

ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Visit our website - *Click HERE*

RMC70/150/200 Motion Controllers
And
RMCTools Software
User Manual

Version 4.20.0 December 16, 2021

Copyright © 2004- 2021, Delta Computer Systems, Inc. All Rights Reserved.
deltamotion.com

Condensed Contents

1. Introducing the RMC Family 1
Overview of the RMC family capabilities and applications.

2. Starting Up the RMC 7
A Step-by-step guide to quickly get up and running.

3. Controller Features 53
Control Types and more…

4. Using RMCTools 243
Detailed information on how to use the RMCTools motion control software.

5. Programming 337
A guide to programming the RMC.

6. Communications 455
Serial, PROFIBUS, Ethernet.

7. System Components 719
RMC hardware modules.

8. Command Reference 857
Detailed information on how to use each RMC command.

9. Registers 1025
Detailed information on all RMC registers.

10. Wiring and Installation 1449
Detailed wiring diagrams.

11. Troubleshooting 1537
Hints and assistance. Detailed information on error codes.

12. Index 1557

Table of Contents

1. Introducing the RMC Family 1

1.1. RMC Family Motion Controllers 1
1.2. Basics of Operation 2
1.3. Disclaimer 4

2. Starting Up the RMC 7

2.1. RMC Startup Procedure 7
2.2. Scaling 12
2.2.1. Scaling Overview 12
2.2.2. Analog Position Scaling 14
2.2.3. Analog Velocity Scaling 14
2.2.4. Analog Acceleration Scaling 15
2.2.5. Load Cell Scaling 15
2.2.6. Analog Pressure or Force Scaling 17
2.2.7. MDT Scaling 19
2.2.8. SSI Scaling 21
2.2.9. Rotary Scaling 22
2.2.10. Quadrature Scaling (Linear) 24
2.2.11. Scaling Rates on a Position Axis 25
2.3. Tuning 26
2.3.1. Tuning Overview 26
2.3.2. Tuning Wizard 29
2.3.3. Autotuning 30
2.3.4. Creating Plots for Tuning 31
2.3.5. Gain Calculator 32
2.3.6. Tuning a Position Axis 33
2.3.7. Tuning a Pressure/Force System 44
2.3.8. Tuning a Position-Pressure or Position-Force System 46
2.3.9. Tuning a Motor in Torque Mode 49
2.3.10. Tuning a Pneumatic System 50
2.3.11. Tuning Active Damping and Acceleration Control 51

3. Controller Features 53

3.1. RMC Controller Features 53
3.2. General 54
3.2.1. Firmware 54
3.2.2. Loop Time 55
3.2.3. Plotting the Loop Time 59
3.2.4. RUN/PROGRAM/Disabled Mode 60
3.2.5. Registration 62
3.2.6. Homing 64
3.2.7. System Time 68
3.2.8. Event Timers 72
3.2.9. Physical Limit Inputs 77
3.2.10. Feedback Resolution 79
3.2.11. Valve Linearization 82
3.3. Axes 87
3.3.1. Axis Types 87
3.3.2. Defining Axes 90
3.3.3. Control and Reference Axes 91
3.3.3.1. Axis Type: Control 91
3.3.3.2. Axis Type: Reference 92

3.3.3.3. Axis Type: Cascading Outer Loop 93
3.3.4. Other Axis Types 94
3.3.4.1. Axis Type: Rotary and Linear 94
3.3.4.2. Axis Type: Incremental and Absolute 95
3.3.4.3. Virtual Axes 95
3.3.4.4. Input Type: Pressure 96
3.3.4.5. Input Type: Single-Input Force 96
3.3.4.6. Input Type: Dual-Input (Differential) Force 97
3.3.4.7. Input Type: Custom 98
3.4. Halts 99
3.4.1. Halts Overview 99
3.4.2. External Halt 100
3.4.3. Closed Loop Halt 100
3.4.4. Open Loop Halt 102
3.4.5. Direct Output Halt 103
3.5. Control Modes 104
3.5.1. Control Modes Overview 104
3.5.2. Closed Loop Control 105
3.5.3. Open Loop Control 106
3.5.4. Position PID 106
3.5.5. Velocity PID 108
3.5.6. Gain Sets Overview 110
3.5.7. Ratioed Gains 111
3.5.8. Gain Scheduling 112
3.5.9. Unidirectional Mode 113
3.5.10. Velocity and Torque Mode 113
3.5.11. Advanced 114
3.5.11.1. Position I-PD 114
3.5.11.2. Velocity I-PD 116
3.5.11.3. Active Damping 118
3.5.11.4. Acceleration Control 119
3.5.11.5. Cascade Control 120
3.6. Motion 123
3.6.1. Synchronizing Axes 123
3.6.2. Using Rotary Motion 125
3.6.3. Velocity Control 131
3.6.4. Gearing 132
3.6.5. Simulating Motion 136
3.6.6. Step Jumps 139
3.6.7. Curves, Cams, and Splines 140
3.6.7.1. Curves Overview 140
3.6.7.2. Managing Curves in the Curve Tool 142
3.6.7.3. Creating Curves Using the Curve Add Command 144
3.6.7.4. Creating Large Curves using the Curve Add Command 146
3.6.7.5. Curve Interpolation Methods and Options 150
3.6.7.6. Curve Data Formats 155
3.6.7.7. Curve Status Error Codes 161
3.6.7.8. Curve Storage Capacity 163
3.6.7.9. Example: Creating a Curve using the Curve Add Command 166
3.7. Pressure and Force Control 170
3.7.1. Pressure/Force Control Overview 170
3.7.2. Controlling Only Pressure or Force 171
3.7.3. Position-Pressure and Position-Force Control 175
3.7.4. Velocity-Pressure and Velocity-Force Control 180
3.7.5. Pressure/Force Limit 180
3.7.6. Pressure/Force Limit Details 183
3.8. Filtering/Modeling 184
3.8.1. Filtering 184

3.8.2. Filter Algorithm Types 187
3.8.3. Modeling 189
3.9. Plots 190
3.9.1. Using Plots 190
3.9.2. Saving and Exporting Plots 192
3.9.3. Triggering Plots 195
3.9.4. Using Plots with a Host Controller 197
3.9.5. Reading RMC Plots with a Host Controller 197
3.9.6. Mean Squared Error 208
3.10. Custom Feedback 209
3.10.1. Custom Feedback 209
3.10.2. Switching Feedback using Custom Feedback 214
3.10.3. Feedback Linearization using Curves 215
3.10.4. Feedback Linearization Using a Mathematical Formula 216
3.10.5. Redundant Feedback using Custom Feedback 217
3.11. Applications 219
3.11.1. Electric Servo Motor Control 219
3.11.2. Hydraulic Control 220
3.11.3. Pneumatic Control 221
3.12. Transducers Basics 222
3.12.1. MDT Transducer Fundamentals 222
3.12.2. SSI Fundamentals 225
3.12.3. Analog Fundamentals 229
3.12.4. Load Cell and Wheatstone Bridge Fundamentals 231
3.12.5. Quadrature Encoder Fundamentals 235
3.12.6. Resolver Fundamentals 236
3.13. Other 237
3.13.1. Controller Image Upload/Download 237

4. Using RMCTools 243

4.1. Using RMCTools 243
4.2. Using the RMCTools Interface 244
4.3. Project and Controller Data 246
4.4. Project 246
4.4.1. RMCTools Project 247
4.4.2. RMCTools Project Pane 247
4.5. Controller 249
4.5.1. New Controller Wizard: Welcome 249
4.5.2. Connection Path 250
4.5.3. Go Online, Go Offline 251
4.5.4. Using RUN/PROGRAM/Disabled Mode in RMCTools 251
4.5.5. Communication Statistics Window 252
4.5.6. Updating Flash 252
4.5.7. Resetting the RMC to Defaults 253
4.6. Modules 254
4.6.1. Modules List 254
4.6.2. RMC Hardware Configuration Dialog 254
4.7. Axes 255
4.7.1. Axis Tools 255
4.7.1.1. Axis Tools Window 255
4.7.1.2. Axis Status Registers Pane 256
4.7.1.3. Axis Parameters Pane 256
4.7.2. Axis Definitions Dialog 257
4.8. Command Tool 259
4.8.1. Command Tool 259
4.9. Plots 260
4.9.1. Plot Manager Overview 260

4.9.2. Plot Manager Elements 264
4.9.3. Plot Template Editor 267
4.9.4. Using Custom Plot Templates 268
4.9.5. Using XY Plots 271
4.9.6. Plot Quantity Selection Tool 274
4.10. Tuning 275
4.10.1. System Identification 275
4.10.2. Tuning Tools 276
4.11. Programming 279
4.11.1. Programming Folder Overview 279
4.11.2. Step Editor 279
4.11.3. Program Triggers 282
4.11.4. Task Monitor 286
4.11.5. Variable Table Editor 287
4.11.6. User Functions 289
4.11.7. Discrete I/O Configuration Window 292
4.11.8. Discrete I/O Monitor 294
4.12. Curve Tool 295
4.12.1. Curve Tool 295
4.12.2. Curve Properties 298
4.13. Address Maps 299
4.13.1. Address Maps Editor 299
4.13.2. Indirect Data Map Editor 300
4.14. Shortcut Sets 301
4.14.1. Shortcut Commands 301
4.15. Event Log 303
4.15.1. Event Log Monitor 303
4.15.2. Event Log Filtering 305
4.16. General Tools 307
4.16.1. Uploading and Downloading 307
4.16.2. RMCTools File Types 308
4.16.3. Error Icon and Error Bubble 309
4.16.4. Address Selection Tool (Single Register) 310
4.16.5. Output Window 311
4.16.6. Verify Results Window 311
4.16.7. Actuator View 311
4.16.8. RMCTools Options Dialog 312
4.16.9. Communication Log 313
4.16.10. Keyboard and Mouse Shortcuts 314
4.16.11. Copy and Paste 319
4.16.12. Find and Replace 320
4.16.13. Printing in RMCTools 321
4.16.14. Installing RMCTools 321
4.17. Wizards 323
4.17.1. Scale/Offset Wizard Overview 323
4.17.2. Autotuning Wizard: Welcome Page 323
4.17.3. Autotuning Wizard: Enter Move Parameters Page 324
4.17.4. Simulator Wizard 327
4.17.5. Firmware Update Wizard 328
4.18. Menu and Toolbars 329
4.18.1. RMCTools Menu Bar 329
4.18.2. Standard Toolbar 333
4.18.3. Shortcut Command Toolbar 335
4.18.4. RMCTools Status Bar 335
5. Programming 337

5.1. Programming Overview 337
5.2. Issuing Commands to the RMC 338
5.3. Tasks 342
5.4. Variables 345
5.5. Program Triggers 348
5.6. Downloading the Programming 353
5.7. Tag Names 353
5.8. Program Capacity and Time Usage 355
5.9. Programming Security 358
5.10. RMCTools Security Policy and Agreement 361
5.11. User Programs 362
5.11.1. User Programs Overview 362
5.11.2. Creating User Programs 364
5.11.3. Verifying User Programs 368
5.11.4. Running User Programs 369
5.11.5. Stopping User Programs 370
5.11.6. Labeling Steps 370
5.11.7. Exporting and Importing User Programs 371
5.11.8. Using Expressions for Commanded Axes 372
5.11.9. Link Types 373
5.11.9.1. Link Types Overview 373
5.11.9.2. Link Type: Immediate 374
5.11.9.3. Link Type: Jump 374
5.11.9.4. Link Type: Delay 375
5.11.9.5. Link Type: Wait For 376
5.11.9.6. Link Type: Conditional Jump 377
5.11.9.7. Link Type: End 379
5.12. Data Types 379
5.12.1. Data Types 379
5.12.2. BOOL Data Type 380
5.12.3. DINT Data Type 381
5.12.4. DWORD Data Type 381
5.12.5. REAL Data Type 381
5.13. Expressions 382
5.13.1. Expressions Overview 382
5.13.2. Assignment Expressions 383
5.13.3. Condition Expressions 385
5.13.4. Value Expressions 386
5.13.5. Local Variables in User Program Steps 387
5.13.6. Arrays 389
5.13.7. Operators 390
5.13.8. Keywords 392
5.13.9. IF Statement 393
5.13.10. Constants 394
5.13.11. Comments 395
5.13.12. Limitations of 32-bit Numbers 395
5.13.13. Troubleshooting Expressions 396
5.14. Functions 397
5.14.1. Functions Overview 397
5.14.2. Standard Functions 397
5.14.2.1. Standard Functions 397
5.14.2.2. ABS Function 399
5.14.2.3. ACOS Function 400
5.14.2.4. ADDR_OFS Function 400
5.14.2.5. ASHR Function 401
5.14.2.6. ASIN Function 401
5.14.2.7. ATAN Function 402
5.14.2.8. CEIL Function 402

5.14.2.9. COPY Function ... 403
5.14.2.10. COS Function ... 405
5.14.2.11. COSH Function ... 406
5.14.2.12. CRV_EXISTS Function ... 406
5.14.2.13. CRV_FIRST_X Function ... 406
5.14.2.14. CRV_INTERP Functions ... 407
5.14.2.15. CRV_LAST_X Function ... 408
5.14.2.16. DINT_TO_DWORD Function ... 409
5.14.2.17. DINT_TO_REAL Function ... 409
5.14.2.18. DWORD_TO_DINT Function ... 409
5.14.2.19. EXP Function ... 410
5.14.2.20. FILL Function ... 410
5.14.2.21. FLOOR Function ... 412
5.14.2.22. LENGTH Function ... 412
5.14.2.23. LIMIT Function ... 413
5.14.2.24. LOG Function ... 413
5.14.2.25. LN Function ... 414
5.14.2.26. LOG_EVENT Function ... 414
5.14.2.27. MAX Function ... 415
5.14.2.28. MIN Function ... 415
5.14.2.29. MROUND Function ... 415
5.14.2.30. POLY Function ... 416
5.14.2.31. REAL_TO_DINT Function ... 416
5.14.2.32. REG_REAL, REG_DINT, REG_DWORD Functions ... 417
5.14.2.33. ROL Function ... 417
5.14.2.34. ROR Function ... 418
5.14.2.35. ROUND Function ... 418
5.14.2.36. SEL Function ... 419
5.14.2.37. SHL Function ... 419
5.14.2.38. SHR Function ... 420
5.14.2.39. SIGNUM Function ... 420
5.14.2.40. SIN Function ... 421
5.14.2.41. SINH Function ... 421
5.14.2.42. SQRT Function ... 421
5.14.2.43. TAN Function ... 422
5.14.2.44. TANH Function ... 422
5.14.2.45. TRUNC Function ... 423
5.14.2.46. TRUNC_REAL Function ... 423
5.14.3. User Functions ... 423
5.14.3.1. User Functions ... 423
5.14.3.2. Declaring Variables and Parameters in User Functions ... 426
5.14.3.3. Example User Functions ... 428
5.15. Discrete I/O ... 430
5.15.1. Discrete I/O Overview ... 430
5.15.2. Using Discrete I/O ... 432
5.15.3. Configuring Discrete I/O ... 434
5.16. Programming Examples and Tips ... 436
5.16.1. Programming Examples and Tips ... 436
5.16.2. Example: A Basic User Program ... 436
5.16.3. Example: A Simple User Program ... 440
5.16.4. Example: Closed Loop Motion on Startup ... 441
5.16.5. Example: Jogging an Axis ... 443
5.16.6. Example: Creating a Timer ... 447
5.16.7. Example: Time-out ... 448
5.16.8. Example: Using Arrays ... 449
5.16.9. Application Tip: Emergency Stops ... 451
6. Communication ... 455
6.1. RMC Communications Overview ... 455
6.2. Communication Statistics ... 458

6.3. Monitor Port — USB or RS-232 458
6.4. Communicating with HMIs 461
6.5. Discrete I/O for Communications 464
6.6. Command Request and Acknowledge Bits 464
6.7. Ethernet 466
6.7.1. Ethernet Communications Overview 467
6.7.2. Using Ethernet with RMCTools 469
6.7.3. Using the RMC's PLC Ethernet Emulation 470
6.7.4. Communicating Directly over TCP 471
6.7.5. Communicating Directly over UDP 476
6.7.6. Ethernet Link/Act LED 480
6.7.7. Troubleshooting RMCTools Ethernet Connection 481
6.7.8. Ethernet Setup Topics 484
6.7.8.1. Setting Up the RMC Ethernet 484
6.7.8.2. RMC Ethernet Protocols 485
6.7.9. Ethernet Informational Topics 488
6.7.9.1. Setting Up a Standalone TCP/IP Control Network 489
6.7.9.2. Understanding Ethernet IP Addressing 490
6.7.9.3. Ethernet MAC Address 492
6.7.10. Application Protocols 492
6.7.10.1. Modbus/TCP 492
6.7.10.2. CSP 494
6.7.10.3. FINS/UDP 494
6.7.10.4. Mitsubishi Procedure Exist Protocol 495
6.7.10.5. DMCP 502
6.7.10.6. EtherNet/IP 502
6.7.10.6.1. EtherNet/IP Overview 502
6.7.10.6.2. Setting Up an EtherNet/IP I/O Connection 504
6.7.10.6.3. Using an EtherNet/IP I/O Connection 511
6.7.10.6.4. Handling Broken EtherNet/IP I/O Connections 516
6.7.10.6.5. Troubleshooting EtherNet/IP I/O 520
6.7.10.6.6. Multiple Types of EtherNet/IP I/O Connections 522
6.7.10.6.7. EtherNet/IP Performance Considerations 524
6.7.10.6.8. Using EtherNet/IP Explicit Messaging 528
6.7.10.7. PROFINET 531
6.7.10.7.1. PROFINET Overview 531
6.7.10.7.2. Setting up a PROFINET I/O Connection 532
6.7.10.7.3. Using a PROFINET I/O Connection 535
6.7.10.7.4. Using PROFINET Record Data 539
6.7.10.7.5. Handling Broken PROFINET Connections 541
6.7.10.7.6. Troubleshooting PROFINET 544
6.7.11. Other Protocols 544
6.7.11.1. Simple Network Management Protocol (SNMP) 544
6.8. PROFIBUS 545
6.8.1. PROFIBUS-DP Slave Communications Overview 546
6.8.2. Troubleshooting PROFIBUS 547
6.8.3. Configuration 548
6.8.3.1. PROFIBUS Configuration 548
6.8.3.2. Configuring a PROFIBUS-DP Network with COM PROFIBUS 550
6.8.3.3. Configuring a PROFIBUS-DP Network with SST Profibus Configuration 552
6.8.3.4. Configuring a PROFIBUS-DP Network with SyCon 553
6.8.4. Using I/O Modes 555
6.8.4.1. PROFIBUS Mode: I/O Modes 555
6.8.5. Using Basic/Enhanced Modes (RMC75P Only) 559
6.8.5.1. Basic/Enhanced PROFIBUS Modes (RMC75P Only) 559
6.8.5.2. PROFIBUS Mode: Basic 560
6.8.5.3. PROFIBUS Mode: Basic+ 568
6.8.5.4. PROFIBUS Mode: Enhanced 577
6.8.5.5. PROFIBUS Mode: Enhanced+ 588
6.9. Serial (RS-232/485) (RMC70) 599

6.9.1. Serial Communications Overview 599
6.9.2. Using Serial Communications 601
6.9.3. Configuration and Wiring 601
6.9.3.1. Line Drivers: RS-232/485 601
6.9.3.2. Configuring the RMC75S Serial Communications 602
6.9.3.3. Serial Network Topologies 603
6.9.3.4. RS-232 Wiring for the RMC75 605
6.9.3.5. RS-485 Wiring for the RMC75S 606
6.9.3.6. RS-485 Termination and Biasing 608
6.9.4. RMC75S Serial Protocols 610
6.9.4.1. DF1 Protocol (Full- and Half-Duplex) 610
6.9.4.2. Mitsubishi Bidirectional Protocol 611
6.9.4.3. Modbus/RTU Protocol 613
6.10. Using Master Controllers 614
6.10.1. Using Allen-Bradley Controllers via Message Block 614
6.10.2. Using Allen-Bradley Controllers via EtherNet/IP I/O 621
6.10.3. Using Logix Designer Export Components 635
6.10.4. Using AutomationDirect DL-series PLCs 642
6.10.5. Using Factory Talk View with the RMC 643
6.10.6. Using GE PLCs with the RMC 649
6.10.7. Using LabVIEW with RMCs 649
6.10.8. Using Mitsubishi PLCs with the RMC 650
6.10.9. Using Schneider Electric (Modicon) PLCs via Modbus/TCP 650
6.10.10. Using Schneider Electric PLCs via EtherNet/IP I/O 651
6.10.11. Using Omron Controllers via the FINS Protocol 661
6.10.12. Using Omron Controllers via EtherNet/IP I/O 662
6.10.13. Using Siemens S7 PLCs via PROFINET 684
6.10.14. Using Siemens S7 PLCs via PROFIBUS 699
6.10.15. Using Wonderware with the RMC 699
6.10.16. RMCLink ActiveX Control and .NET Assembly 705
6.10.17. Using Other Master Controllers with the RMC 706
6.11. Address Maps 707
6.11.1. Address Maps 707
6.11.2. Indirect Data Map 710
6.11.3. Modbus Address Map 712
6.11.4. FINS Address Map 714
6.11.5. DF1 Address Map 715
6.11.6. PROFINET Data Records Address Map 716
6.11.7. IEC Address Map 717

7. Hardware 719

7.1. RMC Hardware Overview 719
7.2. RMC75 720

7.2.5.3. A2 Expansion Module ... 741
7.2.5.4. AP2 Expansion Module ... 743
7.2.5.5. D8 Expansion Module ... 744
7.2.5.6. Q1 Expansion Module ... 745
7.3. RMC150 ... 747
7.3.1. RMC150 Hardware Overview ... 747
7.3.2. RMC150 Part Numbering ... 749
7.3.3. RMC150E/RMC151E CPU Module ... 750
7.3.4. Analog Modules ... 754
7.3.4.1. Analog (H) Module (RMC150) ... 754
7.3.4.2. Analog (A) Module (RMC150) ... 756
7.3.4.3. Analog (G) Module (RMC150) ... 758
7.3.5. MDT Module ... 760
7.3.5.1. MDT (M) Module (RMC150) ... 760
7.3.6. SSI Module ... 762
7.3.6.1. SSI (S) Module (RMC150) ... 762
7.3.7. Quadrature Module ... 764
7.3.7.1. Quadrature (Q) Module (RMC150) ... 764
7.3.8. Resolver Module ... 767
7.3.8.1. Resolver (R) Module (RMC150) ... 767
7.3.8.2. Resolver (RW) Module (RMC150) ... 769
7.3.9. DI/O Module ... 771
7.3.9.1. DI/O Module (RMC150) ... 771
7.3.10. UI/O Module ... 772
7.3.10.1. UI/O Module (RMC150) ... 772
7.3.10.2. Configuring UI/O High-Speed Channels ... 775
7.3.11. PROFIBUS Module ... 779
7.3.11.1. PROFIBUS Module (RMC150) ... 779
7.4. RMC200 ... 780
7.4.1. RMC200 Hardware Overview ... 780
7.4.2. Comparing RMC200 Lite and Standard ... 784
7.4.3. RMC200 Part Numbering ... 785
7.4.4. Feature Key ... 787
7.4.5. Using the RMC200 SD Card ... 789
7.4.6. Bases ... 792
7.4.6.1. B5L Lite Base (RMC200) ... 792
7.4.6.2. B7L Lite Base (RMC200) ... 792
7.4.6.3. B5 Standard Base (RMC200) ... 793
7.4.6.4. B7 Standard Base (RMC200) ... 794
7.4.6.5. B11 Standard Base (RMC200) ... 795
7.4.6.6. B15 Standard Base (RMC200) ... 795
7.4.7. Power Supplies ... 796
7.4.7.1. PS4D Power Supply (RMC200) ... 796
7.4.7.2. PS6D Power Supply (RMC200) ... 797
7.4.8. CPU Modules ... 798
7.4.8.1. CPU20L (RMC200) ... 798
7.4.8.2. CPU40 (RMC200) ... 804
7.4.8.3. CPU20L and CPU40 Display Screen ... 809
7.4.8.4. RMC200 Real-Time Clock ... 810
7.4.9. I/O Modules ... 811
7.4.9.1. CA4 Module (RMC200) ... 811
7.4.9.2. CV8 Module (RMC200) ... 814
7.4.9.3. LC8 Module (RMC200) ... 816
7.4.9.4. Configuring the LC8 Module ... 818
7.4.9.5. S8 Module (RMC200) ... 819
7.4.9.6. Configuring S8 Channels ... 822
7.4.9.7. A8 Module (RMC200) ... 823
7.4.9.8. Q4 Module (RMC200) ... 826
7.4.9.9. U14 Module (RMC200) ... 830
7.4.9.10. Configuring U14 High-Speed Channels ... 839

