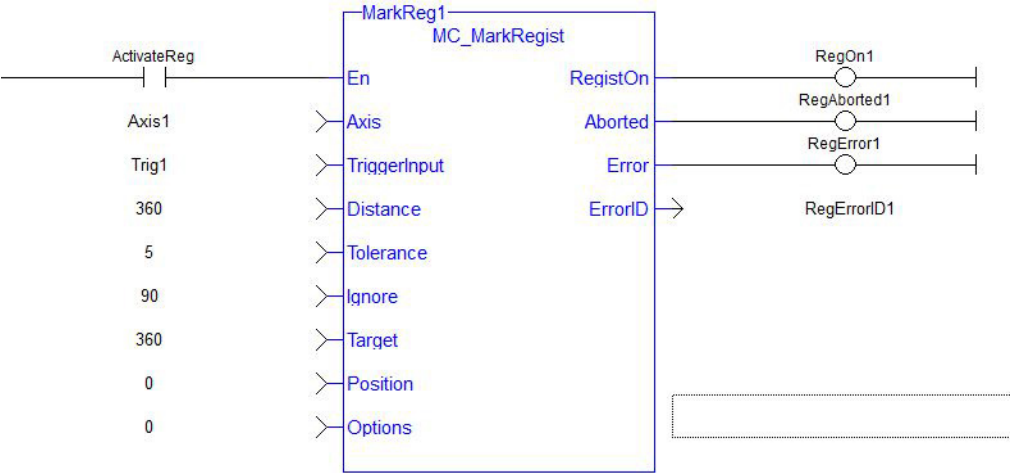
TIP

To use Capture Engine 1, modify the input PDOs that are used and add the Latch Position 1 parameter.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_MarkRegist( ActivateReg, Axist1, Trig1, 360, 5, 90, 360, 0, 0 );
```

See Also

- "MC_ReadParam" (→ p. 650)
- "MC_Stop" (→ p. 627)
- "MC_WriteParam" (→ p. 656)

3.2.14.3 MC_StopRegist

PLCopen 



Function Block - Turns off registration for the specified axis and disarms the specified fast input.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

TIP

To use Capture Engine 1, modify the input PDOs that are used and add the Latch Position 1 parameter.

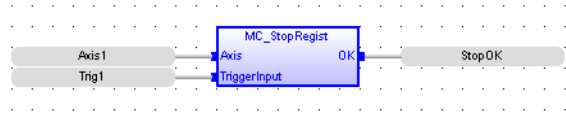
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	Enables execution.
Axis	AXIS_REF	1 to 256	N/A	N/A	Axis registration to turn off.
TriggerInput	TRIGGER_REF	See Description.	N/A	No default	Structure specifying the fast input. (to disarm) The structure elements are: DirectionINT Range = 1, 5. • 1 = Rising edge. • 2 = Falling edge. • 3 = N/A • 4 = Toggle between both, falling edge first. • 5 = Toggle between both, rising edge first. Valid values of InputID are: • 0 = Touch Probe 1 / Capture Engine 0 • 1 = Touch Probe 2 / Capture Engine 1 TrigidINT Range = 0 to 256. • Axis number of the fast input. • 0 (zero) indicates this trigger axis is to be the same as the Axis input. . NOTE TrigMode INT (TriggerInput.TrigMode) is not presently supported by this function.

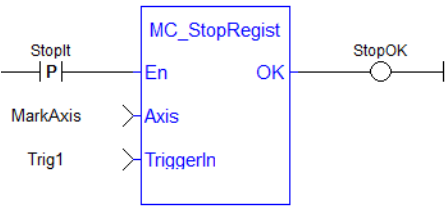
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Execution successful.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
StopOK := MC_StopRegist( Axis1, Trig1);
```

See Also

- "MC_MachRegist" (→ p. 741)
- "MC_MarkRegist" (→ p. 749)

3.3 Motion / Common

- "Coordinated Motion" (→ p. 757)
- "Info" (→ p. 757)
- "Motion" (→ p. 757)
- "Profiles" (→ p. 757)

3.3.1 Coordinated Motion

"Motion / Common - Coordinated Motion" (→ p. 758)

3.3.2 Info

Name	Description
MC_ErrorDescription	Converts the PLCopen error IDs into message strings.

3.3.3 Motion

Name	Description
MLMotionCycleTime	Returns the Motion Base Cycle time in seconds.
MLMotionInit	Initializes the motion library.
MLMotionRstErr	Clears motion engine errors, motion bus driver errors, and EtherCAT network errors.
MLMotionStart	Starts the motion engine, motion bus driver, clears the EtherCAT diagnostic registers of all nodes, and initializes EtherCAT network to operational mode.
MLMotionStatus	Returns the status of the motion engine 0 (zero).
MLMotionStop	Stops the motion bus driver, motion engine, and EtherCAT network operation, resulting in the EtherCAT network transitioning to the Init state.

3.3.4 Profiles

Name	Description
MLProfileBuild	Creates a CAM Profile from application data that can be executed by a CAM block in Pipe Network or PLCopen.
MLProfileCreate	Creates a new CAM Profile Object for use in a PLC Program or Pipe Network CAM block.
MLProfileInit	Initializes a previously created CAM Profile object for use in a PLC Program or Pipe Network CAM block.
MLProfileRelease	Removes a Profile so the ProfileID can be used by a different or new Profile.

3.3.5 Motion / Common - Coordinated Motion

These tables list the Coordinated Motion function blocks and function blocks:

- "Group Control" (→ p. 758)
- "Info" (→ p. 758)
- "Motion" (→ p. 759)
- "Reference" (→ p. 759)

3.3.5.1 Group Control

See "Coordinated Motion - Group Control" (→ p. 760) for more information.

Name	Description
MC_AddAxisToGrp	Adds an axis to an axes group.
MC_CreateAxesGrp	Create an axis group for coordinated motion.
MC_GrpDisable	Changes the state of a group to GroupDisabled.
MC_GrpEnable	Changes the state of a group from GroupDisabled to GroupStandby.
MC_GrpReadBoolPar	Reads a value from the specified Boolean group parameter.
MC_GrpReadParam	Reads a value from the specified group parameter.
MC_GrpReset	Makes the transition from the state GroupErrorStop to GroupStandby by resetting all internal group-related errors.
MC_GrpStop	Performs a controlled motion stop of all axes in the group.
MC_GrpWriteBoolPar	Writes a value to the specified Boolean group parameter.
MC_GrpWriteParam	Writes a value to the specified group parameter.
MC_InitAxesGrp	Initializes the kinematic limits for the axis group.
MC_RemAxisFromGrp	Removes an individual axis from an axis group.
MC_SetKinTra	Sets the kinematic transform between the Machine Coordinate System and the Axes Coordinate System.
MC_UngroupAllAxes	Removes all axes from an axes group.

3.3.5.2 Info

See "Coordinated Motion - Info" (→ p. 799) for more information.

Name	Description
MC_GrpReadActAcc	Reads the actual acceleration of the group and the axes in the group.
MC_GrpReadActPos	Reads the actual position of the axes in the group.
MC_GrpReadActVel	Reads the actual velocity of the group and the axes in the group.
MC_GrpReadCmdPos	Reads the command position of the axes in the group.

Name	Description
MC_GrpReadCmdVel	Reads the command velocity of the axes in the group and the path velocity.
MC_GrpReadError	Reads the Group ErrorID in state ERRORSTOP.
MC_GrpReadStatus	Returns the status of an axes group.

3.3.5.3 Motion

See "Coordinated Motion - Motion" (→ p. 821) for more information.

Name	Description
MC_AxisSetDefaults	Sets the default kinematic parameters for an axis.
MC_ErrorDescription	Converts the PLCopen error IDs into message strings.
MC_GrpHalt	Performs a controlled motion stop of all axes in the group.
MC_GrpSetOverride	Sets the velocity factor that is multiplied to the commanded velocity of all axes in the group.
MC_MoveCircAbs	Commands interpolated circular movement on an axes group to the specified absolute positions in the coordinate system as specified by the CoordSystem argument.
MC_MoveCircRel	Commands interpolated circular movement on an axes group to the specified relative positions in the coordinate system as specified by the CoordSystem argument.
MC_MoveDirAbs	Commands movement of an axes group to an absolute position regardless of path.
MC_MoveDirRel	Commands movement of an axes group to a relative position regardless of path.
MC_MoveLinAbs	Commands interpolated linear movement on an axes group to the specified absolute positions.
MC_MoveLinRel	Commands interpolated linear movement on an axes group to the specified relative positions.

3.3.5.4 Reference

See "Coordinated Motion - Reference" (→ p. 858) for more information.

Name	Description
MC_GrpSetPos	Sets the axis command position for all of the axes in an axes group to the positions specified in the <code>Position</code> input.

3.3.5.5 Coordinated Motion - Group Control

Name	Description
MC_AddAxisToGrp	Adds an axis to an axes group.
MC_CreateAxesGrp	Create an axis group for coordinated motion.
MC_GrpDisable	Changes the state of a group to GroupDisabled.
MC_GrpEnable	Changes the state of a group from GroupDisabled to GroupStandby.
MC_GrpReadBoolPar	Reads a value from the specified Boolean group parameter.
MC_GrpReadParam	Reads a value from the specified group parameter.
MC_GrpReset	Makes the transition from the state GroupErrorStop to GroupStandby by resetting all internal group-related errors.
MC_GrpStop	Performs a controlled motion stop of all axes in the group.
MC_GrpWriteBoolPar	Writes a value to the specified Boolean group parameter.
MC_GrpWriteParam	Writes a value to the specified group parameter.
MC_InitAxesGrp	Initializes the kinematic limits for the axis group.
MC_RemAxisFromGrp	Removes an individual axis from an axis group.
MC_SetKinTra	Sets the kinematic transform between the Machine Coordinate System and the Axes Coordinate System.
MC_UngroupAllAxes	Removes all axes from an axes group.

3.3.5.5.1 MC_AddAxisToGrp

PLCopen Pipe Network **Function Block** - Adds an axis to an axes group.**NOTE**

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

TIP

- An axes group cannot contain more than one instance of an axis.
- Two active groups cannot contain the same axis.
- An active group is one in any state other than GroupDisabled.
- This function or function block does not generate any motion.
- See Coordinated Motion, the top-level topic for Coordinated Motion.
- Both the axis and the axes group must be created prior to calling this function block.
 - See "MC_CreateAxesGrp" (→ p. 764) and Create a PLCopen Axis.
- The **IdentInGroup** input specifies the index of the axis in the group.
 - Axes do not need to be added in sequential order and gaps are acceptable.
 - Gaps are ignored when the group is used.
- The group must be in either the **GroupStandby** or **GroupDisabled** state when the axis is added.
 - The state of the group can be read with "MC_GrpReadStatus" (→ p. 817).
 - This implies that the group cannot be moving when the axis is added.

Inputs

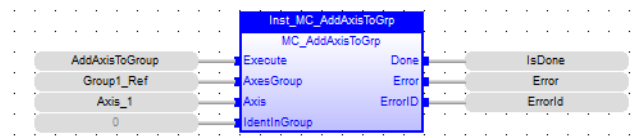
Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, the axis is added to the group.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	Reference to an axes group.
Axis	AXIS_REF	No range	N/A	No default	• Reference to the axis to be added. • An axes group cannot contain more than one instance of an axis.

Input	Data Type	Range	Unit	Default	Description
IdentInGroup	UINT	0, MaxNumberOfAxes - 1	N/A	No default	The zero-based index of the axis in the group. - The axis slot in the group cannot be occupied by another axis. - The index must be less than the maximum number of axes the group can contain. - MaxNumberOfAxes is a property of the axes group and is set when the group is created. - To remove an axis from a group see "MC_RemAxisFromGrp" (→ p. 786).

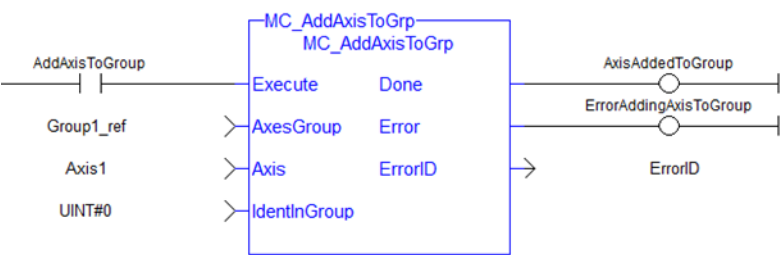
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
(*MC_AddAxisToGrp ST example *)  
Inst_MC_AddAxisToGrp (AddAxisToGrp, Group1_ref, Axis_1, 0);
```

See Also

- "MC_CreateAxesGrp" (→ p. 764)
- "MC_GrpReadStatus" (→ p. 817)
- "MC_RemAxisFromGrp" (→ p. 786)
- "MC_UngroupAllAxes" (→ p. 791)
- "MC_ErrorDescription" (→ p. 863)

3.3.5.5.2 MC_CreateAxesGrp

PLCopen

Pipe Network

 **Function Block** - Create an axis group for coordinated motion.

- See Coordinated Motion, the top-level topic for Coordinated Motion.
- Calls to this function block are automatically generated when the application is compiled.
 - Users should not manually call this function block.
- More than one axes group may be created and be active at the same time but each axis can only be a part of one group at a time.
- If an axis needs to be in more than one group, it can be removed from one and then added to another group.
 - This is done using "MC_RemAxisFromGrp" (→ p. 786) and "MC_AddAxisToGrp" (→ p. 761).

ⓘIMPORTANT

MC_CreateAxesGrp must be called between "MLMotionInit" (→ p. 867) and "MLMotionStart" (→ p. 872).

Example: Valid Setup

```
AxesGroup1:  Axis0, Axis1, Axis2
AxesGroup2:  Axis3, Axis4
```

Example: Invalid Setup

```
AxesGroup1:  Axis0, Axis1, Axis2
AxesGroup2:  Axis2, Axis3, Axis4
```

The invalid setup is not allowed because Axis2 would be a part of two axes groups at the same time.

Inputs

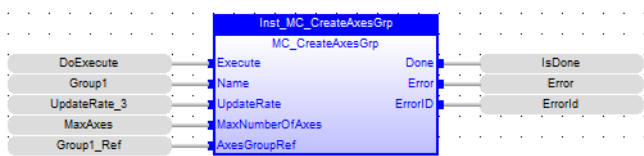
Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, this function block creates a coordinated motion axes group.
Name	STRING	1 to 64 characters	N/A	No default	Axes Group Name The string length is limited to 64 characters for optimal controller performance.

Input	Data Type	Range	Unit	Default	Description	
UpdateRate	Enumerated	3 to 9	N/A	No default	Update rate of the axes group.	
					Value	Update Rate Enumeration
					3	CM_UPDATERATE_125us
					4	CM_UPDATERATE_250us
					5	CM_UPDATERATE_500us
					6	CM_UPDATERATE_1ms
					7	CM_UPDATERATE_2ms
					8	CM_UPDATERATE_4ms
9	CM_UPDATERATE_8ms					
MaxNumberOfAxes	UINT	2 to 256	N/A	No default	The maximum number of axes that can be controlled by the group.	
AxesGroupRef	AXES_GROUP_REF	No range	N/A	No default	The axes group reference variable to be initialized with a reference to the new axes group.	

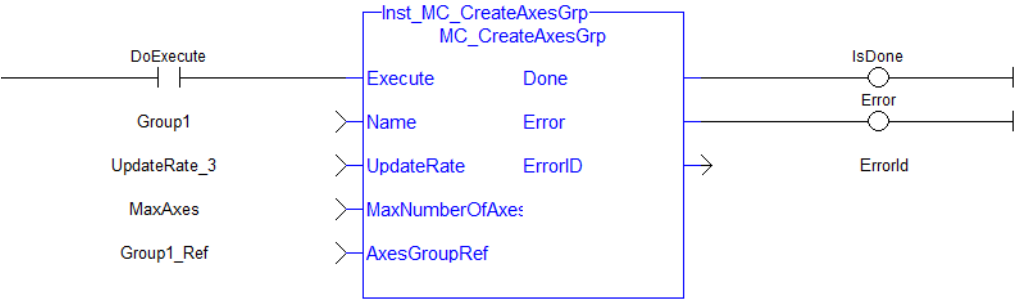
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_CreateAxesGrp( DoExecute, 'Group1', UpdateRate_3, MaxAxes, Group1_
Ref) ;
```

See Also

- "MC_InitAxesGrp" (→ p. 784)
- "MC_ErrorDescription" (→ p. 863)

S

3.3.5.5.3 MC_GrpDisable

PLCopen ✓

Pipe Network ✓

**Function Block** - Changes the state of a group to GroupDisabled.**NOTE**

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.
- See Coordinated Motion, the top-level topic for Coordinated Motion.
- If the group is already in **GroupDisabled**, then **MC_GrpDisable** does nothing.
- This function block can be issued in the group states: (GroupDisabled, GroupStandby, or GroupErrorStop).
 - See Group State Diagrams.

CAUTION

MC_GrpDisable fails if the group is in any state other than GroupStandby or GroupDisabled.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to disable the axis group.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axis group to be disabled.

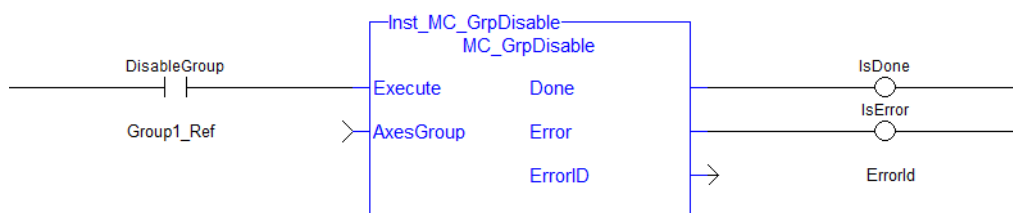
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	No range	N/A	If TRUE, the command completed successfully.
Error	BOOL	No range	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* Inst_MC_GrpDisableST example *)  
Inst_MC_GrpDisable( DisableGroup, Group1_Ref );
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_GrpEnable" (→ p. 769)

3.3.5.5.4 MC_GrpEnable

PLCopen 

Pipe Network 



Function Block - Changes the state of a group from GroupDisabled to GroupStandby.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.
- See Coordinated Motion, the top-level topic for Coordinated Motion.
- The group must be in **GroupStandby** to perform motion.
 - If the group is already in **GroupStandby**, then **MC_GrpEnable** does nothing.
- MC_GrpEnable fails under these conditions:
 - It contains no axes.
 - The group is not in GroupDisabled or GroupStandby.
 - One or more axes in the group are in another group that is not in GroupDisabled.
 - See Group State Diagrams.

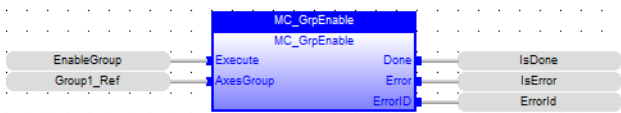
Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to enable the axis group.
AxesGroup	AXES_GROUP_REF	N/A	N/A	No default	The axis group to be enabled.

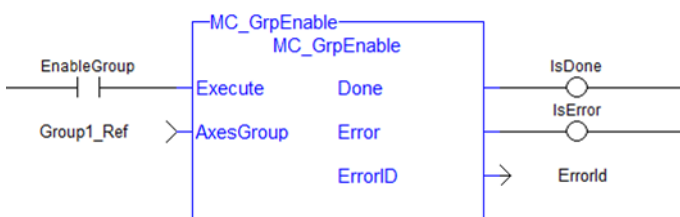
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* Inst_MC_GrpEnableST example *)  
Inst_MC_GrpEnable( EnableGroup, Group1_Ref );
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_GrpDisable" (→ p. 767)

3.3.5.5.5 MC_GrpReadBoolPar

PLCopen Pipe Network **Function Block** - Reads a value from the specified Boolean group parameter.**NOTE**

- This function or function block returns cached data.
- See Program a Multi-Core Controller.
- See Coordinated Motion, the top-level topic for Coordinated Motion.
- See Recovery of the System State After an Axis Error.
- MC_GrpReadBoolPar(Axesgroup_Ref GroupID, Uint BoolID) where BoolID can be one of these two currently defined Booleans:
 - IGNORE_AXIS_ESTOP:
 - ID = 1000
 - The value read is either TRUE or False as set by the MC_GrpWriteBoolPar function block.
 - AXIS_ESTOP_ACTIVE:
 - ID = 1001
 - This read-only parameter is asserted TRUE whenever an axis in the group is experiencing an Axis Estop Error.
 - When there are no Axis Estop Errors present on the axes in a group, this parameter is set to FALSE.

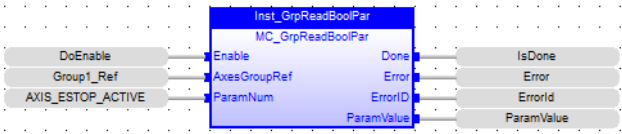
Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	If TRUE, then request to read a value from the specified Boolean group parameter.
AxesGroupRef	AXES_GROUP_REF	N/A	N/A	No default	The axis group the Boolean parameter value is read from.
ParamNum	UINT	See Axes Group Parameters.	UINT	No default	ParamNum can be one of these defined Booleans: • IGNORE_AXIS_ESTOP: • AXIS_ESTOP_ACTIVE:

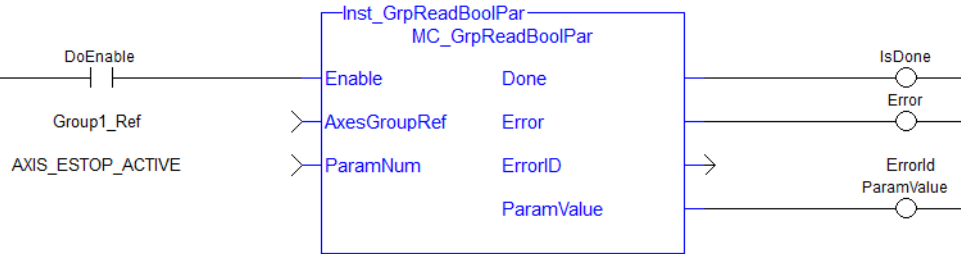
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See PLCopen Function Block ErrorIDs.
ParamValue	BOOL	FALSE, TRUE	N/A	The value of the parameter.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_GrpReadBoolPar( DoEnable, Group1_Ref, AXIS_ESTOP_ACTIVE );
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_GrpWriteBoolPar" (→ p. 780)

3.3.5.5.6 MC_GrpReadParam

PLCopen ✓

Pipe Network ✓

**Function Block** - Reads a value from the specified group parameter.**NOTE**

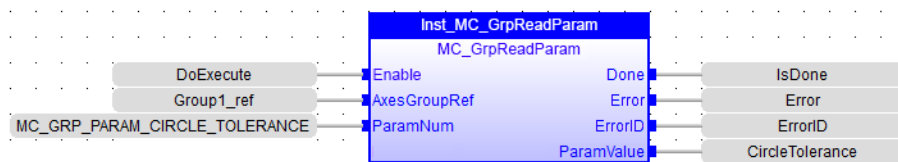
- This function or function block returns cached data.
- See Program a Multi-Core Controller.
- See Coordinated Motion, the top-level topic for Coordinated Motion.

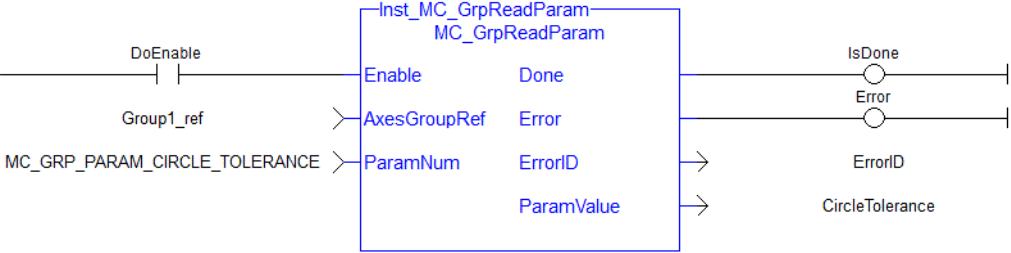
Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	If TRUE, then request to read a value from the specified Boolean group parameter.
AxesGroupRef	AXES_GROUP_REF	No range	N/A	No default	The axis group the Boolean parameter value is read from.
ParamNum	LREAL	See Axes Group Parameters.	LREAL	No default	Only one parameter is supported: MC_GRP_PARAM_CIRCLE_TOLERANCE:

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See PLCopen Function Block ErrorIDs.
ParamValue	UINT	0 to 65535	N/A	The value of the group parameter.

FBD Language Example**FFLD Language Example**



IL Language Example

Not available.

ST Language Example

```
Inst_MC_GrpReadParam(DoEnable, Group1_ref, MC_GRP_PARAM_CIRCLE_TOLERANCE);  
  
CircleTolerance := Inst_MC_GrpReadParam.ParamValue;
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_GrpWriteParam" (→ p. 782)

3.3.5.5.7 MC_GrpReset

PLCopen ✓

Pipe Network ✓



Function Block - Makes the transition from the state **GroupErrorStop** to **GroupStandby** by resetting all internal group-related errors.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.
- See Coordinated Motion, the top-level topic for Coordinated Motion.
- This function or function block does not generate any motion.
- This function block:
 - Does not affect the output of the FB instances.
 - Resets axis errors and drive faults for each axis in the group.

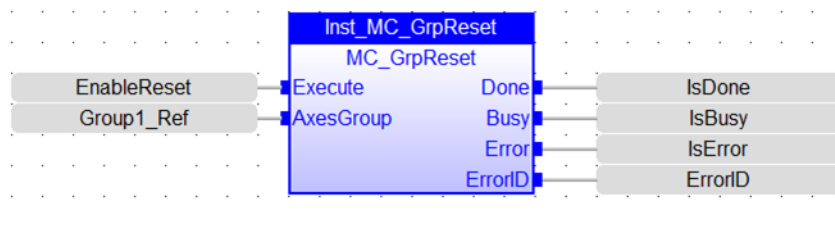
Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, resets group-related errors and all of the axes in the group.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axes group the axes are reset in.

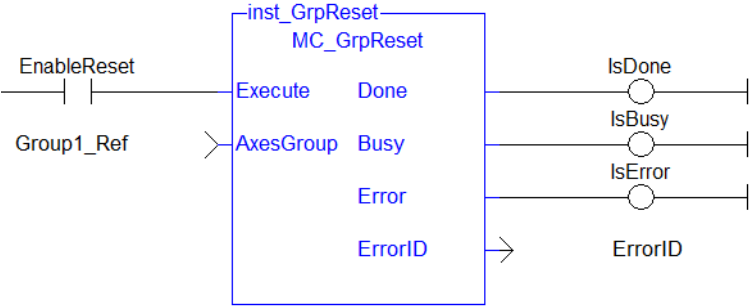
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_GrpReset(EnableReset, Group1_Ref);
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_GrpReadError" (→ p. 815)
- "MC_GrpReadStatus" (→ p. 817)

3.3.5.5.8 MC_GrpStop

PLCopen Pipe Network **Function Block** - Performs a controlled motion stop of all axes in the group.**CAUTION****MC_GrpStop** can not be aborted.**NOTE**

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.
- When the path velocity reaches 0 (zero):
 - Any queued moves are flushed from the buffer.
 - The Done output is set.
- When the Done output is TRUE and the application has cleared, the Execute input state transitions to GroupStandby.
- **MC_GrpStop** does NOT prevent:
 - A single axis from executing.
 - Other Coordinated Motion moves from executing once MC_GrpStop has completed.
- See What MC_GrpStop Does.
- See Differences between MC_GrpHalt and MC_GrpStop.
- See Coordinated Motion, the top-level topic for Coordinated Motion.

Inputs

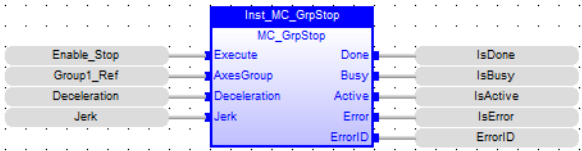
Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, the command to stop all of the axes in the group is initiated.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axes group the axes are stopped in.
Deceleration	LREAL	No range	User units / sec ²	No default	The path deceleration rate for all axes in the group. • If Deceleration is 0 or negative, the deceleration parameter of the active move is used. • If Deceleration is less than the deceleration of the active move, the deceleration parameter of the active move is used. • This guarantees the stop action completes before the endpoint of the move is reached. • See Limitations on Acceleration and Jerk.

Input	Data Type	Range	Unit	Default	Description
Jerk	LREAL	No range	User units / sec ³	No default	The path jerk for all axes in the group. - If Jerk is negative, the jerk parameter of the active move is used. - If Jerk is zero, then zero jerk is used. - If Jerk is less than the jerk of the active move, the jerk parameter of the active move is used. - This guarantees the stop action completes before the endpoint of the move is reached. - See Limitations on Acceleration and Jerk.

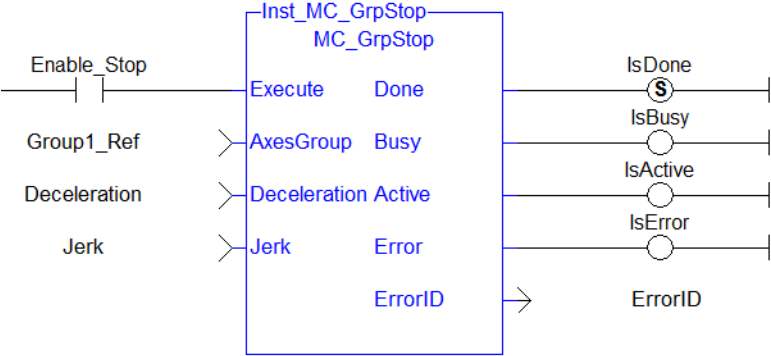
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	TRUE from the moment the EXECUTE input is TRUE until the stop is complete.
Active	BOOL	FALSE, TRUE	N/A	If TRUE, the stop is still executing.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_GrpStop ( EnableStop, Group1_Ref, Deceleration, Jerk );
```

3.3.6 Related Functions

- "MC_ErrorDescription" (→ p. 863)
- "MC_GrpHalt" (→ p. 825)

3.3.6.0.1 MC_GrpWriteBoolPar

PLCopen ✓

Pipe Network ✓

**Function Block** - Writes a value to the specified Boolean group parameter.**NOTE**

- This function or function block returns cached data.
- See Program a Multi-Core Controller.
- See Coordinated Motion, the top-level topic for Coordinated Motion.
- See Recovery of the System State After an Axis Error.
- IGNORE_AXIS_ESTOP
 - BoolID = 1000.
 - Value can be either TRUE or FALSE.
- Setting this Boolean Parameter to TRUE results in the Coordinated Motion Engine NOT stopping all axes in a group when one of them is stopped due to an Axis Estop Error.
 - Only the axis experiencing the error stops when this Parameter is set to TRUE.
- When this parameter is FALSE (Default), all axes in a group are stopped and the power off request is asserted for each axis.

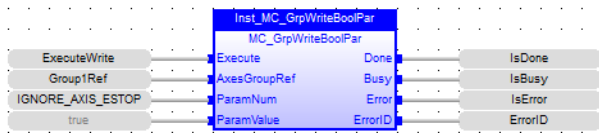
Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to write a value to the specified Boolean group parameter.
AxesGroupRef	AXES_GROUP_REF	No range	N/A	No default	The axis group the Boolean parameter value is written to.
ParamNum	UINT	No range	N/A	No default	The ID number of the Boolean parameter that is to be written. • IGNORE_AXIS_ESTOP • BoolID = 1000
ParamValue	BOOL	FALSE, TRUE	N/A	No default	FALSE, TRUE

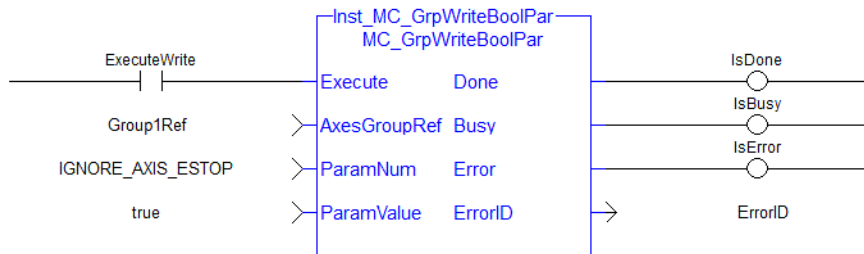
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_GrpWriteBoolPar( ExecuteWrite, Group1Ref, IGNORE_AXIS_ESTOP, true );
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_GrpReadBoolPar" (→ p. 771)
- "MC_ErrorDescription" (→ p. 863)

3.3.6.0.2 MC_GrpWriteParam

PLCopen ✓

Pipe Network ✓

 **Function Block** - Writes a value to the specified group parameter.

NOTE

- This function or function block returns cached data.
 - See Program a Multi-Core Controller.
-
- See Coordinated Motion, the top-level topic for Coordinated Motion.

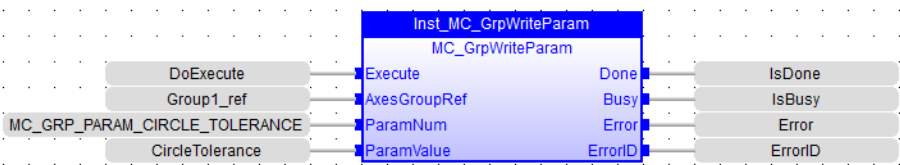
Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to write a value to the specified Boolean group parameter.
AxesGroupRef	AXES_GROUP_REF	No range	N/A	No default	The axis group the Boolean parameter value is written to.
ParamNum	LREAL	See Axes Group Parameters.	LREAL	No default	Only one parameter is supported: • MC_GRP_PARAM_CIRCLE_TOLERANCE
ParamValue	UINT	Parameter dependent.	Parameter dependent.	No default	The new value for the group parameter.

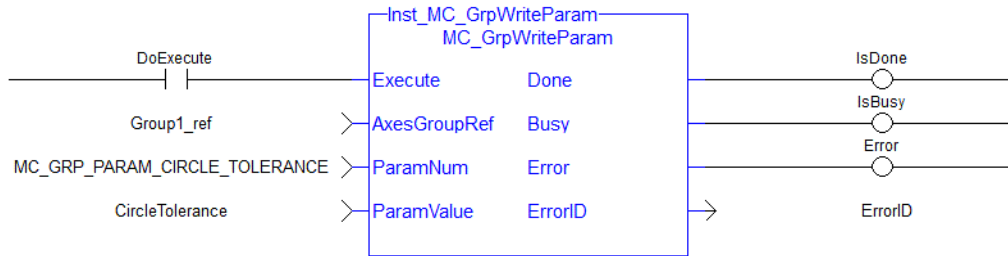
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_GrpWriteParam( DoExecute, Group1_ref, MC_GRP_PARAM_CIRCLE_TOLERANCE,
CircleTolerance );
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_GrpReadParam" (→ p. 773)

3.3.6.0.3 MC_InitAxesGrp

PLCopen Pipe Network **Function Block** - Initializes the kinematic limits for the axis group.**NOTE**

The function block returns an error if the group state is not GroupStandby or GroupDisabled.

- See Coordinated Motion, the top-level topic for Coordinated Motion.
- During a move, the motion engine verifies that the limits are not exceeded.

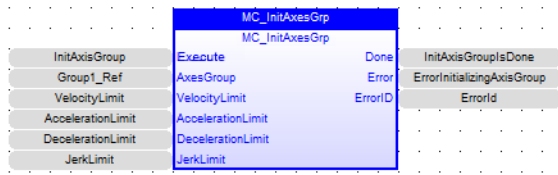
Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, this function block initializes the axis group.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axes group to be initialized.
VelocityLimit	LREAL	Acceleration > (Velocity / 20) and Deceleration > (Velocity / 20) 0 < Velocity < (20 * Acceleration) and 0 < Velocity < (20 * Deceleration)	User units / sec	No default	• Velocity limit. • See Limitations on Acceleration and Jerk.
AccelerationLimit	LREAL	Acceleration > (Velocity / 20)	User units / sec ²	No default	• Acceleration limit. • See Limitations on Acceleration and Jerk.
DecelerationLimit	LREAL	Deceleration > (Velocity / 20)	User units / sec ²	No default	• Deceleration limit. • See Limitations on Acceleration and Jerk.
JerkLimit	LREAL	Jerk > (Acceleration / 2) and Jerk > (Deceleration / 2)	User units / sec ³	No default	• Jerk limit. • See Limitations on Acceleration and Jerk.

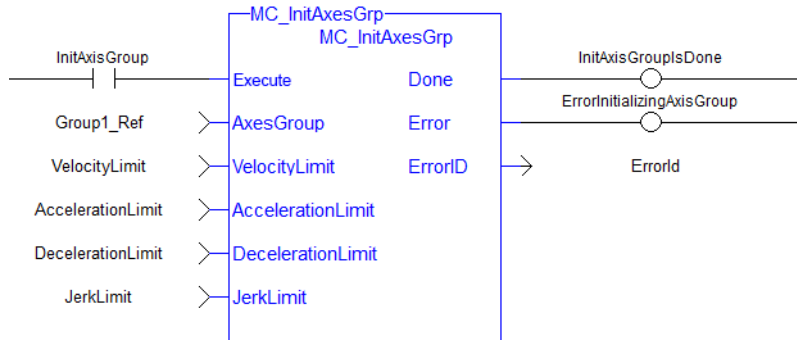
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* Inst_MC_InitAxesGrpST example *)Inst_MC_InitAxesGrp( initAxesGrp, grp,
velLim, accelLim, decelLim, jerkLim );
```

See Also

- "MC_CreateAxesGrp" (→ p. 764)
- "MC_ErrorDescription" (→ p. 863)

3.3.6.0.4 MC_RemAxisFromGrp

PLCopen



Pipe Network

**Function Block** - Removes an individual axis from an axis group.**NOTE**

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.
- See Coordinated Motion, the top-level topic for Coordinated Motion.
- This function or function block does not generate any motion.
- This function block can be issued in the group states: (GroupDisabled, GroupStandby, or GroupErrorStop).
 - The group's state changes to GroupDisabled if the axis removed is the last valid axis in the group.
- **MC_RemAxisFromGrp** fails if the group is in any state other than GroupStandby or GroupDisabled.
 - See Group State Diagrams.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to remove an axis from the group.
AxesGroupRef	AXES_GROUP_REF	No range	N/A	No default	The axis group from which the axis is removed.
IdentInGroup	UINT	0, MaxNumberOfAxes - 1	N/A	No default	The zero-based index of the axis in the group. • The axis index in the group must contain a valid axis. • The index must be less than the maximum number of axes the group can contain. • <code>MaxNumberOfAxes</code> is a property of the axes group and is set when the group is created.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* Inst_MC_InitAxisGrpST example *)
Inst_MC_RemAxisFromGrp( ExecuteRemAxisFromGrp, Group1_Ref, AxisId );
```

See Also

- "MC_AddAxisToGrp" (→ p. 761)
- "MC_ErrorDescription" (→ p. 863)
- "MC_UngroupAllAxes" (→ p. 791)

3.3.6.0.5 MC_SetKinTra



 **Function Block** - Sets the kinematic transform between the Machine Coordinate System and the Axes Coordinate System.

- It is useful for robotics, allowing the application to command motion in Cartesian coordinates for the robotic system.
- After **MC_SetKinTra(...)** is called, the controller automatically calculates the inverse kinematics for the robot axes, converting the robot path motion into the individual robot joint axis trajectories.
- Several transform types are available for common robotic systems and are configurable with the "MC_KIN_REF Structure" (→ p. 793).
- The parameters in the MC_KIN_REF structure define the specific robot geometry.
- See "MC_KIN_REF Structure" (→ p. 793) for a description of the structure.

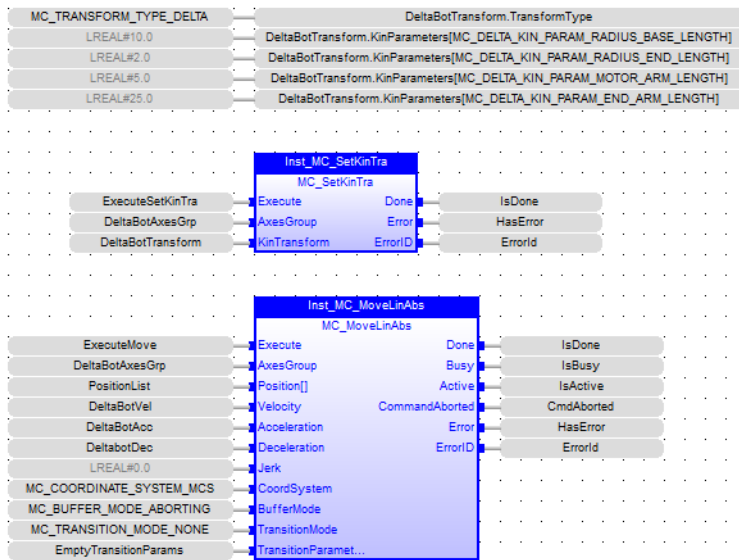
Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to load the kinematic transform.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axes group that receives the axis trajectory values from the kinematic transform.
KinTransform	MC_KIN_REF	No range	N/A	No default	Kinematic robotic transform defined by the "MC_KIN_REF Structure" (→ p. 793).

Outputs

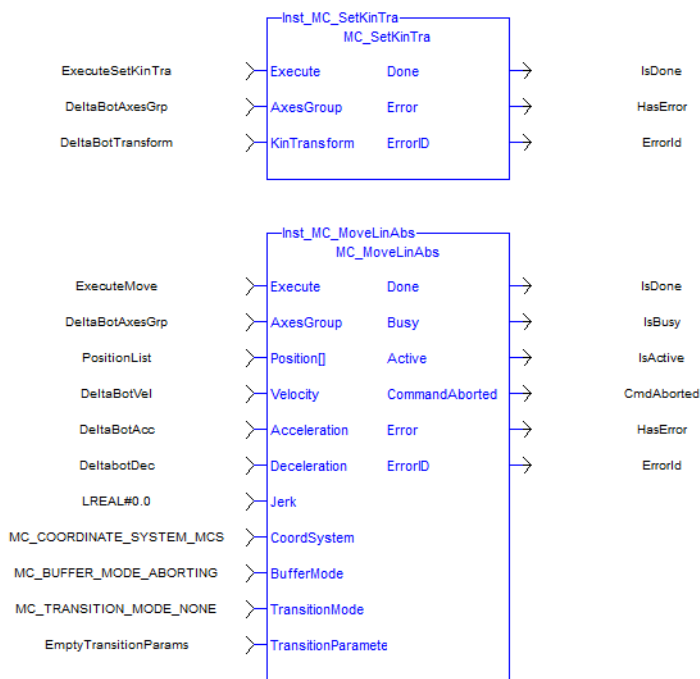
Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example

Network #1



End of Module

IL Language Example

Not available.

