

Technical reference manual

System parameters

Controller software IRC5

RobotWare 5.0



ABB

**Technical reference manual
Controller Software IRC5
System parameters**

RobotWare 5.0

Document ID: 3HAC17076-1
Revision: E

The information in this manual is subject to change without notice and should not be construed as a commitment by ABB. ABB assumes no responsibility for any errors that may appear in this manual.

Except as may be expressly stated anywhere in this manual, nothing herein shall be construed as any kind of guarantee or warranty by ABB for losses, damages to persons or property, fitness for a specific purpose or the like.

In no event shall ABB be liable for incidental or consequential damages arising from use of this manual and products described herein.

This manual and parts thereof must not be reproduced or copied without ABB's written permission, and contents thereof must not be imparted to a third party nor be used for any unauthorized purpose. Contravention will be prosecuted.

Additional copies of this manual may be obtained from ABB at its then current charge.

© Copyright 2004-2007 ABB All rights reserved.

ABB Robotics
SE-721 68 Västerås
Sweden

Manual overview	17
Product documentation, M2004	19
1 About system parameters	21
1.1 About system parameters	21
2 Topic Communication	23
2.1 The Communication topic	23
2.2 Workflows	24
2.2.1 How to define an application protocol	24
2.2.2 How to define a physical channel	25
2.2.3 How to define a transmission protocol	26
2.3 Type Application Protocol	27
2.3.1 The Application Protocol type	27
2.3.2 Name	29
2.3.3 Type	30
2.3.4 Transmission Protocol	31
2.3.5 Server Address	32
2.3.6 Trusted	33
2.3.7 Local Path	34
2.3.8 Server Path	35
2.3.9 Username	36
2.3.10 Password	37
2.3.11 User ID	38
2.3.12 Group ID	39
2.3.13 Maximum File Size	40
2.3.14 Memory Partition Size	41
2.4 Type Physical Channel	42
2.4.1 The Physical Channel type	42
2.4.2 Name	43
2.4.3 Connector	44
2.4.4 Baudrate	45
2.4.5 Parity	46
2.4.6 Number of Bits	47
2.4.7 Number of Stop Bits	48
2.4.8 Duplex	49
2.4.9 Flow Control	50
2.5 Type Transmission Protocol	51
2.5.1 The Transmission Protocol type	51
2.5.2 Name	52
2.5.3 Type	53
2.5.4 Physical Channel	54
2.5.5 Local Address	55
2.5.6 Remote Address	56
3 Topic Controller	57
3.1 The Controller topic	57
3.2 Workflows	58
3.2.1 How to activate hold-to-run control	58
3.2.2 How to define path return region	59
3.3 Type Auto Condition Reset	60
3.3.1 The Auto Condition Reset type	60
3.3.2 Name	61

Table of Contents

3.3.3 Reset	62
3.4 Type Automatic Loading of Modules	63
3.4.1 The Automatic Loading of Modules type	63
3.4.2 File	65
3.4.3 Task	66
3.4.4 Installed	67
3.4.5 Shared.	68
3.4.6 All Tasks	69
3.4.7 Hidden	70
3.5 Type Event Routine	71
3.5.1 The Event Routine type	71
3.5.2 Routine	73
3.5.3 Event	74
3.5.4 Sequence Number	76
3.5.5 Task	77
3.5.6 All Tasks	78
3.6 Type Mechanical Unit Group	79
3.6.1 The Mechanical Unit Group type	79
3.6.2 Name	80
3.6.3 Robot	81
3.6.4 Mechanical Unit 1, 2, 3, 4, 5, 6	82
3.6.5 Use Motion Planner	83
3.7 Type ModPos Settings	84
3.7.1 The ModPos Settings type	84
3.7.2 Name	85
3.7.3 Limited ModPos.	86
3.7.4 Mode	87
3.7.5 Limit Trans.	88
3.7.6 Limit Rot	89
3.7.7 Limit External Trans	90
3.7.8 Limit External Rot	91
3.7.9 Change to LModPos in Auto	92
3.8 Type Operator Safety	93
3.8.1 The Operator Safety type	93
3.8.2 Function	94
3.8.3 Active	95
3.9 Type Path Return Region	96
3.9.1 The Path Return Region type	96
3.9.2 Mode	97
3.9.3 TCP Distance	98
3.9.4 TCP Rotation	99
3.9.5 External Distance	100
3.9.6 External Rotation	101
3.10 Type Run Mode Settings	102
3.10.1 The Run Mode Settings type	102
3.10.2 Name	103
3.10.3 Switch.	104
3.11 Type Safety Run Chain.	105
3.11.1 The Safety Run Chain type	105
3.11.2 Function	106
3.11.3 Active.	107
3.12 Type System Misc.	108
3.12.1 The System Misc type	108

3.12.2 Name	109
3.12.3 Action values	110
3.12.3.1 NoOfRetry	110
3.12.3.2 SimulateMenu	111
3.12.4 Value	112
3.13 Type Task	113
3.13.1 The Task type	113
3.13.2 Task	114
3.13.3 Task in Foreground	115
3.13.4 Type	116
3.13.5 Check Unresolved References	117
3.13.6 Main Entry	118
3.13.7 TrustLevel	119
3.13.8 Use Mechanical Unit Group	120
3.13.9 MotionTask	121
3.13.10 Hidden	122
4 Topic I/O	123
4.1 The I/O topic	123
4.2 Workflows	124
4.2.1 How to define I/O units	124
4.2.2 How to list available I/O unit types	125
4.2.3 How to define the I/O unit type	126
4.2.4 How to define input and output signals	127
4.2.5 How to define a signal group	129
4.2.6 How to define system inputs	130
4.3 Type Access Level	131
4.3.1 The Access Level type	131
4.3.2 Name	132
4.3.3 Rapid	133
4.3.4 Local Client in Manual Mode	134
4.3.5 Local Client in Auto Mode	135
4.3.6 Remote Client in Manual Mode	136
4.3.7 Remote Client in Auto Mode	137
4.4 Type Bus	138
4.4.1 The Bus type	138
4.4.2 Name	140
4.4.3 Type of Bus	141
4.4.4 Connector ID	142
4.4.5 Label at Fieldbus Connector	143
4.4.6 Automatic Bus Restart	144
4.4.7 Recovery Time	145
4.4.8 Path to Bus Configuration File	146
4.5 Type Cross Connection	147
4.5.1 The Cross Connection type	147
4.5.2 Resultant	149
4.5.3 Actor 1	150
4.5.4 Invert Actor 1	151
4.5.5 Operator 1	152
4.5.6 Actor 2	153
4.5.7 Invert Actor 2	154
4.5.8 Operator 2	155
4.5.9 Actor 3	156
4.5.10 Invert Actor 3	157

Table of Contents

4.5.11 Operator 3	158
4.5.12 Actor 4	159
4.5.13 Invert Actor 4	160
4.5.14 Operator 4	161
4.5.15 Actor 5	162
4.5.16 Invert Actor 5	163
4.6 Type Fieldbus Command	164
4.6.1 The Fieldbus Command type	164
4.6.2 Assigned to Unit	165
4.6.3 Type of Fieldbus Command	166
4.6.4 Value	167
4.7 Type Fieldbus Command Type	168
4.7.1 The Fieldbus Command Type type	168
4.7.2 Name	169
4.7.3 Type of Unit	170
4.7.4 Default Value	171
4.7.5 Download Order	172
4.8 Type Signal	173
4.8.1 The Signal type	173
4.8.2 Name	176
4.8.3 Type of Signal	177
4.8.4 Assigned to Unit	178
4.8.5 Signal Identification Label	179
4.8.6 Unit Mapping	180
4.8.7 Category	182
4.8.8 Access Level	183
4.8.9 Default Value	184
4.8.10 Store Signal Value at Power Fail	185
4.8.11 Filter Time Passive	186
4.8.12 Filter Time Active	187
4.8.13 Invert Physical Value	188
4.8.14 Analog Encoding Type	189
4.8.15 Maximum Logical Value	190
4.8.16 Maximum Physical Value	192
4.8.17 Maximum Physical Value Limit	193
4.8.18 Maximum Bit Value	194
4.8.19 Minimum Logical Value	195
4.8.20 Minimum Physical Value	196
4.8.21 Minimum Physical Value Limit	197
4.8.22 Minimum Bit Value	198
4.8.23 Signal Value at System Failure and Power Fail	199
4.9 Type System Input	200
4.9.1 The System Input type	200
4.9.2 Signal Name	201
4.9.3 Action	202
4.9.4 Action Values	203
4.9.4.1 Motors On	203
4.9.4.2 Motors On and Start	204
4.9.4.3 Motors Off	205
4.9.4.4 Load and Start	206
4.9.4.5 Interrupt	208
4.9.4.6 Start	210
4.9.4.7 Start at Main	211
4.9.4.8 Stop	212

4.9.4.9 Quick Stop	213
4.9.4.10 Soft Stop	214
4.9.4.11 Stop at End of Cycle	215
4.9.4.12 Stop at End of Instruction	216
4.9.4.13 Reset Execution Error Signal	217
4.9.4.14 Reset Emergency Stop	218
4.9.4.15 System Restart	219
4.9.5 Argument 1	220
4.9.6 Argument 2	221
4.10 Type System Output	222
4.10.1 The System Output type	222
4.10.2 Status	223
4.10.3 Signal	224
4.10.4 Action values	225
4.10.4.1 Auto On	225
4.10.4.2 Simulated I/O	226
4.10.4.3 Cycle On	227
4.10.4.4 Emergency Stop	228
4.10.4.5 Execution Error	229
4.10.4.6 Motors Off State	230
4.10.4.7 Motors On State	231
4.10.4.8 Motors Off	232
4.10.4.9 Motors On	233
4.10.4.10 Motion Supervision On	234
4.10.4.11 Motion Supervision Triggered	235
4.10.4.12 Power Fail Error	236
4.10.4.13 Path Return Region Error	237
4.10.4.14 Run Chain OK	238
4.10.4.15 TCP Speed	239
4.10.4.16 TCP Speed Reference	240
4.10.4.17 Mechanical Unit Active	241
4.10.5 Argument	242
4.11 Type Unit	243
4.11.1 The Unit type	243
4.11.2 Name	245
4.11.3 Type of Unit	246
4.11.4 Connected to Bus	247
4.11.5 Unit Identification Label	248
4.11.6 Unit Trustlevel	249
4.11.7 Unit Startup State	250
4.11.8 Store Unit State at Power Fail	251
4.11.9 Regain Communication Reset	252
4.11.10 Profibus Address	253
4.11.11 Interbus Address	254
4.12 Type Unit Type	255
4.12.1 The Unit Type type	255
4.12.2 Name	257
4.12.3 Type of Bus	258
4.12.4 Vendor Name	259
4.12.5 Product Name	260
4.12.6 Product ID	261
4.12.7 Internal Slave	262
4.12.8 Internal Slave Input Size	263
4.12.9 Internal Slave Output Size	264

Table of Contents

5 Topic Man-machine Communication	265
5.1 The Man-machine Communication topic	265
5.2 Type Most Common I/O Signal	266
5.2.1 The Most Common I/O Signal type	266
5.2.2 Signal Name	267
5.2.3 Signal Type	268
5.3 Type Production Permission	269
5.3.1 The Production Permission type	269
5.3.2 Name	270
5.3.3 Permission	271
5.4 Type Most Common Instruction	272
5.4.1 The Most Common Instruction types	272
5.4.2 Name	274
5.4.3 Parameter Number	275
5.4.4 Alternative Number	276
5.4.5 Instruction Name	277
5.4.6 Only for Motion Task	278
6 Topic Motion	279
6.1 The Motion topic	279
6.2 Workflows	281
6.2.1 How to define base frame	281
6.2.2 How to define work area	282
6.2.3 How to define arm check point	283
6.2.4 How to define arm loads	285
6.2.5 How to optimize drive system parameters	286
6.2.6 How to tune motion supervision	287
6.2.7 How to define transmission gear ratio for independent joints	288
6.2.8 How to define external torque	290
6.2.9 How to define supervision level	291
6.3 Type Acceleration Data	292
6.3.1 The Acceleration Data type	292
6.3.2 Name	293
6.3.3 Nominal Acceleration	294
6.3.4 Nominal Deceleration	295
6.3.5 Acceleration Derivate Ratio	296
6.3.6 Deceleration Derivate Ratio	297
6.4 Type Arm	298
6.4.1 The Arm type	298
6.4.2 Name	299
6.4.3 Independent Joint	300
6.4.4 Upper Joint Bound	301
6.4.5 Lower Joint Bound	302
6.4.6 Independent Upper Joint Bound	303
6.4.7 Independent Lower Joint Bound	304
6.4.8 Calibration Position	305
6.4.9 Load Id Acceleration Ratio	306
6.4.10 Performance Quota	307
6.4.11 Jam Supervision Trim Factor	308
6.4.12 Load Supervision Trim Factor	309
6.4.13 Speed Supervision Trim Factor	310
6.4.14 Position Supervision Trim Factor	311
6.4.15 External Const Torque	312

6.4.16 External Proportional Torque	313
6.4.17 External Torque Zero Angle	314
6.4.18 Angle Acceleration Ratio	315
6.5 Type Arm Check Point	316
6.5.1 The Arm Check Point type	316
6.5.2 Name	317
6.5.3 Position x, y, z	318
6.6 Type Arm Load	319
6.6.1 The Arm Load type	319
6.6.2 Name	320
6.6.3 Mass	321
6.6.4 Mass Center x, y, z	322
6.6.5 Inertia x, y, z	323
6.7 Type Brake	324
6.7.1 The Brake type	324
6.7.2 Name	325
6.7.3 Control Off Speed Limit	326
6.7.4 Control Off Delay	327
6.7.5 Brake Control On Delay	328
6.7.6 Brake Control Min Delay	329
6.7.7 Absolute Brake Torque	330
6.7.8 Brake Ramp Speed Limit	331
6.8 Type Control Parameters	332
6.8.1 The Control Parameters type	332
6.8.2 Name	333
6.8.3 Friction FFW On	334
6.8.4 Friction FFW Level	335
6.8.5 Friction FFW Ramp	336
6.9 Type Drive Module	337
6.9.1 The Drive Module type	337
6.9.2 Name	338
6.9.3 Number	339
6.10 Type Drive System	340
6.10.1 The Drive System type	340
6.10.2 Name	341
6.10.3 Use Drive Unit	342
6.10.4 Current Vector On	343
6.11 Type Force Master	344
6.11.1 The Force Master type	344
6.11.2 Name	345
6.11.3 Use Force Master Control	346
6.11.4 References Bandwidth	347
6.11.5 Use Ramp Time	348
6.11.6 Ramp when Increasing Force	349
6.11.7 Ramp Time	350
6.11.8 Collision LP Bandwidth	351
6.11.9 Collision Alarm Torque	352
6.11.10 Collision Speed	353
6.11.11 Collision Delta Position	354
6.11.12 Force Detection Bandwidth	355
6.11.13 Delay ramp	356
6.11.14 Ramp to real contact	357

Table of Contents

6.12 Type Force Master Control	358
6.12.1 The Force Master Control type	358
6.12.2 Name	360
6.12.3 No. of Speed Limits	361
6.12.4 torque 1	362
6.12.5 torque 2	363
6.12.6 torque 3	364
6.12.7 torque 4	365
6.12.8 torque 5	366
6.12.9 torque 6	367
6.12.10 Speed Limit 1	368
6.12.11 Speed Limit 2	369
6.12.12 Speed Limit 3	370
6.12.13 Speed Limit 4	371
6.12.14 Speed Limit 5	372
6.12.15 Speed Limit 6	373
6.12.16 Kv 1	374
6.12.17 Kv 2	375
6.12.18 Kv 3	376
6.12.19 Kv 4	377
6.12.20 Kv 5	378
6.12.21 Kv 6	379
6.13 Type Friction Compensation	380
6.13.1 The Friction Compensation type	380
6.13.2 Name	381
6.13.3 Friction FFW On	382
6.13.4 Friction FFW Level	383
6.13.5 Friction FFW Ramp	384
6.14 Type Jog Parameters	385
6.14.1 The Jog Parameters type	385
6.14.2 Name	386
6.14.3 Configurable Linear Step Size	387
6.14.4 Configurable Reorient Step Size	388
6.14.5 Configurable Joint Step Size	389
6.15 Type Joint	390
6.15.1 The Joint type	390
6.15.2 Name	391
6.15.3 Logical Axis	392
6.15.4 Use Process	393
6.15.5 Lock Joint in Ipol	394
6.15.6 Follower to Joint	395
6.16 Type Lag Control Master 0	396
6.16.1 The Lag Control Master 0 type	396
6.16.2 Name	397
6.16.3 Kp, Gain Position Loop	398
6.16.4 Kv, Gain Speed Loop	399
6.16.5 Ti Integration Time Speed Loop	400
6.16.6 Forced Control Active	401
6.16.7 Forced Factor for Kp	402
6.16.8 Forced Factor for Ki	403
6.16.9 Raise Time for Kp	404
6.16.10 Notch Filter Active	405
6.16.11 Notch Filter Frequency	406
6.16.12 Notch Filter Width	407

6.16.13 Notch Auto Mode	408
6.16.14 Auto No Weave Frequency	409
6.16.15 Auto Min Frequency	410
6.16.16 Auto Max Relative Change	411
6.16.17 FFW Mode	412
6.16.18 Bandwidth	413
6.16.19 Df	414
6.16.20 Dw	415
6.16.21 Delay	416
6.16.22 Inertia	417
6.16.23 K Soft Max Factor	418
6.16.24 K Soft Min Factor	419
6.16.25 Kp/Kv Ratio Factor	420
6.16.26 Ramp Time	421
6.17 Type Linked M Process	422
6.17.1 The Linked M Process type	422
6.17.2 Name	423
6.17.3 Offset Adjust Delay Time	424
6.17.4 Max Follower Offset	425
6.17.5 Max Offset Speed	426
6.17.6 Offset Speed Ratio	427
6.17.7 Ramp Time	428
6.17.8 Master Follower kp	429
6.18 Type Mains	430
6.18.1 The Mains type	430
6.18.2 Name	431
6.18.3 Mains Tolerance Min	432
6.18.4 Mains Tolerance Max	433
6.19 Type Measurement Channel	434
6.19.1 The Measurement Channel type	434
6.19.2 Name	435
6.19.3 Use Measurement Board Type	436
6.19.4 Disconnect at Deactivate	437
6.19.5 Measurement Link	438
6.19.6 Board Position	439
6.20 Type Mechanical Unit	440
6.20.1 The Mechanical Unit type	440
6.20.2 Name	441
6.20.3 Use Activation Relay	442
6.20.4 Use Brake Relay	443
6.20.5 Use Connection Relay	444
6.20.6 Use Robot	445
6.20.7 Use Single 1, 2, 3, 4, 5, 6	446
6.20.8 Allow Move of User Frame	447
6.20.9 Activate at Startup	448
6.20.10 Deactivation Forbidden	449
6.21 Type Motion Planner	450
6.21.1 The Motion Planner type	450
6.21.2 Name	451
6.21.3 Brake on Time	452
6.21.4 Dynamic Resolution	453
6.21.5 Path Resolution	454
6.21.6 Queue Time	456
6.21.7 Teach Mode Max Speed	457

Table of Contents

6.21.8 Process Update Time	458
6.21.9 Prefetch Time	459
6.21.10 Event Preset Time	460
6.21.11 CPU Load Equalization	461
6.21.12 Use Motion Supervision	462
6.21.13 Motion Supervision Permanent Off	463
6.21.14 Motion Supervision Max Level	464
6.21.15 Remove Corner Path Warning	465
6.21.16 Time Event Supervision	466
6.21.17 High Interpolation Priority	467
6.21.18 Speed Control Warning	468
6.21.19 Speed Control Percent	469
6.22 Type Motion Supervision	470
6.22.1 The Motion Supervision type	470
6.22.2 Name	471
6.22.3 Path Collision Detection	472
6.22.4 Jog Collision Detection	473
6.22.5 Path Collision Detection Level	474
6.22.6 Jog Collision Detection Level	475
6.22.7 Collision Detection Memory	476
6.23 Type Motion System	477
6.23.1 The Motion System type	477
6.23.2 Name	478
6.23.3 Min Temperature Cabinet	479
6.23.4 Max Temperature Cabinet	480
6.23.5 Min Temperature Robot	481
6.23.6 Max Temperature Robot	482
6.24 Type Motor	483
6.24.1 The Motor type	483
6.24.2 Name	484
6.24.3 Use Motor Type	485
6.24.4 Use Motor Calibration	486
6.25 Type Motor Calibration	487
6.25.1 The Motor Calibration type	487
6.25.2 Name	488
6.25.3 Commutator Offset	489
6.25.4 Commutator Offset Valid	490
6.25.5 Calibration Offset	491
6.25.6 Calibration Offset Valid	492
6.25.7 Calibration Sensor Position	493
6.26 Type Motor Type	494
6.26.1 The type Motor Type	494
6.26.2 Name	495
6.26.3 Pole Pairs	496
6.26.4 Stall Torque	497
6.26.5 k_e Phase to Phase	498
6.26.6 Max Current	499
6.26.7 Phase Resistance	500
6.26.8 Phase Inductance	501
6.27 Type Path Sensor Synchronization	502
6.27.1 The Path Sensor Synchronization type	502
6.27.2 Name	503
6.27.3 Max Advance Distance	504
6.27.4 Max Delay Distance	505

6.27.5 Max Synchronization Speed	506
6.27.6 Min Synchronization Speed	507
6.27.7 Synchronization Type	508
6.28 Type Process	509
6.28.1 The Process type	509
6.28.2 Name	510
6.28.3 Use SG Process	511
6.28.4 Use Linked Motor Process	512
6.29 Type Relay	513
6.29.1 The Relay type	513
6.29.2 Output Signal	514
6.29.3 Input Signal	515
6.30 Type Robot	516
6.30.1 The Robot type	516
6.30.2 Name	517
6.30.3 Use Old SMB	518
6.30.4 Use Joint 1, 2, 3, 4, 5, 6	519
6.30.5 Base Frame x, y, z	520
6.30.6 Base Frame q1, q2, q3, q4	521
6.30.7 Base Frame Moved by	522
6.30.8 Gravity Alpha	523
6.30.9 Gravity Beta	524
6.30.10 Gamma Rotation	525
6.30.11 Upper Work Area x, y, z	526
6.30.12 Lower Work Area x, y, z	527
6.30.13 Upper Check Point Bound x, y, z	528
6.30.14 Lower Check Point Bound x, y, z	529
6.30.15 Use Six Axes Corvec	530
6.30.16 Track Conveyor with Robot	531
6.30.17 7 axes high performance motion	532
6.30.18 Time to Inposition	533
6.31 The Robot Serial Number type	534
6.31.1 The Robot Serial Number type	534
6.31.2 Name	535
6.31.3 Robot Serial Number High Part	536
6.31.4 Robot Serial Number Low Part	537
6.32 Type SG Process	538
6.32.1 The SG Process type	538
6.32.2 Name	540
6.32.3 Use Force Master	541
6.32.4 Sync Check Off	542
6.32.5 Close time adjust	543
6.32.6 Close position adjust	544
6.32.7 Force Ready Delay	545
6.32.8 Max Force Control Motor Torque	546
6.32.9 Post-synchronization Time	547
6.32.10 Calibration Mode	548
6.32.11 Calibration Force High	549
6.32.12 Calibration Force Low	550
6.32.13 Calibration Time	551
6.32.14 Number of Stored Forces	552
6.32.15 Tip Force 1, 2, 3, 4, 5, 6, 7, 8, 9, 10	553
6.32.16 Motor Torque 1, 2, 3, 4, 5, 6, 7, 8, 9, 10	555
6.32.17 Soft Stop Timeout	557

Table of Contents

6.33 Type Single	558
6.33.1 The Single type	558
6.33.2 Name	559
6.33.3 Use Single Type	560
6.33.4 Use Joint	561
6.33.5 Base Frame x, y, z	562
6.33.6 Base Frame q1, q2, q3, q4	563
6.33.7 Base Frame Coordinated	564
6.34 Type Single Type	565
6.34.1 The Single Type type	565
6.34.2 Name	566
6.34.3 Mechanics	567
6.35 Type SIS Parameters	568
6.35.1 The SIS Parameters type	568
6.35.2 Name	569
6.35.3 Operational Limit	570
6.35.4 Calendar Limit	571
6.35.5 Operational Warning	572
6.35.6 Calendar Warning	573
6.35.7 Gearbox Warning	574
6.36 Type Stress Duty Cycle	575
6.36.1 The Stress Duty Cycle type	575
6.36.2 Name	576
6.36.3 Speed Absolute Max	577
6.36.4 Torque Absolute Max	578
6.37 Type Supervision	579
6.37.1 The Supervision type	579
6.37.2 Name	580
6.37.3 Brake Release Supervision On	581
6.37.4 Speed Supervision	582
6.37.5 Position Supervision	583
6.37.6 Counter Supervision	584
6.37.7 Jam Supervision	585
6.37.8 Load Supervision	586
6.37.9 Power Up Position Supervision	587
6.37.10 In Position Range	588
6.37.11 Zero Speed	589
6.37.12 Affects Forced Control	590
6.37.13 Forced on Position Limit	591
6.37.14 Forced off Position Limit	592
6.38 Type Supervision Type	593
6.38.1 The type Supervision Type	593
6.38.2 Name	594
6.38.3 Max Force Control Position Error	595
6.38.4 Max Force Control Speed Limit	596
6.38.5 Dynamic Power Up Position Limit	597
6.38.6 Teach Max Speed Main	598
6.38.7 Max Jam Time	599
6.38.8 Max Overload Time	600
6.38.9 Teach Normalized Low Speed	601
6.38.10 Auto Max Speed Supervision Limit	602
6.38.11 Influence Group	603
6.38.12 Alarm Position Limit for Brake Release	604
6.38.13 Position OK Ratio for Brake Release	605

6.39 Type Transmission	606
6.39.1 The Transmission type	606
6.39.2 Name.	607
6.39.3 Rotating Move	608
6.39.4 Transmission Gear Ratio	609
6.39.5 Transmission Gear High.	610
6.39.6 Transmission Gear Low	611
6.40 Type Uncalibrated Control Master 0	612
6.40.1 The Uncalibrated Control Master 0 type	612
6.40.2 Name.	613
6.40.3 K _p , Gain Position Loop	614
6.40.4 K _v , Gain Speed Loop	615
6.40.5 T _i Integration Time Speed Loop	616
6.40.6 Speed Max Uncalibrated	617
6.40.7 Acceleration Max Uncalibrated	618
6.40.8 Deceleration Max Uncalibrated	619

Table of Contents

© Copyright 2004-2007 ABB. All rights reserved.

Manual overview

About this manual

This manual describes the IRC5 system parameters by topic and type in an overview. It also covers some basic workflow descriptions on how to add, edit and delete parameters. This can be done via specific software tools, which are not described here, nor how to use them.

The manual covers the most common types and parameters in the topics *Communication*, *Controller*, *I/O*, *Man-machine Communication*, and *Motion*.

Usage

This manual should be used as a reference during configuration of the robot system.

The manual includes parameters for both the basic robot system and selected software and hardware options. The option parameters require that you have the specified option installed in your robot system.

It is recommended that you create a backup or save the configuration files before changing any parameters.

Note! This should only be performed by a trained technician!

Who should read this manual?

This manual is intended for:

- production technicians
- programmers
- service technicians

Prerequisites

The reader should be familiar with:

- industrial robots and terminology
- the RAPID programming language
- how to configure system parameters using RobotStudio Online or FlexPendant.

Organization of chapters

The manual is organized in the following chapters:

Chapter	Content
1.	About system parameters
2.	Topic Communication
3.	Topic Controller
4.	Topic Man-machine Communication
5.	Topic I/O
6.	Topic Motion

Manual overview

Continued

References

The manual contains references to the following information products:

Reference	Document id
Operating manual - Getting started, IRC5 and RobotStudio Online	3HAC027097-001
Operating manual - IRC5 with FlexPendant	3HAC16590-1
Operating manual - RobotStudio Online	3HAC18236-1
Operating manual - Trouble shooting	3HAC020738-001
Operating manual - Calibration Pendulum	3HAC16578-1
Technical reference manual - RAPID Instructions, Functions and Data types	3HAC16581-1
Technical reference manual - RAPID overview	3HAC16580-1
Technical reference manual - RAPID kernel	3HAC16585-1
Application manual - Additional axes and stand alone controller	3HAC021395-001
Application manual - DeviceNet	3HAC020676-001
Application manual - Engineering tools	3HAC020434-001
Application manual - InterBus	3HAC023009-001
Application manual - Motion coordination and supervision	3HAC18154-1
Application manual - Motion functions and events	3HAC18152-1
Application manual - Motion performance	3HAC18153-1
Application manual - MultiMove	3HAC021272-001
Application manual - ProfiBus	3HAC023008-001
Application manual - Robot communication and I/O control	3HAC020435-001
Application manual - Servo motor control	3HAC020436-001

Revisions

The following revisions of this manual have been released.

Revision	Description
-	First edition, released with IRC5 RW 5.04.
A	Released with IRC5 RW 5.05. The printed manual split in two parts. Chapter <i>Man-machine Communication</i> added. Chapter <i>Motion</i> extended with more types and parameter descriptions.
B	Released with IRC5 RW 5.06. Chapter <i>Motion</i> extended to include all visible parameters.
C	Released with IRC5 RW 5.07.
D	Released with IRC5 RW 5.08. New parameters added in topics <i>Motion</i> and <i>I/O</i> . Type <i>Drive System</i> added in topic <i>Motion</i> .
E	Released with IRC5 RW 5.09. New parameters in <i>Motion</i> and <i>I/O</i> . New types <i>Auto Condition Reset</i> and <i>Run Mode Settings</i> added in topic <i>Controller</i> . All DeviceNet parameters moved to <i>Application manual - DeviceNet</i> .

Product documentation, M2004

General

The robot documentation may be divided into a number of categories. This listing is based on the type of information contained within the documents, regardless of whether the products are standard or optional. This means that any given delivery of robot products *will not contain all* documents listed, only the ones pertaining to the equipment delivered.

However, all documents listed may be ordered from ABB. The documents listed are valid for M2004 robot systems.

Product manuals

All hardware, robots and controllers, will be delivered with a **Product manual** which is divided into two parts:

Product manual, procedures

- Safety information
- Installation and commissioning (descriptions of mechanical installation, electrical connections)
- Maintenance (descriptions of all required preventive maintenance procedures including intervals)
- Repair (descriptions of all recommended repair procedures including spare parts)
- Additional procedures, if any (calibration, decommissioning)

Product manual, reference information

- Reference information (article numbers for documentation referred to in Product manual, procedures, lists of tools, safety standards)
- Part list
- Foldouts or exploded views
- Circuit diagrams

The product manual published as a PDF consists of only one file where the two parts are presented together, as one Product manual.

Technical reference manuals

The following manuals describe the robot software in general and contain relevant reference information:

- **RAPID Overview:** An overview of the RAPID programming language.
- **RAPID Instructions, Functions and Data types:** Description and syntax for all RAPID instructions, functions and data types.
- **System parameters:** Description of system parameters and configuration workflows.

Continued

Application manuals

Specific applications (e.g. software or hardware options) are described in **Application manuals**. An application manual can describe one or several applications.

An application manual generally contains information about:

- The purpose of the application (what it does and when it is useful)
- What is included (e.g. cables, I/O boards, RAPID instructions, system parameters, CD with PC software)
- How to use the application
- Examples of how to use the application

Operating manuals

This group of manuals is aimed at those having first hand operational contact with the robot, i.e. production cell operators, programmers and trouble shooters. The group of manuals includes:

- **Getting started - IRC5 and RobotStudio Online**
- **IRC5 with FlexPendant**
- **RobotStudio Online**
- **Trouble shooting - IRC5** for the controller and robot

1 About system parameters

1.1. About system parameters

Overview

System parameters describe the configuration of the robot system. The parameters are configured according to order on delivery.

By changing the parameters' values, the performance of the system can be adjusted. The system parameters usually only need changing if the robot system is modified due to a changed process.

Parameter structure

The parameters are grouped together in a number of different configuration areas, named topics. These topics are divided into sections, named types. Each type represents a specific configuration task defined by the settings of the system parameters in the type.

Topic definition

A topic is a configuration area with a specific collection of parameters.

There are six topics in the controller, each describing different areas in the robot system.

- Communication: serial channels and file transfer protocols
- Controller: safety and RAPID specific functions
- I/O: I/O boards and signals
- Man-machine Communication: functions to simplify working with the system
- Motion: the robot and external axes
- Process: process-specific tools and equipment

Type definition

A type is a section of a topic, and defines a configuration task. The type can sometimes be reused many times, but the combination of parameter values is unique.

A combination of parameters for a type is called an instance. The instance is named after the type, e.g. an instance (set of parameters) of the type *Signal* is called a Signal.

System parameters definition

The parameters are assigned a value to describe a task in the robot system configuration.

The parameter values are normally predefined on delivery. The values are restricted to data type, and sometimes intervals, this is described for each parameter in *Technical reference manual - System parameters*.

Most parameters require a restart of the controller to take effect after being changed.

Some parameters are visible but not editable since they are a part of the system and should not be changed.

1 About system parameters

1.1. About system parameters

Continued

Working with system parameters

System parameters are configured using RobotStudio Online or the FlexPendant. This is detailed in *Operating manual - RobotStudio Online* and *Operating manual - IRC5 with FlexPendant*.

2 Topic Communication

2.1. The Communication topic

Overview

This chapter describes the types and parameters of the *Communication* topic. Each parameter is described in the section for its type.

Description

The Communication topic contains parameters for configuring the controller's serial channels and file transfer protocols. The parameters are organized in three types:

1. Application protocol
2. Physical channel
3. Transmission protocol

2 Topic Communication

2.2.1. How to define an application protocol

2.2 Workflows

2.2.1. How to define an application protocol

Overview

Application protocols are options for data transmission in the robot system.

Prerequisites

A physical channel and a transmission protocol must be defined before an application protocol can be defined.

The option *NFS Client* or *FTP Client* must be installed in the robot system.

How to define an application protocol

To define an application protocol:

1. In the topic **Communication**, choose the type **Application Protocol**.
2. Select the application protocol to define, or add a new one.
3. Enter, delete or change the parameters of the application protocol. Note that the required parameters vary depending on the option installed. See description of the type *Application Protocol* for examples.

For detailed information about each parameter and examples of typical configurations, see the descriptions in the type *Application Protocol*.

4. Save the changes

Related information

The Application Protocol type on page 27.

How to define a physical channel on page 25.

How to define a transmission protocol on page 26.

Application manual - Robot communication and I/O control.

2.2.2. How to define a physical channel

Overview

With the Physical Channels you configure the serial channels, which can be used for communication with printers, terminals, computers, and other equipment.

How to define a physical channel

To define a physical channel:

1. In the topic **Communication**, choose the type **Physical Channel**.
2. Select the physical channel to define, or add a new one.
3. Enter, delete or change the parameters of the physical channel.

For detailed information about each parameter, see the descriptions in the type *Physical Channel*.

4. Save the changes.

Related information

[The Physical Channel type on page 42.](#)

2 Topic Communication

2.2.3. How to define a transmission protocol

2.2.3. How to define a transmission protocol

Overview

Transmission protocols are standards for data transmission.

Prerequisites

A physical channel must be defined before a transmission protocol can be defined.

How to define a transmission protocol

To define a transmission protocol:

1. In the topic **Communication**, choose the type **Transmission Protocol**.
2. Select the transmission protocol to define, or add a new one.
3. Edit the parameters of the transmission protocol.

For detailed information about each parameter, see descriptions in the type *Transmission Protocol*.

4. Save the changes.

Related information

[The Transmission Protocol type on page 51.](#)

[How to define a physical channel on page 25.](#)

2.3 Type Application Protocol

2.3.1. The Application Protocol type

Overview

This section describes the type *Application Protocol*, which belongs to the topic *Communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

COM_APP

Type description

Application protocols are standards for data transmission on application level. The robot system can use different protocols, but only one at a time.

The application protocols are options for the robot system.

Prerequisites

A physical channel and a transmission protocol must be defined before an application protocol can be defined.

Related information

[How to define an application protocol on page 24.](#)

[How to define a physical channel on page 25.](#)

[How to define a transmission protocol on page 26.](#)

[Application manual - Robot communication and I/O control.](#)

Example: FTP

This is a typical example of a configuration for *FTP Client*.

Parameter:	Value:
Name	MyFTP
Type	FTP
Transmission Protocol	MyTransmissionProtocol
Server Address	100.100.1.10
Trusted	Yes
Local Path	pc:test.prg
Server Path	c:\backup
Username	Operator1
Password	robot
Maximum File Size	1000
Memory Partition Size	500

2 Topic Communication

2.3.1. The Application Protocol type

Continued

Example: NFS

This is a typical example of a configuration for *NFS Client*.

Parameter:	Value:
Name	MyNFS
Type	NFS
Transmission Protocol	TCP/IP
Server Address	255.255.100.105
Trusted	Yes
Local Path	pc:test.prg
Server Path	c:backup
User ID	10
Group ID	0

2.3.2. Name

Parent

Name belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

Name

Description

The name of the application protocol.

Usage

Used as a protocol label (to tell the application protocols apart).

Allowed values

A string with maximum 40 characters.

2 Topic Communication

2.3.3. Type

2.3.3. Type

Parent

Type belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

Type

Description

The type of application protocol.

Usage

Specify the type of application protocol, FTP or NFS.

Allowed values

FTP or NFS

Related information

Application manual - Robot communication and I/O control.

2.3.4. Transmission Protocol

Parent

Transmission Protocol belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

Trp

Description

Specifies which transmission protocol is used by the application protocol.

Usage

Transmission Protocol is set to the same value as the parameter *Name*, in the type *Transmission Protocol*, for the transmission protocol you want to use (e.g. TCP/IP).

Allowed values

A string with maximum 40 characters.

Related information

Name on page 52.

Application manual - Robot communication and I/O control.

2 Topic Communication

2.3.5. Server Address

2.3.5. Server Address

Parent

Server Address belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

ServerAddress

Description

The IP address of the computer that runs the server application that the application protocol communicates with.

Usage

If the application protocol is used for communication with a remote computer, the IP address of that computer is specified in *Server Address*.

Allowed values

Four integers between 0 and 255, separated with dots.

Related information

Application manual - Robot communication and I/O control.

Example

An IP address typically looks like this:

100.100.100.100

2.3.6. Trusted

Parent

Trusted belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

Trusted

Description

A flag that specifies if losing the connection should make the program stop.

Usage

An application protocol used for backups or similar can have *Trusted* set to No. If the connection is lost, the program continues and the backup can be made later.

An application protocol that relies on the connection for safety must have *Trusted* set to Yes. If the connection is lost, the program will stop and no hazardous situations can occur because of the lost connection.

Allowed values

Yes or No.

Related information

Application manual - Robot communication and I/O control.

2 Topic Communication

2.3.7. Local Path

2.3.7. Local Path

Parent

Local Path belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

LocalPath

Description

The controller's reference to the connection.

Usage

When the connection is used from a RAPID program or the FlexPendant, it is referenced with the name defined in *Local Path*.

Defines what the shared unit will be called on the robot. The parameter value must end with a colon (:).

Allowed values

A string with a maximum of 20 characters. The string must end with a colon (:).

Related information

Application manual - Robot communication and I/O control.

Example

The application protocol is used for a connection with unit C: on a remote PC. *Local Path* is set to pc:. The file C:\test.prg can then be accessed from a RAPID program or the FlexPendant as pc:test.prg.

2.3.8. Server Path

Parent

Server Path belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

ServerPath

Description

The name of the disk or folder to connect to, on a remote computer.

Usage

Specify the path of the disk or folder that the application protocol should connect to.

NOTE!



If communicating with an FTP server of type Distinct FTP or MS IIS, the exported path should not be specified.

Allowed values

A string with a maximum of 40 characters.

Related information

Application manual - Robot communication and I/O control.

Example

The usage of *Server Path* may depend on which FTP server is being used.

For most FTP servers

If the application protocol should connect to the folder C:\Robot1\Backup on a remote computer, *Server Path* is set to C:\Robot1\Backup.

For FTP servers Distinct FTP and MS IIS

If the server exports C:\Robot1 and the application protocol want to connect to C:\Robot1\Backup, *Server Path* is set to Backup.

2 Topic Communication

2.3.9. Username

2.3.9. Username

Parent

Username belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

UserName

Description

The user name used by the robot when it logs on to an FTP server on a remote computer.

Usage

Create a user account on the FTP server. The user name of this account is then specified in *Username*, and the password in *Password*.

Limitations

Username is only used with the RobotWare option *FTP Client*.

Allowed values

A string with a maximum of 40 characters.

Related information

Password on page 37.

Application manual - Robot communication and I/O control.

2.3.10. Password

Parent

Password belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

Password

Description

The password used by the robot when it logs on to an FTP server on a remote computer.

Usage

Create a user account on the FTP server. The user name of this account is then specified in *Username*, and the password in *Password*.

Limitations

Password is only used with the RobotWare option *FTP Client*.

Allowed values

A string with a maximum of 40 characters.

Additional information

Note that the password written here will be visible to all who have access to the system parameters.

Related information

Username on page 36.

Application manual - Robot communication and I/O control.

2 Topic Communication

2.3.11. User ID

2.3.11. User ID

Parent

User ID belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

UserID

Description

Used by the NFS protocol as a way of authorizing the user to access a specific server.

Usage

If the NFS server requires a User ID and Group ID for access to the server, these numbers are specified in the parameters *User ID* and *Group ID*.

If this parameter is not used, set it to the default value 0.

Note that *User ID* must be the same for all mountings on one controller.

Limitations

User ID is only used with the RobotWare option *NFS Client*.

Allowed values

An integer between 0 and 2,147,483,647.

Default value is 0.

Related information

Group ID on page 39.

Application manual - Robot communication and I/O control.

2.3.12. Group ID

Parent

Group ID belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

GroupID

Description

Used by the NFS protocol as a way of authorizing the user to access a specific server.

Usage

If the NFS server requires a User ID and Group ID for access to the server, these numbers are specified in the parameters *User ID* and *Group ID*.

If this parameter is not used, set it to the default value 0.

Note that *Group ID* must be the same for all mountings on one controller.

Limitations

Group ID is only used with the RobotWare option *NFS Client*.

Allowed values

An integer between 0 and 2,147,483,647.

Default value is 0.

Related information

User ID on page 38.

Application manual - Robot communication and I/O control.

2 Topic Communication

2.3.13. Maximum File Size

Parent

Maximum File Size belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

MaxFileSize

Description

The parameter *Maximum File Size* defines the maximum file size for files to be transferred between the controller and remote clients.

Usage

Transferring large files between the controller and remote clients, e.g. a pc, can make the system slow. By setting the maximum allowed file size to be transferred, the system will not be slowed down when files are being transferred.

If the file size is larger than the value of *Maximum File Size*, an error report is generated in the event handler.

Limitations

Maximum File Size is only used with the RobotWare option *FTP Client*.

Allowed values

File size in kB (kilo bytes), between 1 and 2000.

Default value is 500 kB.

Related information

Application manual - Robot communication and I/O control.

2.3.14. Memory Partition Size

Parent

Memory Partition Size belongs to the type *Application Protocol*, in the topic *Communication*.

Cfg name

CommPartSize

Description

The parameter *Memory Partition Size* defines the size of the allocated memory partition for the FTP communication.

Usage

By using a separate memory partition for the FTP communication, the risk of disturbing other program execution is avoided.

If no separate memory partition is desired, set the value to 0.

Prerequisites

Memory Partition Size is only used with the RobotWare option *FTP Client*.

Allowed values

Partition size in kB (kilo bytes), between 0 and 2000.

Default value is 300 kB.

Note that values above default value cannot be guaranteed to function. The available memory partition size depends on what other options are installed.

Related information

Application manual - Robot communication and I/O control.

2 Topic Communication

2.4.1. The Physical Channel type

2.4 Type Physical Channel

2.4.1. The Physical Channel type

Overview

This section describes the type *Physical Channel*, which belongs to the topic *Communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

COM_PHY_CHANNEL

Type description

Physical channels are used for configuring the serial channels on the controller.

As standard, the controller has one serial channel, which can be used for communication with printers, terminals, computers, and other equipment.

If the controller has option *Multiple Serial Ports* installed, there can be three serial channels available.

Serial channel:	Description:
COM1	A standard RS232 port
COM2	Only available if <i>Multiple Serial Ports</i> is installed
COM3	Only available if <i>Multiple Serial Ports</i> is installed

Related information

[How to define a physical channel on page 25.](#)

2.4.2. Name**Parent**

Name belongs to the type *Physical Channel*, in the topic *Communication*.

Cfg name

Name

Description

Name specifies the logical connection. It is used when accessing the physical serial channel.

Allowed values

A string with maximum 16 characters.

2 Topic Communication

2.4.3. Connector

2.4.3. Connector

Parent

Connector belongs to the type *Physical Channel*, in the topic *Communication*.

Cfg name

Connector

Description

Connector connects a physical communication port with a specific configuration in the system.

Allowed values

COM1 in a standard system.

Also, COM2 and COM3, in a system with *Multiple Serial Ports* installed.

2.4.4. Baudrate

Parent

Baudrate belongs to the type *Physical Channel*, in the topic *Communication*.

Cfg name

Baudrate

Description

Baudrate defines the baud the controller will operate on for the selected physical channel.

Usage

Baud is the signalling rate of the communication, which determines the maximum speed of the data transfer in serial channels. The higher the baud, the faster the communication can be.

Limitations

Both devices, the serial ports in both ends, that communicate on the channel have to use the same baud. The devices have to be defined with the same transmission speed. Therefore, *Baudrate* must be set to the baud of the device that is connected to the controller.

Allowed values

A value between 300-38,400, specifying the signal rate.

The default value is 9,600.

2 Topic Communication

2.4.5. Parity

2.4.5. Parity

Parent

Parity belongs to the type *Physical Channel*, in the topic *Communication*.

Cfg name

Parity

Description

Parity configures the parity check for the data transfer.

Usage

Parity check is an error detection method to help detect data corruption that might occur during transmission of data. The parity check adds a parity bit to each byte that is transmitted. Depending on whether the transmitted byte contains an odd or even number of 1-bits, the parity bit will be either 0 or 1. Each time a data byte is received, it is checked that the number of 1-bits matches the parity bit.

Limitations

Both receiver and transmitter of data must agree on the type of parity.

Allowed values

Value::	Description:
Odd	The number of 1-bits in a transfer byte must be odd. If they are odd, the parity bit is set to 0.
Even	The number of 1-bits in a transfer byte must be even. If they are even, the parity bit is set to 1.
None	No parity check is performed.

2.4.6. Number of Bits

Parent

Number of Bits belongs to the type *Physical Channel*, in the topic *Communication*.

Cfg name

NoOfBits

Description

Number of Bits defines the number of data bits in each byte.

Usage

The number of bits depends on the device the controller should communicate with. Both receiver and transmitter must agree on the number of data bits as well as the baudrate. There may either be 7 or 8 data bits depending on the selection made.

Limitations

Both receiver and transmitter of data must agree on the number of bits.

Allowed values

7 or 8, specifying the number of data bits.

Related information

Baudrate on page 45.

2 Topic Communication

2.4.7. Number of Stop Bits

2.4.7. Number of Stop Bits

Parent

Number of Stop Bits belongs to the type *Physical Channel*, in the topic *Communication*.

Cfg name

NoOfStopBits

Description

Number of Stop Bits defines the number of stop bits.

Usage

A stop bit is used to identify the end of a data byte when it is transmitted. A stop bit can be detected correctly even if the previous data bit also had a value of 1. This is accomplished by the stop bit's duration.

Limitations

Both receiver and transmitter of data must agree on the number of bits.

Stop bits are excluded from the parity calculation. For more information about parity, see *Parity*.

Allowed values

1 or 2, specifying the number of stop bits.

Related information

[Parity on page 46.](#)

2.4.8. Duplex

Parent

Duplex belongs to the type *Physical Channel*, in the topic *Communication*.

Cfg name

Duplex

Description

Duplex defines whether or not the controller shall be able to send and receive data simultaneously on this physical channel.

Usage

Duplex is the ability to transport data in both directions.

With full duplex the controller is able to both send and receive data at the same time.

With half duplex the data flow is limited to one direction at a time.

Allowed values

FULL or HALF.

2 Topic Communication

2.4.9. Flow Control

2.4.9. Flow Control

Parent

Flow Control belongs to the type *Physical Channel*, in the topic *Communication*.

Cfg name

FlowControl

Description

Flow Control defines which type of data flow control is used between the devices that are communicating on the physical channel.

Usage

Flow control adjusts the data transfer so that no data is sent before the receiving device can receive it. Flow control is extra important when the sending device can send data at a higher speed than the receiving device is able to receive.

Limitation

Both receiver and transmitter of data must agree on the type of flow control used.

Allowed values

Value:	Description:
RTS/CTS	Hardware flow control, uses signals on the serial cable to control if sending or receiving is enabled.
XON/XOFF	Software flow control, uses characters in the communication stream to control sending and receiving of data.
NONE	Flow control will not be used.

2.5 Type Transmission Protocol

2.5.1. The Transmission Protocol type

Overview

This section describes the type *Transmission Protocol* which belongs to the topic *Communication*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

COM_TRP

Type description

Transmission protocols are standards for data transmission.

Related information

[How to define a transmission protocol on page 26.](#)

2 Topic Communication

2.5.2. Name

2.5.2. Name

Parent

Name belongs to the type *Transmission Protocol*, in the topic *Communication*.

Cfg name

Name

Description

Name specifies the name of the transmission protocol. Transmission protocols are used for transferring data.

Allowed values

A string with maximum 16 characters.

2.5.3. Type

Parent

Type belongs to the type *Transmission Protocol*, in the topic *Communication*.

Cfg name

Type

Description

Type defines the type of transmission protocol to be used.

Limitations

TCP/IP settings can only be viewed from the Configuration Editor in RobotStudio Online. For detailed information, see *Operating manual - RobotStudio Online*, chapter *Configure the system*.

Allowed values

Value:	Description:
PPP	Point-To-Point protocol, provides communication for simple links which transport packets between two nodes. Thereby, it is only possible to run PPP on the serial channels, i.e. COM1.
RAW	Protocol without handshakes.

Related information

[The Physical Channel type on page 42.](#)

[Operating manual - RobotStudio Online.](#)

For configuration of the LAN port, see [Operating manual - IRC5 with FlexPendant](#).

2 Topic Communication

2.5.4. Physical Channel

2.5.4. Physical Channel

Parent

Physical Channel belongs to the type *Transmission Protocol*, in the topic *Communication*.

Cfg name

PhyChannel

Description

Physical Channel connects a transmission protocol with a physical channel.

Limitations

It is not possible to connect to the LAN port. For configuration of the LAN port, see
Operating manual - IRC5 with FlexPendant.

Allowed values

COM1, in a standard system.

Also, COM2 and COM3, in a system with *Multiple Serial Ports* installed.

Related information

[*The Physical Channel type on page 42*](#).

Operating manual - IRC5 with FlexPendant.

2.5.5. Local Address

Parent

Local Address belongs to the type *Transmission Protocol*, in the topic *Communication*.

Cfg name

LocalAddress

Description

Local Address specifies an IP address of the controller's serial port, i.e. COM1.