7.4.9.11. D24 Module (RMC200) 842
7.5. General 846
7.5.1. Control Output 846
7.5.2. Enable Output 847
7.5.3. Fault Input 848
7.6. Accessories 849
7.6.1. Quadrature Cable 849
7.6.2. SD Card R2-MEM-SD-1G (for RMC200) 851
7.6.3. VC2124 and VC2100 Voltage-to-Current Converters 851
7.7. Agency Compliance 854

8. Command Reference 857

8.1. RMC Commands Overview 857
8.2. RMC Commands 859
8.3. General Commands 863
8.3.1. Command: No-op (0) 863
8.3.2. Command: Clear Faults (4) 863
8.3.3. Command: Enable Controller (7) 864
8.3.4. Command: Fault Controller (8) 865
8.3.5. Command: Set Enable Output (67) 865
8.3.6. Command: Enable/Disable Axis (97) 866
8.3.7. Command: RUN Mode (98) 866
8.3.8. Command: PROGRAM Mode (99) 867
8.4. Motion Commands 867
8.4.1. Motion Commands 867
8.4.2. Stops 869
8.4.2.1. Command: Stop (Closed Loop) (6) 869
8.4.2.2. Command: Stop (Open Loop) (22) 870
8.4.2.3. Command: Hold Current Position (5) 871
8.4.2.4. Command: Closed Loop Halt (1) 872
8.4.2.5. Command: Open Loop Halt (2) 873
8.4.2.6. Command: Direct Output Halt (3) 874
8.4.3. Open Loop 874
8.4.3.1. Command: Direct Output (9) 874
8.4.3.2. Command: Open Loop Rate (10) 876
8.4.3.3. Command: Open Loop Absolute (11) 876
8.4.3.4. Command: Open Loop Relative (12) 879
8.4.4. Synchronized 880
8.4.4.1. Command: Sync Move Absolute (13) 880
8.4.4.2. Command: Sync Move Relative (14) 882
8.4.4.3. Command: Sync Stop (17) 884
8.4.5. Point-to-Point 884
8.4.5.1. Command: Move Absolute (20) 884
8.4.5.2. Command: Move Relative (21) 887
8.4.5.3. Command: Quick Move Absolute (15) 889
8.4.5.4. Command: Quick Move Relative (16) 891
8.4.5.5. Command: Time Move Absolute (23) 892
8.4.5.6. Command: Time Move Relative (24) 893
8.4.5.7. Command: Advanced Time Move Absolute (26) 895
8.4.5.8. Command: Advanced Time Move Relative (27) 897
8.4.5.9. Command: Move Absolute (I-PD) (28) 898
8.4.5.10. Command: Move Relative (I-PD) (29) 898
8.4.6. Gearing 899
8.4.6.1. Command: Gear Absolute (25) 899
8.4.6.2. Command: Gear Position (Clutch by Time) (30) 902
8.4.6.3. Command: Gear Velocity (31) 904
8.4.6.4. Command: Gear Position (Clutch by Rate) (39) 905
8.4.6.5. Command: Gear Position (Clutch by Distance) (32) 907
8.4.6.6. Command: Phasing (34) 915

8.4.6.7. Command: Geared Slave Offset (35) ... 917
8.4.6.8. Command: Advanced Gear Move (33) ... 919
8.4.6.9. Command: Track Position (57) ... 923
8.4.6.10. Command: Track Position (I-PD) (58) ... 926
8.4.7. Specialty ... 927
8.4.7.1. Command: Speed at Position (36) ... 927
8.4.7.2. Command: Sine Start (72) ... 928
8.4.7.3. Command: Sine Stop (73) ... 933
8.4.7.4. Command: Change Master (79) ... 935
8.4.7.5. Command: Change Target Parameter (80) ... 936
8.4.7.6. Command: Curve Add (82) ... 939
8.4.7.7. Command: Curve Delete (83) ... 942
8.4.7.8. Command: Curve Delete Except (84) ... 943
8.4.7.9. Command: Curve Delete All (85) ... 943
8.4.7.10. Command: Curve Start (86) ... 944
8.4.7.11. Command: Curve Start Advanced (88) ... 947
8.4.8. Velocity ... 954
8.4.8.1. Command: Move Velocity (37) ... 954
8.4.8.2. Command: Move Velocity (I-PD) (38) ... 955
8.4.9. Transitions ... 956
8.4.9.1. Command: Transition Disable (55) ... 956
8.4.9.2. Command: Transition Rate (56) ... 957
8.5. Pressure/Force Control ... 959
8.5.1. Standard ... 960
8.5.1.1. Command: Hold Current Pressure/Force (19) ... 960
8.5.1.2. Command: Stop Pressure/Force (43) ... 961
8.5.1.3. Command: Ramp Pressure/Force (Rate) (18) ... 962
8.5.1.4. Command: Ramp Pressure/Force (S-Curve) (41) ... 963
8.5.1.5. Command: Ramp Pressure/Force (Linear) (42) ... 964
8.5.1.6. Command: Enter Pressure/Force Control (Auto) (44) ... 966
8.5.1.7. Command: Enter Pressure/Force Control (Time) (45) ... 968
8.5.1.8. Command: Enter Pressure/Force Control (Rate) (46) ... 970
8.5.2. Pressure/Force Limit ... 973
8.5.2.1. Command: Set Pressure/Force Limit Mode (40) ... 973
8.5.3. Specialty ... 975
8.5.3.1. Command: Gear Absolute Prs/Frc (59) ... 975
8.5.3.2. Command: Transition Disable Prs/Frc (63) ... 978
8.5.3.3. Command: Transition Rate (Prs/Frc) (64) ... 978
8.5.3.4. Command: Sine Start (Prs/Frc) (76) ... 980
8.5.3.5. Command: Sine Stop (Prs/Frc) (77) ... 985
8.5.3.6. Command: Change Target Parameter (Prs/Frc) (81) ... 987
8.5.3.7. Command: Curve Start (Prs/Frc) (87) ... 989
8.5.3.8. Command: Curve Start Advanced (Prs/Frc) (89) ... 990
8.6. Set Parameters ... 991
8.6.1. Command: Offset Position (47) ... 991
8.6.2. Command: Set Target Position (48) ... 992
8.6.3. Command: Set Actual Position (49) ... 992
8.6.4. Set Actual Pressure/Force (65) ... 993
8.6.5. Command: Set Pos/Vel Ctrl Mode (68) ... 993
8.6.6. Command: Feed Forward Adjust (69) ... 994
8.6.7. Command: Integrator Adjust (70) ... 995
8.6.8. Command: Set Integrator Mode (71) ... 996
8.6.9. Command: Select Gain Set (75) ... 997
8.6.10. Command: Set Control Direction (96) ... 998
8.6.11. Command: Read Register (111) ... 998
8.6.12. Command: Write Register (112) ... 999
8.7. System ... 1000
8.7.1. Command: Arm Home (50) ... 1000
8.7.2. Command: Disarm Home (51) ... 1002
8.7.3. Command: Arm Registration (52) ... 1003

8.7.4. Command: Disarm Registration (53) 1004
8.7.5. Command: Learn Z Alignment (54) 1005
8.7.6. Command: Pause/Resume Log (95) 1005
8.7.7. Command: Update Flash (110) 1006
8.7.8. Command: Arm Event Timer (105) 1007
8.7.9. Command: Disarm Event Timer (106) 1009
8.7.10. Command: Save Controller Image (120) 1009
8.7.11. Command: Restore Controller Image (121) 1011
8.8. Programming 1012
8.8.1. Command: Start Task (90) 1012
8.8.2. Command: Stop Task (91) 1013
8.8.3. Command: Set Discrete Output (60) 1014
8.8.4. Command: Clear Discrete Output (61) 1015
8.8.5. Command: Toggle Discrete Output (62) 1016
8.9. Plots 1017
8.9.1. Command: Start Plot (100) 1017
8.9.2. Command: Stop Plot (101) 1018
8.9.3. Command: Trigger Plot (102) 1018
8.9.4. Command: Rearm Plot (103) 1019
8.9.5. Command: Enable/Disable Plot Trigger (104) 1020
8.10. Step Editor Commands 1020
8.10.1. Command: Expression (113) 1020

9. Register Reference 1025

9.1. Registers 1025
9.2. Register Descriptions 1026
9.2.1. Axis Definitions 1026
9.2.1.1. Axis Definition Registers 1026
9.2.2. Axis Status Registers 1030
9.2.2.1. Axis Status Registers Overview 1030
9.2.2.2. Common 1030
9.2.2.2.1. Status Bits Register 1030
9.2.2.2.2. Error Bits Register 1037
9.2.2.2.3. Last Error Number Register 1045
9.2.2.2.4. Read Response 1045
9.2.2.3. Feedback 1045
9.2.2.3.1. Actual Position 1045
9.2.2.3.2. Actual Velocity 1046
9.2.2.3.3. Actual Acceleration 1047
9.2.2.3.4. Actual Jerk 1049
9.2.2.3.5. Actual Pressure/Force 1049
9.2.2.3.6. Actual Pressure/Force Rate 1051
9.2.2.3.7. Counts 1052
9.2.2.3.8. Voltage 1053
9.2.2.3.9. Current 1055
9.2.2.3.10. Raw Counts 1056
9.2.2.3.11. Channel A, B Raw Counts 1058
9.2.2.3.12. Channel A, B Current 1060
9.2.2.3.13. Channel A, B Voltage 1061
9.2.2.3.14. Channel A, B Force 1063
9.2.2.3.15. Channel A, B Pressure 1064
9.2.2.3.16. Channel A, B Acceleration 1065
9.2.2.3.17. Custom Counts 1066
9.2.2.3.18. Custom Error Bits 1067
9.2.2.3.19. Transducer Status A and B 1068
9.2.2.3.20. Actual Position (Unfiltered) 1078
9.2.2.3.21. Actual Velocity (Unfiltered) 1078
9.2.2.3.22. Actual Acceleration (Unfiltered) 1079
9.2.2.3.23. Actual Jerk (Unfiltered) 1080

9.2.2.3.24. Actual Position (Control) 1080
9.2.2.3.25. Actual Velocity (Control) 1081
9.2.2.3.26. Actual Acceleration (Control) 1082
9.2.2.3.27. Actual Jerk (Control) 1083
9.2.2.3.28. Actual Pressure/Force (Unfiltered) 1084
9.2.2.3.29. Actual Pressure/Force Rate (Unfiltered) 1084
9.2.2.3.30. Actual Pressure/Force (Control) 1085
9.2.2.3.31. Actual Pressure/Force Rate (Control) 1086
9.2.2.3.32. Millivolts/Volt 1087
9.2.2.3.33. Millivolt Input 1088
9.2.2.3.34. Effective Exciter Voltage 1088
9.2.2.4. Output 1089
9.2.2.4.1. Control Output 1089
9.2.2.5. Primary Control 1090
9.2.2.5.1. Position Error 1090
9.2.2.5.2. Velocity Error 1090
9.2.2.5.3. Proportional Output Term 1091
9.2.2.5.4. Integral Output Term 1091
9.2.2.5.5. Double Differential Output Term 1091
9.2.2.5.6. Differential Output Term 1092
9.2.2.5.7. Triple Differential Output Term 1092
9.2.2.5.8. Acceleration Feed Forward Term 1093
9.2.2.5.9. Velocity Feed Forward Term 1093
9.2.2.5.10. Jerk Feed Forward Term 1093
9.2.2.5.11. PFID Output 1094
9.2.2.5.12. Current Control Mode 1095
9.2.2.5.13. Next Pos/Vel Control Mode 1095
9.2.2.5.14. Current Integrator Mode 1096
9.2.2.5.15. Current Gain Set 1097
9.2.2.6. Secondary Control 1097
9.2.2.6.1. Pressure/Force Error 1097
9.2.2.6.2. Pressure/Force Proportional Term 1098
9.2.2.6.3. Pressure/Force Integral Term 1098
9.2.2.6.4. Pressure/Force Differential Term 1099
9.2.2.6.5. Pressure/Force Feed Forward Term 1099
9.2.2.6.6. Pressure/Force Rate Feed Forward Term 1100
9.2.2.7. Target 1100
9.2.2.7.1. Command Position 1100
9.2.2.7.2. Command Velocity 1101
9.2.2.7.3. Target Position 1101
9.2.2.7.4. Target Velocity 1102
9.2.2.7.5. Target Acceleration 1102
9.2.2.7.6. Target Jerk 1103
9.2.2.7.7. Cycles 1103
9.2.2.7.8. Command Pressure/Force 1104
9.2.2.7.9. Target Pressure/Force 1104
9.2.2.7.10. Target Pressure/Force Rate 1104
9.2.2.7.11. Cycles (Pressure or Force) 1105
9.2.2.8. Home/Registration 1105
9.2.2.8.1. Registration 0 Position 1105
9.2.2.8.2. Registration 1 Position 1106
9.2.2.8.3. Encoder Status 1106
9.2.3. Axis Parameter Registers 1114
9.2.3.1. Axis Parameter Registers Overview 1114
9.2.3.2. Feedback 1115
9.2.3.2.1. Position Scale 1115
9.2.3.2.2. Position Offset 1116
9.2.3.2.3. Velocity Scale 1117
9.2.3.2.4. Velocity Offset 1117
9.2.3.2.5. Velocity Deadband 1118
9.2.3.2.6. Acceleration Scale 1119
9.2.3.2.7. Acceleration Offset 1120

9.2.3.2.8. Channel A, B Acceleration Scale 1120
9.2.3.2.9. Channel A, B Acceleration Offset 1121
9.2.3.2.10. Position Unwind 1122
9.2.3.2.11. Count Unwind 1123
9.2.3.2.12. Count Offset 1124
9.2.3.2.13. Linear/Rotary 1125
9.2.3.2.14. Stop Threshold 1126
9.2.3.2.15. Noise Error Rate 1127
9.2.3.2.16. Display Units 1128
9.2.3.2.17. Custom Units 1130
9.2.3.2.18. Primary Input Bits Register 1131
9.2.3.2.19. Filtering - RMC70/150 1132
9.2.3.2.20. Filtering - RMC200 1140
9.2.3.2.21. Modeling 1152
9.2.3.2.22. Pressure & Force 1158
9.2.3.2.23. Limit Inputs 1167
9.2.3.2.24. Transducer Specific 1172
9.2.3.2.25. Custom 1210
9.2.3.3. Simulator 1212
9.2.3.3.1. Simulate Mode 1212
9.2.3.3.2. System Gain (Simulator) 1212
9.2.3.3.3. Positive System Gain (Simulator) 1212
9.2.3.3.4. Negative System Gain (Simulator) 1213
9.2.3.3.5. Time Constant (Simulator) 1214
9.2.3.3.6. Natural Frequency (Simulator) 1214
9.2.3.3.7. Damping Factor (Simulator) 1215
9.2.3.3.8. Positive Physical Limit (Simulator) 1216
9.2.3.3.9. Negative Physical Limit (Simulator) 1217
9.2.3.3.10. Output Deadband (Simulator) 1217
9.2.3.3.11. Output Null (Simulator) 1218
9.2.3.3.12. Weight (Simulator) 1218
9.2.3.3.13. Maximum Force (Simulator) 1219
9.2.3.3.14. Maximum Compression (Simulator) 1220
9.2.3.3.15. Simulator Configuration Register 1220
9.2.3.4. Position/Velocity Control 1221
9.2.3.4.1. In Position Tolerance 1221
9.2.3.4.2. Position Error Tolerance 1221
9.2.3.4.3. At Velocity Tolerance 1222
9.2.3.4.4. Velocity Error Tolerance 1222
9.2.3.4.5. Default Integrator Mode 1223
9.2.3.4.6. Proportional Gain 1225
9.2.3.4.7. Integral Gain 1226
9.2.3.4.8. Differential Gain 1228
9.2.3.4.9. Velocity Feed Forward 1230
9.2.3.4.10. Velocity Feed Forward (Positive) 1231
9.2.3.4.11. Velocity Feed Forward (Negative) 1233
9.2.3.4.12. Acceleration Feed Forward 1234
9.2.3.4.13. Jerk Feed Forward 1235
9.2.3.4.14. Double Differential Gain 1236
9.2.3.4.15. Active Damping Proportional Gain 1238
9.2.3.4.16. Triple Differential Gain 1239
9.2.3.4.17. Active Damping Differential Gain 1240
9.2.3.4.18. Gain Sets 1241
9.2.3.4.19. Symmetrical/Ratioed 1243
9.2.3.4.20. High-Order Control 1244
9.2.3.4.21. Default Pos/Vel Control Mode 1245
9.2.3.4.22. Primary Control Configuration Register 1246
9.2.3.5. Pressure/Force Control 1246
9.2.3.5.1. At Pressure/Force Tolerance 1246
9.2.3.5.2. Pressure/Force Error Tolerance 1247
9.2.3.5.3. Pressure/Force Proportional Gain 1247
9.2.3.5.4. Pressure/Force Integral Gain 1248

9.2.3.5.5. Pressure/Force Differential Gain ... 1250
9.2.3.5.6. Pressure/Force Feed Forward ... 1251
9.2.3.5.7. Pressure/Force Rate Feed Forward ... 1251
9.2.3.5.8. Pressure/Force Orientation ... 1252
9.2.3.5.9. Secondary Control Configuration Register ... 1253
9.2.3.6. Output ... 1254
9.2.3.6.1. Invert Output Polarity ... 1254
9.2.3.6.2. Output Bias ... 1255
9.2.3.6.3. Output Deadband ... 1256
9.2.3.6.4. Deadband Tolerance ... 1258
9.2.3.6.5. Output Filter ... 1260
9.2.3.6.6. Output Limit ... 1260
9.2.3.6.7. Output Scale ... 1261
9.2.3.6.8. Output Type ... 1262
9.2.3.6.9. Control Range ... 1266
9.2.3.6.10. Output at 100% ... 1267
9.2.3.6.11. Output at 0% ... 1267
9.2.3.6.12. Output at -100% ... 1268
9.2.3.6.13. Positive Output Limit ... 1269
9.2.3.6.14. Negative Output Limit ... 1270
9.2.3.6.15. Custom Output Units ... 1271
9.2.3.6.16. Output Gain ... 1271
9.2.3.6.17. Unidirectional Mode ... 1272
9.2.3.6.18. Output Bits Configuration Register ... 1274
9.2.3.6.19. Fault Input Source ... 1275
9.2.3.6.20. Fault Input Polarity ... 1276
9.2.3.6.21. Fault Input Configuration Register ... 1277
9.2.3.6.22. Enable Output Source ... 1277
9.2.3.6.23. Enable Output Behavior ... 1278
9.2.3.6.24. Enable Output Configuration ... 1278
9.2.3.6.25. Valve Linearization Type ... 1279
9.2.3.6.26. Valve Linearization Curve ID ... 1280
9.2.3.6.27. Knee Command Input Knee Command Voltage ... 1281
9.2.3.6.28. Knee Flow Output Knee Flow Percentage ... 1281
9.2.3.7. Target ... 1282
9.2.3.7.1. Positive Travel Limit ... 1282
9.2.3.7.2. Negative Travel Limit ... 1283
9.2.3.7.3. Positive Pressure/Force Limit ... 1284
9.2.3.7.4. Negative Pressure/Force Limit ... 1285
9.2.3.7.5. Requested Jerk ... 1285
9.2.3.7.6. Target Type ... 1288
9.2.3.8. Halts ... 1291
9.2.3.8.1. Auto Stop Configuration ... 1291
9.2.3.8.2. Halt Group Number ... 1293
9.2.3.8.3. Closed Loop Halt Deceleration ... 1294
9.2.3.8.4. Open Loop Halt Ramp ... 1295
9.2.4. Communication Registers ... 1295
9.2.4.1. Ethernet Status ... 1295
9.2.4.2. Ethernet Configuration Bits ... 1296
9.2.4.3. I/O Connection Status ... 1297
9.2.4.4. I/O Connection PLC Status ... 1298
9.2.4.5. PROFIBUS Connection Status ... 1299
9.2.5. Task Registers ... 1300
9.2.5.1. Task Status ... 1300
9.2.5.2. Current Program ... 1300
9.2.5.3. Current Step ... 1301
9.2.5.4. Current Axis ... 1301
9.2.5.5. Current Program/Step ... 1302
9.2.6. Controller Registers ... 1302
9.2.6.1. Controller Status ... 1303
9.2.6.2. Controller Tags ... 1304
9.2.6.3. Controller Loop Time Status Registers ... 1304

9.2.6.4. System Time Registers 1306
9.2.6.5. Loader Command (RMC75/150) Reset Command (RMC200) 1307
9.2.7. Variables 1308
9.2.7.1. Variable Attributes 1308
9.3. Address Formats 1310
9.3.1. Address Formats Overview 1310
9.3.2. DF1 (Allen Bradley) Addressing 1311
9.3.3. IEC-61131 Addressing 1314
9.3.4. FINS (Omron) Addressing 1315
9.3.5. Modbus Addressing 1318
9.4. RMC75 Register Map 1321
9.4.1. RMC75 Register Map 1321
9.5. RMC150 Register Map 1376
9.5.1. RMC150 Register Map 1376
9.6. RMC200 Register Map 1414
9.6.1. RMC200 Register Map 1414

10. Wiring and Installation 1449

10.1. Wiring Guidelines 1449
10.2. RMC75 1450
10.2.1. RMC75 Mounting Instructions 1450
10.2.2. RMC75E Wiring 1452
10.2.3. RMC75S Wiring 1453
10.2.4. RMC75P Wiring 1453
10.2.5. AA Wiring 1453
10.2.6. MAx Wiring 1455
10.2.7. QAx Wiring 1459
10.2.8. A2 Wiring 1462
10.2.9. AP2 Wiring 1463
10.2.10. D8 Wiring 1464
10.2.11. Q1 Wiring 1467
10.3. RMC150 1469
10.3.1. RMC150 Mounting Instructions 1469
10.3.2. RMC150E CPU Module Wiring 1470
10.3.3. RMC150 Control Output (Drive) Wiring 1472
10.3.4. RMC150 MDT Wiring 1472
10.3.5. RMC150 SSI Wiring 1474
10.3.6. RMC150 Quadrature Wiring 1475
10.3.7. RMC150 Analog Input Wiring 1477
10.3.8. RMC150 Resolver Wiring 1481
10.3.9. RMC150 Discrete I/O Wiring 1482
10.3.10. RMC150 UI/O Wiring 1485
10.4. RMC200 1490
10.4.1. RMC200 Mounting Instructions 1490
10.4.2. PS4D and PS6D Wiring 1494
10.4.3. CPU40 Wiring 1495
10.4.4. CA4 Wiring 1496
10.4.5. CV8 Wiring 1499
10.4.6. S8 Wiring 1501
10.4.7. LC8 Wiring 1505
10.4.8. A8 Wiring 1510
10.4.9. Q4 Wiring 1514
10.4.10. U14 Wiring 1518
10.4.11. D24 Wiring 1528

11. Troubleshooting 1537

11.1. Troubleshooting Overview 1537
11.2. Error Codes 1538
11.3. Technical Support 1555

12. Index 1557

13. 1558

14. 1582

1. Introducing the RMC Family

RMCTools and RMC Controllers Help

The RMCTools software is for setting up, tuning, programming and troubleshooting the RMC75, RMC150, and RMC200 motion controllers. RMCTools does not support the RMC100, which requires Delta's RMCWin software.

The RMCTools help provides the most complete and up-to-date information available for the RMC75, RMC150, RMC200 and RMCTools. To ensure you have the latest help file, download the latest version of RMCTools from Delta's website www.deltamotion.com.