ST Language Example

```
// MC_SetKinTra ST Example

// DeltaBotTransform is of type MC_KIN_REF
```

```

DeltaBotTransform.TransformType := MC_TRANSFORM_TYPE_DELTA;
DeltaBotTransform.KinParameters[MC_DELTA_KIN_PARAM_RADIUS_BASE_LENGTH] :=
10.0;
DeltaBotTransform.KinParameters[MC_DELTA_KIN_PARAM_RADIUS_END_LENGTH] :=
2.0;
DeltaBotTransform.KinParameters[MC_DELTA_KIN_PARAM_MOTOR_ARM_LENGTH] :=
5.0;
DeltaBotTransform.KinParameters[MC_DELTA_KIN_PARAM_END_ARM_LENGTH] :=
25.0;

Inst_MC_SetKinTra(True, DeltaBotAxesGrp, DeltaBotTransform);

// ...

Inst_MC_MoveLinAbs(True, DeltaBotAxesGrp, PositionList, DeltaBotVel,
DeltaBotAcc, DeltaBotDec, LREAL#0.0, MC_COORDINATE_SYSTEM_MCS, MC_BUFFER_
MODE_ABORTING, MC_TRANSITION_MODE_NONE, EmptyTransitionParams);

```

3.3.6.0.6 MC_UngroupAllAxes

PLCopen Pipe Network **Function Block** - Removes all axes from an axes group.**NOTE**

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.
- See Coordinated Motion, the top-level topic for Coordinated Motion.
- This function or function block does not generate any motion.
- This function block can be issued in the group states: (GroupDisabled, GroupStandby, or GroupErrorStop).
 - The axes group state is changed to GroupDisabled upon successful completion.
- **MC_UngroupAllAxes** fails if the group is in any state other than GroupStandby or GroupDisabled.
 - See Group State Diagrams.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to remove all axes in the axes group.
AxesGroupRef	AXES_GROUP_REF	No range	N/A	No default	The axis group from which to remove all axes.

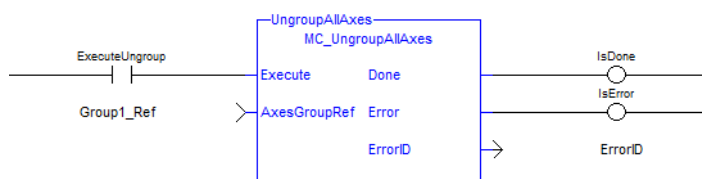
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_UngroupAllAxes( ExecuteUngroup, Group1_Ref );
```

See Also

- "MC_AddAxisToGrp" (→ p. 761)
- "MC_ErrorDescription" (→ p. 863)
- "MC_RemAxisFromGrp" (→ p. 786)

3.3.6.0.7 MC_KIN_REF Structure

The MC_KIN_REF structure defines the robotic system transform type and its parameters.

The parameters are specific to each transform type.

3.3.6.0.8.1 General MC_KIN_REF Structure

Member	Data Type	Description	Related Function Block
TransformType	UINT	A number that identifies the specific robotic system transform. The #defines for the transform types are: "MC_TRANSFORM_TYPE_GANTRY" (→ p. 793) "MC_TRANSFORM_TYPE_GANTRY_WITH_SKEW" (→ p. 793) "MC_TRANSFORM_TYPE_HBOT" (→ p. 794) "MC_TRANSFORM_TYPE_TBOT" (→ p. 794) "MC_TRANSFORM_TYPE_DELTA (*Experimental*)" (→ p. 795) "MC_TRANSFORM_TYPE_SCARA_ELBOW_POS (*Experimental*)" (→ p. 797) "MC_TRANSFORM_TYPE_SCARA_ELBOW_NEG (*Experimental*)" (→ p. 797)	"MC_SetKinTra" (→ p. 788)
KinParameters [0 - 31]	LREAL	- Array to define the robotic system and its kinematic transform. - A maximum of 32 parameters are allowed. - The parameter count (0 to 32) and the definition of each parameter is determined by the specific TransformType.	"MC_SetKinTra" (→ p. 788)

3.3.6.0.9.2 Robotic Transform Parameters

- These parameters must be specified for all ACS axes in the AxisGroup.
- There are two parameters for each ACS axis.

3.3.7 MC_TRANSFORM_TYPE_GANTRY

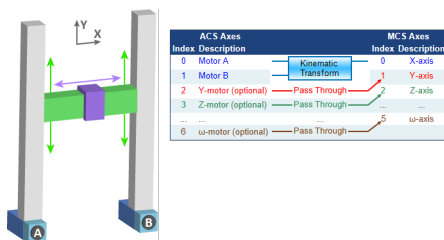


Figure 4-91: MC_TRANSFORM_TYPE_GANTRY

Parameter Count	Parameter #	Description	Range
1	MC_GANTRY_KIN_PARAM_REVERSE_X	Reverse X-axis.	0 = Do Not Reverse. 1 = Reverse.

3.3.8 MC_TRANSFORM_TYPE_GANTRY_WITH_SKEW

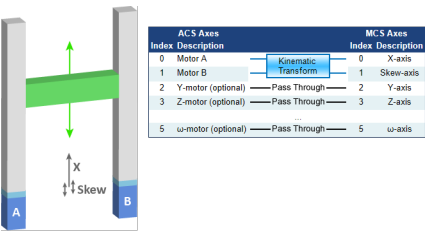


Figure 4-92: MC_TRANSFORM_TYPE_GANTRY_WITH_SKEW

Parameter Count	Parameter #	Description	Range
3	MC_GANTRY_WITH_SKEW_KIN_PARAM_REVERSE_X	Reverse X-axis.	0 = Do Not Reverse. 1 = Reverse.
	MC_GANTRY_WITH_SKEW_KIN_PARAM_SKEW_SCALE	Skew to linear unit ratio.	Positive value.
	MC_GANTRY_WITH_SKEW_KIN_PARAM_REVERSE_SKEW	Reverse Skew-axis.	0 = Do Not Reverse. 1 = Reverse.

3.3.9 MC_TRANSFORM_TYPE_HBOT

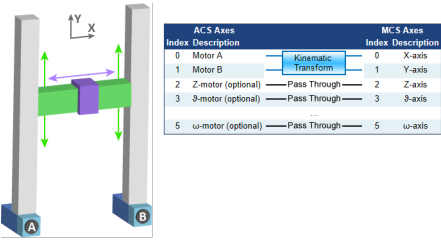


Figure 4-93: MC_TRANSFORM_TYPE_HBOT

Parameter Count	Parameter #	Description	Range
3	MC_HBOT_KIN_PARAM_LIN_TO_ROT_RATIO	Linear to rotational unit ratio. Example: cm/degree.	Positive value.
	MC_HBOT_KIN_PARAM_REVERSE_X	Reverse X-axis.	0 = Do Not Reverse. 1 = Reverse.
	MC_HBOT_KIN_PARAM_REVERSE_Y	Reverse Y-axis.	0 = Do Not Reverse. 1 = Reverse.

3.3.10 MC_TRANSFORM_TYPE_TBOT

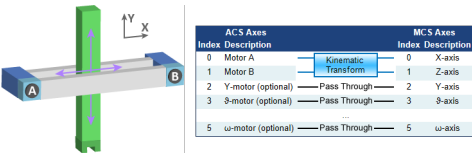


Figure 4-94: MC_TRANSFORM_TYPE_TBOT

Parameter Count	Parameter #	Description	Range
3	MC_TBOT_KIN_PARAM_LIN_TO_ROT_RATIO	Linear to rotational unit ratio. Example: cm/degree.	Positive value.
	MC_TBOT_KIN_PARAM_REVERSE_X	Reverse X-axis.	• 0 = Do Not Reverse. • 1 = Reverse.
	MC_TBOT_KIN_PARAM_REVERSE_Z	Reverse Y-axis.	• 0 = Do Not Reverse. • 1 = Reverse.

3.3.11 MC_TRANSFORM_TYPE_DELTA (*Experimental*)

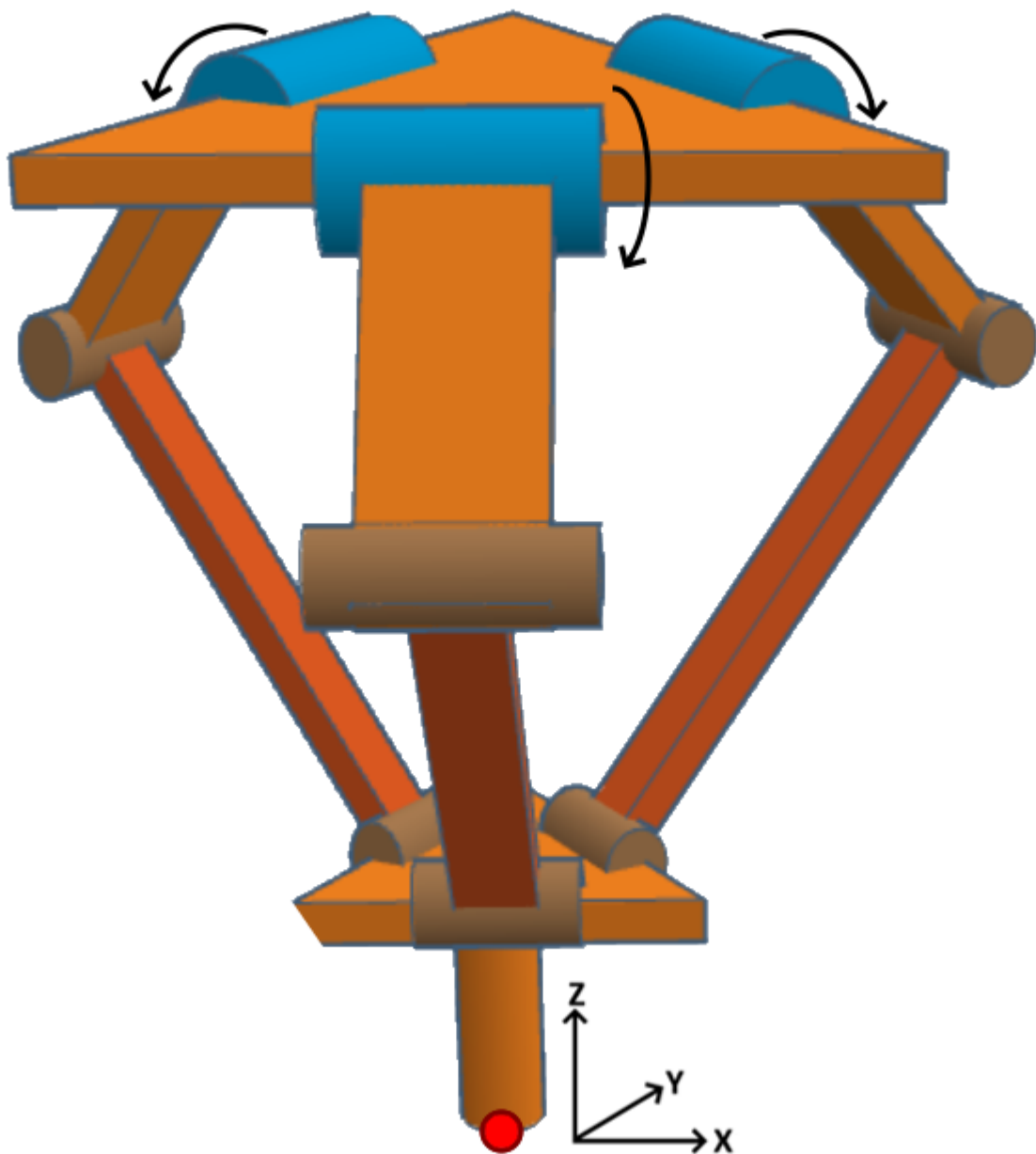


Figure 4-95: MC_TRANSFORM_TYPE_DELTA (*Experimental*)

Parameter Count	Parameter #	Description	Range
4	MC_DELTA_KIN_PARAM_RADIUS_BASE_LENGTH	RadiusBaseLength	Positive, non-zero value.
	MC_DELTA_KIN_PARAM_RADIUS_END_LENGTH	RadiusEndLength	Positive, non-zero value.
	MC_DELTA_KIN_PARAM_MOTOR_ARM_LENGTH	MotorArmLength	Positive, non-zero value.
	MC_DELTA_KIN_PARAM_END_ARM_LENGTH	EndArmLength	Positive, non-zero value.

3.3.12 MC_TRANSFORM_TYPE_SCARA_ELBOW_POS (*Experimental*)

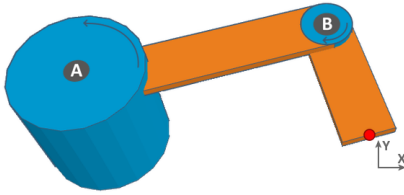


Figure 4-96: MC_TRANSFORM_TYPE_SCARA_ELBOW_POS no Wrist

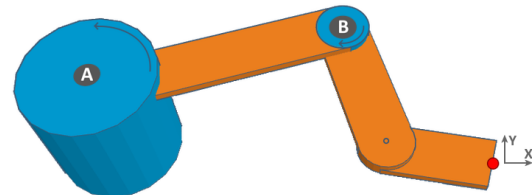


Figure 4-97: MC_TRANSFORM_TYPE_SCARA_ELBOW_POS with Wrist

Parameter Count	Parameter #	Description	Range
2 or 3 The parameter count depends on the axis number of the AxisGroup.	MC_SCARA_KIN_PARAM_UPPER_ARM_LENGTH	RobotUpperArmLength	Positive, non-zero value.
2 AxesGroup: 2 Parameters. (No Wrist) 3 AxesGroup: 3 Parameters. (with Wrist)	MC_SCARA_KIN_PARAM_LOWER_ARM_LENGTH	RobotLowerArmLength	Positive, non-zero value.
	MC_SCARA_KIN_PARAM_WRIST_LENGTH	RobotWristLength Available for SCARA with wrist only.	Positive, non-zero value.

3.3.13 MC_TRANSFORM_TYPE_SCARA_ELBOW_NEG (*Experimental*)

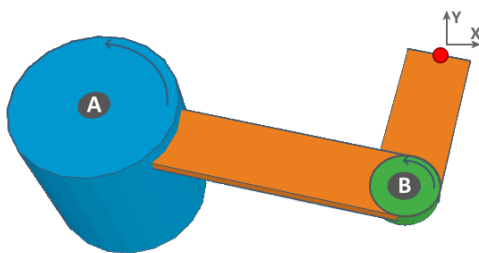


Figure 4-98: MC_TRANSFORM_TYPE_SCARA_ELBOW_NEG no Wrist

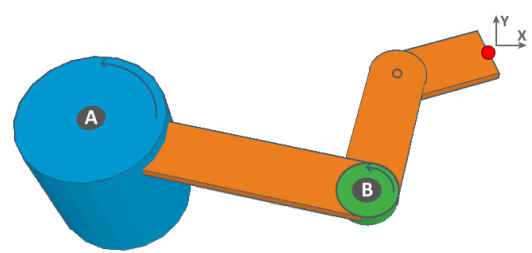


Figure 4-99: MC_TRANSFORM_TYPE_SCARA_ELBOW_NEG with Wrist

Parameter Count	Parameter #	Description	Range
2 or 3 The parameter count depends on the axis number of the AxisGroup. 2 AxesGroup: 2 Parameters. (No Wrist) 3 AxesGroup: 3 Parameters. (with Wrist)	MC_SCARA_KIN_PARAM_UPPER_ARM_LENGTH	RobotUpperArmLength	Positive, non-zero value.
	MC_SCARA_KIN_PARAM_LOWER_ARM_LENGTH	RobotLowerArmLength	Positive, non-zero value.
	MC_SCARA_KIN_PARAM_WRIST_LENGTH	RobotWristLength Available for SCARA with wrist only.	Positive, non-zero value.

3.3.13.1 Coordinated Motion - Info

Name	Description
MC_GrpReadActAcc	Reads the actual acceleration of the group and the axes in the group.
MC_GrpReadActPos	Reads the actual position of the axes in the group.
MC_GrpReadActVel	Reads the actual velocity of the group and the axes in the group.
MC_GrpReadCmdPos	Reads the command position of the axes in the group.
MC_GrpReadCmdVel	Reads the command velocity of the axes in the group and the path velocity.
MC_GrpReadError	Reads the Group ErrorID in state ERRORSTOP.
MC_GrpReadStatus	Returns the status of an axes group.

3.3.13.1.1 MC_GrpReadActAcc

PLCopen ✓

Pipe Network ✓

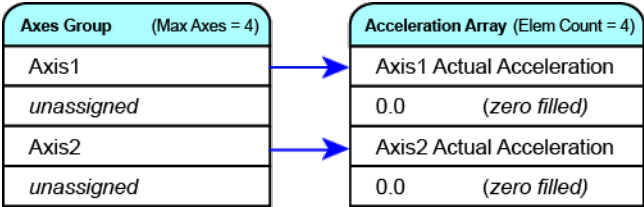
 **Function Block** - Reads the actual acceleration of the group and the axes in the group.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.
- See Coordinated Motion, the top-level topic for Coordinated Motion.
- The **MC_GrpReadActAcc** function block fills the array specified by the Acceleration argument with the actual acceleration of the system in the coordinate system specified by the `CoordSystem` argument.
 - The measured path acceleration is calculated and reported via the `PathAcceleration` output.
- This function or function block does not generate any motion.

NOTE

- The actual acceleration is smoothed over the last 10 samples. This reduces the error in acceleration estimation but introduces a small amount of phase delay in the reported accelerations.
- Only the ACS coordinate system is supported. See Coordinate Systems.
- There is a one-to-one correspondence between the axes in `AxesGroup` and the acceleration values in the Acceleration array.
- If an index in `AxesGroup` is unassigned, the acceleration value for that array element in the Acceleration array is 0 (zero).
- If the element does contain an axis, the Acceleration element is filled with the current actual acceleration for that axis.
 - This is an example of how this works:



Inputs

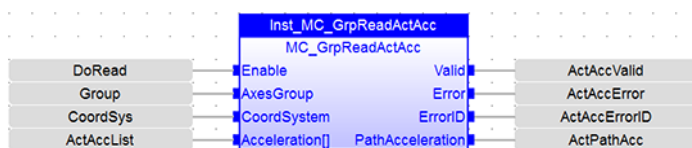
Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	If TRUE, this function block reads the current actual acceleration of the group and the axes in the group.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axes group the actual acceleration is read from.

Input	Data Type	Range	Unit	Default	Description								
CoordSystem	SINT	Enumerated	N/A	No default	- The coordinate system used when reading the actual acceleration. - One of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>CS_ACS</td><td>Axes Coordinate System</td></tr><tr><td>CS_MCS</td><td>Machine Coordinate System</td></tr><tr><td>CS_PCS</td><td>Product Coordinate System</td></tr></table>	Value	Description	CS_ACS	Axes Coordinate System	CS_MCS	Machine Coordinate System	CS_PCS	Product Coordinate System
Value	Description												
CS_ACS	Axes Coordinate System												
CS_MCS	Machine Coordinate System												
CS_PCS	Product Coordinate System												
Acceleration	LREAL[]	Dimension: The number of axes mapped in the AxesGroup. No range	User units / sec ²	No default	An array where the acceleration data is written. - The length of the array must equal the maximum number of axes allowed in the group. - The maximum number of axes is an argument to the "MC_CreateAxesGrp" (→ p. 764) function block used to create axes groups.								

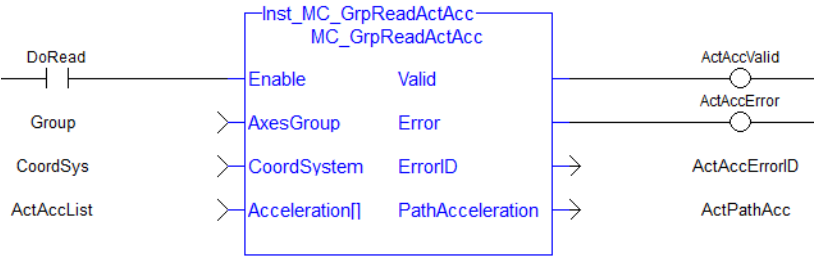
Outputs

Output	Data Type	Range	Unit	Description
Valid	BOOL	FALSE, TRUE	N/A	If TRUE, the accelerations have been read without error.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.
PathAcceleration	LREAL	No range	User units / sec ²	- The current measured path acceleration of the group. - This is measured by taking the square root of the sum of the squared accelerations of each axis.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_GrpReadActAcc( DoRead, Group, CoordSys, AccList );
```

See Also

- "MC_GrpReadActPos" (→ p. 803)
- "MC_GrpReadActVel" (→ p. 806)
- "MC_GrpReadCmdPos" (→ p. 809)
- "MC_GrpReadCmdVel" (→ p. 812)

3.3.13.1.2 MC_GrpReadActPos

PLCopen ✓

Pipe Network ✓

**Function Block** - Reads the actual position of the axes in the group.**NOTE**

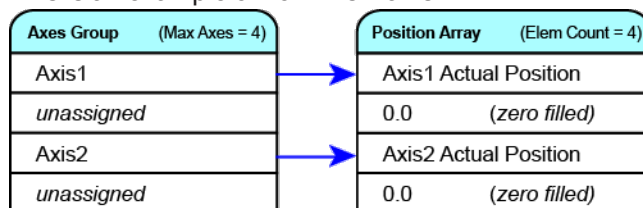
- This function or function block returns cached data.
- See Program a Multi-Core Controller.

- See Coordinated Motion, the top-level topic for Coordinated Motion.
- The **MC_GrpReadActPos** function block fills the array specified by the Position argument with the actual position of the system in the coordinate system specified by the `CoordSystem` argument.
- This function or function block does not generate any motion.

NOTE

Only the ACS coordinate system is supported.
See Coordinate Systems.

- There is a one-to-one correspondence between the axes in AxesGroup and the position values in the Position Array.
- If an index in AxesGroup is unassigned, the position value for that array element in the Position Array is 0 (zero).
- If the element does contain an axis, the position element is filled with the current actual position for that axis.
- This is an example of how this works:

**Inputs**

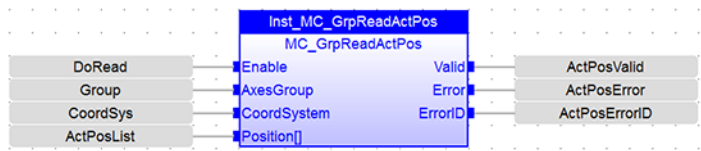
Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	If TRUE, this function block reads the current actual position of the axes in the group.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axes group the actual position is read from.

Input	Data Type	Range	Unit	Default	Description								
CoordSystem	SINT	Enumerated	N/A	No default	- The coordinate system used when reading the actual position. - One of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>CS_ACS</td><td>Axes Coordinate System</td></tr><tr><td>CS_MCS</td><td>Machine Coordinate System</td></tr><tr><td>CS_PCS</td><td>Product Coordinate System</td></tr></table>	Value	Description	CS_ACS	Axes Coordinate System	CS_MCS	Machine Coordinate System	CS_PCS	Product Coordinate System
Value	Description												
CS_ACS	Axes Coordinate System												
CS_MCS	Machine Coordinate System												
CS_PCS	Product Coordinate System												
Position	LREAL[]	Dimension: The number of axes mapped in the AxesGroup. No range	User units	No default	An array where the position data is written. - The length of the array must equal the maximum number of axes allowed in the group. - The maximum number of axes is an argument to the "MC_CreateAxesGrp" (→ p. 764) function block used to create axes groups.								

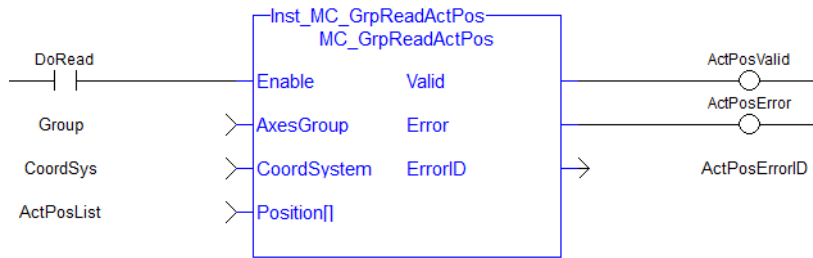
Outputs

Output	Data Type	Range	Unit	Description
Valid	BOOL	FALSE, TRUE	N/A	If TRUE, the positions have been read without error.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_GrpReadActPos( DoRead, Group, CoordSys, PosList );
```

See Also

- "MC_GrpReadActAcc" (→ p. 800)
- "MC_GrpReadActVel" (→ p. 806)
- "MC_GrpReadCmdPos" (→ p. 809)
- "MC_GrpReadCmdVel" (→ p. 812)

3.3.13.1.3 MC_GrpReadActVel

PLCopen ✓

Pipe Network ✓

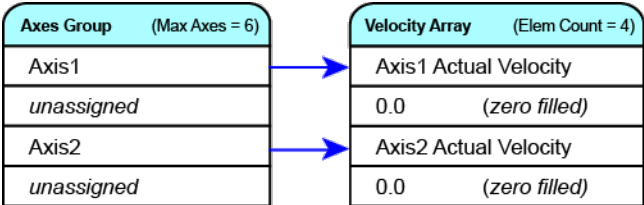
 **Function Block** - Reads the actual velocity of the group and the axes in the group.

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.
- See Coordinated Motion, the top-level topic for Coordinated Motion.
- This function or function block does not generate any motion.
- The **MC_GrpReadActVel** function block fills the array specified by the **Velocity** argument with the actual velocity of the system in the coordinate system specified by the **CoordSystem** argument.
 - The measured path velocity is also calculated and reported by the **PathVelocity** output.

NOTE

- The actual velocity is averaged over the last 10 samples.
 - This reduces the error in velocity estimation but introduces a small amount of phase delay in the reported velocities.
- Only the ACS coordinate system is supported.
See Coordinate Systems.
- There is a one-to-one correspondence between the axes in the Axes Group and the velocity values in the Velocity Array.
- If a index in the Axes Group is unassigned, the velocity value for that array element in the Velocity array is 0 (zero).
- If the element does contain an axis, the velocity value is filled with the current actual velocity for that axis.
 - This is an example of how this works:



Inputs

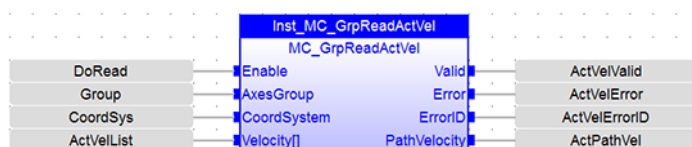
Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	If TRUE, this function block reads the current actual velocity of the group and the axes in the group.
AxesGroup	AXES_GROUP_REF	N/A	N/A	No default	The axes group the actual velocity is read from.

Input	Data Type	Range	Unit	Default	Description								
CoordSystem	SINT	Enumerated	N/A	No default	- The coordinate system used when reading the actual velocity. - One of these enumeration values: <table><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>CS_ACS</td><td>Axes Coordinate System</td></tr><tr><td>CS_MCS</td><td>Machine Coordinate System</td></tr><tr><td>CS_PCS</td><td>Product Coordinate System</td></tr></tbody></table>	Value	Description	CS_ACS	Axes Coordinate System	CS_MCS	Machine Coordinate System	CS_PCS	Product Coordinate System
Value	Description												
CS_ACS	Axes Coordinate System												
CS_MCS	Machine Coordinate System												
CS_PCS	Product Coordinate System												
Velocity	LREAL[]	Dimension: The number of axes mapped in the AxesGroup. No range	User units / sec	No default	An array where the velocity data is written. - The length of the array must equal the maximum number of axes allowed in the group. - The maximum number of axes is an argument to the "MC_CreateAxesGrp" (→ p. 764) function block used to create axes groups.								

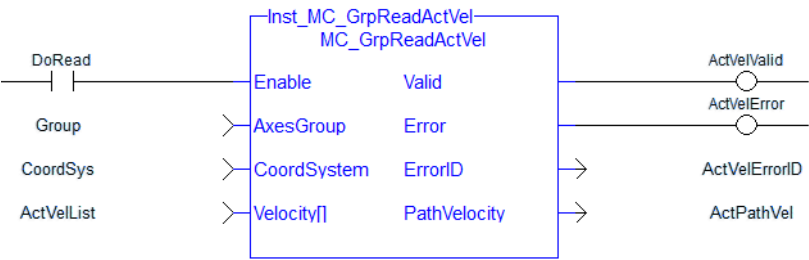
Outputs

Output	Data Type	Range	Unit	Description
Valid	BOOL	FALSE, TRUE	N/A	If TRUE, the velocities were read without error.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.
PathVelocity	LREAL	No range	User units / sec	- The current measured path velocity of the group. - This is measured by taking the square root of the sum of the squared velocities of each axis.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_GrpReadActVel (DoRead, Group, CoordSys, VelList);
```

See Also

- "MC_GrpReadActAcc" (→ p. 800)
- "MC_GrpReadActPos" (→ p. 803)
- "MC_GrpReadCmdPos" (→ p. 809)
- "MC_GrpReadCmdVel" (→ p. 812)

3.3.13.1.4 MC_GrpReadCmdPos

PLCopen ✓

Pipe Network ✓

**Function Block** - Reads the command position of the axes in the group.**NOTE**

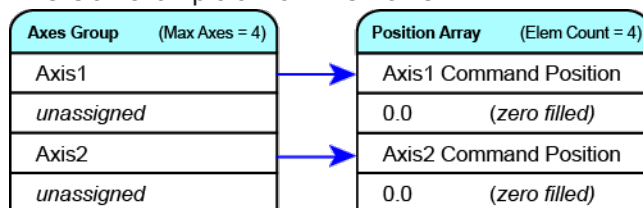
- This function or function block returns cached data.
- See Program a Multi-Core Controller.

- See Coordinated Motion, the top-level topic for Coordinated Motion.
- This function or function block does not generate any motion.
- The **MC_GrpReadCmdPos** function block fills the array (specified by the `Position` argument) with the commanded position of the coordinate system specified by the `CoordSystem` argument.

NOTE

Only the ACS coordinate system is supported.
See Coordinate Systems.

- There is a one-to-one correspondence between the axes in the Axes Group and the position values in the Position Array.
- Each element in the Position Array corresponds to the axis element in the Axis Group array.
- If an index in the Axes Group is unassigned, the position value for that array element in the Position Array is 0 (zero).
- If the element does contain an axis, the position value is filled with the current actual position for that axis.
 - This is an example of how this works:

**Inputs**

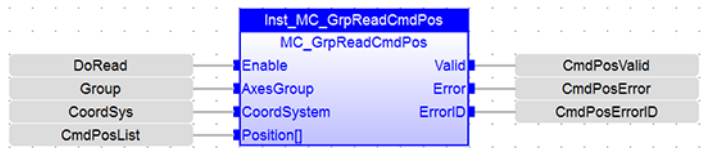
Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	If TRUE, this function block reads the current commanded position of the axes in the group.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axes group the commanded position is read from.

Input	Data Type	Range	Unit	Default	Description								
CoordSystem	SINT	Enumerated	N/A	No default	- The coordinate system used when reading the commanded position. - One of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>CS_ACS</td><td>Axes Coordinate System</td></tr><tr><td>CS_MCS</td><td>Machine Coordinate System</td></tr><tr><td>CS_PCS</td><td>Product Coordinate System</td></tr></table>	Value	Description	CS_ACS	Axes Coordinate System	CS_MCS	Machine Coordinate System	CS_PCS	Product Coordinate System
Value	Description												
CS_ACS	Axes Coordinate System												
CS_MCS	Machine Coordinate System												
CS_PCS	Product Coordinate System												
Position	LREAL[]	Dimension: The number of axes mapped in the AxesGroup. No range	User units	No default	An array where the position data is written. - The length of the array must equal the maximum number of axes allowed in the group. - The maximum number of axes is an argument to the "MC_CreateAxesGrp" (→ p. 764) function block used to create axes groups.								

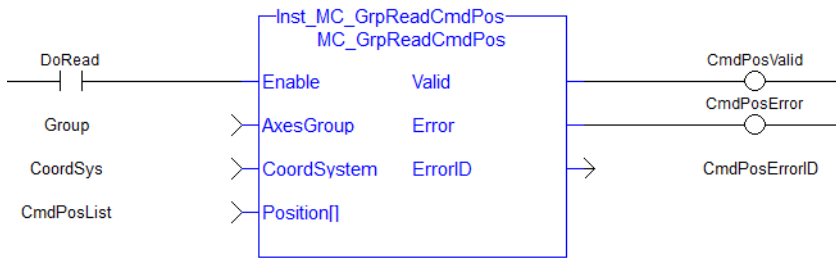
Outputs

Output	Data Type	Range	Unit	Description
Valid	BOOL	FALSE, TRUE	N/A	If TRUE, the positions have been read without error.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(*MC_GrpReadCmdPos ST example *)
Inst_MC_GrpReadCmdPos (DoRead, Group, CoordSys, PosList );
```

See Also

- "MC_GrpReadActAcc" (→ p. 800)
- "MC_GrpReadActPos" (→ p. 803)
- "MC_GrpReadActVel" (→ p. 806)
- "MC_GrpReadCmdVel" (→ p. 812)

3.3.13.1.5 MC_GrpReadCmdVel

PLCopen ✓

Pipe Network ✓

 **Function Block** - Reads the command velocity of the axes in the group and the path velocity.

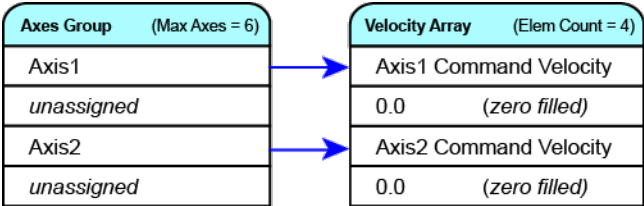
NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.
- See Coordinated Motion, the top-level topic for Coordinated Motion.
- This function or function block does not generate any motion.
- The **MC_GrpReadCmdVel** function block fills the array specified by the `Velocity` argument with the commanded velocity for the coordinate system, which is specified by the `CoordSystem` argument.
 - The path velocity is reported by the `PathVelocity` output.

NOTE

Only the ACS coordinate system is supported.
See Coordinate Systems.

- There is a one-to-one correspondence between the axes in the Axes Group and the velocity values in the Velocity Array.
- If a index in the Axes Group is unassigned, the velocity value for that array element in the Velocity array is 0 (zero).
- If the element does contain an axis, the velocity value is filled with the current actual velocity for that axis.
 - This is an example of how this works:



Inputs

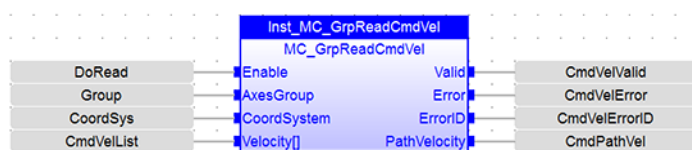
Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	If TRUE, this function block reads the current commanded velocity of the group and the axes in the group.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axes group the commanded velocity is read from.

Input	Data Type	Range	Unit	Default	Description								
CoordSystem	SINT	Enumerated	N/A	No default	- The coordinate system used when reading the commanded velocity. - One of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>CS_ACS</td><td>Axes Coordinate System</td></tr><tr><td>CS_MCS</td><td>Machine Coordinate System</td></tr><tr><td>CS_PCS</td><td>Product Coordinate System</td></tr></table>	Value	Description	CS_ACS	Axes Coordinate System	CS_MCS	Machine Coordinate System	CS_PCS	Product Coordinate System
Value	Description												
CS_ACS	Axes Coordinate System												
CS_MCS	Machine Coordinate System												
CS_PCS	Product Coordinate System												
Velocity	LREAL[]	Dimension: The number of axes mapped in the AxesGroup. No range	User units / sec	No default	An array where the velocity data is written. - The length of the array must equal the maximum number of axes allowed in the group. - The maximum number of axes is an argument to the "MC_CreateAxesGrp" (→ p. 764) function block used to create axes groups.								

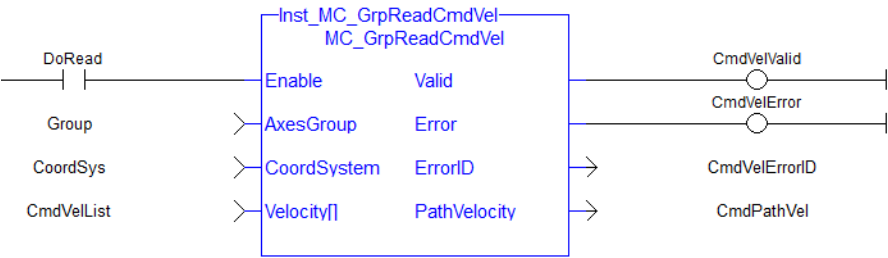
Outputs

Output	Data Type	Range	Unit	Description
Valid	BOOL	FALSE, TRUE	N/A	If TRUE, the velocities were read without error.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.
PathVelocity	LREAL	No range	User units / sec	- The current commanded path velocity of the group. - This is measured by taking the square root of the sum of the squared velocities of each axis.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(*MC_GrpReadCmdVel ST example *)
Inst_MC_GrpReadCmdVel (DoRead, Group, CoordSys, VelList );
```

See Also

- "MC_GrpReadActAcc" (→ p. 800)
- "MC_GrpReadActPos" (→ p. 803)
- "MC_GrpReadActVel" (→ p. 806)
- "MC_GrpReadCmdPos" (→ p. 809)

3.3.13.1.6 MC_GrpReadError

PLCopen ✓

Pipe Network ✓

**Function** - Reads the Group ErrorID in state ERRORSTOP.**NOTE**

- This function or function block returns cached data.
- See Program a Multi-Core Controller.
- See Coordinated Motion, the top-level topic for Coordinated Motion.
- This function or function block does not generate any motion.

Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	Enables execution.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axes group the GroupErrorID is read from.

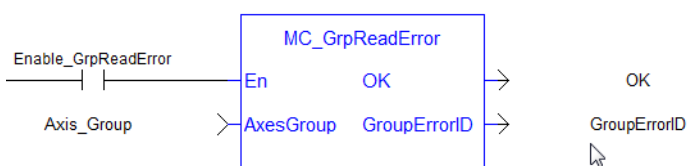
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Indicates the function executed successfully.
GroupErrorID	INT	Enumerated	N/A	• Shows the Error ID for the given Axis Group. • See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Read a group error number
GroupErrorID:= MC_GrpReadError( Axis_Group );
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_GrpReset" (→ p. 775)

3.3.13.1.7 MC_GrpReadStatus

PLCopen



Pipe Network

**Function Block** - Returns the status of an axes group.**NOTE**

- This function or function block returns cached data.
- See Program a Multi-Core Controller.
- See Coordinated Motion, the top-level topic for Coordinated Motion.
- MC_GrpReadStatus returns the status of an axes group.
 - This function block does not cause any motion.
- See Group State Diagrams.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	If TRUE, the axes group status is read.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axes group the status is read from.

Outputs

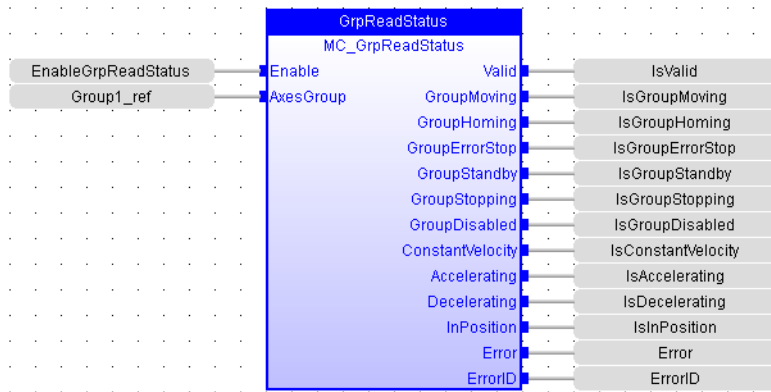
Output	Data Type	Range	Unit	Description
Valid	BOOL	FALSE, TRUE	N/A	TRUE of valid outputs are available.
GroupMoving ¹	BOOL	FALSE, TRUE	N/A	• The axes group is in the Moving state. • This indicates the group is enabled and currently executing a coordinated motion command.
GroupHoming ¹	BOOL	FALSE, TRUE	N/A	Not supported.
GroupErrorStop ¹	BOOL	FALSE, TRUE	N/A	The axes group is in the ErrorStop state due to an axis error or group error. • The group cannot accept coordinated motion commands. • The execution of MC_GrpReset is required to change the group's state from ErrorStop to Standby.
GroupStandby ¹	BOOL	FALSE, TRUE	N/A	The axes group is in the Standby state. • The group: • Is enabled and all its axes are enabled. • Is not currently executing a coordinated motion command. • The axes group is ready to accept coordinated motion commands.

Output	Data Type	Range	Unit	Description
GroupStopping¹	BOOL	FALSE, TRUE	N/A	The axes group is in the Stopping state due the execution of MC_GrpStop. - The axes group is enabled but cannot accept coordinated motion commands while in the Stopping state. - The axes group: - Remains in the Stopping state while MC_GrpStop is executing. - Will remain in the Stopping state while MC_GrpStop's Execute input is held high.
GroupDisabled¹	BOOL	FALSE, TRUE	N/A	The axis group is in the Disabled state and cannot accept coordinated motion commands.
ConstantVelocity	BOOL	FALSE, TRUE	N/A	TRUE if the commanded path velocity is the same between the current scan of the application program and the previous scan. - ConstantVelocity is always TRUE for Direct moves. - The commanded path velocity of Direct moves is always 0 (zero).
Accelerating	BOOL	FALSE, TRUE	N/A	TRUE if the commanded path velocity is accelerating between the current scan of the application program and the previous scan.
Decelerating	BOOL	FALSE, TRUE	N/A	TRUE if the commanded path velocity is decelerating between the current scan of the application program and the previous scan.
InPosition	BOOL	FALSE, TRUE	N/A	TRUE indicates that the axes group is "in position". These must be TRUE for the axes group to be in position: - The axes group is enabled. - There are no moves in the group's queue. - The servo loop is closed for each axis in the group. - There are no moves in the individual axis queue for each axis in the group. - The command delta is zero for each axis in the group. - The actual position is within the In-Position Bandwidth of the command position for each axis in the group.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error identifier if the Error output is set to TRUE. - See PLCopen Function Block ErrorIDs.

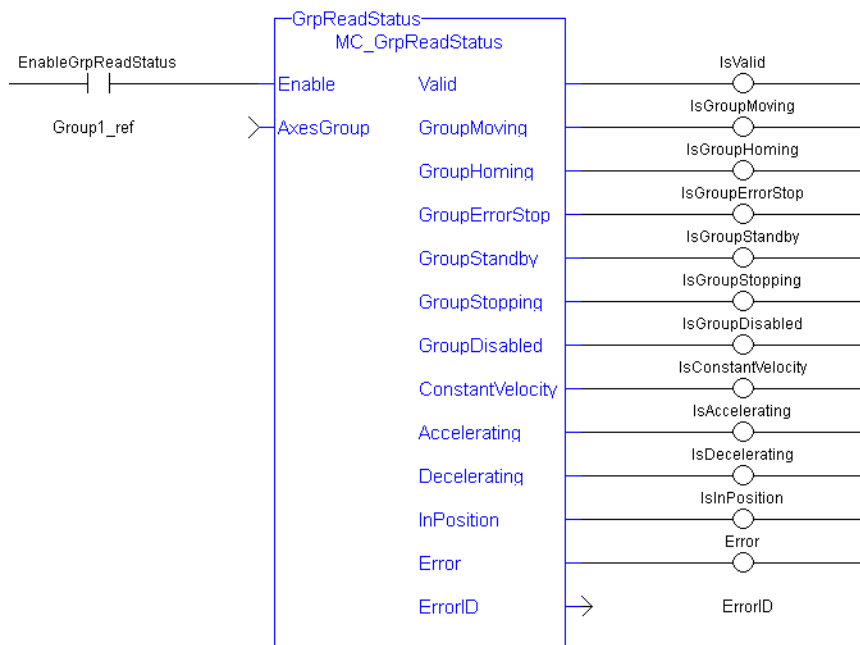
¹ These outputs are mutually exclusive.