Limitations

The parameter *Local Address* can only be used when the parameter *Type* has the value PPP.

Allowed values

A string consisting of 4 integer values between 0 and 255, each specifying one of the four parts, separated by dots.

Related information

[Type on page 53.](#)

Example

An IP address consists of four parts, each with eight bits, separated by dots:

100.100.100.100 or 138.227.1.45.

2 Topic Communication

2.5.6. Remote Address

2.5.6. Remote Address

Parent

Remote Address belongs to the type *Transmission Protocol*, in the topic *Communication*.

Cfg name

RemoteAddress

Description

Remote Address specifies an IP address of the remote peer's serial port. This can be the IP address of the serial port on a PC used for communication with the controller.

Limitations

The parameter *Remote Address* can only be used when the parameter *Type* has the value PPP.

Allowed values

A string consisting of 4 integer values between 0 and 255, each specifying one of the four parts, separated by dots.

Related information

[Type on page 53.](#)

Example

An IP address consists of four parts, each with eight bits, separated by dots:

100.100.100.100 or 138.227.1.45.

3 Topic Controller

3.1. The Controller topic

Overview

This chapter describes the types and parameters of the *Controller* topic. Each parameter is described in the section for its type.

Description

The *Controller* topic contains parameters for safety and RAPID specific functions.

The parameters are organized in the following types:

1. Auto Condition Reset
2. Automatic Loading of Modules
3. Event Routine
4. Mechanical Unit Group
5. ModPos Settings
6. Operator Safety
7. Path Return Region
8. Run Mode Settings
9. Safety Run Chain
10. System Misc
11. Task

3 Topic Controller

3.2.1. How to activate hold-to-run control

3.2 Workflows

3.2.1. How to activate hold-to-run control

Overview

Safety in program execution is essential. The function hold-to-run control is used when extra safety is necessary in the operating mode Manual. The hold-to-run function only allows robot movements when a button is manually actuated and immediately stops these movements when released.

Additional information

The hold-to-run control is always activated in Manual Full Speed mode.

How to activate the hold-to-run control

To activate the hold-to-run control for manual reduced speed mode:

1. In the **Controller** topic, choose the type **Operator Safety**.
2. Edit the parameters for robot movement control and execution. Set the parameter **Active** to True.
For detailed information about the parameters, see the descriptions in the *Operator Safety* type.
3. Save the changes.

Related information

[The Operator Safety type on page 93.](#)

[Operating manual - IRC5 with FlexPendant.](#)

3.2.2. How to define path return region

Return movement

A return movement must take place if the current robot path deviates from the programmed path. This happens for example if an uncontrolled stop has occurred or the robot has been jogged away from its path. A return movement begins when program start is ordered and stops before the program continues with the instruction that was interrupted.

Path return region

In a return movement, the path return region specifies the distance from the current robot position to the last executed path. The maximum path return region can be set both for start in manual mode and for start in automatic mode.

How to define path return region

To define the path return region:

1. In the **Controller** topic, choose the type **Path Return Region**.
2. Edit the **Mode** parameter to specify the operating mode.
3. Edit the parameters for movement in the selected mode. For detailed information about each parameter, see the descriptions in the type *Path Return Region*.
4. Save the changes.

Related information

The Path Return Region type on page 96.

3 Topic Controller

3.3.1. The Auto Condition Reset type

3.3 Type Auto Condition Reset

3.3.1. The Auto Condition Reset type

Overview

This section describes the type *Auto Condition Reset*, which belongs to the topic *Controller*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

AUTO_COND_RESET

Type description

The type *Auto Condition Reset* defines if a number of conditions should be reset when switching to auto mode.

A message box is displayed on the FlexPendant with information about the reset conditions.

Limitations

There can be only one instance of the type *Auto Condition Reset*.

3.3.2. Name

Parent

Name belongs to the type *Auto Condition Reset*, in the topic *Controller*.

Cfg name

name

Allowed values

AllDebugSettings (cannot be changed).

3 Topic Controller

3.3.3. Reset

3.3.3. Reset

Parent

Reset belongs to the type *Auto Condition Reset*, in the topic *Controller*.

Cfg name

reset

Description

Reset defines if a number of conditions should be reset when switching to auto mode.

Usage

If *Reset* is set to YES then the following conditions are reset when switching to auto:

- The Program Pointer (PP) is set to Main module.
- All tasks are enabled.
- Simulation of all simulated I/O signals is removed.
- Speed is set to 100%.

If *Reset* is set to NO, then none of the above conditions are reset automatically.

Allowed values

YES or NO. Default value is YES.

3.4 Type Automatic Loading of Modules

3.4.1. The Automatic Loading of Modules type

Overview

This section describes the type *Automatic Loading of Modules* which belongs to the topic *Controller*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

CAB_TASK_MODULES

Type description

RAPID modules can be loaded automatically when the controller is restarted if they are specified in the type *Automatic Loading of Modules*.

Usage

There must be one instance of the type *Automatic Loading of Modules* for each module to be loaded.

System restart

All changes in the type *Automatic Loading of Modules* will take effect after a normal restart or a restart of the program server (P-Start).

Additional information

If the configuration module is changed, it may in one case (see below) replace the loaded module after a normal restart. In any other case, you will get a warning. To replace the loaded module regardless of task type, do a P-start.

The configuration module replaces the loaded module if the:

- loaded module is a program module AND
- the task is semi-static.

The program pointer is only lost if a configuration change results in unloading of the module that the program pointer is in. If a shared or installed module is changed from True to False, or is moved to another task, the task will be reinstalled and the program pointer is reset. All previously loaded modules are reloaded and unsaved changes will not be lost.

If a changed and unsaved user-loaded module is unloaded due to configuration changes, it will be saved to a recovery directory and pointed out in an ELOG message.

If a changed and unsaved configuration loaded module is unloaded due to configuration changes, it will be saved from where it was loaded.

All tasks are reinstalled with modules according to the configuration after a P-Start. **Note** that after a P-start, all user-loaded modules are lost.

3 Topic Controller

3.4.1. The Automatic Loading of Modules type

Continued

Related information

[The Task type on page 113.](#)

Technical reference manual - RAPID overview.

ELOG messages are described in *Operating manual - Trouble shooting*.

Restarts are described in *Operating manual - IRC5 with FlexPendant*.

3.4.2. File

Parent

File belongs to the type *Automatic Loading of Modules*, in the topic *Controller*.

Cfg name

File

Description

The parameter *File* describes a path to the module file.

Usage

The module file shall contain one module to be loaded, installed or shared.

Allowed values

A path, e.g. HOME:base.sys

Related information

Technical reference manual - RAPID overview.

3 Topic Controller

3.4.3. Task

3.4.3. Task

Parent

Task belongs to the type *Automatic Loading of Modules*, in the topic *Controller*.

Cfg name

Task

Description

Task is the symbolic name of the task to which the module will be loaded.

Usage

The task is defined in the type *Task*.

The available task(s) is shown under the type *Task*.

Limitations

Cannot be combined with *All Tasks* or *Shared*.

Allowed values

A task name with maximum 30 characters.

Additional information

All automatically loaded modules need information on which task they will be loaded/installed in, even if only one task is configured in the system.

Related information

[The Task type on page 113](#).

[All Tasks on page 69](#).

[Shared on page 68](#).

[Application manual - Engineering tools](#).

3.4.4. Installed

Parent

Installed belongs to the type *Automatic Loading of Modules*, in the topic *Controller*.

Cfg name

Installed

Description

A module can be both installed or loaded. A loaded module is visible in remote clients, e.g. RobotStudio^{Online} and FlexPendant. An installed module is not visible, it does not occur in the list of modules.

Usage

Set *Installed* to Yes to install a module, and to No to load a module.

Limitations

Cannot be combined with *Shared*. *Shared* has higher priority.

An installed module cannot be removed, just as ordinary RAPID instructions. To remove an installed module, the system must be restarted.

Allowed values

YES or NO.

The default value is No.

Related information

[Shared on page 68](#).

[Technical reference manual - RAPID overview](#).

3 Topic Controller

3.4.5. Shared

3.4.5. Shared

Parent

Shared belongs to the type *Automatic Loading of Modules*, in the topic *Controller*.

Cfg name

Shared

Description

It is possible to install the module so it (and all its objects) is reachable from all tasks.

Usage

If an installed module should be reachable from any task, set the parameter *Shared* to YES. This installs the module to the system internal shared task, not visible from any user interface or in the configuration.

Limitations

Cannot be combined with *All Tasks*, *Task*, or *Installed*.

Allowed values

YES or NO.

Default value is No.

Additional information

Shared has higher priority than *Installed* if both are set to Yes.

If Shared:	and if Installed:	Then:
Yes	Yes or No	The module is installed and available from all tasks.
No	Yes	The module is installed and only available from the named task.
No	No	The module is loaded.

Related information

[All Tasks on page 69.](#)

[Task on page 66.](#)

[Installed on page 67.](#)

Operating manual - RobotStudio Online.

3.4.6. All Tasks

Parent

All Tasks belongs to the type *Automatic Loading of Modules*, in the topic *Controller*.

Cfg name

AllTask

Description

The *All Tasks* module will be loaded into all tasks available.

Usage

The tasks are defined in the type *Task*.

Limitations

Cannot be combined with *Task* or *Shared*.

Allowed values

YES or NO.

Default value is No.

Additional information

All automatically loaded modules need information on which task they will be loaded/installed in, even if only one task is configured in the system.

Related information

[Task on page 66.](#)

[Shared on page 68.](#)

[The Task type on page 113.](#)

3 Topic Controller

3.4.7. Hidden

3.4.7. Hidden

Parent

Hidden belongs to the type *Automatic Loading of Modules*, in the topic *Controller*.

Cfg name

Hidden

Description

RAPID modules, routines and data may be hidden, which may be used to prevent inexperienced end users from tampering (accidentally deleting or changing) with the contents.

Note that the hidden contents is not protected! It can easily be shown again by setting the parameter value to NO.

Note that any hidden contents will still be available when using the `SetDataSearch` instruction to search RAPID data.

Limitations

This parameter affects only modules, routines and data which are loaded automatically on start-up, i.e. no programs etc. that are loaded by the operator once the system has been started.

Changes to the parameter will become effective only after performing a P-start.

Allowed values

YES or NO.

Default value is NO.

3.5 Type Event Routine

3.5.1. The Event Routine type

Overview

This section describes the type *Event Routine* which belongs to the topic *Controller*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

CAB_EXEC_HOOKS

Type description

The type *Event Routine* contains parameters for event handling. Special system events, such as program stop, can be connected to a RAPID routine. When the event occurs, the connected event routine is executed automatically.

An event routine is made up of one or more instructions. The routine runs in the task specified in parameter *Task* or *All Tasks*.

The tasks available are dependent on the type *Tasks*.

Event routines

The following event routines are available:

- PowerOn
- Start
- Restart
- Stop
- QStop
- Reset

If the connected routine is not loaded, an error message is given.

A *Stop* or *Break* instruction in an event routine will stop the program execution and continue to the stop routine. However, if the event routine is a stop event routine (Qstop not included), the stop event routines are not re-executed.

A *Stop* or *Break* instruction in an event routine included in a list of event routines will stop program execution for the running routine. Program execution continues from the beginning of the next event, i.e. instructions after a *Stop* or *Break* in a routine will never be executed.

A stopped event routine must be restarted by invoking the start command, either by pressing the start button on the FlexPendant or calling the start command via a system I/O. It is not possible to restart the interrupted event routine by triggering another system I/O interrupt. The only way to cancel a stopped event routine from system I/O is to start the program from main. E.g. pushing the stop button when the STOP event routine is executing does not generate a new STOP event. The STOP event routine completes its execution.

System restart

Any changes in configuration of event routines are activated after a normal restart.

Continues on next page

3 Topic Controller

3.5.1. The Event Routine type

Continued

Related information

[The Task type on page 113.](#)

Technical reference manual - RAPID overview.

Technical reference manual - RAPID Instructions, Functions and Data types.

3.5.2. Routine

Parent

Routine belongs to the type *Event Routine*, in the topic *Controller*.

Cfg name

Routine

Description

Routine specifies which routine that should be run for an event.

Usage

Define the routine to be assigned to a system event.

It is advisable to use a routine in a system module.

Limitations

The specified routine must be a procedure without any parameters.

The event Reset requires a routine in a system module.

Allowed values

A string defining a routine.

3 Topic Controller

3.5.3. Event

3.5.3. Event

Parent

Event belongs to the type *Event Routine*, in the topic *Controller*.

Cfg name

Shelf

Description

Event specifies which system event in the robot system the routine should run.

Usage

A system event can trigger a corresponding routine to be run, see *Operating manual - IRC5 with FlexPendant*.

It is advisable to keep the routines short and quick.

Limitations

The following limitation should be considered:

- The events are not activated when manually executing a routine without losing the call hierarchy.
- A maximum of 20 routines may be specified for each system event and each task (multitasking). The same routine can be used in more than one event (e.g. the same routine can be run for both Start and Restart).
- The specified event routine cannot be executed if the task program has semantic errors (reference errors etc.). If this is the case, the system generates an error.
- Only the event routine for Start can have motion instructions. A motion instruction in any other event routine will result in an error.

Allowed values

The following values are allowed.

Value:	Description:
Power On	The specified routine is run when the robot is restarted (warm start or cold start) from a remote client or by power on.
Start	Execution is started from the beginning of the program. This is when you press the start or step buttons after having: <ul style="list-style-type: none">• loaded a new program or a new module• ordered Start from beginning• ordered Debug/Move PP to Main• ordered Debug/Move PP to Routine• moved the program pointer in such a way that the execution order is lost.

Continued

Value:	Description:
Stop	<p>The program was stopped:</p> <ul style="list-style-type: none"> • with the stop button • with a STOP instruction • stop after current instruction. <p>Note: A delayed stop after current cycle will not execute the routines connected to this state.</p> <p>The event is not activated at Exit instruction or stop due to execution error.</p>
QStop	The robot was quick stopped (emergency stop).
Restart	Execution is started from the position where it was stopped, or from another instruction the program pointer has been moved to, without having lost the execution order. The event is not activated after having executed one instruction in step by step mode (FWD or MStep).
Reset	Close and load a new program using the FlexPendant. The event is not activated after having loaded a system module or a program module.

Additional information

The following event routines are predefined for all tasks in all systems and must not be removed.

Event:	Routine:	Sequence no.
Reset	SYS_RESET	0
Start	SYS_RESET	0
Power On	SYS_POWERON	0

Related information

Operating manual - IRC5 with FlexPendant.

3 Topic Controller

3.5.4. Sequence Number

3.5.4. Sequence Number

Parent

Sequence Number belongs to the type *Event Routine*, in the topic *Controller*.

Cfg name

SqNo

Description

Sequence Number specifies in which order the routine should be executed for a specific event.

Usage

Order the event routines in a sequence where the first routine shall have a low value and the routines that shall run last has the highest value.

0 will run first.

Note! If several event routines has the same sequence number, the execution order will be unpredictable.

Allowed values

A value between 0 and 100.

Default value is 0.

3.5.5. Task

Parent

Task belongs to the type *Event Routine*, in the topic *Controller*.

Cfg name

Task

Description

Task specifies the name of the task that the routine will run in.

Usage

The task is defined in the type *Task*.

Limitations

Cannot be combined with *All Tasks*.

Allowed values

Names of configured tasks of the type *Task*.

Additional information

All event routines need information on which task they will run, even though only one task is configured in the system.

Related information

[The Task type on page 113](#).

[All Tasks on page 78](#).

3 Topic Controller

3.5.6. All Tasks

3.5.6. All Tasks

Parent

All Tasks belongs to the type *Event Routine*, in the topic *Controller*.

Cfg name

AllTasks

Description

All Tasks defines if the routine will run in all configured tasks.

Usage

The tasks are defined in the type *Task*.

Limitations

Cannot be combined with *Task*.

Allowed values

YES or NO.

The default behavior is No.

Additional information

All event routines need information on which task they will run, even though only one task is configured in the system.

Related information

[The Task type on page 113](#).

[Task on page 77](#).

3.6 Type Mechanical Unit Group

3.6.1. The Mechanical Unit Group type

Overview

This section describes the type *Mechanical Unit Group*, which belongs to the topic *Controller*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

MECHANICAL_UNIT_GROUP

Type description

With the option *MultiMove* comes the possibility to control several robots from one controller. Each task can control one robot and up to six positioners. The mechanical units that will be controlled by one task are grouped in a mechanical unit group.

Related information

Use Mechanical Unit Group on page 120.

Application manual - MultiMove.

3 Topic Controller

3.6.2. Name

3.6.2. Name

Parent

Name belongs to the type *Mechanical Unit Group*, in the topic *Controller*.

Cfg name

Name

Description

The name of the mechanical unit group.

Usage

This is the public identity of the mechanical unit group. It is used by the parameter *Use Mechanical Unit Group* in the type *Tasks*.

Limitations

Mechanical Unit Group is only used if you have the option *MultiMove*.

Allowed values

A string with maximum 32 characters.

Related information

[Use Mechanical Unit Group on page 120](#).

3.6.3. Robot

Parent

Robot belongs to the type *Mechanical Unit Group*, in the topic *Controller*.

Cfg name

Robot

Description

Specifies the robot (with TCP), if there is any, in the mechanical unit group.

Usage

Robot is set to the same value as the parameter *Name* for the *Mechanical Unit Group* type that it represents.

Limitations

The parameter *Robot* is only used if you have the option *MultiMove*.

Allowed values

A string with maximum 32 characters.

Related information

[Name on page 80.](#)

3 Topic Controller

3.6.4. Mechanical Unit 1, 2, 3, 4, 5, 6

3.6.4. Mechanical Unit 1, 2, 3, 4, 5, 6

Parent

Mechanical Unit 1, Mechanical Unit 2, Mechanical Unit 3, Mechanical Unit 4, Mechanical Unit 5, and Mechanical Unit 6 belongs to the type *Mechanical Unit Group*, in the topic *Controller*.

Cfg name

MechanicalUnit_1
MechanicalUnit_2
MechanicalUnit_3
MechanicalUnit_4
MechanicalUnit_5
MechanicalUnit_6

Description

Mechanical Unit 1 specifies the first mechanical unit without TCP, if there is any, in the mechanical unit group.

Mechanical Unit 2 specifies the second mechanical unit without TCP, if there is more than one, in the mechanical unit group.

Mechanical Unit 3 specifies the third mechanical unit without TCP, if there are more than two, in the mechanical unit group.

Mechanical Unit 4 specifies the fourth mechanical unit without TCP, if there are more than three, in the mechanical unit group.

Mechanical Unit 5 specifies the fifth mechanical unit without TCP, if there are more than four, in the mechanical unit group.

Mechanical Unit 6 specifies the sixth mechanical unit without TCP, if there are more than five, in the mechanical unit group.

Usage

Mechanical Unit is set to the same value as the parameter *Name* for the *Mechanical Unit Group* type that it represents.

Limitations

The parameters *Mechanical Unit* is only used if you have the option *MultiMove*.

Allowed values

A string with maximum 32 characters.

Related information

[Name on page 80.](#)

3.6.5. Use Motion Planner

Parent

Use Motion Planner belongs to the type *Mechanical Unit Group*, in the topic *Controller*.

Cfg name

UseMotionPlanner

Description

Specifies which motion planner shall be used for calculating the movements of the mechanical units in this group.

Usage

Use Motion Planner is set to the same value as the parameter *Name* for the *Motion Planner* type that you want to use.

Limitations

The parameter *Use Motion Planner* is only used if you have the option *MultiMove*.

Allowed values

A string with maximum 32 characters.

Related information

The Motion Planner type on page 450 in the topic *Motion*.

3 Topic Controller

3.7.1. The ModPos Settings type

3.7 Type ModPos Settings

3.7.1. The ModPos Settings type

Overview

This section describes the type *ModPos Settings* which belongs to the topic *Controller*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

HOTEDIT_MODPOS

Type description

It is sometimes desirable to limit how much a robtarget position can be moved by a ModPos or HotEdit operation. The limited deviation concerns both the linear distance and the orientation.

Limitations

There can be only one set of parameters of the type *ModPos Settings* in the system.

3.7.2. Name

Parent

Name belongs to the type *ModPos Settings*, in the topic *Controller*.

Cfg name

name

Description

Name defines that the parameter configuration is for ModPos.

Allowed values

modpos

Related information

Operating manual - IRC5 with FlexPendant.

3 Topic Controller

3.7.3. Limited ModPos

3.7.3. Limited ModPos

Parent

Limited ModPos belongs to the type *ModPos Settings*, in the topic *Controller*.

Cfg name

type

Description

Limited ModPos defines if a ModPos change must be within a limited sphere for the position deviation and within a limited cone for the reorientation.

Usage

Set *Limited ModPos* to False when no limit is required, and to True when limits should apply.

Allowed values

FALSE or TRUE.

Default value is TRUE.

3.7.4. Mode

Parent

Mode belongs to the type *ModPos Settings*, in the topic *Controller*.

Cfg name

mode

Description

Mode defines how the limit is defined; to an absolute point or relative to the current position.

Usage

Setting *Mode* to Absolute means that the limited sphere/cone is around a fixed original point, i.e. position changes are accumulated and the accumulated deviation value is checked against the set max limits each time a change is made.

Setting *Mode* to Relative means that the limited sphere/cone is around the current point and will be moved when you modify the position.

Limitations

Mode is only available if *Limited ModPos* is set to TRUE.

Allowed values

Absolute or Relative.

Default value is Relative.

Related information

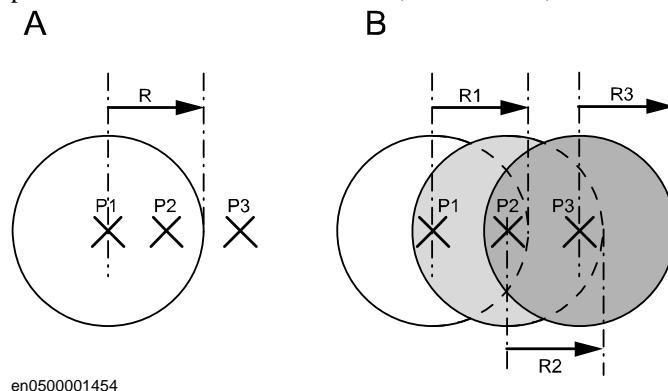
[Limited ModPos on page 86.](#)

Example

In this example, the original point P1 is moved two times, first to P2 and then to P3. In figure A, *Mode* is set to Absolute, and in figure B, *Mode* is set to Relative.

The allowed move distance, R does not change in figure A. This makes it impossible to move the point to P3, as this is beyond R.

In figure B however, the allowed move distance follows the last point. So from P1 it is possible to move as far as R1 allows, and from P2, it is allowed to move as far as R2, etc.



3 Topic Controller

3.7.5. Limit Trans

3.7.5. Limit Trans

Parent

Limit Trans belongs to the type *ModPos Settings*, in the topic *Controller*.

Cfg name

limittrans

Description

Limit Trans defines the maximum allowed deviation in mm from the current or original position.

Limitations

This type is only available if *Limited ModPos* is set to TRUE.

Allowed values

0 - 1000 mm.

Default value is 5.

Related information

[Limited ModPos on page 86.](#)

3.7.6. Limit Rot

Parent

Limit Rot belongs to the type *ModPos Settings*, in the topic *Controller*.

Cfg name

limitrot

Description

Limit Rot defines the maximum allowed reorientation in degrees from the current or original position.

Limitations

This type is only available if *Limited ModPos* is set to TRUE.

Allowed values

0 - 360 degrees (0 - 6.280 radians).

Default value is 10 degrees (0.17 radians).

Additional information

Convert degrees to radians: radians = (degrees/360) * (2*pi)

Related information

[Limited ModPos on page 86.](#)

3 Topic Controller

3.7.7. Limit External Trans

3.7.7. Limit External Trans

Parent

Limit External Trans belongs to the type *ModPos Settings*, in the topic *Controller*.

Cfg name

limitexttrans

Description

Limit External Trans defines the maximum allowed deviation in mm from the current or original position concerning external linear axes.

Limitations

Limit External Trans is only available if *Limited ModPos* is set to TRUE.

Allowed values

0 - 1000 mm.

Default value is 50.

Related information

[Limited ModPos on page 86.](#)

3.7.8. Limit External Rot

Parent

Limit External Rot belongs to the type *ModPos Settings*, in the topic *Controller*.

Cfg name

limitextrot

Description

Limit External Rot defines the maximum allowed deviation in degrees from the current or original position concerning external rotational axes.

Limitations

Limit External Rot is only available if *Limited ModPos* is set to TRUE.

Allowed values

0 - 360 degrees (0 - 6.280 radians).

Default value is 10 degrees (0.17 radians).

Additional information

Convert degrees to radians: radians = (degrees/360) * (2*pi)

Related information

[Limited ModPos on page 86.](#)

3 Topic Controller

3.7.9. Change to LModPos in Auto

3.7.9. Change to LModPos in Auto

Parent

Change to LModPos in Auto belongs to the type *ModPos Settings*, in the topic *Controller*.

Cfg name

ifauto

Description

Change to LModPos in Auto defines if it is possible to force the system to change to *Limited ModPos* automatically when switching from Manual to Auto. When going back to Manual the configured value is valid.

Usage

Setting *Change to LModPos in Auto* to No means that *Limited ModPos* will not be activated when switching from Manual to Auto.

Yes means that *Limited ModPos* will be activated when the operator's key is switched to Auto Mode.

Limitations

Change to LModPos in Auto is only available if *Name* is *ModPos*.

Allowed values

Yes or No.

Default value is No.

Related information

[Limited ModPos on page 86.](#)

[Name on page 85.](#)

3.8 Type Operator Safety

3.8.1. The Operator Safety type

Overview

This section describes the type *Operator Safety* which belongs to the topic *Controller*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

MASTER_BOOL

Type description

The *Operator Safety* type is used to define extra safety for system execution.

Related information

[How to activate hold-to-run control on page 58.](#)

[Operating manual - IRC5 with FlexPendant, chapter Safety.](#)

3 Topic Controller

3.8.2. Function

3.8.2. Function

Parent

Function belongs to the type *Operator Safety*, in the topic *Controller*.

Cfg name

Name

Description

Function defines safety functions for the robot system.

Allowed values

Value:	Description:
Hold-to-run	Hold-to-run enables a functionality that requires a button to be pressed in to allow execution in Manual Reduce Speed mode. When the button is released the executions are immediately stopped. Hold-to-run is always activated in Manual Full Speed operating mode. Hold-to-run is further described in standard ISO 10218 (EN775).

Related information

[How to activate hold-to-run control on page 58.](#)

Operating manual - IRC5 with FlexPendant chapter *Safety*.

3.8.3. Active

Parent

Active belongs to the type *Operator Safety*, in the topic *Controller*.

Cfg name

Select

Description

Active defines whether the value of *Function* is activated.

Allowed values

Value:	Description:
TRUE	Activated
FALSE	Not activated

Related information

[Function on page 94.](#)

3 Topic Controller

3.9.1. The Path Return Region type

3.9 Type Path Return Region

3.9.1. The Path Return Region type

Overview

This section describes the type *Path Return Region* which belongs to the topic *Controller*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

CAB_REGAIN_DIST

Type description

In a return movement, the path return region specifies the distance from the current robot position to the last executed path. The maximum path region can be set both for start in manual mode and for start in automatic mode.

There must be two sets of parameters defined for this type; one for automatic mode (AUTO) and one for manual mode (MAN). Both are predefined on delivery.

Return movements

A return movement must take place when the current path of the robot deviates from the programmed path. For example, this is required when an uncontrolled stop has occurred or when the robot has been jogged away from its path.

A return movement begins when program start is ordered and stops before the program continues with the instruction that was interrupted due to a stop request.

Predefined path return regions

AUTO

MAN

3.9.2. Mode

Parent

Mode belongs to the type *Path Return Region*, in the topic *Controller*.

Cfg name

Name

Description

Mode defines in which operating mode a return movement will start.

Usage

Both Auto and Man mode must be defined in the system and are configured on delivery.

Allowed values

AUTO

MAN

3 Topic Controller

3.9.3. TCP Distance

3.9.3. TCP Distance

Parent

TCP Distance belongs to the type *Path Return Region*, in the topic *Controller*.

Cfg name

TCP_Dist

Description

TCP Distance defines the maximum allowed TCP distance from the current robot position to the last executed path.

Usage

TCP Distance is used to limit the return movement if there is a risk that the robot will collide with an object.

Prerequisites

Specify which operating mode the return movement is valid for. This is defined in the parameter *Mode*.

Allowed values

A value between 0-2.000 meters, specifying the movement in meters.

The default value is 0.05 meter for both manual and automatic mode.

Related information

[Mode on page 97](#).

Application manual - Motion coordination and supervision.

3.9.4. TCP Rotation

Parent

TCP Rotation belongs to the type *Path Return Region*, in the topic *Controller*.

Cfg name

TCP_Rot

Description

TCP Rotation defines the maximum allowed TCP rotation from the current robot position to the last executed path.

Usage

TCP Rotation is used to limit the return movement if there is a risk that the robot will collide with an object.

Prerequisites

Specify which operating mode the return movement is valid for. This is defined in the parameter *Mode*.

Allowed values

A value between 0-6.280, specifying the movement in radians.

The default value is 0.2 radians for manual mode and 1.57 radians for automatic mode.

Additional information

To convert degrees to radians, use this formula:

$$\text{radians} = 2\pi \times \text{degrees}/360$$

Related information

[Mode on page 97](#).

Application manual - Motion coordination and supervision.

3 Topic Controller

3.9.5. External Distance

3.9.5. External Distance

Parent

External Distance belongs to the type *Path Return Region*, in the topic *Controller*.

Cfg name

Ext_Dist

Description

External Distance defines the maximum allowed external axes distance from the current robot position to the last executed path.

Usage

External Distance is used to limit the return movement if there is a risk that the robot will collide with an object.

Prerequisites

Specify which operating mode the return movement is valid for. This is defined in the parameter *Mode*.

Allowed values

A value between 0-2.000, specifying the movement in meters.

The default value is 0.05 meter for both manual and automatic mode.

Related information

[Mode on page 97](#).

Application manual - Motion coordination and supervision.

3.9.6. External Rotation

Parent

External Rotation belongs to the type *Path Return Region*, in the topic *Controller*.

Cfg name

Ext_rot

Description

External Rotation defines the maximum allowed external axes rotation from the current robot position to the last executed path.

Usage

External Rotation is used to limit the regain movement if there is a risk that the robot will collide with an object.

Prerequisites

Specify which operating mode the return movement is valid for. This is defined in the parameter *Mode*.

Allowed values

A value between 0-6.280, specifying the movement in radians.

The default value is 0.2 radians for manual mode and 1.57 radians for automatic mode.

Additional information

To convert degrees to radians, use this formula:

$$\text{radians} = 2\pi \times \text{degrees}/360$$

Related information

[Mode on page 97](#).

Application manual - Motion coordination and supervision.

3 Topic Controller

3.10.1. The Run Mode Settings type

3.10 Type Run Mode Settings

3.10.1. The Run Mode Settings type

Overview block

This section describes the type *Run Mode Settings* which belongs to the topic *Controller*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

RUN_MODE_SETTINGS

Type description

The type *Run Mode Settings* defines if the run mode should change when changing operating mode.

3.10.2. Name

Parent

Name belongs to the type *Run Mode Settings*, in the topic *Controller*.

Cfg name

name

Description

Name of the operating mode setting.

Usage

There can be only one instance with each allowed value, that is a maximum of two instances in the system.

Allowed values

Value	Description
AutoToManual	Defines settings when switching from automatic to manual operating mode.
ManualToAuto	Defines settings when switching from manual to automatic operating mode.

3 Topic Controller

3.10.3. Switch

3.10.3. Switch

Parent

Switch belongs to the type *Run Mode Settings*, in the topic *Controller*.

Cfg name

SwitchTo

Description

Switch defines the run mode when switching operating mode.

Usage

Defines if the run mode should be changed when changing operating mode.

Allowed values

Value	Description
Keep	Keep current run mode.
Single	Set run mode to single cycle.
Continuous	Set run mode to continuous.

3.11 Type Safety Run Chain

3.11.1. The Safety Run Chain type

Overview

This section describes the type *Safety Run Chain* which belongs to the topic *Controller*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

RUNCHN_BOOL

Type description

It is possible to have a delayed stop that gives a smooth stop, this is called a Soft Stop. The type of Soft Stop used in the system is defined in the *Safety Run Chain* type.

There can be more than one set of parameters defined for the type *Safety Run Chain*.

Soft Stop

In the Soft Stop, the robot stops in the same way as a normal program stop with no deviation from the programmed path. After approximately 1 second the power supply to the motors shuts off. The stopping distance can be longer than at a hard stop (e.g. emergency stop) where the power shuts off immediately.

3 Topic Controller

3.11.2. Function

3.11.2. Function

Parent

Function belongs to the type *Safety Run Chain*, in the topic *Controller*.

Cfg name

Name

Description

Function defines if Soft Stop is activated or deactivated.

Usage

A Soft Stop is the same as a safety stop of category 1, which means that the robot will be stopped by the servos.

The robot can easily be started again after a soft stop on path and continue with the activity where it was interrupted.

Allowed values

Value:	Description:
SoftES	Soft emergency stop is activated by pressing the emergency stop push button on the FlexPendant or the control module. SoftES is only used in Auto. In manual mode, SoftES will be a category 0 stop regardless of the value set in the parameter <i>Active</i> .
SoftAS	Soft automatic mode stop is intended for automatic mode during normal program execution. This stop is activated by safety devices such as light curtains, light beams, or sensitive mats.
SoftGS	Soft general stop is activated by safety devices such as light curtains, light beams, or sensitive mats.
SoftSS	Soft superior stop has the same function as a general stop but is intended for externally connected safety devices.

Related information

Operating manual - IRC5 with FlexPendant chapter Safety.

3.11.3. Active

Parent

Active belongs to the type *Safety Run Chain*, in the topic *Controller*.

Cfg name

Select

Description

Active defines whether the Soft Stop is activated or not.

Usage

If *Active* is set to True, the Soft Stop is activated.

Allowed values

TRUE or FALSE.

3 Topic Controller

3.12.1. The System Misc type

3.12 Type System Misc

3.12.1. The System Misc type

Overview

This section describes the type *System Misc*, which belongs to the topic *Controller*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

SYS_MISC

Type description

System Misc contains parameters that are general for the controller.

3.12.2. Name

Parent

Name belongs to the type *System Misc*, in the topic *Controller*.

Cfg name

name

Description

Name defines either of the two arguments for retry or simulated menus.

Limitations

There can be only one instance with *Name* set to *NoOfRetry*, and one set to *SimulateMenu*.

Allowed values

NoOfRetry

SimulateMenu

Related information

NoOfRetry on page 110.

SimulateMenu on page 111.

3 Topic Controller

3.12.3.1. NoOfRetry

3.12.3. Action values

3.12.3.1. NoOfRetry

Parent

NoOfRetry is an action value for the parameter *Name* that belongs to the type *System Misc*, in the topic *Controller*.

Cfg name

NoOfRetry

Description

The action value *NoOfRetry* specifies that there is a limit to the number of times the routine with a recoverable error is called before the error is reported as fatal and execution is stopped. The number of times is set by the parameter *Value*.

Usage

Can be useful e.g. if the network is shaky and the first attempt at opening a file does not work.

Limitations

Works only if an ERROR handler that takes care of the error situation is programmed with the RETRY statement.

Additional information

Changes are activated after a normal restart.

Related information

[Value on page 112.](#)

Example

This example shows that it can take some time before an I/O unit is enabled. Several attempts are needed before it is possible to set the digital output signal.

```
PROC A()
  ...
  IOEnable "cell_1", 0;
  SetDO cell_1_sig3, 1; !This might not work on the first attempt
  ...
  ERROR IF ERRNO = ERR_IOENABLE THEN
    RETRY;
  ENDIF
ENDPROC
```

3.12.3.2. SimulateMenu

Parent

SimulateMenu is an action value for the parameter *Name* that belongs to the type *System Misc*, in the topic *Controller*.

Cfg name

SimulateMenu

Description

The `WaitTime`, `WaitUntil`, `WaitDO`, and `WaitDI` instructions generate an alert box in manual mode to make it possible to simulate the instruction and continue to execute the next instruction. The parameter *Value* defines if *SimulateMenu* is on or off.

Usage

It is useful to switch this parameter off if no alert boxes are desired. Set *Value* to 0 to disable menus.

Limitations

The parameter is only active in manual mode. There are no alert boxes in automatic mode.

Additional information

Changes are activated after a normal restart.

Related information

[Value on page 112](#)[Value on page 112](#).

3 Topic Controller

3.12.4. Value

3.12.4. Value

Parent

Value belongs to the type *Type*, in the topic *Motion*.

Cfg name

value

Description

Defines the values for the action values defined in parameter *Name*.

Allowed values

Name:	Value:	Description:
NoOfRetry	1-1000	Defines number of times the number of times a routine with a recoverable error is called before the system is stopped.
SimulateMenu	0 or 1	Defines if instructions should be possible to simulate in manual mode.

Related information

[Name on page 109](#).

[NoOfRetry on page 110](#).

[SimulateMenu on page 111](#).

3.13 Type Task

3.13.1. The Task type

Overview

This section describes the type *Task*, which belongs to the topic *Controller*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

CAB_TASKS

Type description

Each set of parameters of the *Task* type represents a program task on the controller.

If you have the option *Multitasking*, there can be up to 20 tasks. Otherwise there can be only one.

Related information

Application manual - Engineering tools chapter *Multitasking*.

3 Topic Controller

3.13.2. Task

3.13.2. Task

Parent

Task belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

Name

Description

The name of the task.

Usage

This is the public identity of the task.

Allowed values

A string with maximum 30 characters. The first character may not be a digit.

3.13.3. Task in Foreground

Parent

Task in Foreground belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

Task_in_foreground

Description

Used to set priorities between tasks.

Task in Foreground contains the name of the task that should run in the foreground of this task. This means that the task for which the parameter is set will only execute if the foreground task is idle.

Usage

The default behavior is that all tasks run at the same priority level. If you want to customize the priorities, the *Task in Foreground* parameter can be set for the tasks that should run in the background.

If *Task in Foreground* is set to empty string or to -1 for a task, it runs at the highest priority, i.e. no other task can suspend its execution.

Limitations

The parameter *Task in Foreground* can only be used if you have the option *Multitasking*.

Allowed values

A string with maximum 30 characters.

3 Topic Controller

3.13.4. Type

3.13.4. Type

Parent

Type belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

Type

Description

Controls the start/stop and system restart behavior of a task.

Usage

When creating a new task, use the *Type* parameter to configure how the task should be started.

Limitations

A task that controls a mechanical unit must be of the type NORMAL.

The parameter *Type* can only be used if you have the option *Multitasking*.

Allowed values

Value:	Description:
NORMAL	<p>The task reacts on START/STOP requests given from the FlexPendant or other sources.</p> <p>The task is stopped when an emergency stop occurs.</p>
STATIC	<p>At warm start, the task restarts at the current position.</p> <p>The task is not stopped by emergency stops.</p> <p>The task is normally not stopped by the stop button on the FlexPendant. This can be configured on the FlexPendant by the operator.</p>
SEMISTATIC	<p>The task restarts from the beginning at all warm starts. Modules will be reloaded if the file with automatic loaded modules is updated.</p> <p>The task is not stopped by emergency stops.</p> <p>The task is normally not stopped by the stop button on the FlexPendant. This can be configured on the FlexPendant by the operator.</p>

Default value is SEMISTATIC.

3.13.5. Check Unresolved References

Parent

Check Unresolved References belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

BindRef

Description

Check Unresolved References determines if the system shall check for unresolved references or ignore them.

Usage

This parameter should be set to “0” if the system is to accept unsolved references in the program while linking a module, or otherwise set to “1”.

If set to “1”, a runtime error will occur on execution of an unresolved reference.

Limitations

The parameter has no effect when using instructions Load, StartLoad, WaitLoad, or Erase. In this case the system will never check for unresolved references.

Allowed values

1 or 0.

Default value is 1.

3 Topic Controller

3.13.6. Main Entry

3.13.6. Main Entry

Parent

Main Entry belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

Entry

Description

The name of the start routine for the task.

Usage

The task starts its execution in the routine specified by *Main Entry*. It should be a RAPID routine without any parameters and reachable in this task.

Allowed values

A routine name, with maximum 32 characters.

Default value is main.

3.13.7. TrustLevel

Parent

TrustLevel belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

TrustLevel

Description

TrustLevel handles the system behavior when a SEMISTATIC or STATIC task is stopped or not executable.

Usage

If a task that handles safety supervision stops, it might be dangerous to continue running the task that controls the robot motion. Use *TrustLevel* to set the behavior of NORMAL tasks when a SEMISTATIC or STATIC task stops.

Limitations

The parameter *TrustLevel* can only be used if you have the option *Multitasking*.

Allowed values

Value:	Description:
SysFail	All NORMAL tasks will be stopped. Besides that the system is set to system failure state (SYS_FAIL). All jogging and program start orders will be rejected. Only a new normal restart (warm start) resets the system. This should be used when the task has some safety supervisions.
SysHalt	All NORMAL tasks will be stopped. The system is forced to "motors off". Taking up the system to "motors on" resets the system.
SysStop	All NORMAL tasks will be stopped but are restartable. Jogging is also possible.
NoSafety	Only the task itself will stop.

The default value is SysFail.

3 Topic Controller

3.13.8. Use Mechanical Unit Group

3.13.8. Use Mechanical Unit Group

Parent

Use Mechanical Unit Group belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

UseMechanicalUnitGroup

Description

Defines which mechanical unit group is used for the task.

Usage

A motion task (*MotionTask* set to Yes) controls the mechanical units in the mechanical unit group. A non-motion task (*MotionTask* set to No) will still be able to read values (e.g. the TCP position) for the mechanical units in the mechanical unit group.

Limitations

The parameter *Use Mechanical Unit Group* is only used if you have the option *MultiMove*.

Allowed values

Use Mechanical Unit Group is set to the same value as the parameter *Name* for the type *Mechanical Unit Group*.

A string with maximum 32 characters.

Related information

[MotionTask on page 121](#).

[Name on page 80](#).

[Application manual - MultiMove](#).

3.13.9. MotionTask

Parent

MotionTask belongs to the type *Tasks*, in the topic *Controller*.

Cfg name

MotionTask

Description

Indicates which task is the motion task, e.g. can be able to run RAPID move instructions.

MotionTask must be used even though only one task is configured in the system.

Usage

Set *MotionTask* to YES for the task that will be used for robot move instructions.

Limitations

Only one task in the system can be a motion task unless you have the option *MultiMove*.

The parameter *MotionTask* is only used if you have the option *Multitasking*.

Allowed values

YES or NO.

The default behavior is NO.

The value must be set to YES for one, and only one, task.

Related information

Application manual - MultiMove.

Application manual - Engineering tools.

3 Topic Controller

3.13.10. Hidden

3.13.10. Hidden

Parent

Hidden belongs to the type *Task* in the topic *Controller*.

Cfg name

Hidden

Description

RAPID tasks may be hidden, which may be used to prevent inexperienced end users from tampering (accidentally deleting or changing) with the contents.

Note that the hidden contents is not protected! It can easily be shown again by setting the parameter value to NO.

Note that any hidden contents will still be available when using the `SetDataSearch` instruction to search RAPID data.

Limitation

This parameter is available when using multitasking systems only, such as *MultiMove*.

Changes to the parameter will become effective only after performing a P-start.

Allowed values

YES or NO.

Default value is NO.

4 Topic I/O

4.1. The I/O topic

Overview

This chapter describes the types and parameters of the *I/O* topic. Each parameter is described in the section for its type.

Description

The *I/O* topic contains parameters for I/O boards and signals.

The parameters are organized in the following types:

1. Access Level
2. Bus
3. Cross Connection
4. Fieldbus Command
5. Fieldbus Command Type
6. Signal
7. System Input
8. System Output
9. Unit
10. Unit Type

Configuration results

Changed I/O parameters requires a restart of the controller. Otherwise the changes will have no effect on the system.

4 Topic I/O

4.2.1. How to define I/O units

4.2 Workflows

4.2.1. How to define I/O units

Overview

An I/O unit is a logical software representation of a fieldbus unit that is connected to a fieldbus within the controller. I/O units allow you to control electronic devices and read sensor data. They are used for controlling signals in the robot system.

Available I/O units

Several I/O units can be defined within the robot system. The types of units available depend on what type of fieldbus is being used.

The following I/O units are examples of available units for DeviceNet:

1. Digital I/O
2. Analog I/O
3. AD Combi I/O
4. Relay I/O
5. Gateways
6. Simulated I/O
7. Encoder interface units

Prerequisites

Before defining a Unit, you must:

1. Configure the *Bus*.
2. Make sure the appropriate *Unit Type* is available, either by creating it or using a predefined unit type.

How to define I/O units

To define an I/O unit:

1. In the topic **I/O**, choose the type **Unit**.
2. Select the unit to change, delete or add a new one.
3. Enter, delete or change the values for the parameters.
4. Save the changes.

Related information

[The Unit type on page 243.](#)

[How to define the I/O unit type on page 126.](#)

[The Unit Type type on page 255.](#)

4.2.2. How to list available I/O unit types

Overview

The I/O units are of a specific type, defined for each unit. I/O devices that do not have a unit type equivalent specified, need to be defined.

How to list available I/O unit types

To list all available I/O unit types:

1. In the topic **I/O**, choose the type **Unit Type**.
 2. To get detailed information about a specific unit type, select the unit type.
-

Related information

[How to define I/O units on page 124.](#)

[How to define the I/O unit type on page 126.](#)

4.2.3. How to define the I/O unit type

Overview

The I/O units are of a specific type, defined for each unit. I/O devices that do not have a unit type equivalent specified, need to be defined.

How to define the I/O unit type

To define the I/O unit type:

1. In the topic **I/O**, choose the type **Unit Type**.
 2. Select the Unit Type to change, delete or add a new one.
 3. Enter, delete or change the values for the parameters.
 4. Save the changes.
-

Related information

[How to define I/O units on page 124](#).

[How to list available I/O unit types on page 125](#).

[The Unit Type type on page 255](#).

4.2.4. How to define input and output signals

Overview

An I/O signal is the logical software representation of an I/O signal located on a fieldbus unit that is connected to a fieldbus within the controller.

Available input and output signals

The signals can be of different types.

The type of signals available depends on the actual unit. Typical physical signal types are:

1. Digital inputs and outputs 24 V DC
2. Digital inputs and outputs 120 V DC
3. Analog inputs and outputs +10 V
4. Analog outputs 0 to +10 V

The signal types possible to configure are:

- Digital input, DI
- Digital output, DO
- Analog input, AI
- Analog output, AO
- Group input, GI
- Group output, GO

Limitations

A maximum of 2048 signals can be defined. This includes digital, analog, and group signals of both input and output type.

Prerequisites

Before defining a Signal (that is not Virtual), you must:

1. Configure the *Bus* and
2. Make sure the appropriate *Unit Type* is available, either by creating it or by using a predefined unit type, and
3. Configure the *Unit*, and
4. Make sure the appropriate *Access Level* is available, either by creating it or by using a predefined access level.

How to define input and output signals

To define signals:

1. In the topic **I/O**, choose the type **Signal**.
2. Select the signal to change, delete or add a new one.

4 Topic I/O

4.2.4. How to define input and output signals

Continued

3. Enter, delete or change the values for the parameters. The required parameters depend on the type of signal.

See parameter descriptions and examples of typical configurations in the description of the type *Signal*.

4. Save the changes.

Related information

How to define a signal group on page 129.

The Signal type on page 173.

4.2.5. How to define a signal group

Signal group

Digital signals can be grouped together and be handled as if they were one signal. The value of such a signal will thus be a positive integer that is binary coded using the individual digital signals as a basis.

Limitations

When defining signal groups, you have to consider the following limitation in the robot system:

- A maximum of 16 signals in a signal group can be defined.

How to define a signal group

To define a signal group:

1. In the **I/O** topic, choose the type **Signal**.
2. Select the signal to change, delete or add a new signal.
3. Enter, delete or change the values for the parameters. Set the parameter *Type of Signal* to value *Group Input* or *Group Output*.

The required parameters depend on the type of signal. See parameter descriptions and examples of typical configurations in the description of the type *Signal*.

4. Save the changes.

Related information

[How to define input and output signals on page 127.](#)

[The Signal type on page 173.](#)

Example

If a signal group spans over 4 bits, the maximum value is 15 (2^4-1) and the minimum value is 0.

4 Topic I/O

4.2.6. How to define system inputs

4.2.6. How to define system inputs

Overview

Input signals can be assigned specific inputs. The input triggers a system action that is handled by the system, without using the FlexPendant or other hardware devices.

Prerequisites

A digital input signal with a defined signal name has to be configured in the system.

Limitations

The following limitations have to be considered:

- Only one system action can be assigned to the input signal. However, several input signals can be assigned the same system action.
- When deleting a system action the signal itself remains defined. The signal has to be deleted separately.
- System input signals are only valid for the currently executed program in the system, with exceptions on the action value level. These exceptions are described together with the corresponding action value.
- The system must be in automatic mode to react on the system signal.

How to define system inputs

To define a system input:

1. In the topic **I/O**, choose the type **System Input**.
2. Select the system input to change, delete, or add a new one.
3. Enter, change or delete the values for the parameters.

To add or delete the system action values *Interrupt*, *Load and Start*, *Motors On and Start*, *Start*, and *Start at Main* you must also define the parameter *Argument 1*.

To add or delete the system action values *Interrupt* and *Load and Start* you must also define the parameter *Argument 2*.

4. Save the changes.

Rejected system inputs

If the system is in manual mode or cannot perform the defined system action due to any other unfulfilled requirement, no error message is displayed. When a system action is rejected the error message is stored in the error log (ELOG).

Related information

[The System Input type on page 200.](#)

[The Signal type on page 173.](#)

4.3 Type Access Level

4.3.1. The Access Level type

Overview

This section describes the *Access Level* type which belongs to the topic *I/O*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

EIO_ACCESS

Type description

An I/O *Access Level* is a configuration that defines the write access to different I/O objects for different categories of I/O controlling applications.

Usage

To configure and use an access level it is necessary to limit signal write access. E.g. it should be possible to modify a set of signals from a remote client (e.g. RobotStudio Online) and RAPID in automatic mode. To achieve this it is necessary to create an access level specifying the limitations, and assign the signals to use the new access level.

Limitations

It is not possible to configure different write access levels for different types of remote clients, since the controller does not differ e.g. RobotStudio Online from other remote clients.

Predefined access levels

Access level:	Description:
INTERNAL	No external clients have write access, typically used by safety signals.
DEFAULT	Only allowed to write to signals from RAPID instructions and local clients (e.g. FlexPendant).
ALL	Use with great care since full access are given from all clients (local and remote).

Example

Parameter:	Value:
Name	DEFAULT
Rapid	Write enabled
Local client in manual mode	Write enabled
Local client in auto mode	Read only
Remote client in manual mode	Read only
Remote client in auto mode	Read only

4.3.2. Name

4.3.2. Name

Parent

The parameter *Name* belongs to the type *Access Level*, in the topic *I/O*.

Cfg name

Name

Description

The parameter *Name* specifies the logical name of the access level.

Usage

The name of the access level is used as a reference to the specific access level when configuring the signals.