The RMCTools help is intended to be a reference. For a step-by-step startup procedure, see the Startup Guide that shipped with your controller:

- RMC75 Startup Guide
- RMC150 Startup Guide
- RMC200 Startup Guide

The Startup Guides can also be accessed at deltamotion.com/dloads/.

Video Tutorials

Troubleshooting

See the Troubleshooting topic.

Customer Support

Delta support is ready to assist you. Call if you have questions or need help:

Telephone:	360-254-8688 (24-hour emergency support available)
Fax:	360-254-5435
E-mail:	support@deltamotion.com
Website:	http://www.deltamotion.com

Help in PDF Format

A manual containing the same information as the RMCTools help is available as a Portable Document Format (PDF) file from Delta's website .

1.1. RMC Family Motion Controllers

The RMC family of motion controllers consists of the RMC75, RMC150, RMC200, and the older RMC100. The RMC family brings the benefits of high-performance, easy-to-use motion control to a wide range of industrial applications. The RMCTools software supports the RMC75, RMC150, and RMC200. The legacy RMC100 requires Delta's RMCWin software.



RMC75: 1-2 Axes



RMC150: Up to 8 Axes



RMC200: Up to 50 Axes

The RMC75, RMC150, and RMC200 share nearly all the same control features, and are configured and programmed in the same way via RMCTools. If you know how to set up and configure one controller, you can configure the others just as easily!

To compare the RMC controllers, see the Hardware Overview topic.

For hardware details and specifications see the RMC75, RMC150, and RMC200 topics.

1.2. Basics of Operation

The RMC is a high-performance motion controller. It has many advanced features, yet is easy to use and can be set up quickly due to features such as Scale and Offset wizards and Autotuning.

This topic describes the most basic concepts of using the RMC.

Motion

Motion control falls into two camps: *open loop* and *closed loop*.

Open loop means that a control signal (called Control Output in the RMC) is given to the system to move it without regard to the feedback of the system, such as position, velocity, pressure, etc. Open loop control is not very precise.

Closed loop means that the controller uses the feedback to determine how much Control Output to give to the system to move it to the desired position. Closed loop control can be very precise if the system has high-resolution feedback and a good actuator.

The RMC provides both open loop and closed loop control.

Open Loop control is fairly simple. Just issue an Open Loop Rate (10) command to the RMC to move the axis in open loop control. See the **Issuing Commands** section below for details on issuing commands.

Closed Loop control is more complex. The RMC is all about closed loop motion control. Read on to learn how it works.

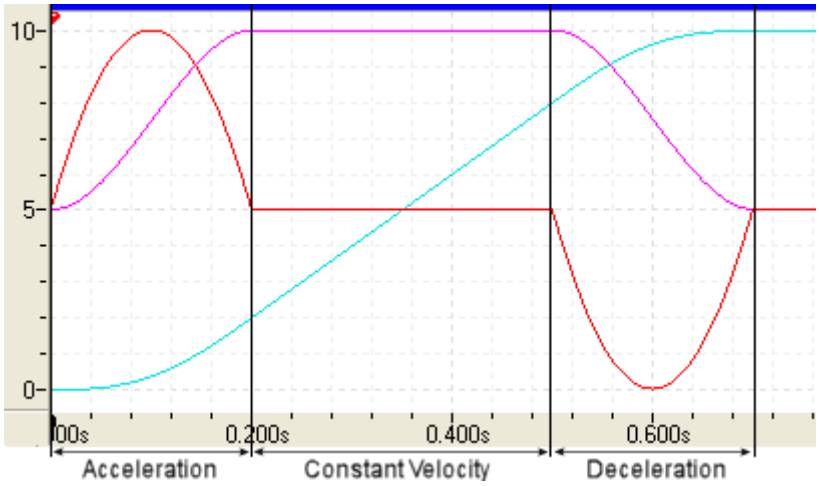
Target Profile

When the RMC is commanded to make a move in closed loop control, it generates a *target profile*, which specifies where the position (or velocity, pressure, or force) should

be at any given point in time. This profile includes the Target Position, Target Velocity, and Target Acceleration.

For example, if a Move Absolute command is issued to the RMC, it will generate a profile according to the parameters of the command. The plot below shows a typical Move Absolute target profile. The position is moving from zero to ten. The RMC uses rounded accelerations to provide smooth motion.

Legend: — Target Position — Target Velocity — Target Acceleration



PID Algorithm

Generating a target profile determines where the axis should be, but does not make the axis move. In order to move the axis, the RMC uses a PID algorithm that calculates how much Control Output is required to make the Actual Position follow the Target Position. The RMC obtains the Actual Position from the transducer.

The PID algorithm uses several values, called **gains**, that specify how the PID calculates the Control Output. For each axis, in every motion application, these gains must be adjusted to obtain optimal control of the system. This process is called "tuning". RMCTools offers a Tuning Wizard that significantly speeds up the process of tuning. Tuning can also be done manually.

An axis *must* be tuned before the RMC can be controlled in closed loop control.

Making Moves

To make an axis move, issue a motion command. The Issuing Commands section below describes how to issue commands. The RMC has many motion commands for various types of motion. See the

Issuing Commands

To make the RMC do something, such as move, turn on a discrete output, or run a user program, you must issue a command. For example, to move the axis to a position, you would issue a Move Absolute command.

To issue a command from RMCTools, use the Command Tool:

1. On the **View** menu, choose **Command Tool**.
2. In the **Cmd** box of the desired axis, click the ellipsis (…) button.
3. Browse to the command you need and click **OK**.
4. If the command has parameters, fill them out.
5. Click **Send** to issue the command to the RMC.
6. If an error occurred, or if you wish to see if the command went through, open the Event Log by double-clicking **Event Log** in the Project pane. It shows everything that happened in the RMC.

Commands can also be issued to the RMC from a host controller, such as a PLC or HMI. You can also create user programs in the RMC that issue commands. See the Issuing Commands topic for more details.

Caution: For each command you wish to issue, write it only once so that it is issued only once. If you issue the same motion command (to the same position) multiple times, it can cause the target position to overshoot the requested position.

Setting Up the RMC

Before you can move an axis in closed loop control, you must set up the items listed below. For instructions on setting all these items, see the Startup Guide that came with the RMC.

- **Axes Definitions**
  You must define the axes. For example, will they be position or pressure or some other control, will there be reference inputs. Which inputs are connected to which output and so on.
- **Feedback Type**
  Some RMC modules require that you define the feed back type, such as voltage, current, Start/Stop, etc.
- **Scale and Offset**
  The Scale and Offset parameters convert the transducer feedback to real-life units, such as inches, millimeters, etc.
- **Tuning**
  The tuning procedure sets the PID gains for closed-loop control.

Communication

The RMC can communicate as a slave with other devices via Ethernet, Serial RS-232/485, or PROFIBUS depending on the RMC. For details on the RMC's communication protocols, see the Communications Overview topic.

The RMC does not perform motion control on feedback via any communication channels. It only controls motion on axes that are directly wired to the RMC's modules.

1.3. Disclaimer

Although great effort has been taken to ensure the accuracy of the information in this documentation, it is intended to be used only as a guide. Knowledge of motion control, hydraulic servos, electric servos, transducers, and safety rules is required. Delta Computer Systems, Inc. does not accept responsibility for problems resulting from errors or omissions in this documentation. The information in this documentation is subject to change without notice.

Neither Delta Computer Systems, Inc. nor anyone else involved in the creation, production, or delivery of this product shall be liable for any direct, indirect, consequential injuries and or damages arising out of the use, the results of use, or the inability to use this product.

All brand names and trademarks referenced in this manual are the property of their respective owners.

RMC Return for Repair

Should an RMC require repair, refer to the Technical Support topic for return details.

Ownership

Title to all intellectual property rights associated with Delta product (including, but not limited to firmware and software) remains with Delta. Any modifications to products, or special versions of products, will become the sole property of Delta, even if such

modifications or special versions are made by request of Customer and paid for, all or in part, by Customer, unless Delta specifically agrees otherwise in writing.

2. Starting Up the RMC

2.1. RMC Startup Procedure

The best resource to start using the RMC is the Startup Guide that was shipped with the controller. The Startup Guide can also be downloaded from Delta's website at www.deltamotion.com. If you do not have a Startup Guide, the following step-by-step procedure will help you get your system up and running.

TIP: Deltas position/pressure simulator provides a simple way to test your program before connecting the controller to a real system.

1. Provide Power to the RMC

Apply power to the RMC as described in the Wiring topics.

2. Connect to Controller in RMCTools

a. **RMC75E, RMC150E, or RMC200:**
Connect a USB cable from the PC to the USB Monitor port on the RMC.

RMC75S or RMC75P:
Connect a null-modem cable from the PC to the RS-232 Monitor port on the RMC. See the Monitor Port topic for cable details.

b. Open RMCTools.

c. On the **File** menu, click **New Project**. In the **New Project Wizard**, fill in the fields and click **Finish**.

d. In the **New Controller Wizard**, type a **Controller Name** and choose **Automatically Detect the Controller Information**.

e. Click **Next**.

f. **RMC75E or RMC150E:**
Choose USB and click **OK**. Choose the desired RMC and click **Next.**

RMC75S or RMC75P:
Choose the COM port that the RMC is connected to.

g. Click **Next**.

h. You may need to wait while RMCTools connects to the controller. Once it has connected, verify that the information is correct and click **Finish**.

3. Configure Module Properties

The following hardware may require configuration before use as part of an axis:

- **RMC150 UI/O Module High-Speed Channels**
  The high-speed channels of the Universal Input/Output module need to be configured for use as a Quadrature axis input or SSI axis input before they can be used as an axis input. For details, see the Configuring UI/O High-Speed Channels topic.

- **RMC200 S8**
  The last two inputs, 6 and 7, can be configured as a single quadrature input. For details, see the S8 Module topic.
- **RMC200 D24**
  The high-speed inputs may be configured as one or two quadrature inputs. For details, see the D24 Module topic.
- **RMC200 U14**
  The high-speed channels of the Universal Input/Output module need to be configured for use as a SSI/MDT or Quadrature axis inputs before they can be used as an axis input. For details, see the Configuring U14 High-Speed Channels topic.

4. Define the Axes

RMCTools provides full flexibility for creating axes and assigning the RMC axes to the hardware. When you create or change axes, you can choose which inputs(s) and outputs are assigned to the axis.

After the RMC powers up, it has default axis definitions. To change these definitions, do the following:

a. In the **Project** tree, expand the **Axes** folder and double-click **Axis Definitions**.
b. Use the dialog to view the axis definitions and change them if you would like. For more details, click the Help button in the dialog.

5. Connect a Feedback Device

Note: This step can be done before or after step 6.

a. **Important:** Turn off power to the RMC and the feedback device before connecting any wires!
b. For each axis you wish to connect a feedback device to, wire it to the RMC according to the instructions in the Wiring topic.
c. After wiring, re-apply power to the RMC and the feedback device.
d. In RMCTools, in the **Project** pane, click the **RMC** controller.
e. On the RMCTools toolbar, click the **Controller** button and choose **Go Online**. Verify that the red circle around the RMC icon in the **Project** pane disappears, indicating that it is online with the RMC.
f. The RMC must be properly configured before it will communicate with the transducer(s). Refer to the procedure for your transducer type:

- SSI transducer
  i. On the RMCTools toolbar, click the **Axis Tools** button.
  ii. In the **Axis Parameters** pane, on the **Setup** tab, under the **Primary Control Setup**, in the **Feedback Type** register, select **SSI**.
  iii. From the information in your SSI transducer data sheet, enter the correct value for each of these registers:
    - **SSI Format** - Binary or Gray
    - **SSI Data bits** - (e.g. 24)
  iv. To apply the changes to the RMC, click the **Download** button or press Ctrl+D.

- Magnetostrictive transducer with Start/Stop or PWM
  i. On the RMCTools toolbar, click the **Axis Tools** button.
  ii. In the **Axis Parameters** pane, on the **Setup** tab, under the **Primary Control Setup**, in the **Feedback Type** register, select **MDT**.

iii. In the **MDT Type** register, select the type of MDT transducer you have. This information should be available on the MDT datasheet. The options are:
- Start/Stop Rising Edge
- Start/Stop Falling Edge
- Pulse-Width Modulated

iv. To apply the changes to the RMC, click the **Download** button or press Ctrl+D.

- Voltage or Current transducer
  i. On the RMCTools toolbar, click the **Axis Tools** button.
  ii. In the **Axis Parameters** pane, on the **Setup** tab, under the **Primary Control Setup**, in the **Input Type** register, select **Voltage or Current**.
  iii. To apply the changes to the RMC, click the **Download** button or press Ctrl+D.

- Quadrature encoder
  There are no parameters required to set up in order to interface to a quadrature encoder. If your motion is rotary, see the rotary topic for details on setting the Rotary vs. Linear parameter.

- Resolver
  i. On the RMCTools toolbar, click the **Axis Tools** button.
  ii. In the **Axis Parameters** pane, on the **Setup** tab, under the **Primary Control Setup**, in the **Resolver Resolution** register, choose a resolution. For details, see the Resolver Resolution topic.
  iii. In the **Reference Frequency** register, choose the frequency required for your resolver. This information should be available from the resolver data sheet.
  iv. In the **Reference Amplitude** register, click the ellipsis button to open the dialog. Click the **Help** button for details on how to set the Reference Amplitude.
  v. To apply the changes to the RMC, click the **Download** button or press Ctrl+D.

g. On the RMCTools toolbar, click the **Axis Tools** button.
h. In the **Axis Tools**, in the **Axis Status Registers** pane, on the **Basic Position** tab, look at the **Actual Position** register. It may be changing slightly.
i. Move the axis and look for a corresponding change in the **Actual Position** register. If it does not change, recheck the wiring, verify that the **Primary Control Setup** registers in the **Axis Parameters** pane are correct, and check for changing **Actual Position** again. When it is working correctly, proceed to the next step.

Note:
It is important that the transducer is connected and is working properly before continuing the Start-up procedure.

j. On the **Controller** menu, click **Update Flash**. This stores your changes in the RMC even in the event of a power outage.

k. Press Ctrl+S to save the project.

6. Connect an Actuator

> **Note:**
> Read this section completely *before* executing any commands on the RMC.

a. **Important:** Turn off power to the RMC and the actuator before connecting any wires!

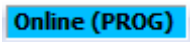
b. For each axis you wish to connect an actuator to, wire it to the RMC according to the Wiring topic.

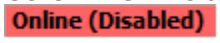
c. After wiring, re-apply power to the RMC, the feedback device, and the actuator.

> **Note:**
> To test the actuator, you will supply a Control Output voltage from the RMC to the actuator. Before doing this, make sure that the axis may safely move in either direction!

d. In RMCTools, in the **Project** pane, click the **RMC** controller.

e. On the RMCTools toolbar, click the **Controller** button 

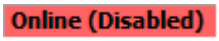
and choose **Go Online**. The toolbar will indicate that RMCTools is online 

or 

.

f. On the RMCTools toolbar, click the **Axis Tools** button .

g. In the **Axis Parameters** pane, on the **All** tab, expand the **Simulate** section and verify that **Simulate Mode** is unchecked. If you make changes, you must click the Download button  or press Ctrl+D to apply the changes to the RMC.

h. RMC200 Only: On the toolbar, click 

and choose Program Mode.

i. Enable the axis: In the Command Tool, in the **Cmd** box, type "Enable", then choose **Enable Controller (7)** from the list. Click **Send**.

j. In the **Axis Tools**, in the **Axis Status Registers** pane, on the **Basic** tab, verify that the **Control Output** for the axis is 0 (zero).

k. Turn on power to the motor or hydraulics for the axis being set up. Note that after starting, it is normal for the axis to drift slowly.

l. In the **Command Tool**, do the following steps to apply 0.1 V to the axis Control Output:

l. **Caution:** Use the Direct Output command with caution! It disables the safety features of the RMC!

- Double-click the **Cmd** box, type "D" and choose the **Direct Output (9)** command.
- Set the **Output** to 0.1 V.
- Set the **Ramp Rate** to 100 V/s.
- Click **Send Command**.
- Enter 0 in the **Output** box. This will allow you to quickly issue a Direct Output Command with 0 drive in the next steps.

m. Verify that the **Control Output** register for the axis is at 0.1 V.

n. Observe the **Actual Position** register (on the **Basic Position** tab) and note whether it is increasing or decreasing. One of three things could have happened at this point:

- **The actuator moved and the Actual Position increased.**
  In this case you are ready to go to the next section and set the Scale.
- **The actuator moved and the Actual Position decreased.**
  In this case, first verify that the wiring is correct. If it is not, fix it and repeat the process. If it is, invert the Output Polarity to reverse the direction:

m.

1. In the **Axis Parameters** pane, select the **Setup** tab and expand the **Primary Control Setup** section.
2. Double-click the **Invert Output Polarity** register in the axis you are using. The **Invert Output Polarity** box should now have a checkmark in it.
3. Click the **Download** button or press Ctrl+D to apply the changes to the controller.
4. Repeat the process again: Issue the Direct Output command again, observe the Actual Position, and see which of the three things happened.

m.

- **The actuator does not move.**
  In this case, you may need to increase the Control Output. First, verify the following:
-

1. Check that the physical Control Output voltage really is 0.1 volts.
2. Check that the actuator is enabled.

If these items check out fine, repeat the process above with a larger **Output**. Some systems may require much more than 0.1 volt to move.

n. **Note:**
It is essential that the Actual Position increase when a positive output voltage is specified with Direct Output command. If this condition is not met, you will not be able to control the axis in closed loop.

o.

p. In the Command Tool, enter 0 in the **Output** box and click **Send Command**, or press Alt + S, to turn the **Control Output** voltage off.

q. In the Command tool, enter -0.1 in the **Output** box and click **Send Command**, or press Alt + S. The **Control Output** register should change to -0.1 and the axis should move in the opposite direction.

Once the actuator moves fine, you are ready to go to the next step.

7. Set the Scale and Offset

The **Scale** and **Offset** parameters convert the **Counts** from the transducer into meaningful measurement units. In order to do so, the **Scale** and **Offset** must first be configured correctly.

Set the Scale and Offset:

a. In the **Axis Tools**, in the **Axes Parameters** pane, select the **Setup** tab. Note that the first two registers under the **Primary Control Setup** section are **Position Scale** and **Position Offset**. These are the parameters you will configure.

b. To calculate what the scale and offset should be, you will need to refer to the Scaling topic.

c. When you have determined what value the scale or offset parameter should be, type the new value in the cell in the **Axis Parameters** pane. Then click the **Download** button or press Ctrl+D to apply the changes to the controller.

d. Move your axis and verify that the positions are correct.

e. On the **Controller** menu, click **Update Flash**. This stores your changes in the RMC even in the event of a power outage.

f. Press Ctrl+S to save the project.

Set the Display Units:

a. In the **Axis Tools**, in the **Axes Parameters** pane, select the **Setup** tab.
b. In the **Primary Control Setup** section, locate the **Display Units** parameter. From the drop-down list, select a unit.
If you wish to use units not listed, choose **Custom**, then type up to 4 characters in the **Custom Units** parameter.
c. For dual-loop axes, repeat this in the **Secondary Control Setup** section.
d. Click the **Download** button  to apply the changes.

8. Tune Each Axis

In order to control an axis in closed-loop control, it must first be tuned. Refer to the online help for the tuning procedure:

a. In RMCTools, on the **Tools** menu, click **Tuning Tools**. This pane provides everything you need for tuning the axes.
b. Follow the instruction in the Tuning topic to tune the system.

9. Set up and Configure the Communications

If your RMC will be communicating with an external controller, such as a PLC, PC, or other device, you must set up the communication. Refer to one or more of the following topics for details on configuring the communication type of your RMC controller:

- Ethernet Overview
- PROFIBUS-DP
- Serial Overview

10. Save Your Configuration Settings

Make sure to save your settings! You must do the following:

a. **Save your RMCTools project**
To save the project, on the **File** menu, click **Save**. Before doing this, make sure the project data is the same as the controller data, by either downloading or uploading.

b. **Save the RMC data to non-volatile memory**
On the **Controller** menu, click **Update Flash**.

Tip: To do both these steps at once, on the **File** menu, choose **Save and Update Flash**.

2.2. Scaling

2.2.1. Scaling Overview

Scaling refers to converting the transducer feedback into meaningful units. The RMC uses the Scale and Offset parameters to convert the transducer Counts or Raw Counts into measurement units (position, velocity, pressure, force). For example, the Voltage returned by an analog position transducer must be converted to positions in order to be useful for control. In order to correctly convert the transducer feedback to useful units, you must calculate the Scale and Offset parameters.

Units

The RMC uses the default terms **Position Units (pu)**, **Pressure Units (Pr)**, and **Force Units (Fr)** for the scaled feedback. You can change these to feedback units of your preference by choosing from a list or entering a custom unit. See Display Units for details.

Calculating the Scale and Offset

RMCTools provides Scale/Offset wizards to help you calculate the Scale and Offset parameters. You can also calculate the parameters manually.

Scale/Offset Wizards

To access the Scale/Offset wizard, in **Axis Tools**, in the **Axis Parameters** pane, click the **Setup** tab. Expand the **Tools and Wizards** section. Click **Launch** to open the Scale/Offset Wizard.

Note:
Before using the Scale/Offset Wizard, you must define the axes and set the transducer type parameters.

Manual Calculation

The methods of calculating the scale and offset parameters depends on the axis type. See the scaling topic for your axis type:

Analog Position Scaling

Analog Velocity Scaling

Analog Acceleration Scaling

Analog Pressure/Force Scaling

Load Cell Scaling

MDT Scaling

SSI Scaling

Quadrature Scaling

Resolver Scaling

Rotary Scaling

Scale and Offset Parameters

The following parameters converting the transducer feedback Counts into meaningful units. Each axis, whether a control axis or reference axis, has these parameters:

- **Scale**
  On a linear axis, the Position, Pressure, Force, or Velocity Scale defines the number of Actual units (position, velocity, pressure or force) per count or volt returned from the transducer.
- **Offset**
  On a linear axis, the Position, Pressure, Force, or Velocity Offset moves the zero point of the Actual units to where the user wants it.
  On a rotary axis, the Position Offset defines the lower range of the position range.
- **Count Offset**
  This parameter is used for Absolute Rotary axes to determine where the zero point of the counts should be.
- **Position Unwind**
  On a rotary axis, the Position Unwind defines the span of the position range.
- **Count Unwind**
  On a rotary axis, the Count Unwind defines the range of counts over which the Position Unwind spans.
- **Current or Voltage Offset**
  Provides an offset value that is applied before the Scale and Offset parameters.
- **Millivolts/Volt Offset**

Provides an offset value that is applied before the Scale and Offset parameters.

- **Negative Correction Factor**

  This parameter is used for bidirectional load cells, which typically have a slight gain difference between the positive and negative side.

2.2.2. Analog Position Scaling

To have any useful meaning, the Voltage or Current from an analog transducer must be scaled to measurement units. This topic describes how to manually calculate the Scaling and Offset parameters for an analog position input.

Delta Recommends using the Scale/Offset Wizard for scaling the position. If you need to do it manually, read this topic.

Manually Scaling Voltage or Current to Position Units

The RMC calculates the Actual Position every control-loop time using either of the following formulas:

Actual Position = (Voltage x Position Scale) + Position Offset

Actual Position = (Current x Position Scale) + Position Offset

Calculating the Position Scale and Offset

The Scale/Offset Wizards provide the easiest method of scaling your axis. If you want to compute your scale and offset manually, do the following:

The accuracy of this method depends on how accurately you can measure two positions of the axis.

1. Physically measure the axis' position at two points and record the value of the Volts register at each point. Call the *smaller* measured position P_0, and its corresponding voltage V_0 or current C_0. Call the *greater* measured position P_1, and its corresponding voltage V_1 or current C_1.
2. Calculate the Position Scale with the following equation:

   $$\text{Position Scale} = (P_0-P_1)/(V_0-V_1)$$
   $$\text{Position Scale} = (P_0-P_1)/(C_0-C_1)$$
3. Calculate the Position Offset with the following equation:

   $$\text{Position Offset} = P_0 - \text{Position Scale} \times V_0$$
   $$\text{Position Offset} = P_0 - \text{Position Scale} \times C_0$$

2.2.3. Analog Velocity Scaling

To have any useful meaning, the Voltage or Current from an analog transducer must be scaled to measurement units. This topic describes how to manually calculate the Scaling and Offset parameters for an analog velocity input.