- Only one is TRUE at a time.
- All others are FALSE.
- See Group State Diagrams.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Check boolean status bits for an Axis Group
Inst_MC_GrpReadStatus( EnableGrpReadStatus, Group1_Ref );

IsGroupMoving:= Inst_MC_GrpReadStatus.GroupMoving;
```

```
IsGroupErrorStop:= Inst_MC_GrpReadStatus.GroupErrorStop;  
IsGroupStandby:= Inst_MC_GrpReadStatus.GroupStandby;  
IsGroupDisabled:= Inst_MC_GrpReadStatus.GroupDisabled;  
Accelerating:= Inst_MC_GrpReadStatus.Accelerating;  
IsConstantVelocity:= Inst_MC_GrpReadStatus.ConstantVelocity;  
IsInPosition:= Inst_MC_GrpReadStatus.InPosition;
```

See Also

"MC_ErrorDescription" (→ p. 863)

3.3.13.2 Coordinated Motion - Motion

Name	Description
MC_ AxisSetDefaults	Sets the default kinematic parameters for an axis.
MC_ ErrorDescription	Converts the PLCopen error IDs into message strings.
MC_GrpHalt	Performs a controlled motion stop of all axes in the group.
MC_ GrpSetOverride	Sets the velocity factor that is multiplied to the commanded velocity of all axes in the group.
MC_ MoveCircAbs	Commands interpolated circular movement on an axes group to the specified absolute positions in the coordinate system as specified by the CoordSystem argument.
MC_MoveCircRel	Commands interpolated circular movement on an axes group to the specified relative positions in the coordinate system as specified by the CoordSystem argument.
MC_MoveDirAbs	Commands movement of an axes group to an absolute position regardless of path.
MC_MoveDirRel	Commands movement of an axes group to a relative position regardless of path.
MC_MoveLinAbs	Commands interpolated linear movement on an axes group to the specified absolute positions.
MC_MoveLinRel	Commands interpolated linear movement on an axes group to the specified relative positions.

3.3.13.2.1 MC_AxisSetDefaults

PLCopen



Pipe Network

**Function Block** - Sets the default kinematic parameters for an axis.**NOTE**

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.
- See Coordinated Motion, the top-level topic for Coordinated Motion.
- The **MC_AxisSetDefaults** function block sets the default kinematic variables for "MC_MoveDirAbs" (→ p. 842) and "MC_MoveDirRel" (→ p. 845).
 - These variables are only used with the MC_MoveDir function blocks.
- Each axis in the group must have the default kinematic parameters of Velocity, Acceleration, and Deceleration set to values greater than 0 (zero).
 - Each axis in the group must have these values set before a direct move can be started.
- The function block returns an error if the group state is not GroupStandby or GroupDisabled.

NOTE

- Jerk with a non-zero value is not supported by "MC_MoveDirAbs" (→ p. 842) or "MC_MoveDirRel" (→ p. 845).
- Jerk parameters are ignored.

Inputs

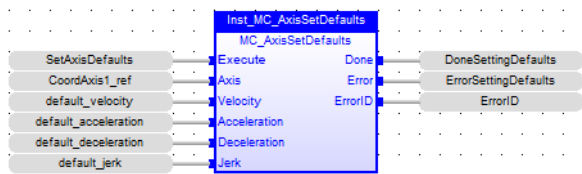
Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to set the default kinematic parameters.
Axis	AXIS_REF	No range	N/A	No default	Reference to the axis which has its default kinematic parameters set.
Velocity	LREAL	$0 < \text{Velocity} < (20 * \text{Acceleration})$ and $0 < \text{Velocity} < (20 * \text{Deceleration})$	User units / sec	No default	• The default velocity. • See Limitations on Acceleration and Jerk.
Acceleration	LREAL	$\text{Jerk} > (\text{Acceleration} / 2)$	User units / sec ²	No default	• Trapezoidal: Acceleration rate. • S-curve: Maximum acceleration. • See S-curve and Trapezoidal Acceleration / Deceleration. • See Acceleration and Jerk Parameters for Function Blocks. • See Limitations on Acceleration and Jerk.

Input	Data Type	Range	Unit	Default	Description
Deceleration	LREAL	Jerk > (Deceleration / 2)	User units / sec ²	No default	- Trapezoidal: Deceleration rate. - S-curve: Unused.
Jerk	LREAL	Jerk > (Acceleration / 2) and Jerk > (Deceleration / 2)	User units / sec ³	No default	- Trapezoidal: 0 (zero). - S-curve: Constant jerk. NOTE - The Jerk value is ignored for motion. - Only trapezoidal motion is supported. - See S-curve and Trapezoidal Acceleration / Deceleration. - See Acceleration and Jerk Parameters for Function Blocks.

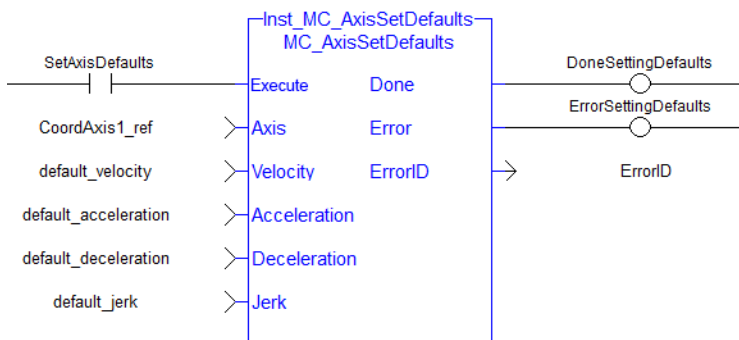
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* ST MC_AxisSetDefaults Example *)

default_velocity      := 50.0;
default_acceleration  := 250.0;
default_deceleration  := 300.0;
default_jerk          := 1000.0;

Inst_MC_AxisSetDefaults ( TRUE, CoordAxis1_ref, default_velocity, default_
acceleration, default_deceleration, default_jerk);
```

See Also

- Differences Between Functions and Function Blocks
- Function Call
- "MC_ErrorDescription" (→ p. 863)
- "MC_MoveDirAbs" (→ p. 842)
- "MC_MoveDirRel" (→ p. 845)

3.3.13.2.2 MC_GrpHalt

PLCopen Pipe Network **Function Block** - Performs a controlled motion stop of all axes in the group.**NOTE**

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.

- **MC_GrpHalt** can be aborted.
- When the path velocity reaches 0 (zero):
 - Any queued moves are flushed from the buffer.
 - The Done output is set.
 - The state transitions to GroupStandby.
- **MC_GrpHalt** does NOT prevent:
 - A single axis from executing.
 - Other Coordinated Motion moves from executing once MC_GrpHalt has completed.
- See What MC_GrpHalt Does.
- See Differences between MC_GrpHalt and MC_GrpStop.
- See Coordinated Motion, the top-level topic for Coordinated Motion.

Inputs

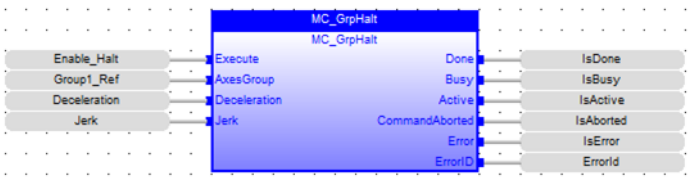
Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, the command to halt all of the axes in the group is initiated.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axes group the axes are stopped in.
Deceleration	LREAL	No range	User units / sec ²	No default	The path deceleration rate for all axes in the group. • If Deceleration is 0 or negative, the deceleration parameter of the active move is used. • If Deceleration is less than the deceleration of the active move, the deceleration parameter of the active move is used. • This guarantees the stop action completes before the endpoint of the move is reached. • See Limitations on Acceleration and Jerk.

Input	Data Type	Range	Unit	Default	Description
Jerk	LREAL	No range	User units / sec ³	No default	The path jerk for all axes in the group. - If Jerk is negative, the jerk parameter of the active move is used. - If Jerk is zero, then zero jerk is used. - If Jerk is less than the jerk of the active move, the jerk parameter of the active move is used. - This guarantees the stop action completes before the endpoint of the move is reached. - See Limitations on Acceleration and Jerk.

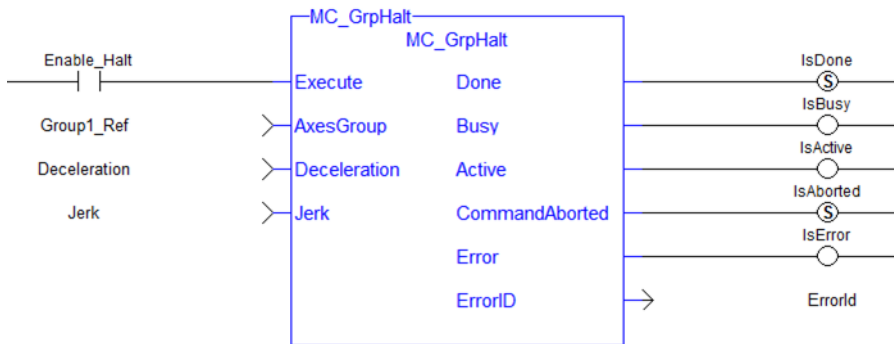
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	TRUE from the moment the EXECUTE input is TRUE until the halt is complete.
Active	BOOL	FALSE, TRUE	N/A	If TRUE, the halt is still executing.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_GrpHalt ( EnableHalt, Group1_Ref, Deceleration, Jerk );
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_GrpStop" (→ p. 777)

3.3.13.2.3 MC_GrpSetOverride

PLCopen

Pipe Network

Function Block - Sets the velocity factor that is multiplied to the commanded velocity of all axes in the group.

- See Coordinated Motion, the top-level topic for Coordinated Motion.
- This function or function block does not generate any motion.
- Changing the velocity factor during an S-Curve Corner transition does not change the profile of the transition.
 - The new velocity factor is applied after the S-Curve Corner transition completes.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, changes the velocity multiplier for the axes group.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axes group the velocity multiplier is applied to.
VelFactor	REAL	0.0 to 2.0	N/A	No default	The new multiplier factor for the commanded velocity of the axes group.

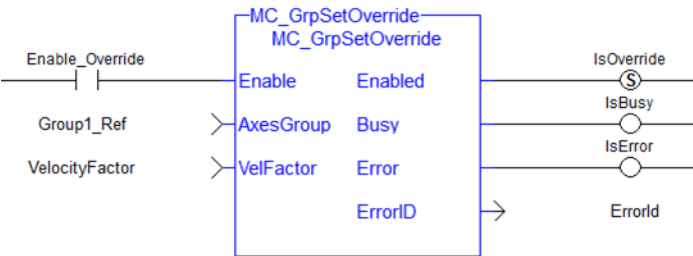
Outputs

Output	Data Type	Range	Unit	Description
Enabled	BOOL	FALSE, TRUE	N/A	Indicates that the override was successful.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_GrpSetOverride( EnableOverride, Group1_Ref, VelocityFactor );
```

See Also

"MC_ErrorDescription" (→ p. 863)

3.3.13.2.4 MC_MoveCircAbs

PLCopen ✓

Pipe Network ✓

 **Function Block** - Commands interpolated circular movement on an axes group to the specified absolute positions in the coordinate system as specified by the CoordSystem argument.

- See Coordinated Motion, the top-level topic for Coordinated Motion.
- See Circular Moves Diagrams about movement options.
- An error is returned if:
 - The group is in the GroupDisabled state.
 - The input parameters do not meet the required precision.
 - See Precision Requirements for Circular Move Input Parameters for more information.
- When all motion has completed successfully, the axes group state is GroupStandby.

NOTE

- Circular motion is only supported for axes groups with only two attached axes.

Inputs

Input	Data Type	Range	Unit	Default	Description						
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to perform a circular absolute move.						
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axis group that performs the circular absolute move.						
CircMode	SINT	Enumerated	N/A	No default	• Specifies the meaning of the AuxPoint[] input. • One of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>CIRC_MODE_BORDER</td><td>The specified point lies on the border of the circle.</td></tr><tr><td>CIRC_MODE_CENTER</td><td>The specified point is the center of the circle.</td></tr></table>	Value	Description	CIRC_MODE_BORDER	The specified point lies on the border of the circle.	CIRC_MODE_CENTER	The specified point is the center of the circle.
Value	Description										
CIRC_MODE_BORDER	The specified point lies on the border of the circle.										
CIRC_MODE_CENTER	The specified point is the center of the circle.										

Input	Data Type	Range	Unit	Default	Description						
AuxPoint	LREAL[]	Dimension: The number of axes mapped in the AxesGroup. No range	N/A	No default	Required Precision 1 part in 100,000. - See Precision Requirements for Circular Move Input Parameters. Description - Array of absolute positions for each axis in the group. - The meaning depends on the value of the CircMode input: - CIRC_MODE_BORDER - AuxPoint defines a point on the circle which is crossed on the path from the starting to the end point. - CIRC_MODE_CENTER - AuxPoint defines the center point of the circle.						
EndPoint	LREAL[]	Dimension: The number of axes mapped in the AxesGroup. No range	N/A	No default	Required Precision 1 part in 100,000. - See Precision Requirements for Circular Move Input Parameters. Description Array of absolute end positions for each axis in the group.						
PathChoice	SINT	Enumerated	N/A	No default	- Specifies the direction of the path. - This argument is only relevant when CircMode is CIRC_MODE_CENTER. One of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>CIRC_PATH_CLKWISE</td><td>Move in the clockwise direction.</td></tr><tr><td>CIRC_PATH_COUNTERCLKWISE</td><td>Move in the counter clockwise direction.</td></tr></table>	Value	Description	CIRC_PATH_CLKWISE	Move in the clockwise direction.	CIRC_PATH_COUNTERCLKWISE	Move in the counter clockwise direction.
Value	Description										
CIRC_PATH_CLKWISE	Move in the clockwise direction.										
CIRC_PATH_COUNTERCLKWISE	Move in the counter clockwise direction.										
Velocity	LREAL	0 < Velocity < (20 * Acceleration) and 0 < Velocity < (20 * Deceleration)	User units / sec	No default	- Maximum velocity of the defined path. - See Limitations on Acceleration and Jerk.						

Input	Data Type	Range	Unit	Default	Description								
Acceleration	LREAL	Acceleration > (Velocity / 20)	User units / sec ²	No default	- Maximum acceleration. - See Limitations on Acceleration and Jerk.								
Deceleration	LREAL	Deceleration > (Velocity / 20)	User units / sec ²	No default	- Maximum deceleration. - See Limitations on Acceleration and Jerk.								
Jerk	LREAL	Trapezoidal velocity profiles: 0 (zero) S-Curve velocity profiles: Jerk > (Acceleration / 2) and Jerk > (Deceleration / 2)	User units / sec ³	No default	- Maximum jerk. - See Limitations on Acceleration and Jerk.								
CoordSystem	SINT	Enumerated	N/A	No default	The coordinate system used when commanding the circular absolute move. - Only the ACS and MCS coordinate system are supported. - See Coordinate Systems. One of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>CS_ACS</td><td>Axes Coordinate System</td></tr><tr><td>CS_MCS</td><td>Machine Coordinate System</td></tr><tr><td>CS_PCS</td><td>Product Coordinate System</td></tr></table>	Value	Description	CS_ACS	Axes Coordinate System	CS_MCS	Machine Coordinate System	CS_PCS	Product Coordinate System
Value	Description												
CS_ACS	Axes Coordinate System												
CS_MCS	Machine Coordinate System												
CS_PCS	Product Coordinate System												

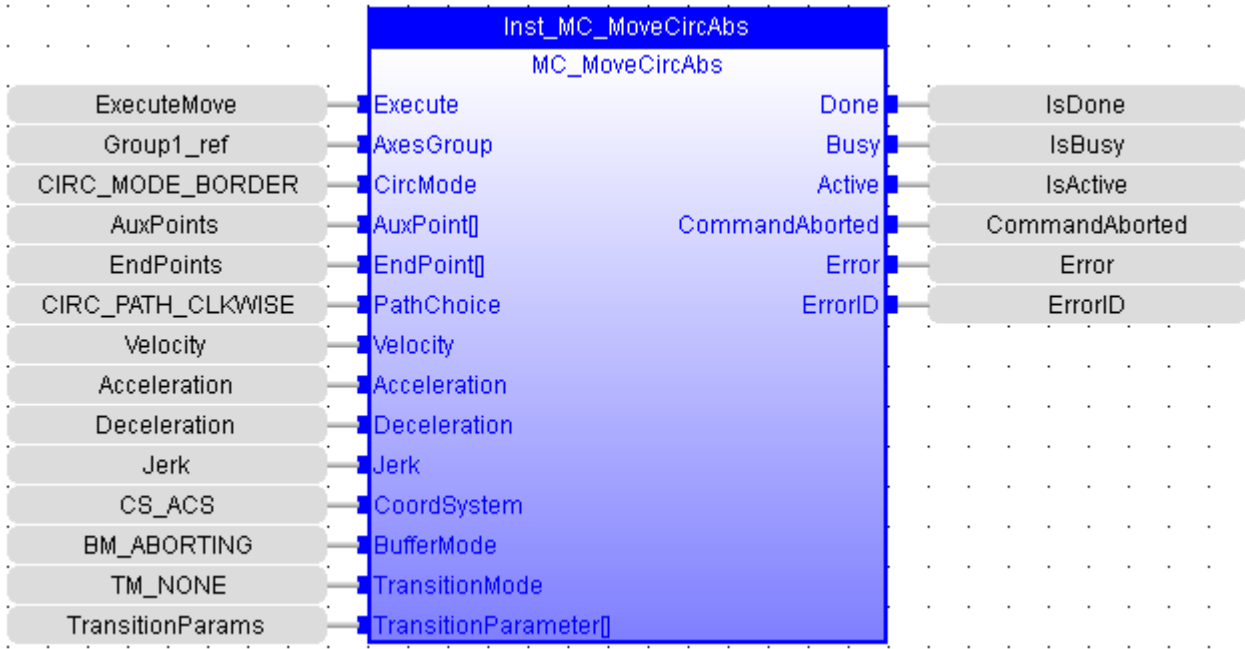
Input	Data Type	Range	Unit	Default	Description						
BufferMode	SINT	Enumerated	N/A	No default	Defines the chronological sequence of the function block relative to the previous block. - BM_ABORTING is not allowed with this function block. - See Buffer Modes. - Some individual axis velocities can make an abrupt change if the path of the next move travels in a different direction. - A transition move can be programmed in the TransitionMode input to avoid this.						
TransitionMode	SINT	Enumerated	N/A	No default	- Coupled with the TransitionParameter, this input defines the shape and dynamics of the inserted contour to connect the current motion with the next motion in the queue. - See Transition Between Moves. - Circular moves do not support the TransitionMode TM_SCURVE_CORNER. - One of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>TM_NONE</td><td>No transition.</td></tr><tr><td>TM_CORNER_DISTANCE</td><td>The velocity value of the transition segment.</td></tr></table>	Value	Description	TM_NONE	No transition.	TM_CORNER_DISTANCE	The velocity value of the transition segment.
Value	Description										
TM_NONE	No transition.										
TM_CORNER_DISTANCE	The velocity value of the transition segment.										
TransitionParameter	LREAL[]	Dimension: 1 to N No range	N/A	No default	This array is dependent on the specified TransitionMode. - N values are supplier specified dependent on the selected TransitionMode. - Transition parameter values are applied to the AxesGroup. - See Transition Between Moves.						

Outputs

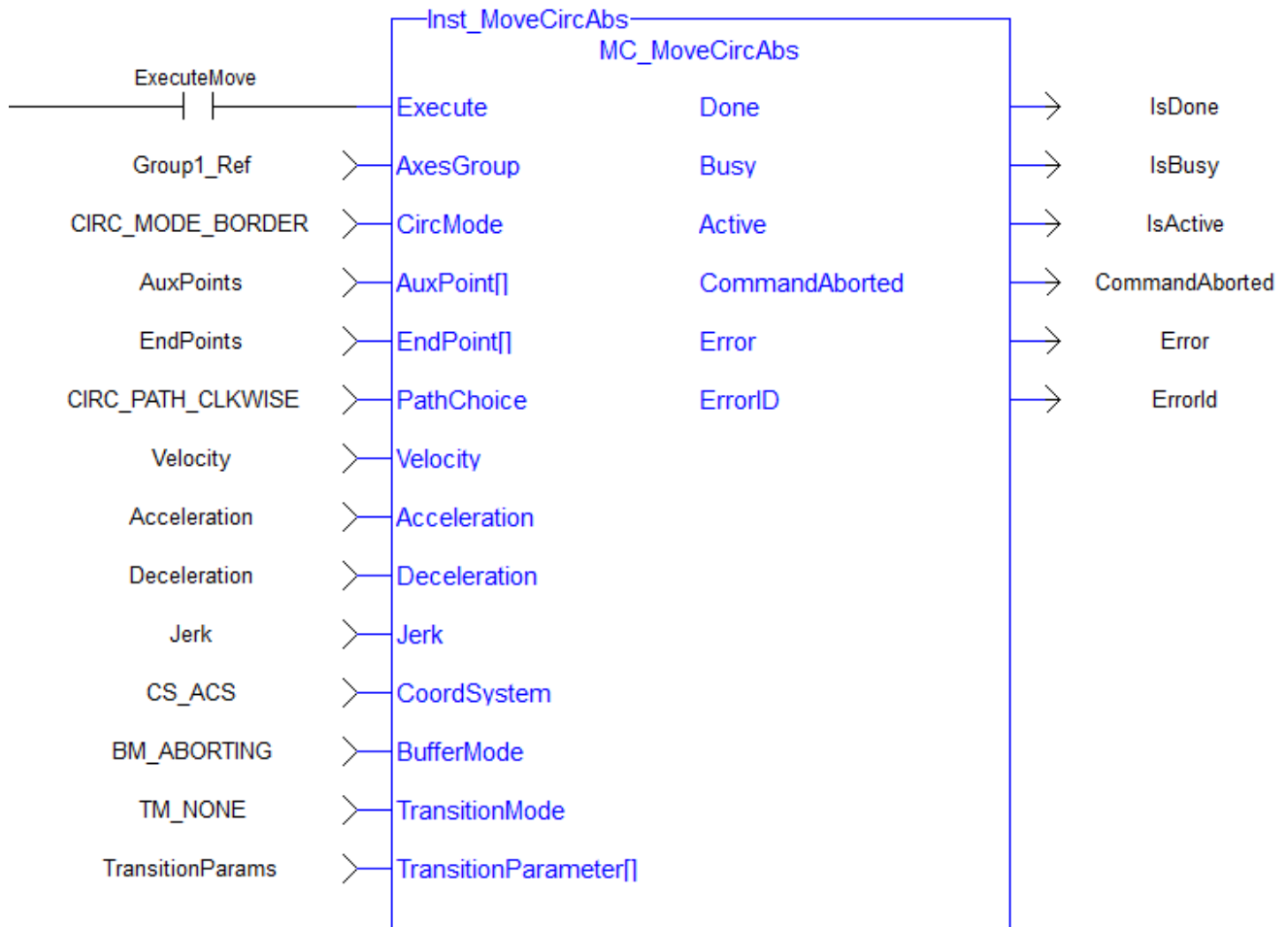
Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.

Output	Data Type	Range	Unit	Description
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Active	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is controlling motion.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_MoveCircAbs(ExecuteMove, Group1_Ref, MC_CIRC_MODE_BORDER, AuxPoints,
EndPoints, CIRC_PATH_CLKWISE, Velocity, Acceleration, Deceleration, Jerk, CS_
ACS, BM_ABORTING, TM_NONE, TransitionParams);
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_MoveCircRel" (→ p. 836)

3.3.13.2.5 MC_MoveCircRel

PLCopen

Pipe Network

 **Function Block** - Commands interpolated circular movement on an axes group to the specified relative positions in the coordinate system as specified by the CoordSystem argument.

- See Coordinated Motion, the top-level topic for Coordinated Motion.
- See Circular Moves Diagrams about movement options.
- An error is returned if the group is in the GroupDisabled state.
- When all motion has completed successfully, the axes group state is GroupStandby.

NOTE

- Circular motion is only supported for axes groups with only two attached axes.

Inputs

Input	Data Type	Range	Unit	Default	Description						
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to perform a circular relative move.						
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axis group that performs the circular relative move.						
CircMode	SINT	Enumerated	N/A	No default	• Specifies the meaning of the AuxPoint[] input. • One of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>CIRC_MODE_BORDER</td><td>The specified point lies on the border of the circle.</td></tr><tr><td>CIRC_MODE_CENTER</td><td>The specified point is the center of the circle.</td></tr></table>	Value	Description	CIRC_MODE_BORDER	The specified point lies on the border of the circle.	CIRC_MODE_CENTER	The specified point is the center of the circle.
Value	Description										
CIRC_MODE_BORDER	The specified point lies on the border of the circle.										
CIRC_MODE_CENTER	The specified point is the center of the circle.										

Input	Data Type	Range	Unit	Default	Description						
AuxPoint	LREAL[]	Dimension: The number of axes mapped in the AxesGroup. No range	N/A	No default	Required Precision 1 part in 100,000. - See Precision Requirements for Circular Move Input Parameters. Description - Array of relative positions for each axis in the group. - In all cases the points are relative to the starting point. - The meaning depends on the value of the CircMode input: - CIRC_MODE_BORDER - AuxPoint defines a point on the circle which is crossed on the path from the starting to the end point. - CIRC_MODE_CENTER - AuxPoint defines the center point of the circle.						
EndPoint	LREAL[]	Dimension: The number of axes mapped in the AxesGroup. No range	N/A	No default	Required Precision 1 part in 100,000. - See Precision Requirements for Circular Move Input Parameters. Description Array of relative end positions for each axis in the group.						
PathChoice	SINT	Enumerated	N/A	No default	- Specifies the direction of the path. - This argument is only relevant when CircMode is CIRC_MODE_CENTER. - One of these enumeration values: <table><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>CIRC_PATH_CLKWISE</td><td>Move in the clockwise direction.</td></tr><tr><td>CIRC_PATH_COUNTERCLKWISE</td><td>Move in the counter clockwise direction.</td></tr></tbody></table>	Value	Description	CIRC_PATH_CLKWISE	Move in the clockwise direction.	CIRC_PATH_COUNTERCLKWISE	Move in the counter clockwise direction.
Value	Description										
CIRC_PATH_CLKWISE	Move in the clockwise direction.										
CIRC_PATH_COUNTERCLKWISE	Move in the counter clockwise direction.										

Input	Data Type	Range	Unit	Default	Description								
Velocity	LREAL	0 < Velocity < (20 * Acceleration) and 0 < Velocity < (20 * Deceleration)	User units / sec	No default	- Maximum velocity of the defined path. - See Limitations on Acceleration and Jerk.								
Acceleration	LREAL	Acceleration > (Velocity / 20)	User units / sec ²	No default	- Maximum acceleration. - See Limitations on Acceleration and Jerk.								
Deceleration	LREAL	Deceleration > (Velocity / 20)	User units / sec ²	No default	- Maximum deceleration. - See Limitations on Acceleration and Jerk.								
Jerk	LREAL	Trapezoidal velocity profiles: 0 (zero) S-Curve velocity profiles: Jerk > (Acceleration / 2) and Jerk > (Deceleration / 2)	User units / sec ³	No default	- Maximum jerk. - See Limitations on Acceleration and Jerk.								
CoordSystem	SINT	Enumerated	N/A	No default	The coordinate system used when commanding the circular relative move. - Only the ACS and MCS coordinate system are supported. - See Coordinate Systems. One of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>CS_ACS</td><td>Axes Coordinate System</td></tr><tr><td>CS_MCS</td><td>Machine Coordinate System</td></tr><tr><td>CS_PCS</td><td>Product Coordinate System</td></tr></table>	Value	Description	CS_ACS	Axes Coordinate System	CS_MCS	Machine Coordinate System	CS_PCS	Product Coordinate System
Value	Description												
CS_ACS	Axes Coordinate System												
CS_MCS	Machine Coordinate System												
CS_PCS	Product Coordinate System												

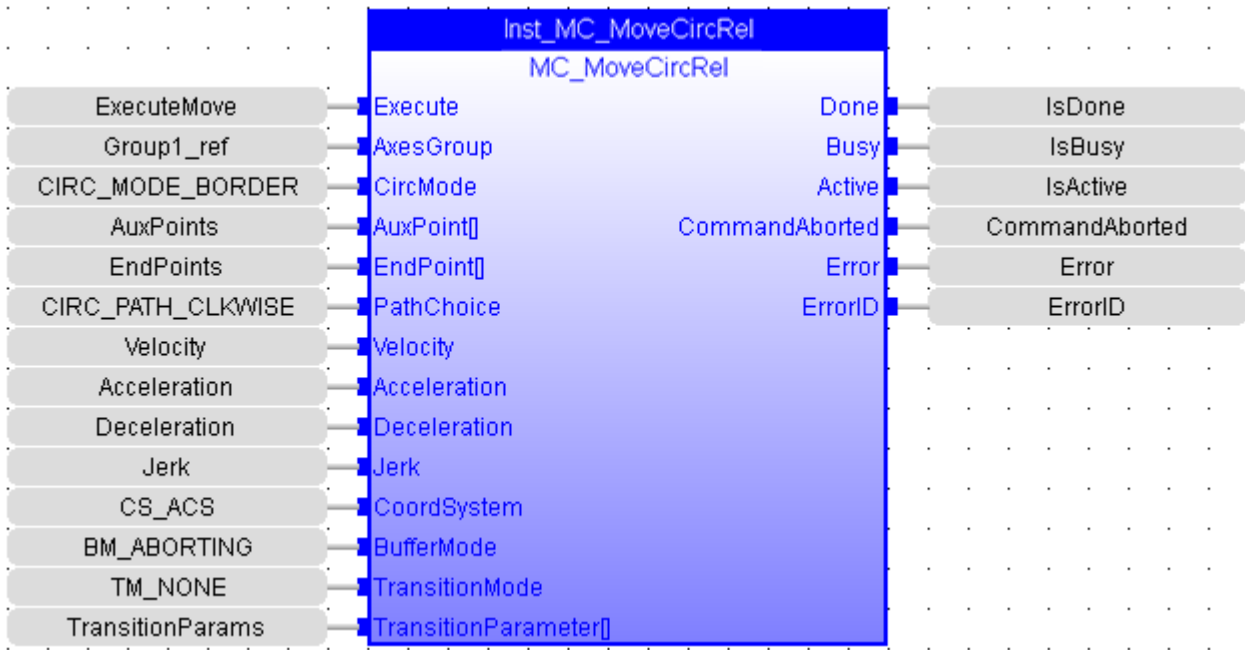
Input	Data Type	Range	Unit	Default	Description						
BufferMode	SINT	Enumerated	N/A	No default	Defines the chronological sequence of the function block relative to the previous block. - BM_ABORTING is not allowed with this function block. - See Buffer Modes. - Some individual axis velocities can make an abrupt change if the path of the next move travels in a different direction. - A transition move can be programmed in the TransitionMode input to avoid this.						
TransitionMode	SINT	Enumerated	N/A	No default	- Coupled with the TransitionParameter, this input defines the shape and dynamics of the inserted contour to connect the current motion with the next motion in the queue. - See Transition Between Moves. - Circular moves do not support the TransitionMode TM_SCURVE_CORNER. - One of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>TM_NONE</td><td>No transition.</td></tr><tr><td>TM_CORNER_DISTANCE</td><td>The velocity value of the transition segment.</td></tr></table>	Value	Description	TM_NONE	No transition.	TM_CORNER_DISTANCE	The velocity value of the transition segment.
Value	Description										
TM_NONE	No transition.										
TM_CORNER_DISTANCE	The velocity value of the transition segment.										
TransitionParameter	LREAL[]	Dimension: 1 to N No range	N/A	No default	This array is dependent on the specified TransitionMode. - N values are supplier specified dependent on the selected TransitionMode. - Transition parameter values are applied to the AxesGroup. - See Transition Between Moves.						

Outputs

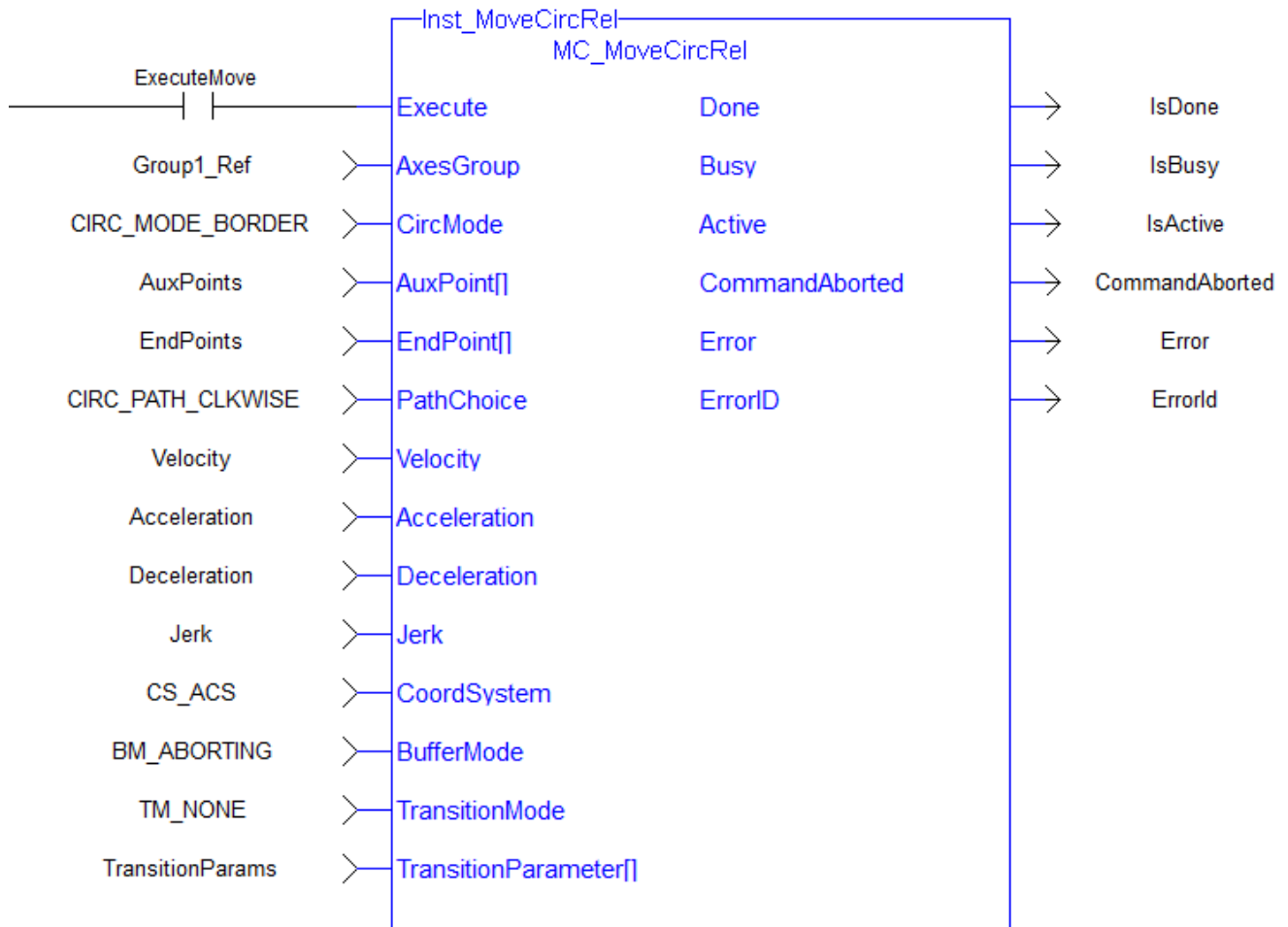
Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.

Output	Data Type	Range	Unit	Description
Active	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is controlling motion.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_MoveCircRel( ExecuteMove, Group1_Ref, MC_CIRC_MODE_BORDER, AuxPoints,
EndPoints, CIRC_PATH_CLKWISE, Velocity, Acceleration, Deceleration, Jerk, CS_
ACS, BM_ABORTING, TM_NONE, TransitionParams );
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_MoveCircAbs" (→ p. 830)

3.3.13.2.6 MC_MoveDirAbs

PLCopen

✓

Pipe Network

✓

 **Function Block** - Commands movement of an axes group to an absolute position regardless of path.

- See Coordinated Motion, the top-level topic for Coordinated Motion.
- When all motion has completed successfully, the axes group state is GroupStandby.
- **MC_MoveDirAbs** commands the movement of an axes group to a specified absolute position in the specified coordinate system regardless of how (on which path) the target position is reached.

NOTE

- An error is returned if the group is in the GroupDisabled state.
- This function block does not have its own Acceleration, Deceleration, Velocity, and Jerk arguments.
 - These are set using "MC_AxisSetDefaults" (→ p. 822).
- The maximum number of axes is set by the **MaxNumberOfAxes** input set in the "MC_CreateAxesGrp" (→ p. 764) function block.

NOTE

- Circular motion is only supported for axes groups with only two attached axes.

Inputs

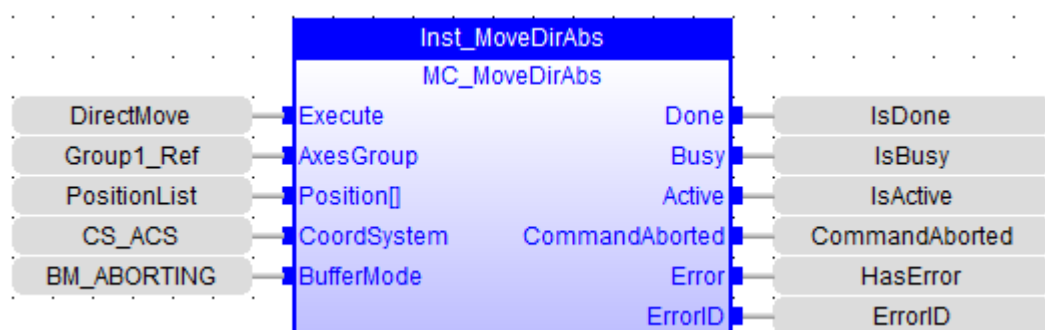
Input	Data Type	Range	Unit	Default	Description								
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to perform a direct absolute move.								
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	Reference to an axes group.								
Position	LREAL[]	• Dimension: The number of axes mapped in the AxesGroup. • No range	User units	No default	Array of absolute end positions for each axis in the group.								
CoordSystem	SINT	Enumerated	N/A	No default	The coordinate system used when commanding the direct absolute move. • Only the ACS and MCS coordinate system are supported. • See Coordinate Systems. One of these enumeration values:<table><tr><th>Value</th><th>Description</th></tr><tr><td>CS_ACS</td><td>Axes Coordinate System</td></tr><tr><td>CS_MCS</td><td>Machine Coordinate System</td></tr><tr><td>CS_PCS</td><td>Product Coordinate System</td></tr></table>	Value	Description	CS_ACS	Axes Coordinate System	CS_MCS	Machine Coordinate System	CS_PCS	Product Coordinate System
Value	Description												
CS_ACS	Axes Coordinate System												
CS_MCS	Machine Coordinate System												
CS_PCS	Product Coordinate System												

Input	Data Type	Range	Unit	Default	Description						
BufferMode	SINT	Enumerated	N/A	No default	- Defines the chronological sequence of the function block relative to the previous block. - Blending buffer modes are not supported by direct moves. These are the valid buffer modes for this function block: <table><tr><th>Value</th><th>Description</th></tr><tr><td>BM_ABORTING</td><td>Aborting Mode</td></tr><tr><td>BM_BUFFERED</td><td>Buffered Mode</td></tr></table> See Buffer Modes.	Value	Description	BM_ABORTING	Aborting Mode	BM_BUFFERED	Buffered Mode
Value	Description										
BM_ABORTING	Aborting Mode										
BM_BUFFERED	Buffered Mode										

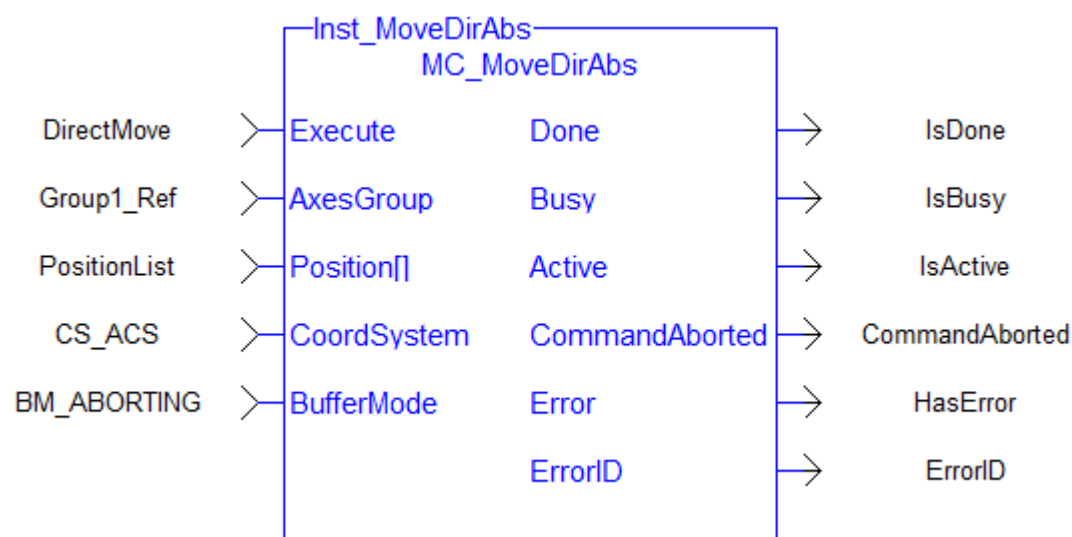
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Active	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is controlling motion.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_MoveDirAbs(DirectMove, Group1_Ref, PositionList, CS_ACS, BM_ABORTING);
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_MoveDirRel" (→ p. 845)

3.3.13.2.7 MC_MoveDirRel

PLCopen



Pipe Network



Function Block - Commands movement of an axes group to a relative position regardless of path.

- See Coordinated Motion, the top-level topic for Coordinated Motion.
- When all motion has completed successfully, the axes group state is GroupStandby.
- **MC_MoveDirRel** commands a movement of an axes group to a relative position in the specified coordinate system regardless of how (on which path) the target position is reached.

NOTE

- An error is returned if the group is in the GroupDisabled state.
- This function block does not have its own Acceleration, Deceleration, Velocity, and Jerk arguments.
 - These are set using "MC_AxisSetDefaults" (→ p. 822).
- The maximum number of axes is set by the **MaxNumberOfAxes** input set in the "MC_CreateAxesGrp" (→ p. 764) function block.

NOTE

- Circular motion is only supported for axes groups with only two attached axes.

Inputs

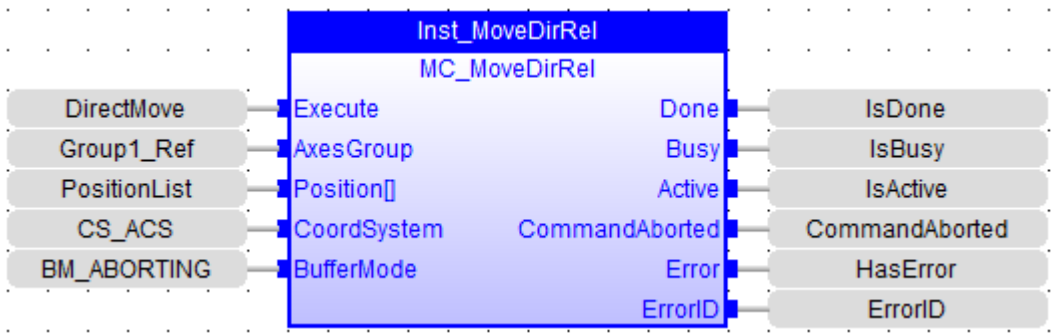
Input	Data Type	Range	Unit	Default	Description								
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to perform a direct relative move.								
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	Reference to an axes group.								
Distance	LREAL[]	Dimension: The number of axes mapped in the AxesGroup. No range	User units	No default	An array containing the distance for each axis in the group.								
CoordSystem	SINT	Enumerated	N/A	No default	The coordinate system used when commanding the direct relative move. - Only the ACS and MCS coordinate system are supported. - See Coordinate Systems. One of these enumeration values:<table><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>CS_ACS</td><td>Axes Coordinate System</td></tr><tr><td>CS_MCS</td><td>Machine Coordinate System</td></tr><tr><td>CS_PCS</td><td>Product Coordinate System</td></tr></tbody></table>	Value	Description	CS_ACS	Axes Coordinate System	CS_MCS	Machine Coordinate System	CS_PCS	Product Coordinate System
Value	Description												
CS_ACS	Axes Coordinate System												
CS_MCS	Machine Coordinate System												
CS_PCS	Product Coordinate System												

Input	Data Type	Range	Unit	Default	Description						
BufferMode	SINT	Enumerated	N/A	No default	- Defines the chronological sequence of the function block relative to the previous block. - Blending buffer modes are not supported by direct moves. These are the valid buffer modes for this function block: <table><tr><th>Value</th><th>Description</th></tr><tr><td>BM_ABORTING</td><td>Aborting Mode</td></tr><tr><td>BM_BUFFERED</td><td>Buffered Mode</td></tr></table> See Buffer Modes.	Value	Description	BM_ABORTING	Aborting Mode	BM_BUFFERED	Buffered Mode
Value	Description										
BM_ABORTING	Aborting Mode										
BM_BUFFERED	Buffered Mode										

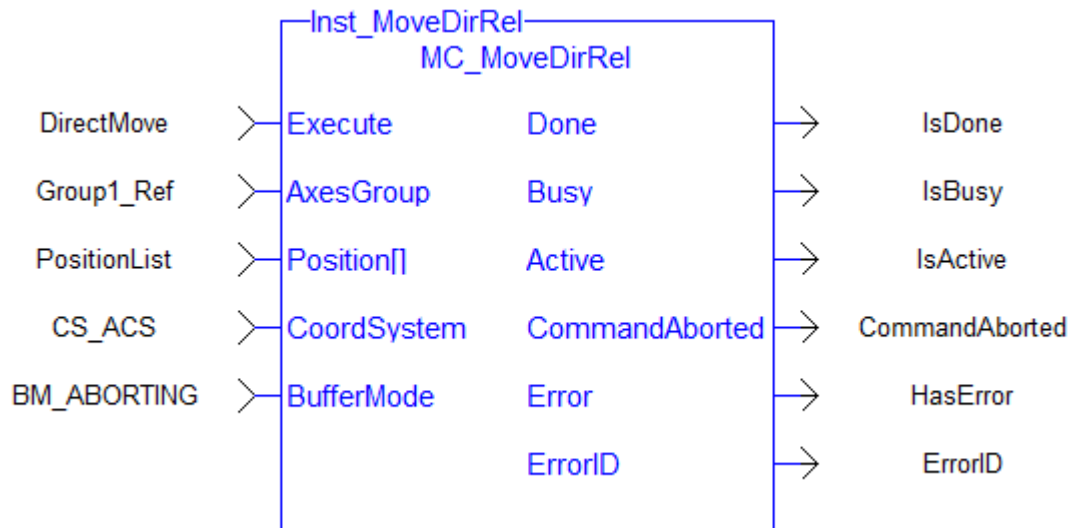
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Active	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is controlling motion.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_MoveDirRel(ExecuteMove, Group1_Ref, DistanceList, , CS_ACS, BM_ABORTING);
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_MoveDirAbs" (→ p. 842)

3.3.13.2.8 MC_MoveLinAbs

PLCopen Pipe Network 

Function Block - Commands interpolated linear movement on an axes group to the specified absolute positions.

- See Coordinated Motion, the top-level topic for Coordinated Motion.
- When all motion has completed successfully, the axes group state is GroupStandby.
- **MC_MoveLinAbs** commands interpolated linear movement on an axes group to the specified absolute positions in the coordinate system as specified by the **CoordSystem** argument.
 - The dimensionality of the move is determined by the number of axes mapped to the group.

NOTE

- An error is returned if the group is in the GroupDisabled state.
- The maximum number of axes is set by the **MaxNumberOfAxes** input set in the "MC_CreateAxesGrp" (→ p. 764) function block.

NOTE

- Circular motion is only supported for axes groups with only two attached axes.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to perform a linear absolute move.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axis group that performs the linear absolute move.
Position	LREAL[]	Dimension: The number of axes mapped in the AxesGroup. No range	User units	No default	Array of absolute end positions for each axis in the group.
Velocity	LREAL	$0 < \text{Velocity} < (20 * \text{Acceleration})$ and $0 < \text{Velocity} < (20 * \text{Deceleration})$	User units / sec	No default	• Maximum velocity of the defined path. • See Limitations on Acceleration and Jerk.
Acceleration	LREAL	$\text{Acceleration} > (\text{Velocity} / 20)$	User units / sec ²	No default	• Maximum acceleration. • See Limitations on Acceleration and Jerk.
Deceleration	LREAL	$\text{Deceleration} > (\text{Velocity} / 20)$	User units / sec ²	No default	• Maximum deceleration. • See Limitations on Acceleration and Jerk.

Input	Data Type	Range	Unit	Default	Description								
Jerk	LREAL	Trapezoidal velocity profiles: 0 (zero) S-Curve velocity profiles: Jerk > (Acceleration / 2) and Jerk > (Deceleration / 2)	User units / sec ³	No default	- Maximum jerk. - See Limitations on Acceleration and Jerk.								
CoordSystem	SINT	Enumerated	N/A	No default	The coordinate system used when commanding the linear absolute move. - Only the ACS and MCS coordinate system are supported. - See Coordinate Systems. One of these enumeration values: <table><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>CS_ACS</td><td>Axes Coordinate System</td></tr><tr><td>CS_MCS</td><td>Machine Coordinate System</td></tr><tr><td>CS_PCS</td><td>Product Coordinate System</td></tr></tbody></table>	Value	Description	CS_ACS	Axes Coordinate System	CS_MCS	Machine Coordinate System	CS_PCS	Product Coordinate System
Value	Description												
CS_ACS	Axes Coordinate System												
CS_MCS	Machine Coordinate System												
CS_PCS	Product Coordinate System												
BufferMode	SINT	Enumerated	N/A	No default	Defines the chronological sequence of the function block relative to the previous block. - BM_ABORTING is not allowed with this function block. - See Buffer Modes. - Some individual axis velocities can make an abrupt change if the path of the next move travels in a different direction. - A transition move can be programmed in the TransitionMode input to avoid this.								

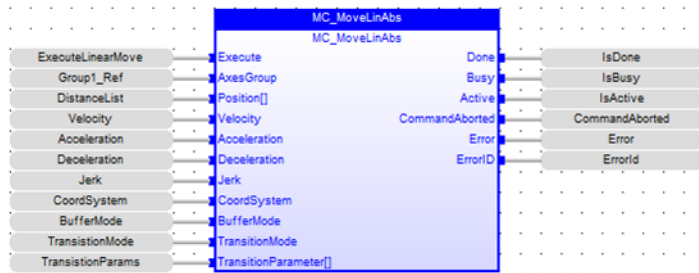
Input	Data Type	Range	Unit	Default	Description								
TransitionMode	SINT	Enumerated	N/A	No default	Coupled with the TransitionParameter, this input defines the shape and dynamics of the inserted contour to connect the current motion with the next motion in the queue. - The transition mode is limited to TM_NONE or TM_SCURVE_CORNER for groups with more than three axes. - See Transition Between Moves. One of these enumeration values: <table><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>TM_NONE</td><td>No transition.</td></tr><tr><td>TM_CORNER_DISTANCE</td><td>Corner distance transition.</td></tr><tr><td>TM_SCURVE_CORNER</td><td>S-Curve corner transition.</td></tr></tbody></table>	Value	Description	TM_NONE	No transition.	TM_CORNER_DISTANCE	Corner distance transition.	TM_SCURVE_CORNER	S-Curve corner transition.
Value	Description												
TM_NONE	No transition.												
TM_CORNER_DISTANCE	Corner distance transition.												
TM_SCURVE_CORNER	S-Curve corner transition.												
TransitionParameter	LREAL[]	Dimension: 1 to N No range	N/A	No default	This array is dependent on the specified TransitionMode. - N values are supplier specified dependent on the selected TransitionMode. - Transition parameter values are applied to the AxesGroup. - See Transition Between Moves.								

Outputs

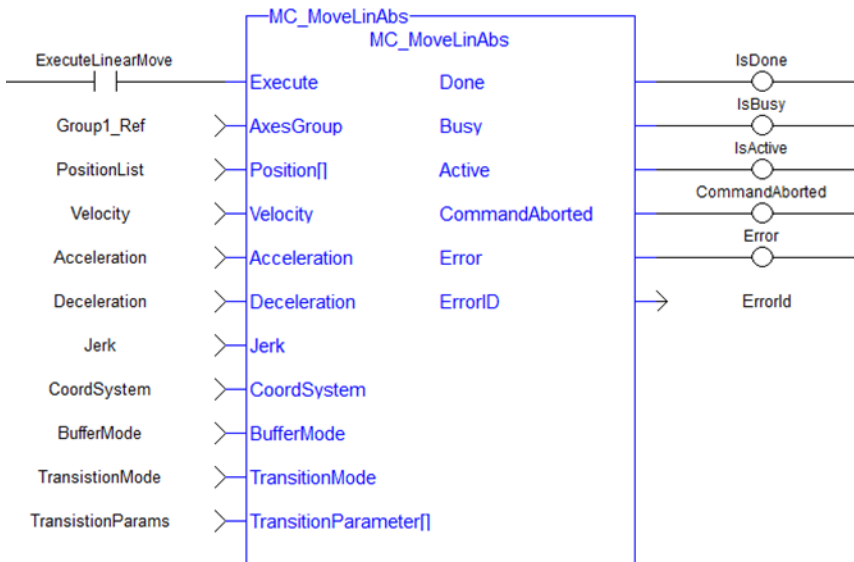
Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Active	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is controlling motion.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.

Output	Data Type	Range	Unit	Description
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* Inst_MC_MoveLinAbsST example *)
Inst_MC_MoveLinAbs( ExecuteLinearMove, Group1_Ref, PositionList, Velocity,
Acceleration, Deceleration, Jerk, CoordSystem, BufferMode, TransitionMode,
TransitionParams );
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_MoveLinRel" (→ p. 853)

3.3.13.2.9 MC_MoveLinRel

PLCopen



Pipe Network



Function Block - Commands interpolated linear movement on an axes group to the specified relative positions.

- See Coordinated Motion, the top-level topic for Coordinated Motion.
- The dimensionality of the move is determined by the number of axes mapped to the group.
- When all motion has completed successfully, the axes group state is GroupStandby.
- MC_MoveLinRel commands interpolated linear movement of an axes group to the specified relative positions.

NOTE

- An error is returned if the group is in the GroupDisabled state.
- The maximum number of axes is set by the **MaxNumberOfAxes** input set in the "MC_CreateAxesGrp" (→ p. 764) function block.

NOTE

- Circular motion is only supported for axes groups with only two attached axes.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to perform a linear relative move.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axis group that performs the linear relative move.
Distance	LREAL[]	Dimension: The number of axes mapped in the AxesGroup. No range	User units	No default	Array of distances for each axis in the group.
Velocity	LREAL	$0 < \text{Velocity} < (20 * \text{Acceleration})$ and $0 < \text{Velocity} < (20 * \text{Deceleration})$	User units / sec	No default	• Maximum velocity of the defined path. • See Limitations on Acceleration and Jerk.
Acceleration	LREAL	$\text{Acceleration} > (\text{Velocity} / 20)$	User units / sec ²	No default	• Maximum acceleration. • See Limitations on Acceleration and Jerk.
Deceleration	LREAL	$\text{Deceleration} > (\text{Velocity} / 20)$	User units / sec ²	No default	• Maximum deceleration. • See Limitations on Acceleration and Jerk.

Input	Data Type	Range	Unit	Default	Description								
Jerk	LREAL	Trapezoidal velocity profiles: 0 (zero) S-Curve velocity profiles: Jerk > (Acceleration / 2) and Jerk > (Deceleration / 2)	User units / sec ³	No default	- Maximum jerk. - See Limitations on Acceleration and Jerk.								
CoordSystem	SINT	Enumerated	N/A	No default	The coordinate system used when commanding the linear relative move. - Only the ACS and MCS coordinate system are supported. - See Coordinate Systems. One of these enumeration values: <table><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>CS_ACS</td><td>Axes Coordinate System</td></tr><tr><td>CS_MCS</td><td>Machine Coordinate System</td></tr><tr><td>CS_PCS</td><td>Product Coordinate System</td></tr></tbody></table>	Value	Description	CS_ACS	Axes Coordinate System	CS_MCS	Machine Coordinate System	CS_PCS	Product Coordinate System
Value	Description												
CS_ACS	Axes Coordinate System												
CS_MCS	Machine Coordinate System												
CS_PCS	Product Coordinate System												
BufferMode	SINT	Enumerated	N/A	No default	Defines the chronological sequence of the function block relative to the previous block. - BM_ABORTING is not allowed with this function block. - See Buffer Modes. - Some individual axis velocities can make an abrupt change if the path of the next move travels in a different direction. - A transition move can be programmed in the TransitionMode input to avoid this.								

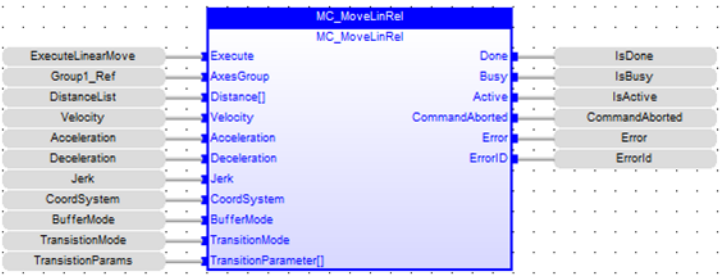
Input	Data Type	Range	Unit	Default	Description								
TransitionMode	SINT	Enumerated	N/A	No default	Coupled with the TransitionParameter, this input defines the shape and dynamics of the inserted contour to connect the current motion with the next motion in the queue. - The transition mode is limited to TM_NONE or TM_SCURVE_CORNER for groups with more than three axes. - See Transition Between Moves. One of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>TM_NONE</td><td>No transition.</td></tr><tr><td>TM_CORNER_DISTANCE</td><td>Corner distance transition.</td></tr><tr><td>TM_SCURVE_CORNER</td><td>S-Curve corner transition.</td></tr></table>	Value	Description	TM_NONE	No transition.	TM_CORNER_DISTANCE	Corner distance transition.	TM_SCURVE_CORNER	S-Curve corner transition.
Value	Description												
TM_NONE	No transition.												
TM_CORNER_DISTANCE	Corner distance transition.												
TM_SCURVE_CORNER	S-Curve corner transition.												
TransitionParameter	LREAL[]	Dimension: 1 to N No range	N/A	No default	This array is dependent on the specified TransitionMode. - N values are supplier specified dependent on the selected TransitionMode. - Transition parameter values are applied to the AxesGroup. - See Transition Between Moves.								

Outputs

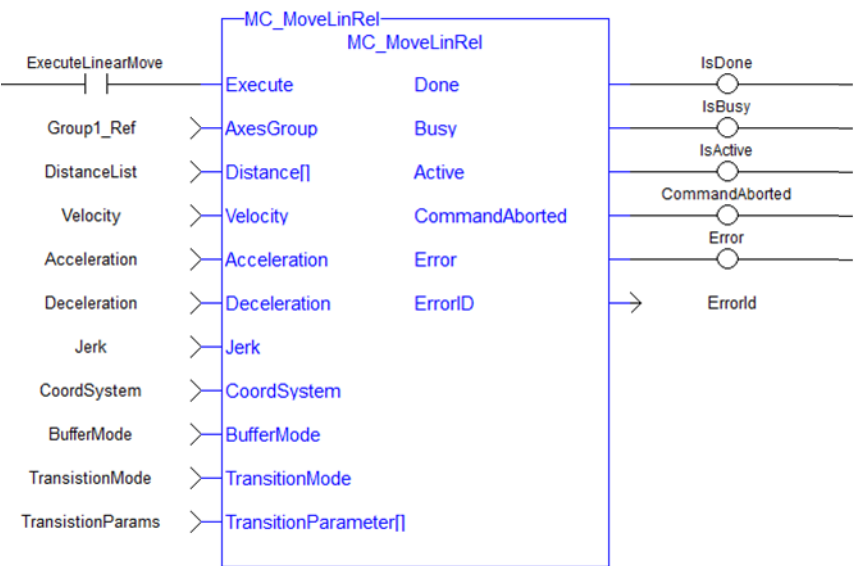
Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Active	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is controlling motion.
CommandAborted	BOOL	FALSE, TRUE	N/A	If TRUE, the command was aborted by another function block.

Output	Data Type	Range	Unit	Description
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
(* Inst_MC_MoveLinRelST example *)
Inst_MC_MoveLinRel( ExecuteLinearMove, Group1_Ref, DistanceList, Velocity,
Acceleration, Deceleration, Jerk, CoordSystem, BufferMode, TranstionMode,
TransitionParams );
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_MoveLinAbs" (→ p. 848)

3.3.13.3 Coordinated Motion - Reference

Name	Description
MC_ GrpSetPos	Sets the axis command position for all of the axes in an axes group to the positions specified in the <code>Position</code> input.

3.3.13.3.1 MC_GrpSetPos

PLCopen Pipe Network 

Function Block - Sets the axis command position for all of the axes in an axes group to the positions specified in the `Position` input.

NOTE

- This function block starts a motion-related action and stores data for calculations and error checking.
- If using a dual-core controller, see Program a Multi-Core Controller.
- This function or function block does not generate any motion.
- See Coordinated Motion, the top-level topic for Coordinated Motion.
- The axes group must be enabled and in Standby mode for MC_GrpSetPos to execute.
 - If it is not, this FB returns an error and the axis positions remains unchanged.
 - The command position is that returned by the FB **MC_GrpReadCmdPos**.

Inputs

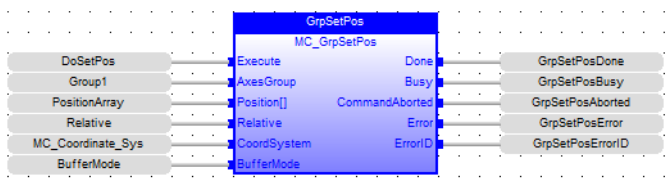
Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request to set the position of the group.
AxesGroup	AXES_GROUP_REF	No range	N/A	No default	The axis group to set the positions.
Position[]	LREAL	Dimension: The number of axes mapped in the AxesGroup. No range	N/A	No default	An array containing the position for each axis in the group. • If Relative is set, position represents a distance rather than an absolute position. • The length of the array must equal the maximum number of axes allowed in the group. • The maximum number of axes is an argument to MC_CreateAxesGrp, which is used to create axes groups.
Relative	BOOL	FALSE, TRUE	N/A	No default	Request to set position: • (1) relative. • (0) absolute.

Input	Data Type	Range	Unit	Default	Description								
CoordSystem	SINT	Enumerated	N/A	No default	The coordinate system used when setting the positions. - Only the ACS and MCS coordinate system are supported. - See Coordinate Systems. One of these enumeration values:<table><tr><th>Value</th><th>Description</th></tr><tr><td>CS_ACS</td><td>Axes Coordinate System</td></tr><tr><td>CS_MCS</td><td>Machine Coordinate System</td></tr><tr><td>CS_PCS</td><td>Product Coordinate System</td></tr></table>	Value	Description	CS_ACS	Axes Coordinate System	CS_MCS	Machine Coordinate System	CS_PCS	Product Coordinate System
Value	Description												
CS_ACS	Axes Coordinate System												
CS_MCS	Machine Coordinate System												
CS_PCS	Product Coordinate System												
BufferMode	SINT	0 to 0	N/A	No default	Unused.								

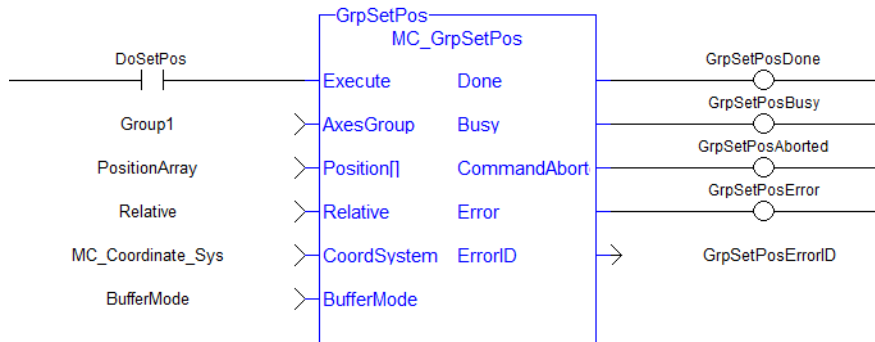
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	Returns FALSE.
CommandAborted	BOOL	FALSE, TRUE	N/A	Returns FALSE.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See PLCopen Function Block ErrorIDs.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_MC_GrpSetPos( DoSetPos, Group1, PositionArray, Relative, CS_ACS, 0 );
```