Allowed values

A string following the RAPID rules described in the manual *RAPID overview*, chapter *Basic elements*.

The name must be unique among all named objects in the I/O system configuration. Note that names differing only in upper and lower case are considered to be equal.

4.3.3. Rapid

Parent

The parameter *Rapid* belongs to the type *Access Level*, in the topic *I/O*.

Cfg name

Rapid

Description

The parameter *Rapid* specifies the level of access granted to RAPID instructions.

Usage

Specify the level of access that should be granted to RAPID instructions when accessing objects associated with this access level.

Allowed values

Write enabled

Read only

4.3.4. Local Client in Manual Mode

Parent

The parameter *Local Client in Manual Mode* belongs to the type *Access Level*, in the topic *I/O*.

Cfg name

LocalManual

Description

The parameter *Local Client in Manual Mode* specifies the level of access granted to local RobAPI clients in manual mode.

A local client is a client using RobAPI and is connected directly to the controller, e.g. a FlexPendant.

Usage

Specify the level of access that should be granted to local RobAPI clients in manual mode when accessing objects associated with this access level.

Allowed values

Write enabled

Read only

4.3.5. Local Client in Auto Mode

Parent

The parameter *Local Client in Auto Mode* belongs to the type *Access Level*, in the topic *I/O*.

Cfg name

LocalAuto

Description

The parameter *Local Client in Auto Mode* specifies the level of access granted to local RobAPI clients in automatic mode.

A local client is a client using RobAPI and is connected directly to the controller, e.g. a FlexPendant.

Usage

Specify the level of access that should be granted to local RobAPI clients in automatic mode when accessing objects associated with this access level.

Allowed values

Write enabled

Read only

4.3.6. Remote Client in Manual Mode

Parent

The parameter *Remote Client in Manual Mode* belongs to the type *Access Level*, in the topic *I/O*.

Cfg name

RemoteManual

Description

The parameter *Remote Client in Manual Mode* specifies the level of access granted to remote RobAPI clients in manual mode.

A remote client is a client or application using RobAPI and not being connected directly to the controller, e.g. RobotStudio Online.

Usage

Specify the level of access that should be granted to remote RobAPI clients in manual mode when accessing objects associated with this access level.

Allowed values

Write enabled

Read only

4.3.7. Remote Client in Auto Mode

Parent

The parameter *Remote Client in Auto Mode* belongs to the type *Access Level*, in the topic *I/O*.

Cfg name

RemoteAuto

Description

The parameter *Remote Client in Auto Mode* specifies the level of access granted to remote RobAPI clients in automatic mode.

A remote client is a client or application using RobAPI and not being connected directly to the controller, e.g. RobotStudio Online.

Usage

Specify the level of access that should be granted to remote RobAPI clients in automatic mode when accessing objects associated with this access level.

Allowed values

Write enabled

Read only

4 Topic I/O

4.4.1. The Bus type

4.4 Type Bus

4.4.1. The Bus type

Overview

This section describes the type *Bus*, which belongs to the topic *I/O*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

EIO_BUS

Type description

An I/O bus is a logical software representation of a fieldbus within the controller.

Usage

By specifying a bus, a logical representation of the real fieldbus is created. The bus configuration defines the specific parameters that will determine the behavior for the fieldbus, like communication speed and bus recovery time.

The bus is used when defining the units and other objects in the I/O system.

Prerequisites

Before defining a new bus it is necessary to have the desired fieldbus option installed in the system.

The fieldbus option typically consists of software to configure buses of the specific type, and the hardware required to equip the controller with the physical interfaces needed for the specific fieldbus.

Limitations

The bus has the following limitations:

- The maximum number of buses in the system depends on the installed fieldbus options.
- It is only possible to configure buses of types for which the respective option has been installed in the system.

Predefined buses

Bus:	Description:
Local	Local is used for communication with the safety I/O boards and the built in 8DI/8DO user I/O unit. No extra units can be configured to this bus beside the user I/O unit.
Virtual1	Virtual1 is a virtual bus that can be used for configuration of virtual units.

Depending on the installed options, there can be other predefined buses not described in this manual.

Continued

Related information

More information about the bus configuration can be found in the manual for the respective fieldbus option.

Application manual - DeviceNet.

Example DeviceNet

This is a typical DeviceNet bus. Please refer to *Application manual - DeviceNet* for more information about DeviceNet.

Parameter:	Value:
Name	MyDeviceNet
Type of bus	DeviceNet
Connector ID	First board
Label at Fieldbus Connector	DeviceNet connector on board in PCI slot 4
Recovery time	5
DeviceNet master address	2
DeviceNet communication speed	250 kbps

4.4.2. Name

Parent

Name belongs to the type *Bus*, in the topic *I/O*.

Cfg name

Name

Description

The parameter *Name* specifies the name of the bus.

Usage

The name of the bus is used as a reference to the specific bus when configuring the units on the bus.

Allowed values

A string following the RAPID rules described in the manual *Technical reference manual - RAPID overview*, chapter *Basic elements*.

The name must be unique among all named objects in the I/O system configuration. Note that names differing only in upper and lower case are considered as equal.

4.4.3. Type of Bus

Parent

Type of Bus belongs to the type *Bus*, in the topic *I/O*.

Cfg name

BusType

Description

The parameter *Type of Bus* specifies the type of fieldbus this bus is representing.

Usage

Defining the type of bus is vital for being able to identify which fieldbus this bus is representing. The unique identification of the specific fieldbus is made by the actual combination of the parameters *Type of Bus* and *Connector ID*.

Prerequisites

The fieldbus option for the desired type of bus must have been installed in the controller.

Limitations

All configured buses in the system must have a unique combination of the parameters *Type of Bus* and *Connector ID*.

Allowed values

Values are determined by the available and installed fieldbus options.

Examples of allowed values, depending on installed fieldbus options:

- Virtual (always available)
- Local (always available)
- DeviceNet
- Profibus
- Interbus
- EtherNet/IP

Additional information

The value Virtual is useful if a configuration from another robot system should be temporarily tested without having all the I/O hardware available. If *Type of Bus* is set to Virtual for buses not available, it is possible to run the system in simulated mode using the imported configuration.

Related information

[Connector ID on page 142](#).

4 Topic I/O

4.4.4. Connector ID

Parent

Connector ID belongs to the type *Bus*, in the topic *I/O*.

Cfg name

ConnectorID

Description

The parameter *Connector ID* specifies where the hardware is located for the specific type of fieldbus being represented by this bus configuration.

Usage

Connector ID is used to connect a bus to a specific hardware. If the bus is representing a hardware bus, for example DeviceNet, *Connector ID* is an enumeration of the hardware (of the same type) located in the computer module. The enumeration is made on a left to right basis, i.e. for the PCI bus the first board is the board located in the slot furthest to the left. If the hardware supports multiple channels (e.g. the dual PCI card for DeviceNet) the enumeration is made from the top to bottom.

Limitations

The fieldbus option installed in the system must support several boards/channels of the same type in the controller, otherwise the value first board or channel is the only valid value.

There can be maximum two boards in the system.

Each board can have maximum two channels.

Allowed values

The values that the parameter *Connector ID* can assume are determined by the available and installed fieldbus options.

Value:	Description:
First board/channel	Always available
Second board/channel	Fieldbus specific
Third channel	Fieldbus specific
Fourth channel	Fieldbus specific

Additional information

If the type of bus is Virtual or if the installed fieldbus option supports several boards of the same type, the value Second board and higher can be available.

4.4.5. Label at Fieldbus Connector

Parent

Label at Fieldbus Connector belongs to the type *Bus*, in the topic *I/O*.

Cfg name

ConnectorLabel

Description

Label at Fieldbus Connector provides a way to label the actual bus.

Usage

Use of the parameter *Label at Fieldbus Connector* is optional. It provides a label to identify the physical bus or connector that this bus configuration is representing.

Allowed values

A string with maximum 80 characters.

4.4.6. Automatic Bus Restart

4.4.6. Automatic Bus Restart

Parent

Automatic Bus Restart belongs to the type *Bus*, in the topic *I/O*.

Cfg name

AutomaticBusRestartDisabled

Description

Automatic Bus Restart is used to enable and disable the automatically recovery for the bus.

Usage

If this parameter is enabled and the bus in some way have stopped (e.g. ended in error state) it tries to automatically recover to running state.

If *Automatic Bus Restart* is set to Disable and the bus have stopped, the bus must be started manually from the Flexpendant or from RAPID.

Allowed values

Enable or Disable.

Related information

The bus can be started manually from RAPID with the function `IOBusStart` , see *Technical reference manual - RAPID overview*.

4.4.7. Recovery Time

Parent

Recovery Time belongs to the type *Bus*, in the topic *I/O*.

Cfg name

RecoveryTime

Description

The parameter *Recovery Time* defines how often bus recovery shall be performed by defining the time of the interval in seconds.

Bus recovery is performed regularly to regain contact with I/O units that the controller has lost communication with.

Limitations

The minimum value is 5 seconds.

Allowed values

Recovery Time is specified in intervals of 5 seconds.

Additional information

Enabling and disabling of I/O units can be done using the RAPID commands `IOEnable` and `IODisable`, see *Technical reference manual - RAPID overview*.

Related information

Technical reference manual - RAPID overview.

4.4.8. Path to Bus Configuration File

4.4.8. Path to Bus Configuration File

Parent

Path to Bus Configuration File belongs to the type *Bus*, in the topic *I/O*.

Cfg name

CfgPath

Description

The parameter *Path to Bus Configuration File* specifies the path to a file containing detailed bus configuration information necessary for some types of fieldbuses.

Prerequisites

The target file for this parameter must be created and transferred to the controller. Creation and transfer must be done according to the requirements for the type of bus being represented by this configuration.

Allowed values

A string following the requirements:

- Maximum 80 characters
- Must be a valid file path

The path can be specified as HOME. This is interpreted as the path to the home directory of the current system.

Example

To target a file with the name my_pbus_cfg.bin located in the home directory of the current system:

HOME\my_pbus_cfg.bin

4.5 Type Cross Connection

4.5.1. The Cross Connection type

Overview

This section describes the type *Cross Connection* which belongs to the topic *I/O*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

EIO_CROSS

Type description

An I/O cross connection is a software-configured connection between digital I/O signals that allow one or several signals to automatically affect the state of other signals.

Usage

Using cross connections is a simple way to interconnect signals and let the system handle I/O activity without having to execute any RAPID code.

Cross connecting signals is a good alternative if there is an input signal in the process that, when activated, automatically activates one or several output signals. If logical conditions between several actor signals is needed, the RobotWare option *Logical Cross Connections* is available.

For detailed information on how to use the *Logical Cross Connections* option, see *Application manual - Robot communication and I/O control*.

Limitations

The maximum number of cross connections handled by the system is limited to 100.

Cross connections must not form a chain that is deeper than 20 levels. A chain is formed when cross connections are interlinked so that a signal that is part of a resultant expression in one cross connection is also part of the actor expression of another cross connection, and so on. The depth of such chain is the number of transitions from the first actor signal to the last resultant signal.

Cross connections must not form closed chains since that would cause infinite evaluation and oscillation. A closed chain appears when cross connections are interlinked so that the chain of cross connections forms a circle.

Ambiguous resultant signals are not allowed since the outcome would depend on the order of evaluation (which cannot be controlled). Ambiguous resultant signals occur when the same signal is resultant in several cross connections.

The expressions are evaluated from left to right, i.e. the priorities of the logical operator OR and the logical operator AND are the same. For clarity, our advise is to avoid mixing the logical operator OR and the logical operator AND in the same expression.

4 Topic I/O

4.5.1. The Cross Connection type

Continued

Additional information

Unless the option *Logical Cross Connections* is installed, only direct cross connections consisting of the two parameters *Actor 1* and *Resultant* signal can be specified.

If the option *Logical Cross Connections* is installed, more complex actor conditions can be constructed by combining up to five different actor signals with operators. The actor signals can also be inverted.

Related information

See the chapter about *Logical Cross Connections* in the manual *Application manual - Robot communication and I/O control*.

4.5.2. Resultant

Parent

Resultant belongs to the type *Cross Connection*, in the topic *I/O*.

Cfg name

Res

Description

The parameter *Resultant* specifies the digital signal to which the result of the condition formed by the actor signals will be stored.

Whenever the outcome of the condition formed by the actor signals is altered the signal will take the same value as that outcome.

Usage

Specify the signal that will be effected by the outcome of the condition formed by the actor signals.

Allowed values

A digital signal that is defined in the system.

4 Topic I/O

4.5.3. Actor 1

4.5.3. Actor 1

Parent

Actor 1 belongs to the type *Cross Connection*, in the topic *I/O*.

Cfg name

Act1

Description

The parameter *Actor 1* specifies the first digital signal that forms the actor expression of the cross connection.

Whenever the value of the signal referred to by *Actor 1* is altered, the logical condition formed by the cross connection will be evaluated and the value of the signal referred to by *Resultant* will be updated (if needed).

Usage

Specify the first of the digital signals that forms the condition that will control the value of the signal referred to by *Resultant*.

If you have the option *Logical Cross Connections*, the *Actor 1* parameter can be part of a more complex statement formed by combining it with other parameters such as *Invert Actor 1*, *Operator 1*, and *Actor 2*.

Allowed values

A digital signal that is defined in the system.

Related information

[Resultant on page 149](#).

4.5.4. Invert Actor 1

Parent

Invert Actor 1 belongs to the type *Cross Connection*, in the topic *I/O*.

Cfg name

Act1_invert

Description

The parameter *Invert Actor 1* specifies whether the inverted value of the signal referred to by parameter *Actor 1* will be used in the evaluation instead of the actual signal value.

Usage

The *Invert Actor 1* parameter can be used when forming complex cross connection expressions by specifying if the inverted value of *Actor 1* should be used.

Limitations

Invert Actor 1 can only be used if you have the RobotWare option *Logical Cross Connections*.

Allowed values

Yes or No.

Related information

[Actor 1 on page 150.](#)

4 Topic I/O

4.5.5. Operator 1

4.5.5. Operator 1

Parent

Operator 1 belongs to the type *Cross Connection*, in the topic *I/O*.

Cfg name

Oper1

Description

The parameter *Operator 1* specifies the logical operation to be performed between the signals referred to by parameter *Actor 1* and *Actor 2*.

Usage

The *Operator 1* parameter is used to specify the logical operation between *Actor 1* and *Actor 2*.

If only one actor signal is used, *Operator 1* is left out.

Prerequisites

By specifying *Operator 1* it is explicitly demanded that the parameter *Actor 2* must also be specified.

Limitations

Operator 1 can only be used if you have the RobotWare option *Logical Cross Connections*.

Allowed values

AND or OR.

Related information

[Actor 1 on page 150](#).

[Actor 2 on page 153](#).

4.5.6. Actor 2

Parent

Actor 2 belongs to the type *Cross Connection*, in the topic *I/O*.

Cfg name

Act2

Description

The parameter *Actor 2* specifies the second digital signal that forms the actor expression of the cross connection.

Whenever the value of the signal referred to by *Actor 2* is altered, the logical condition formed by the cross connection will be evaluated and the value of the signal referred to by *Resultant* will be updated (if needed).

Usage

Specify the second of the digital signals that forms the condition that will control the value of the signal referred to by *Resultant*. If only one actor signal is used, *Actor 2* is left out.

Prerequisites

Actor 2 will be ignored unless the parameter *Operator 1* is specified.

Limitations

Actor 2 can only be used if you have the RobotWare option *Logical Cross Connections*.

Allowed values

A digital signal that is defined in the system.

Related information

Resultant on page 149.

Operator 1 on page 152.

4 Topic I/O

4.5.7. Invert Actor 2

4.5.7. Invert Actor 2

Parent	<i>Invert Actor 2</i> belongs to the type <i>Cross Connection</i> , in the topic <i>I/O</i> .
Cfg name	Act2_invert
Description	The parameter <i>Invert Actor 2</i> specifies whether the inverted value of the signal referred to by parameter <i>Actor 2</i> will be used in the evaluation instead of the actual signal value.
Usage	The <i>Invert Actor 2</i> parameter can be used when forming complex cross connection expressions by specifying if the inverted value of <i>Actor 2</i> should be used.
Limitations	<i>Invert Actor 2</i> can only be used if you have the RobotWare option <i>Logical Cross Connections</i> .
Allowed values	Yes or No.
Related information	Actor 2 on page 153.

4.5.8. Operator 2

Parent

Operator 2 belongs to the type *Cross Connection*, in the topic *I/O*.

Cfg name

Oper2

Description

The parameter *Operator 2* specifies the logical operation to be performed between the signals referred to by parameter *Actor 2* and *Actor 3*.

Usage

The *Operator 2* parameter is used to specify the logical operation between *Actor 2* and *Actor 3*.

If no more than two actor signals are used, then *Operator 2* is left out.

Prerequisites

By specifying *Operator 2* it is explicitly demanded that the parameter *Actor 3* must also be specified.

Limitations

Operator 2 can only be used if you have the RobotWare option *Logical Cross Connections*.

Allowed values

AND or OR.

Related information

[Actor 2 on page 153](#).

[Actor 3 on page 156](#).

4.5.9. Actor 3

Parent

Actor 3 belongs to the type *Cross Connection*, in the topic *I/O*.

Cfg name

Act3

Description

The parameter *Actor 3* specifies the third digital signal that forms the actor expression of the cross connection.

Whenever the value of the signal referred to by *Actor 3* is altered, the logical condition formed by the cross connection will be evaluated and the value of the signal referred to by *Resultant* will be updated (if needed).

Usage

Specify the third of the digital signals that forms the condition that will control the value of the signal referred to by *Resultant*. If no more than two actor signal are used, then *Actor 3* is left out.

Prerequisites

Actor 3 will be ignored unless the parameters *Operator 1* and *Operator 2* are specified.

Limitations

Actor 3 can only be used if you have the RobotWare option *Logical Cross Connections*.

Allowed values

A digital signal that is defined in the system.

Related information

[Resultant on page 149.](#)

[Operator 1 on page 152.](#)

[Operator 2 on page 155.](#)

4.5.10. Invert Actor 3

Parent

Invert Actor 3 belongs to the type *Cross Connection*, in the topic *I/O*.

Cfg name

Act3_invert

Description

The parameter *Invert Actor 3* specifies whether the inverted value of the signal referred to by parameter *Actor 3* will be used in the evaluation instead of the actual signal value.

Usage

The *Invert Actor 3* parameter can be used when forming complex cross connection expressions by specifying if the inverted value of *Actor 3* should be used.

Limitations

Invert Actor 3 can only be used if you have the RobotWare option *Logical Cross Connections*.

Allowed values

Yes or No.

Related information

[Actor 3 on page 156.](#)

4 Topic I/O

4.5.11. Operator 3

4.5.11. Operator 3

Parent

Operator 3 belongs to the type *Cross Connection*, in the topic *I/O*.

Cfg name

Oper3

Description

The parameter *Operator 3* specifies the logical operation to be performed between the signals referred to by parameter *Actor 3* and *Actor 4*.

Usage

The *Operator 3* parameter is used to specify the logical operation between *Actor 3* and *Actor 4*.

If no more than three actor signals are used, then *Operator 3* is left out.

Prerequisites

By specifying *Operator 3* it is explicitly demanded that the parameter *Actor 4* must also be specified.

Limitations

Operator 3 can only be used if you have the RobotWare option *Logical Cross Connections*.

Allowed values

AND or OR.

Related information

[Actor 3 on page 156](#).

[Actor 4 on page 159](#).

4.5.12. Actor 4

Parent

Actor 4 belongs to the type *Cross Connection*, in the topic *I/O*.

Cfg name

Act4

Description

The parameter *Actor 4* specifies the fourth digital signal that forms the actor expression of the cross connection.

Whenever the value of the signal referred to by *Actor 4* is altered, the logical condition formed by the cross connection will be evaluated and the value of the signal referred to by *Resultant* will be updated (if needed).

Usage

Specify the fourth of the digital signals that forms the condition that will control the value of the signal referred to by *Resultant*. If no more than three actor signal are used, then *Actor 4* is left out.

Prerequisites

Actor 4 will be ignored unless the parameters *Operator 1*, *Operator 2* and *Operator 3* are specified.

Limitations

Actor 4 can only be used if you have the RobotWare option *Logical Cross Connections*.

Allowed values

A digital signal that is defined in the system.

Related information

[Resultant on page 149](#).

[Operator 1 on page 152](#).

[Operator 2 on page 155](#).

[Operator 3 on page 158](#).

4 Topic I/O

4.5.13. Invert Actor 4

4.5.13. Invert Actor 4

Parent

Invert Actor 4 belongs to the type *Cross Connection*, in the topic *I/O*.

Cfg name

Act4_invert

Description

The parameter *Invert Actor 4* specifies whether the inverted value of the signal referred to by parameter *Actor 4* will be used in the evaluation instead of the actual signal value.

Usage

The *Invert Actor 4* parameter can be used when forming complex cross connection expressions by specifying if the inverted value of *Actor 4* should be used.

Limitations

Invert Actor 4 can only be used if you have the RobotWare option *Logical Cross Connections*.

Allowed values

Yes or No.

Related information

[Actor 4 on page 159](#).

4.5.14. Operator 4

Parent

Operator 4 belongs to the type *Cross Connection*, in the topic *I/O*.

Cfg name

Oper4

Description

The parameter *Operator 4* specifies the logical operation to be performed between the signals referred to by parameter *Actor 4* and *Actor 5*.

Usage

The *Operator 4* parameter is used to specify the logical operation between *Actor 4* and *Actor 5*.

If no more than four actor signals are used, then *Operator 4* is left out.

Prerequisites

By specifying *Operator 4* it is explicitly demanded that the parameter *Actor 5* must also be specified.

Limitations

Operator 4 can only be used if you have the RobotWare option *Logical Cross Connections*.

Allowed values

AND or OR.

Related information

[Actor 4 on page 159](#).

[Actor 5 on page 162](#).

4.5.15. Actor 5

4.5.15. Actor 5

Parent

Actor 5 belongs to the type *Cross Connection*, in the topic *I/O*.

Cfg name

Act5

Description

The parameter *Actor 5* specifies the fifth digital signal that forms the actor expression of the cross connection.

Whenever the value of the signal referred to by *Actor 5* is altered, the logical condition formed by the cross connection will be evaluated and the value of the signal referred to by *Resultant* will be updated (if needed).

Usage

Specify the fifth of the digital signals that forms the condition that will control the value of the signal referred to by *Resultant*. If no more than four actor signal are used, then *Actor 5* is left out.

Prerequisites

Actor 5 will be ignored unless the parameters *Operator 1*, *Operator 2*, *Operator 3* and *Operator 4* are specified.

Limitations

Actor 5 can only be used if you have the RobotWare option *Logical Cross Connections*.

Allowed values

A digital signal that is defined in the system.

Related information

[Resultant on page 149](#).

[Operator 1 on page 152](#).

[Operator 2 on page 155](#).

[Operator 3 on page 158](#).

[Operator 4 on page 161](#).

4.5.16. Invert Actor 5

Parent

Invert Actor 5 belongs to the type *Cross Connection*, in the topic *I/O*.

Cfg name

Act5_invert

Description

The parameter *Invert Actor 5* specifies whether the inverted value of the signal referred to by parameter *Actor 5* will be used in the evaluation instead of the actual signal value.

Usage

The *Invert Actor 5* parameter can be used when forming complex cross connection expressions by specifying if the inverted value of *Actor 5* should be used.

Limitations

Invert Actor 5 can only be used if you have the RobotWare option *Logical Cross Connections*.

Allowed values

Yes or No.

Related information

[Actor 5 on page 162.](#)

4 Topic I/O

4.6.1. The Fieldbus Command type

4.6 Type Fieldbus Command

4.6.1. The Fieldbus Command type

Overview

This section describes the type *Fieldbus Command*, which belongs to the topic *I/O*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

EIO_COMMAND

Type description

A *Fieldbus Command* is a startup command for a specific unit on a fieldbus.

Usage

The *Fieldbus Command* type is used to send commands to specific units on the fieldbus.

This is done:

- at startup.
- when connecting the unit after a power fail.
- when activating the unit from RobotStudio Online or the FlexPendant.

Several units of the same type, e.g. all d350 units, can receive commands via the type *Fieldbus Command Type*. To address one specific unit, use the type *Fieldbus Command*.

Related information

[The Fieldbus Command Type type on page 168.](#)

Example

Parameter:	Value:
Assigned to unit	My_d350
Type of Fieldbus Command	LinkAddr
Value	5

4.6.2. Assigned to Unit

Parent

Assigned to Unit belongs to the type *Fieldbus Command*, in the topic *I/O*.

Cfg name

Unit

Description

The parameter *Assigned to Unit* specifies the name of the I/O unit to which this fieldbus command is associated.

Allowed values

The parameter must correspond to the name of a defined unit.

Note: Names differing only in upper and lower case are considered to be equal.

Related information

[The Unit type on page 243.](#)

4.6.3. Type of Fieldbus Command

Parent

Type of Fieldbus Command belongs to the type *Fieldbus Command*, in the topic *I/O*.

Cfg name

CommandType

Description

The parameter *Type of Fieldbus Command* is a reference to a fieldbus command type that describes this fieldbus command.

Allowed values

The parameter must correspond to the name of a defined fieldbus command type.

Note: Names differing only in upper and lower case are considered to be equal.

Related information

The Fieldbus Command Type type on page 168.

4.6.4. Value

Parent

Value belongs to the type *Type of Fieldbus*, in the topic *I/O*.

Cfg name

Value

Description

The parameter *Value* specifies the value of the command for the unit specified.

Usage

The controller sends commands to the fieldbus on startup. Use the type *Fieldbus Command* to address specific units on the fieldbus and the type *Fieldbus Command Type* to address all units of a specific type.

Allowed values

A string with maximum 32 characters.

Related information

The Fieldbus Command Type type on page 168.

4 Topic I/O

4.7.1. The Fieldbus Command Type type

4.7 Type Fieldbus Command Type

4.7.1. The Fieldbus Command Type type

Overview

This section describes the type *Fieldbus Command Type*, which belongs to the topic *I/O*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

EIO_COMMAND_TYPE

Type description

A *Fieldbus Command Type* is a startup command for a specific type of unit on a fieldbus.

Usage

The type *Fieldbus Command Type* is used at startup to send commands to a specific type of unit on the fieldbus, e.g. all units of the type d350.

Additional information

The type *Fieldbus Command Type* has lower priority than the type *Fieldbus Command*.

Related information

[The Fieldbus Command type on page 164.](#)

Example

Parameter:	Value:
Name	LinkAddr
Type of unit	d350
Default value	1
Download order	1
Path	6,20 64 24 01 30 01,C6,1
Service	Set Attribute Single

4.7.2. Name

Parent

Name belongs to the type *Fieldbus Command Type*, in the topic *I/O*.

Cfg name

Name

Description

The parameter *Name* specifies the logical name of the fieldbus command type.

Usage

The name of the fieldbus command type is used as a reference to the specific fieldbus command type when configuring the fieldbus commands.

Allowed values

A string following the RAPID rules described in the manual *Technical reference manual - RAPID overview*, chapter *Basic elements*.

4.7.3. Type of Unit

4.7.3. Type of Unit

Parent

Type of Unit belongs to the type *Fieldbus Command Type*, in the topic *I/O*.

Cfg name

UnitType

Description

The parameter *Type of Unit* specifies which type of fieldbus unit this fieldbus command type is associated with.

Allowed values

The parameter must correspond to the name of a defined *Unit Type*.

Note: Names differing only in upper and lower case are considered to be equal.

Related information

The Unit Type type on page 255.

4.7.4. Default Value

Parent

Default Value belongs to the type *Fieldbus Command Type*, in the topic *I/O*.

Cfg name

DefValue

Description

The parameter *Default Value* specifies the default value for commands based on this fieldbus command type.

Usage

If a fieldbus command of this fieldbus command type has its parameter *Value* omitted, this default value will act as substitute.

Allowed values

A string with maximum 32 characters.

Related information

Value on page 167.

4.7.5. Download Order

Parent

Download Order belongs to the type *Fieldbus Command Type*, in the topic *I/O*.

Cfg name

OrderNr

Description

The parameter *Download Order* specifies the sequence number in which fieldbus commands of this fieldbus command type shall be downloaded to a unit that has more than one fieldbus command assigned to it.

Usage

Use *Download Order* to control the order in which the fieldbus commands are downloaded (and executed) on a unit.

Lower download orders are downloaded **before higher download orders**.

Allowed values

0 - 100.

4.8 Type Signal

4.8.1. The Signal type

Overview

This section describes the type *Signal*, which belongs to the topic *I/O*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

EIO_SIGNAL

Type description

An I/O signal is the logical software representation of an I/O signal located on a fieldbus unit that is connected to a fieldbus within the controller.

Usage

By specifying a signal, a logical representation of the real I/O signal is created. The signal configuration defines the specific system parameters for the signal that will control the behavior of the signal.

Many of the parameters depend on the type of signal, therefore it is recommended that the parameter *Type of Signal* is assigned first.

Prerequisites

Before defining a new signal it is necessary to make sure that the appropriate *Unit* and *Access Level* are available (either by creating them or using a predefined *Unit* respectively *Access Level*).

Limitations

The signal has the following limitations:

- Maximum number of signals in the system is 2048.
- Maximum number of signals on one (1) unit is 512.

Predefined signals

There are a number of predefined signals in the robot controller. These are described in *Operating manual - IRC5 with FlexPendant*.

Depending on options installed there can also be other predefined signals.

Related information

[The Unit type on page 243.](#)

[The Access Level type on page 131.](#)

Operating manual - IRC5 with FlexPendant.

Application manual - DeviceNet.

Application manual - ProfiBus.

Application manual - InterBus.

Continues on next page

4 Topic I/O

4.8.1. The Signal type

Continued

Example digital input

This is a typical example of a digital input signal, DI.

Parameter:	Value:
Name	ObjectAtPlace
Type of Signal	Digital Input
Assigned to Unit	board10
Signal Identification Label	X4:4
Unit Mapping	11
Category	
Access Level	DEFAULT
Default Value	0
Filter Time Passive	0
Filter Time Active	0
Invert Physical Value	No

Example analog output

This is a typical example of an analog output signal, AO.

Parameter:	Value:
Name	Speed
Type of Signal	AO
Assigned to Unit	board10
Signal Identification Label	X6:4
Unit Mapping	16-31
Category	
Access Level	DEFAULT
Default Value	0
Store Signal Value at Power Fail	No
Analog Encoding Type	Twos complement
Maximum Logical Value	21474.8
Maximum Physical Value	10
Maximum Physical Value Limit	0
Maximum Bit Value	32767
Minimum Logical Value	-21474.8
Minimum Physical Value	-10
Minimum Physical Value Limit	0
Minimum Bit Value	-32767
Signal Value in System Failure State	Keep current value (no change)

Continued

Example group input

This is a typical example of a group input signal (GI).

Parameter:	Value:
Name	StatusGroup
Type of Signal	Group input
Assigned to Unit	board10
Signal Identification Label	X2:1-X2:8
Unit Mapping	0-7
Category	
Access Level	DEFAULT
Default Value	0
Invert Physical Value	No

4.8.2. Name

4.8.2. Name

Parent

Name belongs to the type *Signal*, in the topic *I/O*.

Cfg name

Name

Description

The parameter *Name* specifies the name of the signal.

Usage

The name of the signal is used as a reference to the specific signal when:

- Accessing the signal (i.e. reading or writing its value) in RAPID
 - Configuring cross connections
 - Configuring system inputs and system outputs
-

Allowed values

A string following the RAPID rules described in the manual *Technical reference manual - RAPID overview*, chapter *Basic elements*.

The name must be unique among all named objects in the I/O system configuration. Note that names differing only in upper and lower case are considered to be equal.

Related information

[The Cross Connection type on page 147](#).

[The System Input type on page 200](#).

[The System Output type on page 222](#).

4.8.3. Type of Signal

Parent

Type of Signal belongs to the type *Signal*, in the topic *I/O*.

Cfg name

SignalType

Description

Type of Signal specifies the signal's representation, behavior and direction.

Usage

Each signal must be classified as one of the predefined types. The type of signal will determine the behavior of the signal as well as how it will be represented and interpreted.

As the behavior of the signal depends upon its type, the settings of other parameters will vary, therefore it is recommended that the *Type of Signal* parameter is assigned before any other parameter of the signal.

Allowed values

- Digital Input
- Digital Output
- Analog Input
- Analog Output
- Group Input
- Group Output

4.8.4. Assigned to Unit

4.8.4. Assigned to Unit

Parent

Assigned to Unit belongs to the type *Signal*, in the topic *I/O*.

Cfg name

Unit

Description

The parameter *Assigned to Unit* specifies which unit the signal is connected to (if any).

Usage

A unit is made physical by mapping a signal to it. This means that the unit is physical in the sense that the signal is associated with a real I/O signal (a signal that exists on the fieldbus). Note also that signals mapped against virtual units will be considered and treated as physical signals.

If the signal is not mapped against a unit (i.e. the *Assigned to Unit* is unspecified) it will be considered as virtual.

Allowed values

The parameter must either be unspecified (i.e. denote a virtual signal) or it must correspond to the name of a defined unit.

Note that names differing only in upper and lower case are considered to be equal.

Related information

The Unit type on page 243.

4.8.5. Signal Identification Label

Parent

Signal Identification Label belongs to the type *Signal*, in the topic *I/O*.

Cfg name

SignalLabel

Description

The parameter *Signal Identification Label* provides a free-text label to a signal.

Usage

Signal Identification Label is optional for use in providing a label for the physical contact or cable that this signal configuration represents.

Assign an easy-to-understand name (free-text) to the signal to make it easy to physically identify the signal (e.g. map the signal to a physical identification such as a cable marking or an outlet label).

Allowed values

A string of maximum 80 characters.

Example

Conn. X4, Pin 1

4.8.6. Unit Mapping

4.8.6. Unit Mapping

Parent

Unit Mapping belongs to the type *Signal*, in the topic *I/O*.

Cfg name

UnitMap

Description

The parameter *Unit Mapping* specifies which bit(s) in the I/O memory map of the assigned unit the signal is mapped to.

Usage

All physical signals (i.e. signals connected to a physical unit) must be mapped.

Limitations

A signal must be completely mapped to bits on the same unit, e.g. it is not possible to map a group signal to bits on different units.

Allowed values

A string with maximum 80 characters.

The string should contain the mapping order of the individual bits of the signal, using the following syntax:

- Refer to a bit in the I/O memory map by the index of the bit, the bits are indexed from 0 (zero) and upward
- If the signal is mapped to several continuous bits, these can be given as a range:<first bit in range> - <last bit in range>
- If the signal is mapped to several discontinuous bits and/or ranges, these should be separated by commas:<bit/range>, <bit/range>, <bit/range>

Additional information

Signals can be mapped to bits in the range 0 to 511 unless the physical characteristics of the unit to which the signal is assigned adds further constraints on this range.

Overlapping unit maps is not allowed, i.e. the *Unit Mapping* must not refer to the same bit(s) more than once.

The size of the signal (i.e. the number of bits in the *Unit Mapping*) is restricted. The restriction depends upon the type of signal:

- Digital signals must be mapped to exactly one bit
- Analog signals must be mapped between 2 and 32 bits
- Group signals must be mapped between 2 and 32 bits

Mapping more than one signal against the same bit(s) in the I/O memory map can cause unpredictable start-up values for these signals since their order of evaluation cannot be controlled. E.g. if an inverted group output signal is mapped to the same bits as some digital

Continued

output signals, the status of these bits will depend on the order in which the signals are dealt with, a feature that cannot be controlled.

It is highly recommended that mapping of several signals to the same bit(s) be avoided.

Example

Examples of valid mapping of a digital signal (1 bit):

- 0
- 13

Examples of valid mapping of an analog or group signal (2-32 bits):

- 4, 6-7
- 16-31
- 8-15, 0-7

Example of invalid mapping (bit 7 is overlapped):

- 0-7, 15-7

4.8.7. Category

4.8.7. Category

Parent

Category belongs to the type *Signal*, in the topic *I/O*.

Cfg name

Category

Description

The parameter *Category* provides a free-text categorizing to a signal.

Usage

Category is optional for use in categorizing the signals in different categories so that tools (e.g. software tools) will be able to offer filtering and sorting of signals based on these categories.

Limitations

Signals defined as Safety or Internal are hidden for the user by RobotStudio Online and FlexPendant.

Allowed values

A string of maximum 16 characters.

Additional information

The category of all safety-related signals (internally loaded by the system) are set to Safety.

4.8.8. Access Level

Parent

Access Level belongs to the type *Signal*, in the topic *I/O*.

Cfg name

Access

Description

The parameter *Access Level* specifies which clients have write access to the signal.

Usage

Access Level defines the write access of the I/O signal for different categories of I/O controlling applications, such as RobotStudio Online and RAPID programs.

Allowed values

A string corresponding to the name of a defined *Access Level* type.

Note: Names differing only in upper and lower case are considered to be equal.

Related information

The Access Level type on page 131.

4 Topic I/O

4.8.9. Default Value

4.8.9. Default Value

Parent

The parameter *Default Value* belongs to the type *Signal*, in the topic *I/O*.

Cfg name

Default

Description

The parameter *Default Value* specifies the signal value to be used at start-up.

Usage

The default value:

- will be used for initializing the signal at start-up if the signal is neither stored nor a resultant in a cross connection.
- is used for evaluation of cross connections whenever the unit to which this signal is assigned is disconnected.
- is not used after a system failure if the value of *Signal Value at System Failure and Power Fail* is set to *Keep current value*.

Limitations

This parameter has lower priority than the *Store Signal Value at Power Fail* parameter.

Allowed values

Depending on the type of signal, the following values are allowed:

Type of signal:	Allowed value:
Digital	0 or 1
Analog	Any value in the range <i>Minimum Logical Value</i> to <i>Maximum Logical Value</i>
Group	Any value in the range 0 to $2^{\text{size}} - 1$ (<i>size</i> = number of bits in the <i>Unit Mapping</i> parameter)

Related information

[Store Signal Value at Power Fail on page 185](#).

[Signal Value at System Failure and Power Fail on page 199](#).

[Minimum Logical Value on page 195](#).

[Maximum Logical Value on page 190](#).

[Unit Mapping on page 180](#).

4.8.10. Store Signal Value at Power Fail

Parent

Store Signal Value at Power Fail belongs to the type *Signal*, in the topic *I/O*.

Cfg name

Store

Description

The parameter *Store Signal Value at Power Fail* specifies whether the signal should be set to the value stored in the permanent memory pool or not at start up.

Usage

Setting this parameter will cause the system to initialize the signal with the last known value at start up, i.e. the value that was stored in the permanent memory pool at the previous power down.

Prerequisites

This parameter is applicable on output signals only, i.e. the *Type of Signal* must be set to one of the output signal types or this parameter will be ignored.

Allowed values

Yes or No.

Additional information

This parameter has higher priority than the *Default Value* parameter.

Related information

[Default Value on page 184.](#)

[Type of Signal on page 177.](#)

4.8.11. Filter Time Passive

Parent

Filter Time Passive belongs to the type *Signal*, in the topic *I/O*.

Cfg name

FiltPas

Description

The parameter *Filter Time Passive* specifies the filter time for detection of negative flanks (i.e. physical signal value goes from active to passive).

Usage

The passive filter time filters signals from noise that could otherwise be interpreted as a pulse of the signal.

The passive filter time specifies the period in ms (milliseconds) that the physical value of the signal must remain passive before the signal will be considered passive and the logical signal is changed to passive, i.e. if the time period that the physical value is passive is shorter than *Filter Time Passive*, the logical signal is not changed.

Prerequisites

This parameter is applicable on digital input and group input signals only, i.e. the *Type of Signal* parameter must be set to one of these types or this parameter will be ignored.

Allowed values

Value:	Description:
0	No filter
10-32000	Filter time in ms

Additional information

Note that many units have built-in hardware for filtering signals. This filter time is then added to the value of *Filter Time Passive*.

Related information

[Type of Signal on page 177.](#)

[Filter Time Active on page 187.](#)

4.8.12. Filter Time Active

Parent

Filter Time Active belongs to the type *Signal*, in the topic *I/O*.

Cfg name

FiltAct

Description

The parameter *Filter Time Active* specifies the filter time for detection of positive flanks (i.e. physical signal value goes from passive to active).

Usage

The active filter time filters signals from noise that could otherwise be interpreted as a pulse of the signal.

The active filter time specifies the period in ms (milliseconds) that the physical value of the signal must remain active before the signal will be considered active and the logical signal is changed to active, i.e. if the time period that the physical value is active is shorter than *Filter Time Active*, the logical signal is not changed.

Prerequisites

This parameter is applicable on digital input and group input signals only, i.e. the *Type of Signal* parameter must be set to one of these types or this parameter will be ignored.

Allowed values

Value:	Description:
0	No filter
10 - 32000	Filter time in ms

Additional information

Note that many units have built-in hardware for filtering signals. This filter time is then added to the value of *Filter Time Active*.

Related information

[Type of Signal on page 177.](#)

[Filter Time Passive on page 186.](#)

4 Topic I/O

4.8.13. Invert Physical Value

4.8.13. Invert Physical Value

Parent

Invert Physical Value belongs to the type *Signal*, in the topic *I/O*.

Cfg name

Invert

Description

The parameter *Invert Physical Value* specifies whether the physical representation should be the inverted of the logical representation.

Usage

Use this parameter to apply an inversion between the physical value of the signal and its logical representation in the system.

How to invert the signal depends on the direction of the signal (see *Type of Signal*):

- The logical value of an **input** signal will be the inversion of its physical value
- The physical value of an **output** signal will be the inversion of its logical value.

Inverting a grouped signal will make each individual bit in the group inverted.

Prerequisites

This parameter is only applicable to non-analog signals, i.e. the *Type of Signal* must be set to one of the digital signal or group signal types or this parameter will be ignored.

Allowed values

Yes or No.

Related information

[Type of Signal on page 177.](#)

4.8.14. Analog Encoding Type

Parent

Analog Encoding Type belongs to the type *Signal*, in the topic *I/O*.

Cfg name

EncType

Description

The parameter *Analog Encoding Type* specifies how the value of an analog signal is interpreted.

Usage

Use this parameter to specify whether the physical representation of an analog signal should be interpreted as a signed (twos complement) or unsigned value.

Prerequisites

This parameter is only applicable on analog signals, i.e. the *Type of Signal* must be set to one of the analog signal types or this parameter will be ignored.

Allowed values

Value:	Description:
Twos complement	If the physical analog range for a specific signal is symmetric around 0, e.g. -32768 to +32767, the signal is most likely coded as twos complement.
Unsigned	Unsigned is used for signals ranging from 0 and upwards.

Related information

Type of Signal on page 177.

4 Topic I/O

4.8.15. Maximum Logical Value

Parent

Maximum Logical Value belongs to the type *Signal*, in the topic *I/O*.

Cfg name

MaxLog

Description

The parameter *Maximum Logical Value* specifies the logical value that will correspond to the *Maximum Physical Value*.

Usage

The logical values offer a way to access the signals (e.g. through RAPID programs) by using logical quantities rather than physical.

By setting up the extremes (minimum and maximum values) of the logical and physical values the system will be able to calculate scale and offset factors for transforming signal values between the different quantities.

Prerequisites

This parameter is only applicable to analog signals, i.e. the *Type of Signal* must be set to one of the analog signal types or this parameter will be ignored.

Limitations

The value must be greater than the value of the *Minimum Logical Value*.

Allowed values

-3.4×10^{38} to 3.4×10^{38}

If both *Minimum Logical Value* and *Maximum Logical Value* are set to zero (0), the logical values will be directly mapped against the physical values:

- minimum logical value = minimum physical value
- maximum logical value = maximum physical value

Hence there is no scaling or offset factor between the logical and physical representation of the value of a signal.

Additional information

The logical value is a representation of a signal that makes it possible to handle the signal in quantities known from the real world feature it corresponds to rather than the physical value used to control it. E.g. it would be more natural to set the speed of a moving axis in mm/s (the logical value) rather than the amount of voltage needed to attain that speed (the physical value).

Continued

Related information

Minimum Logical Value on page 195.

Maximum Physical Value on page 192.

Minimum Physical Value on page 196.

Type of Signal on page 177.

4 Topic I/O

4.8.16. Maximum Physical Value

4.8.16. Maximum Physical Value

Parent

Maximum Physical Value belongs to the type *Signal*, in the topic *I/O*.

Cfg name

MaxPhys

Description

The parameter *Maximum Physical Value* specifies the physical value that will correspond to the *Maximum Bit Value*.

Usage

The physical value directly corresponds to the value of the I/O signal that this system parameter corresponds to, e.g. the amount of voltage given by a sensor or the current feed into a manipulator.

By setting up the extremes (minimum and maximum values) of the bit and physical values the system will be able to calculate scale and offset factors for transforming signal values between the bit and physical quantities.

Prerequisites

This parameter is only applicable to analog signals, i.e. the *Type of Signal* must be set to one of the analog signal types or this parameter will be ignored.

Limitations

The value must be greater than the value of the *Minimum Physical Value*.

Allowed values

-3.4×10^{38} to 3.4×10^{38}

If both *Minimum Physical Value* and *Maximum Physical Value* are set to zero (0), the physical values will be directly mapped against the bit values:

- minimum physical value = minimum bit value
- maximum physical value = maximum bit value

Hence there is no scaling or offset factor between the physical and bit representation of the value of a signal.

Related information

[Minimum Physical Value on page 196](#).

[Maximum Logical Value on page 190](#).

[Maximum Bit Value on page 194](#).

[Minimum Bit Value on page 198](#).

[Type of Signal on page 177](#).

4.8.17. Maximum Physical Value Limit

Parent

Maximum Physical Value Limit belongs to the type *Signal*, in the topic *I/O*.

Cfg name

MaxPhysLimit

Description

The parameter *Maximum Physical Value Limit* specifies the maximum allowed physical value, acting as a working range limiter.

Usage

The *Maximum Physical Value Limit* limits the allowed maximum physical value, e.g. if a bit or logical value is given that would exceed this limit, the physical value is automatically adjusted to *Maximum Physical Value Limit*.

Prerequisites

This parameter is only applicable to analog signals, i.e. the *Type of Signal* must be set to one of the analog signal types or this parameter will be ignored.

Limitations

The value must be greater than the value of the *Minimum Physical Value Limit*.

Allowed values

-3.4×10^{38} to 3.4×10^{38}

If both *Minimum Physical Value Limit* and *Maximum Physical Value Limit* are set to zero (0), the physical value limits will be directly mapped against the physical values:

- minimum physical value limit = minimum physical value
 - maximum physical value limit = maximum physical value
-

Related information

[Minimum Physical Value on page 196](#).

[Maximum Physical Value on page 192](#).

[Type of Signal on page 177](#).

4 Topic I/O

4.8.18. Maximum Bit Value

Parent

Maximum Bit Value belongs to the type *Signal*, in the topic *I/O*.

Cfg name

MaxBitVal

Description

The parameter *Maximum Bit Value* specifies the bit value that will correspond to the *Maximum Logical Value*.

Usage

The bit value is the signals representation when transmitted on the fieldbus.

The bit value is used when calculating the physical and logical values.

Prerequisites

This parameter is only applicable to analog signals, i.e. the *Type of Signal* must be set to one of the analog signal types or this parameter will be ignored.

Limitations

The value must be greater than the value of the *Minimum Bit Value*.

Allowed values

-2,147,483,648 to 2,147,483,647

If both *Minimum Bit Value* and *Maximum Bit Value* are set to zero (0) then the bit values will be calculated based on the selected *Analog Encoding Type*.

If *Analog Encoding Type* is set to Twos complement:

- maximum bit value = $2^{(\text{no of bits in Unit Mapping})-1} - 1$
- minimum bit value = $2^{(\text{no of bits in Unit Mapping})-1}$

If *Analog Encoding Type* is set to Unsigned:

- maximum bit value = $2^{(\text{no of bits in Unit Mapping})-1} - 1$
 - minimum bit value = 0
-

Related information

[Minimum Bit Value on page 198](#).

[Maximum Logical Value on page 190](#).

[Maximum Physical Value on page 192](#).

[Analog Encoding Type on page 189](#).

[Type of Signal on page 177](#).

4.8.19. Minimum Logical Value

Parent

Minimum Logical Value belongs to the type *Signal*, in the topic *I/O*.

Cfg name

MinLog

Description

The parameter *Minimum Logical Value* specifies the logical value that will correspond to the *Minimum Physical Value*.

Usage

See *Maximum Logical Value*.

Prerequisites

This parameter is only applicable to analog signals, i.e. the *Type of Signal* must be set to one of the analog signal types or this parameter will be ignored.

Limitations

The value must be less than the value of the *Maximum Logical Value*.

Allowed values

See *Maximum Logical Value*.

Related information

[Maximum Logical Value on page 190](#).

[Minimum Physical Value on page 196](#).

[Type of Signal on page 177](#).

4.8.20. Minimum Physical Value

Parent

Minimum Physical Value belongs to the type *Signal*, in the topic *I/O*.

Cfg name

MinPhys

Description

The parameter *Minimum Physical Value* specifies the physical value that will correspond to the *Minimum Logical Value*.

Usage

See *Maximum Physical Value*.

Prerequisites

This parameter is only applicable to analog signals, i.e. the *Type of Signal* must be set to one of the analog signal types or this parameter will be ignored.

Limitations

The value must be less than the value of the *Maximum Physical Value*.

Allowed values

See *Maximum Physical Value*.

Related information

[Maximum Physical Value on page 192](#).

[Type of Signal on page 177](#).

4.8.21. Minimum Physical Value Limit

Parent

Minimum Physical Value Limit belongs to the type *Signal*, in the topic *I/O*.

Cfg name

MinPhysLimit

Description

The parameter *Minimum Physical Value Limit* specifies the minimum allowed physical value, hence it acts as a working range limiter.

Usage

See *Maximum Physical Value Limit*.

Prerequisites

This parameter is only applicable to analog signals, i.e. the *Type of Signal* must be set to one of the analog signal types or this parameter will be ignored.

Limitations

The value must be less than the value of the *Maximum Physical Value Limit*.

Allowed values

See *Maximum Physical Value Limit*.

Related information

[Maximum Physical Value Limit on page 193](#).

[Type of Signal on page 177](#).

4.8.22. Minimum Bit Value

Parent

Minimum Bit Value belongs to the type *Signal*, in the topic *I/O*.

Cfg name

MinBitVal

Description

The parameter *Minimum Bit Value* specifies the bit value that will correspond to the *Minimum Logical Value*.

Usage

See *Maximum Bit Value*.

Prerequisites

This parameter is only applicable to analog signals, i.e. the *Type of Signal* must be set to one of the analog signal types or this parameter will be ignored.

Limitations

The value must be less than the value of the *Maximum Bit Value*.

Allowed values

See *Maximum Bit Value*.

Related information

[Maximum Bit Value on page 194](#).

[Type of Signal on page 177](#).

4.8.23. Signal Value at System Failure and Power Fail

Parent

Signal Value at System Failure and Power Fail belongs to the type *Signal*, in the topic *I/O*.

Cfg name

SysfailReset

Description

The parameter *Signal Value at System Failure and Power Fail* specifies whether this output signal should keep its current value or take the signal's default value in case of system failure or at a power fail.

Usage

If a signal is required to assume a specific value when the power to the robot controller is shut off then it can be useful to use this parameter.