Delta Recommends using the Scale/Offset Wizard for scaling the velocity. If you need to do it manually, read this topic.

Scaling Voltage or Current to Velocity Units for Tachometers

The RMC calculates the Actual Velocity from the tachometer voltage every control-loop time using either of the following formulas:

Actual Velocity = (Voltage + Velocity Offset) x Velocity Scale

Actual Velocity = (Current + Velocity Offset) x Velocity Scale

Calculating the Velocity Scale and Offset

1. From the tachometer specification sheet, determine the voltage or current at which the velocity is zero. Call it V_{ZERO} or C_{ZERO}. Or, for greater accuracy, make sure the tachometer is stopped, then look at the voltage or current feedback value in the Status Registers pane. Call this value Call it V_{ZERO} or C_{ZERO}.
2. Negate V_{ZERO} or C_{ZERO} and enter it as the Velocity Offset:

   Velocity Offset = $-V_{ZERO}$

   Velocity Offset = $-C_{ZERO}$
3. From the tachometer specification sheet, determine what the maximum velocity is. Call it Max Vel.
4. From the tachometer specification sheet, determine what the voltage or current output should be at the Max Vel. Call it V_{MAX} or C_{MAX}. Calculate the Velocity Scale with one of the following equations:

   Velocity Scale = Max Vel / (V_{MAX} - V_{ZERO})

   Velocity Scale = Max Vel / (C_{MAX} - C_{ZERO})

1.

2.2.4. Analog Acceleration Scaling

To have any useful meaning, the Voltage or Current from an analog transducer on an acceleration input must be scaled to measurement units. This topic describes how to manually calculate the Scaling and Offset parameters for an analog acceleration transducer.

Notice that Acceleration inputs are only used in advanced applications.

Scaling Acceleration Units

The RMC calculates the Actual Acceleration every control-loop time using the following formula:

Actual Acceleration = ((Voltage or Current) + Acceleration Offset) x Acceleration Scale

Calculating the Acceleration Scale and Offset

1. From the accelerometer specification sheet, determine the voltage or current at which the acceleration is zero. Call it V_{ZERO} or C_{ZERO}. Negate that value, and enter it as the Acceleration Offset.

   Acceleration Offset = $-V_{ZERO}$

   Acceleration Offset = $-C_{ZERO}$
2. From the accelerometer specification sheet, determine what the maximum acceleration is. Call it Max Accel.
3. From the accelerometer specification sheet, determine what the voltage or current output should be at the Max Accel. Call it V_{MAX} or C_{MAX}. Calculate the Acceleration Scale with one of the following equations:

   Acceleration Scale = Max Accel / (V_{MAX} - V_{ZERO})

   Acceleration Scale = Max Accel / (C_{MAX} - C_{ZERO})

1.

2.2.5. Load Cell Scaling

To have any useful meaning, the mV/V signal from a load cell input must be scaled to force units. The Force Scale, Force Offset, Millivolts/Volt Offset, and Negative Correction Factor

parameters are used to define force units as a function of the load cell mV/V signal. This topic describes how to manually calculate these parameters for a load cell input.

Delta Recommends using the Scale/Offset Wizard for scaling the load cell force. If you need to do it manually, read this topic.

Scaling Load Cell Force

The RMC calculates the Actual Force every control loop using the following formula:

If (Millivolts/Volt + Millivolts/Volt Offset) < 0 Then

Force = (**Millivolts/Volt** + **Millivolts/Volt Offset**) x Negative Correction Factor x Force Scale + Force Offset

Else

Force = (**Millivolts/Volt** + **Millivolts/Volt Offset**) x Force Scale + Force Offset

EndIf

To correctly scale the axis for your particular application, use one of the methods below to find the scaling parameters.

Method 1: Use Transducer Specifications

This method uses the calibration data for the load cell.

Unidirectional Load Cells (Tension-Only or Compression-Only)

1. From the transducer data sheet:
    a. Find the mV/V output of your load cell at the rated load. Call these Max mV/V and Max Force.
    b. Find the mV/V value at zero force. This may be called Zero Balance.
2. Calculate the Force Scale with the following formula:

    Force Scale = Max Force / (Max mV/V - Zero Balance)
3. Enter the negative of the Zero Balance value in the Millivolts/Volts Offset parameter.

Bidirectional Load Cells (Tension and Compression)

1. From the transducer data sheet:
    a. Find the mV/V output of your load cell at the rated positive load. Call these Max Positive mV/V and Max Positive Force.
    b. Find the mV/V output of your load cell at the negative rated load. Call these Max Negative mV/V and Max Negative Force.
    c. Find the mV/V value at zero force. This may be called Zero Balance.
2. Calculate the Force Scale with the following formula:

    Force Scale = Max Positive Force / (Max Positive mV/V - Zero Balance)
3. Calculate the Negative Correction Factor with the following formula:

    Negative Correction Factor = (Max Negative Force / (Max Negative mV/V - Zero Balance)) / (Max Positive Force / (Max Positive mV/V - Zero Balance))
4. Enter the negative of the Zero Balance value in the Millivolts/Volts Offset parameter.

Method 2: Force / Millivolts Per Volt

The accuracy of this method depends on how accurately you can measure the force of the axis with some external device such as a reference load cell or some known mass.

Unidirectional Load Cells (Tension-Only or Compression-Only)

1. For two different force values (Force0 and Force1), apply a known force, and record the mV/V values (mVV0 and mVV1) at those forces.

2. Calculate the Force Scale and Force Offset with the following formula:

   Force Scale = (Force0 - Force1) / (mVV0 - mVV1)

   Force Offset = Force0 - Force Scale x mVV0

Bidirectional Load Cells (Tension and Compression)

1. For three different force values, apply a known force. The force values should be a large positive value (ForcePos), a large negative value (ForceNeg), and one close to zero (ForceZero). For each force value, record the mV/V values at those forces (mVVPos, mVVNeg, and mVVZero).
2. Calculate the Force Scale, Force Offset, and Negative Correction Factor with the following formulas:

   Force Scale = (ForcePos - ForceZero) / (mVVPos - mVVZero)

   Force Offset = ForcePos - Force Scale x mVVPos

   Negative Correction Factor = (ForceNeg / mVVNeg) / (ForcePos / mVVPos)

2.2.6. Analog Pressure or Force Scaling

To have any useful meaning, the Voltage or Current from the analog pressure or force transducer must be scaled to pressure or force units. The Pressure/Force Scale, Pressure/Force Offset, and Invert Feedback Polarity parameters are used to define pressure or force units as a function of transducer voltage. This topic describes how to manually calculate these parameters for an analog pressure transducer.

Delta Recommends using the Scale/Offset Wizard for scaling the pressure or force. If you need to do it manually, read this topic.

Scaling Single-Input Pressure or Force

The RMC calculates the Actual Pressure/Force every control-loop using the following formula (may be either pressure or force):

Actual Pressure = (Voltage (or Current) x Pressure Scale) + Pressure Offset

To correctly scale the axis for your particular application, use one of the methods below to find the Pressure/Force Scale and Pressure/Force Offset parameters.

Method 1: Use Transducer Specifications

- From the transducer data sheet, find the maximum output voltage of your pressure transducer and the pressure at that voltage.
- Calculate the Pressure Scale with the following formula:

  Pressure Scale = Max Pressure / Max Voltage

  - Make sure the system has 0 pressure and then record the value of the Voltage register. Enter the negative of that value as the Pressure Offset.

Method 2: P0/P1 Calculation

The accuracy of this method depends on how accurately you can measure the pressure of the axis.

1. Physically measure two different pressures of the axis and record the value of the Voltage register at each point. (The difference between the two pressure should be as large as possible). Call the *smaller* measured pressure P_0, and its corresponding Voltage V_0. Call the *greater* measured pressure P_1, and its corresponding Voltage V_1.
2. Calculate the Pressure Scale with the following equation:

   Pressure Scale = $(P_0-P_1)/(V_0-V_1)$

If the Pressure Scale is negative, ignore the minus sign. RMCTools accepts only a positive Scale.

3. Calculate the Pressure Offset with the following equation:

   Pressure Offset = P_0 - Pressure Scale x V_0

Scaling Dual-Input (Differential) Force - RMC75 and RMC150

Dual-Input Force requires two pressure transducers. Each pressure transducer must be individually scaled. The RMC75 and RMC150 calculate the Actual Differential Force every control loop using the following formulas:

Actual Force A = (Channel A Voltage (or Current) x Force A Scale) + Force A Offset

Actual Force B = (Channel B Voltage (or Current) x Force B Scale) + Force B Offset

Actual Differential Force = Actual Force A - Actual Force B

To correctly scale the axis for your particular application, use the method below to find the A and B Force Scale and Force Offset parameters.

Method: Use Transducer Specifications

This method assumes the two pressure transducers provide zero volts at zero pressure or force.

- From the transducer data sheet, find the maximum output voltage of your pressure transducer and the pressure at that voltage.
- Calculate the area of the A side (cap end) of the piston.
- Multiply the area of the A side with the maximum pressure of the A transducer to get the maximum A force.
- Calculate the Force A Scale with the following formula:

  Force A Scale = maximum A force / Max A Voltage
- Calculate the effective area of the B side (rod end) of the piston. This is the area of the cylinder minus the area of the rod.
- Multiply the area of the B side with the maximum pressure of the B transducer to get the maximum B force.
  - Calculate the Force B Scale with the following formula:

    Force B Scale = maximum B force / Max B Voltage
  - Make sure the system has 0 force and then record the value of the Channel A Actual Force (to find this register, in Axis Status Registers pane, click the All tab, and expand the Pressure/Force Feedback section). Negate the value and enter it as the Force A Offset.

Scaling Dual-Input (Differential) Force - RMC200

Dual-Input Force requires two pressure transducers. Each pressure transducer must be individually scaled. As shown by the formulas below, the RMC200 calculates the pressures for each channel, then the forces for each channel, then takes the difference and applies the Dual Channel Force Offset to arrive at the Actual Force.

Channel A Pressure = Channel A Voltage (or Current) * Channel A Pressure Scale + Channel A Pressure Offset

Channel B Pressure = Channel B Voltage (or Current) * Channel B Pressure Scale + Channel B Pressure Offset

Channel A Force = Channel A Pressure * Channel A Force Scale

Channel B Force = Channel B Pressure * Channel B Force Scale

Actual Force = Channel A Force - Channel B Force + Dual Channel Force Offset

To correctly scale the axis for your particular application, use the method below to find the A and B Force Scale and Force Offset parameters.

Method: Use Transducer Specifications

This method assumes the two pressure transducersprovide zero volts at zero pressure or force.

- From the transducer data sheet, find the maximum output voltage of your pressure transducer and the pressure at that voltage.
- Calculate the Channel A Pressure Scale with the following formula:

  Channel A Pressure Scale = Maximum A Pressure / Max A Voltage or Current
- Calculate the area of the A side (cap end) of the piston.
- Calculate the Force A Scale with the following formula:

  Channel Force A Scale = area of A side of the piston
- Repeat for Channel B
  - Make sure the system has 0 force and then record the value of the Actual Force. Negate the value and enter it as the Dual Channel Force Offset.

2.2.7. MDT Scaling

To have any useful meaning, the counts from the transducer must be scaled to position units. The Position Scale and Position Offset parameters define the position units as a function of transducer counts. This topic describes how to correctly calculate these parameters for an MDT transducer.

Delta Recommends using the Scale/Offset Wizard for scaling the position. If you need to do it manually, read this topic.

Scaling Counts to Position Units

The RMC calculates the Actual Position every control-loop time using the following formula:

Actual Position [pu] = (Counts [cnt] x Position Scale [pu/cnt]) + Position Offset [pu]

Note:
If the Actual Position filter is applied, the RMC filters the Actual Position after calculating it with the above formula.

Manually Calculating the Scale and Offset

The Scale/Offset Wizards provide the easiest method of scaling your axis. If you prefer to do it manually, read this section.

Determining the Scale and Offset

There are two methods of finding the correct Position Scale and Position Offset.

Method 1: P_0/P_1 Calculation

The accuracy of this method depends on how accurately you can measure two positions of the axis.

1. Physically measure the axis position at two points and record the value of the counts register at each point. Call the *smaller* measured position P_0, and its corresponding Counts C_0. Call the *greater* measured position P_1, and its corresponding Counts C_1.
2. Calculate the Position Scale with the following equation:

$$\text{Position Scale} = (P_0\text{-}P_1)/(C_0\text{-}C_1)$$

3. Calculate the Position Offset with the following equation:

$$\text{Position Offset} = P_0 - \text{Position Scale} \times C_0$$

Method 2: Using the MDT Calibration Number

1. Obtain the following information (usually from the transducer label or data sheet):
   - Calibration Number (calibration constant) in µs/in or µs/µm (typically about 9 µs/in)
   - Number of Recirculations (if it has no recircs, use a value of 1)
2. Calculate the Position Scale with the following formula:

RMC75 and RMC200:

$$\text{Position Scale (pu/counts)} = \frac{1}{\text{Calibration Number (µs/pu)} \times 240\text{Mhz} \times \text{\# of Recircs}}$$

RMC150:

$$\text{Position Scale (pu/counts)} = \frac{1}{\text{Calibration Number (µs/pu)} \times 120\text{Mhz} \times \text{\# of Recircs}}$$

where

- "pu" is the position units, e.g. in, mm, m, etc. If you want position units other than what is given by the Calibration Number (e.g. in,mm, etc.), you must convert the calibration number to the units you want.
- The 240MHz and 120 MHz values comes from the RMC's internal counter. If you change the units from the above equation, make sure the units for the equation work out properly. The units for MHz is [10^6/sec].

3. If you wish to reverse the direction of the feedback, make the Scale negative.
4. Download the Scale to the RMC.
5. Calculate the Position Offset:
   - Make sure the Position Offset is zero (0) and downloaded to the controller before continuing.
   - In the Axis Tools window, look at the Actual Position register.
   - Move the axis to a point where you know what you want the position to be (such as 0). Called this the Desired Position. Look at the **Actual Position** register. Call this position P_0.
   - Calculate the Position Offset using the following equation: Position Offset = Desired Position - P_0.
6. Download the Offset to the RMC.

Example

1. Connected to an RMC75 is an MDT transducer with a calibration number of 9.1615 µs/in and no recirculations (this means it has just 1 recirculation). The system needs to be scaled to inches.
   - To calculate the Scale:

$$\text{Position Scale} = \frac{1}{9.1615[\mu\text{sec/in}] \times 240[\text{MHz}] \times 1} = 0.0004548018[\text{in/count}]$$

   - Download the Scale to the RMC.
   - To calculate the Offset: Move the axis to where it should be 0. The Actual Position shows 2.4528. The Position Offset should be: 0 - 2.4528 = -2.4528.
   - Download the Offset to the RMC.

2. Connected to an RMC75 is an MDT transducer with a calibration number of 9.012 µs/in and 2 recirculations. The system needs to be scaled to inches.
    - To calculate the Scale:

$$\text{Position Scale} = \frac{1}{9.012[\mu\text{sec/in}] \times 240[\text{MHz}] \times 2} = 0.00023117[\text{in/count}]$$

    - Download the Scale to the RMC.
    - To calculate the Offset: Move the axis to where it should be 10.125. The Actual Position shows 11.241. The Position Offset should be: 10.125 - 11.241 = -1.116.
    - Download the Offset to the RMC.

3. Connected to an RMC150 is an MDT transducer with a calibration number of 9.1313 µs/in and no recirculations (this means it has just 1 recirculation). The system needs to be scaled to millimeters.
    - We must first calculate the scale, but in order to do that we need to know how many position units there are in an inch: 25.4 mm/in.

    - To calculate the Scale:

$$\text{Position Scale} = \frac{1}{9.1313[\mu\text{sec/in}] \times \frac{1[\text{in}]}{25.4[\text{mm}]} \times 120[\text{MHz}] \times 1} = 0.02318[\text{mm/count}]$$

    - Download the Scale to the RMC.
    - To calculate the Offset: Move the axis to where it should be 0 mm. The Actual Position shows 24.8. The Position Offset should be: 0 - 24.8 = -24.8.
    - Download the Offset to the RMC.

2.2.8. SSI Scaling

To have any useful meaning, the counts from the transducer must be scaled to position units. The Position Scale and Position Offset parameters define the position units as a function of transducer counts. RMCTools provides Scale/Offset wizards to guide you through the scaling process. This topic describes how to manually calculate these parameters for an SSI transducer.

To scale rotary SSI axes, see the Rotary Scaling topic.

Delta Recommends using the Scale/Offset Wizard for scaling the position. If you need to do it manually, read this topic.

Scaling Counts to Position Units

The RMC calculates the Actual Position every control-loop time using the following formula:

Actual Position [pu] = (Counts [cnt] x Position Scale [pu/cnt]) + Position Offset [pu]

Note:
If the Actual Position filter is applied, the RMC filters the Actual Position after calculating it with the above formula.

To correctly scale the axis for your particular applications, use the methods below to find the Position Scale and Position Offset parameters.

Determining the Scale and Offset

There are two methods of finding the correct Position Scale and Position Offset.

Method 1: P_0/P_1 Calculation

The accuracy of this method depends on how accurately you can measure two positions of the axis.

1. Physically measure the axis position at two points and record the value of the counts register at each point. Call the *smaller* measured position P_0, and its corresponding Counts C_0. Call the *greater* measured position P_1, and its corresponding Counts C_1.
2. Calculate the Position Scale with the following equation:

   $$\text{Position Scale} = (P_0-P_1)/(C_0-C_1)$$

3. Calculate the Position Offset with the following equation:

   $$\text{Position Offset} = P_0 - \text{Position Scale} \times C_0$$

Method 2: Using the Transducer Resolution

Linear Transducers:

1. Obtain the transducer resolution (usually from the transducer data sheet), for example 0.005 mm. The resolution of the transducer indicates the smallest increment of the transducer. This increment is one count in the RMC.
2. Determine the position units you would like for your Actual Position, for example inches, meter, mm, etc.
3. The Position Scale is equal to the number of position units per count, for example, using inches and a resolution of 0.005 mm:

   $$\text{Position Scale} = \frac{1[\text{in}]}{25.4[\text{mm}]} \times 0.005\ [\text{mm/count}] = 0.0001968 5[\text{in/count}]$$

4. If you wish to reverse the direction of the feedback, make the Scale negative.
5. Download the Scale to the RMC.
6. Calculate the Position Offset:
   - Look at the **Actual Position** register and the **Counts** register.
   - Move the axis to a point where you know what you want the position to be (such as 0). Called this the Desired Position. Look at the **Actual Position** register. Call this position P_0.
   - Calculate the Position Offset using the following equation:

     $$\text{Position Offset} = \text{Desired Position} - P_0.$$

6. Download the Offset

2.2.9. Rotary Scaling

SSI, resolver, and quadrature axes can be configured as rotary axes. To have any useful meaning, the counts from the encoder must be scaled to position units. This topic describes the scaling parameters for a rotary axis and provides instructions on how to scale the axis.

Delta Recommends using the Scale/Offset Wizard for scaling the position. If you need to do it manually, read this topic.

Calculating the Scale Parameters

To calculate the scale parameters, use the Rotational Scale/Offset wizard.

You can also enter the scale parameter manually. Refer to the **Scaling Parameter Details** section below for a description of each parameter.

Scaling Parameter Details

A rotary axis uses the Count Unwind, Position Unwind, Position Offset, and Count Offset parameters to calculate the position units from the counts:

- **Count Unwind**

  The counts for a rotary input are kept within a defined range. When the input goes beyond one end of this range, the counts wrap to the other end of the range. The Count Unwind parameter defines this range. Any control being performed using this input is not interrupted; this is not seen as a position discontinuity.

  For absolute rotary axes, this parameter must be a power of 2. If it were not, the wrap point of the counts would not coincide with the wrap point of the encoder and it would not be possible to determine absolute position.

  RMC Resolver axes will always give 65,536 counts per revolution.

  Example:
  Consider a single-turn SSI encoder with 8192 counts per revolution. If the Count Unwind is set to 8192, the counts will go from 0 to 8191 and wrap once per revolution. If the Count Unwind is set to 2048, the counts will go from 0 to 2048 and will wrap four times per revolution.

- **Position Unwind and Position Offset**

  For the range of counts described above, the positions must be kept within a corresponding range. The Position Unwind and Position Offset parameters define this range.

  The Position Unwind defines how many position units the range will span. The Position Offset adjusts the modulo position range up or down.

  For absolute rotary axes, the Counts are in the opposite direction if the Position Unwind is negative.

  The Counts formula for a positive Position Unwind is:

  Counts = (RawCounts + CountOffset) MOD MaxCounts

  The Counts formula for a positive Negative Unwind is:

  Counts = ([MaxCounts-1] - RawCounts + CountOffset) MOD MaxCounts

  Example:
  Consider an SSI rotary axis with a Count Unwind of 1024. If the Position Unwind is set to 10, the counts range from 0 to 1024 will span 10 position units. If the Count Offset is set to 0, then the range will go from 0 up to, but not including, 10. If the Count Offset is set to 3, then the range will go from 3 up to, but not including, 13.

- **Count Offset**

  The count offset is important for absolute rotary axes. The position range—as defined by the Count Unwind, Position Unwind and Position Offset parameters—will start at the same point where the counts are 0. The Count Offset adjusts the point where the counts are zero, and thereby adjusts the point where the position range will start.

  This parameter is not used by incremental rotary axes, such as quadrature axes.

  Example:
  Consider a single-turn SSI absolute rotary encoder with 2048 counts. The axis is set up with a Count Unwind of 2048, a Position Unwind of 360 and a Position Offset of -180. Therefore, the position range will go from -180 to 180 in one revolution of the encoder. Negative 180 will occur at the point where the transducer counts are 0. Now, assume the encoder was mounted on the machine such that -180 pointed straight up, but you want -180 to be pointing down. To fix this, rotate the encoder so that it is pointing straight down. Record the counts, and enter the negative of that value in the Count Offset parameter. -180 will now be pointing down.

2.2.10. Quadrature Scaling (Linear)

To have any useful meaning, the counts from the transducer must be scaled to position units. For a quadrature axis, the Position Scale parameter defines the position units as a function of transducer counts. This topic describes how to correctly calculate this parameter for an Quadrature encoder.

Delta recommends using the Scale/Offset Wizard for scaling the position. If you need to manually calculate the scale and offset, read this topic.

If your motion is linear, read this topic. If your motion is purely rotary, see the Rotary Scaling topic.

Pulses, Lines and Counts

As described in the Quadrature Fundamentals topic, the resolution of a quadrature encoder is typically given in Pulses per Revolution (PPR), also called Lines per Revolution. The RMC feedback sees each rising or falling edge of the A or B signal as one count. Therefore, each pulse or line of the encoder will result in four counts on the RMC quadrature input. For example, a 1000-line encoder will give the RMC 4000 counts per revolution.

Scaling Counts to Position Units

The RMC calculates the quadrature axis Actual Position every control-loop time using the following formula:

Actual Position [pu] = (Change in Counts [cnt] x Position Scale [pu/cnt]) + Last Position

Note:
If the Actual Position filter is applied, the RMC filters the Actual Position after calculating it with the above formula.

Manually Calculating the Scale and Offset

The Scale/Offset Wizards provide the easiest method of scaling your axis. If you prefer to do it manually, read this section.

Determining the Scale and Offset

There are two methods of finding the correct Position Scale and Position Offset.

Method 1: P_0/P_1 Calculation

The accuracy of this method depends on how accurately you can measure two positions of the axis.

1. Physically measure the axis position at two points and record the value of the counts register at each point. Call the first measured position P_0, and its corresponding Counts C_0. Call the second measured position P_1, and its corresponding Counts C_1.
2. Calculate the Position Scale with the following equation:

$$\text{Position Scale} = (P_0-P_1)/(C_0-C_1)$$

Method 2: Using the Encoder Resolution

1. Obtain the encoder resolution (usually from the transducer data sheet), for example 4000 counts/turn.
2. Determine the position units you would like for your Actual Position, for example inches, meter, mm, etc.