See Also

- "MC_ErrorDescription" (→ p. 863)
- "MC_GrpReadCmdPos" (→ p. 809)

3.3.14 Motion / Common - Info

Name	Description
MC_ErrorDescription	Converts the PLCopen error IDs into message strings.

3.3.14.1 MC_ErrorDescription

PLCopen ✓

Pipe Network ✓



Function - Converts the PLCopen error IDs into message strings.

- Return a text description corresponding to a motion control error ID code.
- The message strings can be used for display or logging.

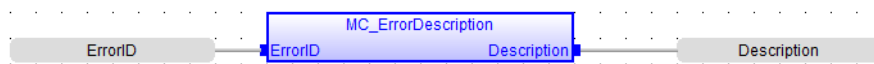
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	If TRUE, this function converts the ErrorID to a string message.
ErrorID	INT	0 to 69	N/A	No default	• ErrorID generated from a PLCopen function block. • See PLCopen Function Block ErrorIDs.

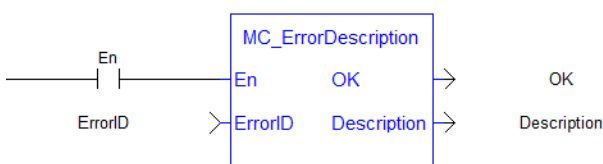
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Description	STRING	No range	N/A	String error description.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Description := MC_ErrorDescription(ErrorID);
```

3.3.15 Motion / Common - Motion

Name	Description
MLMotionCycleTime	Returns the Motion Base Cycle time in seconds.
MLMotionInit	Initializes the motion library.
MLMotionRstErr	Clears motion engine errors, motion bus driver errors, and EtherCAT network errors.
MLMotionStart	Starts the motion engine, motion bus driver, clears the EtherCAT diagnostic registers of all nodes, and initializes EtherCAT network to operational mode.
MLMotionStatus	Returns the status of the motion engine 0 (zero).
MLMotionStop	Stops the motion bus driver, motion engine, and EtherCAT network operation, resulting in the EtherCAT network transitioning to the Init state.

3.3.15.1 State Machine

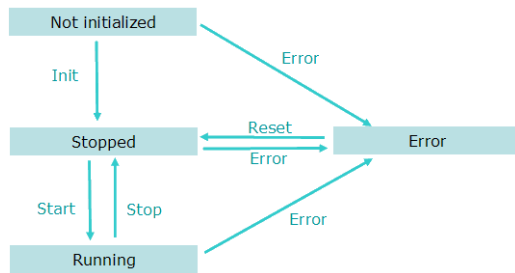


Figure 4-100: Motion State Machine

- The Motion State Machine is driven by the IEC 61131-3 application with the help of the "Motion / Common - Motion" (→ p. 864) function blocks.
- Each arrow represents a transition from one State to another one.

3.3.15.2 MLMotionCycleTime

PLCopen ✓

Pipe Network ✓

 **Function** - Returns the Motion Base Cycle time in seconds.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	Enable execution of the function.

Outputs

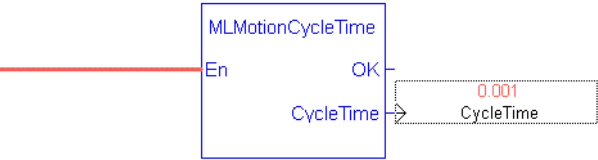
Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
CycleTime	LREAL	0.00025 to 0.004	Seconds	PLC cycle period in seconds.

FBD Language Example



FFLD Language Example

Network #1



End of Module

IL Language Example

Not available.

ST Language Example

```
//Read EtherCAT cycle rate in seconds
CycleTime := MLMotionCycleTime();
```

3.3.15.3 MLMotionInit

PLCopen ✓

Pipe Network ✓



Function - Initializes the motion library.

- Must be called before any other Motion Library function.
- Returns TRUE if the function succeeded.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	FALSE	Enable the function.
BasePeriod	LREAL	Minimum: 125	Microseconds	1000	Duration of one motion cycle. • With the EtherCAT Motion Bus: • The EtherCAT cycle time overrides the BasePeriod argument (the cycle time is defined in the EtherCAT Master Settings tab). • The EtherCAT cycle time becomes the base cycle time for the Motion Engine. • Without the EtherCAT Motion Bus: • The BasePeriod argument establishes the base cycle time (in microseconds) for the Motion Engine when running simulations

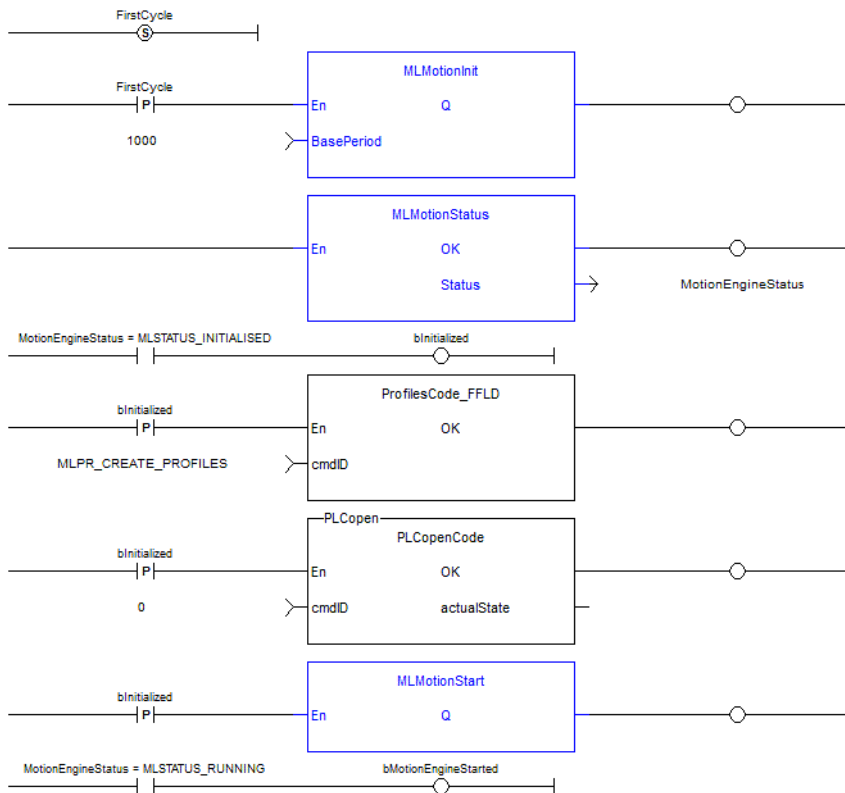
Outputs

Output	Data Type	Range	Unit	Description
Q	BOOL	FALSE, TRUE	N/A	• Output is set to high with initial successful execution. • It remains high until the function does NOT execute successfully.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Initialization code to start EtherCAT network.
//First initialize network with MLMotionInit command
//Then wait for command to finish by monitoring MLMotionStatus output
//Once initialized, create any cam profiles and PLCopen or Pipenetwork
devices
//Then call MLMotionStart and monitor MLMotionStatus again before beginning
rest of program
FirstCycle := TRUE;

On FirstCycle DO //Initialize the motion engine
    MLMotionInit( 1000);
END_DO;

MotionEngineStatus := MLMotionStatus();//Check the current status of the
motion engine
//Once motion engine is initialized, create CAM profiles and defined Axis,
then start the motion engine
ON MotionEngineStatus = MLSTATUS_INITIALISED DO
    Profiles( MLPR_CREATE_PROFILES );
    PLCopen( 0 );
    MLMotionStart();
```

```
END_DO;  
  
IF MotionEngineStatus = MLSTATUS_RUNNING THEN  
    bMotionEngineStarted := TRUE;  
ELSE  
    bMotionEngineStarted := FALSE;  
END_IF;
```

3.3.15.4 MLMotionRstErr

PLCopen ✓

Pipe Network ✓

 **Function** - Clears motion engine errors, motion bus driver errors, and EtherCAT network errors.

- Re-initializes the motion engine after a motion error.
- **MLMotionRstErr** returns the motion engine status to the Stopped state if an error condition was cleared successfully.
- Motion errors are for example communication errors of the motion bus.
- Returns TRUE if the function succeeded.

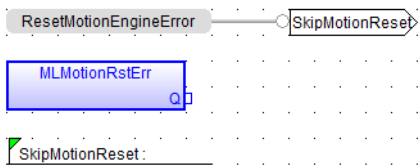
Inputs

There are no Inputs for this function / function block.

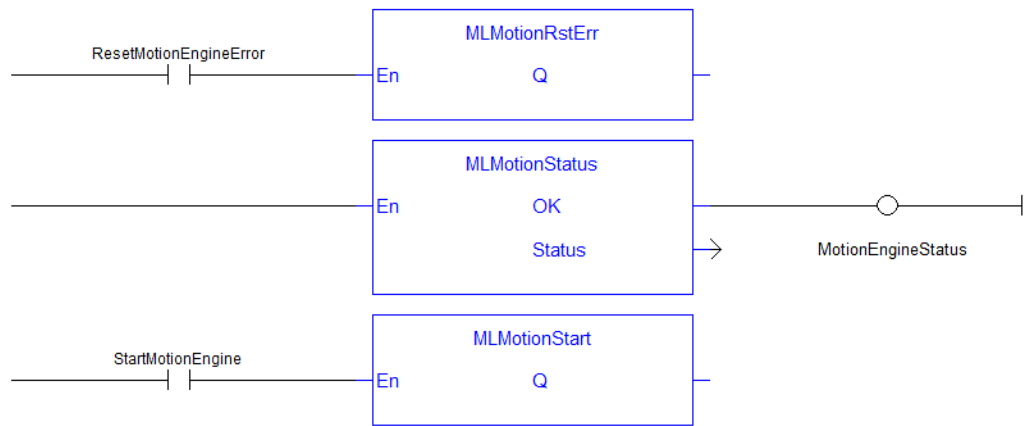
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Returns TRUE if errors have been reset.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Reset and restart motion engine
//Done to restart EtherCAT after controller error such as
//E30 or E33 that stops network communication
//First have to reset error, then start network again

ON ResetMotionEngineError DO
  MLMotionRstErr();
END_DO;

MotionEngineStatus:= MLMotionStatus();

ON StartMotionEngine DO
  MLMotionStart();
END_DO;
```

See Also

- "MLMotionStart" (→ p. 872)
- "MLMotionStatus" (→ p. 875)
- "MLMotionStop" (→ p. 878)

3.3.15.5 MLMotionStart

PLCopen ✓

Pipe Network ✓

 **Function** - Starts the motion engine, motion bus driver, clears the EtherCAT diagnostic registers of all nodes, and initializes EtherCAT network to operational mode.

- Applicable to PLCopen and Pipe Network motion engines.
- MLMotionStart does not clear any pre-existing error conditions.
- Returns TRUE if the function succeeded.
- Returns FALSE if the motion engine is in the Error state.

NOTE

- This function returns FALSE when the number of axes created in the application is greater than the maximum number permitted for the controller model.
 - When this axis limit is exceeded, the controller generates error E45.
 - This error is shown on the 7-segment display and on the Diagnostics tab of the Web server.
 - See E45 for an explanation and recommended corrective actions.
- The maximum axis count specified for each controller model applies only to physical axes.
 - This includes both PLCopen and Pipe Network axes.
- This limit **does not** apply to virtual axis or digitizing axis.

Inputs

There are no Inputs for this function / function block.

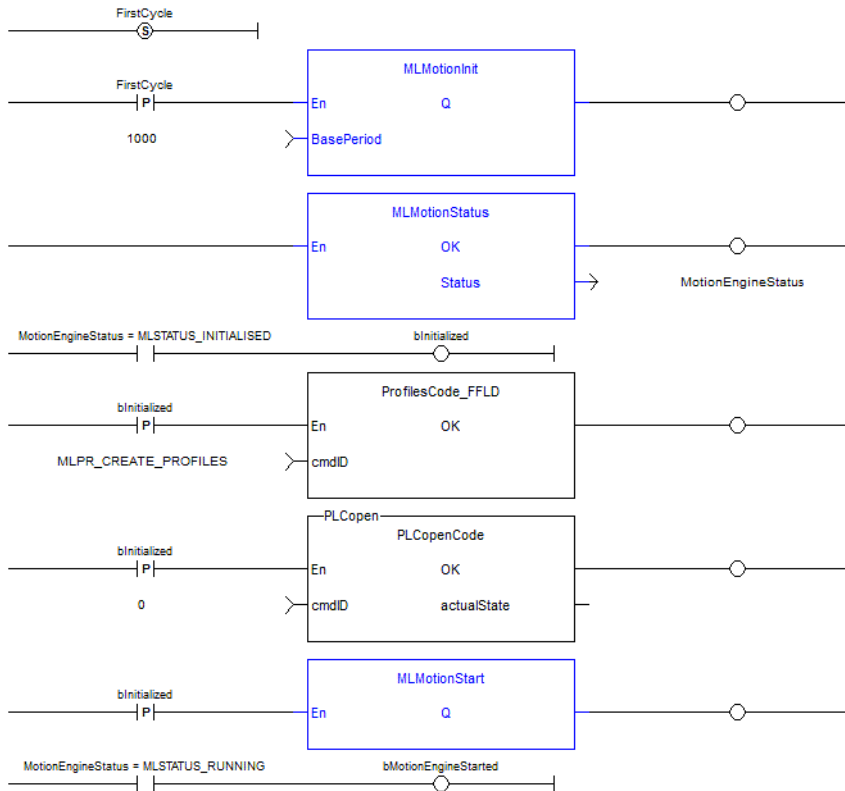
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	• Returns TRUE if all bus drivers have successfully started. •

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Initialization code to start EtherCAT network.
//First initialize network with MLMotionInit command
//Then wait for command to finish by monitoring MLMotionStatus output
//Once initialized, create any cam profiles and PLCopen or Pipenetwork
devices
//Then call MLMotionStart and monitor MLMotionStatus again before beginning
rest of program
FirstCycle := TRUE;

On FirstCycle DO //Initialize the motion engine
    MLMotionInit( 1000);
END_DO;

MotionEngineStatus := MLMotionStatus();//Check the current status of the
motion engine
//Once motion engine is initialized, create CAM profiles and defined Axis,
then start the motion engine
ON MotionEngineStatus = MLSTATUS_INITIALISED DO
    Profiles( MLPR_CREATE_PROFILES );
    PLCopen( 0 );
    MLMotionStart();
```



```
END_DO;  
  
IF MotionEngineStatus = MLSTATUS_RUNNING THEN  
    bMotionEngineStarted := TRUE;  
ELSE  
    bMotionEngineStarted := FALSE;  
END_IF;
```

See Also

- "MLMotionRstErr" (→ p. 870)
- "MLMotionStatus" (→ p. 875)
- "MLMotionStop" (→ p. 878)

3.3.15.6 MLMotionStatus

PLCopen ✓

Pipe Network ✓



Function - Returns the status of the motion engine 0 (zero).

NOTE

- This function or function block returns cached data.
- See Program a Multi-Core Controller.

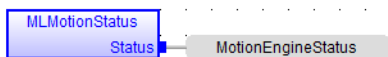
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	FALSE, TRUE	N/A	No default	Enable the function.

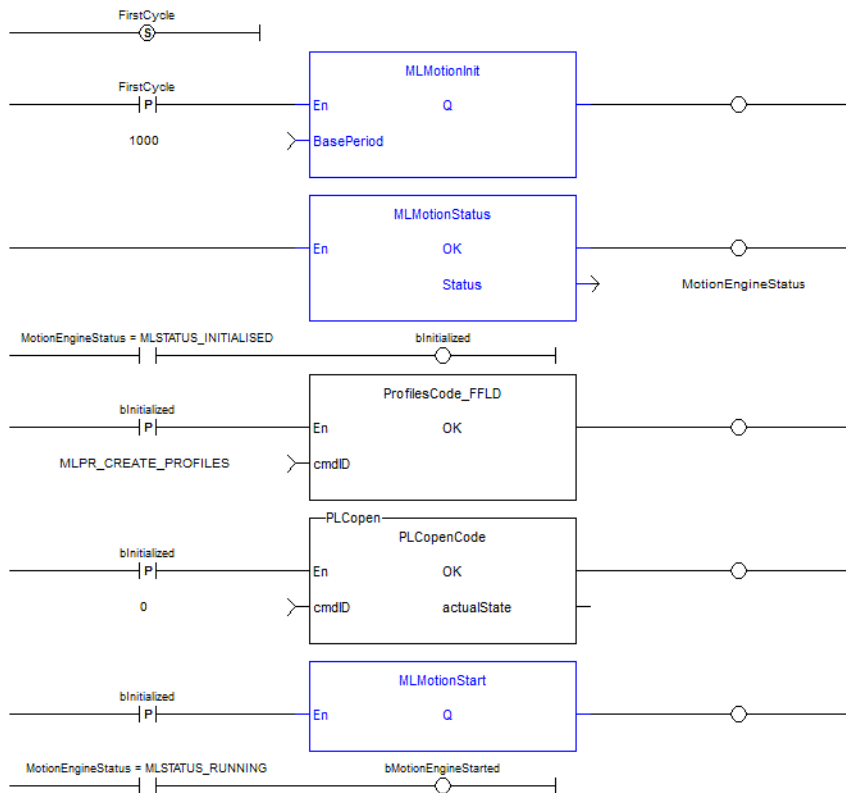
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Indicates the initialization is complete. •
Status	DINT	0 to 3	N/A	Status of the Motion Engine. Based on the Internal Defines, the status is one of these: • #define MLSTATUS_NOT_INITIALISED 0 (*Motion not initialized*) • #define MLSTATUS_RUNNING 1 (*Motion is running*) • #define MLSTATUS_STOPPED 2 (*Motion is stopped*) • #define MLSTATUS_ERROR 3 (*Motion is in error*)

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Initialization code to start EtherCAT network.
//First initialize network with MLMotionInit command
//Then wait for command to finish by monitoring MLMotionStatus output
//Once initialized, create any cam profiles and PLCopen or Pipenetwork
devices
//Then call MLMotionStart and monitor MLMotionStatus again before beginning
rest of program
FirstCycle := TRUE;

On FirstCycle DO //Initialize the motion engine
MLMotionInit( 1000);
END_DO;

MotionEngineStatus := MLMotionStatus();//Check the current status of the
motion engine
//Once motion engine is initialized, create CAM profiles and defined Axis,
then start the motion engine
ON MotionEngineStatus = MLSTATUS_INITIALISED DO
Profiles( MLPR_CREATE_PROFILES );
PLCopen( 0 );
MLMotionStart();
```

```
END_DO;  
  
IF MotionEngineStatus = MLSTATUS_RUNNING THEN  
  bMotionEngineStarted := TRUE;  
ELSE  
  bMotionEngineStarted := FALSE;  
END_IF;
```

See Also

- "MLMotionCycleTime" (→ p. 866)
- "MLMotionInit" (→ p. 867)
- "MLMotionRstErr" (→ p. 870)
- "MLMotionStart" (→ p. 872)
- "MLMotionStop" (→ p. 878)

3.3.15.7 MLMotionStop

PLCopen ✓

Pipe Network ✓

 **Function** - Stops the motion bus driver, motion engine, and EtherCAT network operation, resulting in the EtherCAT network transitioning to the Init state.

- Results in the EtherCAT network transitioning to the Init state.
- Returns TRUE if the function succeeded.

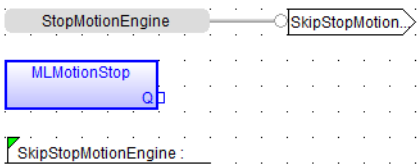
Inputs

There are no Inputs for this function / function block.

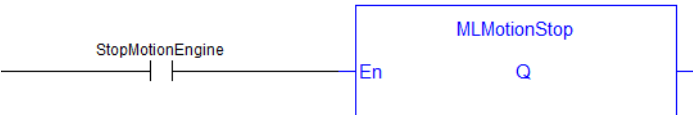
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Returns TRUE after requesting all bus drivers to stop.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Stop the EtherCAT network
ON StopMotionEngine DO
  MLMotionStop();
END_DO;
```

See Also

- "MLMotionCycleTime" (→ p. 866)
- "MLMotionInit" (→ p. 867)
- "MLMotionRstErr" (→ p. 870)

- "MLMotionStart" (→ p. 872)
- "MLMotionStatus" (→ p. 875)

3.3.16 Motion / Common - Profiles

Name	Description
MLProfileBuild	Creates a CAM Profile from application data that can be executed by a CAM block in Pipe Network or PLCopen.
MLProfileCreate	Creates a new CAM Profile Object for use in a PLC Program or Pipe Network CAM block.
MLProfileInit	Initializes a previously created CAM Profile object for use in a PLC Program or Pipe Network CAM block.
MLProfileRelease	Removes a Profile so the ProfileID can be used by a different or new Profile.

3.3.16.1 MLProfileBuild

PLCopen ✓

Pipe Network ✓



Function - Creates a CAM Profile from application data that can be executed by a CAM block in Pipe Network or PLCopen.

- This function takes input as CAM data and CAM Profile properties from application data memory and compile the input data to a form the controller can use to calculate cam positions.
 - See Profile Editor's Table for information.
- The input CAM data and CAM Profile properties are similar to the CAM data entered in the KAS-IDE's CAM Editor and the runtime's CAM Profile Properties dialog.