Prerequisites

This parameter is only applicable to output signals, i.e. the *Type of Signal* must be set to one of the output signal types or this parameter will be ignored.

Allowed values

Keep current value (no change).

Set the default value.

Related information

[Type of Signal on page 177.](#)

4 Topic I/O

4.9.1. The System Input type

4.9 Type System Input

4.9.1. The System Input type

Overview

This section describes the type *System Input* which belongs to the topic *I/O*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

SYSSIG_IN

Type description

Input signals can be assigned specific inputs, e.g. Start or Motors on. The input triggers a system action that is handled by the system, without using the FlexPendant or other hardware devices.

It is possible to use a PLC to trigger the system inputs.

Rejected system inputs

If the system is in manual mode or cannot perform the action due to any other unfulfilled requirement, no error messages are displayed. When a system action is rejected the error messages are stored in the error log.

Limitations

The following limitations have to be considered:

- Only one system action can be assigned to the input signal. However, several input signals can be assigned the same system action.
- When deleting a system action the input signal itself remains defined. The signal has to be deleted separately.
- System input signals are only valid for the Main task, with exceptions on the action value level. These are described together with the corresponding action value.

Additional information

All system inputs are 0 to 1 level sensitive. The pulse length has to exceed 50 ms or according to the configured filter settings for I/O signals.

Related information

[How to define system inputs on page 130.](#)

[Filter Time Passive on page 186.](#)

[Filter Time Active on page 187.](#)

4.9.2. Signal Name

Parent

Signal Name belongs to the type *System Inputs* in the topic *I/O*.

Cfg name

Signal

Description

Signal Name is the name of the configured digital input signal to use. It connects the system input with a configured digital input signal.

Prerequisite

A digital input signal with a defined name has to be configured in the system.

Allowed values

Available configured digital input signal names.

Related information

The type [The Signal type on page 173](#).

[How to define system inputs on page 130](#).

4.9.3. Action

4.9.3. Action

Parent

Action belongs to the type *System Inputs* in the topic *I/O*.

Cfg name

Action

Description

Input signals can be assigned specific system actions. *Action* defines the system action to be triggered by the signal. The system actions are handled by the system without input from a user.

Allowed values

The following values are allowed and described on the following pages:

- [Motors On on page 203](#).
- [Motors On and Start on page 204](#).
- [Motors Off on page 205](#).
- [Load and Start on page 206](#).
- [Interrupt on page 208](#).
- [Start on page 210](#).
- [Start at Main on page 211](#).
- [Stop on page 212](#).
- [Quick Stop on page 213](#).
- [Soft Stop on page 214](#).
- [Stop at End of Cycle on page 215](#).
- [Stop at End of Instruction on page 216](#).
- [Reset Execution Error Signal on page 217](#).
- [Reset Emergency Stop on page 218](#).
- [System Restart on page 219](#).

Related information

[How to define system inputs on page 130](#).

4.9.4. Action Values

4.9.4.1. Motors On

Parent

Motors On is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O*.

Cfg name

MotorOn

Description

The action value *Motors On* sets the controller in the Motors On state.

This action can be used by a PLC to set the controller in Motors On state.

Prerequisites

The following prerequisites have to be considered:

- A digital signal input with a defined signal name has to be configured in the system.
- The safety chain has to be closed. To check if the safety chain is closed, use the parameter *Run Chain OK* of the type *System Output*.

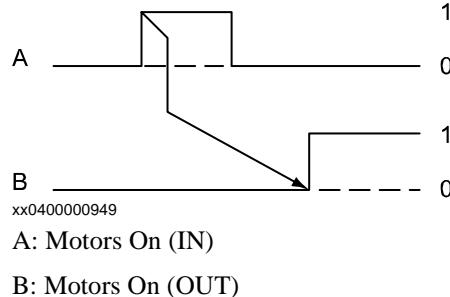
Limitations

The action value has the following limitations:

- The controller has to be in automatic mode.
- The controller cannot be in the Motors On state if the system input signal action *Motors Off* is high.

Additional information

The signal sequences for *Motors On* is:



Related information

[Motors Off on page 205](#).

[Run Chain OK on page 238](#).

[Operating manual - IRC5 with FlexPendant, chapter Handling inputs and outputs, I/O.](#)

4 Topic I/O

4.9.4.2. Motors On and Start

4.9.4.2. Motors On and Start

Parent

Motors On and Start is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O*.

Cfg name

MotOnStart

Description

The action value *Motors On and Start* sets the controller in the Motors On state and starts the RAPID program from the current instruction, continuous or cycle execution.

Motor On and Start can be used by a PLC to set Motors On in one single step and start a RAPID program, instead of using the FlexPendant and the control panel.

Arguments

When the parameter *Action* is set to *Motors On and Start*, the parameter *Argument1* must also be used, specifying *Continuous* or *Cycle*. The default value is *Continuous*.

Prerequisites

The following prerequisites have to be considered:

- A digital signal input with a defined signal name has to be configured in the system.
- The parameter *Argument 1* has to be defined with the run mode Continuous or Cycle.

Limitations

The action value has the following limitations:

- The controller has to be in automatic mode.
- You cannot use this action value if the *Stop*, *Stop at end of Cycle*, *Stop at end of Instruction* or *Motors Off* actions are set.

Related information

[Argument 1 on page 220](#).

[Action on page 202](#).

[Operating manual - IRC5 with FlexPendant chapter Handling inputs and outputs, I/O](#).

4.9.4.3. Motors Off

Parent

Motors Off is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O*.

Cfg name

MotorOff

Description

The action value *Motors Off* sets the controller in the Motors Off state. If a program is executing, it is stopped before changing state.

Motors Off can be used by a PLC to set the controller in Motors Off state.

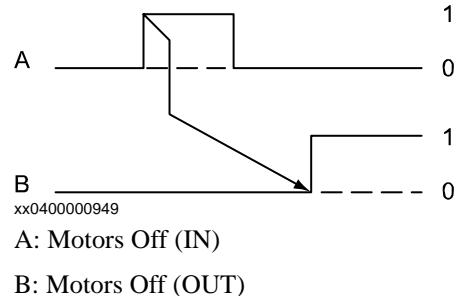
Prerequisites

The following prerequisites have to be considered:

- A digital signal input with a defined signal name has to be configured in the system.
- It is advisable to stop the program execution before using the action *Motors Off* to secure a controlled stop.

Additional information

The signal sequence for *Motors Off* is:



Related information

Operating manual - IRC5 with FlexPendant, chapter Handling inputs and outputs, I/O.

4 Topic I/O

4.9.4.4. Load and Start

4.9.4.4. Load and Start

Parent

Load and Start is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O*.

Cfg name

LoadStart

Description

The action value *Load and Start* loads a RAPID program (files of type .mod, .prg, and .pgf) from a mass storage device. The program starts from the beginning.

Note! The previously loaded files (of type .prg or .pgf) will be unloaded.

Load and Start can be used by a PLC to load and start a program, instead of using the FlexPendant.

Arguments

When the parameter *Action* is set to *Load and Start*, the parameters *Argument1* and *Argument2* must also be used.

Parameter:	Allowed value:
Argument1	The name of the program file to load, including the file format (.mod, .prg or .pgf). Always define the path to the file, e.g. HOME:ModuleA.mod
Argument2	The task in which the routine defined in <i>Argument 1</i> should execute.

Prerequisites

The following prerequisites have to be considered:

- A digital signal input with a defined signal name has to be configured in the system.
- The controller has to be in Motors On state and the program control has to be available, i.e. not used by any other resource.
- The parameter *Argument 1* has to be defined with an existing program file name.
- If the option *MultiMove* is installed, the parameter *Argument 2* must be defined with a task for which the program or module should be loaded.

Limitations

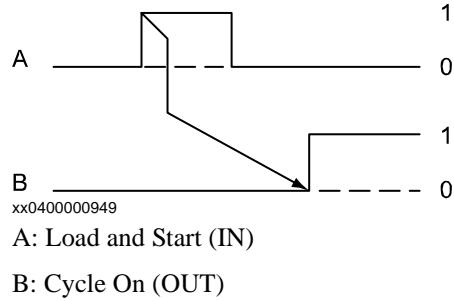
This action value has the following limitations:

- The controller has to be in automatic mode.
- You cannot use this action if the *Stop*, *Stop at end of Cycle*, or *Stop at end of Instruction* actions are set.
- *Load and Start* action is not valid during program execution.
- The run mode will always be set to Cyclic.
- If the controller is in Motors Off state, only the load is performed.
- If the current program has been changed, the changes will not be saved before the load.

Continued

Additional information

The signal sequence for *Load Start* is:



A: Load and Start (IN)

B: Cycle On (OUT)

Related information

[Action on page 202.](#)

[Argument 1 on page 220.](#)

[Argument 2 on page 221.](#)

4.9.4.5. Interrupt

Parent

Interrupt is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O*.

Cfg name

Interrupt

Description

The action value *Interrupt* executes a routine and after running the routine the execution will resume to the same instruction as before. A regain movement is always performed before the interrupt routine executes.

Interrupt can be used by a PLC to let the robot go to a service position.

Arguments

When the parameter *Action* is set to *Interrupt*, the parameters *Argument1* and *Argument2* must also be used.

Parameter:	Allowed value:
Argument1	The name of the routine to be executed.
Argument2	The task in which the module or program defined in <i>Argument 1</i> should be loaded.

Prerequisites

The following prerequisites have to be considered:

- A digital signal input with a defined signal name has to be configured in the system.
- The program execution has to be stopped.
- The parameter *Argument 1* has to be defined with the name of the routine to be executed, e.g. routine 1.
- If the option *MultiMove* is installed, the parameter *Argument 2* has to be defined with a task in which the routine should execute.

Limitations

The parameter has the following limitations:

- The system has to be in automatic mode.
- You cannot use this action value if the *Stop*, *Stop at end of Cycle*, or *Stop at end of Instruction* actions are set.
- The *Interrupt* action is not valid during program execution.

Continued

Additional information

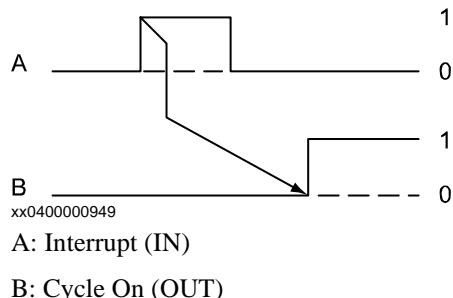
When the execution is stopped, the robot still remembers the point to which it is supposed to go. To prevent the robot going to this position when the Interrupt routine starts and delay it until after the Interrupt, the following RAPID sequence can be used in the Interrupt routine:

```
PROC A()
  StopMove\Quick;!Prevent current move instruction to continue
  StorePath;!For later use
  currpos:=CRobT();!Save current position
  -----
  -----! Place the code for the routine to run here.
  -----
  MoveJ currpos,v600,fine,toolx;!Move back to programmed position
  RestoPath;!Restore StorePath
  StartMove;!Restore StopMove
ENDPROC
```

After the `StartMove` instruction, the stopped movement will continue to move to its fine point. When the routine A has been executed, the normal program can be restarted.

Signal sequence

The signal sequence for *Interrupt* is:



Related information

[Action on page 202.](#)

[Argument 1 on page 220.](#)

[Argument 2 on page 221.](#)

4 Topic I/O

4.9.4.6. Start

4.9.4.6. Start

Parent

Start is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O*.

Cfg name

Start

Description

The action value *Start* starts a RAPID program from the current instruction, continuous or cycle run mode.

Start can be used by a PLC to start the program execution.

Arguments

When the parameter *Action* is set to *Start*, the parameter *Argument1* must also be used, specifying *Continuous* or *Cycle*. The default value is *Continuous*.

Prerequisites

The following prerequisites have to be considered:

- A digital signal input with a defined signal name has to be configured in the system.
- The controller has to be in Motors On state and the program control has to be available, i.e. not used by any other resource.
- The parameter *Argument1* has to be defined with the running mode *Continuous* or *Cycle*.

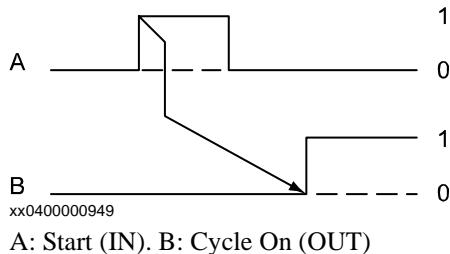
Limitations

This action value has the following limitations:

- The controller has to be in automatic mode.
- You cannot use this action if the *Stop*, *Stop at end of Cycle*, or *Stop at end of Instruction* actions are set.

Additional information

The signal sequence for *Start* is:



Related information

[Argument 1 on page 220](#).

[Action on page 202](#).

4.9.4.7. Start at Main

Parent

Start at Main is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O*.

Cfg name

StartMain

Description

The action value *Start at Main* starts a RAPID program from the beginning, continuous or cycle run.

Start at Main can be used by a PLC to start the program execution from the beginning.

Arguments

When the parameter *Action* is set to *Start at Main*, the parameter *Argument1* must also be used, specifying *Continuous* or *Cycle*. The default value is *Continuous*.

Prerequisites

The following prerequisites have to be considered:

- A digital signal input with a defined signal name has to be configured in the system.
- The controller has to be in Motors On state and the program control has to be available, i.e. not used by any other resource.
- The parameter *Argument 1* has to be defined with the running mode Continuous or Cycle.

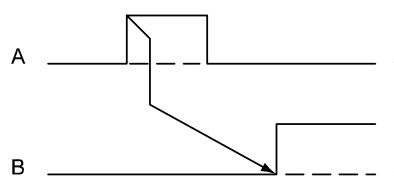
Limitations

This action value has the following limitations:

- The controller has to be in automatic mode.
- You cannot use this action if the *Stop*, *Stop at end of Cycle*, or *Stop at end of Instruction* actions are set.
- *Start at Main* action is not valid during program execution.

Additional information

The signal sequence for *Start at Main* is:



A: Start at Main (IN). B: Cycle On (OUT)

Related information

[Argument 1 on page 220](#).

[Action on page 202](#).

4.9.4.8. Stop

4.9.4.8. Stop

Parent

Stop is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O*.

Cfg name

Stop

Description

The action value *Stop* stops the RAPID program execution. A program cannot be started when this signal is high.

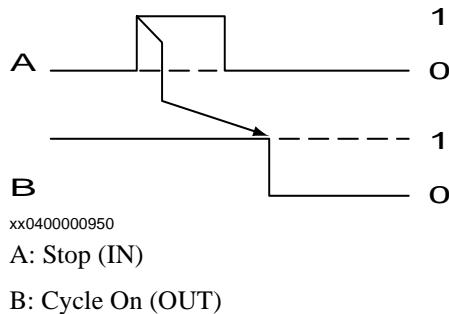
Stop can be used by a PLC to stop the program execution.

Prerequisites

A digital signal input with a defined signal name has to be configured in the system.

Additional information

The signal sequence for *Stop* is:



4.9.4.9. Quick Stop

Parent

Quick Stop is an action value for the parameter *Action* that belongs to the type *System Input*, in the topic *I/O*.

Cfg name

QuickStop

Description

The action value *Quick Stop* stops the RAPID program execution quickly, much like an emergency stop. The robot may slide off path.

Note! Emergency stops should not be used for normal program stops as this causes extra, unnecessary wear on the robot. *Quick Stop* can preferably be used for safety equipment, such as gates.

Prerequisites

A digital signal input with a defined signal name has to be configured in the system.

4.9.4.10. Soft Stop

4.9.4.10. Soft Stop

Parent

Soft Stop is an action value for the parameter *Action* that belongs to the type *System Input*, in the topic *I/O*.

Cfg name

SoftStop

Description

The action value *Soft Stop* stops the RAPID program execution. *Soft Stop* is much like an ordinary program stop, but slightly faster. The robot may slide off path.

Prerequisites

A digital signal input with a defined signal name has to be configured in the system.

4.9.4.11. Stop at End of Cycle

Parent

Stop at End of Cycle is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O*.

Cfg name

StopCycle

Description

The action value *Stop at End of Cycle* stops the RAPID program when the complete program is executed, i.e. when the last instruction in the main routine has been completed. A program cannot be started when this signal is high.

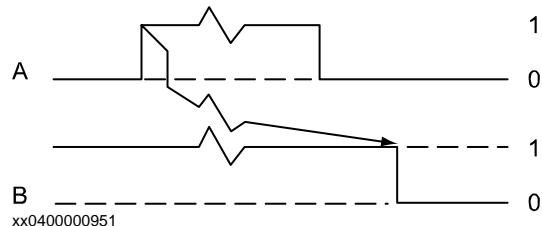
Stop at End of Cycle can be used by a PLC to stop the program execution when the complete program has been executed.

Prerequisites

A digital signal input with a defined signal name has to be configured in the system.

Additional information

The signal sequence for *Stop at End of Cycle* is:



A: Stop at end of Cycle (IN)

B: Cycle On (OUT)

4.9.4.12. Stop at End of Instruction

Parent

Stop at End of Instruction is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O*.

Cfg name

StopInstr

Description

The action value *Stop at End of Instruction* stops program execution after the current instruction is completed. A program cannot start when this signal is high.

Stop at end of Instruction can be used by a PLC to stop the program execution when the current instruction is completed.

Prerequisites

A digital signal input with a defined signal name has to be configured in the system.

Additional information

If using *Stop at End of Instruction* in combination with an instruction that is waiting for a signal or an instruction, e.g. *WaitSyncTask*, *WaitDI*, or *SyncMoveOn*, then the waiting instruction may not be finished. The recommendation is to use system input *Stop* together with *Stop at End of Instruction* to avoid the program from hanging.

Related information

[Stop on page 212.](#)

Example

If a *WaitTime* instruction is executed, it may take a while before the execution is stopped.

4.9.4.13. Reset Execution Error Signal

Parent

Reset Execution Error Signal is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O*.

Cfg name

ResetError

Description

The action value *Reset Execution Error Signal* resets the system output signal action *Execution Error*.

This action can be used by a PLC to reset the error signal.

Prerequisite

A digital signal input with a defined signal name has to be configured in the system.

Related information

Execution Error on page 229.

4 Topic I/O

4.9.4.14. Reset Emergency Stop

4.9.4.14. Reset Emergency Stop

Parent

Reset Emergency Stop is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O*.

Cfg name

ResetEStop

Description

The action value *Reset Emergency Stop* confirms the reset of an emergency stop. When an emergency stop has occurred, it must first be restored mechanically and the reset has to be confirmed. The controller can then be set to the Motors On state.

It is possible to use a PLC to confirm the reset of the emergency stop instead of using the Motors On button.

Prerequisites

The following prerequisites have to be considered:

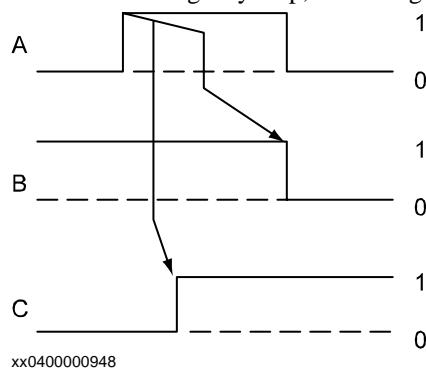
- A digital signal input with a defined signal name has to be configured in the system.
- The safety chain must be closed by restoring the emergency stop mechanically.

Limitations

The controller has to be in automatic mode.

Additional information

To reset an emergency stop, set the signal sequences according to the image.



A: Reset Emergency Stop (IN), Order

B: Emergency Stop (OUT), Response

C: Run Chain OK (OUT), Response

4.9.4.15. System Restart

Parent

System Restart is an action value for the parameter *Action* that belongs to the type *System Input* in the topic *I/O*.

Cfg name

SysReset

Description

The action value *System Restart* performs a controller restart, similar to power off/on.

This action can be used by a PLC to restart the controller.

Prerequisites

The following prerequisites have to be considered:

- A digital signal input with a defined signal name has to be configured in the system.
- It is advisable to stop all RAPID programs before using the action.

4.9.5. Argument 1

4.9.5. Argument 1

Parent

Argument 1 belongs to the type *System Inputs* in the topic *I/O*.

Cfg name

Arg1

Description

Argument 1 is an argument required to perform the system actions *Interrupt*, *Load and Start*, *Motors On and Start*, *Start*, or *Start at Main*, i.e. when the parameter *Action* has one of the action values listed above, *Argument 1* must be set too.

Allowed values

The following values are allowed:

System action:	Allowed value:
Interrupt	The name of the routine to be executed.
Load and Start	The name of the program file to load, including the file format (.mod, .prg or .pgf). Always define the path to the file, e.g. HOME:ModuleA.mod
Motors On and Start	Continuous or Cycle (default value is Continuous)
Start	Continuous or Cycle (default value is Continuous)
Start at Main	Continuous or Cycle (default value is Continuous)

Related information

[Action on page 202.](#)

[Interrupt on page 208.](#)

[Load and Start on page 206.](#)

[Motors On and Start on page 204.](#)

[Start on page 210.](#)

[Start at Main on page 211.](#)

[Argument 2 on page 221.](#)

4.9.6. Argument 2

Parent

Argument 2 belongs to the type *System Input*, in the topic *I/O*.

Cfg name

Arg2

Description

Argument 2 is an argument required to perform the system actions *Load and Start* and *Interrupt*, i.e. when the parameter *Action* is set to *Load and Start* or *Interrupt*, *Argument 2* must be set too.

Usage

Argument 2 is used to define a task.

Prerequisites

Action must be set to *Load and Start* or *Interrupt*.

Limitations

Argument 2 is only used with the option *MultiMove*.

Allowed values

System action:	Allowed value:
<i>Load and Start</i>	The task in which the routine defined in <i>Argument 1</i> should execute.
<i>Interrupt</i>	The task in which the module or program defined in <i>Argument 1</i> should be loaded.

If *MultiMove* is not installed, then *Argument 2* must be set to T_ROB1.

Related information

[Action on page 202.](#)

[Load and Start on page 206.](#)

[Interrupt on page 208.](#)

[Argument 1 on page 220.](#)

4 Topic I/O

4.10.1. The System Output type

4.10 Type System Output

4.10.1. The System Output type

Overview

This section describes the type *System Output* which belongs to the topic *I/O*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

SYSSIG_OUT

Type description

System output signals can be assigned for a specific system action. These signals are set automatically by the system without user input when the system action occurs.

System output signals can be both digital and analog.

Prerequisites

A signal must be configured in the system. The signal name must be a string of maximum 32 characters.

Limitations

The following limitations have to be considered:

- Several output signals can be assigned the same system action, but several system actions may not be assigned to the same signal.
- When deleting a system action the signal itself remains defined. The signal must be deleted separately.
- The predefined system output for the Motors On lamp cannot be edited.

Predefined system outputs

Motors On is predefined in the robot system. This output is linked to the Motors On lamp on the controller.

Additional information

The actions are valid for both manual and automatic mode unless stated otherwise in the value descriptions.

Related information

[The Signal type on page 173.](#)

4.10.2. Status

Parent

Status belongs to the type *System Output*, in the topic *I/O*.

Cfg name

Status

Description

Output signals can be assigned specific system actions. *Status* defines the system status that triggered the signal. The system actions are handled by the system without input from a user.

Allowed values

The following action values are allowed and described on the following pages:

- [Auto On on page 225](#).
- [Simulated I/O on page 226](#).
- [Cycle On on page 227](#).
- [Emergency Stop on page 228](#).
- [Execution Error on page 229](#).
- [Motors Off State on page 230](#).
- [Motors On State on page 231](#).
- [Motors Off on page 232](#).
- [Motors On on page 233](#).
- [Motion Supervision On on page 234](#).
- [Motion Supervision Triggered on page 235](#).
- [Power Fail Error on page 236](#).
- [Path Return Region Error on page 237](#).
- [Run Chain OK on page 238](#).
- [TCP Speed on page 239](#).
- [TCP Speed Reference on page 240](#).
- [Mechanical Unit Active on page 241](#).

4 Topic I/O

4.10.3. Signal

4.10.3. Signal

Parent

Signal belongs to the type *System Output*, in the topic *I/O*.

Cfg name

Signal

Description

Signal is the name of the configured digital output signal to use. It connects the system output with a configured digital output signal.

Prerequisites

A digital output signal with a defined name has to be configured in the system.

Allowed values

Available configured digital output signal names.

Related information

[The Signal type on page 173.](#)

4.10.4. Action values

4.10.4.1. Auto On

Parent

Auto On is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

AutoOn

Description

The action value *Auto On* is set when the controller is in automatic mode.

Related information

Operating manual - IRC5 with FlexPendant.

4 Topic I/O

4.10.4.2. Simulated I/O

4.10.4.2. Simulated I/O

Parent

Simulated I/O is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

Blocked I/O

Description

The action value *Simulated I/O* is set when at least one I/O signal at any I/O unit is in simulated mode.

Additional information

I/O signals can be set to simulated mode during testing, using the FlexPendant.

Related information

Operating manual - IRC5 with FlexPendant.

4.10.4.3. Cycle On

Parent

Cycle On is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

CycleOn

Description

The action value *Cycle On* is set when the robot program is executing.

Additional information

Cycle On is also active for Service and Event Routine execution.

4 Topic I/O

4.10.4.4. Emergency Stop

4.10.4.4. Emergency Stop

Parent

Emergency Stop is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

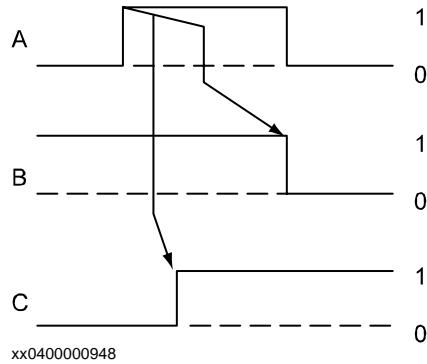
EmStop

Description

The action value *Emergency Stop* is set when the controller is in the Emergency Stop state.

Additional information

The signal sequence for *Emergency Stop* is:



A: Reset Emergency Stop (IN), Order

B: Emergency Stop (OUT), Response

C: Run Chain OK (OUT), Response

4.10.4.5. Execution Error

Parent

Execution Error is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

Error

Description

The action value *Execution Error* is set when the robot program execution has been stopped due to a program error.

Related information

Reset Execution Error Signal on page 217.

4 Topic I/O

4.10.4.6. Motors Off State

Parent

Motors Off State is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

MotOffState

Description

The action value *Motors Off State* is set when the controller is in the Motors Off state.

4.10.4.7. Motors On State

Parent

Motors On State is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

MotOnState

Description

The action value *Motors On State* is set when the controller is in the Motors On state.

4.10.4.8. Motors Off

4.10.4.8. Motors Off

Parent

Motors Off is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

MotorOff

Description

The action value *Motors Off* is set when the controller is in the Motors Off state.

Additional information

When the controller is in Motors Off state and a safety chain is not closed, the output signal pulses.

If only Motors Off state is requested, the action value *Motors Off State* is preferred.

Related information

[Motors Off State on page 230](#).

[Run Chain OK on page 238](#).

4.10.4.9. Motors On

Parent

Motors On is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

MotorOn

Description

The action value *Motors On* is set when the controller is in the Motors On state.

Additional information

If the controller is in guard stop, the output starts pulsing with a frequency of 1 sec. If the controller is not calibrated or the revolution counter is not updated, the output will pulsate even faster in manual mode.

Motors On can be used to detect if the controller is in Motors On and whether the controller is synchronized or not.

Related information

Motors On State on page 231.

4 Topic I/O

4.10.4.10. Motion Supervision On

4.10.4.10. Motion Supervision On

Parent

Motion Supervision On is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

MotSupOn

Description

The action value *Motion Supervision On* is set when the collision detection function is active.

Prerequisites

When the parameter *Action* is set to *Motion Supervision On*, the parameter *Argument* must also be used, specifying which robot the supervision is used for. The default value is ROB_1.

The option *Collision Detection* must be installed.

Additional information

Motion Supervision On is only valid when the robot is moving.

Related information

Application manual - Motion coordination and supervision.

4.10.4.11. Motion Supervision Triggered

Parent

Motion Supervision Triggered is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

MotSupTrigg

Description

The action value *Motion Supervision Triggered* is set when the collision detection function has been triggered.

Prerequisites

The option *Collision Detection* must be installed.

Additional information

The value *Motion Supervision Triggered* is cleared when the program is restarted.

Related information

Application manual - Motion coordination and supervision.

4 Topic I/O

4.10.4.12. Power Fail Error

Parent

Power Fail Error is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

PSError

Description

The action value *Power Fail Error* is set when a program cannot continue from its current position after a power failure.

Additional information

The program will not restart after the value *Power Fail Error* is set. Usually, the program can be started, but it will always start from the beginning.

4.10.4.13. Path Return Region Error

Parent

Path Return Region Error is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

RegainDistError

Description

The action value *Path Return Region Error* is set when an attempt to start the robot program has been made but failed since the robot was too far from the programmed path.

Additional information

The value *Path Return Region Error* is set if the current movement is interrupted and then:

- The robot is jogged too far from the programmed path and then restarted.
- An emergency stop has occurred and the robot has slid too far away from its programmed path and then restarted.

The signal is reset by one of the following actions:

- The robot is jogged into the regain zone, then a restart of the program will succeed.
- The Program Pointer is manually moved to Main.
- The Program Pointer is manually moved and the program is restarted.

The distances of the zones can be configured in the type *Return Region* in the topic *Controller*.

Related information

[The Path Return Region type on page 96](#), in the topic *Controller*.

4 Topic I/O

4.10.4.14. Run Chain OK

Parent

Run Chain OK is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

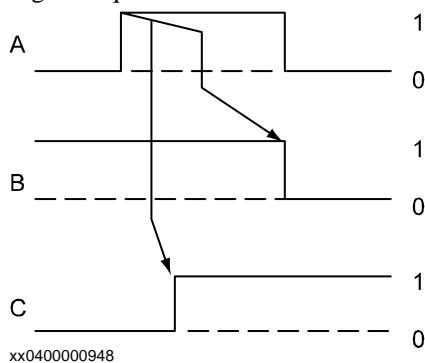
RunchOk

Description

The action value *Run Chain OK* is set when the safety chain is closed. The safety chain must be closed to be able to go to Motors On.

Additional information

Signal sequence:



A: Reset Emergency Stop (IN), Order

B: Emergency Stop (OUT), Response

C: Run Chain OK (OUT), Response

Example

In Manual mode the safety chain is opened and *Run Chain OK* is not set.

4.10.4.15. TCP Speed

Parent

TCP Speed is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

TCPSpeed

Description

The action value *TCP Speed* is an analog signal that reflects the speed of the robot's TCP.

Prerequisites

When the parameter *Action* is set to *TCP Speed*, the parameter *Argument* must also be used, specifying which robot the speed refers to. The default value is ROB_1.

Additional information

The logical value of the signal is specified in m/s, e.g. a speed of 2000 mm/s corresponds to the logical value 2 m/s. The scaling factor for the physical value is specified in the parameters of the corresponding signal.

The analog output is set approximately 60 ms before the actual TCP speed occurs. This prediction time is constant during acceleration and deceleration.

Related information

[Maximum Logical Value on page 190](#).

[Maximum Physical Value on page 192](#).

4.10.4.16. TCP Speed Reference

Parent

TCP Speed Reference is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

TCPSpeedRef

Description

The action value *TCP Speed Reference* is an analog signal describing the programmed speed of the robot's TCP.

Prerequisites

When the parameter *Action* is set to *TCP Speed Reference*, the parameter *Argument* must also be used, specifying which robot the programmed speed refers to. The default value is ROB_1.

Additional information

TCP Speed Reference works in the same way as *TCP Speed* but uses the programmed speed.

Note: *TCP Speed* can differ from *TCP Speed Reference*, e.g. at acceleration or if the override speed has been changed.

Related information

[TCP Speed on page 239](#).

4.10.4.17. Mechanical Unit Active

Parent

Mechanical Unit Active is an action value for the parameter *Status* that belongs to the type *System Output* in the topic *I/O*.

Cfg name

MechUnit Active

Description

The action value *Mechanical Unit Active* is a signal that reflects whether the configured mechanical unit is active.

Arguments

When the parameter *Action* is set to *Mechanical Unit Active*, the parameter *Argument* must also be used, specifying which mechanical unit the signal is reflecting. The default value is ROB_1.

Additional information

If the configured mechanical unit is active, the system output will be set.

If the mechanical unit is configured to be active, the system output will already be set at startup.

It is possible to deactivate a mechanical unit in the jogging window on the FlexPendant or via RAPID.

Related information

[Argument on page 242.](#)

4.10.5. Argument

Parent

Argument belongs to the type *System Outputs*, in the topic *I/O*.

Cfg name

Arg1

Description

Argument is an argument required to perform the system actions *TCP Speed*, *TCP Speed Reference*, or *Motion Supervision On*, i.e. when the parameter *Action* has one of the action values listed above, *Argument* must be set too.

Allowed values

If the action values *TCP Speed*, *TCP Speed Reference*, or *Motion Supervision On* are used, the allowed value is a robot from the type *Robot* in the topic *Motion*. Default value is ROB_1.

If the action value is *Mechanical Unit Active*, the allowed value is a mechanical unit of the type *Mechanical Unit Group* in the topic *Motion*. Default value is ROB_1.

Related information

Action value *TCP Speed* on page 239.

Action value *TCP Speed Reference* on page 240.

Action value *Motion Supervision On* on page 234.

Action value *Mechanical Unit Active* on page 241.

The Robot type on page 516 in the topic *Motion*.

The Mechanical Unit Group type on page 79.

4.11 Type Unit

4.11.1. The Unit type

Overview

This section describes the type *Unit*, which belongs to the topic *I/O*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

EIO_UNIT

Type description

An I/O unit is a logical software representation of a fieldbus unit that is connected to a fieldbus within the controller. I/O units allow you to control electronic devices and read sensor data. They are used for controlling signals in the robot system.

Usage

By specifying a unit, a logical representation of the real I/O unit is created. The unit configuration defines the specific parameters that will control the behavior of the unit.

The *Unit* is used when defining the signals and other objects in the I/O system.

Prerequisites

Before defining a new unit:

1. Configure the *Bus* and
2. Make sure that the appropriate *Unit Type* is available (either by creating it or using a predefined unit type).

Limitations

The unit has the following limitations:

- Maximum number of units in the system is 40 (not including units on the bus Local)
- Maximum number of units on one bus is 20
- The type of unit and the bus referred to by parameters *Type of Unit* and *Connected to Bus* must not be in conflict with each other, i.e. the bus types of the referenced unit type and the referenced bus must be identical unless the bus type of the bus is Virtual.

Predefined units

The following units are predefined and located on the safety related signals:

- PANEL
- DRV_1
- DRV_2
- DRV_3
- DRV_4

Depending on installed options, there can be other predefined units not described in this manual.

Continues on next page

4 Topic I/O

4.11.1. The Unit type

Continued

Related information

[Type of Unit on page 246.](#)

[Connected to Bus on page 247.](#)

[The Bus type on page 138.](#)

[The Unit Type type on page 255.](#)

For more information on safety signals, please see *Operating manual - IRC5 with FlexPendant*.

Example

Parameter:	Value:
Name	board10
Type of Unit	d327
Connected to Bus	MyDeviceNet
Unit Identification Level	U137, placed in process cabinet C5
Unit Trust Level	1 - Error when lost
Unit Startup State	Enabled
Store Unit State at Power Fail	No
DeviceNet Address	10

4.11.2. Name

Parent

Name belongs to the type *Unit*, in the topic *I/O*.

Cfg name

Name

Description

The parameter *Name* specifies the name of the unit.

Usage

The name of the unit is used as a reference to the specific unit when configuring the signals and fieldbus commands on the unit.

Allowed values

A string following the RAPID rules, as described in *Technical reference manual - RAPID overview*, chapter *Basic elements*.

The name must be unique among all named objects in the I/O system configuration. Note that names differing only in upper and lower case are considered to be equal.

4.11.3. Type of Unit

4.11.3. Type of Unit

Parent

Type of Unit belongs to the type *Unit*, in the topic *I/O*.

Cfg name

UnitType

Description

The parameter *Type of Unit* specifies the type of the unit.

Usage

Instead of specifying all properties of a unit within the unit itself, a reference to a specific fieldbus unit type is made. The referred unit type contains all characteristics that are common for all units of that type.

Limitations

The bus type of the unit type referred to by this parameter must not be in conflict with the bus referred to by the parameter *Connected to Bus*.

Allowed values

The parameter must correspond to the name of a defined *Unit Type*.

Note that names differing only in upper and lower case are considered to be equal.

Related information

[Connected to Bus on page 247](#).

[The Unit Type type on page 255](#).

4.11.4. Connected to Bus

Parent

Connected to Bus belongs to the type *Unit*, in the topic *I/O*.

Cfg name

Bus

Description

The parameter *Connected to Bus* specifies which bus this unit is physically connected to.

Limitations

The bus type of the bus referred to by this parameter must not be in conflict with the bus referred to by the parameter *Type of Unit*.

Allowed values

The parameter must correspond to the name of a defined bus.

Note that names differing only in upper and lower case are considered to be equal.

Related information

[Type of Unit on page 246.](#)

[The Bus type on page 138.](#)

4.11.5. Unit Identification Label

4.11.5. Unit Identification Label

Parent

Unit Identification Label belongs to the type *Unit*, in the topic *I/O*.

Cfg name

UnitLabel

Description

The parameter *Unit Identification Label* provides a way to label the actual unit.

Usage

The parameter *Unit Identification Label* is an optional way to provide a label that will help the operator to identify the unit physically.

Allowed values

A string of maximum 80 characters.

4.11.6. Unit Trustlevel

Parent

Unit Trustlevel belongs to the type *Unit*, in the topic *I/O*.

Cfg name

TrustLevel

Description

The parameter *Unit Trustlevel* specifies the desired system behavior (i.e. error handling and actions taken) in case communication with the unit is lost or re-established.

Usage

A state change message is logged every time the unit is connected (i.e. communication with the unit is reestablished) unless the *Unit Trustlevel* is set to 2 - Loss accepted.

Allowed values

The value specifies system behavior when the unit is connected.

Value:	Description:
0 - Required	Error logged and system transferred to system failure mode.
1 - Error when lost	Error logged.
2 - Loss accepted	No action.
3 - Stop when lost	Error logged and program stopped.

4 Topic I/O

4.11.7. Unit Startup State

Parent

Unit Startup State belongs to the type *Unit*, in the topic *I/O*.

Cfg name

Disabled

Description

The parameter *Unit Startup State* specifies whether or not the I/O system should try to contact this unit at startup.

Limitations

Unit Startup State has lower priority than *Store Unit State at Power Fail*.

Usage

If *Unit Startup State* for this unit is set to Enabled, the system tries to establish contact at startup. A fault report is generated if no connection is established.

If the robot is used in a tool changing application, it is recommended that *Unit Startup State* is set to Disabled and RAPID instructions (`IOenable` and `IOdisable`) are used to check connections and enable connected units at start-up.

Allowed values

Enabled or Disabled

Related information

[Store Unit State at Power Fail on page 251](#).

[RAPID reference manual - Instructions](#).

4.11.8. Store Unit State at Power Fail

Parent

Store Unit State at Power Fail belongs to the type *Unit*, in the topic *I/O*.

Cfg name

StoreLogicalState

Description

The parameter *Store Unit State at Power Fail* specifies whether this unit should assume the disable or enable state it had before power down.

Usage

If *Store Unit State at Power Fail* is enabled (Yes), the unit assumes the state it had before power down.

If the robot is used in a tool changing application, it is recommended that *Store Unit State at Power Fail* is set to No and RAPID instructions (`IOenable` and `IOdisable`) is used to check the state at startup.

Additional information

If *Store Unit State at Power Fail* is enabled, it has a higher priority than *Unit Startup State*.

Allowed values

Yes or No.

Related information

[Unit Startup State on page 250](#).

[Technical reference manual - RAPID Instructions, Functions and Data types](#).

4.11.9. Regain Communication Reset

Parent

Regain Communication Reset belongs to the type *Unit*, in the topic *I/O*.

Cfg name

RegainCommunicationReset

Description

Regain Communication Reset defines if this unit should reset the signal values to the default values when it regains communication (e.g. when a unit goes from unconnected to running state).

Usage

If both *Regain Communication Reset* and *Store Signal Value at Power Fail* are used and the system powers down, then *Store Signal Value at Power Fail* has higher priority when the system starts up.

Allowed values

Enabled or Disabled.

Related information

[Default Value on page 184](#), in the type *Signal*.

[Store Signal Value at Power Fail on page 185](#), in the type *Signal*.

4.11.10. Profibus Address

Parent

Profibus Address belongs to the type *Unit*, in the topic *I/O*.

Cfg name

PB_Address

Description

The parameter *Profibus Address* specifies the address of this I/O unit on the network.

Usage

This unit specifies the address that the I/O unit uses on the network, to which the master should try to setup a connection.

Profibus Address is a Profibus specific parameter that is only available for Profibus units.

Prerequisites

A Profibus fieldbus option must be installed in the system.

Allowed values

0 - 125

Related information

Application manual - ProfiBus.

4 Topic I/O

4.11.11. Interbus Address

4.11.11. Interbus Address

Parent

The parameter *Interbus Address* belongs to the type *Unit*, in the topic *I/O*.

Cfg name

IB_Address

Description

The parameter *Interbus Address* specifies the address of the I/O unit on the network.

Usage

Interbus Address specifies the address that the I/O unit is assumed to use on the network and which the master should try to setup a connection against.

Prerequisites

InterBus option must be installed.

Allowed values

The value can be:

- 0.240 (used for internal slave)
- 1.0 - 255.63

There are limitations for the values set by the vendor.

Related information

Application manual - InterBus.

4.12 Type Unit Type

4.12.1. The Unit Type type

Overview

This section describes the type *Unit Type*, which belongs to the topic *I/O*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

EIO_UNIT_TYPE

Type description

An I/O *Unit Type* is a software description of a specific fieldbus unit type that can be connected to a fieldbus within the controller.

Limitations

The following limitations must be considered:

- Maximum number of unit types in the system is 50.
- The maximum size of a *Unit Type* that can be handled by the controller is 64 bytes of input data and 64 bytes of output data.

Usage

The *Unit Type* can be seen as an identification and description of the device that shall be connected to the fieldbus.

The *Unit Type* is used when detecting *Units* and *Fieldbus Command Types* in the I/O system.

Predefined unit types

The following unit types are predefined:

- Virtual
- LOCAL_GENERIC

Related information

[How to define the I/O unit type on page 126.](#)

Example

This is an example of a unit type with a DeviceNet bus from ABB Robotics.

Parameter:	Value:
Name	d327
Type of Bus	DeviceNet
Vendor Name	ABB Robotics
Product Name	Combi Unit
Vendor ID	75
Product Code	2
Device Type	100

Continues on next page

4 Topic I/O

4.12.1. The Unit Type type

Continued

Parameter:	Value:
<i>Major Revision</i>	5
<i>Minor Revision</i>	0
<i>Production Inhibit Time</i>	10
<i>Explicit Messaging</i>	Enabled
<i>Quick Connect</i>	Enabled
<i>Connection 1 Type</i>	Polled connection
<i>Connection 1 Interval</i>	50
<i>Connection 1 Output Size</i>	6
<i>Connection 1 Input Size</i>	2

4.12.2. Name

Parent

Name belongs to the type *Unit Type*, in the topic *I/O*.

Cfg name

Name

Description

The parameter *Name* specifies the name of the unit type.

Usage

The name of the unit type is used as a reference to the specific unit type when:

- Configuring units
- Configuring cross connections

Allowed values

A string following the RAPID rules described in *Technical reference manual - RAPID overview*, chapter *Basic elements*.

The name must be unique among all named objects in the I/O system configuration. Note that names differing only in upper and lower case are considered to be equal.

4 Topic I/O

4.12.3. Type of Bus

Parent

Type of Bus belongs to the type *Unit Type*, in the topic *I/O*.

Cfg name

BusType

Description

The parameter *Type of Bus* specifies the type of fieldbus this logical type of unit is representing.

Usage

Defining the type of bus is vital for identifying which fieldbus this unit type is representing.

Prerequisites

The fieldbus option for the desired type of bus must be installed in the controller.

Allowed values

The values that the parameter *Type of Bus* can assume are determined by the available and installed fieldbus options.

- DeviceNet
- Profibus-DP
- Virtual
- Interbus

4.12.4. Vendor Name

Parent

Vendor Name belongs to the type *Unit Type*, in the topic *I/O*.

Cfg name

VendorName

Description

The parameter *Vendor Name* specifies the name of the unit vendor.

Usage

This parameter is optional and only used as information.

Allowed values

A string with maximum 80 characters.

4 Topic I/O

4.12.5. Product Name

4.12.5. Product Name

Parent

Product Name belongs to the type *Unit Type*, in the topic *I/O*.

Cfg name

ProductName

Description

The parameter *Product Name* specifies the name of the product for this unit type.

Usage

This parameter is optional and only used as information.

Allowed values

A string with maximum 80 characters.

4.12.6. Product ID

Parent

Product ID belongs to the type *Unit Type*, in the topic *I/O*.

Cfg name

PB_ProductId

Description

Product ID is used as an identification of the I/O unit.

Usage

Product ID is a Profibus specific parameter that is only available for Profibus unit types.

If *Product ID* is assigned the value 0 (zero), the robot controller ignores what product ID the connected unit has.

Prerequisites

Profibus-DP m/s option must be installed.

Allowed values

Allowed values are integers 0-65535.

The value of *Product ID* can be found in the Electronic Data Sheet (GSD) for the unit (called *Ident_number* in GSD file). Note that *Ident_number* is often expressed as a hexadecimal number in the GSD file.

Additional information

The I/O unit Product ID is assigned by PNO (Profibus Nützer Org.) to the vendor of the specific I/O unit.

Related information

Application manual - ProfiBus.

4 Topic I/O

4.12.7. Internal Slave

4.12.7. Internal Slave

Parent

Internal Slave belongs to the type *Unit Type*, in the topic *I/O*.

Cfg name

PB_InternalSlave

Description

Internal Slave specifies the type of the slave device.

Usage

A slave device is either internal or external. The DSQC 637 unit has a built-in slave module and it is defined as internal slave. All other slave devices are defined as external slaves.

Internal Slave is a PROFIBUS specific parameter that is only available for PROFIBUS unit types.

Prerequisites

PROFIBUS DP Master/Slave option must be installed.

Allowed values

Yes or No.

Related information

Application manual - ProfiBus.

4.12.8. Internal Slave Input Size

Parent

Internal Slave Input Size belongs to the type *Unit Type*, in the topic *I/O*.

Cfg name

PB_InputSize

Description

Internal Slave Input Size defines the data size received by the unit.

Usage

Internal Slave Input Size is a PROFIBUS specific parameter that is only available for internal slave unit types.

Prerequisites

PROFIBUS-DP Master/Slave option must be installed.

Allowed values

A value between 0-128, specifying the data size in bytes.

Related information

Application manual - ProfiBus.

4 Topic I/O

4.12.9. Internal Slave Output Size

Parent

Internal Slave Output Size belongs to the type *Unit Type*, in the topic *I/O*.

Cfg name

PB_OutputSize

Description

Internal Slave Output Size defines the data size transmitted by the unit.

Usage

Internal Slave Output Size is a PROFIBUS specific parameter that is only available for internal slave unit types.

Prerequisites

PROFIBUS-DP Master/Slave option must be installed.

Allowed values

A value between 0-128, specifying the data size in bytes.

Related information

Application manual - ProfiBus.

5 Topic Man-machine Communication

5.1. The Man-machine Communication topic

Overview

This chapter describes the types and parameters of the *Man-machine Communication* topic.

Description

The *Man-machine Communication* topic contains parameters for creating customized lists for instructions and I/O signals, simplifying everyday work.

The parameters are organized in the following types:

1. Most Common I/O Signal
2. Production permission
3. Most Common Instruction - List 1
4. Most Common Instruction - List 2
5. Most Common Instruction - List 3

The types for Most Common Instructions are identical and therefore only described in one section, but valid for all three types.

5 Topic Man-machine Communication

5.2.1. The Most Common I/O Signal type

5.2 Type Most Common I/O Signal

5.2.1. The Most Common I/O Signal type

Overview

This section describes the type *Most Common I/O Signal* which belongs to the topic *Man-machine Communication*.

Cfg name

IO_MOST_COMMON

Type description

It is possible to have hundreds of I/O signals in the system. To simplify working with them it is possible to group them to a list of the mostly used signals. This list is defined by the type *Most Common I/O Signal*.

Prerequisites

A signal must be configured in the system for the signal name.

Example

This is a typical example of an often used I/O to be included in the list.

Parameter:	Value:
Signal Name	MySignalDI1
Signal Type	DI

5.2.2. Signal Name

Parent

Signal Name belongs to the type *Most Common I/O Signal*, in the topic *Man-machine Communication*.

Cfg name

name

Description

The *Signal Name* is the I/O signal to be part of the Most Common List.

Prerequisites

A signal must be configured in the system.

Allowed values

A signal configured in the system, a name with a maximum of 16 characters.

Related information

The Signal type on page 173.

5 Topic Man-machine Communication

5.2.3. Signal Type

5.2.3. Signal Type

Parent

Signal Type belongs to the type *Most Common I/O Signal*, in the topic *Man-machine Communication*.

Cfg name

type

Description

Signal Type defines the type of signal to be used in the common list.

Allowed values

The following values are allowed.

Value:	Description:
DI	Digital Input
DO	Digital Output
AI	Analog Input
AO	Analog Output
GI	Group Input
GO	Group Output

5.3 Type Production Permission

5.3.1. The Production Permission type

Overview

This section describes the type *Production Permission* which belongs to the topic *Man-Machine Communication*.

Cfg name

PROD_PERMISSION

Type description

Different types of operating restrictions and other features may be connected to specific operating modes. Such connections are specified in the *Production Permission* type.

5 Topic Man-machine Communication

5.3.2. Name

5.3.2. Name

Parent

Name belongs to the type *Production Permission* in the topic *Man-Machine Communication*.

Cfg name

name

Description

The parameter *Name* specifies the name of the permission.

Usage

The name of the permission is used as a reference to a specific permission when configuring the system.

Allowed values

RUN Mode.

5.3.3. Permission

Parent

Permission belongs to the type *Production Permission* in the topic *Man-Machine Communication*.

Cfg name

permission

Description

The parameter *Permission* specifies whether switching to *Cycle_mode* while running in the *Auto mode* should be allowed or not.

While running in the *Auto Mode*, it is normally possible to choose between *Cycle_mode* and *Continuous_mode*. In certain circumstances, this is not desired: always when running in the *Auto Mode*, the *Continuous_mode* must be active.

The parameter type restricts or permits switching to *Cycle_mode* while in the *Auto mode*.

If the name is set to *RUN Mode*, the permission may be set to *Restricted* in *Auto*, and it will not be possible to switch from *Continuous_mode* to *Cycle_mode* while in the *Auto Mode*.

Allowed values

Value	Description
Changeable in Auto	This setting enables the system to be switched to <i>Cycle_mode</i> or <i>Continuous_mode</i> while running in the <i>Auto Mode</i> .
Restricted in Auto	This setting prohibits the system to be switched to <i>Cycle_mode</i> while running in the <i>Auto Mode</i> . Only <i>Continuous_mode</i> is possible.

Default value is *Changeable in Auto*.