3. Determine how many position units correspond to one count.
   For example, consider an encoder with 4000 counts per revolution mounted to a motor that drives a linear axis through a 4:1 gearbox and then a rack and pinion. The rack moves 5 inches for every turn of the pinion. The desired position-units of the linear rack motion are inches.
   To find the resolution:
   a. Determine the number of counts generated in one revolution of the encoder: From the datasheet, this is 4000 counts.
   b. Calculate the linear travel (position units) in one revolution of the encoder: One revolution of the encoder equals 1/4 revolution of the pinion. One revolution of the pinion equals a travel distance of 5 inches. Therefore, the linear travel in one turn of the encoder is 5/4 = 1.2 in.
   c. To calculate Scale, divide the calculated linear travel (position-units) by the number of counts:
   1.2/4000 = **0.0003** position-units/count

4. If you wish to reverse the direction of the feedback, make the Scale negative.
5. Download the Scale to the RMC.

2.2.11. Scaling Rates on a Position Axis

Revolutions per Minute and Feet per Minute

If your system is using a position feedback device (such as an encoder), but your system is primarily intended for velocity control, and the system must be scaled so that the velocity of the system is given in feet per minute or revolutions per minute, then this topic is for you.

If you are using a feedback device that gives only velocity feedback, such as a tachometer, see the Analog Velocity Scaling topic instead.

To scale a position system to show the velocity in fpm or rpm, you need to set the Position Scale parameter. The Position Offset need not be set if you are only concerned with velocity and not positions.

Position Scale Parameter for a Linear Axis

1. Determine how many Counts the feedback device gives in one foot or one revolution.
2. Open the Scale/Offset Wizard. If it gives a choice, choose **Position/Counts**.
3. In the **First Measured Position** section, enter zero in both boxes.
4. In the **Second Measured Position** section:
   - Enter the number of counts per foot or revolution in the **Raw Counts** box.
   - Enter 60 in the **Measured Position** box.
5. Click **Next**. The calculated Position Scale will be displayed.
6. Click **Next** and complete the wizard.
7. Run the system and make sure that the Actual Velocity is correct.
8. Set the Display Units axis parameter to the desired units. For RPM, choose rpm·s. This will result in the velocity being displayed as rpm. For FPM, choose fpm·s. This will result in the velocity being displayed as fpm.

Position Scale Parameter for a Rotary Axis

To scale a rotary axis to RPM:

1. Set the Position Unwind to 60.

2. Set the Display Units axis parameter to rpm·s. This will result in the velocity being displayed as rpm.

2.3. Tuning

2.3.1. Tuning Overview

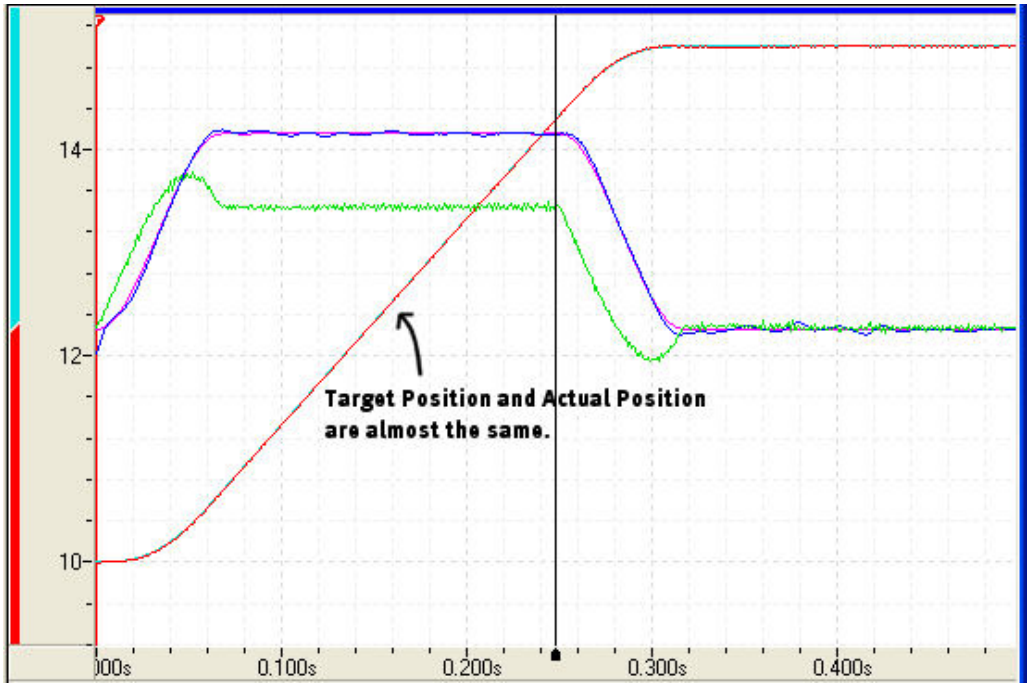
Once your system is set up and ready for use, it must be tuned in order to control it. Tuning is the process of adjusting the tuning parameters for optimum control of the system. The better tuned a system is, the closer the actual movement follows the desired path of movement.

Why is Tuning Necessary?

Tuning is required for moving an axis in closed loop control. In closed loop control, the RMC generates a target profile. The tuning parameters (gains) then dictate how much Control Output the RMC should generate each loop time in order to get the Actual Position to follow the target profile as closely as possible. The gains must be properly tuned to achieve precise motion.

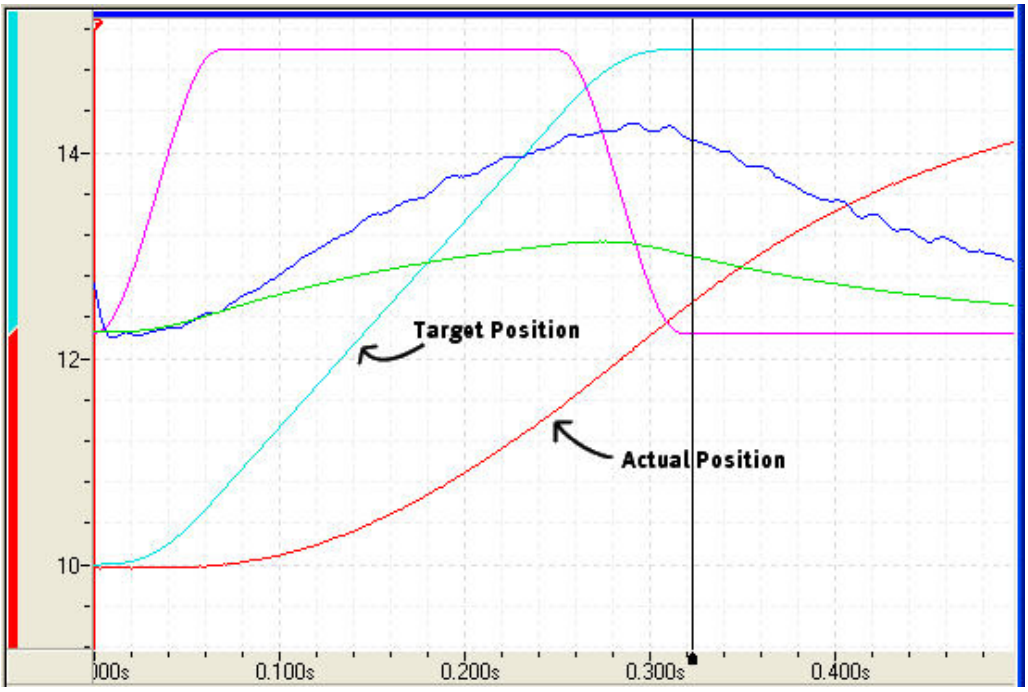
On a well-tuned system, the Actual Position will closely follow the Target Position. To see a plot, **click here**.

This is a plot of a system that is well-tuned:



On a poorly tuned system, the Actual Position will not follow the Target Position very well. To see a plot, **click here**.

This is a plot of a system that is poorly tuned:



Tuning Wizard and Autotuning

The Tuning Wizard makes the tuning process very easy. The Tuning Wizard provides *autotuning* and *tuning based on existing plot*.

Autotuning

Autotuning automatically moves the axis and then provides a range of gains from which you can choose with a simple slider bar. The RMC autotuning can be used on many systems. Even if autotuning is used, the user should posses a solid understanding of manual tuning to ensure proper tuning.

Autotuning can be used for position axes, including rotary and linear axes, and motors in velocity mode or torque mode. Autotuning does not support velocity, pressure, or force axes, although pressure or force axes can be tuned using an existing plot as described below.

See the Autotuning topic for details on how to perform autotuning.

Autotuning may not work for every situation. If it doesn't work for tuning your axis, you can use Tuning Wizard with an existing plot, as explained below.

Tuning Using an Existing Plot

If your axis cannot be autotuned, use this method to tune your axis:

1. Tune the axis using only the Proportional Gain.
2. Increase the Proportional Gain until the axis tracks well.
3. Move the axis in both directions, making sure to get a plot of each direction of motion.
4. In the Tuning Tools, click the **Tuning Wizard** button.
5. In the wizard, choose **Use Existing Plots** and continue.
6. In the wizard, choose the two recent plots you made of motion.
7. Complete the wizard, and the Gain Calculator will open. Now you can choose gains and move the axis.

See the Tuning Wizard topic for more details.

Note: Plots chosen for the **Use an Existing Plot** method should have been captured in capture mode. Do not use plots captured in trend mode, as they can cause inaccurate results.

Manual Tuning

Manual Tuning Procedures

Tuning procedures differ depending on the type of system. Please read the **Tuning Guidelines** below before continuing to any of the tuning procedures. Click the following links for suggested tuning procedures:

- Tuning a Hydraulic Position Axis or Motor in Velocity Mode **- the most common tuning procedure**
- Tuning a Motor in Torque Mode
- Tuning Position-Pressure or Position-Force
- Tuning Pressure/Force
- Tuning Active Damping and Acceleration Control
- Tuning a Pneumatic System

Manual Tuning Guidelines

Keep the following guidelines in mind throughout the tuning procedure. There is no substitute for experience when tuning an axis. The procedures offer some guidelines, tips, and suggestions for tuning your system. While the steps will work for many systems, they may not be the best for a particular system.

- **Use the Tuning Tools**
  The Tuning Tools in the Plot Manager provide a single place where you can issue repeated commands and change and download gains. In addition, plots are automatically uploaded after a command.
- **Reiterate these Steps:**
  The tuning procedure is a reiteration of the following general steps. Use these steps throughout the tuning procedure:
  1. **Make a move**
     Typically, this is done using the Move Absolute (20) command. The Tuning Tools provide two buttons to easily issue two commands, typically one for each direction.

     Tip:
     For a typical hydraulic cylinder, the **Accel** and **Decel** parameters of the Move absolute command should be on the order of 20 -100 pu/sec^2. The speed is typically between 1 and 30 pu/sec.
  2. **View the Plot**
     When you issue a command in the Tuning Tools, the plot is automatically uploaded from the RMC. The plot will help you determine which parameters must be changed.
     In the Tuning Tools, you can easily disable the automatic plot upload for one of the command buttons. If you are focusing on tuning a certain direction, you can keep the plot from uploading when moving the other direction.
  3. **Change a Tuning Parameter**
     Change a gain in the Axis Parameters section of the Tuning Tools. After changing the value you must click the download button  or press Ctrl+D to download the changes to the RMC.
  4. **Repeat these steps**
     Repeat these steps until the parameter is at the desired value.

- **Use the Mean Squared Error**
  The Mean Squared Error in the RMCTools Plot Manager can help determine how the last parameter change affected the system. The Mean Squared Error provides a rough indication of how closely the Actual Position is tracking the Target Position. If this number decreases significantly, the last parameter change was good. If this number increases significantly, the last parameter change was bad.
- **Start with Long, Slow Moves, then Use Faster Moves**
  Begin the tuning procedure with fairly long, slow moves and low accelerations. This will prevent you from losing control of and potentially damaging the system. The moves should be short enough to fit in the plot.
  After gaining control of the system, you should increase the speed to values you expect to use during machine operation. Always tune your system at the speeds of intended operation.
- **Set Auto Stops**
  You may want to turn off some of the Auto Stop bits. The Auto Stops cause the axis to halt if an error occurs. In the initial stages of tuning, a Following Error or other error may occur, causing an undesired halt. Setting these bits to "Status Only" will make the RMC ignore the errors so you can tune the axis. Once you gain sufficient control of the axis, set the Auto Stop bits to halt. Turning off the Auto Stops may not be possible on some systems because of safety concerns.
- **Update Flash and Save the Project**
  When editing the parameters in the project, you must download them to apply the changes to the RMC. However, they are not updated in the RMC's Flash memory until you issue an Update Flash command. This will save the parameters even if power is disconnected. To save the RMC's parameters to the project, upload the parameters from the RMC and then save the project.

2.3.2. Tuning Wizard

The Tuning Wizard automatically calculates tuning gains for you. The wizard first guides you through the process of obtaining plots, then automatically calculates a range of valid gains, and finally presents you with a simple slider bar to choose conservative or aggressive gains.

The tuning wizard can be used with position, pressure and force axes. The tuning wizard cannot be used for velocity axes.

The Tuning Wizard works as follows:

1. Using plots of motion, the Tuning Wizard computes a mathematical model of the system. The Tuning Wizard provides two methods of obtaining plots:
   - **Autotuning Wizard - Position Axes Only**
     The Autotuning Wizard automatically moves the axis as specified by the user. With the plots generated by the motion, the Autotuning Wizard determines the system model.
   - **Using Existing Plots - Position, Pressure, and Force Axes**
     For position axes, the user chooses existing plots of motion from which the Tuning Wizard determines a system model. The existing plots must be listed in the Plot History pane.
     For pressure or force axes, the Tuning Wizard uses the currently displayed plot to determine a system model.

2. The Gain Calculator uses the system model to determines valid gains. In the Gain Calculator, the user positions a slider bar to pick from a range of appropriate gains for the system. The Gain Calculator stays open, making it easy to move the axis to try out various sets of gains. If a certain set of gains is not acceptable, simply move the slider and click **Apply** to try out a new set of gains.

Using the Tuning Wizard

Position Axes:

1. **Open the Tuning Wizard:**
   In the Plot Manager, on the **Tuning** tab, click **Tuning Wizard**.
2. Choose **Use Autotuning Wizard** if you would like RMCTools to move the axis and compute a system model. Choose **Use an existing plot** if you already have plots of motion of your system that can be used to compute a system model.
   a. **Autotuning**
      If you chose **Use Autotuning Wizard**, follow the instructions in the wizard. For more detailed information, click the **Help** button in the wizard.
   b. **Use an Existing Plot**
      If you chose **Select an existing plot**, you must already have plots of motion of your system in the Plot History pane. Make sure the Output Polarity was correct when the plot was captured, that is, that a positive Control Output results in movement in the direction of increasing position units.
      For details on generating plots suitable for the Tuning Wizard, see the Creating Plots for the Tuning Wizard topic.
3. After the wizard completes, it will open the Gain Calculator. For more details, see the Gain Calculator topic.

Pressure or Force Axes:

1. First, generate a plot that shows the pressure or force changing. To generate a good plot, do the following:
   a. Set the Pressure/Force Proportional Gain to a small value. For systems scaled in pounds, 0.01 is often a good starting value.
   b. Move the axis to where it encounters pressure or force.
   c. Use the Hold Current Pressure/Force (19) command to enter pressure or force control.
   d. Increase the Pressure/Force Proportional Gain until the Actual Pressure/Force starts moving toward the Target Pressure/Force. It should not get all the way toward to Target Pressure/Force. Give just enough gain to make sure it comes somewhat close.
   e. Use the Ramp Pressure/Force (S-Curve) (41) command to ramp the pressure or force.
   f. Upload the plot. Make sure the Actual Pressure or Force changed significantly in the plot. If it did not, repeat the process. For more details on generating plots suitable for the Tuning Wizard, see the Creating Plots for the Tuning Wizard topic.
2. In the Plot Manager, on the **Tuning** tab, choose the **Pressure** or **Force** tab and click **Tuning Wizard**.
3. Choose your plot from the list and click **Next**.
4. A dialog will appear with the calculated model. Click **Finish** to open the Gain Calculator. For more details, see the Gain Calculator topic.

2.3.3. Autotuning

Autotuning is part of the Tuning Wizard. Autotuning automatically moves the axis as specified by the user. With the plots generated by the motion, RMCTools determines the system model, from which it determines valid gains (if you prefer to use plots you took previously, the Tuning Wizard can generate a model from existing plots). The user then positions a slider bar on the Gain Calculator to pick from a range of appropriate gains for the system.

Autotuning can only be used to calculate control gains for position axes. It does not support calculating velocity, pressure, or force control gains.

Autotuning **can** be used for:

- Linear position axes such as with hydraulic, pneumatic, or other velocity mode motors or actuators.
- Rotary position axes with hydraulic, pneumatic, or other velocity mode motors or actuators.
- Rotary position axes with torque mode motors.

For position, pressure or force axes types that cannot be autotuned, the Tuning Wizard can still be used to calculate control gains based on plots manually generated by the user. See the Tuning Wizard topic for details.

Using Autotuning

Before autotuning, make sure you have completed all the pre-tuning steps in the Startup Procedure and all the pre-Tuning Tools steps in the tuning procedure you are using. Specifically, the axis must be properly scaled, the Output Polarity must be correct, the Positive and Negative Travel Limits must be set (for linear axes), and the axis must be able to move in open loop.

Motion control systems often do not behave identically at low speeds and high speeds. Therefore, it may be wise to use the Tuning Wizard as a first pass at tuning. Once the system is controllable, you can capture plots of motion at the intended speed and conditions of normal machine operation. You can then use these plots in the Tuning Wizard to compute a system model that more accurately represents the system during its normal operation.

To start autotuning:

1. Launch the Tuning Wizard:
    - From the Axis Tools, on the **Tune** tab of the **Axis Parameters** pane, in the **Tools and Wizards** section, on the **Position Tuning Wizard** row, click **Launch**.
    - Or from the Plot Manager, on the **Tuning** tab, click **Tuning Wizard**.
2. In the Tuning Wizard, choose **Use Autotuning Wizard**, and click **Next**.
3. Follow the instructions of the Tuning Wizard.

If Autotuning Doesn't Work

Autotuning may not work for every situation. If it doesn't work for tuning your axis, you can still take advantage of the Tuning Wizard and the Gain Calculator as follows:

1. Use the Tuning Tools to tune the axis with only the Proportional Gain.
2. Increase the Proportional Gain until the axis is tracking reasonably well.
3. Make sure to get plots of motion in both direction.
4. Open the Tuning Wizard and choose **Use an Existing Plot**.
5. Complete the tuning wizard, selecting the plots you made.
6. After the completing the wizard, the Gain Calculator will open. Use the slider bar to pick from a range of appropriate gains for the system.

2.3.4. Creating Plots for Tuning

To successfully calculate valid gains using the **Tuning Wizard - Use Existing Plot** method, you must first generate good plots of motion. The plots should show some motion with

changes. Make sure the plot does not include hitting the end, getting stuck, etc. Make sure the Output Polarity was correct when the plot was captured, that is, that a positive Control Output results in movement in the direction of increasing position units.

Plots of motion should:

- Contain significant motion. Slight changes in position, pressure, or force are usually not enough.
- Include the Control Output and the Actual Position, Actual Pressure, or Actual Force.
- Not include long sections of no motion.
- Contain motion in one direction only. If your position system behaves differently in each direction of motion, you will need to generate two plots—one for each direction of motion.
- Be captured at the smallest loop time possible and uploaded in capture mode. Do NOT use plots captured in trend mode, as they can cause inaccurate results.

Generating Plots

To generate a plot of motion:

1. Tune the axis using only the Proportional Gain.
2. Increase the Proportional Gain until the axis tracks well.
3. Move the axis in both directions, making sure to get a plot of each direction of motion.
4. In the Tuning Tools, click the **Tuning Wizard** button.
5. In the wizard, choose **Use Existing Plots** and continue.
6. In the wizard, choose the two recent plots you made of motion.
7. Complete the wizard, and the Gain Calculator will open. Now you can choose gains and move the axis.

If you are not able to move the axis with Proportional Gain, try using Open Loop commands to move the axis. With the Open Loop Rate (10) command, move the axis and stop it, then upload the plot. You may need to set the plot duration to be long enough to capture both the start and stop in the plot.

2.3.5. Gain Calculator

The Gain Calculator calculates gains for the axis based on the system model that was generated by the Tuning Wizard. After the Tuning Wizard completes, the Gain Calculator opens. Using a simple slider bar, the user can choose gains from conservative to aggressive. While the Gain Calculator is open, the user can use the Tuning Tools to send move commands to the axis to try various gains until satisfied.

If the axis has multiple gain sets, the Gain Calculator will apply the gains to the first gain set.

Using the Gain Calculator

Opening the Gain Calculator

In the Plot Manager, in the Tuning Tools, click **Gain Calculator**. If the axis does not have a model, you will be prompted to first run the Tuning Wizard. When the wizard completes, the Gain Calculator will open.

Choosing and Applying Gains

1. Move the slider bar to choose a set of gains. Always start with conservative gains.
2. Click **Apply Gains** to download the gains to the RMC.

3. Move the axis to test the gains. You can leave the Gain Calculator open while you make moves.
   Use the command buttons in the Tuning Tools to do this:
   1. On the **Tuning** tab, click one of the command buttons labeled **Click to set up**.
   2. Enter the position, velocity, acceleration, and deceleration for the Move Absolute command. For systems scaled in inches, typical Accel and Decel values are 10 to 100.
   3. Do the same for the other command button, using the same velocity, acceleration, and deceleration, but use a different position.
   4. You can click the buttons you just configured to move the axis back and forth to try your selected gains. The plots will automatically be uploaded so you can see how the motion is performing.
4. Repeat until you are satisfied that the axis is controlling well.

2.3.6. Tuning a Position Axis

The following manual tuning procedure may be used to tune many position axes, hydraulic axes and motors in velocity mode. This procedure works for Position PID control. Please read the Tuning Overview topic before following this procedure. There is no substitute for experience when tuning an axis. This procedure offers some guidelines, tips, and suggestions for tuning your system. While these steps will work for some systems, they may not be the best for a particular system.

Before beginning the tuning procedure, set all the gains and feed forwards on the axis to zero.

Tuning Procedure

Before beginning the tuning procedure, make sure you have completed the steps in the StartUp Procedure up to the Tuning section. You must be able to move your actuator properly (a positive Control Output causes increasing position) and have valid feedback.

Pre-Tuning Steps:

These steps come before actually tuning the gains.

1. Test Wiring and Polarity

This step is for verifying that the system wiring and setup is correct before doing any closed loop control.

- In the Command Tool, issue a Direct Output (9) command to the axis. Use small **Output** value, such as 0.050-0.150 V. Use a **Ramp Rate** of 100. If the axis does not move, increase the **Output** until the axis begins to move.

  DANGER: The Direct Output (9) command disables the safety features on the RMC! Use this command carefully!

- A positive Control Output should yield increasing position units. If it does not, do the following:
  i. On the RMCTools toolbar, click the Axis Tools button.
  ii. In the Axis Parameters Pane, on the **Setup** tab, toggle the **Invert Output Polarity** parameter, then click the **Download** button.
- Issue a Direct Output (9) command again with a negative drive. This should yield decreasing position units.

2. Check the Deadband

If the axis exhibits a deadband, you may need to use the Deadband parameters.

Check Whether the System Exhibits a Deadband:

a. Give increasing amounts of **Output** to the axis with the Direct Output (9) command until the system starts to move.

b. The value of **Output** at which the system starts to move is your deadband. If this value is approximately 0.4 V or greater, or if the axis begins to move quickly at the deadband value, you should probably use the deadband parameters. If the point at which it begins to move it is less than 0.4 V, it is left to the discretion of the designer.

Set the Deadband Parameters:

If you found that your system has a deadband, set the deadband parameters in the following manner:

a. On the RMCTools toolbar, click the **Axis Tools** button.

b. In the Axis Parameters Pane, on the **All** tab, expand the **Output** section.

c. Set the Output Deadband parameter to the value of your deadband.

d. Set the Deadband Tolerance to a small value.

e. Click the **Download** button to apply the changes to the RMC.

3. Non-linear **V**alves

If you are using a non-linear hydraulic valve, set the valve linearization parameters as described in the Valve Linearization topic. Delta does not recommend using a non-linear valve if a linear valve is available.