MLProfileBuild internally performs these functions:

1. Compile the CAM data (like the cam editor performs in the KAS-IDE).
2. Puts the compiled profile into the profile object so it can be used by other Profile Function Blocks. Provides similar functionality to "MLProfileInit" (→ p. 892).

NOTE

Prior to using **MLProfileBuild**, you must call "MLProfileCreate" (→ p. 890) to create the profile object. The ID output of **MLProfileCreate** is used as the `ProfileID` input to **MLProfileBuild**.

MLProfileCreate must be performed in the application **before** the "MLMotionStart" (→ p. 872) command is executed.

- **MLProfileBuild** compiles the CAM Profile data specified at the `CamData` input and writes the resulting profile to the CAM Profile object specified at input `ProfileID`.
- The created profile can be used as an input to PLCopenCAM Function Blocks (e.g., [MC_CamTblSelect](#), [MC_CamIn](#), [MC_CamOut](#)), or any Pipe NetworkCAM Profile Function/Function Blocks.
- When the operation is complete, the `Done` output is high.
 - If an error is encountered, the `Error` output is high and the `ErrorID` output returns an error code.
 - If the Error can be attributed to a specific profile element in the `CamData` array, `ErrorElem` attempts to indicate the element in error.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	• Enables execution. • Starts on the rising edge.
Cam_Props	CamProps_ref	No range	N/A	No default	Structures containing the CAM Profile properties.
Cam_Data	CamData_ref	N=3 to 20,000 elements	N/A	No default	Array of structures containing the CAM Profile data.
CamDataCount	UINT	3 to 20,000	Elements	No default	Number of elements in the <code>Cam_Data</code> array to be used.

Input	Data Type	Range	Unit	Default	Description
ProfileID	DINT	-2147483648 to 2147483647	N/A	No default	ID number of an initialized CAM Profile.
Cyclic	BOOL	FALSE, TRUE	N/A	No default	- FALSE: One time through the profile. - TRUE: Repeating profile.
Options	INT	- CAM_PROFILE_OPTION_DEFAULT: Allows the use of point and line segments. - CAM_PROFILE_OPTION_PARABOLIC: Allows the use of line and parabolic segments.	N/A	No default	Describes the combinations of segments that can be used to build a cam profile.
NOTE - The DEFAULT and PARABOLIC options cannot be combined. A CAM Profile can only use point and line segments, or parabolic and line segments. - Point and parabolic segments cannot both be used in the same profile.					

3.3.16.1.1

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	Indication of whether or not the profile was successfully compiled and built.
Busy	BOOL	FALSE, TRUE	N/A	Indication that the function is executing. - TRUE if executing. - FALSE if not executing.
Err	BOOL	FALSE, TRUE	N/A	- Indication that the function did not execute correctly. - ErrorID output is valid and indicate the reason.
ErrorID	INT	No range	N/A	- Indicates the reason for the failure. - See the "Error Codes" (→ p. 884) table.
ErrorElem	UINT	No range	N/A	The array element number of the CAM data where an error is detected.

Remarks

3.3.16.1.2.1 CamProps_Ref Structure

See Master/Input offset about the parameters which transform the CAM Profile.

CamProps_Ref Structure

The CAM properties structure (CamProps_Ref) contains these data members:

Parameter	Type	Description
InputScale	LREAL	The input amplitude or master axis multiplier applied to the CAM Profile.
OutputScale	LREAL	The output amplitude or slave axis multiplier applied to the CAM Profile.
InputOffset	LREAL	input offset or master axis shift applied to the CAM Profile.
OutputOffset	LREAL	The output offset or slave axis shift applied to the CAM Profile.

3.3.16.1.3.2 CamData_Ref Structure

The **Cam_Data** function block input is an array of **CamData_Ref** structures.

CamData_Ref Structure

Each element of the structure contains these data members:

Parameter	Type	Description
Accel	LREAL	CAM_SEGMENT_TYPE_POINT Accel represents the cam acceleration at the master position specified by MasterIn. - Units: (slave position user units) / (master position user units) CAM_SEGMENT_TYPE_LINE and CAM_SEGMENT_TYPE_PARABOLIC Accel is ignored.
MasterIn	LREAL	Master position (in the unit range [0 - InputScale]).
SegType	UINT	Defines the segment type for the segment following the master positions defined by MasterIn . - CAM_SEGMENT_TYPE_POINT = Point 5th order polynomial). - CAM_SEGMENT_TYPE_LINE = Line (constant velocity segment). - CAM_SEGMENT_TYPE_PARABOLIC = Parabolic (constant acceleration). See CAM Profile Segments about the segment types.
SlaveOut	LREAL	Slave position (in the unit range [0 - OutputScale]).
Vel	LREAL	CAM velocity at the master position specified by MasterIn . Units: (slave position user units) / (master position user units)

- The type of the Nth CAM segment is defined by the Nth Cam_Data element.
- The CAM is constructed with one less segment than the Cam_Data elements.
 - The last element's SegType is not used.
- See Profile Editor's Table.

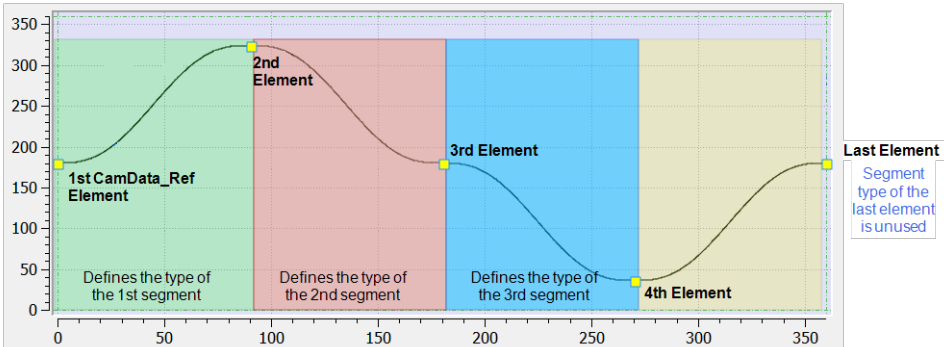


Figure 4-101: MLProfileBuild Cam Segments graph

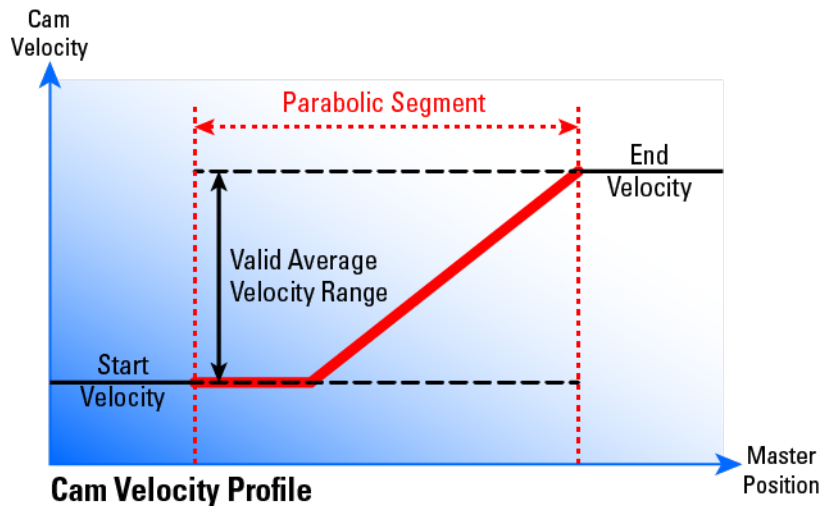
Error Codes

NOTE

If **Cyclic** is TRUE and the Vel/Accel of the first and last elements do not match, **MLProfileBuild** automatically copies the first element's Vel/Accel to the last element's. A log warning message is posted indicating that this change has occurred.

[Click to see the Error Codes list](#)

ErrorID	Description
100	Cam_Data array does not have CamDataCount elements. The Cam_Data array is not large enough to hold the specified number of elements.
101	Invalid master or slave scale. The scale cannot be less than zero.
102	Element master or slave position is outside the range defined by Cam_Props.
103	A segment type was specified that is not supported by the value of the Options argument.
104	Master position of an element is too close to the master position of a previous element. • 0.0000125 * InputScale is the minimum distance allowed. • See "CamProps_Ref Structure" (→ p. 883) for more information. • Each element is compared to all previous elements. • If the master distance between any two elements in the list are too close, this error is generated.
106	Invalid ProfileID. This can occur if the ProfileID: • Does not exist. • Has not been created yet. • The ProfileID is not a profile.
107	CamDataCount exceeds maximum array size of 20,000.
108	Profile cannot be released because it is in use by the motion engine or currently selected by an active CAM block.

ErrorID	Description
109	Attempting to build a profile already containing elements. Profile needs to be released first using <code>MLProfileRelease</code> .
110	The controller is running low on memory and could not allocate memory to hold the cam table data.
111	<code>CamDataCount</code> is not large enough. The minimum allowed value is 3.
112	CAM_PROFILE_OPTION_PARABOLIC - Elements are not sorted in increasing order by master position. - After the first element, each element must have its master position be greater than the master position of the previous element.
113	CAM_SEGMENT_TYPE_PARABOLIC - The average velocity for the segment is outside of range defined by start and end element velocities. - This diagram illustrates the valid range for the average velocity:  Cam Velocity Profile Figure 4-102: MLProfileBuild Cam Velocity Profile graph
200	First element's <code>MasterIn</code> value not equal to zero.
201	Last element's <code>MasterIn</code> value does not equal value of X-amplitude.
202	Cannot modify the first element in the cam element table. <code>SlaveOut</code> value is outside the output range specified by <code>Cam_Props</code> .
203	Cannot modify the last element in the cam element table. <code>SlaveOut</code> value is outside the output range specified by <code>Cam_Props</code> .

3.3.16.1.4.3 Example: Use `MLProfileBuild`

This example has these procedure steps:

- "1. Create a Profile" (→ p. 886)
- "2. Define the Profile Data" (→ p. 886)
- "3. Define the Profile Properties" (→ p. 886)
- "4. Call the `MLProfileBuild` Function Block" (→ p. 886)
- "5. Use the New Profile" (→ p. 887)

MLProfileBuild Procedures

3.3.16.1.5.4 1. Create a Profile**ⓘ IMPORTANT**

Prior to using **MLProfileBuild**, you must first create a profile.
This must be done prior to **MLMotionStart**.

```
// Allocate space for a profile that will be built later

profileID := MLProfileCreate('ProfileName');
```

3.3.16.1.6.5 2. Define the Profile Data

1. Create an array of CamData_Ref structures in the data dictionary.
2. Enter each of the elements into that newly created structure.
In this example, ProfileData is the name of the CamData_Ref structure.

```
// Define the profile data
ProfileData[0].MasterIn := 0.0;
ProfileData[0].SlaveOut := 180.0;
ProfileData[0].SegType := CAM_SEGMENT_TYPE_POINT;
ProfileData[0].Velocity := 0.0;
ProfileData[0].Acceleration := 0.0;

ProfileData[1].MasterIn := 180.0;
ProfileData[1].SlaveOut := 324.0;
ProfileData[1].SegType := CAM_SEGMENT_TYPE_POINT;
ProfileData[1].Velocity := 0.5;
ProfileData[1].Acceleration := -0.025;

ProfileData[2].MasterIn := 360.0;
ProfileData[2].SlaveOut := 240.0;
ProfileData[2].SegType := CAM_SEGMENT_TYPE_POINT;
ProfileData[2].Velocity := 0.0;
ProfileData[2].Acceleration := 0.0;
```

3.3.16.1.7.6 3. Define the Profile Properties

1. Create a CamProps_Ref structure in the data dictionary.
2. Enter each of the properties into the newly created structure.
In this example, ProfileProps is the name of the CamProps_Ref structure.

```
// Define the profile properties
ProfileProps.InputScale := 360.0; // Must be Positive!
ProfileProps.OutputScale := 360.0; // Must be Positive!
ProfileProps.InputOffset := 0.0;
ProfileProps.OutputOffset := 0.0;
```

3.3.16.1.8.7 4. Call the MLProfileBuild Function Block

Use the IEC language of choice.

As part of this call it is recommended that you validate the Done and Error output before proceeding.

```
// Build the profile
Inst_MLProfileBuild( TRUE, ProfileProps, ProfileData, 3, ProfileID, TRUE,
CAM_PROFILE_OPTION_DEFAULT);
```

3.3.16.1.9.8 5. Use the New Profile

After verifying that **MLProfileBuild** is Done and there are no errors, you can use the newly generated profile.

The next step depends on the motion engine in use.

- PLCopen: call "MC_CamTblSelect" (→ p. 707).
- Pipe Network: call either "MLCamInit" (→ p. 411) or "MLCamSwitch" (→ p. 413).

❗IMPORTANT

Pipe Network: To correctly set the CAM scales and offsets (defined by the Cam_Props argument) "MLPrfWriteIScale" (→ p. 425), "MLPrfWriteOScale" (→ p. 429), "MLPrfWriteIOffset" (→ p. 423) and "MLPrfWriteOOffset" (→ p. 427) must be called before calling "MLCamSwitch" (→ p. 413).

```
// Switch Pipe Network Profile
MLPrfWriteIScale(profileID, ProfileProps.InputScale);
MLPrfWriteOScale(profileID, ProfileProps.OutputScale);
MLPrfWriteIOffset(profileID, ProfileProps.InputOffset);
MLPrfWriteOOffset(profileID, ProfileProps.OutputOffset);
MLCamSwitch(PipeNetwork.CAM, profileID);
```

3.3.16.1.10.9 Example: Building a Parabolic Cam Profile

To build a parabolic CAM Profile, your CAM data elements must use **CAM_SEGMENT_TYPE_PARABOLIC** or **CAM_SEGMENT_TYPE_LINE** when defining the cam data array.

Example Code: Building a Parabolic Cam Profile

```
// Define the profile data
ProfileData[0].MasterIn      := 0.0;
ProfileData[0].SlaveOut     := 0.0;
ProfileData[0].SegType      := CAM_SEGMENT_TYPE_PARABOLIC;
ProfileData[0].Velocity     := 0.0;
ProfileData[0].Acceleration := 0.5;

ProfileData[1].MasterIn      := 50.0;
ProfileData[1].SlaveOut     := 150.0;
ProfileData[1].SegType      := CAM_SEGMENT_TYPE_LINE;
ProfileData[1].Velocity     := 5.0;
ProfileData[1].Acceleration := 0.0;           // Not used

ProfileData[2].MasterIn      := 55.0;
ProfileData[2].SlaveOut     := 175.0;
ProfileData[2].SegType      := CAM_SEGMENT_TYPE_PARABOLIC;
ProfileData[2].Velocity     := 5.0;
ProfileData[2].Acceleration := 0.0;           // No limit to the acceleration rate
of the segment.
```

```
ProfileData[3].MasterIn      := 105.0;
ProfileData[3].SlaveOut     := 250.0;
ProfileData[3].SegType      := CAM_SEGMENT_TYPE_PARABOLIC;
ProfileData[3].Velocity     := 0.0;
ProfileData[3].Acceleration := 0.5;

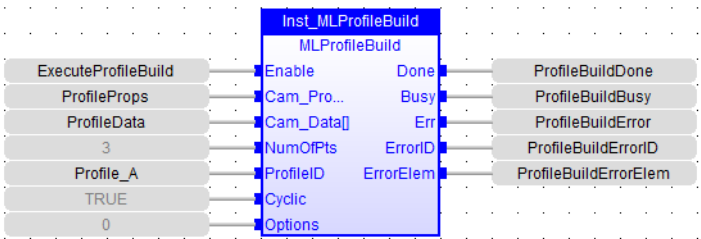
ProfileData[4].MasterIn      := 225.0;
ProfileData[4].SlaveOut     := 125.0;
ProfileData[4].SegType      := CAM_SEGMENT_TYPE_PARABOLIC;
ProfileData[4].Velocity     := -10.0;
ProfileData[4].Acceleration := 0.5;

ProfileData[5].MasterIn      := 360.0;
ProfileData[5].SlaveOut     := 0.0;
ProfileData[5].SegType      := CAM_SEGMENT_TYPE_PARABOLIC;      // Not used
ProfileData[5].Velocity     := 0.0;
ProfileData[5].Acceleration := 0.0;                          // Not used
```

- When calling the MLProfileBuild function block, verify CAM_PROFILE_OPTION_PARABOLIC is specified for the Option argument: in the IEC language of choice.
- As part of this call, it is recommended that you validate the **Done** and **Error** output before proceeding.

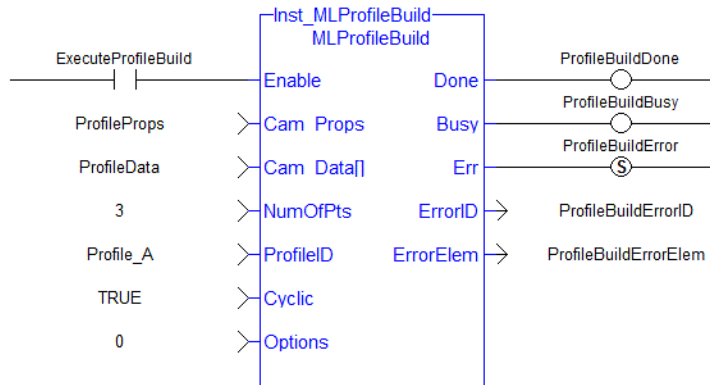
```
// Build the profile
Inst_MLProfileBuild( TRUE, ProfileProps, ProfileData, 3, ProfileID, TRUE,
CAM_PROFILE_OPTION_PARABOLIC );
```

FBD Language Example



FFLD Language Example

Network #2



IL Language Example

Not available.

ST Language Example

```
// Build the profile
Inst_MLProfileBuild( TRUE, ProfileProps, ProfileData, 3, ProfileID, TRUE,
CAM_PROFILE_OPTION_DEFAULT);
```

See Also

- CAM Profile Segments
- "MC_CamIn" (→ p. 693)
- "MC_CamOut" (→ p. 700)
- [MC_CamTblSelect](#)
- "MLCamInit" (→ p. 411)
- "MLCamSwitch" (→ p. 413)
- "MLProfileCreate" (→ p. 890)
- "MLProfileInit" (→ p. 892)
- "MLProfileRelease" (→ p. 894)

3.3.16.2 MLProfileCreate

PLCopen ✓

Pipe Network ✓

 **Function** - Creates a new CAM Profile Object for use in a PLC Program or Pipe Network CAM block.

This function is automatically called if a Profile is created in the Project Explorer with user-defined settings entered in the CAM Profile Properties screen.

- Profiles are created and initiated separately.
- The shape is modified with the CAM Editor.
 - With the Editor, profiles can be changed graphically or by manually changing values in a numeric table relating input and output values with specific slopes.
 - The Cam Editor software tool provides the capability to visualize, analyze, edit, and smooth profiles.
- Profile switching can be done on the fly without losing synchronization and without dead time.
 - Offsets and ratios of CAM Profiles can be changed on the fly.

NOTE

- Profile objects are normally created in the Project Explorer.
- You do not have to add **MLCamInit** function blocks to their programs.
- Right click the Profiles folder under PLC->Motion and click Add to create a new profile.
- Parameters are entered directly in a window and the code is automatically added to the current project.

 TIP

This function should be called after "MLMotionInit" (→ p. 867) is called and before "MLMotionStart" (→ p. 872) is called.

Inputs

Input	Data Type	Range	Unit	Default	Description
Name	STRING	No range	N/A	No default	Name of initialized CAM Profile

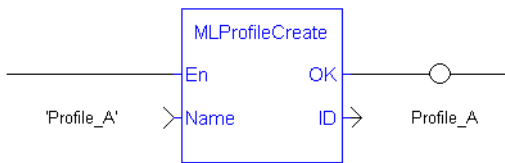
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Indicates the profile has been created. •
ID	DINT	No range	N/A	Returns the ID number of the created CAM Profile. • If MLProfileCreate(...) fails, the ID is zero (NULL). • A CAM ProfileID = 0 (zero) is not valid for cam profile functions.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Create a new Profile
Profile_A := MLProfileCreate( 'Profile_A' );
```

See Also

- "MLProfileInit" (→ p. 892)
- "MLCamInit" (→ p. 411)

3.3.16.3 MLProfileInit

PLCopen 

Pipe Network 



Function - Initializes a previously created CAM Profile object for use in a PLC Program or Pipe Network CAM block.

This function is automatically called if a Profile is created in the Project Explorer with user-defined settings entered in the CAM Profile Properties screen.

- Profiles are created and initiated separately.
- The shape is modified with the CAM Editor.
 - With the Editor, profiles can be changed graphically or by manually changing values in a numeric table relating input and output values with specific slopes.
 - The Cam Editor software tool provides the capability to visualize, analyze, edit, and smooth profiles.
- Profile switching can be done on the fly without losing synchronization and without dead time.
 - Offsets and ratios of CAM Profiles can be changed on the fly.

NOTE

- Profile objects are normally created in the Project Explorer.
- You do not have to add **MLCamInit** function blocks to their programs.
- Right click the Profiles folder under PLC->Motion and click Add to create a new profile.
- Parameters are entered directly in a window and the code is automatically added to the current project.

ⓘIMPORTANT

- Loading a Profile Editor-generated profile into a ProfileID released by **MLProfileRelease** should be done with care.
- The **MLProfileInit ()** function call can take in excess of 4 milliseconds to execute.
- Application execution is suspended during this time until the function call is completed.

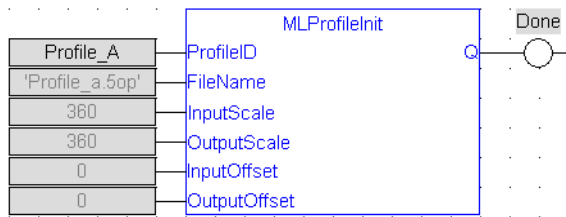
Inputs

Input	Data Type	Range	Unit	Default	Description
ProfileID	DINT	0 to 2147483647	N/A	No default	ID number of a created CAM Profile.
FileName	STRING	No range	N/A	No default	Filename used to save Profile on the computer's hard disk.
InputScale	LREAL	Positive	N/A	No default	The input amplitude or X-axis multiplier applied to the CAM Profile.
OutputScale	LREAL	No range	N/A	No default	The output amplitude or Y-axis multiplier applied to the CAM Profile.
InputOffset	LREAL	No range	N/A	No default	The input offset or X-axis shift applied to the CAM Profile.
OutputOffset	LREAL	No range	N/A	No default	The output offset or Y-axis shift applied to the CAM Profile.

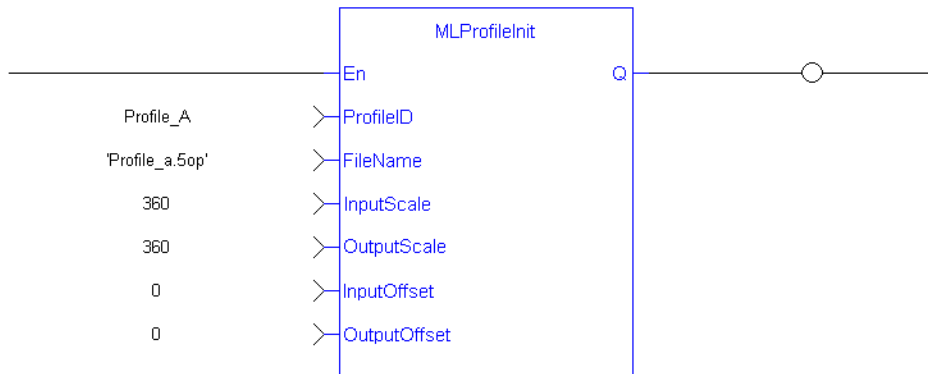
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	Returns TRUE if a new CAM Profile is initialized.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Initialize a previously created CAM Profile
MLProfileCreate( Profile_A , 'Profile_A.5op' , 360, 360, 0, 0 );
```

See Also

- "MLProfileCreate" (→ p. 890)
- "MLCamInit" (→ p. 411)

3.3.16.4 MLProfileRelease

PLCopen 

Pipe Network 



Function Block - Removes a Profile so the ProfileID can be used by a different or new Profile.

-
- An application program is limited to 256 ProfileIDs.
- Once the existing ProfileID definition has been successfully released, the ProfileID can be used by either "MLProfileInit" (→ p. 892) or "MLProfileBuild" (→ p. 881) to create a new Profile.
- The ProfileID selected by the input parameter must not be in-use by a motion engine.

Inputs

Input	Data Type	Range	Unit	Default	Description
Enable	BOOL	FALSE, TRUE	N/A	No default	Enables execution. • Successful completion results in a profile ID no longer assigned to a specific profile and can be reused for a different/new Profile. • Prior to reusing this Profile ID, it must be re-initialized by either an MLProfileInit function call or by calling MLProfileBuild.
ProfileID	DINT	1 to 256	N/A	No default	Specify a Profile ID that has been created by "MLProfileCreate" (→ p. 890). • This is the profile ID released so it can be reused for a different or new Profile. • This Profile ID must not be in use by a motion engine.

3.3.16.4.1

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	• If TRUE, the command completed successfully. • The Profile can now be reused.
Err	BOOL	FALSE, TRUE	N/A	• If TRUE, the function block did not complete successfully. • Reason is given in ErrorID.
ErrorID	INT	No range	N/A	• Indicates the reason for the failure. • See the "Error Codes" (→ p. 895) table.

Remarks

3.3.16.4.2.1 In-use Definitions

- Pipe Network - Must not be currently selected for use by an active CAM block in an active pipe.
 - Pipe has been activated by "MLCamSwitch" (→ p. 413).
- PLCopen - Selected for use by "MC_CamIn" (→ p. 693) and has an active move.

There are a number of ways to change an in-use profile to one that is not in-use (deactivated):

- Pipe Network - Perform a "MLCamSwitch" (→ p. 413) on an active Pipe to a different Profile or deactivate the pipe.
- PLCopen - Whenever the active profile move is halted or aborted, the profile is no longer in use.
 - "MC_CamOut" (→ p. 700) is one way of aborting the profile move.
 - Any PLCopen motion command that aborts a profile move deactivates a profile.

NOTE

- Any ProfileID created by "MC_CamTblSelect" (→ p. 707) from the specified ProfileID is destroyed and must be recreated when this function block is completed.
- All derived ProfileID's created by the **MC_CamTblSelect** function block **must not** be in use by the PLCopen motion engine for this function to succeed.

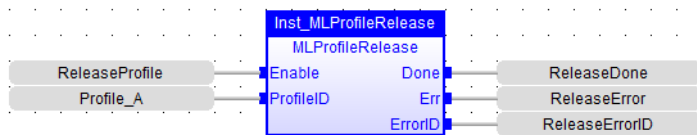
IMPORTANT

- Loading a Profile Editor-generated profile into a ProfileID released by **MLProfileRelease** should be done with care.
- The **MLProfileInit ()** function call can take in excess of 4 milliseconds to execute.
- Application execution is suspended during this time until the function call is completed.

Error Codes

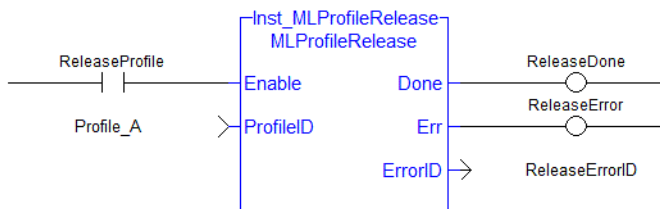
ErrorID	Description
106	Invalid ProfileID. This can occur if the ProfileID: • Does not exist. • Has not been created yet. • The ProfileID is not a profile.
108	Profile cannot be released because it is in use by the motion engine or currently selected by an active CAM block.

FBD Language Example



FFLD Language Example

Network #2



IL Language Example

Not available.

ST Language Example

```
//Release a Cam Profile
Inst_MLProfileRelease( Profile_A , 'Profile_A.5op');

If Inst_MLProfileRelease.Done THEN
    // Do Something
ELSIF Inst_MLProfileRelease.Err THEN
    // Handle Error
END_IF;
```

See Also

- "MC_CamIn" (→ p. 693)
- "MC_CamOut" (→ p. 700)
- "MC_CamTblSelect" (→ p. 707)
- "MLCamInit" (→ p. 411)
- "MLProfileBuild" (→ p. 881)
- "MLProfileCreate" (→ p. 890)
- "MLProfileInit" (→ p. 892)

4 System Library

These tables list the System functions and function blocks:

- "Controller" (→ p. 897)
- "File" (→ p. 897)
- "Profiling" (→ p. 898)
- "Recipe Function Blocks" (→ p. 898)
- "TCP/IP Sockets" (→ p. 898)
- "UDP" (→ p. 898)

See "PrintMessage" (→ p. 1002) - Generates a log message with any wanted strings in the Log Messages window.

4.1 Controller

Name	Description
ClearCtrlErrors	Clears the list of active errors and alarms on the controller.
CtrlReadBoolPar	Reads an Axis Boolean property of a Pipe Network Axis.
GetCtrlErrors	Get a list of the active errors and alarms on the controller.
GetCtrlInfo	Get the serial, model, and/or part number of the controller.
GetCtrlPerf	Returns controller CPU performance statistics.

4.2 File

Name	Description
FileClose	Closes an open file.
FileCopy	Copies a file's contents to a new file.
FileDelete	Removes a file from the file system.
FileEOF	Test if the end of the file is reached in a file that is open for reading.
FileExists	Tests if a file exists.
FileOpenA	Create or open a file in append mode.
FileOpenR	Open a file for reading.
FileOpenW	Create or reset a file and open it for writing.
FileReadBinData	Read binary data from a file.
FileReadLine	Reads a string value from a text file.
FileRename	Renames a file.
FileSeek	Sets the current position in an open file.
FileSize	Gets the size of a file.
FileWriteBinData	Write binary data to a file.
FileWriteLine	Writes a string value to a text file.

4.3 Profiling

Name	Description
Stopwatch	Measures the elapsed time between two events and gathers basic statistics when measurements are repeated.

4.4 Recipe Function Blocks

Name	Description
RecipeApplyInstance	Copy values to from a recipe instance to the program data.
RecipeClose	Close a recipe object.
RecipeOpen	Open a recipe file and create a recipe object.
RecipeQuickLoad	Load a recipe file and immediately apply the items it contains.
RecipeSave	Save a recipe object to a file.
RecipeSaveRetainVars	Save all retain variables, and their values, into a new recipe file.
RecipeStoreInstance	Store the present values of the program data in a recipe instance.

4.5 TCP/IP Sockets

Name	Description
TcpAccept	Performs the accept operation.
TcpBinReceive	Receives characters over a socket connection.
TcpBinSend	Sends characters over a socket.
TcpClose	Closes and releases a socket.
TcpConnect	Creates a new socket and performs the connect operation.
TcpIsConnected	Tests if a client socket is connected.
TcpIsValid	Tests if a socket is valid.
TcpListen	Creates a new socket by performing the bind and listen operations.
TcpReceive	Receives characters over a socket connection.
TcpSend	Sends characters over a socket.

4.6 UDP

Name	Description
udpAddrMake	Build an address buffer for UDP functions.
udpClose	Closes a socket.
udpCreate	Creates a UDP socket.
udpIsValid	Tests if a socket is valid.
udpRcvFrom	Receives a UDP telegram.
udpRcvFromArray	Receives a byte array through UDP.

Name	Description
udpRcvFromVar	Receives the contents of a variable from another controller through UDP.
udpSendTo	Sends a UDP telegram.
udpSendToArray	Sends a byte array through UDP.
udpSendToVar	Sends the contents of a local variable to another controller through UDP.

4.7 Controller Functions

Name	Description
ClearCtrlErrors	Clears the list of active errors and alarms on the controller.
CtrlReadBoolPar	Reads an Axis Boolean property of a Pipe Network Axis.
GetCtrlErrors	Get a list of the active errors and alarms on the controller.
GetCtrlInfo	Get the serial, model, and/or part number of the controller.
GetCtrlPerf	Returns controller CPU performance statistics.

4.7.1 ClearCtrlErrors

PLCopen ✓

Pipe Network ✓



Function - Clears the list of active errors and alarms on the controller.

- Only clearable errors are cleared.
- See Errors for a list of errors and alarms that may be generated.

NOTE

If clearable and non-clearable errors are present and this function is called, the Output Q is turned to TRUE but the non-clearable errors remain.

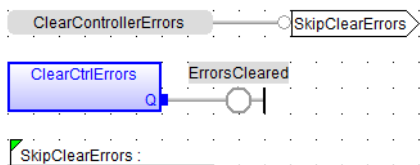
Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enable the function.

Outputs

Output	Data Type	Range	Unit	Description
Q	BOOL	FALSE, TRUE	N/A	Returns TRUE if the function succeeded.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Attempt to clear active controller level errors and alarms (such as E33)
IF ClearControllerErrors THEN
    ClearCtrlErrors();
END_IF;
```

4.7.2 CtrlReadBoolPar

PLCopen

Pipe Network

This function/function block was added in KAS v4.05.

 **Function Block** - Reads an Axis Boolean property of a Pipe Network Axis.

Inputs

Input	Data Type	Range	Unit	Default	Description
ParameterNumber	INT	Enumerated	N/A	No default	See "MLAxisReadBoolParam" (→ p. 363).

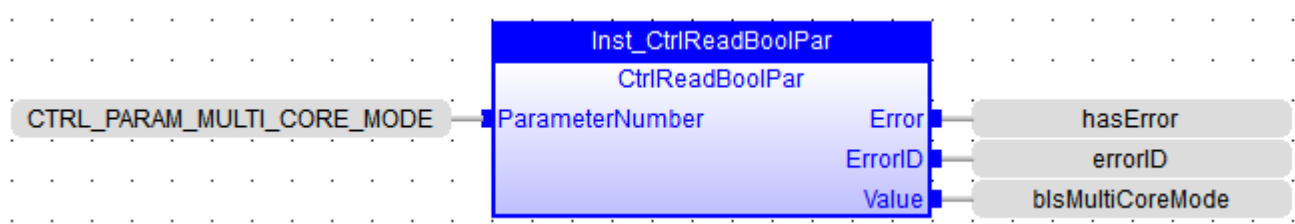
Outputs

Output	Data Type	Range	Unit	Description
Error	BOOL	FALSE, TRUE	N/A	If TRUE, indicates an invalid input was specified.
ErrorID	INT	Enumerated	N/A	- Indicates the error if Error output is TRUE.
Value	BOOL	FALSE, TRUE	N/A	Value read.

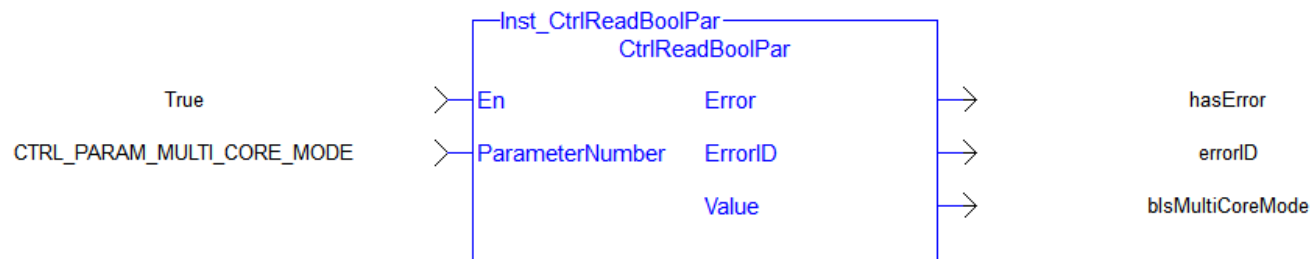
Controller Parameters

ID	Parameter	Name	R/W	Update Rate Type	Description
(missing or bad snippet)	CTRL_PARAM_MULTI_CORE_MODE	Controller Multicore Mode	Read-only	Static	- If TRUE, indicates the controller is running in multi-core mode. - If FALSE, indicates the controller is a single-core controller or running in single-core compatibility mode.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_CtrlReadBoolPar(CTRL_PARAM_MULTI_CORE_MODE);  
IF NOT Inst_CtrlReadBoolPar.Error THEN  
    bIsMultiCoreMode := Inst_CtrlReadBoolPar.Value;  
END_IF;
```

See Also

"MLAxisWriteBoolParam" (→ p. 394)

4.7.3 GetCtrlErrors

PLCopen ✓

Pipe Network ✓

 **Function** - Get a list of the active errors and alarms on the controller.

- Returns active errors and alarms on the controller in two arrays of 100 Booleans.
- Every index in the array corresponds to the error and alarm numbers in the tables.
 - See Controller Errors and Alarms for a list of possible generated errors and alarms.

 **TIP**

See Check for existing EtherCAT Alarms and Errors in the Example: EtherCAT Communication Diagnosis Steps section for an example of implementing this function.

Inputs

Input	Data Type	Range	Unit	Default	Description
EN	BOOL	FALSE, TRUE	N/A	No default	Enable the function.
ActiveError	BOOL [100]	Dimension: 0 to 99 Elements: FALSE, TRUE	N/A	No default	Array of BOOL with the size equal to 100.
ActiveAlarm	BOOL [100]	Dimension: 0 to 99 Elements: FALSE, TRUE	N/A	No default	Array of BOOL with the size equal to 100.

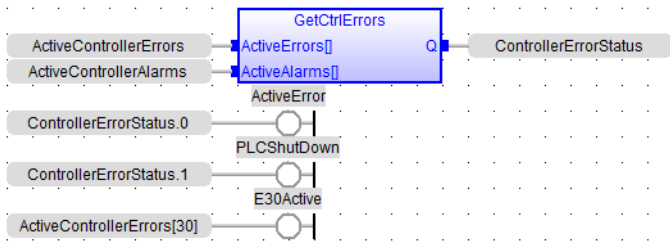
Outputs

Output	Data Type	Range	Unit	Description
Q	DINT	Enumerated	N/A	See "Status Meaning" (→ p. 904).

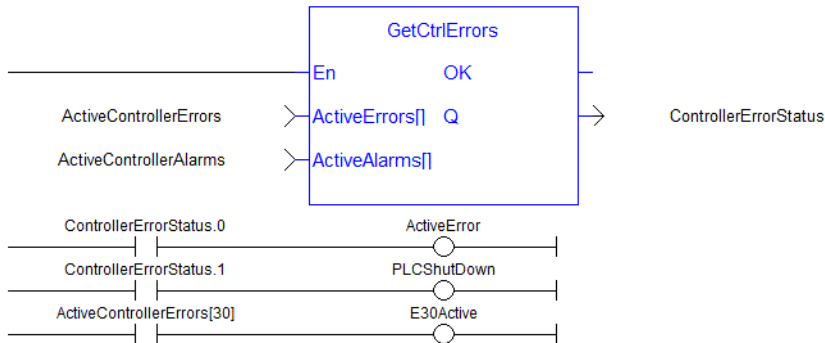
4.7.3.0.1 Status Meaning

Bit	Value	Description
Bit 0	Value 0	No error, no alarm, no shut down.
Bit 0	Value 1	There is an active error or an active alarm (i.e., there is something in the array ActiveError/ActiveAlarm).
Bit 1	Value 0	No shut down.
Bit 1	Value 1	The PLC processes is shut down. This starts 10 seconds after the error is triggered.
Bits 2 to 15	Value 2 4 9 to 2147483648	Reserved.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
//Retrieve active controller level alarm and errors.
//Check status output to see if any error or alarm is active and if PLC is
shutting down
ControllerErrorStatus := GetCtrlErrors(ActiveControllerErrors,
ActiveControllerAlarms);
ActiveError := ControllerErrorStatus.0;
PLCShutDown := ControllerErrorStatus.1;
E30Active := ActiveControllerErrors[30];
```


4.7.4 GetCtrlInfo

PLCopen ✓

Pipe Network ✓

 **Function Block** - Get the serial, model, and/or part number of the controller.
Returns a String containing the value of the control parameter requested.

Inputs

Input	Data Type	Range	Unit	Default	Description		
Execute	BOOL	0 to 1	N/A	No default	Rising edge of enable initiates read of parameter.		
Parameter Number	INT	1, 6	N/A	No default	Parameter number to read.		
					These parameters are also Internal Defines:		
					Value	Integer Representation	Description
					CTRLINFO_SERIAL_NUMBER	1	Controller serial number.
					CTRLINFO_MODEL_NUMBER	2	Controller model number.
					CTRLINFO_PART_NUMBER	3	Controller part number.
					CTRLINFO_CPU_CORE_COUNT	4	Number of CPU cores in the controller.
					CTRLINFO_PROJECT_BUILD_NO	5	The project build number of the running application.
					CTRLINFO_PROJECT_COMPILE_TIME	6	The project compile time of the running application.
CTRLINFO_MODEL_FAMILY	7	Controller model family.					

Outputs

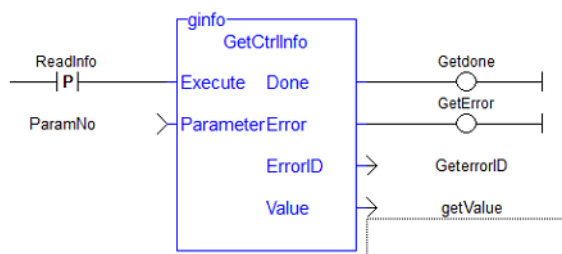
Output	Data Type	Range	Unit	Description
Done	BOOL	No range	N/A	Indicates whether this function block has completed without error.

Output	Data Type	Range	Unit	Description															
Error	BOOL	No range	N/A	Indicates whether this function block has completed with error.															
ErrorID	INT	No range	N/A	Error value to indicate error condition. These parameters are also Internal Defines: <table><tr><th>Value</th><th>Integer Representation</th><th>Description</th></tr><tr><td>CTRLINFO_ERROR_NO_ERROR</td><td>0</td><td>No error.</td></tr><tr><td>CTRLINFO_ERROR_INV_PARAMETER</td><td>1</td><td>Invalid parameter.</td></tr><tr><td>CTRLINFO_ERROR_CANT_READ_DATA</td><td>2</td><td>Error reading data.</td></tr><tr><td>CTRLINFO_ERROR_NOT_PDMM</td><td>3</td><td>Not valid on a non-PCMM2G controller.</td></tr></table>	Value	Integer Representation	Description	CTRLINFO_ERROR_NO_ERROR	0	No error.	CTRLINFO_ERROR_INV_PARAMETER	1	Invalid parameter.	CTRLINFO_ERROR_CANT_READ_DATA	2	Error reading data.	CTRLINFO_ERROR_NOT_PDMM	3	Not valid on a non-PCMM2G controller.
Value	Integer Representation	Description																	
CTRLINFO_ERROR_NO_ERROR	0	No error.																	
CTRLINFO_ERROR_INV_PARAMETER	1	Invalid parameter.																	
CTRLINFO_ERROR_CANT_READ_DATA	2	Error reading data.																	
CTRLINFO_ERROR_NOT_PDMM	3	Not valid on a non-PCMM2G controller.																	
Value	STRING	No range	N/A	String containing data that was read. IF Parameter Number = CTRLINFO_MODEL_FAMILY (7): <table><tr><th>Value</th><th>Integer Representation</th><th>Description</th></tr><tr><td>CTRLINFO_MODEL_FAMILY_PCMM2G</td><td>1</td><td>PCMM2G - 2nd generation controller.</td></tr><tr><td>CTRLINFO_MODEL_FAMILY_SIMULATOR</td><td>2</td><td>KAS Simulator</td></tr><tr><td>CTRLINFO_MODEL_FAMILY_UNKNOWN</td><td>3</td><td>Unknown controller family.</td></tr></table>	Value	Integer Representation	Description	CTRLINFO_MODEL_FAMILY_PCMM2G	1	PCMM2G - 2nd generation controller.	CTRLINFO_MODEL_FAMILY_SIMULATOR	2	KAS Simulator	CTRLINFO_MODEL_FAMILY_UNKNOWN	3	Unknown controller family.			
Value	Integer Representation	Description																	
CTRLINFO_MODEL_FAMILY_PCMM2G	1	PCMM2G - 2nd generation controller.																	
CTRLINFO_MODEL_FAMILY_SIMULATOR	2	KAS Simulator																	
CTRLINFO_MODEL_FAMILY_UNKNOWN	3	Unknown controller family.																	

FBD Language Example

Not available.

FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_GetCtrlInfo( ExecuteRead, CTRLINFO_SERIAL_NUMBER);  
  
if Inst_GetCtrlInfo.Done then  
    serialNumber := Inst_GetCtrlInfo.Value;  
end_if;
```

4.7.5 GetCtrlPerf

PLCopen 

Pipe Network 



Function Block - Returns controller CPU performance statistics.