5 Topic Man-machine Communication

5.4.1. The Most Common Instruction types

5.4 Type Most Common Instruction

5.4.1. The Most Common Instruction types

Overview

This section describes the types *Most Common Instruction - List 1*, *Most Common Instruction - List 2*, and *Most Common Instruction - List 3* which belongs to topic *Man-machine Communication*.

Cfg names

MMC_MC1
MMC_MC2
MMC_MC3

Type description

The system contains lists of instructions to use when programming the robot. There are also three lists available to adapt to personal requirements. These are called *Most Common Instruction - List 1*, *Most Common Instruction - List 2*, and *Most Common Instruction - List 3*.

The three lists are set up of a number of parameters equal between the lists. Therefore the parameters are described together in this manual.

Required parameters

Only the system parameter *Name* requires a value.

Related information

Instructions and their optional arguments and syntax are described in *Technical reference manual - RAPID Instructions, Functions and Data types*.

Example: Instruction without argument

To create a MoveJ instruction without arguments, only the parameter *Name* is required if *Name* is set to MoveJ, exactly as spelled in RAPID.

Parameter:	Value:
Name	MoveJ
Parameter Number	
Alternative Number	
Instruction Name	
Only for Motion Task	

Example: Instruction with argument

To create a MoveL instruction with the option Time set to the alternative T for motion tasks, use the following values.

Parameter:	Value:
Name	MoveL /T

Continued

Parameter:	Value:
Parameter Number	5
Alternative Number	2
Instruction Name	MoveL
Only for Motion Task	Yes

By setting *Name* to MoveL/T, the button label in the picklist will clearly state to the user that this is a MoveL instruction, using the Time option. The parameter number we use is 5, see table below, and we use alternative 2 for [\T]. Since *Name* is not set to only MoveL, we must use *Instruction Name* to specify to the system that it is a MoveL instruction. *Only for Motion Task* states that it will only be available for motion tasks.

The syntax for the MoveL instruction is:

Parameter Number:	Value:
<instr>	MoveL
1	[\Conc]
2	ToPoint
3	[\ID]
4	Speed
5	[\V] or [\T]
6	Zone
7	[\Z]
8	[\Inpos]
9	Tool
10	[\WObj]
11	[\Corr]

5 Topic Man-machine Communication

5.4.2. Name

5.4.2. Name

Parent

Name belongs to the types *Most Common Instruction - List 1*, *Most Common Instruction - List 2*, and *Most Common Instruction - List 3* in the topic *Man-machine Communication*.

Cfg name

name

Description

Name defines the name to be visible on the button in the picklist.

Usage

If *Name* is set to an instruction or procedure spelled exactly as in RAPID, no other parameters require a value. But, if *Name* contains more information, as recommended when using instructions with arguments, then the parameter *Instruction Name* specifies the actual instruction syntax.

Allowed values

The instruction name, a string with maximum 32 characters, e.g. "MoveJ".

Note! Do not use a backslash (\) in the name! Names using a backslash will cause errors, unlike when programming in RAPID.

If an additional switch or argument is used, it is recommended to include this in the name for clarity and append the name with a slash (/) and the argument, e.g. "ArcL/On". Furthermore if an optional argument is included in the name then the parameter *Instruction Name* must be set to the instruction.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types.

[Instruction Name on page 277](#).

Examples

Value:	Description:
MoveJ	The instruction MoveJ.
ArcL/On	The instruction ArcL with the argument On.

5.4.3. Parameter Number

Parent

Parameter Number belongs to the types *Most Common Instruction - List 1*, *Most Common Instruction - List 2*, and *Most Common Instruction - List 3* in the topic *Man-machine Communication*.

Cfg name

param_nr

Description

Parameter Number specifies which argument should be used for instructions with optional arguments.

Usage

If an instruction with optional arguments is used, then *Parameter Number* specifies which of the arguments should be used. The instructions with parameter numbers are described in *Technical reference manual - RAPID Instructions, Functions and Data types*.

If left blank, no optional argument is used.

Allowed values

A positive integer value, starting from 0.

Additional information

If *Parameter Number* is used, then *Alternative Number* must also be used.

Related information

[Instruction Name on page 277](#).

[Alternative Number on page 276](#).

[Technical reference manual - RAPID Instructions, Functions and Data types](#).

5 Topic Man-machine Communication

5.4.4. Alternative Number

5.4.4. Alternative Number

Parent

Alternative Number belongs to the types *Most Common Instruction - List 1*, *Most Common Instruction - List 2*, and *Most Common Instruction - List 3* in the topic *Man-machine Communication*.

Cfg name

alt_nr

Description

Alternative Number defines which of the optional argument's alternatives to be used for the instruction.

Usage

If the instruction has optional arguments, then *Alternative Number* specifies which of the alternatives to use. The *Parameter Number* specifies which argument to be used.

Prerequisites

The parameter *Parameter Number* must be used.

Allowed values

The following values are allowed (depending on the number of alternatives available for the instruction):

Value:	Description:
0	no alternative is used
1	the first alternative is used
n...	the n th alternative is used

Related information

[Instruction Name on page 277](#).

[Parameter Number on page 275](#).

[Technical reference manual - RAPID Instructions, Functions and Data types](#).

5.4.5. Instruction Name

Parent

Instruction Name belongs to the types *Most Common Instruction - List 1*, *Most Common Instruction - List 2*, and *Most Common Instruction - List 3* in the topic *Man-machine Communication*.

Cfg name

instr_name

Description

Instruction Name defines which instruction to use if the parameter *Name* contains more information than only the instruction.

Usage

If the instruction contains optional arguments, it is recommended to mark this in the parameter *Name*. Then *Instruction Name* is used to specify the instruction, as spelled in RAPID.

Allowed values

The instruction name, a string with maximum 32 characters, as spelled in RAPID.

Related information

[Name on page 274](#).

[Parameter Number on page 275](#).

[Alternative Number on page 276](#).

[Technical reference manual - RAPID Instructions, Functions and Data types](#).

5 Topic Man-machine Communication

5.4.6. Only for Motion Task

5.4.6. Only for Motion Task

Parent

Only for Motion Task belongs to the types *Most Common Instruction - List 1*, *Most Common Instruction - List 2*, and *Most Common Instruction - List 3* in the topic *Man-machine Communication*.

Cfg name

only_mec_task

Description

Only for Motion Task defines if the instruction only should be visible in Motion Tasks, i.e. should control the robot movement, e.g. MoveJ.

Usage

Set *Only for Motion Task* to True if the instruction only should be visible to Motion Tasks.

Allowed values

True or False.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types.

6 Topic Motion

6.1. The Motion topic

Overview

This chapter describes the types and parameters of the *Motion* topic. Each parameter is described in the section for its type.

The topic *Motion* is extensive, with some 40 types. This manual revision covers the most commonly used parameters and types.

Description

Motion contains parameters associated with motion control in the robot and external equipment. The topic includes configuring the calibration offset and the working space limits.

The described parameters are organized in the following types:

1. Acceleration Data
2. Arm
3. Arm Check Point
4. Arm Load
5. Brake
6. Control Parameters
7. Drive Module
8. Drive System
9. Force Master
10. Force Master Control
11. Friction Compensation
12. Jog Parameters
13. Joint
14. Lag Control Master 0
15. Linked M Process
16. Mains
17. Measurement Channel
18. Mechanical Unit
19. Motion Planner
20. Motion Supervision
21. Motion System
22. Motor
23. Motor Calibration
24. Motor Type
25. Path Sensor Synchronization
26. Process

6 Topic Motion

6.1. The Motion topic

Continued

- 27. Relay
- 28. Robot
- 29. Robot Serial Number
- 30. SG Process
- 31. Single
- 32. Single Type
- 33. SIS Parameters
- 34. Stress Duty Cycle
- 35. Supervision
- 36. Supervision Type
- 37. Transmission
- 38. Uncalibrated Control Master 0

Configuration results

Changed motion parameters requires a restart of the controller. Otherwise the changes will not have any effect on the system.

An exception to the rule is the motion supervision parameters which do not require a restart. See the type *Motion Supervision* section for more information.

6.2 Workflows

6.2.1. How to define base frame

The robot and the base frame

Normally, the base frame of the robot coincides with the world frame. However, the base frame can be moved relative to the world frame.

CAUTION!



The programmed positions are always related to the world frame. Therefore, all positions are also moved, as seen from the robot.

How to define the base frame

To define the base frame:

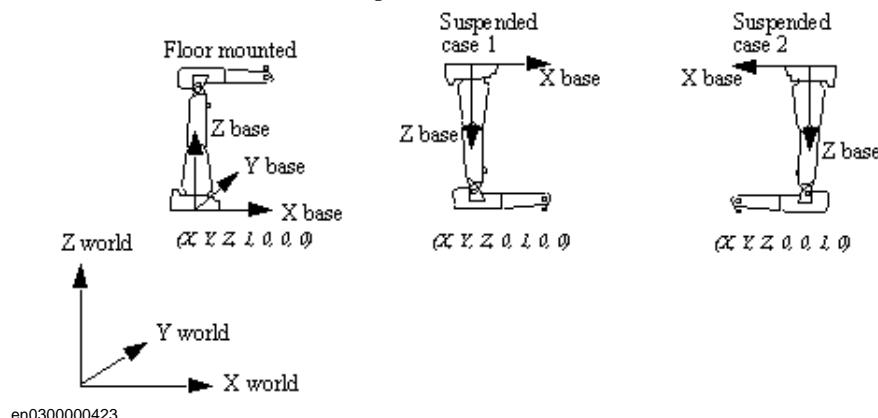
1. In the **Motion** topic, choose the type **Robot**.
2. Select the robot to define the base frame for.
3. Edit the parameters defining the base frame:
 - *Base Frame x*
 - *Base Frame y*
 - *Base Frame z*
 - *Base Frame q1*
 - *Base Frame q2*
 - *Base Frame q3*
 - *Base Frame q4*
 - *Base Frame Moved by*

For detailed information about each parameter, see the descriptions in the *Robot* type section.

4. Save the changes.

Additional information

The illustration shows some examples of frame definitions.



Related information

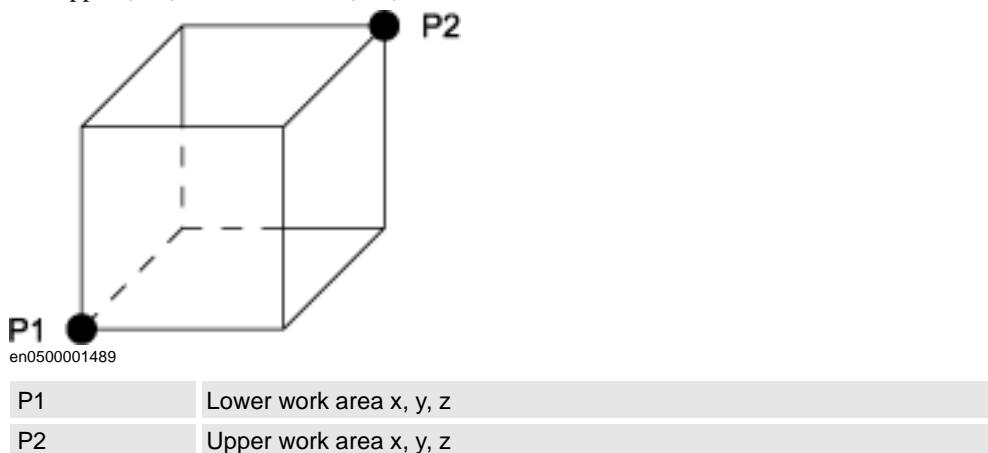
[The Robot type on page 516.](#)

6 Topic Motion

6.2.2. How to define work area

Robot work area

The robot's working area is set by defining a cube (for articulated robots) or by a cylinder (for parallel arm robots) in which the TCP is allowed to move. The cube is defined by six coordinates, three upper (x, y, z) and three lower (x, y, z), and the cylinder by four coordinates, two upper (x, z) and two lower (x, z).



The coordinates are defined in the base coordinate system.

How to define the robot work area

To define the robot work area:

1. In the **Motion** topic, choose the type **Robot**.
2. Edit the parameters *Upper Work Area* and *Lower Work Area* for the coordinates x, y, and z.
See illustration above for descriptions.
3. Save the changes.

Related information

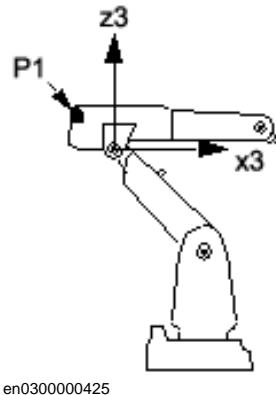
[Upper Work Area x, y, z on page 526.](#)

[Lower Work Area x, y, z on page 527.](#)

6.2.3. How to define arm check point

Arm check point

If an extra load, such as a transformer or a welding-bar roller, is attached to arm 3, a point on this equipment can be defined as a check point. The robot will then monitor the speed of this point so that it does not exceed 250 mm/s in manual reduced speed mode.



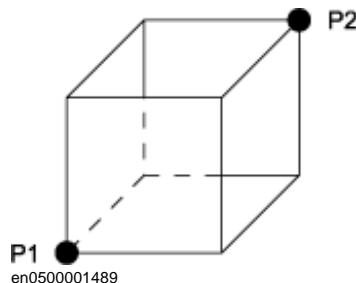
P1	Arm check point
z3	z-axis for arm 3
x3	x-axis for arm 3

Limitations

The value for the *Use Check Point* parameter must be identical to the name used for the arm check point.

Bound check point

The check point can also be restricted to stay within a defined cube, see illustration. The cube is defined by six coordinates, three upper and three lower.



P1	Lower check point bound x, y, z
P2	Upper check point bound x, y, z

How to define arm check point

To define the arm check point:

1. In the **Motion** topic, choose the type **Arm Check Point**.
2. Edit the parameters for the check point.

For detailed information, see the descriptions in the *Arm Check Point type* section.

3. Make a note of the *Name* parameter value to use later.

Continues on next page

6 Topic Motion

6.2.3. How to define arm check point

Continued

4. Save the changes.
5. In the topic **Motion**, choose the type **Arm**.
6. First select arm 3 to connect the check point to the arm. Then edit the parameter *Use Check Point*. The value has to be identical to the name used for the arm check point (step 2-3 above).
For detailed information about the parameters, see sections *Arm type* and *Arm Check Point type*.
7. Save the changes.
8. To restrict the check point, choose the type **Robot** in the topic **Motion**.
9. Edit the parameters *Upper Check Point Bound* and *Lower Check Point Bound* for the six coordinates.
For detailed information about the parameters, see section *Robot type*.
10. Save the changes.

Related information

[The Arm type on page 298](#).

[The Arm Check Point type on page 316](#).

[Upper Check Point Bound x, y, z on page 528](#).

[Lower Check Point Bound x, y, z on page 529](#).

The Product manual for the robot.

6.2.4. How to define arm loads

Arm load

Arm load is used for defining loads from equipment mounted on robot arms. If arm load is not defined when equipment is mounted on robot arms, the performance of the robot is negatively affected.

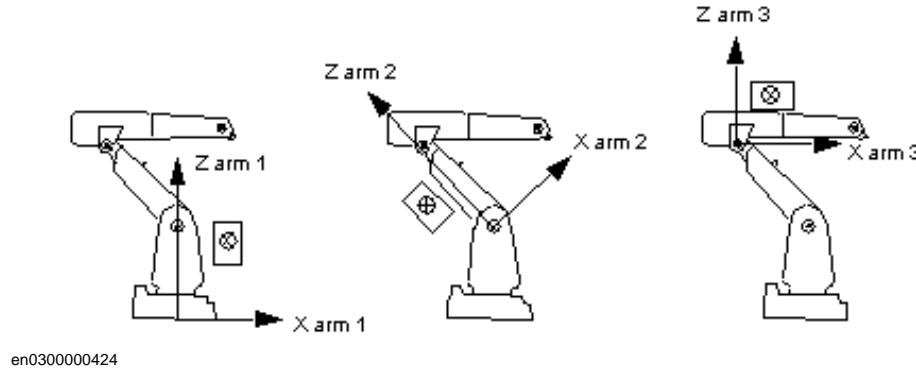
For more information about arm loads, see the type *Arm Load*.

Prerequisites

The mass, the mass center, and the moment of inertia of the load have to be measured or calculated before defining the arm load.

Arms for relating arm load to

Arm loads can be related to robot arms 1, 2 and 3, see the illustration below. All loads mounted on the upper arm are related to arm 3, including loads mounted on the rotating part.



If more than one load is mounted on the same arm, the total weight and center of gravity for the loads have to be calculated.

How to define an arm load

To define an arm load:

1. In the topic **Motion**, choose the type **Arm Load**.
2. Select the arm load to define.
3. Enter or change the parameters of the arm load and save your changes.

For detailed information about each parameter, see the descriptions in the type *Arm Load*.

4. In the topic **Motion**, choose the type **Arm** and select the arm that the load is mounted on.
5. For the selected arm, choose the parameter *Use Customer Arm Load* and specify the name of the arm load.
6. Save your changes.

Related information

[The Arm Load type on page 319.](#)

[The Arm type on page 298.](#)

6.2.5. How to optimize drive system parameters

6.2.5. How to optimize drive system parameters

The drive system parameters

The drive system can be configured so that it corresponds to the robot's installation. The parameters related to the drive system are organized in two types.

To optimize the...	...use the parameters of the type:
tolerance for the mains power supply	<i>Mains</i>
cable type and length	<i>Cable</i>

Default and optimal values

All drive system parameters have nominal values after installation. For improving the robot's performance, these parameters can be adjusted according to the robot's actual installation.

CAUTION!



Parameter settings outside the range of the robot's installation may negatively affect the robot's performance.

How to optimize the mains tolerance

To optimize the tolerance for the mains power supply:

1. In the topic **Motion**, choose the type **Mains**.
2. Edit the *Mains Tolerance Min* parameter according to the robot's installation.
For detailed information about each parameter, see the descriptions in the type *Mains*.
3. Save the changes.

Related information

The Mains type on page 430.

6.2.6. How to tune motion supervision

Motion supervision

Motion supervision is functionality for collision detection with the option *Collision detection*.

How to tune the motion supervision

To tune the motion supervision:

1. In the **Motion** topic, choose the type **Motion Supervision**.
2. Decide which robot to tune the supervision for.
3. Edit the parameters for motion supervision. For detailed information about each parameter, see the descriptions in the type *Motion Supervision*.
4. Save the changes.

Related information

The Motion Supervision type on page 470.

Application manual - Motion coordination and supervision.

6 Topic Motion

6.2.7. How to define transmission gear ratio for independent joints

6.2.7. How to define transmission gear ratio for independent joints

Transmission gear ratio

An independent joint can rotate in one direction for a long time, resetting the measurement system regularly. A small round-off in the transmission gear ratio can build up to large errors over time. The transmission gear ratio must therefore be given as an exact fraction (e.g. 10/3 instead of 3.333).

Define the transmission gear ratio by setting *Transmission Gear High* to the numerator and *Transmission Gear Low* to the denominator.

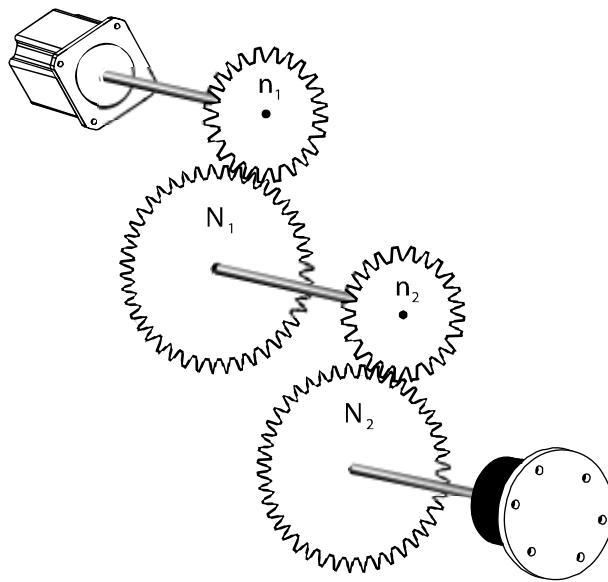
Limitations

The parameters *Transmission Gear High* and *Transmission Gear Low* are only useful if you have the RobotWare option *Independent Axes*.

When a joint is not in independent mode, it uses the parameter *Transmission Gear Ratio* instead of *Transmission Gear High* and *Transmission Gear Low*.

How to calculate transmission gear ratio

If the proportions for the transmission gear ratio are complex, count the cogs to get the exact ratio.



xx0300000285

In the illustration, the total transmission gear ratio is:

$$\frac{N_1 \times N_2}{n_1 \times n_2}$$

xx0300000272

N_1 , N_2 , n_1 and n_2 represent the number of cogs on each gearwheel.

To get an exact representation of the transmission gear ratio:

1. In the **Motion** topic, choose the type **Transmission**.
2. Decide which joint to define the transmission gear ratio.
3. Set the parameter *Transmission Gear High* to the value $N_1 \times N_2$.
4. Set the parameter *Transmission Gear Low* to the value $n_1 \times n_2$.

6.2.7. How to define transmission gear ratio for independent joints

Continued

Related information

The Transmission type on page 606.

Application manual - Motion functions and events.

6 Topic Motion

6.2.8. How to define external torque

External torque

When external equipment, for instance a cable or a coiled hose, affects any joint significantly, the external torque should be defined using the following formula:

$$T = A + |k \times (0 - \theta_0)|$$

T = external torque [Nm]

A = constant torque [Nm]

k = scale factor for position dependent torque [Nm]

θ_0 = joint position when position dependent torque is zero [rad]

If the estimated value of a significant external torque is too low, there can be unnecessary path deviations and the manipulator might be damaged. If the estimated value is too high, the performance of the manipulator is reduced due to restrictive acceleration limits.

How to define external torque

To define external torque:

1. In the **Motion** topic, choose the type **Arm**.
2. Select the arm to edit.
3. Set the desired values for the parameters *External Const Torque*, *External Proportional Torque*, and *External Torque Zero Angle*.
4. Save the changes

Related information

[The Arm type on page 298](#).

[External Const Torque on page 312](#).

[External Proportional Torque on page 313](#).

[External Torque Zero Angle on page 314](#).

Example

A coiled hose is mounted and affects joint 6 as follows:

0 Nm at 0 degrees.

5 Nm at 200 degrees.

This external torque can be defined using the following formula: $A = 0$, $\theta_0 = 0$, $k = 5 / (200 \times (\pi / 180))$

6.2.9. How to define supervision level

Supervision level

It is possible to change the default supervision levels if a system needs to be more or less tolerant to external disturbances. A higher tune factor than 1.0 gives a more tolerant robot system, and vice versa. E.g. increasing the tune factor from 1.0 to 2.0, doubles the allowed supervision levels, which makes the robot system more tolerant to external disturbances.

NOTE!



Increasing the tune factors can reduce the lifetime of the robot.

How to define the supervision level

To define the supervision level:

1. In the **Motion** topic, choose the type **Arm**.
2. Select the arm to change.
3. For the selected arm, set the desired values of the parameters *Jam Supervision Trim Factor*, *Load Supervision Trim Factor*, *Speed Supervision Trim Factor*, and *Position Supervision Trim Factor*.
4. Save the changes.

Related information

[The Arm type on page 298](#).

[Jam Supervision Trim Factor on page 308](#).

[Load Supervision Trim Factor on page 309](#).

[Speed Supervision Trim Factor on page 310](#).

[Position Supervision Trim Factor on page 311](#).

6 Topic Motion

6.3.1. The Acceleration Data type

6.3 Type Acceleration Data

6.3.1. The Acceleration Data type

Overview

This section describes the type *Acceleration Data*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

ACC_DATA

Type description

The type *Acceleration Data* is used to specify some acceleration characteristics for axes without any dynamic model. This is the case for certain additional axes.

For axes that have a dynamic model, *Acceleration Data* must still be specified even if a more complex model is normally used for the acceleration characteristics.

6.3.2. Name**Parent**

Name belongs to the type *Acceleration Data*, in the topic *Motion*.

Cfg name

name

Description

The name of the set of *Acceleration Data*.

Usage

Name is used to reference a set of *Acceleration Data* from the parameter *Use Acceleration Data* in the type *Arm*.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.3.3. Nominal Acceleration

Parent

Nominal Acceleration belongs to the type *Acceleration Data*, in the topic *Motion*.

Cfg name

wc_acc

Description

Worst case motor acceleration.

Usage

Set *Nominal Acceleration* to a value of the acceleration the axis can always perform (even when gravity and friction are unfavorable).

Nominal Acceleration is always used by axes without any dynamic model. For axes with dynamic model, it is only used in independent mode.

Allowed values

A numeric value between 0 and 1000, in rad/s² (or m/s²) on the arm side.

6.3.4. Nominal Deceleration

Parent

Nominal Deceleration belongs to the type *Acceleration Data*, in the topic *Motion*.

Cfg name

wc_dec

Description

Worst case motor deceleration.

Usage

Set *Nominal Deceleration* to a value of the deceleration the axis can always perform (even when gravity and friction are unfavorable).

Nominal Deceleration is always used by axes without any dynamic model. For axes with dynamic model, it is only used in independent mode.

Allowed values

A numeric value between 0 and 1000, in rad/s² (or m/s²) on the arm side.

6 Topic Motion

6.3.5. Acceleration Derivate Ratio

6.3.5. Acceleration Derivate Ratio

Parent

Acceleration Derivate Ratio belongs to the type *Acceleration Data*, in the topic *Motion*.

Cfg name

wc_dacc_ratio

Description

Acceleration Derivate Ratio defines how fast the acceleration can build up, i.e. an indication of the derivative of the acceleration.

Usage

If the derivative of the acceleration is not limiting the acceleration, set *Acceleration Derivate Ratio* to 1. If the acceleration must be increased at a slower rate, set *Acceleration Derivate Ratio* to a ratio of the maximum acceleration derivative (e.g. 0.5 to increase the acceleration half as fast as possible).

Limitations

Acceleration Derivate Ratio is not used during independent joint motion.

Allowed values

A numeric value between 0.1 and 1. The value has no unit, but is a ratio of the maximum acceleration derivative.

The default value is 1.

6.3.6. Deceleration Derivate Ratio

Parent

Deceleration Derivate Ratio belongs to the type *Acceleration Data*, in the topic *Motion*.

Cfg name

wc_ddec_ratio

Description

Deceleration Derivate Ratio defines how fast the deceleration can build up, i.e. an indication of the derivative of the deceleration.

Usage

If the derivative of the deceleration is not limiting the deceleration, set *Deceleration Derivate Ratio* to 1. If the deceleration must be increased at a slower rate, set *Deceleration Derivate Ratio* to a ratio of the maximum deceleration derivative (e.g. 0.5 to increase the deceleration half as fast as possible).

Limitations

Deceleration Derivate Ratio is not used during independent joint motion.

Allowed values

A numeric value between 0.1 and 1. The value has no unit, but is a ratio of the maximum deceleration derivative.

The default value is 1.

6 Topic Motion

6.4.1. The Arm type

6.4 Type Arm

6.4.1. The Arm type

Overview

This section describes the type *Arm*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

ARM

Type description

The *Arm* type contains a number of parameters that defines the characteristics for an arm. There is one set of parameters of the type *Arm* for each joint.

Related information

[How to define supervision level on page 291.](#)

[How to define external torque on page 290.](#)

6.4.2. Name**Parent**

Name belongs to the type *Arm*, in the topic *Motion*.

Cfg name

Name

Description

Name defines the name of the set of parameters for type *Arm*.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.4.3. Independent Joint

6.4.3. Independent Joint

Parent

Independent Joint belongs to the type *Arm*, in the topic *Motion*.

Cfg name

independent_joint_on

Description

Independent Joint is a flag for each axis that indicates whether the axis can be changed to independent mode.

Usage

Normally, all external axes and robot axis 6 allow independent mode. To prevent one of these axes moving independently, set *Independent Joint* to OFF for that axis.

Limitations

Independent Joint is only useful if you have the RobotWare option *Independent Axes*.

Allowed values

ON or OFF.

Related information

Application manual - Motion functions and events.

6.4.4. Upper Joint Bound

Parent

Upper Joint Bound belongs to the type *Arm*, in the topic *Motion*.

Cfg name

upper_joint_bound

Description

Upper Joint Bound defines the upper limit of the working area for this joint.

Usage

Upper Joint Bound can be used to limit the working area (in radians) of the joint.

Note that it is not possible to use a value that is larger than the maximal allowed limit for the specific joint. Trying this will cause the system to use the maximal allowed value instead.

Allowed values

A value between -1,256,637 and 1,256,637 radians.

6 Topic Motion

6.4.5. Lower Joint Bound

Parent

Lower Joint Bound belongs to the type *Arm*, in the topic *Motion*.

Cfg name

lower_joint_bound

Description

Lower Joint Bound defines the lower limit of the working area for this joint.

Usage

Lower Joint Bound can be used to limit the working area (in radians) of the joint.

Note that it is not possible to use a value that is smaller than the minimal allowed limit for the specific joint. Trying this will cause the system to use the minimal allowed value instead.

Allowed values

A value between -1,256,637 and 1,256,637 radians.

6.4.6. Independent Upper Joint Bound

Parent

Independent Upper Joint Bound belongs to the type *Arm*, in the topic *Motion*.

Cfg name

ind_upper_joint_bound

Description

Defines the upper limit of the working area for the joint when operating in independent mode.

Usage

Independent Upper Joint Bound is used together with *Independent Lower Joint Bound* to limit the work area for a joint that is in independent mode.

Limitations

Independent Upper Joint Bound is only useful if you have the option *Independent Axes*.

Allowed values

Any number (in radians).

Related information

Application manual - Motion functions and events.

6 Topic Motion

6.4.7. Independent Lower Joint Bound

6.4.7. Independent Lower Joint Bound

Parent

Independent Lower Joint Bound belongs to the type *Arm*, in the topic *Motion*.

Cfg name

ind_lower_joint_bound

Description

Defines the lower limit of the working area for the joint when operating in independent mode.

Usage

Independent Lower Joint Bound is used together with *Independent Upper Joint Bound* to limit the work area for a joint that is in independent mode.

Limitations

Independent Lower Joint Bound is only useful if you have the option *Independent Axes*.

Allowed values

Any number (in radians).

Related information

Application manual - Motion functions and events.

6.4.8. Calibration Position

Parent

Calibration Position belongs to the type *Arm*, in the topic *Motion*.

Cfg name

cal_position

Description

Calibration Position defines the position of the axis when it was fine calibrated.

Usage

If this value is to be updated, i.e. a fine calibration is to be performed, use *Calibration pendulum* to achieve the correct kinematic position of the axis and then fine calibrate the axis. It is then necessary to subsequently fine calibrate axes of higher order.

Allowed values

A value between -1000 and 1000, specifying the position in radians.

Related information

Product Manual for the manipulator.

Operating manual - Calibration Pendulum.

6 Topic Motion

6.4.9. Load Id Acceleration Ratio

Parent

Load Id Acceleration Ratio belongs to the type *Arm*, in the topic *Motion*.

Cfg name

load_id_acc_ratio

Description

Load Id Acceleration Ratio can be used to reduce the acceleration of the joint during load identification.

Usage

Reducing the acceleration of the joint during load identification can be useful if the torque supervision is triggered when identifying payloads with large inertia. If this happens, try to reduce the value of *Load Id Acceleration Ratio* until the problem disappears.

Allowed values

A number between 0.2 and 1.0.

6.4.10. Performance Quota

Parent

Performance Quota belongs to the type *Arm*, in the topic *Motion*.

Cfg name

performance_quota

Description

Performance Quota can be used to reduce the acceleration for the joint.

Usage

Setting *Performance Quota* value to 1.0 gives normal performance, but if less acceleration is desired, a lower value can be entered.

Allowed values

A number between 0.45 and 1.0.

6 Topic Motion

6.4.11. Jam Supervision Trim Factor

6.4.11. Jam Supervision Trim Factor

Parent

Jam Supervision Trim Factor belongs to the type *Arm*, in the topic *Motion*.

Cfg name

supervision_jam_time_factor

Description

Jam Supervision Trim Factor defines the tune factor for jam supervision.

Usage

The tune factor influences the maximum time allowed at zero speed with maximum torque.

Allowed values

A number between 0.1 and 10.0.

Related information

[How to define supervision level on page 291](#)

6.4.12. Load Supervision Trim Factor

Parent

Load Supervision Trim Factor belongs to the type *Arm*, in the topic *Motion*.

Cfg name

supervision_load_factor

Description

Load Supervision Trim Factor defines the tune factor for load supervision.

Usage

The factor influences the maximum time allowed at non-zero speed with maximum torque.

Allowed values

A number between 0.1 and 10.0.

Related information

[How to define supervision level on page 291.](#)

6 Topic Motion

6.4.13. Speed Supervision Trim Factor

Parent

Speed Supervision Trim Factor belongs to the type *Arm*, in the topic *Motion*.

Cfg name

supervision_speed_factor

Description

Speed Supervision Trim Factor defines the tune factor for speed supervision.

Usage

The factor influences the maximum allowed speed error.

Allowed values

A number between 0.05 and 10.0.

Related information

How to define supervision level on page 291.

6.4.14. Position Supervision Trim Factor

Parent

Position Supervision Trim Factor belongs to the type *Arm*, in the topic *Motion*.

Cfg name

supervision_pos_factor

Description

Position Supervision Trim Factor defines the tune factor for position supervision.

Usage

The factor influences the maximum allowed position error.

Allowed values

A number between 0.1 and 10.0.

Related information

How to define supervision level on page 291.

6 Topic Motion

6.4.15. External Const Torque

6.4.15. External Const Torque

Parent

External Const Torque belongs to the type *Arm*, in the topic *Motion*.

Cfg name

ext_const_torque

Description

External Const Torque defines the external constant torque.

Usage

The value of *External Const Torque* is used in the formula for calculation of external torque.

Allowed values

A value between 0 and 100,000, specifying the constant torque in Nm.

Related information

[How to define external torque on page 290.](#)

6.4.16. External Proportional Torque

Parent

External Proportional Torque belongs to the type *Arm*, in the topic *Motion*.

Cfg name

ext_prop_torque

Description

External Proportional Torque defines the scale factor for position-dependent torque.

Usage

The value of *External Proportional Torque* is used in the formula for calculation of external torque.

Allowed values

A value between -100,000 and 100,000, specifying the scale factor in Nm/rad.

Related information

How to define external torque on page 290.

6 Topic Motion

6.4.17. External Torque Zero Angle

Parent

External Torque Zero Angle belongs to the type *Arm*, in the topic *Motion*.

Cfg name

ext_prop_zero_angle

Description

External Torque Zero Angle defines the joint position when position-dependent torque is zero.

Usage

The value of *External Torque Zero Angle* is used in the formula for calculation of external torque.

Allowed values

A value between -100,000 and 100,000, specifying the position in radians.

Related information

[How to define external torque on page 290.](#)

6.4.18. Angle Acceleration Ratio

Parent

Angle Acceleration Ratio belongs to the type *Arm*, in the topic *Motion*.

Cfg name

angle_acc_ratio

Description

Angle Acceleration Ratio defines the maximum angle acceleration ratio for the motor sensor.

Usage

This parameter should only be changed by ABB.

Allowed values

A value between 0.02 and 1.0. Default value is 1.0.

6 Topic Motion

6.5.1. The Arm Check Point type

6.5 Type Arm Check Point

6.5.1. The Arm Check Point type

Overview

This section describes the type *Arm Check Point*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic.

Cfg name

ARM_CHECK_POINT

Type description

If an extra load, such as a transformer or a welding-bar roller, is attached to arm 3, a point on this equipment can be defined as a check point. The robot will then monitor the speed of this point so that it does not exceed 250 mm/s in manual reduced speed mode.

Related information

[How to define arm check point on page 283.](#)

6.5.2. Name

Parent

Name belongs to the type *Arm Check Point*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the arm check point. A check point can be used to let the robot monitor the speed of that specified point

Allowed values

A string with maximum 24 characters.

Related information

How to define arm check point on page 283.

6 Topic Motion

6.5.3. Position x, y, z

6.5.3. Position x, y, z

Parent

Position x, Position y, and Position z belongs to the type *Arm Check Point*, in the topic *Motion*.

Cfg names

position_x

position_y

position_z

Description

Position x defines the x-coordinate of the position of the check point, specified on the basis of the current frame of the arm (in meters).

Position y defines the y-coordinate of the position of the check point, specified on the basis of the current frame of the arm (in meters).

Position z defines the z-coordinate of the position of the check point, specified on the basis of the current frame of the arm (in meters).

Allowed values

A value between -3 to 3, specifying the position in meters.

Related information

How to define arm check point on page 283.

6.6 Type Arm Load

6.6.1. The Arm Load type

Overview

This section describes the type *Arm Load*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

ARM_LOAD

Type description

Arm Load is used for defining loads from equipment mounted on robot arms. If the arm load is not defined when equipment is mounted on the robot arm, the performance of the robot is negatively affected.

Related information

How to define arm loads on page 285.

6 Topic Motion

6.6.2. Name

6.6.2. Name

Parent

Name belongs to the type *Arm Load*, in the topic *Motion*.

Cfg name

name

Description

Name specifies the name of the arm load setting it belongs to.

Allowed values

A string with maximum 32 characters, specifying the name.

Related information

How to define arm loads on page 285.

6.6.3. Mass

Parent

Mass belongs to the type *Arm Load*, in the topic *Motion*.

Cfg name

mass

Description

Mass specifies the mass of the equipment mounted on a robot arm.

Allowed values

A value between 0 and 500, specifying the weight in kg.

Related information

How to define arm loads on page 285.

6 Topic Motion

6.6.4. Mass Center x, y, z

6.6.4. Mass Center x, y, z

Parent

Mass Center x, *Mass Center y*, and *Mass Center z* belongs to the type *Arm Load*, in the topic *Motion*.

Cfg names

mass_centre_x
mass_centre_y
mass_centre_z

Description

Mass Center x specifies the x-coordinate of the mass center for an arm load in the arm frame.

Mass Center y specifies the y-coordinate of the mass center for an arm load in the arm frame.

Mass Center z specifies the z-coordinate of the mass center for an arm load in the arm frame.

Allowed values

A value between -3 and + 3, specifying the coordinate in meters.

Related information

How to define arm loads on page 285.

6.6.5. Inertia x, y, z

Parent

Inertia x, *Inertia y*, and *Inertia z* belongs to the type *Arm Load*, in the topic *Motion*.

Cfg names

inertia_x
inertia_y
inertia_z

Description

Inertia x defines the x-component of the arm load's moment of inertia relative to the load's mass center around the arm's coordinate axes.

Inertia y defines the y-component of the arm load's moment of inertia relative to the load's mass center around the arm's coordinate axes.

Inertia z defines the z-component of the arm load's moment of inertia relative to the load's mass center around the arm's coordinate axes.

Allowed values

A value between 0 and 100, specifying the moment of inertia in kgm^2 .

Related information

[How to define arm loads on page 285.](#)

6 Topic Motion

6.7.1. The Brake type

6.7 Type Brake

6.7.1. The Brake type

Overview

This section describes the type *Brake* which belongs to the topic *Motion*.

Cfg name

BRAKE

Type description

The type *Brake* is used to specify brake parameters for a specific joint.

Related information

The Joint type on page 390.

6.7.2. Name**Parent**

Name belongs to the type *Brake*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the brake.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.7.3. Control Off Speed Limit

6.7.3. Control Off Speed Limit

Parent

Control Off Speed Limit belongs to the type *Brake*, in the topic *Motion*.

Cfg name

control_off_speed_limit

Description

Control Off Speed Limit defines the speed for selection of delay time.

Usage

The value for *Control Off Speed Limit* should not be modified.

Allowed values

A value between 0 and 1.

Default value is 0.02.

6.7.4. Control Off Delay

Parent

Control Off Delay belongs to the type *Brake*, in the topic *Motion*.

Cfg name

control_off_delay_time

Description

Control Off Delay specifies the time of normal control before the motor torque is set to zero.

Usage

Control Off Delay is used when the joint is at zero speed when the brake algorithm is activated. The controller must be active to avoid the joint to fall by gravity before the mechanical brake is engaged.

Time must be longer than the time for mechanical brake to engage.

Allowed values

A value between 0 and 30 seconds.

Default value is 0.010 seconds.

6 Topic Motion

6.7.5. Brake Control On Delay

Parent

Brake Control On Delay belongs to the type *Brake*, in the topic *Motion*.

Cfg name

brake_control_on_delay_time

Description

Brake Control On Delay specifies the time of normal control before the motor torque is set to zero.

Usage

Brake Control On Delay is used if the joint is moving when the brake algorithm is activated. The controller must be active to avoid oscillations when the mechanical brake is engaged.

The time must be longer than the time for mechanical brake to engage. Normally set to same value as parameter *Control Off Delay*.

Allowed values

A value between 0 and 30 seconds.

Default value is 0.

Related information

[Control Off Delay on page 327](#).

6.7.6. Brake Control Min Delay

Parent

Brake Control Min Delay belongs to the type *Brake*, in the topic *Motion*.

Cfg name

brake_control_on_min_delay_time

Description

Brake Control Min Delay defines the minimum delay time.

Usage

Brake Control Min Delay should not be changed.

Allowed values

A value between 0 and 5 seconds.

Default value is 0.010.

6 Topic Motion

6.7.7. Absolute Brake Torque

Parent

Absolute Brake Torque belongs to the type *Brake*, in the topic *Motion*.

Cfg name

absolute_brake_torque

Description

Absolute Brake Torque defines the brake torque to be used for a simulated electrical brake.

Usage

Absolute Brake Torque should not be changed.

Allowed values

A value between 0 and 100,000 Nm.

Default value is 0.

6.7.8. Brake Ramp Speed Limit

Parent

Brake Ramp Speed Limit belongs to the type *Brake*, in the topic *Motion*.

Cfg name

brake_ramp_speed_limit

Description

Brake Ramp Speed Limit is the point of torque reduction for simulated electrical brake.

Usage

Brake Ramp Speed Limit should not be changed.

Allowed values

A value between 0 and 1.

Default value is 1 (equal to 100%).

6 Topic Motion

6.8.1. The Control Parameters type

6.8 Type Control Parameters

6.8.1. The Control Parameters type

Overview

This section describes the type *Control Parameters*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

CONTROL_PARAMETERS

Type description

Each set of parameters of the type *Control Parameters* belongs to a joint (robot joint or external axis).

The parameters in *Control Parameters* define what compensations should be made for the friction in the joint.

Limitation

Changing the parameter values in *Control Parameters* is only useful if you have the RobotWare option *Advanced Shape Tuning*.

The type *Control Parameters* is only used by robot models IRB 1400 and IRB 1410. All other robot models use the type *Friction Compensation* instead. The parameters are the same however.

Related information

Application manual - Motion performance, chapter *Advanced Shape Tuning*.

6.8.2. Name

Parent

Name belongs to the type *Control Parameters*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name to use for the control parameters.

Limitations

Name is only useful if you have the RobotWare option *Advanced Shape Tuning*.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.8.3. Friction FFW On

6.8.3. Friction FFW On

Parent

Friction FFW On belongs to the type *Control Parameters*, in the topic *Motion*.

Cfg name

friction_ffw_on

Description

Friction FFW On determines if the RobotWare option *Advanced Shape Tuning* is active or not.

Usage

Set *Friction FFW On* to TRUE if you want to use *Advanced Shape Tuning*.

Limitations

Friction FFW On is only useful if you have the RobotWare option *Advanced Shape Tuning*.

Allowed values

TRUE or FALSE.

Related information

Application manual - Motion performance.

6.8.4. Friction FFW Level

Parent

Friction FFW Level belongs to the type *Control Parameters*, in the topic *Motion*.

Cfg name

friction_ffw_level

Description

Friction FFW Level is set to the level of friction in the robot axis. By setting a value that closely corresponds to the real friction, and using the RobotWare option *Advanced Shape Tuning*, the friction effects can be compensated.

Usage

Friction effects can cause path deviations when performing advanced shapes. By compensating for the friction with the correct friction level value, these effects can be minimized.

Permanent adjustments of the friction level can be made with *Friction FFW Level*. The friction level can also be temporarily tuned with RAPID commands.

Limitations

Friction FFW Level is only useful if you have the RobotWare option *Advanced Shape Tuning*.

Allowed values

A decimal number between 0 and 15 (in Nm).

Related information

Application manual - Motion performance.

6 Topic Motion

6.8.5. Friction FFW Ramp

6.8.5. Friction FFW Ramp

Parent

Friction FFW Ramp belongs to the type *Control Parameters*, in the topic *Motion*.

Cfg name

friction_ffw_ramp

Description

Friction FFW Ramp is set to the speed of the robot axis when the friction has reached the constant friction level defined in *Friction FFW Level*. See illustration below.

Usage

Friction effects can cause path deviations when performing advanced shapes. *Friction FFW Ramp* is used when compensating for these friction effects.

Permanent adjustments of the friction ramp can be made with *Friction FFW Ramp*. The friction ramp can also be temporarily tuned with RAPID commands.

Limitations

Friction FFW Ramp is only useful if you have the RobotWare option *Advanced Shape Tuning*.

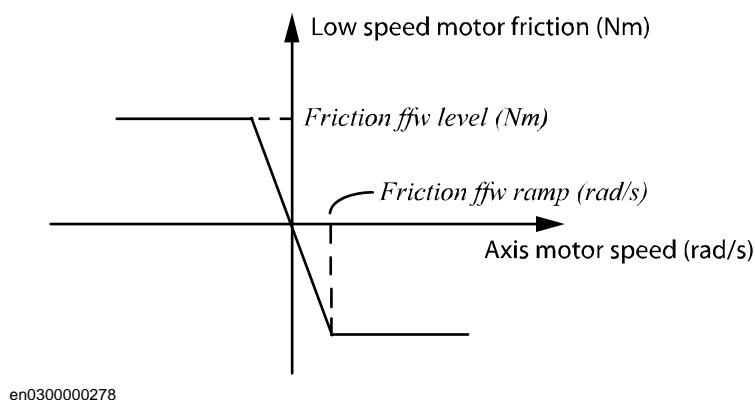
Allowed values

A number between 0.001 and 10 (in radians/second).

Related information

Application manual - Motion performance.

Illustration



© Copyright 2004-2007 ABB. All rights reserved.

6.9 Type Drive Module

6.9.1. The Drive Module type

Overview

This section describes the type *Drive Module*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

DRIVE_MODULE

Type description

The type *Drive Module* is used to identify and specify each drive module used in the robot system. There is one set of parameters of the type *Drive Module* for each drive module in the robot system.

Limitations

If the robot system does not use *MultiMove*, there is only one drive module, and therefore only set of parameters of the type *Drive Module*.

6 Topic Motion

6.9.2. Name

6.9.2. Name

Parent

Name belongs to the type *Drive Module*, in the topic *Motion*.

Cfg name

name

Description

Defines the unique name of the drive module.

Allowed values

A string with maximum 32 characters.

6.9.3. Number

Parent

Number belongs to the type *Drive Module*, in the topic *Motion*.

Cfg name

number

Description

Defines the identifying number of the drive module.

Usage

The drive module number is used to identify the drive module by other system parameters.

Allowed values

An integer between 1 and 4.

6 Topic Motion

6.10.1. The Drive System type

6.10 Type Drive System

6.10.1. The Drive System type

Overview

This section describes the type *Drive System*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

DRIVE_SYSTEM

Type description

The type *Drive System* is used to identify and specify each drive system used in the robot system.

6.10.2. Name**Parent**

Name belongs to the type *Drive System*, in the topic *Motion*.

Cfg name

name

Description

Defines the name for the drive system.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.10.3. Use Drive Unit

6.10.3. Use Drive Unit

Parent

Use Drive Unit belongs to the type *Drive System*, in the topic *Motion*.

Cfg name

use_drive_unit

Description

Use Drive Unit determines which drive unit should be used.

Usage

The name of the drive unit corresponds to which node it is located on.

For example: M1DM1: node 1, drive module 1 etc.

Allowed values

A string with maximum 32 characters.

6.10.4. Current Vector On

Parent

Current Vector On belongs to the type *Drive System*, in the topic *Motion*.

Cfg name

current_vector_on

Description

Current Vector On defines if the vector control is active.

Usage

Current Vector On controls an activation switch. It is used to prevent that an axis with uncommutated motor runs away at start up.

The parameter is reset by the service routine COMMUTATION, or manually via RobotStudio Online or FlexPendant.

Allowed values

True or False.

Default value is False.

Related information

Application manual - Additional axes and stand alone controller, section *Tuning*.

6 Topic Motion

6.11.1. The Force Master type

6.11 Type Force Master

6.11.1. The Force Master type

Overview

This section describes the type *Force Master*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

FORCE_MASTER

Type description

Force Master is used to define how a servo gun behaves during the two faces of the gun closing:

- when approaching the point where position regulation is replaced by force control
- during force control.

Values for position, torque, force, etc. are specified for calibration and gun closing.

Limitations

Force Master can only be used for servo tools.

Non-editable parameters

The following parameters are visible but not editable in the software configuration tools:

- *Force Detection Speed*
- *Max Pos Err Closing*

As a consequence, the above parameters are not described in the manual.

Related information

Application manual - Servo motor control.

6.11.2. Name**Parent**

Name belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

name

Description

The name of the *Force Master*.

Usage

Name is used to reference a *Force Master* from the parameter *Use Force Master* in the type *SG Process*.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.11.3. Use Force Master Control

Parent

Use Force Master Control belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

use_force_master_control

Description

Use Force Master Control determines which *Force Master Control* should be used.

Usage

Use Force Master Control is a reference to the parameter *Name* in the type *Force Master Control*.

Prerequisites

A *Force Master Control* must be configured before *Use Force Master Control* can refer to it.

Limitations

Use Force Master Control can only be used for servo tools.

Allowed values

A string with maximum 32 characters.

Related information

[The Force Master Control type on page 358.](#)

6.11.4. References Bandwidth

Parent

References Bandwidth belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

bandwidth_ramping

Description

The frequency limit for the low pass filter for reference values. During position regulation, when approaching the plate thickness, position and speed values will be filtered in this low pass filter to avoid sharp step functions.

Usage

A high value on *References Bandwidth* will make little use of the low pass filter.

If the servo tool is vibrating due to irregular movements, *References Bandwidth* can be set to a lower value. A low value will make the servo tool movements slower.

Limitations

References Bandwidth can only be used for servo tools.

Allowed values

A numeric value between 1 and 124 (Hz).

The default value is 25 Hz.

6 Topic Motion

6.11.5. Use Ramp Time

Parent

Use Ramp Time belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

ramp_time_switch

Description

Determines if the ramping of the tip force should use a constant time or a constant gradient.

Usage

If the tip force should be ramped up to its ordered value during the time specified in *Ramp Time*, set *Use Ramp Time* to Yes. The ramp rate will then vary to make the ramp time constant.

If the tip force should be increased at a constant rate, specified in *Ramp when Increasing Force*, set *Use Ramp Time* to No. The ramp time will then vary to make the ramp rate constant.

Limitations

Use Ramp Time can only be used for servo tools.

Allowed values

Yes or No.

6.11.6. Ramp when Increasing Force

Parent

Ramp when Increasing Force belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

ramp_torque_ref_closing

Description

Ramp when Increasing Force decides how fast the torque is ramped up to the ordered torque after contact position is reached at a close gun command.

Usage

A higher value of *Ramp when Increasing Force* will make the tip force build up faster.