4. Adjust the Output Bias

Some systems may drift significantly when the **Control Output** is at zero volts, which may adversely affect control. Use the **Output Bias** parameter to adjust the output such that the axis does not move when you issue an Open Loop command with zero volts **Control Output**. The RMC always adds the **Output Bias** to the **Control Output**.

Check if your system need Output Bias:

a. Issue an Open Loop Rate (10) command to the axis. Use an **Output** of zero and a **Ramp Rate** of 100.

b. If the axis moves significantly, you need to set the Output Bias.

Set the Output Bias:

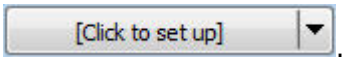
a. In the **Axis Tools**, in the Axis Parameters Pane, on the **Tune** tab, enter a small number in the **Output Bias** parameter, such as 0.05 V or -0.05 V.

b. Keep increasing (or decreasing) the number until the axis stands still.

c. Click the **Download** button to apply the changes to the RMC.

5. Set Up the Tuning Tools

In the Tuning Tools, set up the command buttons so that one button will move the axis one direction, and the other will move it in the other direction.

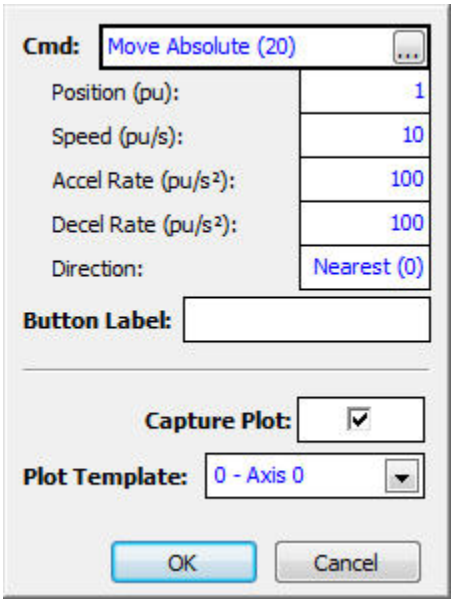
a. Click the down arrow on the command button 

.

b. Enter the Position, Speed, Acceleration, Deceleration, and Direction (**Nearest** for linear axes) for the Move Absolute command, then click **OK**.

> **Tip:**
> For a typical hydraulic cylinder, the **Accel** and **Decel** parameters of the Move absolute command are typically on the order of 10 to 100pu/sec^2. The speed is typically between 1 and 30pu/sec.

Example:



c. Repeat the previous step for the other command button. Enter the same velocity, acceleration, and deceleration, but a different position.

6. Set Symmetrical/Ratioed

If your system behaves differently in each direction of motion, you will need to set the Symmetrical/Ratioed parameter is set to **Ratioed**. This will provide two Velocity Feed Forwards, one in each direction of motion. The other gains will be ratioed according to the Velocity Feed Forwards, resulting in the same control in both directions.

If you have a single-rod hydraulic cylinder, choose **Ratioed**.

The Symmetrical/Ratioed parameter can be accessed in the **Tuning Tools** on the **Tune** tab in the **Axis Parameters** section. Remember to download the changes by clicking the **Download** button.

Tuning Steps:

These steps are for adjusting the tuning gains. Use the Tuning Tools for these steps.

6. Adjust the Proportional Gain

The Proportional Gain must be adjusted to gain some control over the system for continuing the tuning procedure. You will fine-tune it later.

- Start with a small value of Proportional Gain. Remember that it is a floating point number, and you may have to start with a number smaller than 1, depending on your system. When you change the gain, click the **Download** button to apply the change to the RMC.
- From the Command Tool, issue the Move Absolute (20) command to move the axis.

> **Tip:**
> For a typical hydraulic cylinder, the **Accel** and Decel parameters of the Move absolute command should be on the order of 20 -100. The speed is typically between 1 and 30.

- Slowly increase the Proportional Gain as you make moves. Increase it until the following are true:
  - The Actual Position gets to the Command Position reasonably quickly.

- During the constant speed portion of the move, the Actual Position parallels the Target Position.

If the system begins to oscillate, decrease the gain. In this step, do not expect the Actual Position to track the Target Position very well during the move.

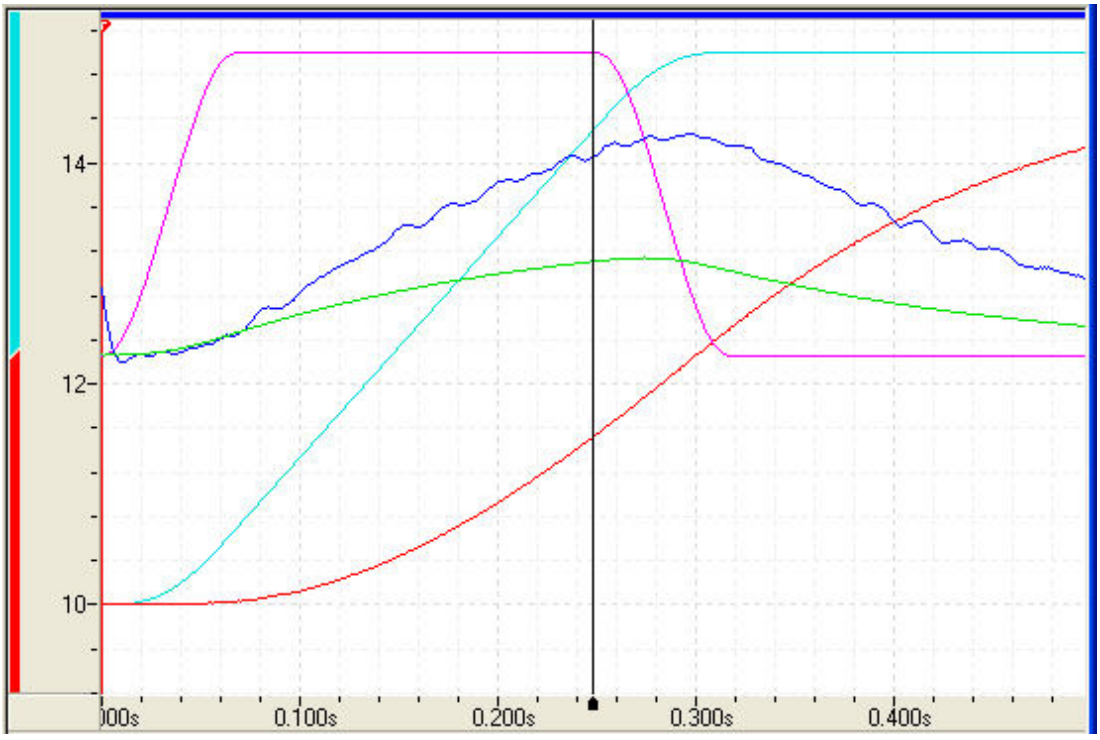
- Once you gain control over the system, increase the speeds and accelerations of your commands to values that will be used during normal machine operation.

Plots

Look at these plots for examples of good and bad plots at this step. Be aware that not all systems will be like these examples.

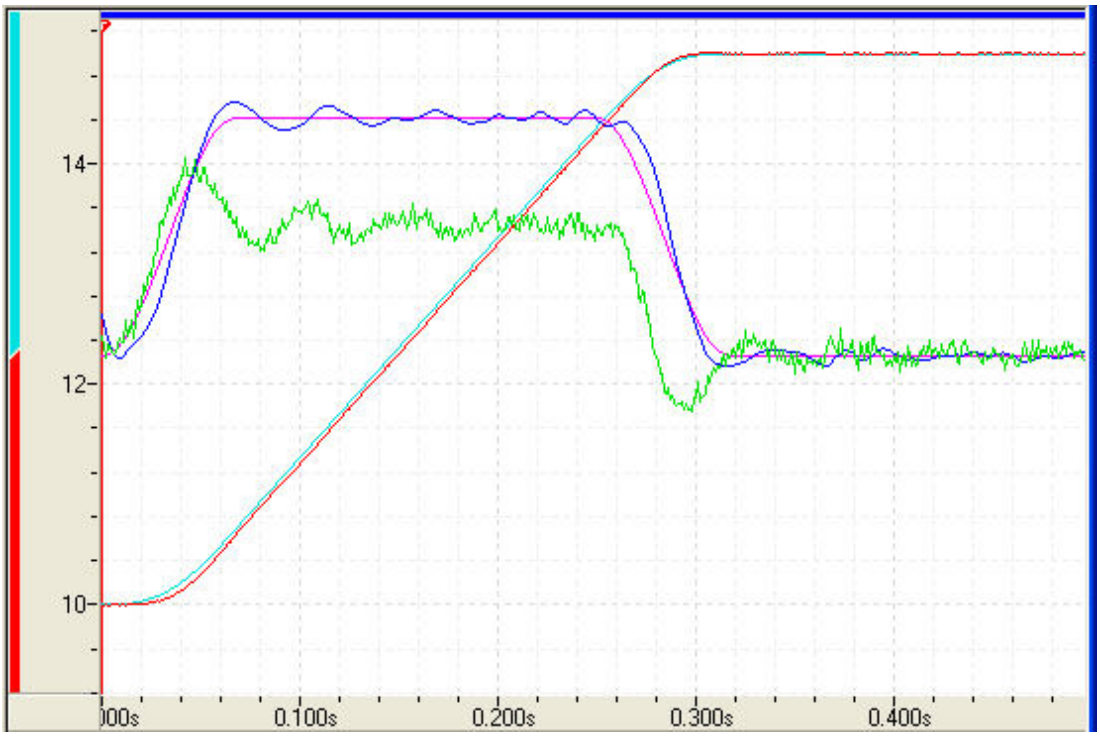
Too little Proportional Gain

The Actual Position lags the Target Position and takes a long time to get to the Command Position.



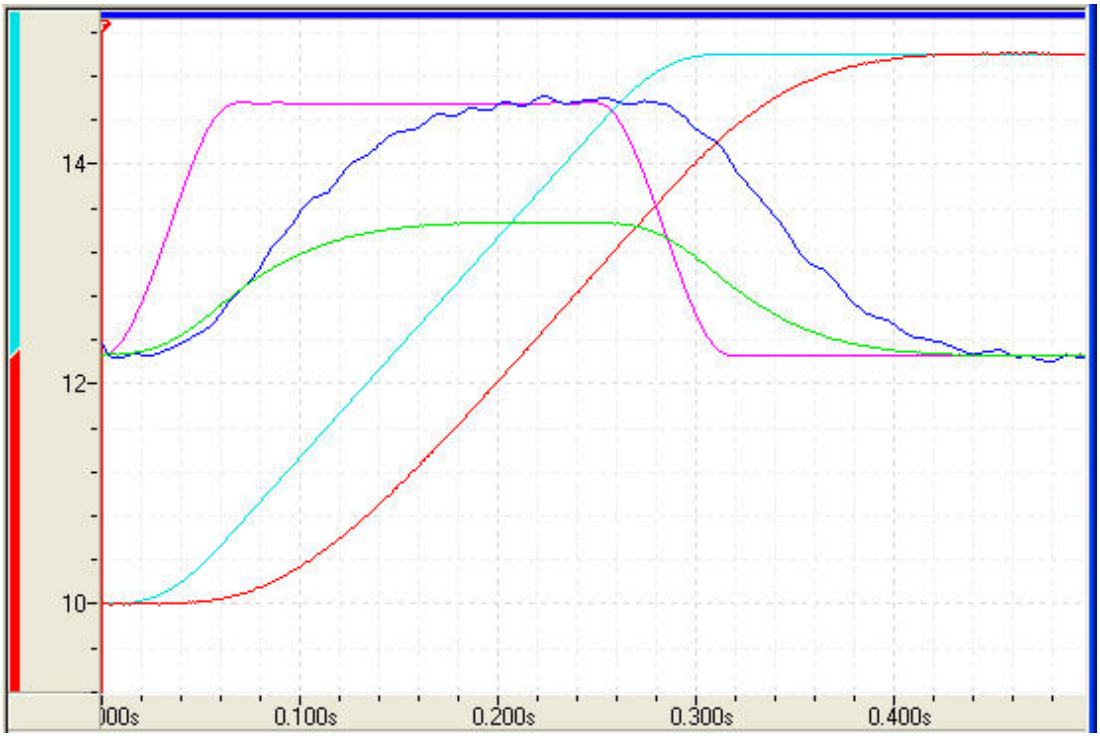
Too much Proportional Gain

The axis has started to oscillate, which is evident from the Actual Speed and Control Output.



Correct Proportional Gain

The Actual Position parallels the Target Position during the constant velocity portion of the move and does not overshoot.



7. Adjust the Velocity Feed Forward

In many systems the Velocity Feed Forward parameter is the most important parameter for position tracking during a move. To adjust the Velocity Feed Forward:

- Start with a small value of Velocity Feed Forward. Remember that it is a floating point number, and you may have to start with a number smaller than 1, depending on your system.

  Tip:
  Use the **Adjust VFF** button in the Tuning Tools to automatically determine the Velocity Feed Forward. Make a move, wait for it to complete, then click **Adjust VFF**. Repeat for the other direction.

- Make long slow moves in both directions. Adjust the Velocity Feed Forward until the axis tracks within 10% in both directions.
- If you are using Ratioed gains, make sure to adjust both Velocity Feed Forwards.
- Click the **Download** button to apply the changes to the RMC.

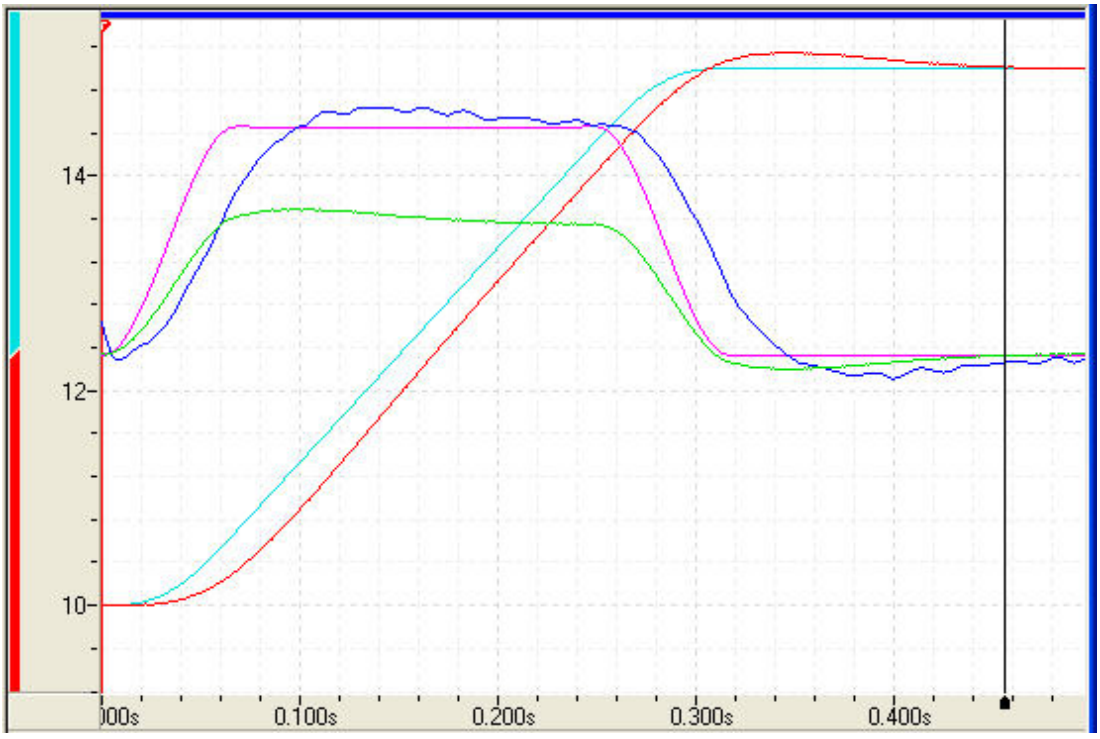
If the Symmetrical/Ratioed parameter is set to **Ratioed**, you will have two Velocity Feed Forwards, one for each direction of motion. Make sure to tune both.

Plots

Look at these plots for examples of good and bad plots at this step. Be aware that not all systems will be like these examples.

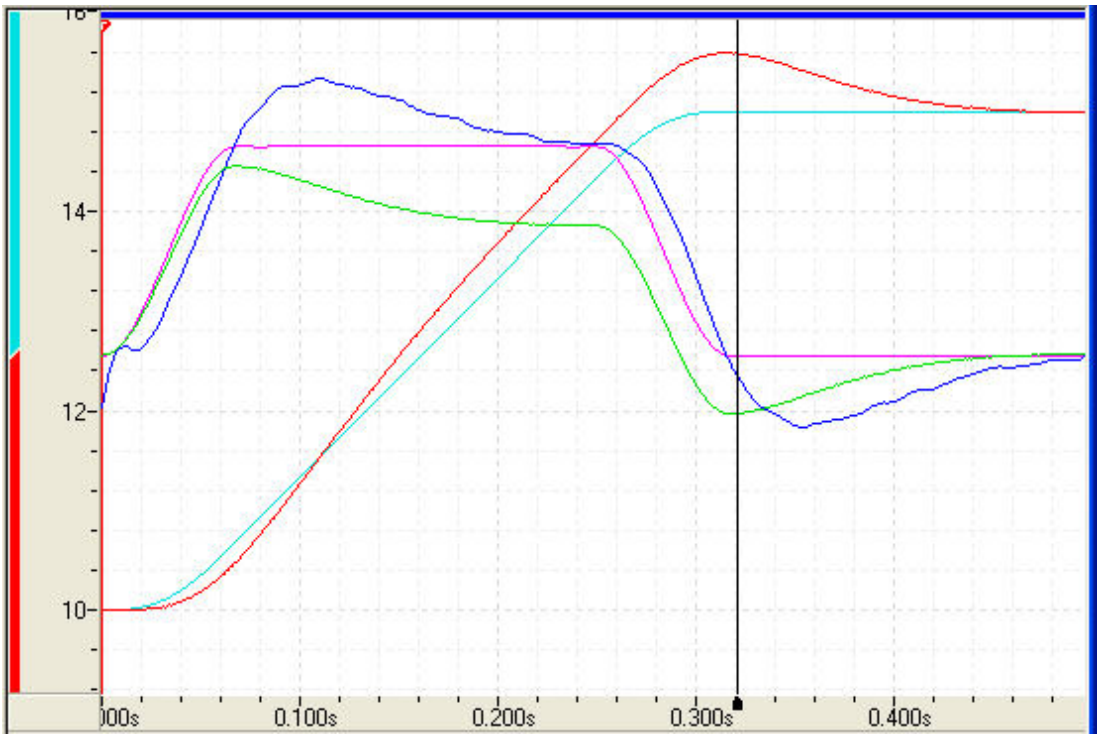
Too little Velocity Feed Forward

The Actual Position lags the Target Position during the entire move.

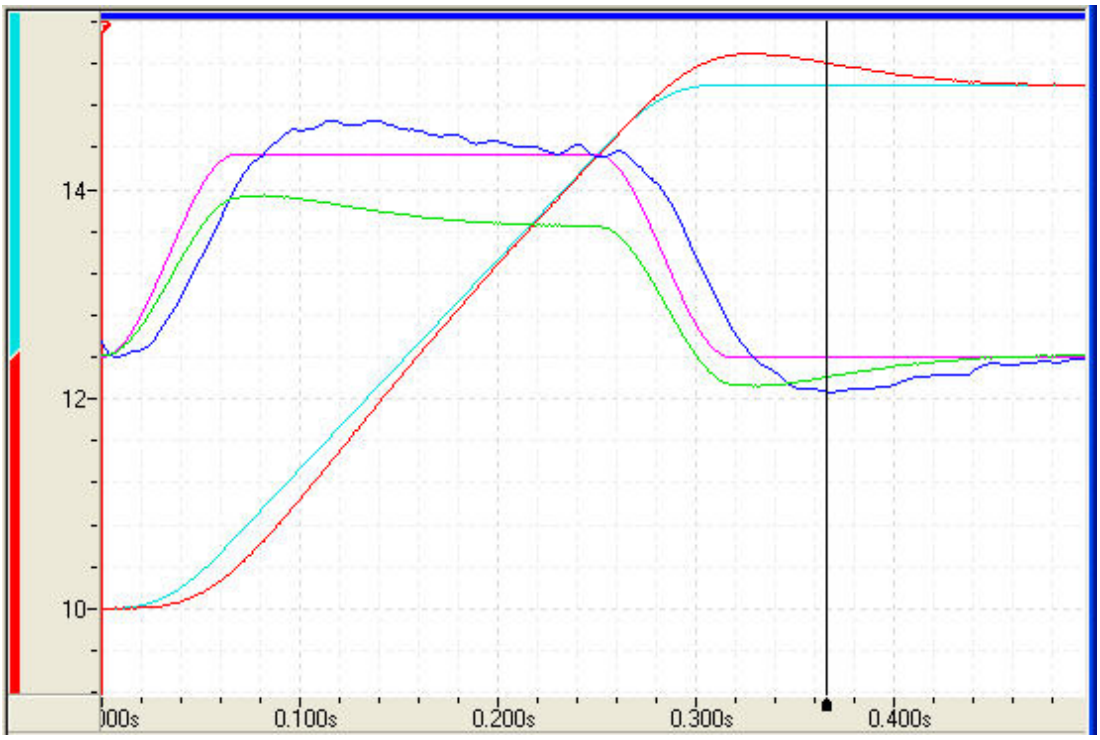


Too much Velocity Feed Forward

The Actual Position starts leading the Target Position.



Correct Velocity Feed Forward
The Actual Position tracks the Target Position perfectly during the latter half of the move.



8. Adjust the Acceleration Feed Forward

The Acceleration Feed Forward parameter is particularly useful for systems moving large masses with relatively small cylinders. Such systems often have a delay before the start of movement. The Acceleration Feed Forward terms can help compensate for this delay.

- Start with a small value of Acceleration Feed Forward. Remember that it is a floating point number, and you will very likely have to start with a number much smaller than 1, such as 0.001, depending on your system.

  Tip:
  For a 2.5 in bore hydraulic cylinder with a max velocity of 30 in/sec, the Acceleration Feed Forward is typically on the order of 0.01 to 0.3. Start with a small value.

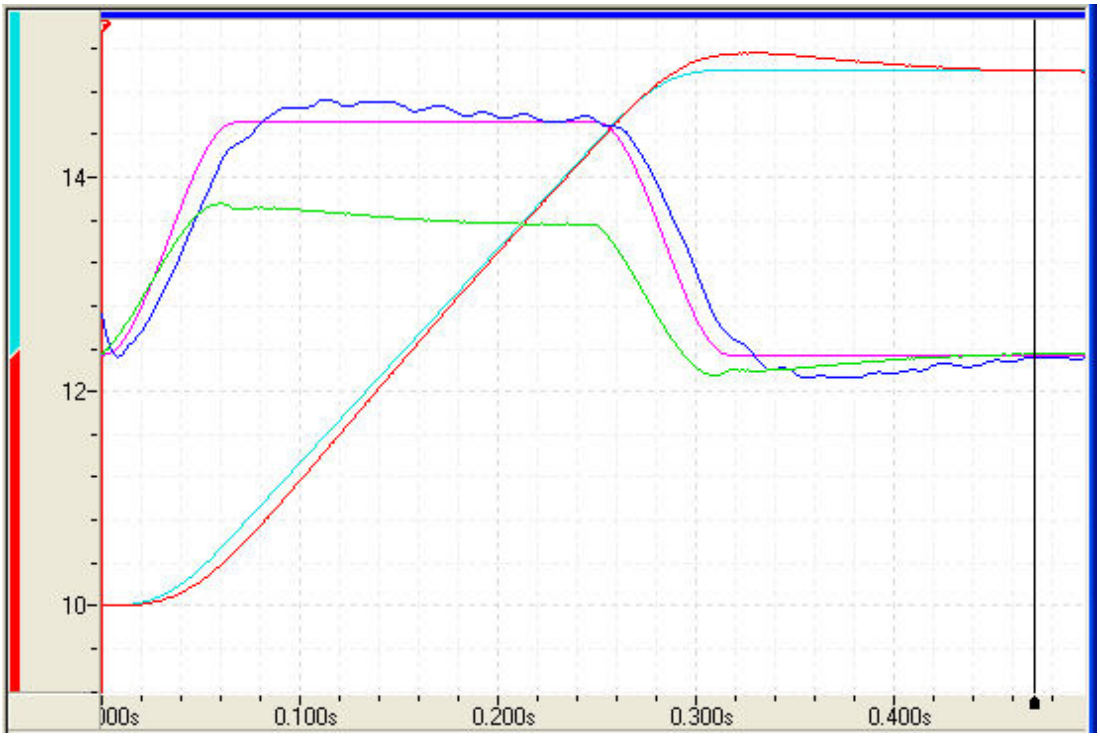
- Look for following errors during acceleration and deceleration. Increase the Acceleration Feed Forward parameter until the errors disappear.
- If you are using Ratioed gains, make sure to adjust both Acceleration Feed Forwards.
- Click the **Download** button to apply the changes to the RMC.