Generates a text file with performance statistics of the controller.

Inputs

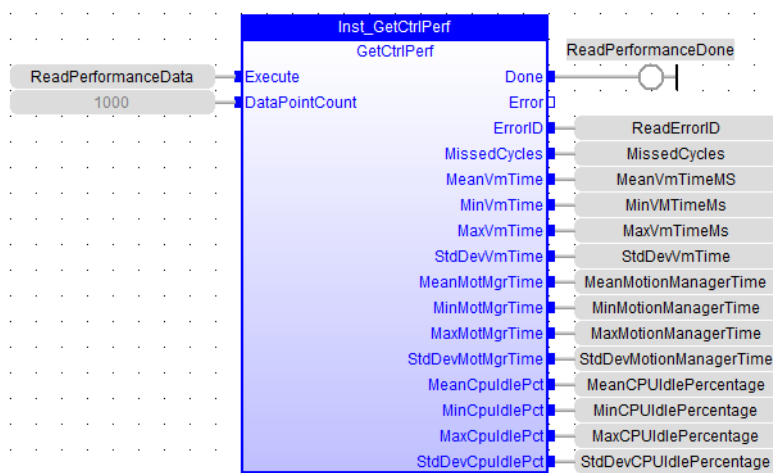
Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, request to collect the controller's performance data.
DataPointCount	UDINT	2 - 350,000,000	Cycles	No default	The number of motion manager cycles over which performance statistics are gathered.

Outputs

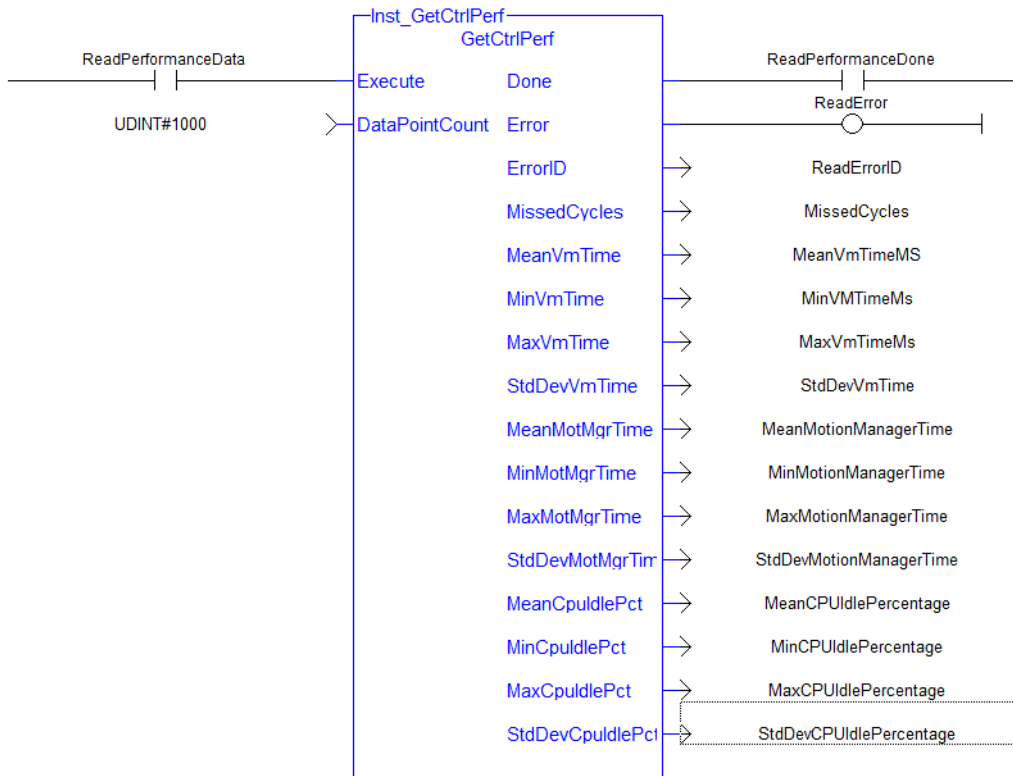
Output	Data Type	Range	Unit	Description
Done	BOOL	No range	N/A	If TRUE, the command completed successfully.
Error	BOOL	No range	N/A	If TRUE, an error has occurred.
ErrorID	INT	No range	N/A	- ErrorID = 0 indicates no error. - ErrorID = 1 indicates an error.
MissedCycles	UDINT	No range	N/A	Indicates the number of missing VM cycles.
MeanVmTime	LREAL	≥ 0 (zero)	Microseconds	The mean VM execution time.
StdDevVmTime	LREAL	≥ 0 (zero)	Microseconds	The standard deviation of the VM execution time.
MinVmTime	LREAL	≥ 0 (zero)	Microseconds	The minimum VM execution time.
MaxVmTime	LREAL	≥ 0 (zero)	Microseconds	The maximum VM execution time.
MeanMotMgrTime	LREAL	≥ 0 (zero)	Microseconds	The mean motion manager execution time.
StdDevMotMgrTime	LREAL	≥ 0 (zero)	Microseconds	The standard deviation of the motion manager execution time.
MinMotMgrTime	LREAL	≥ 0 (zero)	Microseconds	The minimum motion manager execution time.
MaxMotMgrTime	LREAL	≥ 0 (zero)	Microseconds	The maximum motion manager execution time.

Output	Data Type	Range	Unit	Description
MeanCpuidlePct	LREAL	0 to 100	Percent	The mean percentage of time the controller CPU is idle. - PCMM2G: CPU idle period is measured across the CPU cores that run the motion engine and PLC program. - See Item 3, Channel 4: PCMM2G - PLC Margin in the 7. Control Panel buttons section for more information.
StdDevCpuidlePct	LREAL	0 to 100	N/A	- PCMM2G: CPU idle period is measured across the CPU cores that run the motion engine and PLC program. - See Item 3, Channel 4: PCMM2G - PLC Margin in the 7. Control Panel buttons section for more information.
MinCpuidlePct	LREAL	0 to 100	N/A	- PCMM2G: CPU idle period is measured across the CPU cores that run the motion engine and PLC program. - See Item 3, Channel 4: PCMM2G - PLC Margin in the 7. Control Panel buttons section for more information.
MaxCpuidlePct	LREAL	0 to 100	N/A	- PCMM2G: CPU idle period is measured across the CPU cores that run the motion engine and PLC program. - See Item 3, Channel 4: PCMM2G - PLC Margin in the 7. Control Panel buttons section for more information.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```

//Read controller performance data from last 1000 cycles (1 second at T#1ms
update rate)
Inst_GetCtrlPerf( ReadPerformanceData, 1000 );

IF Inst_GetCtrlInfo.Done THEN
    MissedCycles:= Inst_GetCtrlPerf.MissedCycles;
    MaxVmTimeMs:= Inst_GetCtrlPerf.MaxVmTime;
    MaxMotionManagerTime:= Inst_GetCtrlPerf.MaxMotMgrTime;
    MeanCPUIidlePercentage:= Inst_GetCtrlPerf.MeanCpuIdlePct;
    MaxCPUIidlePercentage:= Inst_GetCtrlPerf.MaxCpuIdlePct;
END IF;

```

See Also

- Call a Function Block
- Differences Between Functions and Function Blocks

4.8 File Function Blocks

This section documents function blocks that can be applied to files.

Name	Description
FileClose	Closes an open file.
FileCopy	Copies a file's contents to a new file.
FileDelete	Removes a file from the file system.
FileEOF	Test if the end of the file is reached in a file that is open for reading.
FileExists	Tests if a file exists.
FileOpenA	Create or open a file in append mode.
FileOpenR	Open a file for reading.
FileOpenW	Create or reset a file and open it for writing.
FileReadBinData	Read binary data from a file.
FileReadLine	Reads a string value from a text file.
FileRename	Renames a file.
FileSeek	Sets the current position in an open file.
FileSize	Gets the size of a file.
FileWriteBinData	Write binary data to a file.
FileWriteLine	Writes a string value to a text file.

4.8.1 FileClose



Function Block - Closes an open file.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, request to perform closing a file.
ID	UDINT	N/A	N/A	No default	The ID of the open file.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* FileClose example *)
CASE StepCounter OF
0:
  Inst_FileClose(TRUE, MyOutputFileID);
  StepCounter := StepCounter + 1;
1:
  Inst_FileClose(TRUE, MyOutputFileID);
  IF Inst_FileClose.Done THEN
    Inst_FileClose(FALSE, 0);
    StepCounter := StepCounter + 1;
```



```
END_IF;
END_CASE;
```

See Also

- File Management
- "FileCopy" (→ p. 915)
- "FileDelete" (→ p. 917)
- "FileEOF" (→ p. 919)
- "FileExists" (→ p. 921)
- "FileOpenA" (→ p. 923)
- "FileOpenR" (→ p. 925)
- "FileOpenW" (→ p. 927)
- "FileReadBinData" (→ p. 929)
- "FileReadLine" (→ p. 931)
- "FileRename" (→ p. 933)
- "FileSeek" (→ p. 935)
- "FileSize" (→ p. 937)
- "FileWriteBinData" (→ p. 939)
- "FileWriteLine" (→ p. 941)

4.8.2 FileCopy



Function Block - Copies a file's contents to a new file.

- Large files take a noticeable amount of time to complete.
 - Example: A 1000KB file takes approximately 0.6 seconds.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, request to perform copying a file.
Src	STRING	N/A	N/A	No default	The path to the source file.
Dst	STRING	N/A	User units	No default	The destination path to the source file.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* FileCopy example *)
CASE StepCounter OF
0:
  Inst_FileCopy(TRUE, 'Source.txt', 'Dest.txt');
  StepCounter := StepCounter + 1;
```

```

1:
  Inst_FileCopy(TRUE, 'Source.txt', 'Dest.txt');
  IF Inst_FileCopy.Done THEN
    Inst_FileCopy(FALSE, '', '');
    StepCounter := StepCounter + 1;
  END_IF;
END_CASE;

```

See Also

- File Management
- "FileClose" (→ p. 913)
- "FileDelete" (→ p. 917)
- "FileEOF" (→ p. 919)
- "FileExists" (→ p. 921)
- "FileOpenA" (→ p. 923)
- "FileOpenR" (→ p. 925)
- "FileOpenW" (→ p. 927)
- "FileReadBinData" (→ p. 929)
- "FileReadLine" (→ p. 931)
- "FileRename" (→ p. 933)
- "FileSeek" (→ p. 935)
- "FileSize" (→ p. 937)
- "FileWriteBinData" (→ p. 939)
- "FileWriteLine" (→ p. 941)

4.8.3 FileDelete



Function Block - Removes a file from the file system.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, request to perform deletion of a file.
Path	STRING	N/A	N/A	No default	The path to the file.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* FileDelete example *)
CASE StepCounter OF
0:
  Inst_FileDelete(TRUE, 'Test.txt');
  StepCounter := StepCounter + 1;
1:
  Inst_FileDelete(TRUE, 'Test.txt');
  IF Inst_FileDelete.Done THEN
    Inst_FileDelete(FALSE, '');
    StepCounter := StepCounter + 1;
```

```
END_IF;
END_CASE;
```

See Also

- File Management
- "FileClose" (→ p. 913)
- "FileCopy" (→ p. 915)
- "FileEOF" (→ p. 919)
- "FileExists" (→ p. 921)
- "FileOpenA" (→ p. 923)
- "FileOpenR" (→ p. 925)
- "FileOpenW" (→ p. 927)
- "FileReadBinData" (→ p. 929)
- "FileReadLine" (→ p. 931)
- "FileRename" (→ p. 933)
- "FileSeek" (→ p. 935)
- "FileSize" (→ p. 937)
- "FileWriteBinData" (→ p. 939)
- "FileWriteLine" (→ p. 941)

4.8.4 FileEOF



Function Block - Test if the end of the file is reached in a file that is open for reading.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, test if the end of the file is reached.
ID	UDINT	N/A	N/A	No default	The ID of the open file.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
EOF	BOOL			TRUE if the end of the file has been encountered.

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* FileEOF example *)
CASE StepCounter OF
0:
  Inst_FileEOF(TRUE, MyInputFileID);
  StepCounter := StepCounter + 1;
1:
  Inst_FileEOF(TRUE, MyInputFileID);
  IF Inst_FileEOF.Done THEN
```

```

    EofReached := Inst_FileEOF.EOF;
    Inst_FileEOF(FALSE, 0);
    StepCounter := StepCounter + 1;
  END_IF;
END_CASE;

```

See Also

- File Management
- "FileClose" (→ p. 913)
- "FileCopy" (→ p. 915)
- "FileDelete" (→ p. 917)
- "FileExists" (→ p. 921)
- "FileOpenA" (→ p. 923)
- "FileOpenR" (→ p. 925)
- "FileOpenW" (→ p. 927)
- "FileReadBinData" (→ p. 929)
- "FileReadLine" (→ p. 931)
- "FileRename" (→ p. 933)
- "FileSeek" (→ p. 935)
- "FileSize" (→ p. 937)
- "FileWriteBinData" (→ p. 939)
- "FileWriteLine" (→ p. 941)

4.8.5 FileExists



Function Block - Tests if a file exists.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, test to see if a file exists.
Path	STRING	N/A	N/A	No default	The path to the file.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
Exists	BOOL			The existence of the file. - TRUE: Indicates the file exists - FALSE: Indicates the file does not exist.

FBD Language Example

Not available.

FPLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* FileExists example *)
CASE StepCounter OF
0:
  Inst_FileExists(TRUE, 'Test.txt');
  StepCounter := StepCounter + 1;
1:
  Inst_FileExists(TRUE, 'Test.txt');
```



```

IF Inst_FileExists.Done THEN
  TestFileExists := Inst_FileExists.Exists;
  Inst_FileExists(FALSE, '');
  StepCounter := StepCounter + 1;
END_IF;
END_CASE;

```

See Also

- File Management
- "FileClose" (→ p. 913)
- "FileCopy" (→ p. 915)
- "FileDelete" (→ p. 917)
- "FileEOF" (→ p. 919)
- "FileOpenA" (→ p. 923)
- "FileOpenR" (→ p. 925)
- "FileOpenW" (→ p. 927)
- "FileReadBinData" (→ p. 929)
- "FileReadLine" (→ p. 931)
- "FileRename" (→ p. 933)
- "FileSeek" (→ p. 935)
- "FileSize" (→ p. 937)
- "FileWriteBinData" (→ p. 939)
- "FileWriteLine" (→ p. 941)

4.8.6 FileOpenA



Function Block - Create or open a file in append mode.

Files must be closed using the "FileClose" (→ p. 913) function block.

NOTE

The controller allows only 32 open files at any given time.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, request to open a file.
Path	STRING	N/A	N/A	No default	The path to the file.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
ID	UDINT			The ID of the open file.

FBD Language Example

Not available.

FPLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* FileOpenW example *)
(* Replace FileOpenW with FileOpenA or FileOpenR as needed*)
CASE StepCounter OF
0:
  Inst_FileOpenW(TRUE, 'Results.txt');
  StepCounter := StepCounter + 1;
```

```

1:
  Inst_FileOpenW(TRUE, 'Results.txt');
  IF Inst_FileOpenW.Done THEN
    ResultsFileID := Inst_FileOpenW.ID;
    Inst_FileOpenW(FALSE, '');
    StepCounter := StepCounter + 1;
  END_IF;
END_CASE;

```

See Also

- File Management
- "FileClose" (→ p. 913)
- "FileCopy" (→ p. 915)
- "FileDelete" (→ p. 917)
- "FileEOF" (→ p. 919)
- "FileExists" (→ p. 921)
- "FileOpenR" (→ p. 925)
- "FileOpenW" (→ p. 927)
- "FileReadBinData" (→ p. 929)
- "FileReadLine" (→ p. 931)
- "FileRename" (→ p. 933)
- "FileSeek" (→ p. 935)
- "FileSize" (→ p. 937)
- "FileWriteBinData" (→ p. 939)
- "FileWriteLine" (→ p. 941)

4.8.7 FileOpenR



Function Block - Open a file for reading.

Files must be closed using the "FileClose" (→ p. 913) function block.

NOTE

The controller allows only 32 open files at any given time.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, request to open a file.
Path	STRING	N/A	N/A	No default	The path to the file.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
ID	UDINT			The ID of the open file.

FBD Language Example

Not available.

FPLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* FileOpenW example *)
(* Replace FileOpenW with FileOpenA or FileOpenR as needed*)
CASE StepCounter OF
0:
  Inst_FileOpenW(TRUE, 'Results.txt');
  StepCounter := StepCounter + 1;
```

```

1:
  Inst_FileOpenW(TRUE, 'Results.txt');
  IF Inst_FileOpenW.Done THEN
    ResultsFileID := Inst_FileOpenW.ID;
    Inst_FileOpenW(FALSE, '');
    StepCounter := StepCounter + 1;
  END_IF;
END_CASE;

```

See Also

- File Management
- "FileClose" (→ p. 913)
- "FileCopy" (→ p. 915)
- "FileDelete" (→ p. 917)
- "FileEOF" (→ p. 919)
- "FileExists" (→ p. 921)
- "FileOpenA" (→ p. 923)
- "FileOpenW" (→ p. 927)
- "FileReadBinData" (→ p. 929)
- "FileReadLine" (→ p. 931)
- "FileRename" (→ p. 933)
- "FileSeek" (→ p. 935)
- "FileSize" (→ p. 937)
- "FileWriteBinData" (→ p. 939)
- "FileWriteLine" (→ p. 941)

4.8.8 FileOpenW



 **Function Block** - Create or reset a file and open it for writing.

- If a file already exists, it is overwritten.
- Files must be closed using the "FileClose" (→ p. 913) function block.

NOTE

The controller allows only 32 open files at any given time.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, request to open a file.
Path	STRING	N/A	N/A	No default	The path to the file.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
ID	UDINT			The ID of the open file.

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* FileOpenW example *)
(* Replace FileOpenW with FileOpenA or FileOpenR as needed*)
CASE StepCounter OF
0:
    Inst_FileOpenW(TRUE, 'Results.txt');
```

```

    StepCounter := StepCounter + 1;
1:
    Inst_FileOpenW(TRUE, 'Results.txt');
    IF Inst_FileOpenW.Done THEN
        ResultsFileID := Inst_FileOpenW.ID;
        Inst_FileOpenW(FALSE, '');
        StepCounter := StepCounter + 1;
    END_IF;
END_CASE;

```

See Also

- File Management
- "FileClose" (→ p. 913)
- "FileCopy" (→ p. 915)
- "FileDelete" (→ p. 917)
- "FileEOF" (→ p. 919)
- "FileExists" (→ p. 921)
- "FileOpenA" (→ p. 923)
- "FileOpenR" (→ p. 925)
- "FileReadBinData" (→ p. 929)
- "FileReadLine" (→ p. 931)
- "FileRename" (→ p. 933)
- "FileSeek" (→ p. 935)
- "FileSize" (→ p. 937)
- "FileWriteBinData" (→ p. 939)
- "FileWriteLine" (→ p. 941)

4.8.9 FileReadBinData



Function Block - Read binary data from a file.

- **FileReadBinData** stops reading from the file if it fills the passed **Frame** array, reads **FrameSize** bytes, or encounters the end of file, whichever comes first.
- After a successful call to **FileReadBinData**, use **SerializeIn** to extract variable data from the binary data read from a file.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, read the size of a file.
ID	UDINT	N/A	N/A	No default	The ID of the open file.
Frame	USINT[]	N/A	N/A	No default	Where the binary data is stored.
FrameSize	DINT	N/A	N/A	No default	Number of bytes to store in the Frame array.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
NbRead	DINT			The number of bytes read from the file.

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example


```

(* FileReadBinData example *)
CASE StepCounter OF
0:
  Inst_FileReadBinData(TRUE, MyInputFileID, InputFrame, 128);
  StepCounter := StepCounter + 1;
1:
  Inst_FileReadBinData(TRUE, MyInputFileID, InputFrame, 128);
  IF Inst_FileReadBinData.Done THEN
    BytesRead := Inst_FileReadBinData.NbRead;
    Inst_FileReadBinData(FALSE, 0, InputFrame, 0);
    StepCounter := StepCounter + 1;
  END_IF;
END_CASE;

```

See Also

- File Management
- "FileClose" (→ p. 913)
- "FileCopy" (→ p. 915)
- "FileDelete" (→ p. 917)
- "FileEOF" (→ p. 919)
- "FileExists" (→ p. 921)
- "FileOpenA" (→ p. 923)
- "FileOpenR" (→ p. 925)
- "FileOpenW" (→ p. 927)
- "FileReadLine" (→ p. 931)
- "FileRename" (→ p. 933)
- "FileSeek" (→ p. 935)
- "FileSize" (→ p. 937)
- "FileWriteBinData" (→ p. 939)
- "FileWriteLine" (→ p. 941)

4.8.10 FileReadLine



Function Block - Reads a string value from a text file.

FileReadLine stops reading from the file if either:

- 255 characters are read (the maximum length of the STRING type).
- A new line is encountered.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, read the size of a file.
ID	UDINT	N/A	N/A	No default	The ID of the open file.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
Value	STRING			The string value read from the file.

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* FileReadLine example *)
CASE StepCounter OF
0:
  Inst_FileReadLine(TRUE, MyInputFileID);
  StepCounter := StepCounter + 1;
1:
  Inst_FileReadLine(TRUE, MyInputFileID);
```

```

IF Inst_FileReadLine.Done THEN
  lineText := Inst_FileReadLine.Value;
  Inst_FileReadLine(FALSE, 0);
  StepCounter := StepCounter + 1;
END_IF;
END_CASE;

```

See Also

- File Management
- "FileClose" (→ p. 913)
- "FileCopy" (→ p. 915)
- "FileDelete" (→ p. 917)
- "FileEOF" (→ p. 919)
- "FileExists" (→ p. 921)
- "FileOpenA" (→ p. 923)
- "FileOpenR" (→ p. 925)
- "FileOpenW" (→ p. 927)
- "FileReadBinData" (→ p. 929)
- "FileRename" (→ p. 933)
- "FileSeek" (→ p. 935)
- "FileSize" (→ p. 937)
- "FileWriteBinData" (→ p. 939)
- "FileWriteLine" (→ p. 941)

4.8.11 FileRename



Function Block - Renames a file.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, request to rename a file.
Path	STRING	N/A	N/A	No default	The path to the file.
NewName	STRING	N/A	User units	No default	The new name of the file.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* FileRename example *)
CASE StepCounter OF
0:
  Inst_FileRename(TRUE, 'Original.txt', 'Renamed.txt');
  StepCounter := StepCounter + 1;
1:
  Inst_FileRename(TRUE, 'Original.txt', 'Renamed.txt');
```

```

IF Inst_FileRename.Done THEN
  Inst_FileRename(FALSE, '', '');
  StepCounter := StepCounter + 1;
END_IF;
END_CASE;

```

See Also

- File Management
- "FileClose" (→ p. 913)
- "FileCopy" (→ p. 915)
- "FileDelete" (→ p. 917)
- "FileEOF" (→ p. 919)
- "FileExists" (→ p. 921)
- "FileOpenA" (→ p. 923)
- "FileOpenR" (→ p. 925)
- "FileOpenW" (→ p. 927)
- "FileReadBinData" (→ p. 929)
- "FileReadLine" (→ p. 931)
- "FileSeek" (→ p. 935)
- "FileSize" (→ p. 937)
- "FileWriteBinData" (→ p. 939)
- "FileWriteLine" (→ p. 941)

4.8.12 FileSeek



Function Block - Sets the current position in an open file.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, test if the end of the file is reached in a file that is open for reading.
ID	UDINT	N/A	N/A	No default	The ID of the open file.
POS	DINT	N/A	N/A	No default	Number of bytes to offset from ORG. - If ORG = SEEK_SET, then POS should be ≥ 0 (zero). - If ORG = SEEK_END, then POS should be ≤ 0 (zero). - If ORG = SEEK_CUR, then POS can be positive or negative.
ORG	DINT	- SEEK_SET - SEEK_CUR - SEEK_END	N/A	No default	Origin of the move. - SEEK_SET = Beginning of the file. - SEEK_CUR = Current position. - SEEK_END = End of the file.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* FileSeek example *)
CASE StepCounter OF
0:
    (* Move to beginning of the file *)
    Inst_FileSeek(TRUE, MyInputFileID, 0, SEEK_SET);
    StepCounter := StepCounter + 1;
1:
    Inst_FileSeek(TRUE, MyInputFileID, 0, SEEK_SET);
    IF Inst_FileSeek.Done THEN
        Inst_FileSeek(FALSE, 0, 0, SEEK_SET);
        StepCounter := StepCounter + 1;
    END_IF;
END_CASE;
```

See Also

- File Management
- "FileClose" (→ p. 913)
- "FileCopy" (→ p. 915)
- "FileDelete" (→ p. 917)
- "FileEOF" (→ p. 919)
- "FileExists" (→ p. 921)
- "FileOpenA" (→ p. 923)
- "FileOpenR" (→ p. 925)
- "FileOpenW" (→ p. 927)
- "FileReadBinData" (→ p. 929)
- "FileReadLine" (→ p. 931)
- "FileRename" (→ p. 933)
- "FileSize" (→ p. 937)
- "FileWriteBinData" (→ p. 939)
- "FileWriteLine" (→ p. 941)

4.8.13 FileSize



Function Block - Gets the size of a file.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, read the size of a file.
Path	STRING	N/A	N/A	No default	The path to the file.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
Size	DINT			The size of the file in bytes.

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* FileSize example *)
CASE StepCounter OF
0:
  Inst_FileSize(TRUE, 'Test.txt');
  StepCounter := StepCounter + 1;
1:
  Inst_FileSize(TRUE, 'Test.txt');
  IF Inst_FileSize.Done THEN
    TestFileSize := Inst_FileSize.Size;
    Inst_FileSize(FALSE, '');
  END IF;
END CASE;
```



```

        StepCounter := StepCounter + 1;
    END_IF;
END_CASE;

```

See Also

- File Management
- "FileClose" (→ p. 913)
- "FileCopy" (→ p. 915)
- "FileDelete" (→ p. 917)
- "FileEOF" (→ p. 919)
- "FileExists" (→ p. 921)
- "FileOpenA" (→ p. 923)
- "FileOpenR" (→ p. 925)
- "FileOpenW" (→ p. 927)
- "FileReadBinData" (→ p. 929)
- "FileReadLine" (→ p. 931)
- "FileRename" (→ p. 933)
- "FileSeek" (→ p. 935)
- "FileWriteBinData" (→ p. 939)
- "FileWriteLine" (→ p. 941)

4.8.14 FileWriteBinData



Function Block - Write binary data to a file.

Before using **FileWriteBinData**, use **SerializeOut** to copy variable data to a binary frame.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, write binary data to a file.
ID	UDINT	N/A	N/A	No default	The ID of the open file.
Frame	USINT[]	N/A	N/A	No default	The array of binary data to write.
FrameSize	DINT	N/A	N/A	No default	Number of bytes to write from the Frame array.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* FileWriteBinData example *)
CASE StepCounter OF
0:
    Inst_FileWriteBinData(TRUE, MyOutputFileID, OutputFrame, 128);
```

```

    StepCounter := StepCounter + 1;
1:
    Inst_FileWriteBinData(TRUE, MyOutputFileID, OutputFrame, 128);
    IF Inst_FileWriteBinData.Done THEN
        Inst_FileWriteBinData(FALSE, 0, OutputFrame, 0);
        StepCounter := StepCounter + 1;
    END_IF;
END_CASE;

```

See Also

- File Management
- "FileClose" (→ p. 913)
- "FileCopy" (→ p. 915)
- "FileDelete" (→ p. 917)
- "FileEOF" (→ p. 919)
- "FileExists" (→ p. 921)
- "FileOpenA" (→ p. 923)
- "FileOpenR" (→ p. 925)
- "FileOpenW" (→ p. 927)
- "FileReadBinData" (→ p. 929)
- "FileReadLine" (→ p. 931)
- "FileRename" (→ p. 933)
- "FileSeek" (→ p. 935)
- "FileSize" (→ p. 937)
- "FileWriteLine" (→ p. 941)

4.8.15 FileWriteLine



Function Block - Writes a string value to a text file.

An end-of-line character is systematically written after the string value.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, write a string value to a file.
ID	UDINT	N/A	N/A	No default	The ID of the open file.
IN	STRING	N/A	N/A	No default	The string value to be written.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).

4.8.15.0.1 String Escape Sequences

For greater formatting control over the STRING output, escape the STRING by pre-pending a \$ and use a pre-defined sequence. This is called a string escape sequence.

Escape Sequence	Result
\$'	'
\$\$	\$
\$L	linefeed
\$N	newline
\$P	page (form feed)
\$R	return
\$T	tab
\$xx	hex value

Example: The is how STRING escape sequences are used.

```
ID:=FileOpenW('c:\ myfile.txt');
WOK:=FileWriteLine(ID, '123456$N');
//WOK:=FileWriteLine(ID, '$N');
WOK:=FileWriteLine(ID, 'abcd$N');
```

```
K:=FileWriteLine(ID, 'the end');
K:=FileClose(ID);
```

The example outputs a file which reads:

```
123456
abcd
the end
```

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* FileWriteLine example *)
CASE StepCounter OF
0:
  Inst_FileWriteLine(TRUE, MyOutputFileID, 'Hello, world. ');
  StepCounter := StepCounter + 1;
1:
  Inst_FileWriteLine(TRUE, MyOutputFileID, 'Hello, world. ');
  IF Inst_FileWriteLine.Done THEN
    Inst_FileWriteLine(FALSE, 0, '');
    StepCounter := StepCounter + 1;
  END_IF;
END_CASE;
```

See Also

- File Management
- "FileClose" (→ p. 913)
- "FileCopy" (→ p. 915)
- "FileDelete" (→ p. 917)
- "FileEOF" (→ p. 919)
- "FileExists" (→ p. 921)
- "FileOpenA" (→ p. 923)
- "FileOpenR" (→ p. 925)
- "FileOpenW" (→ p. 927)
- "FileReadBinData" (→ p. 929)
- "FileReadLine" (→ p. 931)
- "FileRename" (→ p. 933)
- "FileSeek" (→ p. 935)
- "FileSize" (→ p. 937)
- "FileWriteBinData" (→ p. 939)

4.9 Profiling Function Blocks

Name	Description
Stopwatch	Measures the elapsed time between two events and gathers basic statistics when measurements are repeated.

4.9.1 Stopwatch

PLCopen 

Pipe Network 



Function Block - Measures the elapsed time between two events and gathers basic statistics when measurements are repeated.

- When tuning the performance of an application, Stopwatch can be used to measure the execution time of a program, subprogram, UDFB, loops, or individual functions or function blocks.
- StopWatch **does not** set outputs when passed the STOPWATCH_START action.
 - This is to allow users to observe the last collected values when observing values via the KAS-IDE.
- StopWatch generates undefined values if called with a STOPWATCH_STOP is prior to being called with a STOPWATCH_START action.

Inputs

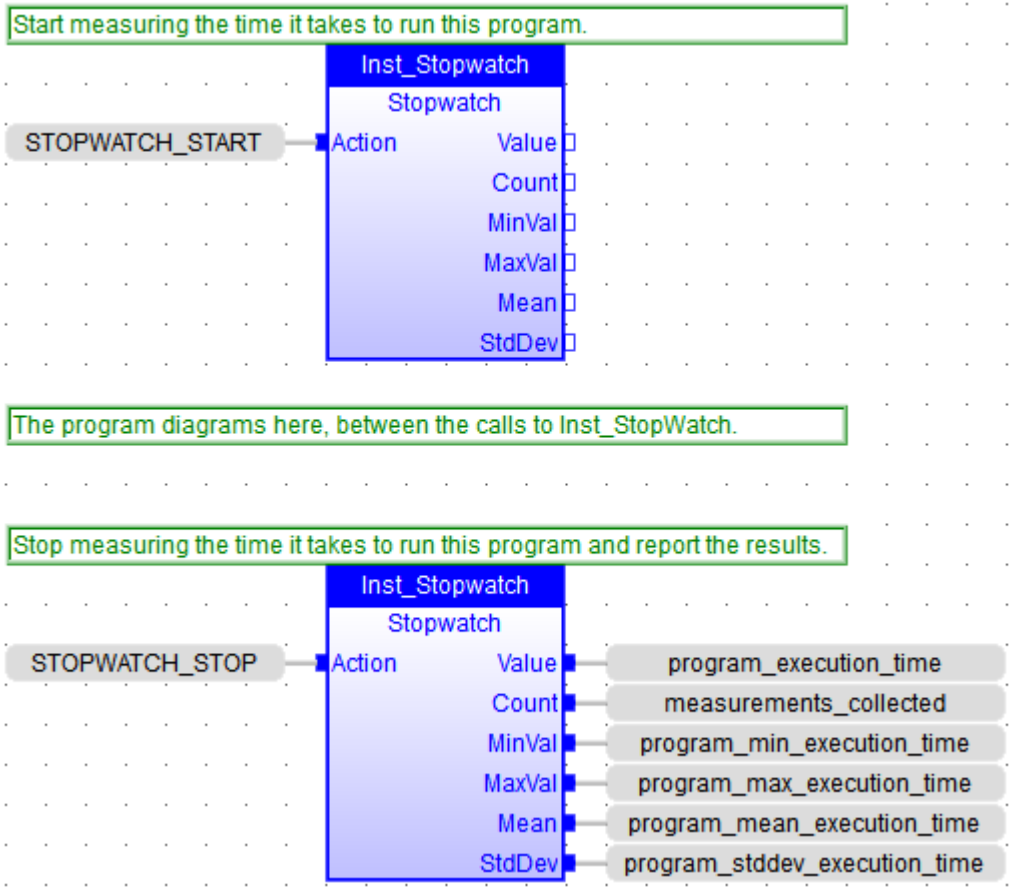
Input	Data Type	Range	Unit	Default	Description								
Action	INT	Enumerated	N/A	No default	The stopwatch action to perform: Uses one of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>STOPWATCH_START</td><td>Start the measurement of a time interval.</td></tr><tr><td>STOPWATCH_STOP</td><td>Stop the measurement of a time interval.</td></tr><tr><td>STOPWATCH_RESET</td><td> - Removes all information about previous measurements. - Resets the statistical information. </td></tr></table>	Value	Description	STOPWATCH_START	Start the measurement of a time interval.	STOPWATCH_STOP	Stop the measurement of a time interval.	STOPWATCH_RESET	- Removes all information about previous measurements. - Resets the statistical information.
Value	Description												
STOPWATCH_START	Start the measurement of a time interval.												
STOPWATCH_STOP	Stop the measurement of a time interval.												
STOPWATCH_RESET	- Removes all information about previous measurements. - Resets the statistical information.												

Outputs

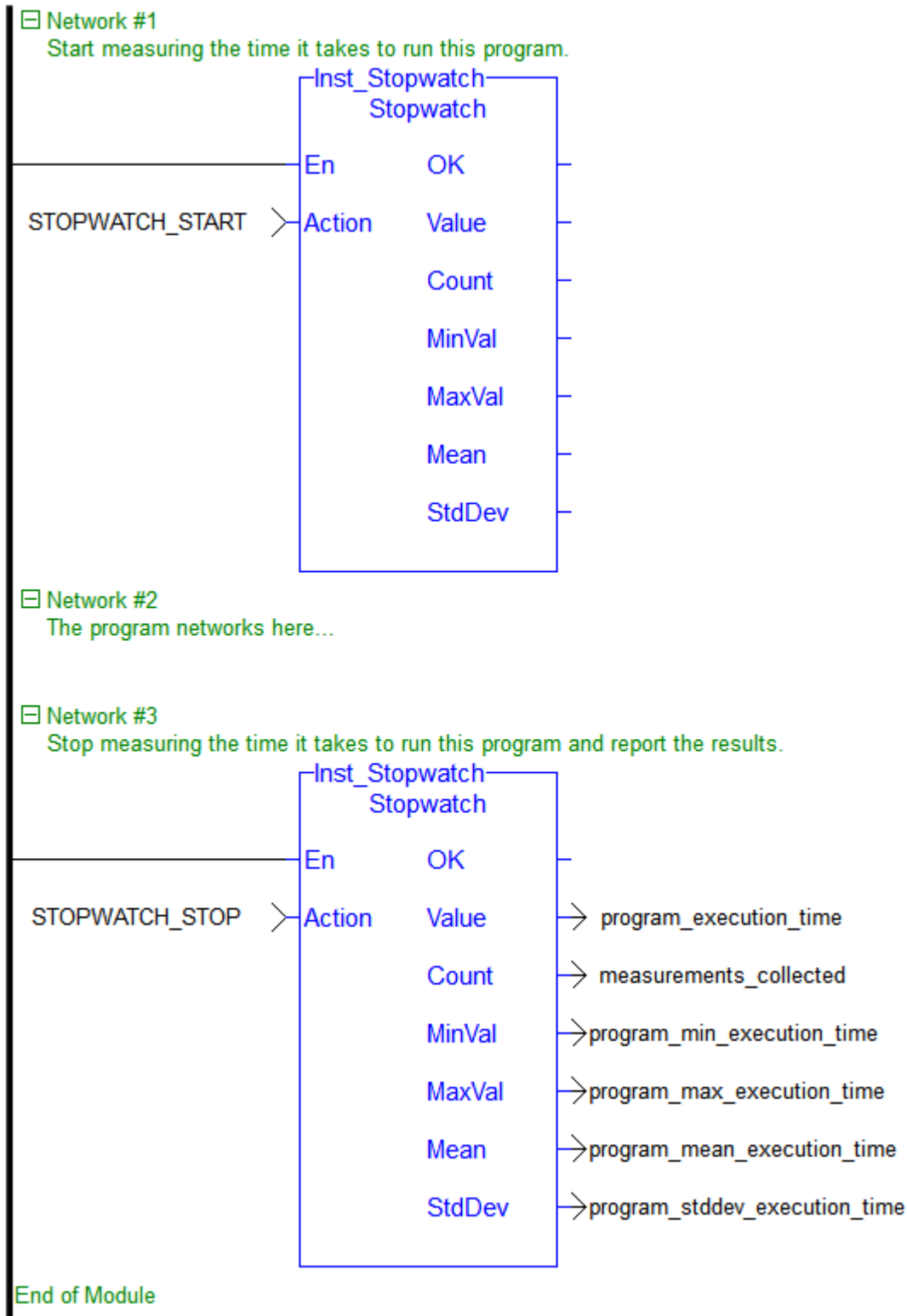
Output	Data Type	Range	Unit	Description
Value	LREAL	≥ 0 (zero)	Microseconds	- The last measured elapsed time measurement, set when a STOPWATCH_STOP action is used. - A STOPWATCH_RESET action sets Value to 0 (zero).
Count	LREAL	≥ 0 (zero)	N/A	- The number of elapsed time measurements gathered. - A STOPWATCH_RESET action sets Count to 0 (zero).
MinVal	LREAL	≥ 0 (zero)	Microseconds	- The minimum of gathered elapsed time measurements. - A STOPWATCH_RESET action sets MinVal to 0 (zero).

Output	Data Type	Range	Unit	Description
MaxVal	LREAL	≥ 0 (zero)	Microseconds	- The maximum of gathered elapsed time measurements. - A STOPWATCH_RESET action sets MaxVal to 0 (zero).
Mean	LREAL	≥ 0 (zero)	Microseconds	- The mean value of gathered elapsed time measurements. - A STOPWATCH_RESET action sets Mean to 0 (zero).
StdDev	LREAL	≥ 0 (zero)	Microseconds	- The standard deviation of gathered elapsed time measurements. - If less than 2 measurements have been gathered, then StdDev is set to zero. - A STOPWATCH_RESET action sets StdDev to 0 (zero).

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_Stopwatch(STOPWATCH_START);  
// The body of a program  
Inst_Stopwatch(STOPWATCH_STOP);  
mean_program_time_us := Inst_Stopwatch.Mean; // microseconds
```

See Also

- Controller Performance
- Troubleshooting > FAQs > Performance

4.10 Recipe Function Blocks

Name	Description
RecipeApplyInstance	Copy values to from a recipe instance to the program data.
RecipeClose	Close a recipe object.
RecipeOpen	Open a recipe file and create a recipe object.
RecipeQuickLoad	Load a recipe file and immediately apply the items it contains.
RecipeSave	Save a recipe object to a file.
RecipeSaveRetainVars	Save all retain variables, and their values, into a new recipe file.
RecipeStoreInstance	Store the present values of the program data in a recipe instance.

4.10.1 RecipeApplyInstance

This function/function block was added in KAS 5.00.



Function Block - Copy values to from a recipe instance to the program data.

- Validation occurs before writing data to the program data.
 - This avoids partial writes when an error occurs.

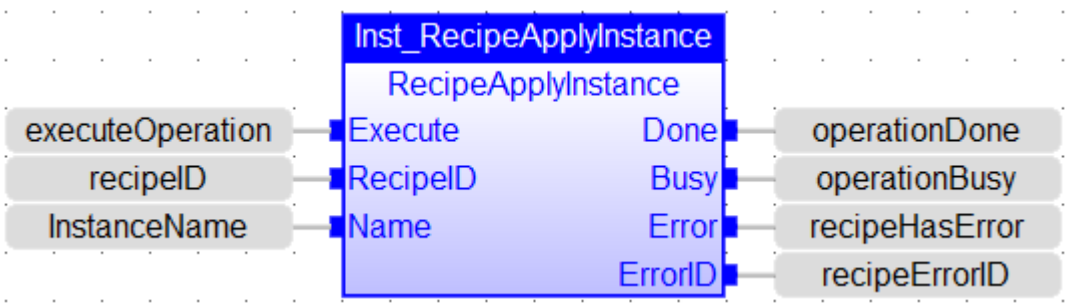
Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, copies the values of the specified instance into the program data.
RecipeID	UDINT	No range	N/A	No default	• The ID of the recipe object to use. • RecipeID is the handle ID returned by the "RecipeOpen" (→ p. 954) function block.
Name	STRING	No range	N/A	No default	• The name of the instance to apply. • If an empty string is supplied, the name "Instance" is used. • This is the same name applied to unnamed instances when a recipe is loaded from a file.