Prerequisites

Ramp when Increasing Force is only used if *Use Ramp Time* is set to No.

Limitations

Ramp when Increasing Force can only be used for servo tools.

Allowed values

A value between 1 and 10000, specifying the torque increase in Nm/s.

The default value is 100 Nm/s.

6 Topic Motion

6.11.7. Ramp Time

6.11.7. Ramp Time

Parent

Ramp Time belongs to the type *Force Control*, in the topic *Motion*.

Cfg name

ramp_time

Description

Ramp Time decides how fast the torque is ramped up to the ordered torque after contact position is reached at a close gun command.

Usage

A lower value of *Ramp Time* will make the tip force build up faster.

Prerequisites

Ramp Time is only used if *Use Ramp Time* is set to Yes.

Limitations

Ramp Time can only be used for servo tools.

Allowed values

A numeric value between 0.001 and 1 (seconds).

The default value is 0.07 s.

6.11.8. Collision LP Bandwidth

Parent

Collision LP Bandwidth belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

bandwidth_lp

Description

Frequency limit for the low pass filter used for tip wear calibration. Position and speed reference values will be filtered in this low pass filter to avoid sharp step functions.

Usage

The only reason for changing *Collision LP Bandwidth* is if repetitive tip wear calibrations give varying results. A lower value for the low pass filter can stabilize the servo tool during the calibration.

Limitations

Collision LP Bandwidth can only be used for servo tools.

Allowed values

A numeric value between 0 and 124 (Hz).

The default value is 25 Hz.

6 Topic Motion

6.11.9. Collision Alarm Torque

6.11.9. Collision Alarm Torque

Parent

Collision Alarm Torque belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

alarm_torque

Description

Collision Alarm Torque determines how hard the tool tips will be pressed together during the first gun closing of new tips calibrations and tool change calibrations.

Usage

Collision Alarm Torque is used for the first gun closing of new tips calibrations and tool change calibrations. This affects the position calibration.

The best way to determine the collision position (where the tool tips meet) is to keep closing the gun until the motor torque reaches the value specified in *Collision Alarm Torque*. The distance the gun then has moved beyond the collision position is defined by the parameter *Collision Delta Position*.

Limitations

Collision Alarm Torque can only be used for servo tools.

Allowed values

A value between 0 and 50 (Nm).

The default value is 1.5 Nm.

6.11.10. Collision Speed

Parent

Collision Speed belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

col_speed

Description

Collision Speed determines the servo gun speed during the first gun closing of new tips calibrations and tool change calibrations. These calibrations affect the position calibration.

Usage

The only reason for changing *Collision Speed* is if repetitive tip wear calibrations give varying results. A lower speed can improve the repeatability.

Limitations

Collision Speed can only be used for servo tools.

Allowed values

A value between 0 and 5 (m/s).

The default value is 0.02 m/s.

6 Topic Motion

6.11.11. Collision Delta Position

Parent

Collision Delta Position belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

distance_to_contact_position

Description

Collision Delta Position defines the distance the servo tool has gone beyond the contact position when the motor torque has reached the value specified in *Collision Alarm Torque*.

Usage

Collision Delta Position is used for the first gun closing of new tips calibrations and tool change calibrations. This affects the position calibration.

The best way to determine the collision position (where the tool tips meet) is to keep closing the gun until the motor torque reach the value specified in *Collision Alarm Torque*. The distance the gun then has moved beyond the collision position is defined in *Collision Delta Position*.

Changing the value of *Collision Delta Position* can remove a constant calibration error, but does not affect if repetitive tip wear calibrations give varying results.

Limitations

Collision Delta Position can only be used for servo tools.

Allowed values

A value between 0 and 1 (m).

The default value is 0.0019 m.

6.11.12. Force Detection Bandwidth

Parent

Force Detection Bandwidth belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

force_ready_detection_bandwidth

Description

Defines the bandwidth for the force detection filter.

Usage

The force detection filter is used to filter the speed of the servo tool. The filtered speed is used to detect if the ordered force has been reached.

Limitations

Force Detection Bandwidth can only be used for servo tools.

Allowed values

A value between 1 and 124 Hz.

6 Topic Motion

6.11.13. Delay ramp

6.11.13. Delay ramp

Parent

Delay ramp belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

delay_ramp

Description

Delays the starting of torque ramp when force control is started.

Usage

Delay ramp can be used to give the servo gun some time to stabilize before the force control starts. A higher value of *Delay ramp* can result in better accuracy of the squeeze force but will increase the cycle time.

Limitations

Delay ramp can only be used for servo tools.

Allowed values

A numeric value between 0 and 1 (seconds).

6.11.14. Ramp to real contact

Parent

Ramp to real contact belongs to the type *Force Master*, in the topic *Motion*.

Cfg name

ramp_to_real_contact

Description

Determines if the feedback position should be used instead of reference position when deciding the contact position.

Usage

Setting *Ramp to real contact* to Yes will make the detection of the contact position (where the force control starts) more exact and improve the accuracy of the squeeze force, but increase the cycle time.

Limitations

Ramp to real contact can only be used for servo tools.

Allowed values

Yes or No.

6 Topic Motion

6.12.1. The Force Master Control type

6.12 Type Force Master Control

6.12.1. The Force Master Control type

Overview

This section describes the type *Force Master Control*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

FORCE_MASTER_CONTROL

Type description

Force Master Control is used to prevent a servo tool from closing with too high a speed. If a servo tool is not completely closed when the force control starts, it can gain too much speed, which can cause damages when contact is reached. This can happen if the programmed thickness is too high, or if the servo tool tips are not properly calibrated. If the tool is ordered to close with a higher force, it might tolerate a higher speed at impact. The speed limit can be defined as a function of the closing torque, which is a function of the ordered tip force. The loop gain used for regulating the speed when it exceeds the limit is also specified.

Up to 6 points can be defined for speed limit and speed loop gain.

Ordered closing torque:	Speed limit:	Speed loop gain:
torque 1	Speed Limit 1	Kv 1
torque 2	Speed Limit 2	Kv 2
torque 3	Speed Limit 3	Kv 3
torque 4	Speed Limit 4	Kv 4
torque 5	Speed Limit 5	Kv 5
torque 6	Speed Limit 6	Kv 6

Speed limit 1 and *Kv 1* are valid for all torque values lower than *torque 1*. The highest defined speed limit and loop gain are valid for all torque values higher than the highest defined torque. For torque values between defined points, linear interpolation is used.

If only one point is defined, that speed limit and speed loop gain is valid for all torque values.

Limitations

Force Master Control can only be used if you have servo tools.

Related information

Application manual - Servo motor control.

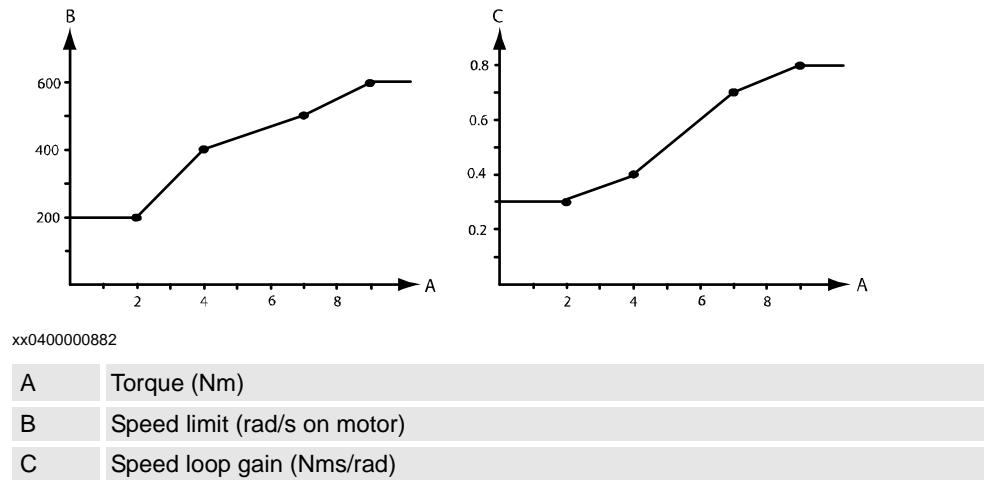
*Continued***Example**

In this example, four points are used to define the speed limit and speed loop gain. Any values given for point 5 and 6 are ignored.

The parameters in the type *Force Master Control* are set to the following values:

Parameter:	Value:
No. of speed limits	4
torque 1	2
torque 2	4
torque 3	7
torque 4	9
Speed Limit 1	200
Speed Limit 2	400
Speed Limit 3	500
Speed Limit 4	600
Kv 1	0.3
Kv 2	0.4
Kv 3	0.7
Kv 4	0.8

The results of this configuration are the following graphs for speed limit and speed loop gain:



6 Topic Motion

6.12.2. Name

6.12.2. Name

Parent

Name belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

name

Description

The name of the *Force Master Control*.

Usage

Name is used to reference a *Force Master Control* from the parameter *Use Force Master* in the type *Force Master*.

Allowed values

A string with maximum 32 characters.

6.12.3. No. of Speed Limits

Parent

No. of Speed Limits belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

no_of_posts

Description

No. of Speed Limits defines the number of torque values you want to define for speed limit and speed loop gain, i.e. the number of points in the speed limit graph (see *Example on page 359*).

Usage

Define the speed limit and speed loop gain you want for a number of torque values. Set *No. of Speed Limits* to the number of torque values you want to specify.

Limitations

No. of Speed Limits can only be used if you have servo tools.

Allowed values

An integer between 1 and 6.

The default value is 1.

Related information

Application manual - Servo motor control.

6 Topic Motion

6.12.4. torque 1

6.12.4. torque 1

Parent

torque 1 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

torque_1

Description

torque 1 defines the ordered closing torque for the first point in the speed limit graph (see [Example on page 359](#)).

Usage

Define the speed limit and speed loop gain you want for some torque values. Set *torque 1* to the torque value of the first point you want to specify.

Limitations

torque 1 is used for servo tools and can only be used if you have the option *Servo Tool Control*.

Allowed values

A number between -1000 and 1000 in Nm.

The default value is 1 Nm.

Related information

Application manual - Servo motor control.

6.12.5. torque 2

Parent

torque 2 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

torque_2

Description

torque 2 defines the ordered closing torque for the second point (if more than one) in the speed limit graph (see [Example on page 359](#)).

Usage

Define the speed limit and speed loop gain you want for some torque values. Set *torque 2* to the torque value of the second point you want to specify.

Prerequisites

No. of Speed Limits must be set to 2 or higher, otherwise the value of *torque 2* is not used.

Limitations

torque 2 can only be used if you have servo tools.

Allowed values

A number between -1000 and 1000 in Nm.

The default value is 2 Nm.

Related information

[No. of Speed Limits on page 361](#).

[Application manual - Servo motor control](#).

6 Topic Motion

6.12.6. torque 3

6.12.6. torque 3

Parent

torque 3 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

torque_3

Description

torque 3 defines the ordered closing torque for the third point (if more than two) in the speed limit graph (see [Example on page 359](#)).

Usage

Define the speed limit and speed loop gain you want for some torque values. Set *torque 3* to the torque value of the third point you want to specify.

Prerequisites

No. of Speed Limits must be set to 3 or higher, otherwise the value of *torque 3* is not used.

Limitations

torque 3 can only be used if you have servo tools.

Allowed values

A number between -1000 and 1000 in Nm.

The default value is 3 Nm.

Related information

[No. of Speed Limits on page 361](#).

[Application manual - Servo motor control](#).

6.12.7. torque 4

Parent

torque 4 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

torque_4

Description

torque 4 defines the ordered closing torque for the fourth point (if more than three) in the speed limit graph (see [Example on page 359](#)).

Usage

Define the speed limit and speed loop gain you want for some torque values. Set *torque 4* to the torque value of the fourth point you want to specify.

Prerequisites

No. of Speed Limits must be set to 4 or higher, otherwise the value of *torque 4* is not used.

Limitations

torque 4 can only be used if you have servo tools.

Allowed values

A number between -1000 and 1000 in Nm.

The default value is 4 Nm.

Related information

[No. of Speed Limits on page 361](#).

[Application manual - Servo motor control](#).

6 Topic Motion

6.12.8. torque 5

6.12.8. torque 5

Parent

torque 5 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

torque_5

Description

torque 5 defines the ordered closing torque for the fifth point (if more than four) in the speed limit graph (see [Example on page 359](#)).

Usage

Define the speed limit and speed loop gain you want for some torque values. Set *torque 5* to the torque value of the fifth point you want to specify.

Prerequisites

No. of Speed Limits must be set to 5 or higher, otherwise the value of *torque 5* is not used.

Limitations

torque 5 can only be used if you have servo tools.

Allowed values

A number between -1000 and 1000 in Nm.

The default value is 5 Nm.

Related information

[No. of Speed Limits on page 361](#).

[Application manual - Servo motor control](#).

6.12.9. torque 6

Parent

torque 6 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

torque_6

Description

torque 6 defines the ordered closing torque for the sixth point (if all six points are used) in the speed limit graph (see *Example on page 359*).

Usage

Define the speed limit and speed loop gain you want for some torque values. Set *torque 6* to the torque value of the sixth point you want to specify.

Prerequisites

No. of Speed Limits must be set to 6, otherwise the value of *torque 6* is not used.

Limitations

torque 6 can only be used if you have servo tools.

Allowed values

A number between -1000 and 1000 in Nm.

The default value is 6 Nm.

Related information

No. of Speed Limits on page 361.

Application manual - Servo motor control.

6 Topic Motion

6.12.10. Speed Limit 1

Parent

Speed Limit 1 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

speed_lim_1

Description

Speed Limit 1 defines the maximum allowed speed for the torque specified in *torque 1*.

Usage

Set *Speed Limit 1* to the speed limit for the first point you want to specify in the speed limit graph (see [Example on page 359](#)).

Limitations

Speed Limit 1 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100000 in rad/s on the motor side.

The default value is 300.

Related information

[torque 1 on page 362](#).

[Application manual - Servo motor control](#).

6.12.11. Speed Limit 2

Parent

Speed Limit 2 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

speed_lim_2

Description

Speed Limit 2 defines the maximum allowed speed for the torque specified in *torque 2*.

Usage

Set *Speed Limit 2* to the speed limit for the second point (if more than one) you want to specify in the speed limit graph (see [Example on page 359](#)).

Prerequisites

No. of Speed Limits must be set to 2 or higher, otherwise the value of *Speed Limit 2* is not used.

Limitations

Speed Limit 2 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100000 in rad/s on the motor side.

The default value is 300.

Related information

[torque 2 on page 363](#).

[No. of Speed Limits on page 361](#).

[Application manual - Servo motor control](#).

6 Topic Motion

6.12.12. Speed Limit 3

Parent

Speed Limit 3 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

speed_lim_3

Description

Speed Limit 3 defines the maximum allowed speed for the torque specified in *torque 3*.

Usage

Set *Speed Limit 3* to the speed limit for the third point (if more than two) you want to specify in the speed limit graph (see [Example on page 359](#)).

Prerequisites

No. of Speed Limits must be set to 3 or higher, otherwise the value of *Speed Limit 3* is not used.

Limitations

Speed Limit 3 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100000 in rad/s on the motor side.

The default value is 300.

Related information

[torque 3 on page 364](#).

[No. of Speed Limits on page 361](#).

[Application manual - Servo motor control](#).

6.12.13. Speed Limit 4

Parent

Speed Limit 4 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

speed_lim_4

Description

Speed Limit 4 defines the maximum allowed speed for the torque specified in *torque 4*.

Usage

Set *Speed Limit 4* to the speed limit for the fourth point (if more than three) you want to specify in the speed limit graph (see [Example on page 359](#)).

Prerequisites

No. of Speed Limits must be set to 4 or higher, otherwise the value of *Speed Limit 4* is not used.

Limitations

Speed Limit 4 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100000 in rad/s on the motor side.

The default value is 300.

Related information

[torque 4 on page 365](#).

[No. of Speed Limits on page 361](#).

[Application manual - Servo motor control](#).

6 Topic Motion

6.12.14. Speed Limit 5

Parent

Speed Limit 5 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

speed_lim_5

Description

Speed Limit 5 defines the maximum allowed speed for the torque specified in *torque 5*.

Usage

Set *Speed Limit 5* to the speed limit for the fifth point (if more than four) you want to specify in the speed limit graph (see [Example on page 359](#)).

Limitations

Speed Limit 5 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100000 in rad/s on the motor side.

The default value is 300.

Related information

[torque 5 on page 366](#).

[No. of Speed Limits on page 361](#).

6.12.15. Speed Limit 6

Parent

Speed Limit 6 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

speed_lim_6

Description

Speed Limit 6 defines the maximum allowed speed for the torque specified in *torque 6*.

Prerequisites

No. of Speed Limits must be set to 6, otherwise the value of *Speed Limit 6* is not used.

Usage

Set *Speed Limit 6* to the speed limit for the sixth point (if all six points are used) you want to specify in the speed limit graph (see [Example on page 359](#)).

Limitations

Speed Limit 6 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100000 in rad/s on the motor side.

The default value is 300.

Related information

[torque 6 on page 367](#).

[No. of Speed Limits on page 361](#).

[Application manual - Servo motor control](#).

6 Topic Motion

6.12.16. Kv 1

6.12.16. Kv 1

Parent

Kv 1 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

Kv_1

Description

Kv 1 defines the proportional gain in the speed loop for the torque specified in *torque 1*. This gain determines how fast the speed is regulated when the speed limit is exceeded.

Usage

Set *Kv 1* to the proportional gain you want for the first point in the speed limit graph (see [Example on page 359](#)).

Limitations

Kv 1 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100.

The default value is 0.5.

Related information

[torque 1 on page 362](#).

[Application manual - Servo motor control](#).

6.12.17. Kv 2

Parent

Kv 2 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

Kv_2

Description

Kv 2 defines the proportional gain in the speed loop for the torque specified in *torque 2*. This gain determines how fast the speed is regulated when the speed limit is exceeded.

Usage

Set *Kv 2* to the proportional gain you want for the second point (if more than one) in the speed limit graph (see [Example on page 359](#)).

Prerequisites

No. of Speed Limits must be set to 2 or higher, otherwise the value of *Kv 2* is not used.

Limitations

Kv 2 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100.

The default value is 0.5.

Related information

[torque 2 on page 363](#).

[No. of Speed Limits on page 361](#).

[Application manual - Servo motor control](#).

6 Topic Motion

6.12.18. Kv 3

6.12.18. Kv 3

Parent

Kv 3 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

Kv_3

Description

Kv 3 defines the proportional gain in the speed loop for the torque specified in *torque 3*. This gain determines how fast the speed is regulated when the speed limit is exceeded.

Usage

Set *Kv 3* to the proportional gain you want for the third point (if more than two) in the speed limit graph (see [Example on page 359](#)).

Prerequisites

No. of Speed Limits must be set to 3 or higher, otherwise the value of *Kv 3* is not used.

Limitations

Kv 3 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100.

The default value is 0.5.

Related information

[torque 3 on page 364](#).

[No. of Speed Limits on page 361](#).

[Application manual - Servo motor control](#).

6.12.19. Kv 4

Parent

Kv 4 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

Kv_4

Description

Kv 4 defines the proportional gain in the speed loop for the torque specified in *torque 4*. This gain determines how fast the speed is regulated when the speed limit is exceeded.

Usage

Set *Kv 4* to the proportional gain you want for the fourth point (if more than three) in the speed limit graph (see [Example on page 359](#)).

Prerequisites

No. of Speed Limits must be set to 4 or higher, otherwise the value of *Kv 4* is not used.

Limitations

Kv 4 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100.

The default value is 0.5.

Related information

[torque 4 on page 365](#).

[No. of Speed Limits on page 361](#).

[Application manual - Servo motor control](#).

6 Topic Motion

6.12.20. Kv 5

6.12.20. Kv 5

Parent

Kv 5 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

Kv_5

Description

Kv 5 defines the proportional gain in the speed loop for the torque specified in *torque 5*. This gain determines how fast the speed is regulated when the speed limit is exceeded.

Usage

Set *Kv 5* to the proportional gain you want for the fifth point (if more than four) in the speed limit graph (see [Example on page 359](#)).

Prerequisites

No. of Speed Limits must be set to 5 or higher, otherwise the value of *Kv 5* is not used.

Limitations

Kv 5 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100.

The default value is 0.5.

Related information

[torque 5 on page 366](#).

[No. of Speed Limits on page 361](#).

[Application manual - Servo motor control](#).

6.12.21. Kv 6

Parent

Kv 6 belongs to the type *Force Master Control*, in the topic *Motion*.

Cfg name

Kv_6

Description

Kv 6 defines the proportional gain in the speed loop for the torque specified in *torque 6*. This gain determines how fast the speed is regulated when the speed limit is exceeded.

Usage

Set *Kv 6* to the proportional gain you want for the sixth point (if all six points are used) in the speed limit graph (see [Example on page 359](#)).

Prerequisites

No. of Speed Limits must be set to 6, otherwise the value of *Kv 6* is not used.

Limitations

Kv 6 can only be used if you have servo tools.

Allowed values

A number between 0.001 and 100.

The default value is 0.5.

Related information

[torque 6 on page 367](#).

[No. of Speed Limits on page 361](#).

[Application manual - Servo motor control](#).

6 Topic Motion

6.13.1. The Friction Compensation type

6.13 Type Friction Compensation

6.13.1. The Friction Compensation type

Overview

This section describes the type *Friction Compensation*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

CFRIC_BLOCK

Type description

Each set of parameters of the type *Friction Compensation* belongs to a joint (robot joint or external axis).

The parameters in *Friction Compensation* define what compensations should be made for the friction in the joint.

Limitation

Changing the parameter values in *Friction Compensation* is only useful if you have the RobotWare option *Advanced Shape Tuning*.

The type *Friction Compensation* equivalent to the type *Control Parameters*. The type *Control Parameters* is used by robot models IRB 1400 and IRB 1410, all other robot models use the type *Friction Compensation*. The parameters are the same however.

Related information

Application manual - Motion performance, chapter *Advanced Shape Tuning*.

6.13.2. Name**Parent**

Name belongs to the type *Friction Compensation*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the friction compensation.

Limitations

Name is only useful if you have the RobotWare option *Advanced Shape Tuning*.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.13.3. Friction FFW On

6.13.3. Friction FFW On

Parent

Friction FFW On belongs to the type *Friction Compensation*, in the topic *Motion*.

Cfg name

friction_ffw_on

Description

Friction FFW On determines if the RobotWare option *Advanced Shape Tuning* is active or not.

Usage

Set *Friction FFW On* to TRUE if you want to use *Advanced Shape Tuning*.

Limitations

Friction FFW On is only useful if you have the RobotWare option *Advanced Shape Tuning*.

Allowed values

TRUE or FALSE.

Related information

Application manual - Motion performance.

6.13.4. Friction FFW Level

Parent

Friction FFW Level belongs to the type *Friction Compensation*, in the topic *Motion*.

Cfg name

friction_ffw_level

Description

Friction FFW Level is set to the level of friction in the robot axis. By setting a value that closely corresponds to the real friction, and using the RobotWare option *Advanced Shape Tuning*, the friction effects can be compensated.

Usage

Friction effects can cause path deviations when performing advanced shapes. By compensating for the friction with the correct friction level value, these effects can be minimized.

Permanent adjustments to the friction level can be made with *Friction FFW Level*. The friction level can also be temporarily tuned with RAPID commands. For more information, see *Application manual - Motion performance*.

Limitations

Friction FFW Level is only useful if you have the RobotWare option *Advanced Shape Tuning*.

Allowed values

A decimal number between 0 and 15 (in Nm).

Related information

Application manual - Motion performance.

6 Topic Motion

6.13.5. Friction FFW Ramp

6.13.5. Friction FFW Ramp

Parent

Friction FFW Ramp belongs to the type *Friction Compensation*, in the topic *Motion*.

Cfg name

friction_ffw_ramp

Description

Friction FFW Ramp is set to the speed of the robot axis when the friction has reached the constant friction level defined in *Friction ffw level*. See illustration below.

Usage

Friction effects can cause path deviations when performing advanced shapes. *Friction FFW Ramp* is used when compensating for these friction effects.

Permanent adjustments to the friction ramp can be made with *Friction FFW Ramp*. The friction ramp can also be temporarily tuned with RAPID commands. For more information, see *Application manual - Motion performance*.

Limitations

Friction FFW Ramp is only useful if you have the RobotWare option *Advanced Shape Tuning*.

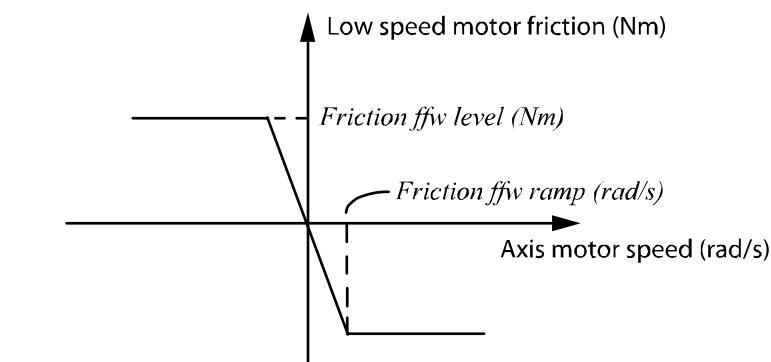
Allowed values

A number between 0.001 and 10 (in radians/second).

Related information

Application manual - Motion performance.

Illustration



en0300000278

6.14 Type Jog Parameters

6.14.1. The Jog Parameters type

Overview

This section describes the type *Jog Parameters*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic.

Cfg name

JOG_PARAMETERS

Type description

The *Jog Parameters* type contains parameters that define the step size in the different jogging modes when using incremental jogging with user-defined step.

Incremental movement

Incremental movement is used to adjust the position of the robot exactly. Each time the joystick is moved, the robot moves one step (one increment).

6 Topic Motion

6.14.2. Name

6.14.2. Name

Parent

Name belongs to the type *Jog Parameters*, in the topic *Motion*.

Cfg name

Name

Description

Name defines the name of the Jog parameters data.

Allowed values

A string with maximum 32 characters.

6.14.3. Configurable Linear Step Size

Parent

Configurable Linear Step Size belongs to the type *Jog Parameters*, in the topic *Motion*.

Cfg name

linear_step_size

Description

Configurable Linear Step Size defines the step size for user-defined incremental linear jogging.

Usage

Linear jogging step size is set in meters.

Allowed values

0 - 0.005 meters.

6 Topic Motion

6.14.4. Configurable Reorient Step Size

Parent

Configurable Reorient Step Size belongs to the type *Jog Parameters*, in the topic *Motion*.

Cfg name

reorient_step_size

Description

Configurable Reorient Step Size defines the step size for user-defined incremental reorient jogging.

Usage

Reorient jogging step size is set in radians.

Convert degrees to radians: `radians = (degrees/360) * (2*pi)`

Allowed values

0 - 0.009 radians.

6.14.5. Configurable Joint Step Size

Parent

Configurable Joint Step Size belongs to the type *Jog Parameters*, in the topic *Motion*.

Cfg name

joint_step_size

Description

Configurable Joint Step Size defines the step size for user-defined incremental axes jogging.

Usage

Axes jogging step size is set in radians.

Convert degrees to radians: radians = (degrees/360) * (2*pi)

Allowed values

0 - 0.0025 radians.

6 Topic Motion

6.15.1. The Joint type

6.15 Type Joint

6.15.1. The Joint type

Overview

This section describes the type *Joint* which belongs to the topic *Motion*. Each parameter is described in a separate information topic in this section.

Cfg name

JOINT

Type description

The *Joint* type contains parameters that define a joint.

Related information

[The Arm type on page 298.](#)

[The Measurement Channel type on page 434.](#)

6.15.2. Name**Parent**

Name belongs to the type *Joint*, in the topic *Motion*.

Cfg name

name

Description

Name defines the unique name to use for this joint.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.15.3. Logical Axis

6.15.3. Logical Axis

Parent

Logical Axis belongs to the type *Joint*, in the topic *Motion*.

Cfg name

logical_axis

Description

Logical Axis defines the axis number as seen by a RAPID program.

Usage

The value of *Logical Axis* is used by RAPID programs to identify individual axes in mechanical units.

Two mechanical units can have the same value set for *Logical Axis*, but then they cannot be activated at the same time by a RAPID program.

Robots from ABB normally use the values 1-6, while additional axes use 7-12.

Allowed values

A value between 1 and 12.

Related information

Application manual - Additional axes and stand alone controller.

6.15.4. Use Process

Parent

Use Process belongs to the type *Joint*, in the topic *Motion*.

Cfg name

use_process

Description

Use Process defines which process to use for this joint.

Usage

Use Process points to a process id defined by the parameter *Name* in the type *Process*.

The process can be used to define the joints behavior for either *Electronically Linked Motors* or *Spot Servo*.

Prerequisites

The additional axes must be configured before setting *Use Process*.

Limitations

Use Process is only used for additional axes.

Use Process is only useful if you have either of the RobotWare options *Electronically Linked Motors* or *Spot Servo*.

Allowed values

A string.

Related information

[Name on page 510](#).

[Application manual - Servo motor control](#).

6 Topic Motion

6.15.5. Lock Joint in Ipol

6.15.5. Lock Joint in Ipol

Parent

Lock Joint in Ipol belongs to the type *Joint*, in the topic *Motion*.

Cfg name

lock_joint_in_ipol

Description

A flag that locks the axis so it is not used in the path interpolation.

Usage

When setting *Lock Joint in Ipol* to TRUE, this axis will not be used for path interpolation.

When using *Electronically Linked Motors*, this parameter must be set to TRUE for the follower axis.

Prerequisites

The additional axes must be configured before setting *Lock Joint in Ipol*.

Limitations

Lock Joint in Ipol is only used for additional axes.

Allowed values

TRUE or FALSE.

Related information

Application manual - Servo motor control.

6.15.6. Follower to Joint

Parent

Follower to Joint belongs to the type *Joint*, in the topic *Motion*.

Cfg name

follower_to_joint

Description

When using *Electrically Linked Motors*, *Follower to Joint* defines which master axis this axis should follow.

Usage

When using *Electrically Linked Motors*, the follower axis has the *Follower to Joint* set to the name of the master axis.

Prerequisites

The additional axes must be configured before setting *Follower to Joint*.

Limitations

Follower to Joint is only used for external axes.

Follower to Joint is only useful if you have the RobotWare option *Electrically Linked Motors*.

Allowed values

A string.

Related information

Application manual - Servo motor control.

6 Topic Motion

6.16.1. The Lag Control Master 0 type

6.16 Type Lag Control Master 0

6.16.1. The Lag Control Master 0 type

Overview

This section describes the type *Lag Control Master 0*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

LCM0

Type description

The type *Lag Control Master 0* is normally used for control of axes without any dynamic model. This is the case for some additional axes.

For axes that have a dynamic model, *Lag Control Master 0* is only used in exceptional cases.

6.16.2. Name

Parent

Name belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

name

Description

The name of the *Lag Control Master 0*.

Usage

Name is used to reference a *Lag Control Master 0* from the parameter *Normal Control Master* in the type *Joint*.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.16.3. Kp, Gain Position Loop

6.16.3. Kp, Gain Position Loop

Parent

Kp, Gain Position Loop belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

Kp

Description

Proportional gain in the position control loop.

Usage

The higher the value of *Kp, Gain Position Loop*, the better tracking and disturbance rejection.
If the position control overshoots, decrease *Kp, Gain Position Loop*.

Allowed values

A numeric value between 0 and 1000 (1/s).

6.16.4. Kv, Gain Speed Loop

Parent

Kv, Gain Speed Loop belongs to the type *Lag Control Master*, in the topic *Motion*.

Cfg name

Kv

Description

Proportional gain in the speed regulation loop.

Usage

The higher the value of *Kv, Gain Speed Loop*, the better tracking and disturbance rejection.
If the level of oscillation or noise is too high, decrease *Kv, Gain Speed Loop*.

Allowed values

A numeric value between 0 and 100 (Nms/rad).

6 Topic Motion

6.16.5. Ti Integration Time Speed Loop

Parent

Ti Integration Time Speed Loop belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

Ti

Description

Integration time in the speed regulation loop.

Usage

The lower the value of *Ti Integration Time Speed Loop*, the better tracking and disturbance rejection.

If the level of oscillation or noise is too high, increase *Ti Integration Time Speed Loop*.

Allowed values

A numeric value between 0 and 10 (seconds).

The default value is 10 seconds.

6.16.6. Forced Control Active

Parent

Forced Control Active belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

use_inpos_forced_control

Description

Determines whether forced control is active for this joint.

Usage

Forced Control can be used if the last part of the movement before a fine point is to slow. The function changes the parameters *Forced Factor for Kp* and *Forced Factor for Ki* in the last part of the movement.

Note! Wrongly used Forced Control (too high force factors) might impair the movement with oscillations.

If *Forced Control Active* is set to True, *Affects forced ctrl* in type *Supervision* should normally also be set to True for this joint.

Allowed values

True or False.

Related information

[Forced Factor for Kp on page 402](#).

[Forced Factor for Ki on page 403](#).

[Affects Forced Control on page 590](#), in the type *Supervision*.

[Application manual - Additional axes and stand alone controller](#).

6 Topic Motion

6.16.7. Forced Factor for Kp

Parent

Forced Factor for Kp belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

Kp_forced_factor

Description

The forced factor for K_p , if forced gain control is active.

Usage

Forced Factor for Kp defines the gain increase factor.

A typical value is 2.

Allowed values

A numeric value between 1 and 4.

6.16.8. Forced Factor for Ki

Parent

Forced Factor for Ki belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

Ki_forced_factor

Description

The forced factor for K_i , if forced gain control is active.

Usage

Forced Factor for Ki defines the gain increase factor.

K_i equals K_v/T_i , integral gain.

A typical value is 2.

Allowed values

A numeric value between 1 and 4.

6 Topic Motion

6.16.9. Raise Time for Kp

6.16.9. Raise Time for Kp

Parent

Raise Time for Kp belongs to the type *Lag Control Master*, in the topic *Motion*.

Cfg name

Kp_raise_time

Description

Defines the raise time for forced *Kp*.

Usage

To avoid transient effects, *Kp* must be increased slowly over a period of time. This period is defined by *Raise Time for Kp*.

A typical value is 0.2.

Allowed values

A numeric value between 0.002 and 0.5 seconds.

6.16.10. Notch Filter Active

Parent

Notch Filter Active belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

notch_filter_active

Description

Defines if the notch filter is activated or not.

Usage

Notch filters are only used in certain arc welding applications.

Notch Filter Active can be used to avoid interference between the additional axis and the weaving frequency that can cause vibrations in the additional axis.

Allowed values

True or False.

Related information

Application manual - Additional axes and stand alone controller.

6 Topic Motion

6.16.11. Notch Filter Frequency

6.16.11. Notch Filter Frequency

Parent

Notch Filter Frequency belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

notch_filter_frequency

Description

Defines the frequency of speed variation for the notch filter.

Usage

Notch Filter Frequency is used when *Notch Auto Mode* is set to False. **Note!** It is always recommended to use *Notch Auto Mode* (set to True) if the notch function is needed.

A typical value is $2 * \text{Weld speed}/\text{Weave length}$.

Limitations

This parameter is only used when *Notch Auto Mode* is set to False.

Allowed values

A numeric value between 1 and 100 (Hz).

Related information

[Notch Auto Mode on page 408](#).

6.16.12. Notch Filter Width

Parent

Notch Filter Width belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

notch_filter_width

Description

Defines the width of the notch filter.

Usage

A higher value increases the width but can also have a negative effect on the performance (response) of the additional axis. Recommended value is 0.2.

Allowed values

A numeric value between 0.01 and 0.4.

6 Topic Motion

6.16.13. Notch Auto Mode

Parent

Notch Auto Mode belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

notch_auto_mode

Description

Defines if the notch filter frequency will adjust automatically to the weave frequency.

Usage

If *Notch Auto Mode* is set to True, the notch filter frequency will automatically adjust to the weave frequency according to the formula: $\text{Notch Filter Frequency} = 2 * \text{Weld speed}/\text{Weave length}$.

It is recommended to always set *Notch Auto Mode* to True if a notch filter function is needed.

Allowed values

True or False.

6.16.14. Auto No Weave Frequency

Parent

Auto No Weave Frequency belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

notch_auto_no_weave_freq

Description

Defines the default frequency for the auto mode notch filter.

Usage

The default value should only be changed by advanced programmers.

Allowed values

A numeric value between 1 and 100 Hz.

Related information

Notch Auto Mode on page 408.

6 Topic Motion

6.16.15. Auto Min Frequency

Parent

Auto Min Frequency belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

notch_auto_min_frequency

Description

Minimum frequency for the on line notch filter.

Usage

Auto Min Frequency defines the minimum notch filter frequency when *Notch Auto Mode* is set to True.

The default value should only be changed by advanced programmers.

Limitations

This parameter is only used when *Notch Auto Mode* is set to True.

Allowed values

A numeric value between 1 and 100 Hz.

Related information

[Notch Auto Mode on page 408](#).

6.16.16. Auto Max Relative Change

Parent

Auto Max Relative Change belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

notch_auto_max_rel_change

Description

A factor that sets the maximum instant change in the notch filter when *Notch Auto Mode* is set to True.

Usage

If the instant change of the notch filter should be limited to 10%, set *Auto Max Relative Change* to 0.1.

The default value should only be changed by advanced programmers.

Limitations

This parameter is only used when *Notch Auto Mode* is set to True.

Allowed values

A numeric value between 0 and 10.

Related information

[Notch Auto Mode on page 408](#).

6 Topic Motion

6.16.17. FFW Mode

Parent

FFW Mode belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

ffw_mode

Description

FFW Mode defines the control type to use, i.e. if feed forward should be used.

Usage

To regulate the position, you can:

- use only the desired position as reference.
 - in addition to the position, use feed forward of the current speed value.
 - in addition to the position, use feed forward of the current speed and torque values.
-

Allowed values

FFW Mode can have the following values:

Value:	Name:	Description:
0	No	The controller is driven by the position error (lag). Because a relatively large lag is needed to move the axis, the position error can be large.
1	Spd	The controller receives information about the desired speed of the axis. As a result, the position lag is greatly reduced compared to the No configuration. For this reason, Spd is the recommended configuration.
2	Trq	The controller uses the desired speed and acceleration of the axis to calculate the desired motor torque. This requires knowledge of the mass moment of inertia of the axis, which must be supplied by the user. For this reason this configuration is more difficult to tune. It is only recommended for experienced users.

The default value is 0. Recommended value is 1.

Related information

Application manual - Additional axes and stand alone controller.

6.16.18. Bandwidth

Parent

Bandwidth belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

bandwidth

Description

Defines the controller bandwidth when *FFW Mode* is set to 1 or 2.

Usage

A high bandwidth value gives faster control but increases risk of vibrations and overshoot.
The default value is recommended, but can be reduced if undesired vibrations occur.

Allowed values

A value between 3 and 40. Default value is 25.

Related information

FFW Mode on page 412.

6 Topic Motion

6.16.19. Df

6.16.19. Df

Parent

Df belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

resonance_frequency

Description

Reduces oscillations.

Usage

Df can be used to damp oscillations of the axis due to mechanical resonance.

Initially *Df* should be left at its default value. It can be adjusted once the other controller parameters have been fixed (*Kv Gain Speed Loop*, *Kp Gain Position Speed Loop*, *Ti Integration Time Speed Loop*, and *Inertia*).

Df is only used when *FFW Mode* is set to 2.

Allowed values

A value between 2 and 100. Default value is 100.

Related information

[*FFW Mode on page 412.*](#)

[*Kp, Gain Position Loop on page 398.*](#)

[*Kv, Gain Speed Loop on page 399.*](#)

[*Ti Integration Time Speed Loop on page 400.*](#)

[*Inertia on page 417.*](#)

6.16.20. Dw**Parent**

Dw belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

resonance_damping

Description

Can reduce oscillations further when *Df* is set.

Usage

The default value of *Dw* is recommended.

Allowed values

A value between 0.002 to 1. Default value is 0.01.

Related information

Df on page 414.

6 Topic Motion

6.16.21. Delay

6.16.21. Delay

Parent

Delay belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

delay_time

Description

Reduces overshoot.

Usage

Delay can be used when *Df* is set, to reduce overshoot but it impairs the axis coordination when increased.

The default value of *Delay* should normally not be changed.

Allowed values

A value between 0.0 and 0.02. Default value is 0.004.

Related information

Df on page 414.

Dw on page 415.

6.16.22. Inertia

Parent

Inertia belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

inertia

Description

Defines the external axis' inertia (if rotation) or mass (if translation).

Usage

Inertia is used for calculating the torque when *FFW Mode* is set to 2.

Allowed values

A value between 0.0 and 10,000.

Related information

FFW Mode on page 412.

6 Topic Motion

6.16.23. K Soft Max Factor

6.16.23. K Soft Max Factor

Parent

K Soft Max Factor belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

soft_servo_K_max_factor

Description

Determines the value of the product *Kp Gain Position Loop* * *Kv Gain Speed Loop* when the soft servo is used with softness 0%.

Usage

K Soft Max Factor should be in the range 0.1 - 2.0 (default 1.0). When the soft servo is activated with 0% softness, the control parameters *Kp Gain Position Loop* (*Kp*) and *Kv Gain Speed Loop* (*Kv*) will be tuned such that $Kp \cdot Kv = (Kp \cdot Kv)_{\text{normal}} \cdot K \text{ Soft Max Factor}$, where $(Kp \cdot Kv)_{\text{normal}}$ is the product of *Kp* and *Kv* during normal operation.

Allowed values

A value between 0.1 and 2.0. Default value is 1.0.

Related information

Kp, Gain Position Loop on page 398.

Kv, Gain Speed Loop on page 399.

6.16.24. K Soft Min Factor

Parent

K Soft Min Factor belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

soft_servo_K_min_factor

Description

Determines the value of the product *Kp Gain Position Loop* * *Kv Gain Speed Loop* if the soft servo is used with softness 100%.

Usage

K Soft Min Factor should be in the range 0.001 - 0.1 (default 0.01). When the soft servo is activated with 100% softness, the control parameters *Kp Gain Position Loop* (*Kp*) and *Kv Gain Speed Loop* (*Kv*) are tuned such that $Kp \cdot Kv = (Kp \cdot Kv)_{normal} \cdot K Soft Min Factor$.

Allowed values

A value between 0.001 and 0.1. Default value is 0.01.

Related information

[Kp, Gain Position Loop on page 398](#).

[Kv, Gain Speed Loop on page 399](#).

6 Topic Motion

6.16.25. Kp/Kv Ratio Factor

Parent

Kp/Kv Ratio Factor belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

soft_servo_Kp_Kv_ratio_factor

Description

Defines the factor used to tune the *Kp Gain Position Loop/Kv Gain Speed Loop* ratio.

Usage

Kp/Kv Ratio Factor is used to alter the *Kp Gain Position Loop/Kv Gain Speed Loop* ratio during soft servo. *Kp/Kv Ratio Factor* should be in the range 0.1 - 1.0 (default 1.0). In soft servo mode, *Kp* and *Kv* are tuned such that $Kp/Kv = (Kp/Kv)_{\text{normal}} * Kp/Kv \text{ Ratio Factor}$.

Allowed values

A value between 0.1 and 1.0.

Related information

[Kp, Gain Position Loop on page 398](#).

[Kv, Gain Speed Loop on page 399](#).

6.16.26. Ramp Time

Parent

Ramp Time belongs to the type *Lag Control Master 0*, in the topic *Motion*.

Cfg name

soft_servo_t_ramp

Description

Defines the default Soft Servo ramp time.

Usage

Ramp Time is used to define the default time for activation of the soft servo.

Allowed values

A value between 0.01 and 0.5. Default value is 0.05.

6 Topic Motion

6.17.1. The Linked M Process type

6.17 Type Linked M Process

6.17.1. The Linked M Process type

Overview

This section describes the type *Linked M Process*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

LINKED_M_PROCESS

Type description

A *Linked M Process* contains information about alignments between the master axis and the follower axis for *Electronically Linked Motors*.

Limitations

Linked M Process is only useful if you have the RobotWare option *Electronically Linked Motors*.

Related information

Application manual - Servo motor control, chapter *Electronically Linked Motors*.

6.17.2. Name

Parent

Name belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

name

Description

Name defines the identity of the linked motor process.

Usage

The *Name* is used when referencing the linked motor process.

The linked motor process defines the behavior of a joint for *Electronically Linked Motors*.

Limitations

This parameter is only useful if you have the RobotWare option *Electronically Linked Motors*.

Allowed values

A string.

6 Topic Motion

6.17.3. Offset Adjust Delay Time

Parent

Offset Adjust Delay Time belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

offset_adj_delay_time

Description

Offset Adjust Delay Time defines the time delay from control on until the follower axis starts to follow its master axis.

Usage

When using *Electronically Linked Motors*, you might want to give the master axis some time to stabilize before the follower axis starts following.

Limitations

Offset Adjust Delay Time is only useful if you have the RobotWare option *Electronically Linked Motors*.

Allowed values

A value between 0 and 2, specifying the delay in seconds.

Default value: 0.2

6.17.4. Max Follower Offset

Parent

Max Follower Offset belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

max_offset

Description

Max Follower Offset defines the maximum allowed difference in position between the master and the follower axis.

Usage

If the follower offset exceeds the *Max Follower Offset*, emergency stop is activated and automatic offset adjustment is prohibited.

Limitations

Max Follower Offset is only useful if you have the RobotWare option *Electronically Linked Motors*.

Allowed values

A value between 0 and 5, specifying the maximum offset in radians (for rotational axes) or meters (for linear axes) on the arm side.

Default value: 0.05.

6 Topic Motion

6.17.5. Max Offset Speed

Parent

Max Offset Speed belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

max_offset_speed

Description

Max Offset Speed defines the maximum allowed difference in speed between the master and the follower axis.

Usage

If the speed difference exceeds the *Max Offset Speed*, emergency stop is activated and automatic offset adjustment is prohibited.

Limitations

Max Offset Speed is only useful if you have the RobotWare option *Electronically Linked Motors*.

Allowed values

A value between 0 and 1000, specifying the maximum difference in rad/s (for rotational axes) or m/s (for linear axes) on the arm side.

Default value: 0.05.

6.17.6. Offset Speed Ratio

Parent

Offset Speed Ratio belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

offset_speed_ratio

Description

Offset Speed Ratio defines how large a part of the *Max Offset Speed* can be used to compensate for position error.

Usage

Offset Speed Ratio multiplied by *Max Offset Speed* is the highest speed by which the position offset is reduced.

Limitations

Offset Speed Ratio is only useful if you have the RobotWare option *Electronically Linked Motors*.

Allowed values

A value between 0 and 1. The value has no unit since it is a multiplication factor.

Default value: 0.33.

Related information

[Max Offset Speed on page 426](#).

6 Topic Motion

6.17.7. Ramp Time

6.17.7. Ramp Time

Parent

Ramp Time belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

ramp_time

Description

Ramp Time defines the acceleration up to *Max Offset Speed*.

Usage

The proportion constant for position regulation is ramped from zero up to its final value (*Master Follower kp*) during *Ramp Time*.

Limitations

Ramp Time is only useful if you have the RobotWare option *Electronically Linked Motors*.

Allowed values

A value between 0.01 and 10, specifying the time in seconds.

Default value: 0.05

Related information

[*Master Follower kp on page 429*](#).

[*Max Offset Speed on page 426*](#).

6.17.8. Master Follower kp

Parent

Master Follower kp belongs to the type *Linked M Process*, in the topic *Motion*.

Cfg name

kp_offset

Description

Master Follower kp is the proportion constant for position regulation.

Usage

Master Follower kp determines how fast the position error is compensated. If the value is too low, the compensation will be slow. If the value is to large, the compensation will be unstable.

Limitations

Master Follower kp is only useful if you have the RobotWare option *Electronically Linked Motors*.

Allowed values

A value between 0 and 5 (unit is 1/s).

Default value: 0.05.

6 Topic Motion

6.18.1. The Mains type

6.18 Type Mains

6.18.1. The Mains type

Overview

This section describes the type *Mains*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

MAINS

Type description

The type *Mains* defines the drive system's mains power tolerance. The parameters of the Mains type have nominal values.

The parameters of the type *Mains* can be used to improve the robot's performance by adjusting them according to the robot's actual installation.

CAUTION!



Parameter settings outside the range of the robot's installation may negatively affect the robot's performance.

Related information

How to optimize drive system parameters on page 286.

6.18.2. Name

Parent

Name belongs to the type *Mains*, in the topic *Motion*.

Cfg name

name

Description

Name specifies the name of the mains tolerance setting it belongs to.

Allowed values

A string with maximum 32 characters, specifying the name.

Related information

How to optimize drive system parameters on page 286.

6 Topic Motion

6.18.3. Mains Tolerance Min

Parent

Mains Tolerance Min belongs to the type *Mains*, in the topic *Motion*.

Cfg name

u_tolerance_min

Description

Mains Tolerance Min specifies the minimum value of the mains tolerance as a percentage. The value is set to -15% on delivery. If the minimum tolerance is less than 15%, the cycle time can be improved by changing the parameter.

Allowed values

A value between -1 and +1 (equals -100% and 100%).

The default value is -0.15 (equals -15%).

Related information

[How to optimize drive system parameters on page 286.](#)

6.18.4. Mains Tolerance Max

Parent

Mains Tolerance Max belongs to the type *Mains*, in the topic *Motion*.

Cfg name

u_tolerance_max

Description

Mains Tolerance Max specifies the maximum value of the mains tolerance. Its default value is 0.1 (10%). This value should not be increased since the equipment is rated for this maximum mains tolerance and might be damaged if the voltage is increased.

Limitations

This parameter should not be changed as that could cause damage to the equipment.

Allowed values

The default value is 0.1.

Related information

How to optimize drive system parameters on page 286.

6 Topic Motion

6.19.1. The Measurement Channel type

6.19 Type Measurement Channel

6.19.1. The Measurement Channel type

Overview

This section describes the type *Measurement Channel* which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

MEASUREMENT_CHANNEL

Type description

The type *Measurement Channel* describes which channel is used to send measurement data from the axis computer to the controller.

Non-editable parameters

The following parameters are visible but not editable in the software configuration tools:

- *Max Normalized Input Level*
- *Min Normalized Input Level*

As a consequence, the above parameters are not described in the manual.

6.19.2. Name**Parent**

Name belongs to the type *Measurement Channel*, in the topic *Motion*.

Cfg name

name

Description

Name defines the axis computer's channel name.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.19.3. Use Measurement Board Type

Parent

Use Measurement Board Type belongs to the type *Measurement Channel*, in the topic *Motion*.

Cfg name

use_measurement_board_type

Description

Use Measurement Board Type defines which type of measurement board is used.

Usage

The type *Measurement Board Type* defines the measurement board data.

Allowed values

A string with maximum 32 characters.

6.19.4. Disconnect at Deactivate

Parent

Disconnect at Deactivate belongs to the type *Measurement Channel*, in the topic *Motion*.

Cfg name

disconnect_at_deactivate

Description

Disconnect at Deactivate defines if the channel should be deactivated when the mechanical unit is deactivated.

Usage

Set *Disconnect at Deactivate* to True to avoid error reports when the resolver is disconnected, for instance when switching between tools.

Allowed values

True or False.

Default value is False.