Plots

Look at these plots for examples of good and bad plots at this step. Be aware that not all systems will be like these examples.

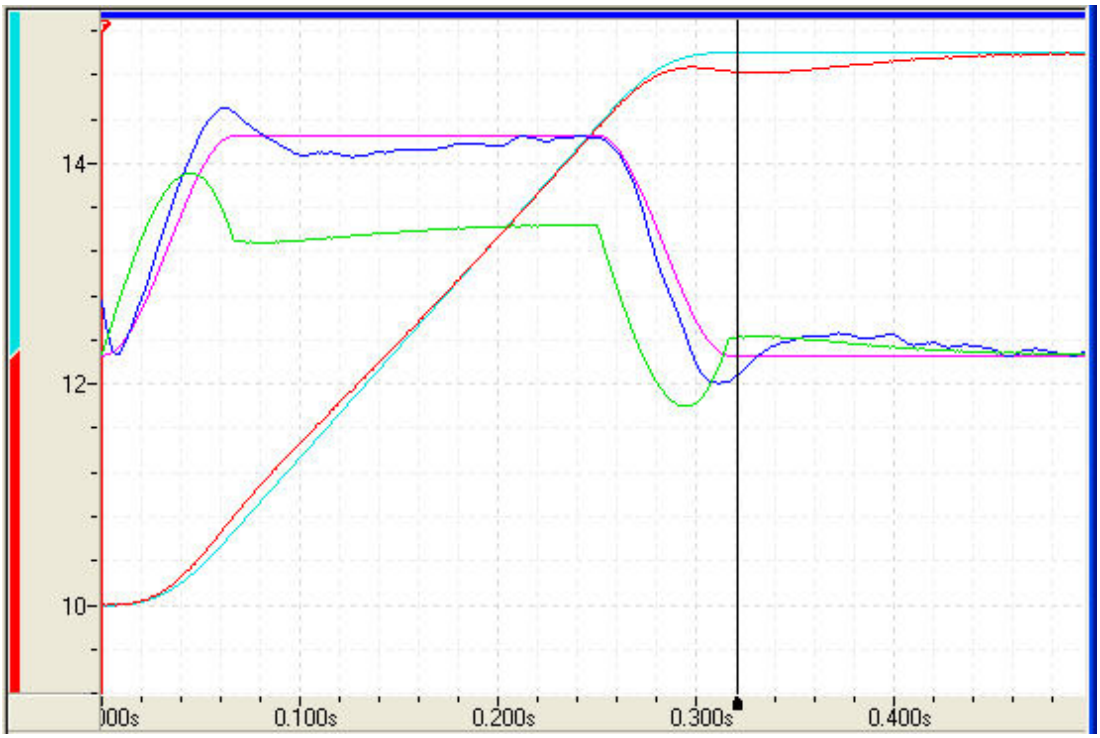
Too little Acceleration Feed Forward

The Actual Position lags the Target Position during the acceleration at the beginning of the move, and overshoots during the deceleration at the end of the move.



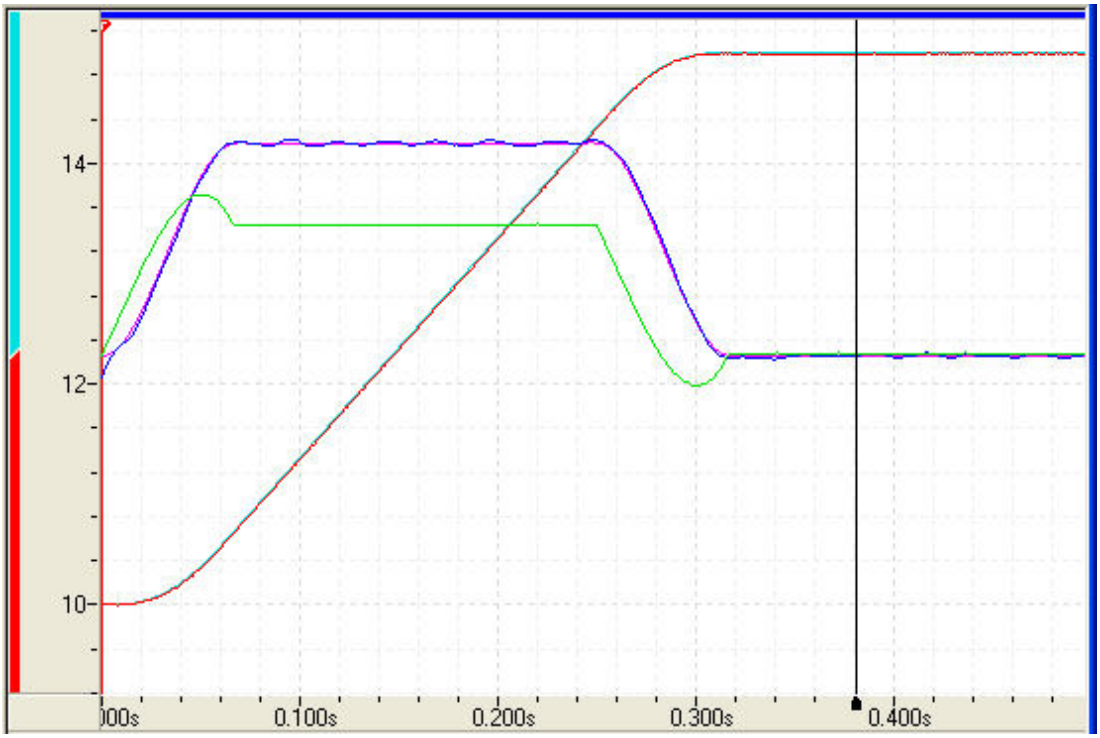
Too much Acceleration Feed Forward

The Actual Position starts leading during the acceleration and undershoots during the deceleration.



Correct Acceleration Feed Forward

The Actual Position tracks the Target Position well during the acceleration and deceleration.



9. Readjust the Proportional Gain

Proportional Gain affects the responsiveness of the system. Low gains make the system sluggish and unresponsive. Gains that are too high make the axis oscillate or vibrate.

- Slowly increase the gain. When you see a tendency to oscillate as the axis moves or stops, reduce the gain by 10 to 30 percent.

  Tip:
  For a 2.5 in bore hydraulic cylinder with a max velocity of 30 in/sec, the Proportional Gain is typically on the order of 20 to 300. Start with a small value.

- Click the Download button to apply the changes to the RMC.

10. Adjust the Integral Gain

Many hydraulic systems do not require a large Integral Gain. However, it is usually desirable to have some Integral Gain to help compensate for valve null drift or changes in system dynamics. Some systems may require larger Integral Gain, in particular if they are moving a large mass or are nonlinear. Too much Integral Gain will cause oscillations and overshoot. The Integral Gain is helpful for getting into position and for tracking during long, slow moves. It will not significantly affect tracking during short, fast moves.

Tip:
Typically, the Integral Gain should be 3 to 5 times greater than the Proportional Gain.

- Click the **Download** button to apply the changes to the RMC.

Too little Integral Gain:

The system may not get into position very quickly, and it will take a long time to recover from quick changes in the system, for example static friction, load changes, or obstacles.

Too much Integral Gain:

The system will begin oscillating.

Correct Integral Gain:

The system will get into position quickly, and any system changes will be quickly corrected.

11. Adjust the Differential Gain

Differential Gain may greatly enhance performance on many hydraulic systems. It is used mainly on systems that have a tendency to oscillate. This happens when heavy loads are moved with relatively small cylinders. Differential Gain will tend to dampen out oscillations and help the axis track during acceleration and deceleration. This will positively affect short, fast moves.

A disadvantage of Differential Gain is that it amplifies position measurement noise. If there is too much noise or the gain is too high, this can cause the system to chatter or oscillate. However, this can be compensated for by using the Output Filter. Systems that are difficult to tune can sometimes be drastically improved by using the Output Filter together with the Differential gain.

Many systems do not require any Differential Gain.

- Start with a small value of Differential Gain. Remember that it is a floating point number, and you will may have to start with a number much smaller than 1, depending on your system.

  Tip:
  For a 2.5 in bore hydraulic cylinder with a max velocity of 30 in/sec, the Differential Gain, if any, is typically on the order of 0.01 to 2. Start with a small value.

- Increase the Differential Gain. It may help the system track better. If it starts oscillating or chattering, decrease the gain.
- If the drive output during the constant velocity portion of the move is smooth, the Differential Gain is perhaps not set high enough.
- When the Differential Gain is properly adjusted, the drive output may look "fuzzy." This indicates that the drive is responding to the minute errors of the axis. Not all systems allow the differential gain to be set high enough for the drive to be "fuzzy".

  Note:
  If you use Differential Gain, you may be able to increase the Proportional Gain somewhat without making the system oscillate.

- If the system starts chattering or oscillating before it can achieve proper control, try using the Output Filter.
- Click the Download button  to apply the changes to the RMC.

Too little Differential Gain:

The system will not keep up during acceleration and deceleration.

Too much Differential Gain:

The system will exhibit rapid oscillation.

Correct Differential Gain:

The system will track properly during acceleration and deceleration.

12. Adjust the Output Filter

If the axis is not tracking very well at this point, the Output Filter may help. It can significantly improve control of difficult systems. Without the Output Filter, the Differential Gain can cause the Control Output to oscillate, causing oscillation of the axis. By using the output filter, the Differential Gain can be increased significantly to help the Actual Velocity track the Target Velocity.

Typically, the Output Filter can be set to a value close to the natural frequency of the system. For example, if a system tends to oscillate at 10 Hz, a good starting value for the Output Filter is 10 or higher. After setting the Output Filter, you may be able to increase the Differential and Proportional Gains to improve control. If it does not help, try smaller and higher values of Output Filter, then try increasing the Differential and Proportional Gains again.

Keep in mind that the Output Filter does not always help.

13. Increase System Speed

If you have not yet done so, increase the Speed and Acceleration values of the moves. Look for following errors, overshoot, or oscillations.

- If an Output Saturated error occurs, there is not enough drive capacity to drive the axis at the requested Speed or Acceleration. Should this occur, reduce the Speed and/or Acceleration and Deceleration.
- If a Following Error occurs during acceleration and deceleration and adjusting the Gains and Acceleration Feed Forward does not help, the Acceleration and Deceleration ramps are too steep for the response of the system.
- If the actual position lags or leads the target position during the entire constant velocity section of the move, adjust the Feed Forwards.
- Should the system seem a little sloppy, try increasing the Proportional Gain.
- If the Control Output never gets very high, the gains can probably be increased for better control. If the Control Output is too high, or an overdrive error occurs, the system is not capable of performing the requested move. The Speed, and/or Accelerations may need to be decreased.

- If the system vibrates while in position, the gains may be too high, or the Dead Band value may need to be increased. However, if the oscillation is not caused by a deadband in the system, adjusting the Dead Band value will not help! A rule of thumb is to set the Dead Band Eliminator value to half of the peak-to-peak oscillation of the drive output while in position.
- The final tuning of the system should be made at the speed of intended operation.

14. Save your Settings to Flash

To retain your settings in the RMC in the event of a power loss, you must save the settings to Flash:

- On the **Controller** menu, click **Update Flash**.

2.3.7. Tuning a Pressure/Force System

The following procedure may be used to tune a pressure-only or force-only axis. For details on setting up a pressure-only or force-only axis, see Controlling Only Pressure or Force.

If you are tuning a position-pressure or position-force axis, see the Tuning a Position-Pressure or Position-Force System topic.

Please read the Tuning Overview topic before performing the tuning procedure.

There is no substitute for experience when tuning an axis. This procedure offers some guidelines, tips, and suggestions for tuning your system. While these steps will work for some systems, they may not be the best for a particular system.

Position/Pressure Tuning Procedure

Pre-Tuning Steps:

These steps come before actually tuning the gains.

1. Set the Pressure/Force Limits

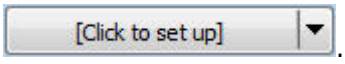
Set the Positive Pressure/Force Limit and Negative Pressure/Force Limit. These specify the pressure/force range in which the axis will be allowed to control. If the Target Pressure/Force exceeds these values, an error bit will be set and will halt the axis if the AutoStops are set to do so.

2. Set the AutoStops

Set the Pressure/Force Following Error AutoStop to Status Only to prevent following errors from halting the axis during the tuning. After the tuning is complete, set the AutoStops to values that are safe for machine operation.

3. Set Up the Tuning Tools

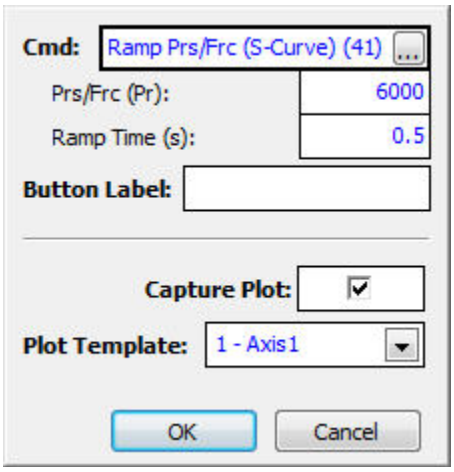
In the Tuning Tools, on the **Pressure** or **Force** tab, set up the command buttons so that one button will ramp the pressure or force in one direction, and the other button will ramp it in the other direction.

a. Click the down arrow on the command button 

.

b. Enter the **Pressure/Force**, and **Ramp Time**, then click **OK**.

Example:



c. Repeat the previous step for the other command button, and enter a different value for the **Pressure/Force**.

Tuning Steps:

1. Enter Pressure/Force Control

Set the Pressure/Force Proportional Gain to a small value, such as 0.01.

In the Tuning Tools, on the **Pressure/Force** tab, click the **Hold Prs/Frc** button to enter pressure/force control. Then click one of the buttons you set up to ramp the Command Pressure to the desired value.

You may wish to move the axis in open loop to where it encounters pressure/force before you click the **Hold Prs/Frc** button.

If the axis halts due to an Output Saturated error, reduce the Proportional Gain and repeat.

2. Adjust the Proportional Gain

Start with a small value. If the application is a hydraulic cylinder controlling force in pounds, a typical starting value may be 0.01 or smaller.

In the Tuning Tools, on the **Pressure/Force** tab, notice that the **Target Pressure/Force** and **Actual Pressure/Force** are displayed.

Keep increasing the Proportional Gain until the Actual Pressure/Force starts moving toward the Target Pressure/Force. It should not get all the way toward to Target Pressure/Force. Give just enough gain to make sure the Actual begins approaching the Target.

3. Adjust the Integral Gain

Add some Pressure/Force Integral gain, approximately twice as much as the Pressure/Force Proportional Gain. The Actual Pressure/Force should go towards the Target Pressure/Force.

Do not give a lot of Integral Gain, just enough to make sure it eventually gets to the Target Pressure/Force.

4. Ramp the Pressure/Force Up and Down

For the remainder of the tuning (steps 5, 6 and 7), repeat the following steps for each gain parameter you change:

1. Ramp the pressure/force with one of the buttons you set up in the Tuning Tools.
2. View the plot.

3. Change the gain.

Tip: You can use the Tuning Wizard to calculate a model and use the Gain Calculator to choose gains. After uploading a plot of pressure or force control, where the axis is controlling a changing pressure or force, click the **Tuning Wizard**.
If you use the Tuning Wizard, you do not need to continue manually changing the gains.

5. Adjust the Feed Forwards

Pressure/Force Rate Feed Forward

Ramp the pressure/force, and adjust the Pressure/Force Rate Feed Forward. Start with a small value. The Pressure/Force Rate Feed Forward should help the Actual Pressure/Force track the Target Pressure/Force during the ramp. Your system may not need any Pressure/Force Rate Feed Forward.

Pressure/Force Feed Forward

Most systems do not require the Pressure/Force Feed Forward. If your system exerts a pressure or force that is roughly linear to the amount of Control Output, you may need the Pressure/Force Feed Forward. To tune the Pressure/Force Feed Forward, move the Target Pressure/Force to various values, and adjust the Pressure/Force Feed Forward while the Target Pressure/Force is stationary. Do this for several value of pressure/force to ensure the best Pressure/Force Feed Forward.

6. Re-adjust the Proportional and Integral

Ramp the pressure/force, and re-adjust the proportional and integral gains. This should make it track better.

7. Adjust the Differential Gain

If you need better control, add some Differential Gain. The Differential Gain is typically a couple orders of magnitude less than the Proportional Gain.

8. Fine-tune the System

The final tuning of the system should be made at the rate and pressure/force range of intended operation. Look for following errors, overshoot, or oscillations, and consider the following:

- Should the system seem a little sloppy, try adjusting the Proportional Gain.
- If the actual pressure lags or leads the target position during the entire constant velocity section of the move, adjust the Feed Forwards.
- If an Output Saturated error occurs, there is not enough drive capacity to drive the axis at the requested rate of pressure/force change. Should this occur, increase the Ramp Time or decrease the speed of the system.
- If the Control Output is not high, the gains can probably be increased for better control, depending on system stability.
- Adding or changing the pressure/force feedback filter may help if noisy feedback is suspected.

2.3.8. Tuning a Position-Pressure or Position-Force System

The following procedure may be used to tune a system that uses pressure/force control or pressure/force limit. This procedure uses pressure/force control, but once an axis is tuned for pressure/force control, those gains can be used for pressure/force limit with few or no changes.

Please read the following topics before performing the tuning procedure:

- Tuning Overview
- Tuning a Position Axis

There is no substitute for experience when tuning an axis. This procedure offers some guidelines, tips, and suggestions for tuning your system. While these steps will work for some systems, they may not be the best for a particular system.

Position/Pressure Tuning Procedure

Pre-Tuning Steps:

These steps come before actually tuning the gains.

1. Set the Pressure/Force Limits

Set the Positive Pressure/Force Limit and Negative Pressure/Force Limit. These specify the pressure/force range in which the axis will be allowed to control. If the Target Pressure/Force exceeds these values, an error bit will be set and will halt the axis if the AutoStops are set to do so.

2. Set the AutoStops

Set the Position Following Error and Pressure/Force Following Error AutoStop to Status Only to prevent following errors from halting the axis during the tuning. After the tuning is complete, set the AutoStop to a safe setting for machine operation.

Tuning Steps:

1. Tune the Position Gains

The position gains should be tuned before attempting to tune the axis for pressure/force. Obtaining control of the axis' position greatly simplifies the tuning of the pressure/force gains. If you have not yet tuned the position gains, follow the procedure outlined in the Tuning a Position Axis topic before continuing.

2. Enter Pressure/Force Control

Set the Pressure/Force Proportional Gain to a small value, such as 0.01.

You may wish to move the axis to where it encounters pressure/force before you issue the Hold Current Pressure/Force (19) command.

Then enter pressure/force control with the Hold Current Pressure/Force (19) command. Issue the Ramp Pressure/Force (S-Curve) (41) or Ramp Pressure/Force (Linear) (42) or command to set the Command Pressure to the desired value.

If the axis halts due to an Output Saturated error, reduce the Proportional Gain and repeat.

3. Adjust the Proportional Gain

Keep increasing the Pressure/Force Proportional Gain until the Actual Pressure/Force starts moving toward the Target Pressure/Force. It should not get all the way toward to Target Pressure/Force. Give just enough gain to make sure it is beginning to.

4. Adjust the Integral Gain

Add some Pressure/Force Integral gain, approximately twice as much as the Pressure/Force Proportional Gain. The Actual Pressure/Force should go towards the Target Pressure/Force.

Do not give a lot of Integral Gain, just enough to make sure it eventually gets to the Target Pressure/Force.

5. Ramp the Pressure/Force Up and Down

For the remainder of the tuning, repeat the following steps:

1. Ramp the pressure/force.
   This can be done with the Ramp Pressure/Force (Linear) (42) or Ramp Pressure/Force (S-Curve) (41) command.
2. Change a gain.
3. View the plot.

Tip: You can use the Tuning Wizard to calculate a model and use the Gain Calculator to choose gains. After uploading a plot of pressure or force control, where the axis is controlling a changing pressure or force, click the **Tuning Wizard**.

The Tuning Tools make this very easy. In the Tuning Tools, set up the command buttons so that one button will ramp the pressure/force in one direction, and the other will ramp it in the other direction.

- In the Tuning Tools, click one of the buttons labeled **Click to set up**.
- Choose the Ramp Pressure/Force (Linear) (42) or Ramp Pressure/Force (S-Curve) (41) command and enter valid command parameters for your axis.
- Repeat the previous step on the other command button. Enter the same command, except for a different **Requested Pressure/Force**.
- Click **OK**. Now you will see that the command buttons are labeled.

To ramp the pressure/force, just click one of the buttons, and the plot will automatically be uploaded.

6. Adjust the Feed Forwards

Pressure/Force Rate Feed Forward

Ramp the pressure/force, and adjust the Pressure/Force Rate Feed Forward. Start with a small value. The Pressure/Force Rate Feed Forward should help the Actual Pressure/Force track the Target Pressure/Force during the ramp.

Pressure/Force Feed Forward

Most systems do not require the Pressure/Force Feed Forward. If your system exerts a pressure or force that is roughly linear to the amount of Control Output, you may need the Pressure/Force Feed Forward. To tune the Pressure/Force Feed Forward, move the Target Pressure/Force to various values, and adjust the Pressure/Force Feed Forward while the Target Pressure/Force is stationary. Do this for several value of pressure/force to ensure the best Pressure/Force Feed Forward.

7. Re-adjust the Proportional and Integral

Ramp the pressure/force, and re-adjust the proportional and integral gains. This should make it track better.

8. Adjust the Differential Gain

If you need better control, add some Differential Gain. The Differential Gain is typically a couple orders of magnitude less than the Proportional Gain.

9. Tune the Transition

Now that both position and pressure/force are tuned, you can tune the transition between position and pressure/force control or pressure/force limit.

Issue the commands for transitioning between position and pressure/force as you intend to do during normal machine operation. For some applications, this may involve creating a simple user program.

View the transition on a plot. If the pressure/force drops off at the transition, increase the Integrator Preload parameter in the Enter Pressure/Force command.

10. Fine-tune the System

The final tuning of the system should be made at the rate and pressure/force range of intended operation. Look for following errors, overshoot, or oscillations, and consider the following:

- Should the system seem a little sloppy, try adjusting the Proportional Gain.
- If the actual pressure lags or leads the target position during the entire constant velocity section of the move, adjust the Feed Forwards.
- If an Output Saturated error occurs, there is not enough drive capacity to drive the axis at the requested rate of pressure/force change. Should this occur, increase the Ramp Time or decrease the speed of the system.
- If the Control Output is not high, the gains can probably be increased for better control, depending on system stability.
- Adding or changing the pressure/force feedback filter may help if noisy feedback is suspected.

2.3.9. Tuning a Motor in Torque Mode

When tuning a motor in torque mode, use the basic instructions in the Tuning a Position Axis topic, but use the order of the gains below.

Tuning Order

Torque motors generally do not have much damping. Damping must be provided for the system, or it will be difficult to control. Providing some Differential Gain will effectively dampen the system. Therefore, the Differential Gain must be set early in the tuning process:

Method 1:

a. Set all the gains to zero.
b. Increase the Proportional Gain to the point where the motor moves somewhat when given a Move Absolute (20) command. You should notice that the axis has a tendency to overshoot.
c. Increase the Differential Gain so that the axis ceases overshooting. If the axis begins oscillating or chattering, decrease the Differential Gain.
d. Continue tuning the gains in the order given in the Tuning a Position Axis topic.