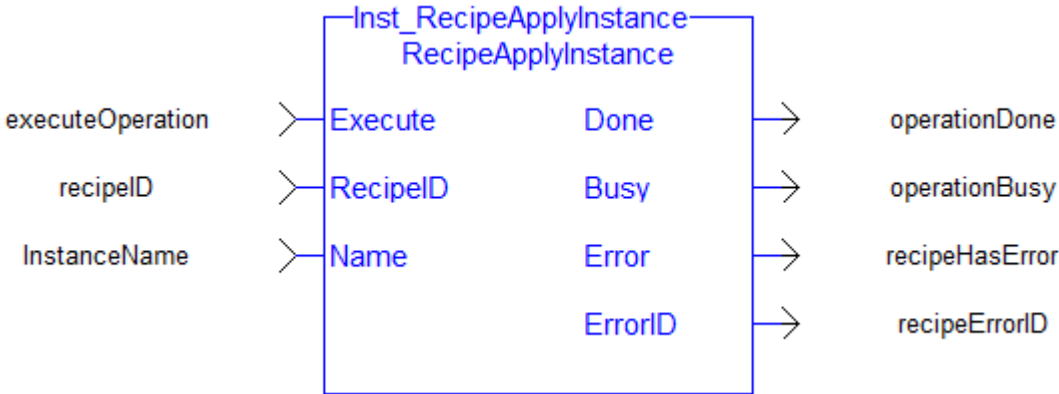
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	DINT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_RecipeApplyInstance(True, recipeID, 'Widgets');  
IF Inst_RecipeApplyInstance.Error THEN  
    errorMessage := MC_ErrorDescription(Inst_RecipeApplyInstance.ErrorID);  
END_IF;
```

4.10.2 RecipeClose

This function/function block was added in KAS 5.00.



Function Block - Close a recipe object.

RecipeClose unloads a recipe object from the actively running program.

- **RecipeClose** does **not** delete the recipe file used to create the recipe object.
 - To delete a recipe file, use the [FileDelete](#) function block.
- **RecipeClose** does **not** save changes made to the recipe object.
- If instances were modified via "RecipeStoreInstance" (→ p. 962), call "RecipeSave" (→ p. 958) before to persist the changes to the file system.

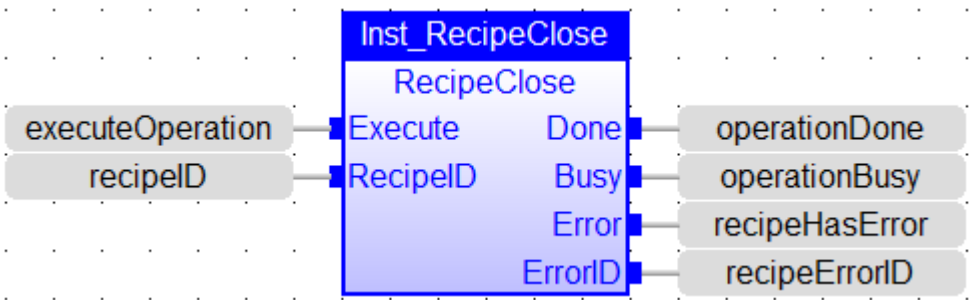
Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, close the specified recipe object.
RecipeID	UDINT	No range	N/A	No default	• The ID of the recipe object to close. • RecipeID is the handle ID returned by the "RecipeOpen" (→ p. 954) function block.

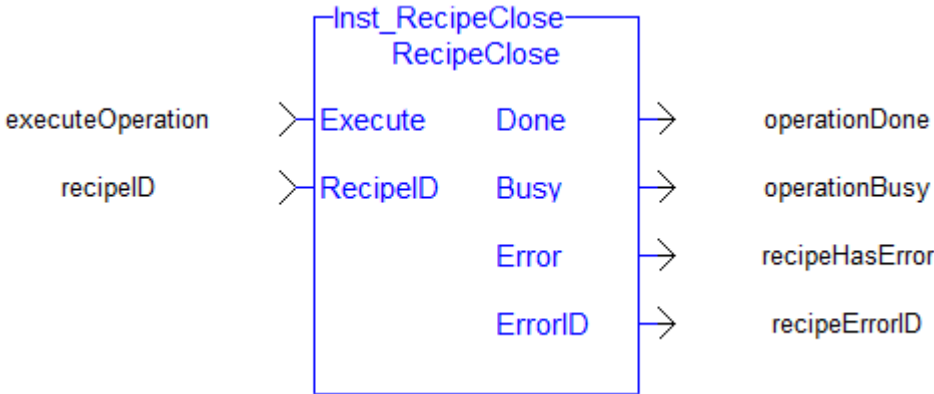
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	DINT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_RecipeClose(True, recipeID);  
IF Inst_RecipeClose.Error THEN  
    errorMessage := MC_ErrorDescription(Inst_RecipeClose.ErrorID);  
END_IF;
```


4.10.3 RecipeOpen

This function/function block was added in KAS 5.00.



Function Block - Open a recipe file and create a recipe object.

Recipe files opened with RecipeOpen must include a recipe definition section.

- Instance sections are optional.
- Recipes without instance sections can be opened and used later with "RecipeStoreInstance" (→ p. 962) to create instances from active program data.

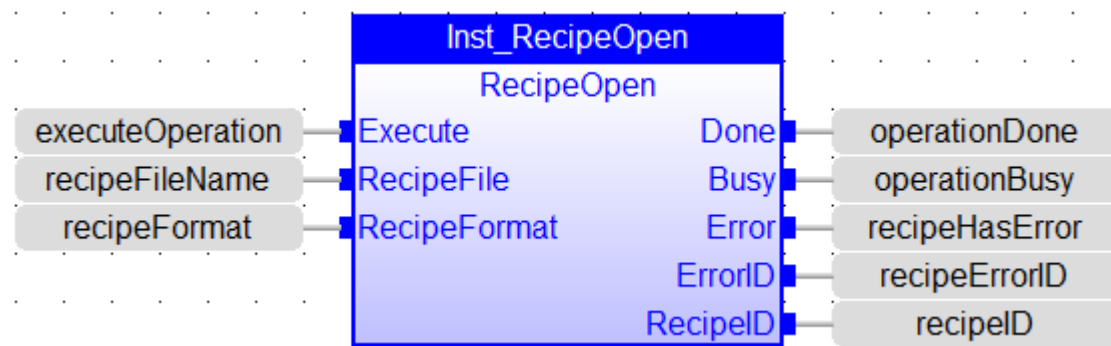
Inputs

Input	Data Type	Range	Unit	Default	Description						
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, load a recipe file.						
RecipeFile	STRING	No range	N/A	No default	The location of the recipe file.						
RecipeFormat	SINT	Enumerated	N/A	No default	• The recipe file format. • Valid values include one of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>RCP_FMT_XML</td><td>XML file format.</td></tr><tr><td>RCP_FMT_TOML</td><td>TOML file format.</td></tr></table>	Value	Description	RCP_FMT_XML	XML file format.	RCP_FMT_TOML	TOML file format.
Value	Description										
RCP_FMT_XML	XML file format.										
RCP_FMT_TOML	TOML file format.										

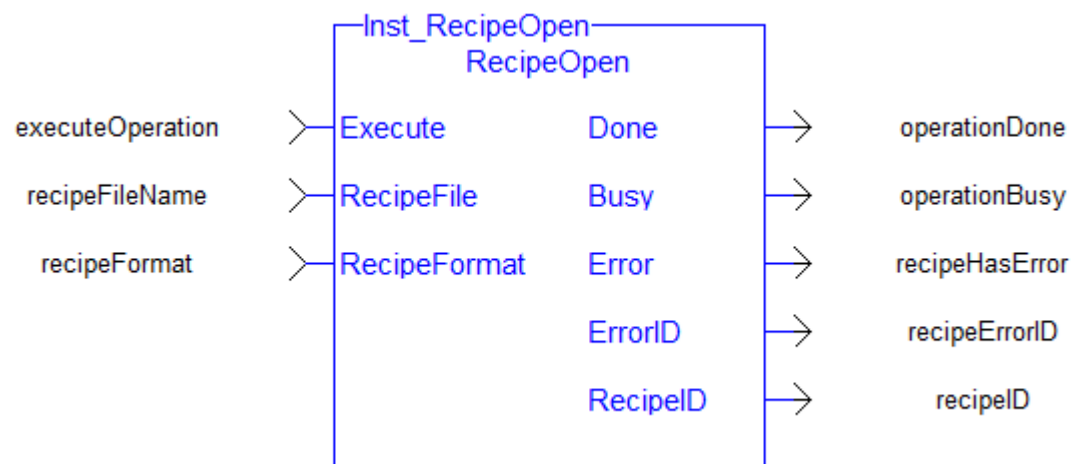
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	DINT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
RecipeID	UDINT	No range	N/A	The ID of the recipe object.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```

Inst_RecipeOpen(True, 'Parts.xml', RCP_FMT_XML);
IF Inst_RecipeOpen.Error THEN
    errorMessage := MC_ErrorDescription(Inst_RecipeOpen.ErrorID);
ELSE
    recipeID = Inst_RecipeOpen.RecipeID
END_IF;
//
Inst_RecipeOpen(True, '/usbflash/recipes/Parts.xml', RCP_FMT_XML);
IF Inst_RecipeOpen.Error THEN
    errorMessage := MC_ErrorDescription(Inst_RecipeOpen.ErrorID);
ELSE
    recipeID = Inst_RecipeOpen.RecipeID
END_IF;

```

4.10.4 RecipeQuickLoad

This function/function block was added in KAS 5.00.



Function Block - Load a recipe file and immediately apply the items it contains.

RecipeQuickLoad loads a recipe instance without requiring that a recipe definition be loaded first.

- Only the first recipe instance found in the file is applied.
- When ContinueOnError is TRUE, all validation errors encountered are added to the controller log.
 - See [Log Messages tabs](#).

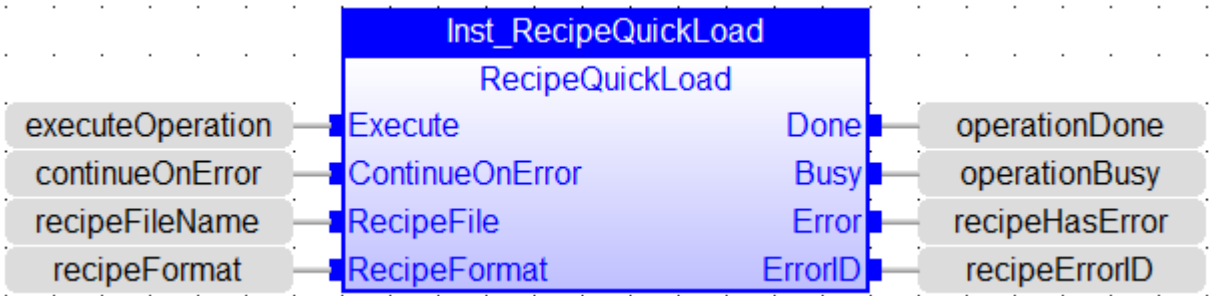
Inputs

Input	Data Type	Range	Unit	Default	Description						
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, request loading and applying the recipe stored in the file RecipeFile.						
ContinueOnError	BOOL	FALSE, TRUE	N/A	No default	• If FALSE, no recipe items are applied if an error occurs. • If TRUE, an error does not halt application of recipe items. • All recipe items that can be applied are applied.						
RecipeFile	STRING	No range	N/A	No default	The location of the recipe file.						
RecipeFormat	SINT	Enumerated	N/A	No default	• The recipe file format. • Valid values include one of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>RCP_FMT_XML</td><td>XML file format.</td></tr><tr><td>RCP_FMT_TOML</td><td>TOML file format.</td></tr></table>	Value	Description	RCP_FMT_XML	XML file format.	RCP_FMT_TOML	TOML file format.
Value	Description										
RCP_FMT_XML	XML file format.										
RCP_FMT_TOML	TOML file format.										

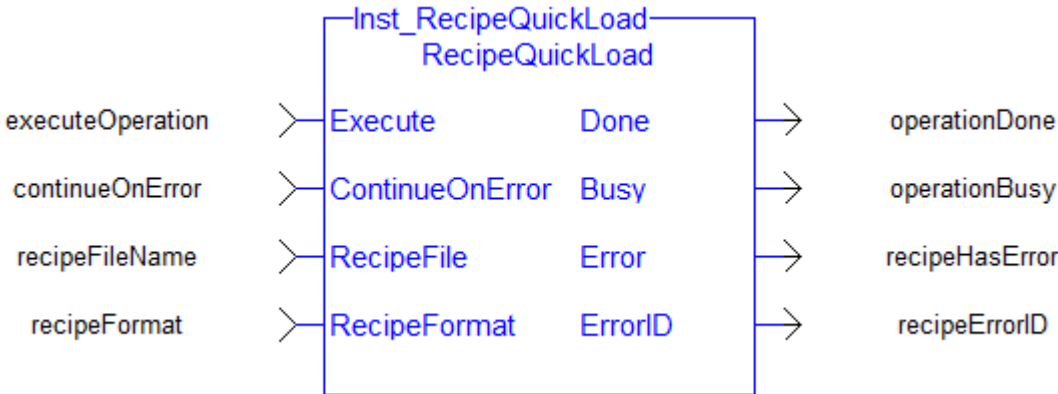
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	DINT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_RecipeQuickLoad(True, False, 'CogsRecipeInst.xml', RCP_FMT_XML);  
IF Inst_RecipeQuickLoad.Error THEN  
    errorMessage := MC_ErrorDescription(Inst_RecipeQuickLoad.ErrorID);  
END_IF;
```

4.10.5 RecipeSave

This function/function block was added in KAS 5.00.

PLCopen 

Pipe Network 



Function Block - Save a recipe object to a file.

- **RecipeSave** writes the recipe object (definition + stored instances) to the filesystem.
- It does not read live PLC values.
- To capture PLC values before saving, call "RecipeStoreInstance" (→ p. 962) beforehand.

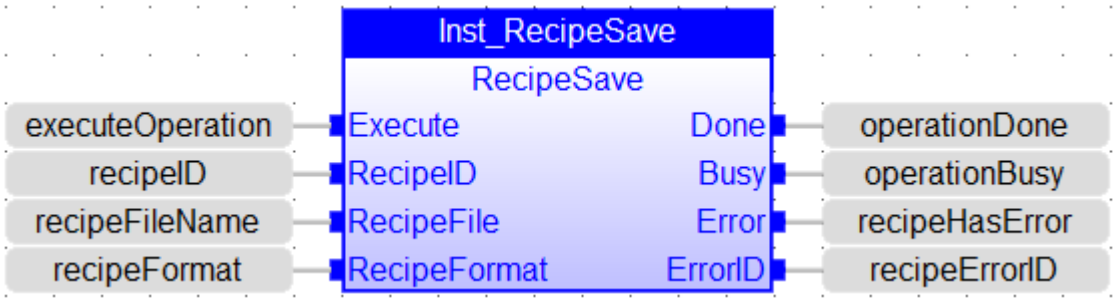
Inputs

Input	Data Type	Range	Unit	Default	Description						
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, save the recipe to the file RecipeFile.						
RecipeID	UDINT	No range	N/A	No default	- The ID of the recipe object to use when saving a recipe. RecipeID is the handle ID returned by the "RecipeOpen" (→ p. 954) function block.						
RecipeFile	STRING	No range	N/A	No default	The path of the destination recipe file.						
RecipeFormat	SINT	Enumerated	N/A	No default	- The recipe file format. - Valid values include one of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>RCP_FMT_XML</td><td>XML file format.</td></tr><tr><td>RCP_FMT_TOML</td><td>TOML file format.</td></tr></table>	Value	Description	RCP_FMT_XML	XML file format.	RCP_FMT_TOML	TOML file format.
Value	Description										
RCP_FMT_XML	XML file format.										
RCP_FMT_TOML	TOML file format.										

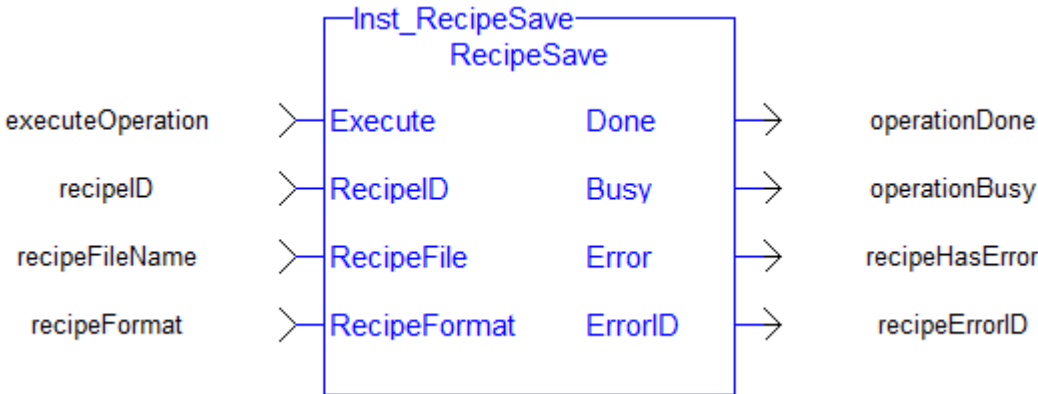
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	DINT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_RecipeSave(True, recipeID, 'Parts_Recipes.xml', RCP_FMT_XML);  
IF Inst_RecipeSave.Error THEN  
    errorMessage := MC_ErrorDescription(Inst_RecipeSave.ErrorID);  
END_IF;
```

4.10.6 RecipeSaveRetainVars

This function/function block was added in KAS 5.00.



 **Function Block** - Save all retain variables, and their values, into a new recipe file.

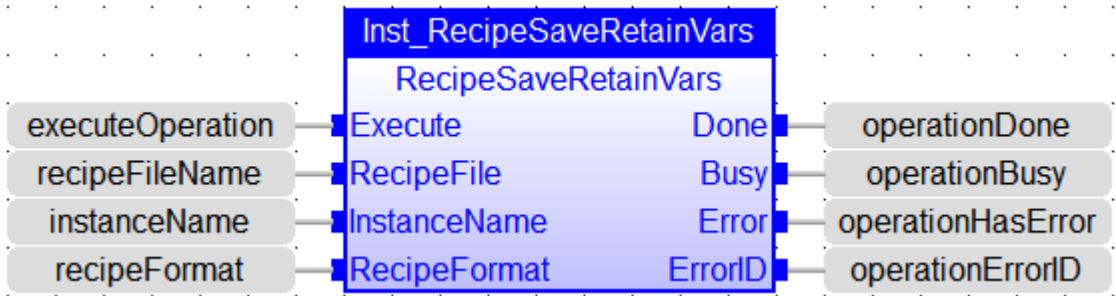
Inputs

Input	Data Type	Range	Unit	Default	Description						
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, save the retain variables, and their values, to a new recipe file.						
RecipeFile	STRING	No range	N/A	No default	- The path of the destination recipe file. - If the file with the same name already exists, it is overwritten.						
InstanceName	STRING	No range	N/A	No default	- The name of the new instance. - If InstanceName is an empty string, the instance is assigned the name Instance.						
RecipeFormat	SINT	Enumerated	N/A	No default	- The recipe file format. - Valid values include one of these enumeration values: <table><tr><th>Value</th><th>Description</th></tr><tr><td>RCP_FMT_XML</td><td>XML file format.</td></tr><tr><td>RCP_FMT_TOML</td><td>TOML file format.</td></tr></table>	Value	Description	RCP_FMT_XML	XML file format.	RCP_FMT_TOML	TOML file format.
Value	Description										
RCP_FMT_XML	XML file format.										
RCP_FMT_TOML	TOML file format.										

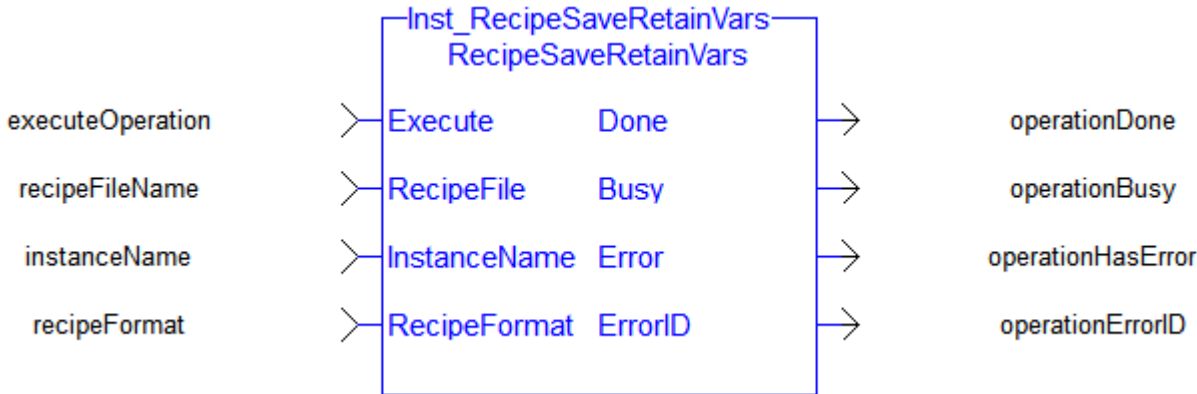
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	DINT	Enumerated	N/A	- Indicates the error if Error output is TRUE. - See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_RecipeSaveRetainVars(True, 'RetainVars.xml', 'Retain', RCP_FMT_XML);
IF Inst_RecipeSaveRetainVars.Error THEN
    errorMessage := MC_ErrorDescription(Inst_RecipeSaveRetainVars.ErrorID);
END_IF;
```


4.10.7 RecipeStoreInstance

This function/function block was added in KAS 5.00.

PLCopen ✓

Pipe Network ✓

 **Function Block** - Store the present values of the program data in a recipe instance.

- If an instance with the specified name already exists, the values in that instance are overwritten.
- If no instance exists with the specified name, a new instance with the specified name is created.
- No range validation is performed while storing.
 - Validation is deferred until a later call to RecipeApplyInstance.

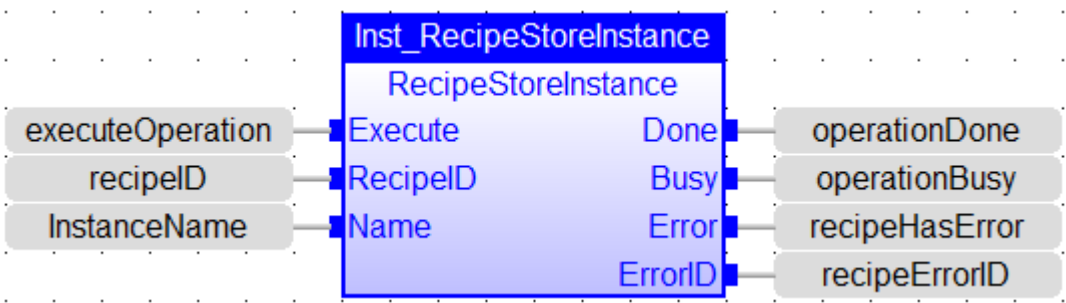
Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	FALSE, TRUE	N/A	No default	On the rising edge, save the program data into the named recipe instance.
RecipeID	UDINT	No range	N/A	No default	• The ID of the recipe object to use. • RecipeID is the handle ID returned by the "RecipeOpen" (→ p. 954) function block.
Name	STRING	No range	N/A	No default	• The name of the new instance. • If Name is an empty string, the instance is assigned the name 'Instance'.

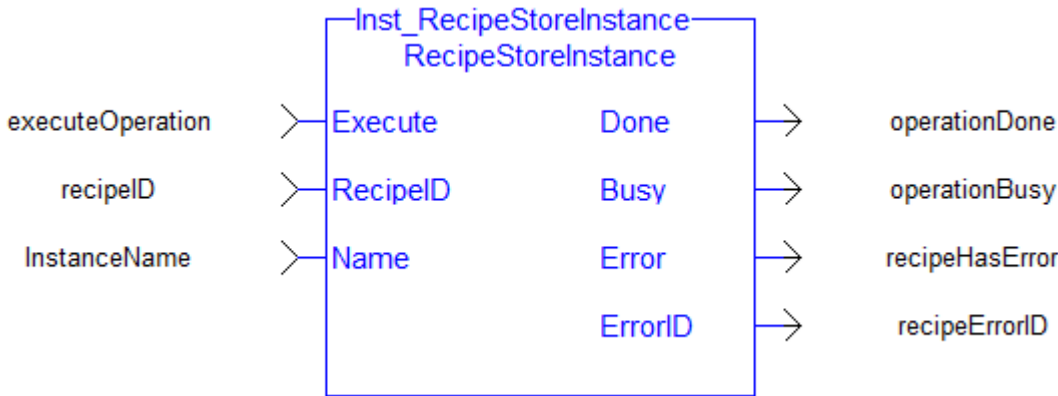
Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL	FALSE, TRUE	N/A	If TRUE, the command completed successfully.
Busy	BOOL	FALSE, TRUE	N/A	If TRUE, the function block is executing.
Error	BOOL	FALSE, TRUE	N/A	If TRUE, an error has occurred.
ErrorID	DINT	Enumerated	N/A	• Indicates the error if Error output is TRUE. • See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Inst_RecipeStoreInstance(True, recipeID, 'Cogs');  
IF Inst_RecipeStoreInstance.Error THEN  
    errorMessage := MC_ErrorDescription(Inst_RecipeStoreInstance.ErrorID);  
END_IF;
```

4.11 TCP/IP Function Blocks

This table lists the function blocks used to communicate via TCP/IP.

Name	Description
TcpAccept	Performs the accept operation.
TcpBinReceive	Receives characters over a socket connection.
TcpBinSend	Sends characters over a socket.
TcpClose	Closes and releases a socket.
TcpConnect	Creates a new socket and performs the connect operation.
TcpIsConnected	Tests if a client socket is connected.
TcpIsValid	Tests if a socket is valid.
TcpListen	Creates a new socket by performing the bind and listen operations.
TcpReceive	Receives characters over a socket connection.
TcpSend	Sends characters over a socket.

4.11.1 TcpAccept



Function Block - Performs the accept operation.

- This function block performs the **accept** operation using default TCP settings.
- Use the "TcpClose" (→ p. 971) function block to release the socket returned by **TcpAccept**.

TIP

It is possible that the socket becomes invalid if an error occurs in the TCP connection after this function block is called.

Use the "TcpsValid" (→ p. 978) function block after "TcpSend" (→ p. 982).

If the socket is no longer valid, close it using the "TcpClose" (→ p. 971) function block.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, accept a new socket connection.
ListenID	UDINT	N/A	N/A	No default	The ID of a server socket returned by the TcpListen function block.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
ID	UDINT			Either: • ID of a new client socket accepted. • Invalid ID if no new connection.

FBD Language Example

Not available.

FPLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```

(* TcpAccept example *)
CASE StepCounter OF
0:
  Inst_TcpAccept(TRUE, MyListenID);
  StepCounter := StepCounter + 1;
1:
  Inst_TcpAccept(TRUE, MyListenID);
  IF Inst_TcpAccept.Done THEN
    MySocketID := Inst_TcpAccept.ID;
    Inst_TcpAccept(FALSE, 0);
    StepCounter := StepCounter + 1;
  END_IF;
END_CASE;

```

See Also

- "TcpBinReceive" (→ p. 967)
- "TcpBinSend" (→ p. 969)
- "TcpClose" (→ p. 971)
- "TcpConnect" (→ p. 973)
- "TcplsConnected" (→ p. 976)
- "TcplsValid" (→ p. 978)
- "TcpListen" (→ p. 980)
- TcpReceive
- "TcpSend" (→ p. 982)

4.11.2 TcpBinReceive



Function Block - Receives characters over a socket connection.

- It is possible that the number of characters actually received is less than the number expected.
 - In this case, call this function again on the next cycle to receive the pending characters.

TIP

It is possible that the socket becomes invalid if an error occurs in the TCP connection after this function block is called.

Use the "TcplValid" (→ p. 978) function block after "TcpSend" (→ p. 982).

If the socket is no longer valid, close it using the "TcpClose" (→ p. 971) function block.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, request to perform copying a file.
ID	UDINT	N/A	N/A	No default	The ID of the client socket.
Data	USINT[]	N/A	N/A	No default	The array where the received data is stored.
MaxChar	DINT	N/A	N/A	No default	The maximum number of bytes to receive.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
NbRcv	DINT			The number of characters actually received.

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* TcpBinReceive example *)
CASE StepCounter OF
0:
  Inst_TcpBinReceive(TRUE, MySocketID, MyDataArray, 256);
  StepCounter := StepCounter + 1;
1:
  Inst_TcpBinReceive(TRUE, MySocketID, MyDataArray, 256);
  IF Inst_TcpBinReceive.Done THEN
    BytesReceived := Inst_TcpBinReceive.NbRcv;
    Inst_TcpBinReceive(FALSE, 0, MyDataArray, 0);
    StepCounter := StepCounter + 1;
  END_IF;
END_CASE;
```

See Also

- "TcpAccept" (→ p. 965)
- "TcpBinSend" (→ p. 969)
- "TcpClose" (→ p. 971)
- "TcpConnect" (→ p. 973)
- "TcplsConnected" (→ p. 976)
- "TcplsValid" (→ p. 978)
- "TcpListen" (→ p. 980)
- TcpReceive
- "TcpSend" (→ p. 982)

4.11.3 TcpBinSend



Function Block - Sends characters over a socket.

- It is possible the number of characters actually sent is less than the number expected.
 - In this case, use this function again to send the pending characters.

TIP

It is possible that the socket becomes invalid if an error occurs in the TCP connection after this function block is called.

Use the "TcpsValid" (→ p. 978) function block after "TcpSend" (→ p. 982).

If the socket is no longer valid, close it using the "TcpClose" (→ p. 971) function block.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, send a string over a socket connection.
ID	UDINT	N/A	N/A	No default	The ID of the client socket.
Data	USINT[]	N/A	N/A	No default	The data array to send.
DataSize	DINT	N/A	N/A	No default	Number of bytes in the Data array to send.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
NbSent	DINT			The number of characters actually sent.

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* TcpBinSend example *)
CASE StepCounter OF
0:
  Inst_TcpBinSend(TRUE, MySocketID, MyDataArray, 256);
  StepCounter := StepCounter + 1;
1:
  Inst_TcpBinSend(TRUE, MySocketID, MyDataArray, 256);
  IF Inst_TcpBinSend.Done THEN
    BytesSent := Inst_TcpBinSend.NbSent;
    Inst_TcpBinSend(FALSE, 0, MyDataArray, 0);
    StepCounter := StepCounter + 1;
  END_IF;
END_CASE;
```

See Also

- "TcpAccept" (→ p. 965)
- "TcpBinReceive" (→ p. 967)
- "TcpClose" (→ p. 971)
- "TcpConnect" (→ p. 973)
- "TcplsConnected" (→ p. 976)
- "TcplsValid" (→ p. 978)
- "TcpListen" (→ p. 980)
- TcpReceive
- "TcpSend" (→ p. 982)

4.11.4 TcpClose



Function Block - Closes and releases a socket.

You are responsible for closing any socket created by the "TcpListen" (→ p. 980), "TcpAccept" (→ p. 965), or "TcpConnect" (→ p. 973) function blocks, even if they have become invalid.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, close a socket.
ID	UDINT	N/A	N/A	No default	The ID of the client socket.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* TcpClose example *)
CASE StepCounter OF
0:
    Inst_TcpClose(TRUE, MySocketID);
    StepCounter := StepCounter + 1;
1:
    Inst_TcpClose(TRUE, MySocketID);
    IF Inst_TcpClose.Done THEN
        Inst_TcpClose(FALSE, 0);
```

```
        StepCounter := StepCounter + 1;  
    END_IF;  
END_CASE;
```

See Also

- "TcpAccept" (→ p. 965)
- "TcpBinReceive" (→ p. 967)
- "TcpBinSend" (→ p. 969)
- "TcpConnect" (→ p. 973)
- "TcplsConnected" (→ p. 976)
- "TcplsValid" (→ p. 978)
- "TcpListen" (→ p. 980)
- TcpReceive
- "TcpSend" (→ p. 982)

4.11.5 TcpConnect



Function Block - Creates a new socket and performs the connect operation.

- Use the "TcpClose" (→ p. 971) function block to release the socket returned by **TcpConnect**.
- This function block uses the default TCP settings and a specified server address and port.

TIP

It is possible that the function returns a valid socket ID even if the connection to the server is not yet actually performed.
After calling this function, use the "TcplsConnected" (→ p. 976) function block to know if the connection is ready.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, request to connect a socket to a server.
Address	STRING	N/A	N/A	No default	The IP Address of the remote server.
Port	DINT	N/A	User units	No default	The network port to use.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
ID	UDINT			The ID of the client socket.

Remarks

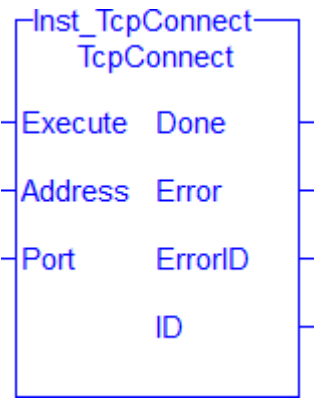


Figure 5-1: TcpConnect

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* TcpConnect example *)
CASE StepCounter OF
0:
    Inst_TcpConnect(TRUE, '192.168.1.1', 1234);
    StepCounter := StepCounter + 1;
1:
    Inst_TcpConnect(TRUE, '192.168.1.1', 1234);
    IF Inst_TcpConnect.Done THEN
        MySocketID := Inst_TcpConnect.ID;
        Inst_TcpConnect(FALSE, '', 0);
        StepCounter := StepCounter + 1;
    END_IF;
END_CASE;
```

See Also

- "TcpAccept" (→ p. 965)
- "TcpBinReceive" (→ p. 967)
- "TcpBinSend" (→ p. 969)
- "TcpClose" (→ p. 971)
- "TcpsIsConnected" (→ p. 976)
- "TcpsIsValid" (→ p. 978)
- "TcpListen" (→ p. 980)

- TcpReceive
- "TcpSend" (→ p. 982)

4.11.6 TcplsConnected



Function Block - Tests if a client socket is connected.

TIP

It is possible that the socket becomes invalid if an error occurs in the TCP connection after this function block is called.
Use the "TcplsValid" (→ p. 978) function block after "TcpSend" (→ p. 982).
If the socket is no longer valid, close it using the "TcpClose" (→ p. 971) function block.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, test whether a socket is connected.
ID	UDINT	N/A	N/A	No default	The ID of the client socket.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
Connected	BOOL			TRUE if a connection is correctly established.

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* TcpIsConnected example *)
CASE StepCounter OF
0:
```

```

    Inst_TcpIsConnected(TRUE, MySocketID);
    StepCounter := StepCounter + 1;
1:
    Inst_TcpIsConnected(TRUE, MySocketID);
    IF Inst_TcpIsConnected.Done THEN
        MyTcpIsConnected := Inst_TcpIsConnected.Connected;
        Inst_TcpIsConnected(FALSE, 0);
        StepCounter := StepCounter + 1;
    END_IF;
END_CASE;

```

See Also

- "TcpAccept" (→ p. 965)
- "TcpBinReceive" (→ p. 967)
- "TcpBinSend" (→ p. 969)
- "TcpClose" (→ p. 971)
- "TcpConnect" (→ p. 973)
- "TcplsValid" (→ p. 978)
- "TcpListen" (→ p. 980)
- TcpReceive
- "TcpSend" (→ p. 982)

4.11.7 TcplsValid



 **Function Block** - Tests if a socket is valid.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, request to perform copying a file.
ID	UDINT	N/A	N/A	No default	The ID of the client socket.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
Valid	BOOL			TRUE if the specified socket is still valid.

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```
(* TcpIsValid example *)
CASE StepCounter OF
0:
  Inst_TcpIsValid(TRUE, MySocketID);
  StepCounter := StepCounter + 1;
1:
  Inst_TcpIsValid(TRUE, MySocketID);
  IF Inst_TcpIsValid.Done THEN
```

```
MyTcpIsValid := Inst_TcpIsValid.Valid;  
Inst_TcpIsValid(FALSE, 0);  
StepCounter := StepCounter + 1;  
END_IF;  
END_CASE;
```

See Also

- "TcpAccept" (→ p. 965)
- "TcpBinReceive" (→ p. 967)
- "TcpBinSend" (→ p. 969)
- "TcpClose" (→ p. 971)
- "TcpConnect" (→ p. 973)
- "TcpIsConnected" (→ p. 976)
- "TcpListen" (→ p. 980)
- TcpReceive
- "TcpSend" (→ p. 982)

4.11.8 TcpListen



 **Function Block** - Creates a new socket by performing the bind and listen operations.

- This function block uses the default TCP settings.
- Use the "TcpClose" (→ p. 971) function block to release the socket returned by **TcpListen**.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, listen for socket connections.
Port	DINT	N/A	User units	No default	The network port to use.
Backlog	DINT	N/A	N/A	No default	The size of the queue for pending connections. If more than Backlog number of connection attempts are made prior to a "TcpAccept" (→ p. 965) call, the controller may refuse the connections.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
ListenID	UDINT			The ID of the new listen socket.

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```

(* TcpListen example *)
CASE StepCounter OF
0:
  Inst_TcpListen(TRUE, 1234, 2);
  StepCounter := StepCounter + 1;
1:
  Inst_TcpListen(TRUE, 1234, 2);
  IF Inst_TcpListen.Done THEN
    MyListenID := Inst_TcpListen.ListenID;
    Inst_TcpListen(FALSE, 0, 0);
    StepCounter := StepCounter + 1;
  END_IF;
END_CASE;

```

See Also

- "TcpAccept" (→ p. 965)
- "TcpBinReceive" (→ p. 967)
- "TcpBinSend" (→ p. 969)
- "TcpClose" (→ p. 971)
- "TcpConnect" (→ p. 973)
- "TcplsConnected" (→ p. 976)
- "TcplsValid" (→ p. 978)
- TcpReceive
- "TcpSend" (→ p. 982)

4.11.9 TcpSend



Function Block - Sends characters over a socket.

- It is possible the number of characters actually sent is less than the number expected.
 - In this case, use this function again to send the pending characters.

TIP

It is possible that the socket becomes invalid if an error occurs in the TCP connection after this function block is called.

Use the "TcplValid" (→ p. 978) function block after "TcpSend" (→ p. 982).

If the socket is no longer valid, close it using the "TcpClose" (→ p. 971) function block.

Inputs

Input	Data Type	Range	Unit	Default	Description
Execute	BOOL	0 to 1	N/A	No default	On the rising edge, send a string over a socket connection.
ID	UDINT	N/A	N/A	No default	The ID of the client socket.
NbChar	DINT	N/A	N/A	No default	The number of characters to send.
Data	STRING	N/A	N/A	No default	The IP Address of the remote server.

Outputs

Output	Data Type	Range	Unit	Description
Done	BOOL			If TRUE, the command completed successfully.
Error	BOOL			If TRUE, an error has occurred.
ErrorID	DINT			Indicates the error if Error output is TRUE. See the table in "File, Recipe, and TCP/IP Function Block ErrorIDs" (→ p. 1005).
NbSent	DINT			The number of characters actually sent.

FBD Language Example

Not available.

FFLD Language Example

Not available.

IL Language Example

Not available.

ST Language Example

```

(* TcpSend example *)
CASE StepCounter OF
0:
  Inst_TcpSend(TRUE, MySocketID, 5, 'Hello');
  StepCounter := StepCounter + 1;
1:
  Inst_TcpSend(TRUE, MySocketID, 5, 'Hello');
  IF Inst_TcpSend.Done THEN
    BytesSent := Inst_TcpSend.NbSent;
    Inst_TcpSend(FALSE, 0, 0, '');
    StepCounter := StepCounter + 1;
  END_IF;
END_CASE;

```

See Also

- "TcpAccept" (→ p. 965)
- "TcpBinReceive" (→ p. 967)
- "TcpBinSend" (→ p. 969)
- "TcpClose" (→ p. 971)
- "TcpConnect" (→ p. 973)
- "TcplsConnected" (→ p. 976)
- "TcplsValid" (→ p. 978)
- "TcpListen" (→ p. 980)
- TcpReceive

4.12 UDP Functions / Function Blocks

UDP is a communications protocol which allows computers to exchange messages across an IP network.

When a UDP packet is sent to a broadcast address (e.g., 255.255.255.255), the controller automatically converts the given broadcast address to the broadcast address of its Ethernet interface.

Example

If the controller's IP address is 192 . 168 . 1 . 10 and the subnet mask is 255 . 255 . 255 . 0, the controller's Ethernet interface broadcast address is 192 . 168 . 1 . 255.

4.12.1 UDP Functions

This list of UDP functions allows a controller to communicate with a remote PC or another KAS controller over an Ethernet network.

Name	Description
udpAddrMake	Build an address buffer for UDP functions.
udpClose	Closes a socket.
udpCreate	Creates a UDP socket.
udpIsValid	Tests if a socket is valid.
udpRcvFrom	Receives a UDP telegram.
udpRcvFromArray	Receives a byte array through UDP.
udpRcvFromVar	Receives the contents of a variable from another controller through UDP.
udpSendTo	Sends a UDP telegram.
udpSendToArray	Sends a byte array through UDP.
udpSendToVar	Sends the contents of a local variable to another controller through UDP.

TIP

See this Wikipedia page about [User Datagram Protocol](#).

4.12.2 udpAddrMake



Function - Build an address buffer for UDP functions.

This function:

- Builds an address buffer for UDP functions.
- Is required for building an internal UDP address to be passed to the "udpSendTo" (→ p. 996) function in case of UDP client processing.

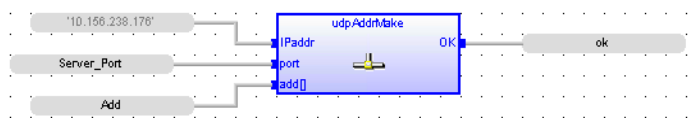
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	0 to 1	N/A	No default	Execute the function.
IPAddr	STRING	0.0.0.0, 255.255.255.255	N/A	No default	IP address in the form XXX.XXX.XXX.XXX.
port	DINT	0 to 65535	N/A	No default	IP port number.
add[]	USINT	0 to 32	N/A	No default	Buffer containing the UDP address (filled on output).

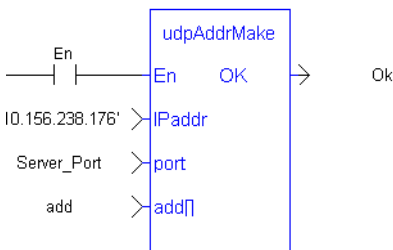
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
bAddrMake := udpAddrMake('10.156.238.176',Server_Port,add); //server details
```

4.12.3 udpClose



 **Function** - Closes a socket.

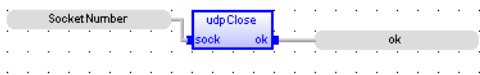
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	0 to 1	N/A	No default	Execute the function.
sock	DINT	0 to 65535	N/A	No default	ID of the socket.