6 Topic Motion

6.19.5. Measurement Link

Parent

Measurement Link belongs to the type *Measurement Channel*, in the topic *Motion*.

Cfg name

measurement_link

Description

An axis resolver is connected to a Serial Measurement Board (SMB). The SMB communicates with the axis computer via a serial measurement link.

Measurement Link defines the number of the measurement link.

Usage

There are two contacts on the axis computer marked Measurement link 1 and Measurement link 2.

An ABB robot is normally connected to link 1.

Allowed values

1 or 2.

Default value is 1.

6.19.6. Board Position

Parent

Board Position belongs to the type *Measurement Channel*, in the topic *Motion*.

Cfg name

board_position

Description

Board Position defines the position number of the board used for the measurement system.

Usage

The value of *Board Position* defines the physical position of the board on the measurement link. Board position one is closest to the axis computer.

Allowed values

An integer value between 1 and 2.

Default value is 1.

6 Topic Motion

6.20.1. The Mechanical Unit type

6.20 Type Mechanical Unit

6.20.1. The Mechanical Unit type

Overview

This section describes the type *Mechanical Unit* which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

MECHANICAL_UNIT

Type description

The *Mechanical Unit* type describes the common parameters for a mechanical unit. There is one set of parameters for each mechanical unit.

This type is only possible to edit for additional axes, not for robots delivered from ABB.

Non-editable parameters

The following parameter is visible but not editable in the software configuration tools:

- *Use Run Enable*

As a consequence, the above parameter is not described in the manual.

Related information

Application manual - Additional axes and stand alone controller.

6.20.2. Name**Parent**

Name belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name for the mechanical unit.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.20.3. Use Activation Relay

Parent

Use Activation Relay belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

use_activation_relay

Description

Use Activation Relay defines the Id name for the activation relay.

Usage

Use Activation Relay points out a relay that will be activated or deactivated when the mechanical unit is activated or deactivated.

More information can be found in *Technical reference manual - RAPID Instructions, Functions and Data types* under the instructions ActUnit/DeactUnit.

Allowed values

A string with maximum 32 characters.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types.

6.20.4. Use Brake Relay

Parent

Use Brake Relay belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

use_brake_relay

Description

Use Brake Relay defines the Id name for the brake relay.

Usage

Use Brake Relay points out what brake relay will be activated or deactivated when the mechanical unit goes to state control on or control off.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.20.5. Use Connection Relay

6.20.5. Use Connection Relay

Parent

Use Connection Relay belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

use_connection_relay

Description

Use Connection Relay defines the Id name for the connection relay.

Usage

Use Connection Relay points out a relay that must be activated when the mechanical unit is activated.

Allowed values

A string with maximum 32 characters.

6.20.6. Use Robot

Parent

Use Robot belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

use_robot

Description

Use Robot defines which robot is part of the mechanical unit.

Usage

The robot is defined in the type *Robot*.

Allowed values

A string with maximum 32 characters.

Related information

[Name on page 517](#), of the type *Robot*.

6 Topic Motion

6.20.7. Use Single 1, 2, 3, 4, 5, 6

6.20.7. Use Single 1, 2, 3, 4, 5, 6

Parent

Use Single 1, Use Single 2, Use Single 3, Use Single 4, Use Single 5, and Use Single 6 belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg names

```
use_single_0  
use_single_1  
use_single_2  
use_single_3  
use_single_4  
use_single_5
```

Description

Use Single defines which singles are part of the mechanical unit.

Usage

The mechanical unit can have six singles, *Use Single 1, Use Single 2, Use Single 3, Use Single 4, Use Single 5, and Use Single 6*. The singles are defined in the type *Single*.

Allowed values

Each single value is a string with maximum 32 characters.

Related information

Name on page 559, in the type *Single*.

6.20.8. Allow Move of User Frame

Parent

Allow Move of User Frame belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

allow_move_of_user_frame

Description

Allow Move of User Frame defines if a robot or single is allowed to move a user frame.

Usage

A user frame can be moved by a robot or a single that is part of the mechanical unit. Set *Allow Move of User Frame* to True to allow a robot or single to move a user frame.

Note that the definition of the work object must allow it to be moved, see `wobjdata` (`ufprog` and `ufmec`) in *Technical reference manual - RAPID Instructions, Functions and Data types*.

Allowed values

True or False.

Related information

Technical reference manual - RAPID Instructions, Functions and Data types

6 Topic Motion

6.20.9. Activate at Startup

6.20.9. Activate at Startup

Parent

Activate at Startup belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

activate_at_start_up

Description

Activate at Startup defines if the mechanical unit should be activated at start-up.

Usage

Set the value to True to activate the mechanical unit at startup.

Allowed values

True or False.

6.20.10. Deactivation Forbidden

Parent

Deactivation Forbidden belongs to the type *Mechanical Unit*, in the topic *Motion*.

Cfg name

deactivation_forbidden

Description

Deactivation Forbidden defines if the mechanical unit is allowed to be deactivated.

Usage

Set *Deactivation Forbidden* to False if the mechanical unit should be allowed to be deactivated.

Robots from ABB always has the value set to True. They should not be deactivated. The value False is only used for additional axes that should be possible to deactivate.

Allowed values

True or False.

6 Topic Motion

6.21.1. The Motion Planner type

6.21 Type Motion Planner

6.21.1. The Motion Planner type

Overview

This section describes the type *Motion Planner*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

NOTE!

When several task programs are run in synchronized mode, the movements of all their mechanical unit groups are calculated by the same motion planner. It is then the first set of parameters of the type *Motion Planner* that is used.

Cfg name

MOTION_PLANNER

Type description

A motion planner is a process on the controller that calculates how mechanical units shall move. A controller that handles more than one robot also has more than one motion planner. Each mechanical unit group has its own motion planner.

Limitations

Unless the option *MultiMove* is installed, there can only be one motion planner configuration.

Related information

Application manual - MultiMove.

6.21.2. Name

Parent

Name belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

name

Description

The name of the motion planner.

Usage

This is the public identity of the motion planner. It is used by the parameter *Use Motion Planner* in the type *Mechanical Unit Group*.

Allowed values

A string with maximum 32 characters. The name must not be changed!

Related information

The Mechanical Unit Group type on page 79 in the topic *Controller*.

6 Topic Motion

6.21.3. Brake on Time

Parent

Brake on Time belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

brake_on_timeout

Description

Brake on Time is used to delay the use of brakes when the robot is waiting to move. It defines the time from when the robot stops to when the mechanical brakes are activated.

 **NOTE!**

The brake on time value should be kept high to maintain the reliability of the servo at high level.

Allowed values

A value between 0.3 to 3,600,000, specifying the time in seconds.

6.21.4. Dynamic Resolution

Parent

Dynamic Resolution belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

dynamic_resolution

Limitation

Dynamic Resolution is optimized for the system at delivery. It should normally not be changed.

Allowed values

A predefined value, specified in seconds.

6 Topic Motion

6.21.5. Path Resolution

6.21.5. Path Resolution

Parent

Path Resolution belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

path_resolution

Description

The parameter corresponds in some sense to the distance between two points in the path. Increasing path resolution means increasing the distance, which leads to a decrease in the resolution of the path!

Increasing path resolution is a way to deal with robot installations that have external axes with long deceleration times due to high CPU load. In such applications the warning "50082 Deceleration limit" can be reported, simultaneously generating a quick-stop. Increasing the path resolution solves the problem.

Prerequisites

It is important to set the path resolution value as low as possible in order to achieve a high path resolution at high speed. Keeping the path resolution low can also give shorter cycle times if the cycle contains many stop points and the move instructions following these stop points have low speeds.

Usage

Path Resolution might require tuning when:

- The acceleration value of an external axis (and the robot) is decreased using the first parameter of the RAPID instruction AccSet.
- The acceleration derivative is decreased using the second parameter of the RAPID instruction AccSet.
- The speed is increased.
- The distances between closely programmed positions are decreased.
- The number of simultaneously controlled axes are increased.
- Using coordinated interpolation.
- Using Weldguide.
- Using the option *Conveyor Tracking*.
- Using RAPID controlled path correction.
- Using multitasking with computationally demanding RAPID programs.
- Reorienting with a small or no TCP movement.

Allowed values

A value between 0.1667 to 6.00, specifying the resolution in seconds.

Additional information

There is also a RAPID instruction named `PathResol` which affects the resolution of the path.

Continues on next page

Continued

Related information

Technical reference manual - RAPID overview.

Application manual - Motion coordination and supervision.

6 Topic Motion

6.21.6. Queue Time

Parent

Queue Time belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

std_servo_queue_time

Description

Increasing *Queue Time* makes the system more tolerant to uneven CPU loads.

 **NOTE!**

The real queue time is a multiple of a sample time related to dynamic resolution. If the parameter value is not an even multiple of the dynamic resolution, the controller will automatically use a queue time as close as possible to the given value.

Allowed values

A value between 0.004032 to 0.290304, specifying the time in seconds.

Additional information

A drawback with increasing the queue time is that the robot reacts more slowly when jogging and when stopping a program execution. However, the emergency brake is not affected. The accuracy of a sensor process, e.g. WeldGuide and Conveyor tracking, may also be affected.

6.21.7. Teach Mode Max Speed

Parent

Teach Mode Max Speed belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

teach_mode_max_speed

Description

Teach Mode Max Speed can be used to set the maximum speed in manual mode to less than the default value 0.25 m/s.

Allowed values

A value between 0.010 to 0.250, specifying the speed in meter per seconds.

The default value is 0.25 m/s.

6 Topic Motion

6.21.8. Process Update Time

6.21.8. Process Update Time

Parent

Process Update Time belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

process_linearization_time

Description

Process Update Time determines how often the process path information is calculated. This information is used for path following in Conveyor tracking, WeldGuide and Rapid Weave, for example.

Usage

Decreasing the process update time improves accuracy but also increases CPU load.
Increasing the parameter decreases the CPU load.

Limitations

When running programs in which the manipulator is moving at high speed, the parameter value should be kept small in order to get the best performance. When the manipulator is moving slowly, the process update time is not critical.

Allowed values

A value between 0.012096 to 1.93536, specifying the time in seconds.

6.21.9. Prefetch Time

Parent

Prefetch Time belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

ipol_prefetch_time

Description

Prefetch Time affects the point in time at which the controller starts to plan for the motion through a corner zone. If the planning time is too short, the corner zone becomes a fine point. This generates a warning called “50024 Corner path failure”.

Usage

If the planning time is too short because of high CPU load, increasing the parameter value may solve the problem. However, it will not solve the problem when it is caused by too many corner zones placed very close together or by incorrect use of instructions, e.g. a corner zone followed by a `WaitDI` instruction. Normally, *Prefetch Time* should only be increased when the corner zone is really needed in the application. When it is not really needed, change the corner zone to a fine point.

Limitations

There is a drawback when increasing the parameter. The difference between the position of the executed RAPID instruction and the current position of the manipulator will increase. This means that after pressing stop during program execution, the program counter on the teach pendant unit may show an instruction that has not yet affected the manipulator. When starting again, the manipulator continues along the original path.

Allowed values

A value between 0 to 10, specifying the time in seconds.

6 Topic Motion

6.21.10. Event Preset Time

Parent

Event Preset Time belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

event_preset_time

Description

Event Preset Time is used to delay the robot to make it possible to activate/control external equipment in advance. This is to compensate for the internal delay of the equipment.

Usage

Adjustment for the internal delay of the equipment can be made with the instruction `TriggEquip`. This takes advantage of the delay between the RAPID commands and the robot movement. In this way an output signal can be set up to about 70 ms in advance. If the delay of the equipment is longer than 70 ms, then *Event Preset Time* must be used to increase the delay of the robot movement.

Configure *Event Preset Time* to the longest equipment delay time needed (if more than 70 ms).

Limitations

Event Preset Time is only useful if you have the RobotWare option *Fixed Position Events* and use the instruction `TriggEquip`.

Allowed values

A value between 0 and 0.5, specifying the time in seconds.

Additional information

Remember that when using *Event Preset Time*, the start of the robot is delayed and the performance of WeldGuide and conveyor will be decreased.

Related information

Application manual - Motion functions and events.

Example

If you use *Fixed Position Event* with the following RAPID instructions, you should configure *Event Preset Time* to 0.2 seconds (the maximum delay required by `TriggEquip`)

```
TriggEquip gunon, 10, 0.2 \DOp:=gun, 1;  
TriggL p1, v500, gunon, z50, gun1;
```

6.21.11. CPU Load Equalization

Parent

CPU Load Equalization belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

cpu_load_equalization

Description

CPU Load Equalization affects the CPU load in terms of peak load versus average load.

Usage

When there is a CPU load problem, indicated for example by error message “50082 Deceleration limit”, one solution could be to use *CPU Load Equalization* to distribute the CPU load over time in some other way. Sometimes a higher peak load can be acceptable, as long as it occurs at a favorable moment in time. Try changing CPU equalization both upwards and downwards to find the optimal value.

Allowed values

An integer value between 1 and 10.

6 Topic Motion

6.21.12. Use Motion Supervision

Parent

Use Motion Supervision belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

use_motion_sup

Description

Use Motion Supervision defines which set of motion supervision parameters to be used for this motion planner.

Usage

Motion supervision is used to activate, deactivate or adjust the collision detection functionality. For detailed information about collision detection, see the *Application manual - Motion coordination and supervision*, chapter *Collision Detection*.

Allowed values

A string with maximum 32 characters.

Related information

[The Motion Supervision type on page 470.](#)

[Application manual - Motion coordination and supervision.](#)

6.21.13. Motion Supervision Permanent Off

Parent

Motion Supervision Permanent Off belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

motion_sup_permanent_off

Description

Motion Supervision Permanent Off is used to turn off all motion supervision to save CPU power.

Allowed values

YES

NO

6 Topic Motion

6.21.14. Motion Supervision Max Level

Parent

Motion Supervision Max Level belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

motion_sup_max_level

Description

The maximum allowed supervision level, both for program execution and jogging.

Usage

Motion Supervision Max Level stops the operator from tuning the supervision level to values that are too high.

The supervision level for program execution is a combination of the parameter *Path Collision Detection Level* and a tuning value set with the RAPID instruction `MotionSup`. *Motion Supervision Max Level* is a maximum limit for this combined value.

Limitations

Changing this parameter only affects the system if the option *Collision Detection* is installed.

Allowed values

An integer in the interval 10 to 500 (percent).

The default value is 300.

Related information

[Path Collision Detection Level on page 474](#).

[Application manual - Motion coordination and supervision](#).

Example

Motion Supervision Max Level is set to 300.

Path Collision Detection Level is set to 250.

A RAPID program uses the instruction `MotionSup` to tune the supervision level with 200%.

Normally this would lead to a supervision level of 500% ($2.5 * 2 = 5$), but since *Motion Supervision Max Level* is 300, the supervision level will not exceed 300%.

6.21.15. Remove Corner Path Warning

Parent

Remove Corner Path Warning belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

remove_corner_path_warning

Description

Remove Corner Path Warning is used to disable the corner path failure warnings. Corner warnings will still be executed as fine points but the warning will not be shown in the event log.

Usage

The warning "50024 Corner Path Failure" occurs when RAPID program execution does not provide a new Move instruction while the robot is entering a corner zone. This may be due to a programming oversight or an explicit desire of the programmer.

Allowed values

YES

NO

6 Topic Motion

6.21.16. Time Event Supervision

Parent

Time Event Supervision belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

require_event_accuracy

Description

Time Event Supervision is used to detect if a programmed event can be accurately positioned or not. If not, the system will stop and display a warning.

Usage

If the event cannot be accurately positioned, suggested program modifications are to either lower the programmed speed or to increase the distance between the start of the segment and the desired event position.

Allowed values

YES or NO

6.21.17. High Interpolation Priority

Parent

High Interpolation Priority belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

high_interpolation_priority

Description

High Interpolation Priority is used to allow the system to temporarily increase the priority of the path planning in critical situations.

Usage

When the warning "50082 Deceleration limit" occurs at installations, this parameter can be useful. The parameters *Path Resolution* and *CPU Load Equalization* might also be useful in this situation.

NOTE!



Using *High Interpolation Priority* might affect the performance of the application, e.g. spot welding or sealing. Thus it is very important to verify the process performance after the parameter has been set.

Allowed values

ON or OFF.

Related information

[Path Resolution on page 454](#).

[CPU Load Equalization on page 461](#).

6 Topic Motion

6.21.18. Speed Control Warning

6.21.18. Speed Control Warning

Parent

Speed Control Warning belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

speed_control_warning

Description

By setting *Speed Control Warning* to Yes, a warning will be given when the robot moves slower than the programmed speed.

Usage

When several robots (and other mechanical units) are in synchronized movement mode, in a MultiMove application, all simultaneous move instruction finish at the same time. This means that if one robot has a longer path or a slower programmed speed than another robot, the speed of the second robot is decreased.

If a robot is working with an application where the speed is important (e.g. arc welding or gluing), *Speed Control Warning* can be used to give a warning when the actual speed is slower than the programmed speed.

Limitations

This parameter is only useful when using the RobotWare option MultiMove.

The speed is only supervised for robot TCP speed. No warning is given for the speed of additional axes.

Allowed values

Yes or No.

Additional information

When several tasks are in synchronized movement mode, all these tasks are planned by the same *Motion Planner* (the first *Motion Planner* of those involved in the synchronization). If this *Motion Planner* has *Speed Control Warning* set to Yes, all the synchronized robot speeds are supervised. If it has *Speed Control Warning* set to No, no robot speeds are supervised.

6.21.19. Speed Control Percent

Parent

Speed Control Percent belongs to the type *Motion Planner*, in the topic *Motion*.

Cfg name

speed_control_percent

Description

If *Speed Control Warning* is set to Yes, a warning will be issued when the actual speed is slower than this percentage of the programmed speed.

Usage

If a robot is working with an application where the speed is important (e.g. arc welding or gluing), *Speed Control Percent* defines the slowest speed (in percent of programmed speed) that is acceptable.

Limitations

This parameter is only useful when using the RobotWare option MultiMove.

The speed is only supervised for robot TCP speed. No warning is given for the speed of additional axes.

Allowed values

A number between 0 and 100 (in percent of programmed speed).

6 Topic Motion

6.22.1. The Motion Supervision type

6.22 Type Motion Supervision

6.22.1. The Motion Supervision type

Overview

This section describes the type *Motion Supervision*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

MOTION_SUP

Type description

Motion supervision is used to activate, deactivate or adjust the collision detection functionality. For detailed information about collision detection, see the *Application manual - Motion coordination and supervision*, chapter *Collision Detection*.

No controller restart required

Most of the motion supervision parameters do not require a restart of the controller when modified.

Limitations

The type *Motion supervision* is mainly used to configure the installed option *Collision detection*. For a system without this option, changing the values for most of the parameters does not affect the system.

Related information

[How to tune motion supervision on page 287.](#)

[Application manual - Motion coordination and supervision.](#)

6.22.2. Name

Parent

Name belongs to the type *Motion Supervision*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the motion supervision setup.

Limitation

This parameter cannot be changed.

Related information

How to tune motion supervision on page 287.

6 Topic Motion

6.22.3. Path Collision Detection

Parent

Path Collision Detection belongs to the type *Motion Supervision*, in the topic *Motion*.

Cfg name

path_col_detect_on

Description

Path Collision Detection turns the collision detection on or off for program execution.

Usage

Setting *Path Collision Detection* to YES turns on the collision detection, NO turns off the collision detection.

Allowed values

YES or NO.

Related information

[How to tune motion supervision on page 287.](#)

6.22.4. Jog Collision Detection

Parent

Jog Collision Detection belongs to the type *Motion Supervision*, in the topic *Motion*.

Cfg name

jog_col_detect_on

Description

Jog collision Detection turns the collision detection on or off for jogging.

Limitation

Changing this parameter only affects the system if the option *Collision detection* is installed.

Allowed values

YES or NO.

Related information

[How to tune motion supervision on page 287.](#)

6 Topic Motion

6.22.5. Path Collision Detection Level

Parent

Path Collision Detection Level belongs to the type *Motion Supervision*, in the topic *Motion*.

Cfg name

path_col_detect_level

Description

Path Collision Detection Level modifies the supervision level for the collision detection for program execution by a specified percentage value.

Usage

The supervision level for collision detection in program execution is specified as a percentage. A large value makes the function less sensitive. The default value is 100%. For detailed information, see the *Application manual - Motion coordination and supervision*.

Limitation

Changing this parameter only affects the system if the option *Collision detection* is installed.

Allowed values

A value in the interval 1 to 300, specifying the supervision level in %.

The default value is 100%.

Related information

[*How to tune motion supervision on page 287.*](#)

[*Application manual - Motion coordination and supervision.*](#)

6.22.6. Jog Collision Detection Level

Parent

Jog Collision Detection Level belongs to the type *Motion Supervision*, in the topic *Motion*.

Cfg name

jog_col_detect_level

Description

Jog Collision Detection Level modifies the supervision level for the collision detection for jogging by a specified percentage value.

Usage

The supervision level for collision detection in jogging is specified as a percentage, where a large value makes the function less sensitive. The default value is 100%. For detailed information, see the *Application manual - Motion coordination and supervision*.

Limitations

Changing this parameter only affects the system if the option *Collision detection* is installed.

Allowed values

A value in the interval 1 to 300, specifying the supervision level in %.

The default level is 100%.

Related information

[How to tune motion supervision on page 287](#).

[Application manual - Motion coordination and supervision](#).

6 Topic Motion

6.22.7. Collision Detection Memory

Parent

Collision Detection Memory belongs to the type *Motion Supervision*, in the topic *Motion*.

Cfg name

collision_detection_memory

Description

Collision Detection Memory defines how much the robot moves back on the path after a collision.

The parameter requires a restart of the controller when modified.

Usage

The robot movement back on the path after a collision is specified in seconds. If the robot was moving quickly before the collision, it will move further back than if the speed was lower. For detailed information, see the *Application manual - Motion coordination and supervision*.

Allowed values

A value in the interval 0.025 to 0.5, specifying the movement in seconds.

Related information

[How to tune motion supervision on page 287](#).

Application manual - Motion coordination and supervision.

6.23 Type Motion System

6.23.1. The Motion System type

Overview

This section describes the type *Motion System*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

MOTION_SYSTEM

Type description

Motion System includes parameters that are common for the entire system.

Non-editable parameters

The following parameter is visible but not editable in the software configuration tools:

- *Sensor Memory Mode*

As a consequence, the above parameter is not described in the manual.

6 Topic Motion

6.23.2. Name

6.23.2. Name

Parent

Name belongs to the type *Motion System*, in the topic *Motion*.

Cfg name

name

Description

Name specifies the name of the *Motion System* type.

Allowed values

A string with maximum 32 characters.

6.23.3. Min Temperature Cabinet

Parent

Min Temperature Cabinet belongs to the type *Motion System*, in the topic *Motion*.

Cfg name

min_temp_ambient_cabinet

Description

Min Temperature Cabinet defines the minimum ambient temperature where the cabinet is situated.

Allowed values

A value between -100 to 100 C, specifying the temperature in degrees Celsius.

6 Topic Motion

6.23.4. Max Temperature Cabinet

Parent

Max Temperature Cabinet belongs to the type *Motion System*, in the topic *Motion*.

Cfg name

max_temp_ambient_cabinet

Description

Max Temperature Cabinet defines the maximum ambient temperature where the cabinet is situated.

Allowed values

A value between -100 to 100 C, specifying the temperature in degrees Celsius.

Additional information

This parameter does not have to be changed if the controller is equipped with an extra fan for the cabinet.

6.23.5. Min Temperature Robot

Parent

Min Temperature Robot belongs to the type *Motion System*, in the topic *Motion*.

Cfg name

min_temp_ambient_robot

Description

Min Temperature Robot defines the minimum ambient temperature where the robot is situated.

Allowed values

A value between -100 to 100 C, specifying the temperature in degrees Celsius.

6 Topic Motion

6.23.6. Max Temperature Robot

6.23.6. Max Temperature Robot

Parent

Max Temperature Robot belongs to the type *Motion System*, in the topic *Motion*.

Cfg name

max_temp_ambient_robot

Description

Max Temperature Robot defines the maximum ambient temperature where the robot is situated.

Allowed values

A value between -100 to 100 C, specifying the temperature in degrees Celsius.

6.24 Type Motor

6.24.1. The Motor type

Overview

This section describes the *Motor* type which belongs to the topic *Motion*. Each parameter is described in a separate information topic in this section.

Cfg name

MOTOR

Type description

The type *Motor* describes the motor used for each axis. There is one configuration of the type *Motor* for each axis.

Note that only external axes are visible, the robot's axes motors are configured on delivery and should not be changed.

6 Topic Motion

6.24.2. Name

6.24.2. Name

Parent

Name belongs to the type *Motor*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the motor.

Allowed values

A string with maximum 32 characters.

6.24.3. Use Motor Type

Parent

Use Motor Type belongs to the type *Motor*, in the topic *Motion*.

Cfg name

use_motor_type

Description

Use Motor Type defines which type of motor is used for this type.

Usage

The type *Motor Type* defines the motor data.

Allowed values

A string with maximum 32 characters.

Related information

The type Motor Type on page 494.

6 Topic Motion

6.24.4. Use Motor Calibration

Parent

Use Motor Calibration belongs to the type *Motor*, in the topic *Motion*.

Cfg name

use_motor_calib

Description

Use Motor Calibration defines which type of motor calibration to be used.

Usage

The type *Motor Calibration* defines the motor's calibration data.

Allowed values

A string with maximum 32 characters.

Related information

The Motor Calibration type on page 487.

6.25 Type Motor Calibration

6.25.1. The Motor Calibration type

Overview

This section describes the type *Motor Calibration*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

MOTOR_CALIB

Type description

With the parameters in the *Motor Calibration* type, you can calibrate the robot's motors by entering the calibration values.

Motor calibration configuration is normally done during robot calibration. However, if the values are known, they can be specified directly.

Limitations

If calibration or commutator offset parameters are set, the corresponding offset valid parameters have to be set to YES, otherwise the offset parameter will not be used.

6 Topic Motion

6.25.2. Name

6.25.2. Name

Parent

Name belongs to the type *Motor Calibration*, in the topic *Motion*.

Cfg name

name

Description

Name specifies the name of the motor calibration setting it belongs to.

Usage

Name is used to reference the *Motor Calibration* from the parameter *Use Motor Calibration* in the type *Motor*.

Allowed values

A string with maximum 32 characters.

6.25.3. Commutator Offset

Parent

Commutator Offset belongs to the type *Motor Calibration*, in the topic *Motion*.

Cfg name

com_offset

Description

Commutator Offset defines the position of the motor (resolver) when the rotor is in the predefined commutation position relative to the stator.

Usage

ABB motors normally uses *Commutation Offset* value 1.57080.

Allowed values

A value between -6.283186 and 6.283186, specifying the offset in radians.

6 Topic Motion

6.25.4. Commutator Offset Valid

Parent

Commutator Offset Valid belongs to the type *Motor Calibration*, in the topic *Motion*.

Cfg name

valid_com_offset

Description

Commutator Offset Valid specifies whether the commutator offset is defined or not.

Allowed values

YES or NO.

Related information

[Commutator Offset on page 489](#).

6.25.5. Calibration Offset

Parent

Calibration Offset belongs to the type *Motor Calibration*, in the topic *Motion*.

Cfg name

cal_offset

Description

Calibration Offset defines the position of the motor (resolver) when the arm is in the calibration (zero) position.

Allowed values

A value between -6.283186 and 6.283186, specifying the offset in radians.

6 Topic Motion

6.25.6. Calibration Offset Valid

6.25.6. Calibration Offset Valid

Parent

Calibration Offset Valid belongs to the type *Motor Calibration*, in the topic *Motion*.

Cfg name

valid_cal_offset

Description

Calibration Offset Valid specifies whether the calibration offset is defined or not.

Allowed values

YES or NO.

Related information

[Calibration Offset on page 491](#).

6.25.7. Calibration Sensor Position

Parent

Calibration Sensor Position belongs to the type *Motor Calibration*, in the topic *Motion*.

Cfg name

cal_sensor_position

Description

Calibration Sensor Position defines the calibration sensor position on the arm side.

Usage

The value is set in degrees.

Allowed values

A value between -180 and 180 degrees.

Default value is 0 degrees.

6 Topic Motion

6.26.1. The type Motor Type

6.26 Type Motor Type

6.26.1. The type Motor Type

Overview

This section describes the type *Motor Type*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

MOTOR_TYPE

Type description

The type *Motor Type* is used to describe characteristics for the motor.

Limitations

The parameter values for *Motor Type* can only be changed for additional axis motors. The values can be observed for robot motors, but cannot be changed.

6.26.2. Name**Parent**

Name belongs to the type *Motor Type*, in the topic *Motion*.

Cfg name

name

Description

The name of the *Motor Type*.

Usage

Name is used to reference a motor type from the parameter *Use Motor Type* in the type *Motor*.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.26.3. Pole Pairs

6.26.3. Pole Pairs

Parent

Pole Pairs belongs to the type *Motor Type*, in the topic *Motion*.

Cfg name

pole_pairs

Description

Defines the number of pole pairs for the motor type.

Usage

Set *Pole Pairs* to the number of pole pairs (i.e. number of poles divided with 2) that the motor has.

Limitations

Pole Pairs can only be changed for additional axis motors. The values can be observed for robot motors, but cannot be changed.

Allowed values

An integer between 0 and 20.

6.26.4. Stall Torque

Parent

Stall Torque belongs to the type *Motor Type*, in the topic *Motion*.

Cfg name

torque_0

Description

The continuous stall torque, i.e. the torque the motor can produce at no speed and during an infinite time.

Usage

Set *Stall Torque* to the stall torque (T_0) specified by the motor manufacturer.

Limitations

Stall Torque can only be changed for additional axis motors. The values can be observed for robot motors, but cannot be changed.

Allowed values

A numeric value between 0 and 100000 (Nm).

6 Topic Motion

6.26.5. ke Phase to Phase

6.26.5. ke Phase to Phase

Parent

ke Phase to Phase belongs to the type *Motor Type*, in the topic *Motion*.

Cfg name

ke

Description

Nominal voltage constant.

Usage

ke Phase to Phase is the induced voltage (phase to phase) that corresponds to the speed 1 rad/s.

Limitations

ke Phase to Phase can only be changed for additional axis motors. The values can be observed for robot motors, but cannot be changed.

Allowed values

A numeric value between 0 and 10 (Vs/rad).

Additional information

Some motor manufacturers specify the value *kt* instead of *ke*. *ke* can then be calculated according to the formula:

$$ke = kt / \sqrt{3}$$

6.26.6. Max Current

Parent

Max Current belongs to the type *Motor Type*, in the topic *Motion*.

Cfg name

i_max

Description

Max current without irreversible magnetization.

Usage

Set *Max Current* to the root-mean-square of the maximum current the motor can withstand without irreversible demagnetization.

Limitations

Max Current can only be changed for additional axis motors. The values can be observed for robot motors, but cannot be changed.

Allowed values

A numeric value between 0 and 100 (A rms).

6 Topic Motion

6.26.7. Phase Resistance

6.26.7. Phase Resistance

Parent

Phase Resistance belongs to the type *Motor Type*, in the topic *Motion*.

Cfg name

r_stator_20

Description

Nominal winding resistance per phase at 20 degrees Celsius.

Usage

Set *Phase Resistance* to the stator phase resistance (R_{20}) specified by the motor manufacturer.

Limitations

Phase Resistance can only be changed for additional axis motors. The values can be observed for robot motors, but cannot be changed.

Allowed values

A numeric value between 0 and 100 (ohm).

6.26.8. Phase Inductance

Parent

Phase Inductance belongs to the type *Motor Type*, in the topic *Motion*.

Cfg name

L_stator

Description

Nominal winding inductance per phase at zero current.

Usage

Set *Phase Inductance* to the stator phase inductance (L_0) specified by the motor manufacturer.

Limitations

Phase Inductance can only be changed for additional axis motors. The values can be observed for robot motors, but cannot be changed.

Allowed values

A numeric value between 0 and 100 (H).

6 Topic Motion

6.27.1. The Path Sensor Synchronization type

6.27 Type Path Sensor Synchronization

6.27.1. The Path Sensor Synchronization type

Parent

This section describes the type *Path Sensor Synchronization* which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

PATH_SENSOR_SYNC

Type description

The type *Path Sensor Synchronization* define settings for sensor synchronization. The parameters of this type are used to set limits for the movements of a robot that is synchronized with an external device. Limits can be set for allowed deviation between calculated and actual position, and minimum/maximum TCP speed.

Limitations

Path Sensor Synchronization can only be used if you have the option *Sensor synchronization* installed.

Related information

Application manual - Motion coordination and supervision, chapter *Sensor synchronization*.

6.27.2. Name**Parent**

Name belongs to the type *Path Sensor Synchronization*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name for the path sensor synchronization.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.27.3. Max Advance Distance

Parent

Max Advance Distance belongs to the type *Path Sensor Synchronization*, in the topic *Motion*.

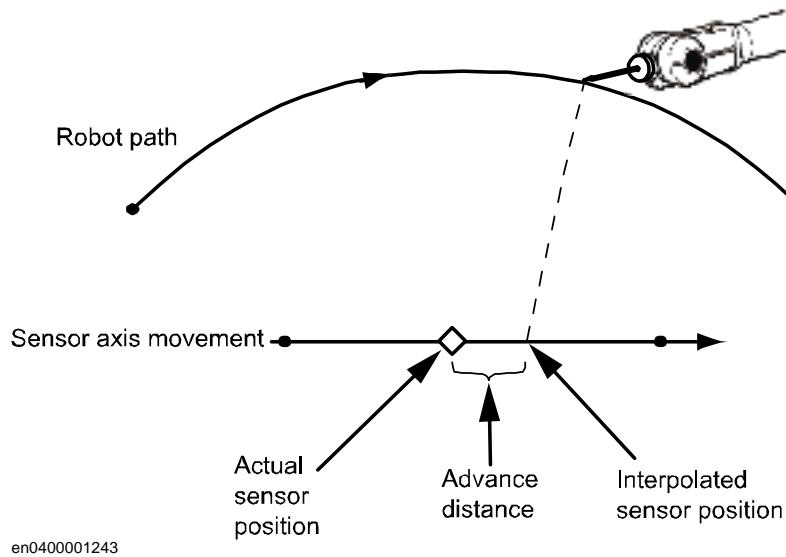
Cfg name

max_adv_dist_for_decel

Description

Max Advance Distance defines the maximum allowed advance distance between the sensor's interpolated position and its actual position.

The interpolated position of the sensor axis corresponds to the robot's position along its path when the robot is synchronized with the sensor.



Usage

If the interpolated position of the sensor axis is ahead of the actual position, a collision may occur. For example, if the robot enter a press based on the information that the press is open, but the press is actually still closed, the robot may move into the closed press. This can be avoided by using *Max Advance Distance*. If *Max Advance Distance* is exceeded, motion and execution is stopped.

Limitations

Max Advance Distance can only be used if you have the option *Sensor synchronization* installed.

Allowed values

A value between 0.01 and 5.0 (meters of movement on the external device that is connected to the sensor).

Default value is 0.1.

6.27.4. Max Delay Distance

Parent

Max Delay Distance belongs to the type *Path Sensor Synchronization*, in the topic *Motion*.

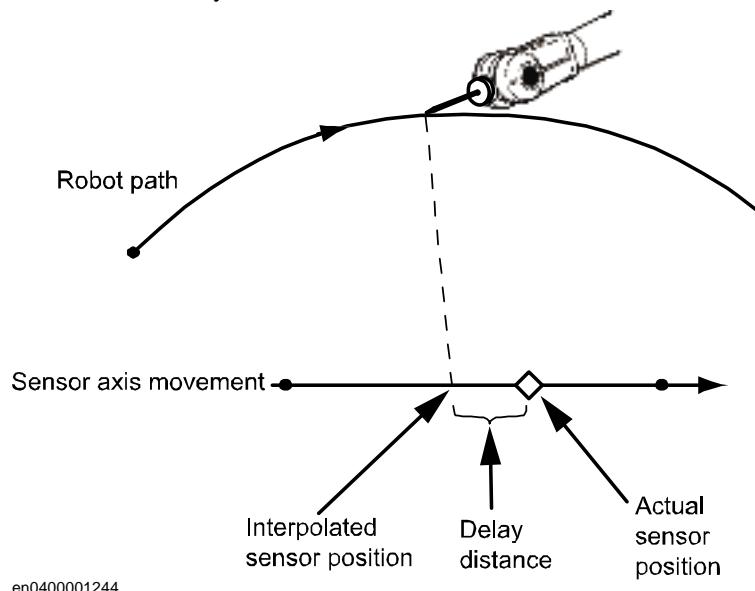
Cfg name

max_delay_dist_for_decel

Description

Max Delay Distance defines the maximum allowed delay distance between sensor's interpolated position and its actual position.

The interpolated position of the sensor axis corresponds to the robot's position along its path when the robot is synchronized with the sensor.



Usage

If the interpolated position of the sensor axis is behind the actual position, a collision may occur. A robot that is moving in an area where the external device will be later in the cycle can collide with the external device because of the incorrect timing. This can be avoided by using *Max Delay Distance*. If *Max Delay Distance* is exceeded, motion and execution is stopped.

Max Delay Distance can be disabled by setting its value to 0.

Limitations

Max Delay Distance can only be used if you have the option *Sensor synchronization* installed.

Allowed values

A numeric value between 0.0 and 5.0 (meters of movement on the external device that is connected to the sensor).

Default value is 0, which means that the supervision of the delay distance is not used.

6 Topic Motion

6.27.5. Max Synchronization Speed

Parent

Max Synchronization Speed belongs to the type *Path Sensor Synchronization*, in the topic *Motion*.

Cfg name

max_sync_speed

Description

Max Synchronization Speed defines the maximum allowed robot TCP speed during synchronization with an external device.

Usage

If the external device (that the robot is synchronized with) moves so fast that the robot should exceed *Max Synchronization Speed*, the robot speed will be limited to *Max Synchronization Speed*. The robot will slip behind, and the interpolated sensor position will be delayed compared to the actual sensor position, until the *Max Delay Distance* is reached.

Limitations

Max Synchronization Speed can only be used if you have the option *Sensor synchronization* installed.

Allowed values

A numeric value between 1.0 and 10.0 (m/s).

Default value is 4.0.

6.27.6. Min Synchronization Speed

Parent

Min Synchronization Speed belongs to the type *Path Sensor Synchronization*, in the topic *Motion*.

Cfg name

min_sync_speed

Description

Min Synchronization Speed defines the minimum allowed robot TCP speed during synchronization with an external device.

Usage

If the external device (that the robot is synchronized with) stops, the robot speed will maintain the *Max Synchronization Speed*. The robot will move ahead, and the interpolated sensor position will be in advance compared to the actual sensor position, until the *Max Advance Distance* is reached.

Limitations

Min Synchronization Speed can only be used if you have the option *Sensor synchronization* installed.

Allowed values

A value between 0.0 and 2.0 (m/s).

Default value is 0.1.

6 Topic Motion

6.27.7. Synchronization Type

Parent

Synchronization Type belongs to the type *Path Sensor Synchronization*, in the topic *Motion*.

Cfg name

sync_type

Description

Synchronization Type defines what type of synchronization to be used.

Limitations

Synchronization Type can only be used if you have the option *Sensor synchronization* installed.

Allowed values

Value:	Description:
MINIMAL_DIST	Synchronization based on distance, actual sensor position in corvec.
NOM_SPEED_SENS	Synchronization based on nominal sensor speed, actual sensor position in corvec.
NOM_SPEED_CALC	Synchronization based on nominal sensor speed, calculated sensor position in corvec.
MIN_DIST_CALC	Synchronization based on distance, calculated sensor position in corvec.
LOW_SPEED_SYNC	When robot and sensor speed is lower than 0.2 m/sec.
ROBOT_TO_ROBOT	To synchronize two robots through DeviceNet bus.
ROBOT_TO_PRESS	To synchronize robot with press moved by electric motor.
ROBOT_TO_HPRESS	To synchronize robot with hydraulic press.
SYNC_TO_IMM	To synchronize with injection moulding machine.

6.28 Type Process

6.28.1. The Process type

Overview

This section describes the type *Process*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

PROCESS

Type description

A process can be called from the parameter *Use Process* in the type *Joint*. The parameters in the type *Process* point out a process in the type *Linked M Process* or *SG Process* that will be used for that joint.

Related information

[Use Process on page 393](#).

[The Linked M Process type on page 422](#).

6 Topic Motion

6.28.2. Name

6.28.2. Name

Parent

Name belongs to the type *Process*, in the topic *Motion*.

Cfg name

name

Description

Name defines the identity of the process.

Usage

The *Name* of the process is used by a joint to call the process.

The process will, in its turn, call either a linked motor process (type *Linked M Process*) or a servo gun process (type *SG Process*).

Limitations

This parameter is only useful if you have either of the RobotWare options *Electronically Linked Motors* or *Spot Servo*.

Allowed values

A string.

6.28.3. Use SG Process

Parent

Use SG Process belongs to the type *Process*, in the topic *Motion*.

Cfg name

use_sg_process

Description

Use SG Process defines which *SG Process* to use.

Usage

Use SG Process refers to a process id defined by the parameter *Name* in the type *SG Process*.
SG Process is used to define a servo tool's behavior.

Limitations

SG Process can only be used for servo tools.

Allowed values

A string.

6 Topic Motion

6.28.4. Use Linked Motor Process

6.28.4. Use Linked Motor Process

Parent

Use Linked Motor Process belongs to the type *Process*, in the topic *Motion*.

Cfg name

use_linked_m_proc

Description

Use Linked Motor Process defines which linked motor process to use.

Usage

Use Linked Motor Process points to a process id defined by the parameter *Name* in the type *Linked M Process*.

The linked motor process is used to define a joint's behavior for *Electronically Linked Motors*.

Limitations

Use Linked Motor Process is only useful if you have the RobotWare option *Electronically Linked Motors*.

Allowed values

A string.

6.29 Type Relay

6.29.1. The Relay type

Overview

This section describes the type *Relay* which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

RELAY

Type description

The type *Relay* defines the characteristics of the relays that are used for the mechanical units, e.g. brake relays and run relays.

All relays for a robot supplied from ABB are defined on delivery. This means that adding or editing parameters of the *Relay* type is only necessary when additional axes are installed.

Related information

Application manual - Additional axes and stand alone controller.

6 Topic Motion

6.29.2. Output Signal

6.29.2. Output Signal

Parent

Output Signal belongs to the type *Relay* in the topic *Motion*.

Cfg name

Out_signal

Description

Output Signal defines the logical name of the output signal to the relay.

Usage

Characteristics of relays for manipulators need to be defined when additional axes are installed.

The value of *Output Signal* must be identical to the name of the signal, including upper and lower case letters.

Prerequisites

The logical signal name must be defined in the type *Signal* in the topic *I/O*.

Allowed values

A string with maximum 32 characters.

Related information

[The Signal type on page 173.](#)

6.29.3. Input Signal

Parent

Input Signal belongs to the type *Relay* in the topic *Motion*.

Cfg name

in_signal

Description

Input Signal defines the logical name of the input signal to the relay.

Usage

Characteristics of relays for manipulators need to be defined when additional axes are installed.

The value of *Input Signal* must be identical to the name of the signal, including upper and lower case letters.

Prerequisites

The logical signal name must be defined in the type *Signal* in the topic *I/O*.

Allowed values

A string with maximum 32 characters.

Related information

[The Signal type on page 173.](#)

6 Topic Motion

6.30.1. The Robot type

6.30 Type Robot

6.30.1. The Robot type

Overview

This section describes the type *Robot* which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

ROBOT

Type description

The type *Robot* contains a number of parameters that are common for a robot in the robot system. The robot is a mechanical unit with more than one joint. Parameters of this type are used to define which joints the robot consists of and the base frame of the robot.

Non-editable parameters

The following parameter is visible but not editable in the software configuration tools:

- *Use Robot Calibration*

As a consequence, the above parameter is not described in the manual.

6.30.2. Name**Parent**

Name belongs to the type *Robot*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the robot.

Limitations

This parameter cannot be changed.

6 Topic Motion

6.30.3. Use Old SMB

6.30.3. Use Old SMB

Parent

Use Old SMB belongs to the type *Robot*, in the topic *Motion*.

Cfg name

use_old_smb

Description

To adapt earlier robot systems, running earlier SMB board versions without flash memory, to later software versions, the parameter *Use Old SMB* is to be set to TRUE.

Usage

Earlier systems, in this context, is any robot system delivered with an SMB board of any of these revisions:

- DSQC 313, all revisions
 - DSQC 520, revision 5 and earlier
 - DSQC 562, revision 2 and earlier
-

Allowed values

True or False.

6.30.4. Use Joint 1, 2, 3, 4, 5, 6

Parent

Use Joint 1, Use Joint 2, Use Joint 3, Use Joint 4 , Use Joint 5, and Use Joint 6 belongs to the type *Robot*, in the topic *Motion*.

Cfg names

use_joint_0
use_joint_1
use_joint_2
use_joint_3
use_joint_4
use_joint_5

Description

Use joint 1 defines which joint data to use as the robot's first joint.
Use joint 2 defines which joint data to use as the robot's second joint.
Use joint 3 defines which joint data to use as the robot's third joint.
Use joint 4 defines which joint data to use as the robot's fourth joint.
Use joint 5 defines which joint data to use as the robot's fifth joint.
Use joint 6 defines which joint data to use as the robot's sixth joint.

Usage

The joints are defined in the type *Joint*.

Allowed values

A string with maximum 32 characters, specifying an already defined joint.

Related information

[The Joint type on page 390.](#)

6 Topic Motion

6.30.5. Base Frame x, y, z

6.30.5. Base Frame x, y, z

Parent

Base Frame x, Base Frame y, and Base Frame z belongs to the type *Robot*, in the topic *Motion*.

Cfg names

base_frame_pos_x
base_frame_pos_y
base_frame_pos_z

Description

Base Frame x defines the x-direction of the base frame position in relation to the world frame (in meters).

Base Frame y defines the y-direction of the base frame position in relation to the world frame (in meters).

Base Frame z defines the z-direction of the base frame position in relation to the world frame (in meters).

Allowed values

A value between -1000 to 1000, specifying the relation in meters.

Related information

[How to define base frame on page 281](#).

6.30.6. Base Frame q1, q2, q3, q4

Parent

Base Frame q1, Base Frame q2, Base Frame q3, and Base Frame q4 belongs to the type *Robot*, in the topic *Motion*.

Cfg name

base_frame_orient_u0
base_frame_orient_u1
base_frame_orient_u2
base_frame_orient_u3

Description

Base Frame q1 defines the first quaternion (q1) of the base frame orientation in relation to the world frame.

Base Frame q2 defines the second quaternion (q2) of the base frame orientation in relation to the world frame.

Base Frame q3 defines the third quaternion (q3) of the base frame orientation in relation to the world frame.

Base Frame q4 defines the fourth quaternion (q4) of the base frame orientation in relation to the world frame.

Allowed values

A value between -1 to 1 specifying the orientation.

Related information

[How to define base frame on page 281.](#)

6 Topic Motion

6.30.7. Base Frame Moved by

Parent

Base Frame Moved by belongs to the type *Robot*, in the topic *Motion*.

Cfg name

base_frame_coordinated

Description

Base Frame Moved by defines the name of a mechanical unit (a robot or a single joint) that moves the base frame of the robot.

Allowed values

A string with maximum 32 characters, specifying the unit name.

Related information

How to define base frame on page 281.

6.30.8. Gravity Alpha

Parent

Gravity Alpha belongs to the type *Robot*, in the topic *Motion*.

Cfg name

gravity_alpha

Description

Gravity Alpha defines the orientation of gravity with respect to the base frame.

Usage

The alpha gravity is a positive rotation direction around the x-axis in the base frame. The value is set in radians.

Allowed values

A value between -6.283186 and 6.283186 radians.

Default value is 0.

6 Topic Motion

6.30.9. Gravity Beta

6.30.9. Gravity Beta

Parent

Gravity Beta belongs to the type *Robot*, in the topic *Motion*.

Cfg name

gravity_beta

Description

Gravity Beta defines the orientation of gravity with respect to the base frame.

Usage

The beta gravity is a positive rotation direction around the y-axis in the base frame. The value is set in radians.

Allowed values

A value between -6.283186 and 6.283186 radians.

Default value is 0.

6.30.10. Gamma Rotation

Parent

Gamma Rotation belongs to the type *Robot*, in the topic *Motion*.

Cfg name

gamma_rotation

Description

Gamma Rotation defines the orientation of the robot's foot on the travel carriage.

Usage

The gamma rotation is a positive rotation direction around the z-axis of the base frame of the travel carriage (track motion). The value is set in radians.

Prerequisites

This parameter is only useful if the parameter *7 axes high performance motion* is set.

Allowed values

A value between -6.283186 and 6.283186 radians.

Default values is 0.

Related information

[7 axes high performance motion on page 532](#).

6 Topic Motion

6.30.11. Upper Work Area x, y, z

6.30.11. Upper Work Area x, y, z

Parent

Upper Work Area x, *Upper Work Area y*, and *Upper Work Area z* belongs to the type *Robot*, in the topic *Motion*

Cfg names

upper_work_area_x
upper_work_area_y
upper_work_area_z

Description

Upper work area x defines the x-coordinate of the upper bound of the work area for the robot.

Upper work area y defines the y-coordinate of the upper bound of the work area for the robot.

Upper work area z defines the z-coordinate of the upper bound of the work area for the robot.

Allowed values

A numeric value higher than the respective *Lower Work Area* value in meters.

Related information

[How to define work area on page 282.](#)

[Lower Work Area x, y, z on page 527.](#)

[How to define base frame on page 281.](#)

6.30.12. Lower Work Area x, y, z

Parent

Lower Work Area x, *Lower Work Area y*, and *Lower Work Area z* belongs to the type *Robot*, in the topic *Motion*.

Cfg names

lower_work_area_x
lower_work_area_y
lower_work_area_z

Description

Lower work area x defines the x-coordinate of the lower bound of the work area for the robot.

Lower work area y defines the y-coordinate of the lower bound of the work area for the robot.

Lower work area z defines the z-coordinate of the lower bound of the work area for the robot.

Allowed values

A numeric value lower than the respective *Upper Work Area* value in meters.

Related information

[How to define work area on page 282.](#)

[Upper Work Area x, y, z on page 526.](#)

[How to define base frame on page 281.](#)

6 Topic Motion

6.30.13. Upper Check Point Bound x, y, z

6.30.13. Upper Check Point Bound x, y, z

Parent

Upper Check Point Bound x, *Upper Check Point Bound y*, and *Upper Check Point Bound z* belongs to the type *Robot*, in the topic *Motion*.

Cfg names

upper_arm_cp_bound_x
upper_arm_cp_bound_y
upper_arm_cp_bound_z

Description

Upper Check Point Bound x defines the cartesian x-coordinate upper check point bound on arm check point.

Upper Check Point Bound y defines the cartesian y-coordinate upper check point bound on arm check point.

Upper Check Point Bound z defines the cartesian z-coordinate upper check point bound on arm check point.

Usage

The arm check point can be bound to restrict the movement area.

Allowed values

A numeric value higher than the respective coordinate *Lower Check Point Bound* in meters.

Related information

How to define arm check point on page 283.