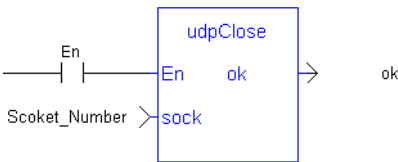
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
udpClose(Socket_Number); //Close socket
```

4.12.4 udpCreate



Function - Creates a UDP socket.

Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	0 to 1	N/A	No default	Execute the function.
port	DINT	0 to 65535	N/A	No default	UDP port number attached to the server socket or 0 for a client socket.

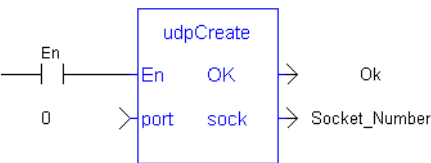
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •
sock	DINT	0 to 65535	N/A	ID of the socket.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
Socket_Number := udpCreate(Client_Port); //create a socket
```

4.12.5 udplsValid



 **Function** - Tests if a socket is valid.

Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	0 to 1	N/A	No default	Execute the function.
sock	DINT	0 to 65535	N/A	No default	ID of the socket.

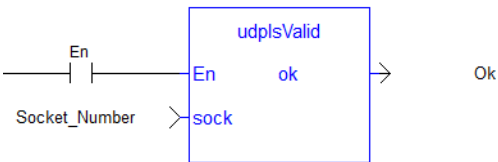
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
bIsValid := udpIsValid(Socket_Number); //Valid socket?
```

4.12.6 udpRcvFrom



Function - Receives a UDP telegram.

- If the characters are received, the function fills the ADD argument with the internal UDP of the sender.
- This buffer can be passed to "udpSendTo" (→ p. 996) to send the answer.

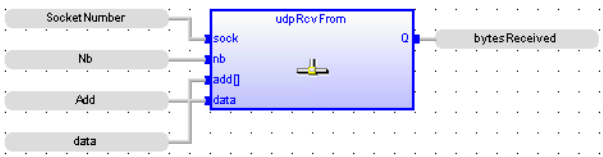
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	0 to 1	N/A	No default	Execute the function.
sock	DINT	0 to 65535	N/A	No default	ID of the socket.
nb	DINT	0 to 65535	N/A	No default	Maximum number of characters received.
add[]	USINT	0 to 32	N/A	No default	Buffer containing the UDP address (filled on output).
data	STRING	0 to 255	N/A	No default	Buffer to store received characters.

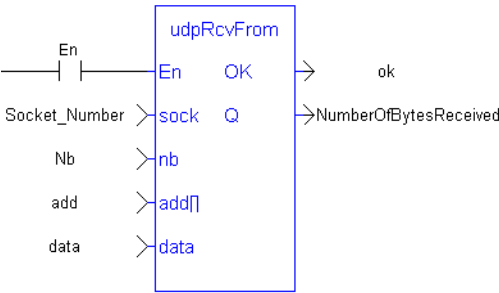
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •
Q	DINT		N/A	Actual number of received characters.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
ReceivedBytes := udpRcvFrom(Socket_Number, 5, add, data); //Read the position
```

4.12.7 udpRcvFromArray



Function - Receives a byte array through UDP.

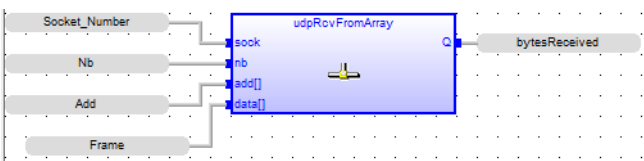
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	0 to 1	N/A	No default	Execute the function.
sock	DINT	0 to 65535	N/A	No default	Socket number, return value from "udpCreate" (→ p. 988).
nb	DINT	0 to 65535	N/A	No default	Number of bytes to be transferred.
add[]	USINT	0 to 32	N/A	No default	Array which contains information about the server.
data[]	USINT	0 to 65535	N/A	No default	Array of bytes to be transferred.

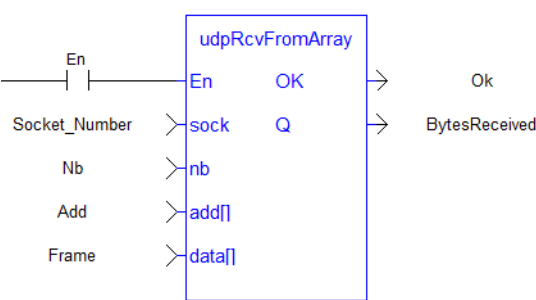
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •
Q	DINT		N/A	Number of bytes received.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
BytesReceived := udpRcvFromArray(Socket_Number,nb,add,Frame);
```


4.12.8 udpRcvFromVar



 **Function** - Receives the contents of a variable from another controller through UDP.

- Receives the contents a variable sent from another controller and saves it to a local variable.
- This allows for the exchange of data across controllers.

[Click to view Limitations](#)

 TIP

Limitations

- Function block instance variable types are not supported.
- These types of variables cannot be sent or received:
 - Variables defined with a UDFB.
 - The Input and Output variables defined for a sub-program.
- "udpSendToVar" (→ p. 1000) and "udpRcvFromVar" (→ p. 994) do **not** automatically swap bytes for big vs. little endian systems.
- Send / receive functionality is:

	PCMM2G	Simulator
PCMM2G		
Simulator		

- Compatibility is based on the endianness of the controller's information.
 - PCMM2G and Simulator are little-endian.

Inputs

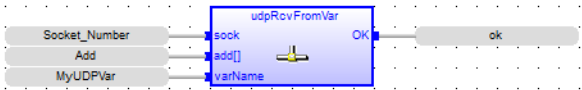
Input	Data Type	Range	Unit	Default	Description
En	BOOL	0 to 1	N/A	No default	Execute the function.
sock	DINT	0 to 65535	N/A	No default	Socket number, return value from "udpCreate" (→ p. 988).
add[]	USINT	0 to 32	N/A	No default	Array which contains information about the server. This includes the sender's IP address.
varName	STRING	N/A	N/A	No default	• The name of a PLC variable (or array or structure) that stores data from the sender. • The variable should be the same type as what is being sent. • See "udpSendToVar" (→ p. 1000).

Outputs

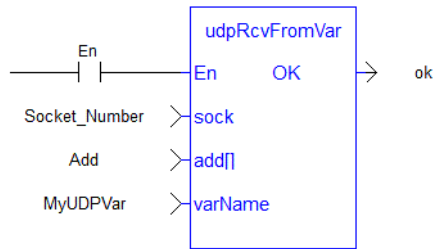
Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	Returns TRUE when the function successfully executes.

Output	Data Type	Range	Unit	Description
Q	DINT	0 to 65535	N/A	Number of bytes received.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
udpRcvFromVar( Socket_Number, Add, MyUDPVar )
```

4.12.9 udpSendTo



 **Function** - Sends a UDP telegram.

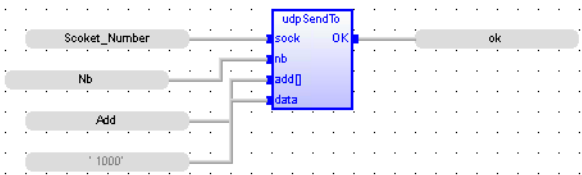
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	0 to 1	N/A	No default	Execute the function.
sock	DINT	0 to 65535	N/A	No default	ID of the client socket.
nb	DINT	0 to 65535	N/A	No default	Number of bytes of data to send.
add[]	USINT	0 to 32	N/A	No default	Buffer containing the UDP address (filled on output).
data	STRING	0 to 255	N/A	No default	The characters to send.

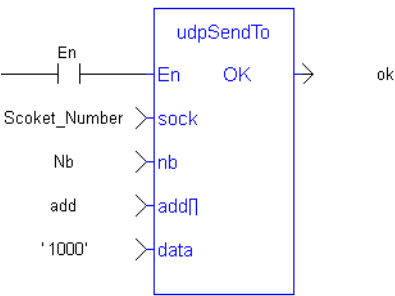
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
bUdpSendTo := udpSendTo(Socket_Number,5,add,'1000');
```

4.12.10 udpSendToArray



Function - Sends a byte array through UDP.

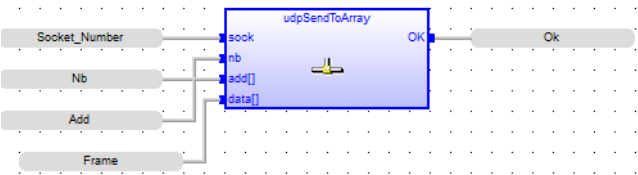
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	0 to 1	N/A	No default	Execute the function.
sock	DINT	0 to 65535	N/A	No default	Socket number, return value from "udpCreate" (→ p. 988).
nb	DINT	0 to 65535	N/A	No default	Number of bytes to be transferred.
add[]	USINT	0 to 32	N/A	No default	Array which contains information about the server.
data[]	USINT	0 to 65535	N/A	No default	Array of bytes to be transferred.

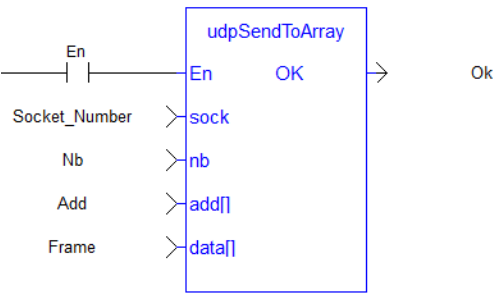
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	• Returns TRUE when the function successfully executes. •

FBD Language Example



FPLD Language Example



IL Language Example

Not available.

ST Language Example

```
Success := udpSendToArray(Socket_Number, nb, add, Frame);
```

4.12.11 udpSendToVar



 **Function** - Sends the contents of a local variable to another controller through UDP.

- Sends the contents of a local variable to another controller.
- This allows for the exchange of data across controllers.

[Click to view Limitations](#)

 TIP

Limitations

- Function block instance variable types are not supported.
- These types of variables cannot be sent or received:
 - Variables defined with a UDFB.
 - The Input and Output variables defined for a sub-program.
- "udpSendToVar" (→ p. 1000) and "udpRcvFromVar" (→ p. 994) do **not** automatically swap bytes for big vs. little endian systems.
- Send / receive functionality is:

	PCMM2G	Simulator
PCMM2G		
Simulator		

- Compatibility is based on the endianness of the controller's information.
 - PCMM2G and Simulator are little-endian.

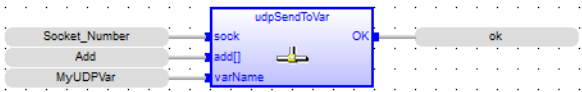
Inputs

Input	Data Type	Range	Unit	Default	Description
En	BOOL	0 to 1	N/A	No default	Execute the function.
sock	DINT	0 to 65535	N/A	No default	Socket number, return value from "udpCreate" (→ p. 988).
add[]	USINT	0 to 32	N/A	No default	• Array which contains information about the server. • This includes the sender's IP address.
varName	STRING	N/A	N/A	No default	• The data of the PLC variable (or array or structure) to send to the receiver. • The variable should be the same type as what is expected by the receiver. • See "udpRcvFromVar" (→ p. 994).

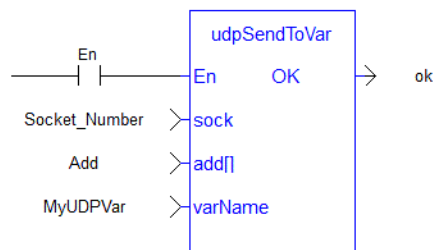
Outputs

Output	Data Type	Range	Unit	Description
OK	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
udpSendToVar( Socket_Number, Add, MyUDPVar )
```


4.13 PrintMessage

PLCopen ✓

Pipe Network ✓

 **Function** - Generates a log message with any wanted strings in the Log Messages window.

4.13.0.1 Source

PrintMessage uses the SYSTEM message type.

To view all messages generated by PrintMessage, go to the log configuration and select the specified level for the SYSTEM source.

4.13.0.2 Level

The message could be sent with a logging level from 0 to 4 that qualifies its importance.

- The highest level, 4, logs critical messages.
 - Available levels are: Debug, Informational, Warning, Error, and Critical.
- Only Error and Critical messages are generated by default.
 - To force the system to generate every message level, use the Configuration Settings to change the settings to the desired level.

ⓘ IMPORTANT

- Enabling all messages can slow down the application's execution.
- To avoid locking up communications between the IDE and Runtime, you must never include a print statement in your program that prints to the log every update cycle.

4.13.0.3 Usage

```
PrintMessage( LEVEL_DEBUG, 'Message string to be logged' );
```

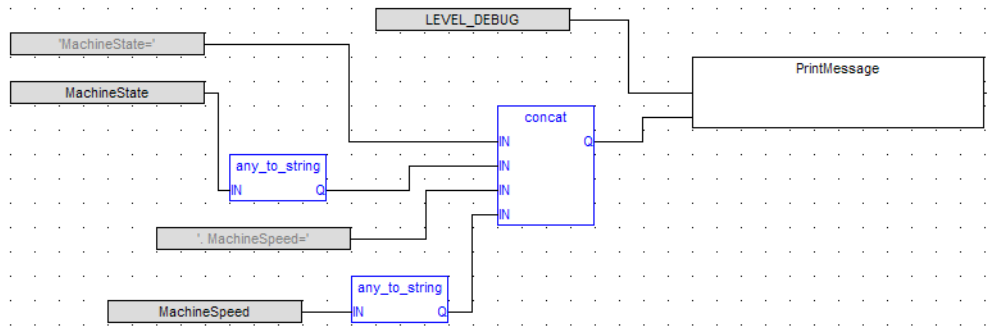
Inputs

Input	Data Type	Range	Unit	Default	Description
Level	DINT	0 to 4	N/A	No default	Level of the logged message (i.e., its importance). • By default, not all messages are shown in the log windows. • Only Error and Critical messages are shown. • Change the log settings to display a lower level. • PrintMessage logs SYSTEM messages. The Range defines are: • 0 = LEVEL_DEBUG • 1 = LEVEL_INFO • 2 = LEVEL_WARNING • 3 = LEVEL_ERROR • 4 = LEVEL_CRITICAL
Message	STRING	1 to 255			• Content of the message. • The string is a maximum of 255 characters.

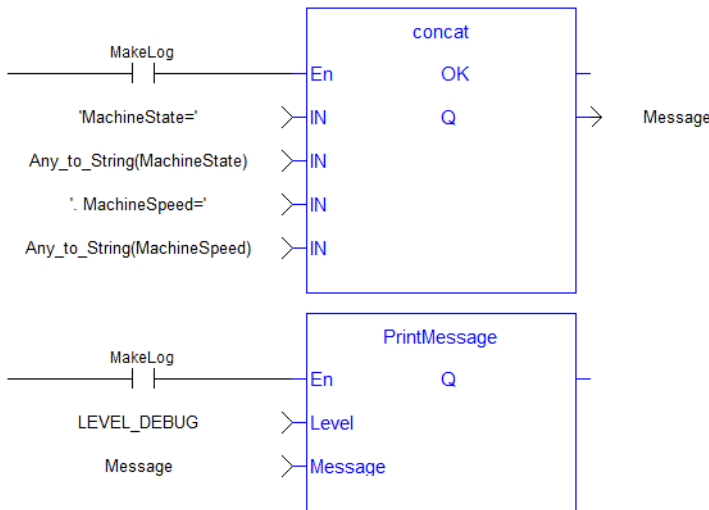
Outputs

Output	Data Type	Range	Unit	Description
Default (.Q)	BOOL	FALSE, TRUE	N/A	- Returns TRUE when the function successfully executes.

FBD Language Example



FFLD Language Example



IL Language Example

Not available.

ST Language Example

```
// It's possible to create a temporary variable with the message.

MESSAGE := CONCAT( 'MachineState=', ANY_TO_STRING(MachineState), '.
MachineSpeed=', ANY_TO_STRING(MachineSpeed) );

// Then print the message to the log window
```

```
PrintMessage( LEVEL_INFO,      MESSAGE );  
PrintMessage( LEVEL_WARNING,   MESSAGE );  
PrintMessage( LEVEL_ERROR,     MESSAGE );  
  
// Or to create the string directly in the function call:  
PrintMessage( LEVEL_CRITICAL, CONCAT( 'MachineState=', ANY_TO_STRING  
(MachineState), '. MachineSpeed=', ANY_TO_STRING(MachineSpeed) ) );
```

4.14 File, Recipe, and TCP/IP Function Block ErrorIDs

This is the list of possible errors that could be returned at the **ErrorID** output of the [File Function Blocks](#), [Recipe Function Blocks](#), and [TCP/IP Function Blocks](#).

- "File ErrorIDs" (→ p. 1005)
- "Recipe ErrorIDs" (→ p. 1006)
- "TCP/IP ErrorIDs" (→ p. 1006)

4.14.1 File ErrorIDs

ErrorID	Description
16000	Error opening file.
16001	All PLC file handles used.
16002	File ID is invalid.
16003	File ID has been closed.
16004	Error in file stream.
16005	End of file encountered.
16006	Internal file FB error.
16007	Unknown file error.
16008	No such file or directory.
16009	Bad file descriptor.
16010	File already exists.
16011	Too many open files in the system.
16012	Text file is busy.
16013	File is too large.
16014	File system is read only.
16015	File locking deadlock.
16016	File name is too long.
16017	File positioning error.
16018	Bad or corrupted file system detected .

4.14.2 Recipe ErrorIDs

ErrorID	Description
16008	No such file or directory.
16019	File inaccessible. (e.g., permission, locks)
16020	Error opening the file for writing.
16200	Error parsing recipe file.
16201	Invalid recipe file format.
16202	Recipe variable not found.
16203	Recipe variable type mismatch.
16204	Recipe variable value out of range.
16205	Invalid recipe file format.
16206	Recipe variable not found.
16207	Duplicate recipe instance name.
16208	Invalid recipe variable definition.
16219	Internal recipe FB error.

4.14.3 TCP/IP ErrorIDs

ErrorID	Description
16100	Connection error.
16101	TCP ID has been closed.
16102	All PLC TCP handles used.
16103	TCP ID is invalid.
16104	Failed to allocate TCP/IP socket.
16105	TCP operation internal error.

5 Data Types

Data types are defined within the common elements of IEC 61131-3.

Why Data Typing?

Data typing is implemented to define the type of any parameter used. This helps prevent errors early on in the programming phase (e.g., avoids dividing a Date by an Integer).

When you have defined whether the data is a string, a date, an integer or a 16-bit Boolean input, there is no longer any confusion, nor any conflict between different people using the textual representation (i.e., the name of the variable).

5.1 Different Data Types

- **Common Data Types:** Boolean, Byte, Date, Integer, Real, String, Time_of_Day, Word.
- **Derived Data Types:** Define personal data types based on the Common data types.
 - Example: Define an analog input channel as a data type and re-use it.

5.2 List of Data Types

Type	Prefix	Description	Values
BOOL		Boolean (bit)	• FALSE, TRUE • Stored in 1 byte.
BYTE		Same as "USINT" (→ p. 1008).	
DINT		Signed double precision integer in 32-bits.	-2147483648 to 2147483647
DWORD		Same as "UDINT" (→ p. 1007).	
INT	INT#	Signed integer in 16-bits.	-32768 to +32767
LINT	LINT#	Long signed integer in 64-bits.	
LREAL ‡	LREAL#	Double precision floating point stored in 64-bits.	• -1.7E308 to 1.7E308 • 14 to 15 significant digits of accuracy
LWORD		Same as "ULINT" (→ p. 1007).	
REAL ‡	REAL# ‡‡	Single precision floating point stored in 32-bits.	• -3.4E38 to 3.4E38 • 6 to 7 significant digits of accuracy.
SINT	SINT#	Small signed integer in 8-bits.	-128 to +127
STRING		• Variable length string with declared maximum length. • Each character is store on 1 byte (i.e., on 8-bits).	Maximum length cannot exceed 255 characters.
TIME	T# or TIME#	Time data type is used to specify a time variable. • Accuracy is 1ms.	0ms - 24hr
UDINT	UDINT#	Unsigned integer in 32-bits.	0 to +4294967295
UINT	UINT#	Unsigned integer in 16-bits.	0 to 65535
ULINT	ULINT#	Long unsigned integer in 64-bits.	

Type	Prefix	Description	Values
USINT	USINT#	Small unsigned integer in 8-bits.	0 to 255
WORD		Same as "UINT" (→ p. 1007).	
WSTRING		- Do not use! - The WSTRING is not supported.	

NOTE

- ‡ **REAL** variables are limited to 6 digits of accuracy.
 - To achieve greater accuracy, a longer mantissa may be specified by prefixing **LREAL** with #.
- Example: To achieve an accuracy of 20 digits for the value of Pi, rather than what **REAL** provides (3.14159), set the type to **LREAL#3.141592653589793238**.

IMPORTANT

‡ **REAL** is restrictive, but because it is the default, it is recommended to explicitly declare real constants with the **LREAL#** prefix.

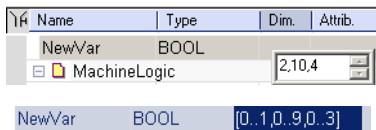
TIP

Use **2#**, **8#**, or **16#** prefixes to specify an integer in binary, octal or hexadecimal basis, respectively.

5.3 Arrays

You can declare arrays for internal variables by specifying dimension(s).

To declare an array, enter the number of elements in the **Dim.** column on the Variables tab.



For a multi-dimensional array (arrays have a maximum of three dimensions), enter the number of elements for each dimension separated by commas:

- Using **2,10,4** as the 3D array:
 - The first dimension has 2 elements.
 - The second dimension has 10 elements.
 - The third dimension has 4 elements.

Arrays are accessed by variable indices.

```
myArray[myIndex1, myIndex2]
```

IMPORTANT

- All indexes are 0 based.
- Example: In a single dimension array, the first element is always identified by `ArrayName[0]`.

5.4 Bitfields

A bitfield is a user-defined data type that combines multiple Boolean or integer values into a single variable.

- Bit Fields are used to define custom variable types and values.
- A bitfield packs multiple pieces of data together in one variable.
- Each field represents an independent piece of information.
 - These fields can be referenced individually within the application.

- Bitfields provide a compact and organized way to group related status and control information while maintaining easy access to individual fields.

Example

A bitfield type named AxisStatus may contain these fields:

Initialized
Enabled
Moving
Faulted

See the [Bit Fields tab](#) about creating bit fields and examples of their use.

5.5 Enumerations

An enumeration (enum) is a user-defined data type that contains a predefined set of named values.

- An enum variable can hold only one value from the defined list at a time.
- Enumerations are useful when a variable should only represent one value from a known set of states, modes, or selections.

Example

Enumeration can be used to represent machine states such as:

Idle
Homing
Running
Faulted

- Using enumerations improves program readability and maintainability by replacing numerical values with meaningful names.
- This makes application logic easier to understand, debug, and maintain.
- See the [Enums tab](#) about creating enumerations and examples of their use.

5.6 Structure

A structure is a complex data type defined as a set of members.

- Members of a structure can have various data types.
 - A member of a structure can have dimensions or can be an instance of another structure.
- When a structure is defined, it can be used like other data types to declare variables.
- Members of a structure can have an initial value.
 - Corresponding members of all declared variables having this structure type are initialized with the initial value of the member.
- To specify a member of a structured variable in PLC languages, use this notation:

```
VariableName.MemberName
```

5.6.1 Limitation

If a member of a structure is an instance of another structure, the nested structure must be declared **BEFORE** in the list.

6 Kollmorgen Documentation Legal Information

The information in these pages is only for the technical documentation (e.g., Safety Notes, User Manuals, etc.) legal information.

This information does not represent Kollmorgen company-wide legal information.

6.1 KAS - Copyrights, Trademarks, Patents, Licenses, and Disclaimers	1011
6.2 Kollmorgen Trademarks	1011
6.3 Kollmorgen Patents	1011
6.4 PCMM2G	1012
6.5 Other Trademarks	1013
6.6 3rd-party Licenses	1014
6.7 Disclaimers	1016

6.1 KAS - Copyrights, Trademarks, Patents, Licenses, and Disclaimers

Copyright 2026 Regal Rexnord Corporation, All Rights Reserved.

Information in this document is subject to change without notice. The software package described in this document is furnished under a license agreement. The software package may be used or copied only in accordance with the terms of the license agreement.

This document is the intellectual property of Kollmorgen and contains proprietary and confidential information. The reproduction, modification, translation or disclosure to third parties of this document (in whole or in part) is strictly prohibited without the prior written permission of Kollmorgen.

Copyright © 2009-2026 Regal Rexnord Corporation, All Rights Reserved.

6.2 Kollmorgen Trademarks

Regal Rexnord and [Kollmorgen](#) are trademarks of [Regal Rexnord Corporation](#) or one of its affiliated companies.

These trademarks or trade names are owned by or under the control of Regal Rexnord Corporation or any of its affiliates:

• Kollmorgen Essentials System	• BLM	• P8000
• Kollmorgen Essentials Cable	• Cartridge DDR	• PCMM
• Kollmorgen Essentials Drive	• DDL	• PCMM2G
• Kollmorgen Essentials Motor	• DDR	• PowerMax
• AKD	• Goldline	• Servostar
• AKD2G	• Kollmorgen Automation Suite	• SafeMotion Monitor
• AKD PDMM	• KAS	• SMM
• AKM	• Kollmorgen Visualization Builder	• TBM
• AKMA	• KVB	• TBM2G
• AKME	• MKD	• WorkBench
• AKMH	• Motioneering	

6.3 Kollmorgen Patents

Any of these Kollmorgen patents may be used and/or identified in Kollmorgen products, firmware, and/or software.

- US Patent 2017/0211640 (method and apparatus for power saving, fail-safe control of an electromechanical brake), patent pending.
- US Patent 8,154,228 (Dynamic Braking For Electric Motors).
- US Patent 8,214,063 (Auto-tune of a Control System Based on Frequency Response).
- US Patent 8,566,415 (Safe Torque Off over network wiring).
- US Patent 10,374,468 (System and method for improved DC power line communication).
- US Patent 10,520,050 (Method and apparatus for power-saving, fail-safe control of an electromechanical brake).
- US Patent 10,840,696 (Method and apparatus for limiting the output voltages of switch mode power supplies).
- US Patent 16,247,478 (method and apparatus for limiting the output voltages of switching mode power supplies), patent pending.

Patents referring to fieldbus functions are listed in the matching fieldbus manual.

6.4 PCMM2G

The PCMM2G's Operating System (OS) and boot loader ([U-Boot](#)) are based on free and open-source software.

- The individual copyright notices, license texts, and disclaimers of warranty for the software components are located in the controller, under the directory: /usr/share/common-licenses/.
 - For more information about accessing the PCMM2G's files, see [SSH Login to a Controller](#).
- The OS, bootloader, and their software component's source codes including modifications, copyright notices, license texts, disclaimers of warranty, and the compilation scripts to build the OS image are available at the Kollmorgen webpage [PCMM2G Operating System \(OS\) License Sources](#).
- The OS image and its corresponding sources file is identified by an "OS-Sources" designator, followed by its version number: OS-Sources-x.xx.x.xxxxx.
- The compilation scripts and sources file are available at the Kollmorgen webpage [PCMM2G Operating System \(OS\) Build Sources](#).
- See [PCMM2G - File Naming Conventions](#) in the KAS online help.

6.5 Other Trademarks

Any of these other trademarks and/or trade names may be used and/or identified in Kollmorgen products, firmware, and/or software.

These are believed to be the trademarks and/or trade names of their respective owners and are not owned or controlled by Regal Rexnord Corporation:

- BiSS: iC-Haus GmbH.
- CANopen: CAN in Automation (CiA).
- DRIVE-CLiQ: Siemens Aktiengesellschaft.
- EnDat: Dr. Johannes Heidenhain GmbH.
- EtherCAT und Safety over EtherCAT: Beckhoff Automation GmbH.
- EtherCAT: Beckhoff Automation GmbH.
- EtherNet/IP Kommunikationsstack: Rockwell Automation.
- EtherNet/IP: ODVA, Inc.
- [Ghostscript](#): Artifex Software, Inc. (unter der [AGPL](#) Lizenz verbreitet).
- HIPERFACE DSL: SICK AG.
- HIPERFACE: SICK AG.
- HUMMEL: HUMMEL AG.
- Hysol: Hysol Huawei Electronics Co.,LTD.
- Instapak: Sealed Air Corporation.
- INTERCONTEC: INTERCONTEC Pfeiffer Industrie-Steckverbindungen GmbH.
- LOCTITE 640: Henkel AG & CO. KGAA.
- LOCTITE: Henkel AG & CO. KGAA.
- Modbus: Schneider Electric USA, Inc.
- MOLYKOTE: DDP Specialty Electronic Materials US 9, LLC.
- Mylar: DUPONT Teijin Films U.S.
- P3-topax: Henkel KGaA.
- PHOENIX CONTACT: Phoenix Contact GmbH & Co. KG.
- [PLCopen](#): An association registered with the Chamber of Commerce in The Netherlands (Reg. 40240111).
- PROFINET: PROFIBUS Nutzerorganisation e.V.
- Scotch-Weld 2214: 3M Company.
- sercos: Sercos International E.V.
- SIMATIC: Siemens Aktiengesellschaft.
- SpeedTec: Liqui-Moly GmbH or TE Connectivity Industrial GmbH.
If SpeedTec is used as a lubricant, it belongs to Liqui-Moly GmbH.
If SpeedTec is used as a connector, it belongs to TE Connectivity Industrial GmbH
- Teflon: The Chemours Company FC, LLC.
- Viton: The Chemours Company FC, LLC.
- Windows: Microsoft Corporation.

6.6 3rd-party Licenses

Any of these 3rd-party licenses may be used and/or identified in Kollmorgen products, firmware, and/or software.

Library	Product	License
7-zip	https://www.7-zip.org/	7-zip - License
Adafruit core graphics library	https://github.com/adafruit	https://github.com/adafruit/Adafruit-GFX-Library/blob/master/license.txt
Adafruit display library	https://github.com/adafruit	https://github.com/adafruit/Adafruit-ST7735-Library/blob/master/README.txt
Arduino Map function	http://creativecommons.org	http://creativecommons.org/licenses/by-sa/3.0/
C++ Mathematical Expression Library	http://www.partow.net/programming/exptrk/index.html	MIT License
civetweb: Embedded C/C++ web server	https://github.com/civetweb/civetweb	MIT License
CoreSight Library	http://www.apache.org	http://www.apache.org/licenses/LICENSE-2.0
curl software library		curl license
DockPanelSuite	http://sourceforge.net/projects/dockpanelsuite/	https://opensource.org/licenses/MIT
EtherNet/IP Communication Stack		Software Distribution License
FatFs	https://github.com/abbrev/fatfs	https://github.com/stm32duino/FatFs/blob/master/Licence.md
FileHelpers	http://sourceforge.net/projects/filehelpers	https://opensource.org/licenses/MIT
Font Awesome	https://fontawesome.com	https://fontawesome.com/license/free
Ghostscript	https://ghostscript.com/index.html	AGPL
Google Fonts Roboto Condensed	http://www.apache.org	http://www.apache.org/licenses/
GSL - Guideline Support Library	https://github.com/Microsoft/GSL	https://github.com/Microsoft/GSL/blob/master/LICENSE

Library	Product	License
libsodium	https://libsodium.org	https://github.com/jedisct1/libsodium/blob/master/LICENSE
jsoncpp software		jsoncpp License
log4net	http://logging.apache.org/log4net/	http://www.apache.org/licenses/LICENSE-2.0
LVGL	https://github.com/lvgl/lvgl	https://github.com/lvgl/lvgl/blob/master/LICENSE.txt
LwIP	https://savannah.nongnu.org	https://savannah.nongnu.org/projects/lwip/
mbd Microcontroller Library	http://www.apache.org	http://www.apache.org/licenses/LICENSE-2.0
mbd TLS	http://www.apache.org	http://www.apache.org/licenses/LICENSE-2.0
Microsoft Edge WebView2	https://developer.microsoft.com/en-us/microsoft-edge/webview2	Microsoft Edge WebView2 Runtime LICENSE
MigraDoc	http://www.pdfsharp.net/migradocoverview.ashx	http://www.pdfsharp.net/MigraDoc_License.ashx
Mongoose Library	https://github.com/cesanta/mongoose/tree/	Mongoose license
MVVM Light Toolkit	https://github.com/lbugnion/mvvmlight	https://galasoft.ch/license_MIT.txt
OpENer EtherNet/IP	https://github.com/EIPStackGroup/OpENer	http://eipstackgroup.github.io/OpENer/de/d7e/license.html
PdfSharp	http://www.pdfsharp.net/migradocoverview.ashx	http://www.pdfsharp.net/PDFsharp_License.ashx
Qt Framework	https://www.qt.io/product/framework	terms of the LGPL3
Qwt project	http://qwt.sourceforge.net/	Qwt
SharpZipLib	https://github.com/icsharpcode/SharpZipLib	https://github.com/icsharpcode/SharpZipLib/blob/master/LICENSE.txt
SNMP		http://www.net-snmp.org/about/license.html
SQLite	https://system.data.sqlite.org	https://system.data.sqlite.org/index.html/doc/trunk/www/copyright.wiki
U-Boot	http://www.denx.de/wiki/U-Boot	GNU General Public License 2.0
Zed Graph	http://sourceforge.net/projects/zedgraph/	https://opensource.org/licenses/LGPL-2.0
Zlib software library	https://www.zlib.net/	Zlib License

6.7 Disclaimers

Technical changes which improve the performance of the device may be made without prior notice!

This document is the intellectual property of Kollmorgen. All rights reserved. No part of this work may be reproduced in any form (by photocopying, microfilm or any other method) or stored, processed, copied or distributed by electronic means without the written permission of Kollmorgen.

The information in this document is believed to be accurate and reliable at the time of its release. Kollmorgen assumes no responsibility for any damage or loss resulting from the use of this help, and expressly disclaims any liability or damages for loss of data, loss of use, and property damage of any kind, direct, incidental or consequential, in regard to or arising out of the performance or form of the materials presented herein or in any software programs that accompany this document.

All timing diagrams, whether produced by Kollmorgen or included by courtesy of the PLCopen organization, are provided with accuracy on a best-effort basis with no warranty, explicit or implied, by Kollmorgen. The user releases Kollmorgen from any liability arising out of the use of these timing diagrams.

Technische Änderungen, die der Verbesserung der Geräte dienen, vorbehalten!

Dieses Dokument ist geistiges Eigentum von Kollmorgen. Alle Rechte vorbehalten. Kein Teil dieses Werkes darf in irgendeiner Form (Fotokopie, Mikrofilm oder in einem anderen Verfahren) ohne schriftliche Genehmigung von Kollmorgen reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

Die Genauigkeit und Verlässlichkeit der Informationen in diesem Dokument entspricht dem Kenntnisstand zum Zeitpunkt der Veröffentlichung. Kollmorgen übernimmt keine Verantwortung für Schäden oder Verluste, die sich aus der Verwendung dieser Hilfe ergeben, und lehnt ausdrücklich jegliche Haftung sowie Schadensersatzansprüche für Datenverlust, entgangene Nutzung und Sachschäden jeglicher Art ab, inklusive direkter, zufälliger oder Folgeschäden im Zusammenhang mit der Funktion oder Form des hierin oder in einem der mit diesem Dokument gelieferten Software enthaltenen Materials.

Es posible que se implementen sin aviso previo cambios técnicos que mejoran el rendimiento del dispositivo.

La propiedad intelectual de este documento pertenece a Kollmorgen. Todos los derechos reservados. Se prohíbe la reproducción de este documento por ningún medio (fotocopia, microfilm u otro método), así como su almacenamiento, procesamiento, copia o distribución por medios electrónicos, sin el consentimiento por escrito de Kollmorgen.

La información incluida en este documento se considera correcta y confiable en el momento de su publicación. Kollmorgen no asume ninguna responsabilidad por ningún daño o pérdida causados por el uso de este documento de ayuda y rechaza explícitamente toda responsabilidad o daño por pérdida de datos, pérdida de uso y daños en la propiedad de cualquier tipo, ya sea directos, incidentales o consecuentes, en relación con el rendimiento o la forma de los materiales aquí publicados, o con los programas de software que acompañan a este documento, o que surjan de estos.

Les modifications techniques qui améliorent les performances du dispositif peuvent être apportées sans avis préalable !

Ce document est la propriété intellectuelle de Kollmorgen. Tous droits réservés. Aucune partie de ce document ne peut être reproduite sous quelque forme que ce soit (photocopie, microfilm ou autre méthode) ni stockée, traitée, copiée ou distribuée de manière électronique sans l'autorisation écrite de Kollmorgen.

Les informations contenues dans ce document sont considérées exactes et fiables au moment de leur publication. Kollmorgen n'assume aucune responsabilité pour tout dommage ou perte résultant de l'utilisation de cette aide, et rejette expressément toute responsabilité ou tous dommages pour la perte de données, la perte d'utilisation et les dommages matériels de toute nature, directs, accessoires ou indirects, en ce qui concerne ou découlant des performances ou de la forme des matériaux présentés ici ou dans tout programme logiciel accompagnant ce document.

Modifiche tecniche volte a migliorare le prestazioni del dispositivo possono essere apportate senza preavviso.

Il presente documento è proprietà intellettuale di Kollmorgen. Tutti i diritti riservati. È vietata la riproduzione di qualsiasi parte di quest'opera in qualsiasi forma (fotocopia, microfilm o qualunque altro metodo) così come la memorizzazione, il trattamento, la copia o la distribuzione tramite mezzi elettronici senza l'autorizzazione scritta di Kollmorgen.

Le informazioni contenute nel presente documento sono ritenute accurate e affidabili al momento della pubblicazione. Kollmorgen non si assume alcuna responsabilità per eventuali danni o perdite risultanti dall'uso di questa guida e declina espressamente qualsiasi responsabilità o danni dovuti a perdita di dati, mancato utilizzo e danni materiali di qualunque tipo, diretti, accidentali o consequenziali, in relazione a o derivanti dal funzionamento o dalla forma dei materiali presentati qui o in eventuali programmi software che accompagnano il presente documento.

デバイスの性能の向上のため、予告なく技術的な変更が実装される場合があります。

本書の知的財産権はKollmorgenに帰属しており、無断複写・転載は禁じられています。本書のいずれの部分も、Kollmorgenの書面による許可なしに、いかなる複製(写真、マイクロフィルムまたはその他の手段)、保管、改変、複写または電子的配布を行うことはできません。

本書に記載されている情報は、発行時において正確かつ信頼できるものですが、Kollmorgenはこのヘルプの使用により生じるいかなる損害や損失についても責任を負いません。また、本書の履行、資料形式または付随するあらゆるソフトウェアプログラムに関連して、あるいはこれらが原因で発生する、直接的、偶発的または結果的な、いかなるデータの損失、利用機会の喪失および財産の損害に対しても、一切の責任や損害賠償を明示的に否認します。

기기의 성능을 향상시키는 기술적 변경 사항은 사전 공지 없이 이루어질 수 있습니다!

본 문서의 지적 재산권은 Kollmorgen에 있습니다. All rights reserved. 본 저작물의 어떤 부분도 Kollmorgen의 서면 허가 없이 어떠한 형태(사진 복사, 마이크로필름 또는 기타 방법)로든 복제하거나 전자 수단으로 저장, 처리, 복사 또는 배포할 수 없습니다.

본 문서의 정보는 공개 시점을 기준으로 정확하고 신뢰할 수 있는 것으로 간주됩니다. Kollmorgen은 본 도움말의 사용으로 인한 어떠한 피해나 손실에 대해 책임을 지지 않으며, 본 문서 또는 본 문서와 함께 제공되는 소프트웨어 프로그램에 제시된 자료의 성능 또는 형태와 관련하여 또는 그로 인해 직접적, 우발적 또는 결과적으로 발생한 모든 유형의 데이터 손실, 사용 손실, 재산상의 피해에 대한 책임이나 손해를 명시적으로 부인합니다.

Alterações técnicas que melhoram o desempenho do dispositivo podem ser feitas sem aviso prévio!

Este documento é propriedade intelectual da Kollmorgen. Todos os direitos reservados. Nenhuma parte deste trabalho pode ser reproduzida sob qualquer forma (por fotocópia, microfilme ou qualquer outro método) ou armazenada, processada, copiada ou distribuída por meios eletrônicos sem a permissão escrita da Kollmorgen.

Acredita-se que as informações contidas neste documento sejam precisas e confiáveis no momento de sua divulgação. A Kollmorgen não assume nenhuma responsabilidade por quaisquer danos ou perdas resultantes do uso desta ajuda, e se isenta expressamente de qualquer responsabilidade ou danos por perda de dados, perda de uso e danos materiais de qualquer espécie, direta, incidental ou subsequente em relação a ou resultante do desempenho ou da forma dos materiais apresentados aqui ou em quaisquer programas de software que acompanhem este documento.

我们可能会进行技术更改以提高设备性能，恕不另行通知！

本文档知识产权归 Kollmorgen 所有，Kollmorgen 保留所有权利。未经 Kollmorgen 书面许可，不得以任何形式（影印、缩微胶片或其他任何方式）复制本文档任何内容，亦不得以任何电子手段存储、处理、复制或分发本文档的任何内容。

据我们所知，本文档所含信息截至发布日是准确可靠的。对于因使用本帮助文档而导致的任何损害或损失，Kollmorgen 概不承担任何责任。Kollmorgen 特此明确声明，对于因本文档中的任何材料或随附本文档之任何软件程序的性能或形式导致的或与之相关的任何类型的数据丢失、使用中断损失或财产损失（包括直接、间接和衍生性损失或损害），Kollmorgen 不承担任何法律责任或损害赔偿赔偿责任。

Support and Services

About Kollmorgen

When you need motion and automation systems for your most demanding applications and environments, count on Kollmorgen - the innovation leader for more than 100 years. We deliver the industry's highest-performing, most reliable motors, drives, AGV control solutions and automation platforms, with over a million standard and easily modifiable products to meet virtually any motion challenge. We offer manufacturing facilities, distributors and engineering expertise in all major regions around the world, so you can bring a better machine to market faster and keep it profitable for many years to come.

Kollmorgen Support Network



See the [Kollmorgen Support Network](#) for product support, product downloads, knowledge base answers, and information.



Kollmorgen Support Locations

North America

Kollmorgen Corporation

201 West Rock Road
Radford, VA 24141, USA

Web: www.kollmorgen.com
Email: kollmorgen.support@regalrexnord.com
Tel.: +1-540-633-3545
Fax: +1-540-639-4162

Europe

Kollmorgen Europe GmbH

Pempelfurtstr. 1
40880 Ratingen, Germany

Web: www.kollmorgen.com
Email: kollmorgen.tech.emeai@regalrexnord.com
Tel.: +49-2102-9394-0
Fax: +49-2102-9394-3155

South America

Altra Industrial Motion do Brasil Equipamentos Industriais Ltda.

Avenida João Paulo Ablas, 2970
Jardim da Glória, Cotia – SP
CEP 06711-250, Brazil

Web: www.kollmorgen.com
Email: kollmorgen.contato@regalrexnord.com
Tel.: (+55 11) 4615-6300

China and SEA

Kollmorgen (Shanghai)

Room 1201, Building A,
Mangoo Hub, No.7 Longai Road,
Xuhui District, Shanghai, China
(Service available in English and Chinese)

Web: www.kollmorgen.cn
Email: kollmorgen.motion.sales.china@regalrexnord.com
Tel.: +86-400 668 2802