Lower Check Point Bound x, y, z on page 529.

6.30.14. Lower Check Point Bound x, y, z

Parent

Lower Check Point Bound x, *Lower Check Point Bound y*, and *Lower Check Point Bound z* belongs to the type *Robot*, in the topic *Motion*.

Cfg names

lower_arm_cp_bound_x
lower_arm_cp_bound_y
lower_arm_cp_bound_z

Description

Lower Check Point Bound x defines the cartesian x-coordinate lower check point bound on arm check point.

Lower Check Point Bound y defines the cartesian y-coordinate lower check point bound on arm check point.

Lower Check Point Bound z defines the cartesian z-coordinate lower check point bound on arm check point.

Usage

The arm check point can be bound to restrict the movement area.

Allowed values

A numeric value lower than the respective coordinate *Upper Check Point Bound* in meters.

Related information

[How to define arm check point on page 283.](#)

[Upper Check Point Bound x, y, z on page 528.](#)

6 Topic Motion

6.30.15. Use Six Axes Corvec

6.30.15. Use Six Axes Corvec

Parent

Use Six Axes Corvec belongs to the type *Robot*, in the topic *Motion*.

Cfg name

use_six_axis_corvec

Description

Defines if the position adjustment is made on six axes.

Usage

Set *Use Six Axes Corvec* to Yes if the position adjustment should be made on six axes. In this case, the orientation of the tool is exact. Otherwise the correction is only made on axis 1, 2, and 3 and the orientation accuracy is lower.

Using *Six Axes Corvec* takes a little more CPU time and should not be used if not needed.

Limitations

Use Six Axes Corvec can only be used if *Conveyor tracking* option is installed.

Use Six Axes Corvec has no effect on coordinated tracks.

Use Six Axes Corvec is only possible to use on six axes robots.

Allowed values

True or False.

Related information

Application manual - Conveyor tracking.

6.30.16. Track Conveyor with Robot

Parent

Track Conveyor with Robot belongs to the type *Robot*, in the topic *Motion*.

Cfg name

track_convey_with_robot

Description

Defines if the robot should track the conveyor.

Usage

Set *Track Conveyor with Robot* to Yes if the robot should track the conveyor without using the track axis, even if robot is coordinated with track. Default value is No.

Limitations

Track Conveyor with Robot can only be used with option *Conveyor tracking* installed.

Allowed values

True or False.

Related information

Application manual - Conveyor tracking.

6 Topic Motion

6.30.17. 7 axes high performance motion

6.30.17. 7 axes high performance motion

Parent

7 axes high performance motion belongs to the type *Robot*, in the topic *Motion*.

Cfg name

seven_axes_hp_motion

Description

7 axes high performance motion defines the name of the single that moves the robot.

Usage

This parameter should only be set if a "high performance track motion"-additional package is present in your mediapool.

Allowed values

A string with maximum 32 characters, specifying the unit name.

6.30.18. Time to Inposition

Parent

Time to Inposition belongs to the type *Robot*, in the topic *Motion*.

Cfg name

time_to_inpos

Description

Time to Inposition defines the delay time between the last position reference and the inposition event when reaching a fine point.

Limitations

Time to Inposition is only used by the option *Conveyor tracking*.

Allowed values

A value between 0 and 2.0 seconds.

Default value is 0.08 seconds. This should not be changed!

Related information

Application manual - Conveyor tracking.

6 Topic Motion

6.31.1. The Robot Serial Number type

6.31 The Robot Serial Number type

6.31.1. The Robot Serial Number type

Overview

This section describes the type *Robot Serial Number*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

ROBOT_SERIAL_NUMBER

Type description

The type *Robot Serial Number* contains parameters that define the robot's serial number.

Related information

[The Robot type on page 516.](#)

6.31.2. Name**Parent**

Name belongs to the type *Robot Serial Number*, in the topic *Motion*.

Cfg name

name

Description

Name specifies the name of the robot that the serial number belongs to.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.31.3. Robot Serial Number High Part

6.31.3. Robot Serial Number High Part

Parent

Robot Serial Number High Part belongs to the type *Robot Serial Number*, in the topic *Motion*.

Cfg name

robot_serial_number_high_part

Description

Robot Serial Number High Part defines the high part of the robot's serial number.

Usage

The high part is the first four characters of the serial number.

The serial number can be found on the robot's identification plate.

Allowed values

A string with maximum four characters.

Default value is 0000.

6.31.4. Robot Serial Number Low Part

Parent

Robot Serial Number Low Part belongs to the type *Robot Serial Number*, in the topic *Motion*.

Cfg name

robot_serial_number_low_part

Description

Robot Serial Number Low Part defines the low part of the robot's serial number.

Usage

The low integer part of the serial number.

The serial number can be found on the robot's identification plate.

Allowed values

An integer value with maximum nine digits.

Default value is 0.

6 Topic Motion

6.32.1. The SG Process type

6.32 Type SG Process

6.32.1. The SG Process type

Overview

This section describes the type *SG Process*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

SG_PROCESS

Type description

The type *SG Process* contains parameters to configure the behavior of a servo gun (or other servo tool). There are parameters for adjusting the timing, force and thickness when closing and opening a servo gun. It is also possible to specify how the tip wear calibration will be performed. The relation between tip force and motor torque is configured as shown below.

Limitations

SG Process can only be used if you have servo tools.

Force-torque relation

Tip Force 1-10 and *Motor Torque 1-10* are used to define the motor torque the motor should apply when a gun closing is ordered with a certain tip force. Due to friction, the relation between force and torque is not always linear.

Between 2 and 10 points can be used to define the motor torque as a function of the tip force. The number of points used is defined in *Number of Stored Forces*.

Ordered closing tip force:	Resulting motor torque:
Tip Force 1	Motor Torque 1
Tip Force 2	Motor Torque 2
Tip Force 3	Motor Torque 3
Tip Force 4	Motor Torque 4
Tip Force 5	Motor Torque 5
Tip Force 6	Motor Torque 6
Tip Force 7	Motor Torque 7
Tip Force 8	Motor Torque 8
Tip Force 9	Motor Torque 9
Tip Force 10	Motor Torque 10

When calculating the force-torque function, the origin (force=0, torque=0) is considered to be an extra point in the diagram. For tip force values between points, linear interpolation is used. For tip force values higher than the highest defined tip force, extrapolation from the last two points is used.

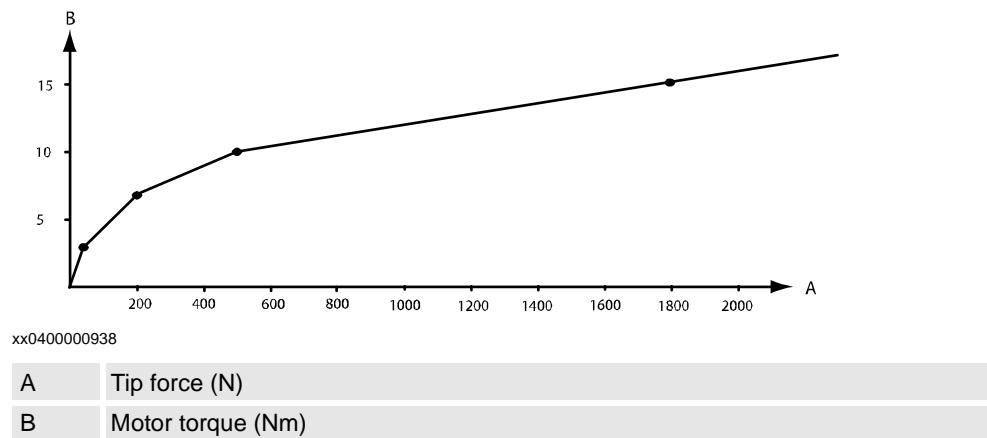
*Continued***Example**

In this example, four points are used to define the relation between tip force and motor torque. Any values given for point 5 to 10 are ignored.

These parameters and values are configured:

Parameter:	Value:
<i>Number of Stored Forces</i>	4
<i>Tip Force 1</i>	50
<i>Tip Force 2</i>	200
<i>Tip Force 3</i>	500
<i>Tip Force 4</i>	1800
<i>Motor Torque 1</i>	3
<i>Motor Torque 2</i>	7
<i>Motor Torque 3</i>	10
<i>Motor Torque 4</i>	15

The results of this configuration is the following graph for motor torque as function of tip force:



6 Topic Motion

6.32.2. Name

6.32.2. Name

Parent

Name belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

name

Description

The name of the *SG Process*.

Usage

Name is used to reference a *SG Process* from the parameter *Use SG Process* in the type *Process*.

Allowed values

A string with maximum 32 characters.

6.32.3. Use Force Master

Parent

Use Force Master belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

use_force_master

Description

Use Force Master determines which *Force Master* should be used.

Usage

Use Force Master is a reference to the parameter *Name* in the type *Force Master*.

Prerequisites

A *Force Master* must be configured before *Use Force Master* can refer to it.

Limitations

Use Force Master can only be used for servo tools.

Allowed values

A string with maximum 32 characters.

Related information

[The Force Master type on page 344](#).

6 Topic Motion

6.32.4. Sync Check Off

6.32.4. Sync Check Off

Parent

Sync Check Off belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

sync_check_off

Description

Defines if the servo tool synchronization check is turned off.

Usage

Set *Sync Check Off* to Yes to disable the servo tool synchronization check. This can be useful to do to manage the servo tool before having done the service calibration.

Limitations

Sync Check Off can only be used for servo tools.

Allowed values

Yes or No.

Related information

Application manual - Servo motor control.

Example

To turn off the synchronization check, use this RAPID code:

```
STTune SERVOGUN, 1, SyncCheckOff;
```

To turn on the synchronization check again:

```
STTuneReset SERVOGUN;
```

6.32.5. Close time adjust

Parent

Close time adjust belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

min_close_time_adjust

Description

Adjustment of the ordered minimum close time of the gun.

Usage

If the servo gun is ordered to start closing before the robot is in position, the tips might touch the work piece too early. By setting *Close time adjust* to a positive value, this can be avoided.

If there is a waiting period when the robot is in position but before the servo gun is closing, the cycle time can be reduced by setting *Close time adjust* to a negative value.

Close time adjust may be used to delay the closing slightly when the synchronized pre closing is used for welding.

Limitations

Close time adjust can only be used if you have servo tools.

Allowed values

Numerical value between -100 and 100 (seconds).

6 Topic Motion

6.32.6. Close position adjust

6.32.6. Close position adjust

Parent

Close position adjust belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

close_position_adjust

Description

Adjustment of the ordered position when closing the gun to a position and force.

When the tool tips reach the position (plate thickness) ordered by the close instruction, the force control starts. This tool tip position can be adjusted with *Close position adjust* to make the force control start earlier.

Usage

To make sure the tool tips do not touch the work piece before the force control starts, *Close position adjust* can be used to leave some space between the tool tips and the work object.

Limitations

Close position adjust can only be used if you have servo tools.

Allowed values

Numeric value between 0 and 0.005 (meters).

6.32.7. Force Ready Delay

Parent

Force Ready Delay belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

pre_sync_delay_time

Description

Force Ready Delay is used to delay the close ready event. This will make the servo gun wait some extra time when the closing is finished and the ordered force is achieved.

Usage

Force Ready Delay can be used if the servo gun needs some extra time for the force to be stabilized.

Limitations

Force Ready Delay can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 30 (seconds).

6 Topic Motion

6.32.8. Max Force Control Motor Torque

Parent

Max Force Control Motor Torque belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

max_motor_torque

Description

Max allowed motor torque for force control. Commanded force will be reduced, if the required motor torque is higher than this value.

Usage

Max Force Control Motor Torque is used to protect the gun from mechanical overload.

Limitations

Max Force Control Motor Torque can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 100 (Nm).

The default value is 7 Nm.

6.32.9. Post-synchronization Time

Parent

Post-synchronization Time belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

post_sync_time

Description

Post-synchronization Time is used to anticipate the open ready event. The open instruction will be considered ready before the servo gun is completely open.

Usage

Post-synchronization Time can be used to save cycle time. The waiting time between the opening of the servo gun and the execution of the next instruction can be reduced.

The synchronization may fail if *Post-synchronization Time* is set too high.

Limitations

Post-synchronization Time can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 0.5 (seconds).

6 Topic Motion

6.32.10. Calibration Mode

Parent

Calibration Mode belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

calib_mode

Description

Number of tip wear calibration points, i.e. the number of times the servo gun closes during a tip wear calibration.

Usage

If the flexibility of a servo gun is not linearly dependent of the force, more than two measurement points may be necessary. This will improve the plate thickness detection.

Limitations

Calibration Mode can only be used if you have servo tools.

Allowed values

An integer between 2 and 10.

The default value is 2.

6.32.11. Calibration Force High

Parent

Calibration Force High belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

calib_force_high

Description

The force used for the last closing when calibrating the tip wear of a servo gun.

Calibration Force High affects the gun stiffness calibration.

Usage

Set *Calibration Force High* to a value close to the highest force you intend to use the servo gun for. This way it will be well calibrated for forces of that size.

Limitations

Calibration Force High can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 12000 (N).

The default value is 3500 N.

Additional information

The force of the first gun closing in a tip wear calibration is specified in *Calibration Force Low*. If more than two measurement points are used, the force of these measurement points are evenly distributed between *Calibration Force Low* and *Calibration Force High*.

6 Topic Motion

6.32.12. Calibration Force Low

Parent

Calibration Force Low belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

calib_force_low

Description

The force used for:

- the second gun closing of a new tips calibration
- the second gun closing of a tool change calibration
- the first gun closing of a tip wear calibration.

Calibration Force Low affects the gun position calibration.

Usage

It is recommended to set *Calibration Force Low* to a value close to the lowest force you intend to use the servo gun for, but not a higher value than half the value of *Calibration Force High*.

Limitations

Calibration Force Low can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 12000 (N).

The default value is 1500 N.

6.32.13. Calibration Time

Parent

Calibration Time belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

calib_time

Description

The time that the servo gun waits in closed position during calibration.

Usage

If the servo gun needs more time to stabilize, *Calibration Time* can be increased. This can improve the gun position calibration.

In order to make the calibrations faster, *Calibration Time* can be decreased.

Limitations

Calibration Time can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 30 (seconds).

The default value is 0.5 seconds.

6 Topic Motion

6.32.14. Number of Stored Forces

Parent

Number of Stored Forces belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

no_of_active_db_posts

Description

Used to define the relation between tip force and motor torque for a servo gun. *Number of Stored Forces* defines for how many tip force values you want to define the motor torque, i.e. the number of points in the force-torque graph (see [Force-torque relation on page 538](#)).

Usage

Measure the tip force and motor torque for a number of points. Set *Number of Stored Forces* to the number of points you want to specify.

Limitations

Number of Stored Forces can only be used if you have servo tools.

Allowed values

An integer between 2 and 10.

The default value is 3.

6.32.15. Tip Force 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Parent

Tip Force 1, Tip Force 2, Tip Force 3, Tip Force 4, Tip Force 5, Tip Force 6, Tip Force 7, Tip Force 8, Tip Force 9, and Tip Force 10 belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

squeeze_force_1
squeeze_force_2
squeeze_force_3
squeeze_force_4
squeeze_force_5
squeeze_force_6
squeeze_force_7
squeeze_force_8
squeeze_force_9
squeeze_force_10

Description

Used to define the relation between tip force and motor torque for a servo gun (see [Force-torque relation on page 538](#)).

Tip Force 1 defines the ordered closing force for the first point in the force-torque graph.

Tip Force 2 defines the ordered closing force for the second point in the force-torque graph.

Tip Force 3 defines the ordered closing force for the third point in the force-torque graph.

Tip Force 4 defines the ordered closing force for the fourth point in the force-torque graph.

Tip Force 5 defines the ordered closing force for the fifth point in the force-torque graph.

Tip Force 6 defines the ordered closing force for the sixth point in the force-torque graph.

Tip Force 7 defines the ordered closing force for the seventh point in the force-torque graph.

Tip Force 8 defines the ordered closing force for the eighth point in the force-torque graph.

Tip Force 9 defines the ordered closing force for the ninth point in the force-torque graph.

Tip Force 10 defines the ordered closing force for the tenth point in the force-torque graph.

Usage

Measure the tip force and the motor torque for some different values.

Set *Tip Force 1* to the tip force value of the first point you want to specify, and *Motor Torque 1* to the corresponding motor torque.

Set *Tip Force 2* to the tip force value of the second point you want to specify, and *Motor Torque 2* to the corresponding motor torque.

Set *Tip Force 3* to the tip force value of the third point you want to specify, and *Motor Torque 3* to the corresponding motor torque.

Set *Tip Force 4* to the tip force value of the fourth point you want to specify, and *Motor Torque 4* to the corresponding motor torque.

Continues on next page

6 Topic Motion

6.32.15. Tip Force 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Continued

Set *Tip Force 5* to the tip force value of the fifth point you want to specify, and *Motor Torque 5* to the corresponding motor torque.

Set *Tip Force 6* to the tip force value of the sixth point you want to specify, and *Motor Torque 6* to the corresponding motor torque.

Set *Tip Force 7* to the tip force value of the seventh point you want to specify, and *Motor Torque 7* to the corresponding motor torque.

Set *Tip Force 8* to the tip force value of the eighth point you want to specify, and *Motor Torque 8* to the corresponding motor torque.

Set *Tip Force 9* to the tip force value of the ninth point you want to specify, and *Motor Torque 9* to the corresponding motor torque.

Set *Tip Force 10* to the tip force value of the tenth point you want to specify, and *Motor Torque 10* to the corresponding motor torque.

Limitations

Tip Force can only be used for servo tools.

Allowed values

A numeric value between 0 and 20000 (N).

6.32.16. Motor Torque 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Parent

Motor Torque 1, Motor Torque 2, Motor Torque 3, Motor Torque 4, Motor Torque 5, Motor Torque 6, Motor Torque 7, Motor Torque 8, Motor Torque 9, and Motor Torque 10 belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

```
squeeze_torque_1
squeeze_torque_2
squeeze_torque_3
squeeze_torque_4
squeeze_torque_5
squeeze_torque_6
squeeze_torque_7
squeeze_torque_8
squeeze_torque_9
squeeze_torque_10
```

Description

Used to define the relation between tip force and motor torque for a servo gun (see [Force-torque relation on page 538](#)).

Motor Torque 1 defines the motor torque for the first point in the force-torque graph.

Motor Torque 2 defines the motor torque for the second point in the force-torque graph.

Motor Torque 3 defines the motor torque for the third point in the force-torque graph.

Motor Torque 4 defines the motor torque for the fourth point in the force-torque graph.

Motor Torque 5 defines the motor torque for the fifth point in the force-torque graph.

Motor Torque 6 defines the motor torque for the sixth point in the force-torque graph.

Motor Torque 7 defines the motor torque for the seventh point in the force-torque graph.

Motor Torque 8 defines the motor torque for the eighth point in the force-torque graph.

Motor Torque 9 defines the motor torque for the ninth point in the force-torque graph.

Motor Torque 10 defines the motor torque for the tenth point in the force-torque graph.

Usage

Measure the tip force and the motor torque for some different values

Set *Motor Torque 1* to the motor torque value of the first point you want to specify, and *Tip Force 1* to the corresponding tip force.

Set *Motor Torque 2* to the motor torque value of the second point you want to specify, and *Tip Force 2* to the corresponding tip force.

Set *Motor Torque 3* to the motor torque value of the third point you want to specify, and *Tip Force 3* to the corresponding tip force.

6 Topic Motion

6.32.16. Motor Torque 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Continued

Set *Motor Torque 4* to the motor torque value of the fourth point you want to specify, and *Tip Force 4* to the corresponding tip force.

Set *Motor Torque 5* to the motor torque value of the fifth point you want to specify, and *Tip Force 5* to the corresponding tip force.

Set *Motor Torque 6* to the motor torque value of the sixth point you want to specify, and *Tip Force 6* to the corresponding tip force.

Set *Motor Torque 7* to the motor torque value of the seventh point you want to specify, and *Tip Force 7* to the corresponding tip force.

Set *Motor Torque 8* to the motor torque value of the eighth point you want to specify, and *Tip Force 8* to the corresponding tip force.

Set *Motor Torque 9* to the motor torque value of the ninth point you want to specify, and *Tip Force 91* to the corresponding tip force.

Set *Motor Torque 10* to the motor torque value of the tenth point you want to specify, and *Tip Force 10* to the corresponding tip force.

Limitations

Motor Torque can only be used for servo tools.

Allowed values

A numeric value between -100 and 100 (Nm).

6.32.17. Soft Stop Timeout

Parent

Soft Stop Timeout belongs to the type *SG Process*, in the topic *Motion*.

Cfg name

soft_stop_timeout

Description

If a soft stop occurs during constant force, *Soft Stop Timeout* defines how long the force will be maintained. The force will be reduced after this time-out, or when opening is commanded.

Usage

If you want the gun to remain closed a short period after a soft stop, set *Soft Stop Timeout* to the desired time-out value.

Setting *Soft Stop Timeout* to 0 will make the gun release its force immediately when a soft stop occurs.

Limitations

Soft Stop Timeout can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 1.2 (seconds).

The default value is 0.3 seconds.

6 Topic Motion

6.33.1. The Single type

6.33 Type Single

6.33.1. The Single type

Overview

This section describes the type *Single*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

SINGLE

Type description

The type *Single* contains a number of parameters that are common for a single in the robot system. The single is a mechanical unit with one joint. Parameters of this type are used to define which joint the single consist of and the base frame of the single.

6.33.2. Name**Parent**

Name belongs to the type *Single*, in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the single.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.33.3. Use Single Type

6.33.3. Use Single Type

Parent

Use Single Type belongs to the type *Single*, in the topic *Motion*.

Cfg name

use_single_type

Description

Use Single Type defines what single type is used.

Usage

The single type is defined in the type *Single Type*.

Allowed values

A string with maximum 32 characters.

Related information

[The Single Type type on page 565.](#)

6.33.4. Use Joint

Parent

Use Joint belongs to the type *Single*, in the topic *Motion*.

Cfg name

use_joint

Description

Use Joint defines which joint data to use for the single.

Usage

The joints are defined in the type *Joint*.

Allowed values

A string with maximum 32 characters.

Related information

The Joint type on page 390.

6 Topic Motion

6.33.5. Base Frame x, y, z

6.33.5. Base Frame x, y, z

Parent

Base Frame x, Base Frame y, and Base Frame z belongs to the type *Single* in the topic *Motion*.

Cfg names

base_frame_pos_x
base_frame_pos_y
base_frame_pos_z

Description

Base Frame x defines the x-direction of the base frame position in relation to the world frame (in meters).

Base Frame y defines the y-direction of the base frame position in relation to the world frame (in meters).

Base Frame z defines the z-direction of the base frame position in relation to the world frame (in meters).

Allowed values

A value between -1,000 and 1,000 meters.

Related information

[How to define base frame on page 281](#).

6.33.6. Base Frame q1, q2, q3, q4

Parent

Base Frame q1, Base Frame q2, Base Frame q3, and Base Frame q4 belongs to the type *Single* in the topic *Motion*.

Cfg names

base_frame_orient_u0
base_frame_orient_u1
base_frame_orient_u2
base_frame_orient_u3

Description

Base Frame q1 defines the first quaternion (q1) of the base frame orientation in relation to the world frame.

Base Frame q2 defines the second quaternion (q2) of the base frame orientation in relation to the world frame.

Base Frame q3 defines the third quaternion (q3) of the base frame orientation in relation to the world frame.

Base Frame q4 defines the fourth quaternion (q4) of the base frame orientation in relation to the world frame.

Allowed values

A value between -1 and 1 specifying the orientation.

Related information

[How to define base frame on page 281.](#)

6 Topic Motion

6.33.7. Base Frame Coordinated

Parent

Base Frame Coordinated belongs to the type *Single* in the topic *Motion*.

Cfg name

base_frame_coordinated

Description

Base Frame Coordinated defines the name of robot or single that moves the base frame of this single.

Allowed values

A string with maximum 32 characters.

Related information

[How to define base frame on page 281.](#)

6.34 Type Single Type

6.34.1. The Single Type type

Overview

This section describes the type *Single Type* which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

SINGLE_TYPE

Type description

The type *Single Type* contains a number of parameters that are common for a single type in the robot system. The single is a mechanical unit with one joint.

Related information

[The Single type on page 558.](#)

6 Topic Motion

6.34.2. Name

6.34.2. Name

Parent

Name belongs to the type *Single Type* in the topic *Motion*.

Cfg name

name

Description

Name defines the name of the single type.

Allowed values

A string with maximum 32 characters.

6.34.3. Mechanics

Parent

Mechanics belongs to the type *Single Type* in the topic *Motion*.

Cfg name

mechanics

Description

Mechanics defines what type of mechanics the single type uses.

Allowed values

The following mechanics are available/allowed:

Value:	Description:
TRACK	Linear track
EXT_LIN	Conveyor, linear
FREE_ROT	Rotating axis
EXT_CTL	For internal use only
EXT_ROT	Conveyor, rotating
SS_ROT	Sensor synchronization, rotating movement
SS_LIN	Sensor synchronization, linear movement

Related information

Application manual - Additional axes and stand alone controller.

6 Topic Motion

6.35.1. The SIS Parameters type

6.35 Type SIS Parameters

6.35.1. The SIS Parameters type

Overview

This section describes the type *SIS Parameters* which belong to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

SIS_EXTERNAL

Type description

The type *SIS Parameters* describes the service intervals and warning levels for the robot. The service interval can be set in both production time and calendar time.

Limitations

Changing the parameter values in *SIS Parameters* is only useful if you have an IRB 6600, IRB 6650, or IRB 7600 robot.

Related information

Product manual for the robot.

6.35.2. Name**Parent**

Name belongs to the type *SIS Parameters* in the topic *Motion*.

Cfg name

name

Description

Name defines the SIS parameters' name.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.35.3. Operational Limit

6.35.3. Operational Limit

Parent

Operational Limit belongs to the type *SIS Parameters* in the topic *Motion*.

Cfg name

prod_time_service_interval

Description

Operational Limit describes the service interval measured in production time.

Usage

The service interval for production time, *Operational Limit*, for ABB robots is normally set to 20,000 hours on delivery and should not be changed.

When the *Operational Limit* is reached, the FlexPendant displays a message from the elog.

If *Operational Limit* is set to 0, the function is disabled.

Allowed values

A value between 0 and 50,000 hours.

6.35.4. Calendar Limit

Parent

Calendar Limit belongs to the type *SIS Parameters* in the topic *Motion*.

Cfg name

calender_time_service_interval

Description

Calendar Limit defines the service interval in calender time.

Usage

The service interval for calendar time, *Calendar Limit*, for ABB robots is normally set to 1 year on delivery and should not be changed.

When the *Calendar Limit* is reached, the FlexPendant displays a message from the elog.

If *Calendar Limit* is set to 0, the function is disabled.

Allowed values

A value between 0 and 20 years.

6 Topic Motion

6.35.5. Operational Warning

6.35.5. Operational Warning

Parent

Operational Warning belongs to the type *SIS Parameters* in the topic *Motion*.

Cfg name

prod_time_warning_level

Description

Operational Warning defines when the warning before reached service level for production time should occur.

Usage

The value of *Operational Warning* is a percentage of the *Operational Limit*. A lower number gives a shorter time between the warning and the reached service level.

If *Operational Warning* is set to 0, the function is disabled.

Allowed values

A value between 0 and 100 %.

Related information

Operational Limit on page 570.

6.35.6. Calendar Warning

Parent

Calendar Warning belongs to the type *SIS Parameters* in the topic *Motion*.

Cfg name

calender_time_warning_level

Description

Calendar Warning defines when the warning before reached service level for calender time should occur.

Usage

The value of *Calendar Warning* is a percentage of the *Calendar Limit*. A lower number gives a shorter time between the warning and the reached service level.

If *Calendar Warning* is set to 0, the function is disabled.

Allowed values

A value between 0 and 100 %.

Related information

[Calendar Limit on page 571](#).

6 Topic Motion

6.35.7. Gearbox Warning

6.35.7. Gearbox Warning

Parent

Gearbox Warning belongs to the type *SIS Parameters* in the topic *Motion*.

Cfg name

gear_box_warning_level

Description

Gearbox Warning defines when the warning before reached service level for gear box should occur.

Usage

The gearbox service level is calculated automatically, using among other other things the value of *Gearbox Warning*.

For an ABB robot using SIS Parameters, the value is typically set to 100.

If *Gearbox Warning* is set to 0, the function is disabled.

Allowed values

A value between 0 and 100%.

6.36 Type Stress Duty Cycle

6.36.1. The Stress Duty Cycle type

Overview

This section describes the type *Stress Duty Cycle*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

STRESS_DUTY_CYCLE

Type description

The type *Stress Duty Cycle* is used to protect axes, gearboxes, etc. Damages due to too high mechanical forces are avoided by setting limits for speed and torque.

Limitations

Parameters of the type *Stress Duty Cycle* can only be defined for additional axes.

6 Topic Motion

6.36.2. Name

6.36.2. Name

Parent

Name belongs to the type *Stress Duty Cycle*, in the topic *Motion*.

Cfg name

name

Description

The name of the *Stress Duty Cycle*.

Usage

Name is used to reference a *Stress Duty Cycle* from the parameter *Use Stress Duty Cycle* in the type *Drive System*.

Allowed values

A string with maximum 32 characters.

6.36.3. Speed Absolute Max

Parent

Speed Absolute Max belongs to the type *Stress Duty Cycle*, in the topic *Motion*.

Cfg name

speed_absolute_max

Description

The absolute highest motor speed to be used.

Usage

Limit the motor speed with *Speed Absolute Max* to avoid too much stress on the axis. If, for example, the gear box is the limiter for the speed, set *Speed Absolute Max* to a value that will protect the gear box.

Allowed values

A numeric value between 0 and 1500 (rad/s on motor side).

6 Topic Motion

6.36.4. Torque Absolute Max

6.36.4. Torque Absolute Max

Parent

Torque Absolute Max belongs to the type *Stress Duty Cycle*, in the topic *Motion*.

Cfg name

torque_absolute_max

Description

The absolute highest motor torque to be used.

Usage

Limit the motor torque with *Torque Absolute Max* to avoid too much stress on the axis. If, for example, the gear box is the limiter for the torque, set *Torque Absolute Max* to a value that will protect the gear box.

Limitation

Torque Absolute Max can only be defined for additional axes.

Allowed values

A numeric value between 0 and 100000 (Nm on motor side).

6.37 Type Supervision

6.37.1. The Supervision type

Overview

This section describes the type *Supervision*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

SUPERVISION

Type description

The type *Supervision* is used for supervision of joints. Each joint has one set of parameters of the type *Supervision*.

Limitation

Parameters of the type *Supervision* can only be defined for additional axes.

Related information

The Joint type on page 390.

6 Topic Motion

6.37.2. Name

6.37.2. Name

Parent

Name belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

name

Description

The name of the supervision.

Allowed values

A string with maximum 32 characters.

6.37.3. Brake Release Supervision On

Parent

Brake Release Supervision On belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

brake_release_supervision_on

Description

Brake Release Supervision On defines if the brake release supervision is on or off.

Usage

Set *Brake Release Supervision On* to True to turn on brake release supervision. This activates a position supervision algorithm during brake release.

Allowed values

True or False.

6 Topic Motion

6.37.4. Speed Supervision

6.37.4. Speed Supervision

Parent

Speed Supervision belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

speed_supervision_on

Description

Defines if the speed supervision should be activated or not.

Usage

Speed supervision should normally be on (True).

Note! Deactivating the speed supervision can be dangerous.

Allowed values

True or False.

6.37.5. Position Supervision

Parent

Position Supervision belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

position_supervision_on

Description

Defines if the position supervision should be activated or not.

Usage

The position supervision should normally be on (True).

Note! Deactivating the position supervision can be dangerous.

Allowed values

True or False.

6 Topic Motion

6.37.6. Counter Supervision

6.37.6. Counter Supervision

Parent

Counter Supervision belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

counter_supervision_on

Description

Defines if the measurement system supervision should be activated or not.

Usage

The counter supervision should normally be on (True).

Note! Deactivating the counter supervision can be dangerous.

Allowed values

True or False.

6.37.7. Jam Supervision

Parent

Jam Supervision belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

jam_supervision_on

Description

Defines if the jam supervision should be activated or not.

Usage

The jam supervision should normally be activated (True).

Note! Deactivating the jam supervision can be dangerous.

Allowed values

True or False.

6 Topic Motion

6.37.8. Load Supervision

6.37.8. Load Supervision

Parent

Load Supervision belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

load_supervision_on

Description

Defines if the load supervision should be activated or not.

Usage

The load supervision should normally be on (True).

Allowed values

True or False.

6.37.9. Power Up Position Supervision

Parent

Power Up Position Supervision belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

power_up_position_on

Description

Defines if the power up position supervision should be activated or not.

Usage

The power up position supervision should normally be on (True).

Note! Deactivating the power up position supervision can be dangerous.

Allowed values

True or False.

6 Topic Motion

6.37.10. In Position Range

Parent

In Position Range belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

in_position_range

Description

Defines the allowed position deviation from fine point when the axis is considered to have reached the fine point.

Usage

Normally set to 1.

Allowed values

A value between 0 and 10 radians on motor side.

6.37.11. Zero Speed

Parent

Zero Speed belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

normalized_zero_speed

Description

Defines the maximum speed when the axis is considered to be standing still.

Usage

Normally set to 0.02.

Allowed values

A value between 0 and 1, where 1 equals max speed.

6 Topic Motion

6.37.12. Affects Forced Control

Parent

Affects Forced Control belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

joint_affect_forced_Kp

Description

Defines if the joint affects the in position forced control used in fine point.

Usage

Set to True if the joint should affect the in position forced control.

The forced control is used to reduce time for axis to go into the fine point.

Allowed values

True or False.

Related information

Forced Control Active on page 401, in the type *Lag Control Master 0*.

6.37.13. Forced on Position Limit

Parent

Forced on Position Limit belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

Kp_forced_on_limit

Description

The upper position limit for activation of forced control, measured from the fine point.

Usage

The upper position limit is measured in radians on the motor shaft.

Allowed values

A value between 0 and 5.

Related information

Affects Forced Control on page 590.

6 Topic Motion

6.37.14. Forced off Position Limit

Parent

Forced off Position Limit belongs to the type *Supervision*, in the topic *Motion*.

Cfg name

Kp_forced_off_limit

Description

The lower position limit for deactivation of forced control used close to the fine point.

Usage

The lower position limit is measured in radians on the motor shaft.

Limitations

Must have a lower value than *Forced on Position Limit*.

Allowed values

A value between 0 and 5.

Related information

Forced on Position Limit on page 591.

Affects Forced Control on page 590.

6.38 Type Supervision Type

6.38.1. The type Supervision Type

Overview

This section describes the type *Supervision Type*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

SUPERVISION_TYPE

Type description

The type *Supervision Type* is used for continuos supervision of position, speed and torque. These values should follow the planned path, within a tolerance interval, or the movement is stopped.

Limitations

Parameters of the type *Supervision Type* can only be defined for additional axes.

6 Topic Motion

6.38.2. Name

6.38.2. Name

Parent

Name belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

name

Description

The name of the *Supervision Type*.

Usage

Name is used to reference a *Supervision Type* from the parameter *Use Supervision Type* in the type *Supervision*.

Allowed values

A string with maximum 32 characters.

6.38.3. Max Force Control Position Error

Parent

Max Force Control Position Error belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

fc_position_limit

Description

Max allowed position error during force control.

If the position error is larger than *Max Force Control Position Error*, all movement is stopped.

Usage

When a servo gun is in force control mode it is not allowed to move more than the distance specified in *Max Force Control Position Error*.

The most common reasons for a servo gun to move during force control are:

- the servo gun is flexible and can give in when high forces are applied
- the force control may start before the gun has closed around the plate, e.g. because the ordered plate thickness is larger than the real plate thickness, or because the parameter *Close position adjust* is set to a value larger than 0.

Limitations

Max Force Control Position Error can only be used if you have servo tools.

Allowed values

A numeric value between 0 and 0.10 (meter).

The default value is 0.03 m.

6 Topic Motion

6.38.4. Max Force Control Speed Limit

Parent

Max Force Control Speed Limit belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

fc_speed_limit_factor

Description

Speed error factor during force control.

The speed limits for force control is defined in the type *Force Master Control*. If this speed limit multiplied with *Max Force Control Speed Limit* is exceeded, all movement is stopped.

Usage

The speed may for a short period of time exceed the speed limit (defined in type *Force Master Control*) before it is regulated to a value within the limits. To allow the speed to exceed the limit during this regulation without stopping all movement, *Max Force Control Speed Limit* must be set to a value larger than 1. How much the speed is allowed to over-shoot the limit is determined by *Max Force Control Speed Limit*.

Limitations

Max Force Control Speed Limit can only be used if you have servo tools.

Allowed values

A numeric value between 1 and 10. The value has no unit, but is a ratio of the speed limit defined in the type *Force Master Control*.

The default value is 1.1.

Related information

[The Force Master Control type on page 358.](#)

6.38.5. Dynamic Power Up Position Limit

Parent

Dynamic Power Up Position Limit belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

dynamic_power_up_position_limit

Description

Defines the maximum accepted power up position error at maximum speed.

Usage

Dynamic Power Up Position Limit sets a dynamic limit for measurement system supervision of moment during power fail.

A typical value is 120% of the maximum brake distance.

Allowed values

A value between 0 and 1000 in radians.

6 Topic Motion

6.38.6. Teach Max Speed Main

Parent

Teach Max Speed Main belongs to the type *Supervision Type*, In the topic *Motion*.

Cfg name

teach_mode_speed_max_main

Description

Defines the maximum speed for references in manual mode.

Usage

Teach Max Speed Main is used to limit the maximum speed in manual mode.

Allowed values

A value between 0 and 1, where 1 equals max speed.

6.38.7. Max Jam Time

Parent

Max Jam Time belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

max_jam_time

Description

Defines the maximum allowed time with maximum torque at zero speed.

Usage

Set *Max Jam Time* to protect the robot and equipment from faults and damage that may occur if the torque is high while the speed is zero.

Allowed values

A value between 0 and 2.0 seconds.

A typical value is 0.5.

6 Topic Motion

6.38.8. Max Overload Time

Parent

Max Overload Time belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

max_overload_time

Description

Defines the maximum allowed time with maximum torque while moving.

Usage

Set *Max Overload Time* to protect the robot and equipment from faults and damage. If *Max Overload Time* is exceeded, the controller will indicate an error in hardware, robot, load, or programming.

Allowed values

A value between 0 and 20 seconds.

A typical value is 0.2.

6.38.9. Teach Normalized Low Speed

Parent

Teach Normalized Low Speed belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

teach_normalized_low_speed

Description

Defines the servo supervision limit in manual mode.

Usage

The value of *Teach Normalized Low Speed* should be set so that the axis speed does not exceed 250 mm/s.

Allowed values

A value between 0 and 1, where 1 equals max speed.

6 Topic Motion

6.38.10. Auto Max Speed Supervision Limit

Parent

Auto Max Speed Supervision Limit belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

auto_mode_max_speed_sup_limit

Description

Defines the maximum speed supervision limit in automatic mode.

Usage

Auto Max Speed Supervision Limit is typically set to 1.2 to allow margin against speed overshoot, interference from external forces, etc.

Allowed values

A value between 0 and 5, where 1 equals max speed.

A typical value is 1.2.

6.38.11. Influence Group

Parent

Influence Group belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

influence_group

Description

Defines the type of influence group for the *Supervision Type*. An influence group is a group of axes, mechanically affecting each other.

Usage

Influence Group is used to calculate supervision levels.

Normally, for axes not affecting each other, deactivate the function by setting *Influence Group* to 0.

Allowed values

An integer between 0 and 10.

6 Topic Motion

6.38.12. Alarm Position Limit for Brake Release

Parent

Alarm Position Limit for Brake Release belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

brake_release_position_alarm_limit

Description

Alarm Position Limit for Brake Release defines the emergency stop limit for position supervision during brake release.

Usage

An emergency stop is generated if the axis motor moves more than the defined value of *Alarm Position Limit for Brake Release* directly after brake release.

Allowed values

A value between 0 and 1000, defined in radians on motor side.

Default value is 1.0.

6.38.13. Position OK Ratio for Brake Release

Parent

Position OK Ratio for Brake Release belongs to the type *Supervision Type*, in the topic *Motion*.

Cfg name

brake_release_position_ok_ratio

Description

Position OK Ratio for Brake Release defines the maximum position error for the axis when the axis should leave the brake supervision state and change to normal operation.

Usage

The value of *Position OK Ratio for Brake Release* is a ratio of the value of parameter *Alarm Position Limit for Brake Release*.

Allowed values

A value between 0 and 1.

Default value is 0.2, a normal value is 0.2 - 0.5.

Related information

[Alarm Position Limit for Brake Release on page 604](#).

6 Topic Motion

6.39.1. The Transmission type

6.39 Type Transmission

6.39.1. The Transmission type

Overview

This section describes the type *Transmission*, which belongs to the topic *Motion*. Each parameter of this type is described in a separate information topic in this section.

Cfg name

TRANSMISSION

Type description

Each set of parameters of the type *Transmission* belongs to a joint (robot joint or external axis).

The parameters in *Transmission* determine the transmission gear ratio between the motor and the axis.

Limitations

The transmission gear ratio can only be defined for additional axes.

The transmission gear ratio for the robot joints are defined by ABB and cannot be changed.

6.39.2. Name

Parent

Name belongs to the type *Transmission*, in the topic *Motion*.

Cfg name

name

Description

The name of the *Transmission*.

Usage

Name is used to reference a *Transmission* from the parameter *Use Transmission* in the type *Joint*.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.39.3. Rotating Move

6.39.3. Rotating Move

Parent

Rotating Move belongs to the type *Transmission*, in the topic *Motion*.

Cfg name

rotating_move

Description

Rotating Move defines if the axis is rotating or linear.

Usage

For rotating axes, set *Rotating Move* to Yes. For linear axes, set *Rotating Move* to No.

Rotating Move affects if the transmission gear ratio is defined as motor radians per joint radians, or motor radian per joint meter.

Allowed values

Yes or No.

The default value is No (i.e. that the axis is linear).

6.39.4. Transmission Gear Ratio

Parent

Transmission Gear Ratio belongs to the type *Transmission*, in the topic *Motion*.

Cfg name

transm_joint

Description

Transmission Gear Ratio defines the transmission gear ratio between motor and joint.

Usage

For rotating axes, set *Transmission Gear Ratio* to the number of revolutions the motor performs for every revolution of the joint. For linear axes, set *Transmission Gear Ratio* to motor radians per meter.

Limitations

Transmission Gear Ratio can only be defined for external axes. *Transmission Gear Ratio* for the robot joints are defined by ABB and cannot be changed.

Allowed values

A numeric value between -100 000 and +100 000.

6 Topic Motion

6.39.5. Transmission Gear High

Parent

Transmission Gear High belongs to the type *Transmission*, in the topic *Motion*.

Cfg name

high_gear

Description

When a joint is in independent mode, *Transmission Gear High* is the numerator in the fraction representing the transmission gear ratio between motor and joint. The denominator is the parameter *Transmission Gear Low*.

Usage

When a joint is set to independent mode, the transmission gear ratio is represented as *Transmission Gear High* divided by *Transmission Gear Low*. See *How to define transmission gear ratio for independent joints* for more information on how to use these parameters.

Limitations

The parameter *Transmission Gear High* is only useful if you have the RobotWare option *Independent Axes*.

When a joint is not in independent mode, it uses the parameter *Transmission Gear Ratio* instead of *Transmission Gear High* and *Transmission Gear Low*.

Allowed values

An integer value.

Related information

[How to define transmission gear ratio for independent joints on page 288](#).

[Transmission Gear Low on page 611](#).

[Application manual - Motion functions and events](#).

6.39.6. Transmission Gear Low

Parent

Transmission Gear Low belongs to the type *Transmission*, in the topic *Motion*.

Cfg name

low_gear

Description

When a joint is in independent mode, *Transmission Gear Low* is the denominator in the fraction representing the transmission gear ratio between motor and joint. The numerator is the parameter *Transmission Gear High*.

Usage

When a joint is set to independent mode, the transmission gear ratio is represented as *Transmission Gear High* divided by *Transmission Gear Low*. See *How to define transmission gear ratio for independent joints* for more information on how to use these parameters.

Limitations

The parameter *Transmission Gear Low* is only useful if you have the RobotWare option *Independent Axes*.

When a joint is not in independent mode, it uses the parameter *Transmission Gear Ratio* instead of *Transmission Gear High* and *Transmission Gear Low*.

Allowed values

An integer value.

Related information

[How to define transmission gear ratio for independent joints on page 288](#).

[Transmission Gear High on page 610](#).

[Application manual - Motion functions and events](#).

6 Topic Motion

6.40.1. The Uncalibrated Control Master 0 type

6.40 Type Uncalibrated Control Master 0

6.40.1. The Uncalibrated Control Master 0 type

Overview

This section describes the type *Uncalibrated Control Master 0*, which belongs to the topic *Motion*. Each parameter of the type is described in a separate information topic in this section.

Cfg name

UCCM0

Type description

The type *Uncalibrated Control Master 0* is used to regulate uncalibrated axes. If one axis in a mechanical unit is uncalibrated, *Uncalibrated Control Master 0* is used to regulate all axes in that mechanical unit.

6.40.2. Name**Parent**

Name belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

Cfg name

UCCM0 name

Description

The name of the *Uncalibrated Control Master 0*.

Usage

Name is used to reference an *Uncalibrated Control Master 0* from the parameter *Uncalibrated Control Master* in the type *Joint*.

Allowed values

A string with maximum 32 characters.

6 Topic Motion

6.40.3. Kp, Gain Position Loop

6.40.3. Kp, Gain Position Loop

Parent

Kp, Gain Position Loop belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

Cfg name

Kp

Description

Proportional gain in the position regulation loop.

Usage

The higher the value of *Kp, Gain Position Loop*, the better tracking and disturbance rejection.
If the position regulation overshoots, decrease *Kp, Gain Position Loop*.

Limitations

Kp, Gain Position Loop only affects the axis when it is uncalibrated (or when another axis in the same mechanical unit is uncalibrated).

Allowed values

A numeric value between 0 and 1000 (1/s).

6.40.4. Kv, Gain Speed Loop

Parent

Kv, Gain Speed Loop belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

Cfg name

Kv

Description

Proportional gain in the speed regulation loop.

Usage

The higher the value of *Kv, Gain Speed Loop*, the better tracking and disturbance rejection.
If the level of oscillation or noise is too high, decrease *Kv, Gain Speed Loop*.

Limitations

Kv, Gain Speed Loop only affects the axis when it is uncalibrated (or when another axis in the same mechanical unit is uncalibrated).

Allowed values

A numeric value between 0 and 100 (Nms/rad).

6 Topic Motion

6.40.5. Ti Integration Time Speed Loop

6.40.5. Ti Integration Time Speed Loop

Parent

Ti Integration Time Speed Loop belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

Cfg name

Ti

Description

Integration time in the speed regulation loop.

Usage

The lower the value of *Ti Integration Time Speed Loop*, the better tracking and disturbance rejection.

If the level of oscillation or noise is too high, increase *Ti Integration Time Speed Loop*.

Limitations

Ti Integration Time Speed Loop only affects the axis when it is uncalibrated (or when another axis in the same mechanical unit is uncalibrated).

Allowed values

A numeric value between 0 and 10 (seconds).

The default value is 10 seconds.

6.40.6. Speed Max Uncalibrated

Parent

Speed Max Uncalibrated belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

Cfg name

speed_max_n

Description

Speed Max Uncalibrated defines the maximum allowed speed for an uncalibrated axis.

Usage

Use *Speed Max Uncalibrated* as a limit for the speed of the axis when it is regulated as an uncalibrated axis.

Limitations

Speed Max Uncalibrated only affects the axis when it is uncalibrated (or when another axis in the same mechanical unit is uncalibrated).

Allowed values

A numeric value between 0 and 670 (rad/s on motor side).

6 Topic Motion

6.40.7. Acceleration Max Uncalibrated

Parent

Acceleration Max Uncalibrated belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

Cfg name

acc_max_n

Description

Acceleration Max Uncalibrated defines the maximum allowed acceleration for an uncalibrated axis.

Usage

Use *Acceleration Max Uncalibrated* as a limit for the acceleration of the axis when it is regulated as an uncalibrated axis.

Limitations

Acceleration Max Uncalibrated only affects the axis when it is uncalibrated (or when another axis in the same mechanical unit is uncalibrated).

Allowed values

A numeric value between 0 and 10000 (rad/s² on motor side).

6.40.8. Deceleration Max Uncalibrated

Parent

Deceleration Max Uncalibrated belongs to the type *Uncalibrated Control Master 0*, in the topic *Motion*.

Cfg name

dec_max_n

Description

Deceleration Max Uncalibrated defines the maximum allowed deceleration for an uncalibrated axis.

Usage

Use *Deceleration Max Uncalibrated* as a limit for the deceleration of the axis when it is regulated as an uncalibrated axis.

Limitations

Deceleration Max Uncalibrated only affects the axis when it is uncalibrated (or when another axis in the same mechanical unit is uncalibrated).

Allowed values

A numeric value between 0 and 10000 (rad/s² on motor side).

6 Topic Motion

6.40.8. Deceleration Max Uncalibrated

A

acceleration data, type 292
access level, type 131
application protocol, type 27
arm check point, type 316
arm load, type 319
arm, type 298
Auto Condition Reset, type 60

B

brake, type 324
bus, type 138

C

communication, topic 23
control parameters, type 332
controller, topic 57
cross connection, type 147

D

drive module, type 337
drive system, type 340

E

event routine, type 71

F

fieldbus command type, type 168
fieldbus command, type 164
force master control, type 358
force master, type 344
friction compensation, type 380

I

I/O, topic 123

J

jog parameters, type 385
joint, type 390

L

lag control master 0, type 396
linked m process, type 422

M

mains, type 430
man-machine communication 265
measurement channel, type 434
mechanical unit group, type 79
mechanical unit, type 440
modpos settings, type 84
most common I/O signal, type 266
most common instruction, type 272
motion planner, type 450
motion supervision, type 470
motion system, type 477
motion, topic 279
motor calibration, type 487
motor type, type 494
motor, type 483

N

NORMAL task type 116
NoSafety trustlevel 119

O

operator safety, type 93

P

path return region, type 96
path sensor synchronization, type 502
physical channel, type 42
process, type 509

R

relay, type 513
robot serial number, type 534
robot, type 516
Run Mode Settings, type 102

S

safety run chain, type 105
SEMISTATIC task type 116
SG process, type 538
signal, type 173
single type, type 565
single, type 558
SIS parameters, type 568
STATIC task type 116
stress duty cycle, type 575
supervision type, type 593
SysFail trustlevel 119
SysHalt trustlevel 119
SysStop trustlevel 119
system input, type 200
system misc, type 108
system output, type 222
system parameter definition 21

T

task, type 113
topic definition 21
transmission protocol, type 51
transmission, type 606
type definition 21

U

uncalibrated control master, type 612
unit type, type 255
unit, type 243

3HAC17076-1, rev E, en



ABB Robotics
S-721 68 VÄSTERÅS
SWEDEN
Telephone: +46 (0) 21 344000
Telefax: +46 (0) 21 132592