Method 2:

a. Set all the gains to zero.
b. Issue the Hold Current Position (5) command to the axis.
c. Increase the Differential Gain until the motor start humming (or chattering or oscillating), then decrease the Differential Gain significantly, perhaps up to 50%, to avoid oscillation later when making moves.
d. Continue tuning the gains in the order given in the Tuning a Position Axis topic.

Small Motors

For very small motors that can be rotated by hand, the Differential gain can be set before any other gains.

a. Set all the gains to zero.
b. Set the Differential Gain to a small value, then issue the Hold Current Position (5) command to put the axis in closed loop control.
c. Rotate the motor manually to get a feel for the resistance to movement (damping). Increase the Differential Gain until the damping is significant. If the motor chatters or oscillates, decrease the gain.
d. Continue tuning the gains in the order given in the Tuning a Position Axis topic.

2.3.10. Tuning a Pneumatic System

When tuning a pneumatic cylinder, use the basic instructions in the Tuning a Position Axis topic, but use the order of the gains below.

Tuning Order

When using high-order gains, increasing the highest order will allow you to increase lower order gains. However, you need to first set the lower gains to some value before you can set the higher ones. Then, you can go back and increase the lower order gains.

The Output Filter can significantly improve control of pneumatics. Without the Output Filter, the Differential Gain can cause the Control Output to oscillate, causing oscillation of the axis. By using the output filter, the differential gain can be increased significantly to help the Actual Velocity track the Target Velocity.

Suggested Tuning Order:

a. Proportional Gain
b. Output Filter
c. Differential Gain
d. Double Differential Gain
e. Triple Differential Gain
f. Double Differential Gain
g. Differential Gain
h. Proportional Gain
i. Integral Gain
j. Repeat several times.

Feed Forwards are not useful on poor systems. They may be more significant on low-friction cylinders. If your application requires Feed Forwards, set them before setting the Integral Gain.

The P, D, DD and TD gains typically differ from each other by an order of magnitude. For example, the ratio of the gains to each other may be as shown below. Keep in mind that the actual value of gains required on your system may differ significantly.

P: 10

I: 10

D: 0.1 or smaller

DD: 0.01 or smaller

TD: 0.001 or smaller

Other tips:

- Use the Output Filter, setting it to a low value, comparable to the natural frequency of the system.
- Deadband may be required on systems with high static friction.
- The Velocity and Acceleration may need to be filtered, or using the model may help.

2.3.11. Tuning Active Damping and Acceleration Control

When tuning Active Damping or Acceleration Control, use the basic instructions in the Tuning a Position Axis topic, but use the order of the gains below.

If you are tuning a pneumatic cylinder, see the Tuning a Pneumatic System topic.

Tuning Order

Systems with High Damping

For system with high damping, you can follow the procedure for Tuning a Position Axis, then set the higher-order gains.

High damping means that the system does not oscillate and stops very quickly as soon as 0 volts is applied.

Systems with Low Damping

For system with low damping, you will need to set the higher-order gains earlier in the tuning procedure to gain any control at all. The higher order gains will provide the damping. When using high-order gains, increasing the highest order may allow you to increase lower order gains. However, you need to first set the lower gains to some value before you can set the higher ones. Then, you can go back and increase the lower order gains.

Suggested Tuning Order:

a. Proportional Gain
b. Differential Gain
c. Double Differential Gain or Active Damping Proportional Gain
d. Triple Differential Gain or Active Damping Differential Gain
e. Double Differential Gain or Active Damping Proportional Gain
f. Differential Gain
g. Proportional Gain
h. Feed Forwards
i. Integral Gain
j. May need to repeat several times.

3. Controller Features

3.1. RMC Controller Features

The RMC75, RMC150, and RMC200 motion controllers provide a host of features to successfully control any motion application. Browse the sections below for descriptions and links to the many features.

Motion Types

- Point-to-Point (Absolute and Relative)
- Open Loop
- Quick Move
- Gearing
- Velocity
- Sinusoidal
- Curves (Cams, Splines)

Control

Types

- Open Loop Control
- Closed Loop Control
- Position PID
- Position I-PD
- Velocity PID
- Velocity I-PD
- Cascade Control
- Unidirectional Mode
- Pressure/Force Control
- Pressure/Force Limit

Controlled Quantities

- Position
- Velocity
- Pressure
- Force
- Torque

Dual-Loop Control

- Position-Pressure
- Position-Force
- Velocity-Pressure
- Velocity-Force

Communications (Slave Only)

- Ethernet
- PROFIBUS
- Serial RS-232/485 (RMC75)
- Address Maps

Transducer Interfaces

- Magnetostrictive Start/Stop and PWM
- SSI
- Analog (±10V, 4-20mA)
- Load Cell
- Quadrature
- Resolver

Custom Feedback

- Custom Feedback
- Switching Feedback on the Fly
- Feedback Linearization
- Redundant Feedback

Axis Types

- Control Axes
- Reference Axes (Half-Axes)
- Rotary Axes

Motion Safety Features

- Halts
- Auto Stops
- Enable Output
- Fault Input

Tuning

- Tuning Tools
- Autotuning
- Tuning Wizard
- Gain Calculator
- Gain Scheduling

Programming

- Pre-programmed Commands
- User Program Overview
- Variables
- Retentive Variables (RMC75E and RMC150E only)
- Program Triggers

Diagnostic Tools

- Plots and XY Plots
- Event Log
- Status Registers

Discrete I/O

- Discrete I/O

Position-Acceleration

Velocity-Acceleration

Control Output

RMC75/150: ±10 V Servo Output

RMC200: ±10 V, 4-20 mA, or ±20 mA Servo Output

High-Order

Active Damping

Acceleration Control

High-Speed Functions

Homing

Registration

3.2. General

3.2.1. Firmware

Firmware is the software that resides in the controller. Each RMC is shipped with the latest version of firmware. Delta regularly releases new versions of firmware that include bug fixes and support for new features.

Each command topic in the RMCTools help lists the firmware version required for the command. Other topics may also specify which firmware versions support the feature.

Updating Firmware

To update the firmware, the RMC must be connected to the computer running RMCTools via USB, Ethernet, or the RS-232 Monitor Port. The second serial port on the RMC75S will not work for updating firmware. To update firmware via Ethernet, the **Allow updating firmware over Ethernet** box must be checked in the Ethernet Settings Page.

To update the firmware in the RMC:

1. Go to Delta's website at www.deltamotion.com.
2. On the **Downloads** page, find the firmware for your module, and save it on your computer.
3. In RMCTools, go online with your controller.
4. In the Project pane, expand the **Modules** folder, double-click the module you wish to update and click **Firmware**. For the main firmware, the module will typically be the CPU.
5. In the **Control Program** section, click **Update Firmware**. This will open the Firmware Update Wizard. Follow the instructions to complete the firmware update.

Backing Up Firmware (RMC75/150 Only)

As part of the firmware update process, the firmware in the RMC75 and RMC150 can be backed up before the new firmware is downloaded to the RMC. Delta recommends backing up the firmware in case you need to revert to that firmware version. To back up the firmware, use the Firmware Update Wizard as described above. Before the download occurs, you will be given the option of backing up the firmware.

Firmware Base Version

Each firmware release made for the RMC is identified primarily by a two- or three-part version number such as ”3.30.0”. Each firmware release may include more than one actual firmware image, with each image having a separate Configuration ID. When a

controller is loaded with firmware, RMCTools chooses the appropriate firmware image to load into the controller.

Special Releases

In some cases, Delta will release firmware that differs in some way from the generally-released firmware. These are designated by an additional *special release code*. For example, the ”S4” special firmware has ”S4” append to the version number, as in ”2.72A-S4”. For more information on the S4 special firmware in particular, see the **Firmware Compatibility** section below.

RMC75 Firmware and Hardware Compatibility

The RMC75 firmware has the following hardware compatibility limitations:

Version	RMC75S	RMC75P	RMC75E	Notes
3.00 and above	2.1D or newer	2.1E or newer	✓	See Note 1
2.72-S4 to 2.99-S4	2.1C or older	2.1D or older	-	See Note 2
2.30 to 2.71	✓	✓	✓	See Note 3
1.00 to 2.21	✓	✓	✓	

Note 1: These firmware releases no longer support older RMC75S and RMC75P controllers, which cannot support new features.

Note 2: These firmware releases are reserved for patch releases for the older RMC75S and RMC75P controllers, which cannot support new features and are therefore not included in the 3.00 and later releases. This firmware is assigned the ”S4” special release code.

Note 3: These firmware releases support all RMC75 controllers, however, new commands added by these releases are not included in firmware for older RMC75S and RMC75P controllers.

3.2.2. Loop Time

The RMC is a deterministic controller. It reads the inputs, computes the control algorithms and updates the outputs at a specific interval. This interval is called the controller **loop time**, in reference to the way the controller repeatedly "loops" through its code.

The RMC will always run at its loop time setting. When it finishes all the calculations for one loop, it waits until the next loop time before doing its calculations again.

Setting the Loop Time

To set the loop time of the RMC, use the Control Loop Time Settings Page.

Available Loop Times vs Axes

The loop times available on the RMC controllers are listed below, along with the expected number of axes supported by that loop time with a full load of motion commands, user programs, communications and plots. The user program time allocation is also affected by loop time, as described in Program Capacity and Time Usage.

High Control Loop Utilization

If you wish to use a loop time with more axes than listed below as being supported with a full load of motion commands, user programs, communications and plots, then you must enable **High Control Loop Utilization** in the Control Loop Time Settings Page of the controller properties dialog.

The **High Control Loop Utilization** option does the following:

- Allows selecting faster loop times given the number of control axes defined in the controller.
- Allows downloading user programming with estimated worst-case processing times that exceed the allocated time. See Program Capacity and Time Usage.

If you choose **High Control Loop Utilization**, then you must ensure that your application will run within the selected control loop time. See the **Exceeding the Loop Time** and **Monitoring the Loop Time** sections below.

RMC75 Loop Times

Loop times that support all axes configurations are indicated by a check mark ✓.

	Loop Time				
	4000 µs	**2000 µs**	**1000 µs**	**500 µs**	**250 µs**
RMC75E	✓	✓	✓	✓	1 axis only
RMC75S	✓	✓	✓	1 axis only with up to one reference input	—
RMC75P (2.1F or newer)	✓	✓	✓	1 axis only with up to one reference input	—
RMC75P (2.1E or older)	✓	✓	1 axis only with up to one reference input	—	—

RM150E Loop Times

Loop times that support all axes configurations are indicated by a check mark ✓.

	Loop Time				
	4000 µs	**2000 µs**	**1000 µs**	**500 µs**	**250 µs**
RMC150E	✓[1]	✓	✓	1 or 2 axes only[2]	1 axis only[3]

[1] 4000 µs requires firmware 3.33.0 or newer or 3.36.0 if MDT modules are present. It is not supported by RMC150E/RMC151E CPU revision 1.2B or older or MDT (-M) modules revision 6.0 or older.

[2] 500 µs is not supported by SSI (-S) modules revision 5.0 or older, or Analog Input (-A, -H, and -G) modules revision 6.0 or older.

[3] 250 µs is not supported by SSI (-S) modules revision 5.0 or older, or any Analog Input (-A, -H, and -G) modules.

RMC200 Loop Times

In the tables below:

- **All** refers to the maximum number of axes supported by the controller, as listed in the Axis Types Overview.
- **Full Load** includes control, user programs, plots and communications.
- **Limited Load** indicates the range of axes possible in applications using less than a full load. Using axes in this range requires **High Control Loop Utilization** to be enabled.
- A **control axis** in this context refers to an axis with at least one PID loop, and includes cascading outer loop axes.

CPU20L

	Expected Number of Axes Supported with:	
Loop Time	**Full Load**	**Limited Load**

125 µs	0 axes	Up to 3 control axes (or 2 dual-loop)
250 µs	0 axes	Up to 7 control axes (or 5 dual-loop)
500 µs	Up to 11 control axes (or 8 dual-loop)	All
1000 µs	All	All
2000 µs	All	All
4000 µs	All	All

CPU40

	Expected Number of Axes Supported with:	
Loop Time	**Full Load**	**Limited Load**
125 µs	0 axes	Up to 6 control axes (or 4 dual-loop)
250 µs	0 axes	Up to 11 control axes (or 8 dual-loop)
500 µs	Up to 14 control axes (or 10 dual-loop)	Up to 32 control axes (or 23 dual-loop)
1000 µs	Up to 50 control axes (or 36 dual-loop)	All
2000 µs	All	All
4000 µs	All	All

When is High Control Loop Utilization Required?

For the RMC200 controller, **High Control Loop Utilization** is required if the estimated axis processing time exceeds the processing time allowed for the selected loop time. The Axis Definitions dialog automatically provides a graphic indication of the estimated axis processing time and whether **High Control Loop Utilization** is required. The following tables are used to determine whether or not High Control Loop Utilization is required:

Maximum Total Axis Processing Time Allowed at Full Load

Loop Time	**CPU20L**	**CPU40**
125 µs	0 µs*	52 µs
250 µs	0 µs*	100 µs
500 µs	152 µs	302 µs
1000 µs	302 µs	not limited
2000 µs	not limited	not limited
4000 µs	not limited	not limited

*This means **High Control Loop Utilization** is always required.

Estimated Processing Time per Axis

Axis Type	**CPU20L**	**CPU40**
Single-loop Control Axis	11.9 µs	10.2 µs
Dual-loop Control Axis	16.0 µs	14.0 µs
Virtual Axis	7.9 µs	6.3 µs
Reference Axis	3.2 µs	2.5 µs

Output Only Axis	1.5 µs	1.2 µs

Notice that the estimated per-axis processing times shown above are conservative estimates assuming maximum processing, for purposes of determining whether High Control Loop Utilization is required. Actual processing time for axes will often be significantly less than these times indicate. However, the user should always verify that the application does not exceed the loop time excessively, as described below.

Exceeding the Loop Time

How the loop time works

The loop time is the pre-selected interval at which the RMC reads its inputs, processes motion commands and user programs, computes the control algorithms and updates the outputs. When it finishes all the processing for one loop, it waits until the next loop time before doing its calculations again. The amount of calculations can vary significantly from

- **Plotting Loop Time Registers**
  To clearly visualize the loop time usage of the RMC, you can add the loop time usage registers to a plot. See Plotting the Loop Time for details.

Reducing the Loop Time Used

If the RMC is exceeding the loop time, you can attempt to reduce the loop time used, or select a larger loop time. To reduce the loop time used, you can plot the loop time registers as described in Plotting the Loop Time to determine which part of the loop is taking the most time, then use one or more of the following tips to reduce the loop time:

User Programs

- Reduce the programming loop time used by following the tips in the Program Capacity and Time Usage topic.

Commands:

- **Stagger simultaneous commands**
  When a command is received, the RMC performs some initial calculations in that loop time, which can be extensive. Sending many commands simultaneously can therefore result in a large loop time load. If it is possible to stagger the commands, the total loop time will not be as large.
- **Do not use the Transition commands**
  The Transition Rate (56) command causes extra processing, not when it is sent, but when a transition is active, such as during the beginning of sine or curve motions. Removing transitions may possibly reduce the loop time load.
- **Do not use Dynamic Retargeting**
  Dynamic Retargeting is the continuous sending of a Move Absolute (20) or similar commands, such as every loop time. This causes a heavy processing load.

Plots:

- **Reduce the number of captured plot items**
  Having a very large number of captured plot items may impact the loop time. By default, each axis has a corresponding plot that triggers when a command is sent to the axis. Reducing the number of plotted items, or changing the settings so that not as many plots are automatically triggered may slightly reduce the loop time load.

Communications:

- **Reduce the Cyclic I/O**
  A large amount of cyclic EtherNet/IP I/O or PROFINET communications at very short intervals can increase the loop time. Reducing the Requested Packet Interval, the number of connections, or the amount of data may reduce the loop time load.

Using the Loop Time in Expressions

Certain calculations in user programs, especially for calculating time, may require accessing the current set loop time. The **_LoopTime** tag can be used to access the loop time when using expressions. _LoopTime is a REAL, and is specified in seconds.

3.2.3. Plotting the Loop Time

To clearly visualize the loop time usage of the RMC, you can add the following Controller Loop Time Usage Registers to a plot to monitor the loop time:

Register Name
Units in seconds
Loop Time Used, Last
Loop Time Used, Maximum

Loop Time Used, Minimum
Units in microseconds
Loop Time Used, Total
Loop Time Used, Axes
Loop Time Used, Programs
Loop Time Used, Plots
Loop Time Used, Comm
Loop Time Used, Overhead

To add these registers to a plot:

1. In the Plot Template Editor, create a new plot template or choose an existing plot.
2. Choose **Custom Plot**.
3. In the **Plotted Data Items** table, click **New Quantity**.
4. On the **Registers** tab, browse to **Controller**, **Loop Time**, and choose the desired **Loop Time Used** registers.

When trending a plot, you can see how the loop time usage varies. When commands are issued, the loop time will typically increase significantly for that loop time. If the total loop time used occasionally exceeds the set loop time by a small amount, there will typically be no adverse effects. Small loop time overages will be reported in the Event Log as a warning, which can be filtered using the Event Log filter settings.

It is very important that a single control loop execution or the accumulated carryover from consecutive loops does not exceed the loop time by more than 100%. The RMC will report an error in the Event Log if the loop time overage or cumulative loop time overage exceeds 80%. For more details, see the Loop Time topic.

3.2.4. RUN/PROGRAM/Disabled Mode

RUN and PROGRAM mode specify whether the User Programs and the Program Triggers can run. RUN or PROGRAM mode do not necessarily affect the motion on an axis. Motion commands can be issued to the RMC whether or not it is in RUN or PROGRAM mode.

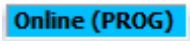
The RMC200 also includes a Disabled Mode, in which no motion commands are accepted, and all axes are disabled.

RUN mode - RMC75/150/200

- The Program Triggers are started (if the Enable the Program Triggers task checkbox is checked in the Programming Properties).
- User Programs can run.
- LEDs
  - The RMC75 Controller LED is steady green.
  - The RMC150 RUN Mode LED is steady green.
  - The RMC200 RUN LED is steady green.
- The RMCTools toolbar will display Online (RUN).

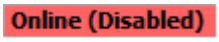
PROGRAM mode - RMC75/150/200

- The Program Triggers are stopped.
- User Programs cannot run.
- LEDs

  - The RMC75 Controller LED slowly flashes green.
  - The RMC150 RUN Mode LED is off.
  - The RMC200 RUN LED is off.
- The RMCTools toolbar will display 

.

Disabled mode - 200 only

- All axes are disabled, and no motion commands are accepted.
- The Program Triggers are stopped.
- User Programs cannot run.
- The CPU Run and En LEDs are off.
- The RMCTools toolbar will display 

.

Entering and RUN, PROGRAM and Disabled Modes

When the RMC enters RUN Mode:

- The controller will be enabled if it wasn't previously. This is indicated by the Enabled bit in the Controller Status register.
- **RMC75/150:** All the axes become enabled if they were not previously enabled. This means that the Enabled status bit turns on for each axis, exactly as if the Enable Controller (7) command had been issued.
- **RMC200:** If the **Auto-Enable Axes** option is checked on the RUN/Disabled page of the Programming Properties, then all the axes become enabled, and all Enable Outputs are turned on.
- All the Program Triggers are now evaluated. Notice that all the conditions are assumed to be false when the RMC is in PROGRAM mode. Therefore, if any condition is true when the RMC enters RUN Mode, it will start the specified actions.

Entering RUN mode will not affect the motion on the axes. The axes will continue doing what they were doing.

When the RMC enters PROGRAM Mode:

- The Program Triggers are stopped.
- The Tasks are stopped. All User Programs stop.
- **RMC200:** The controller will be enabled if it wasn't previously. This is indicated by the Enabled bit in the Controller Status register. If the **Auto-Enable Axes** option is checked on the RUN/Disabled page of the Programming Properties, then all the axes become enabled, and all Enable Outputs are turned on.
- If any discrete outputs have been configured to turn on or off when entering PROGRAM mode, they will do so.

Entering PROGRAM mode will not affect the motion on the axes. The axes will continue doing what they were doing.

When the RMC enters Disabled Mode:

- All axes will halt with a Direct Output Halt.
- The Controller Enabled Bit is cleared.
- The Program Triggers are stopped.
- The Tasks are stopped. All User Programs stop.
- If any discrete outputs have been configured to turn on or off when entering PROGRAM mode, they will do so.

Entering PROGRAM mode will not affect the motion on the axes. The axes will continue doing what they were doing.

Fault Controller

RMC75/150: If the Fault Controller (8) command is sent, the RMC will exit RUN mode and enter PROGRAM mode. All the axes will halt.

RMC200: If the Fault Controller (8) command is sent, the RMC will enter Disabled mode. All the axes will halt.

Methods of Entering RUN, PROGRAM, or Disabled Mode

From RMCTools

- On the toolbar, click the **Controller** button , , or  and choose the desired mode.

From a Host Controller (PLC, HMI, etc.)

- RMC75/150: Send the RUN Mode (98) or PROGRAM Mode (99) commands to the RMC.
- RMC200: Send the RUN Mode (98), PROGRAM Mode (99), or Fault Controller (8) commands to the RMC.

Using a Discrete Input to enter RUN, PROGRAM, or Disabled mode

- In the Project Pane, in the project tree, right-click the **Programming** node, click **Properties**, and click the **RUN/PROGRAM** or **RUN/Disabled** page.
- Select **Define a RUN/PROGRAM discrete input** or **Define a RUN/Disabled discrete input**
- Choose the desired input and click **OK**.

  RMC75/150: When the input transitions from low to high, the RMC will enter RUN mode. When the input transitions from high to low, the RMC will enter PROGRAM mode. The state of the input does not necessarily reflect the RUN/PROGRAM state, because the RUN/PROGRAM mode can also be changed by other methods.

  RMC200: When the input transitions from low to high, the RMC will enter RUN mode. When the input is low, the RMC will be in the Disabled state. If the input is high, the RMC may not necessarily be in the RUN state, because RUN mode can be exited by other methods.

Starting the RMC in RUN, PROGRAM, or Fault mode

By default, the RMC75 and RMC150 start up in PROGRAM mode, and the RMC200 starts up in Disabled mode. You can set the startup mode:

- In the Project Pane, in the project tree, right-click the **Programming** node and click **Properties**.
- On the **RUN/PROGRAM** or **RUN/Disabled** page, choose the **Startup Mode** you want, then click **OK**.
- Right-click the **Programming** node and click **Download Programs**.
- Update Flash so that the settings will be saved when the RMC is powered off.

Tip:
You can start a User Program immediately when the RMC enters RUN Mode by using the FirstScan bit in the Program Triggers. If the Startup Mode is set to RUN Mode, the RMC can run a user program immediately upon powering up.

3.2.5. Registration

Registration is the process of recording the precise position of an axis when an external event occurs. Registration is commonly used for measurements. For example, consider a photo-eye set up on a conveyor belt, where the position of the belt is measured with a quadrature encoder. By registering the precise position at which the photo-eye is broken and the precise

position at which it is reconnected, the length of a widget on the belt can be accurately determined.

The RMC provides registration on quadrature feedback axes only. This is because quadrature feedback is the only type of feedback that can record a position at any given moment, independent of the loop time of the RMC. All other feedback types can only report the position once each loop time (500µs to 4000µsec). Using the programming capabilities of the RMC, such as discrete inputs and the Program Triggers, it is possible to achieve functionality similar to registration with other feedback types, but the accuracy will not be the same as with quadrature feedback.

Registration Inputs

Each quadrature feedback axis can keep track of two independent registration events, called **Registration 0** and **Registration 1**. These registration events can come from a single registration input, or from separate inputs. The table below lists the inputs on which registration can be performed.

<table>
<tr><th>Module</th><th>Registration Sources</th></tr>
<tr><td>RMC75 QAx or
RMC150 Quad</td><td>RegX/PosLim input
RegY/NegLim input</td></tr>
<tr><td>RMC150 UI/O</td><td>DI/O inputs R0 and R1</td></tr>
<tr><td>RMC75 Q1</td><td>Reg input</td></tr>
<tr><td>RMC200 Q4</td><td>Depending on the quadrature input:
<table>
<tr><th>Input</th><th>Registration Sources Inputs</th></tr>
<tr><td>Quadrature input 0 and/or 1</td><td>Reg0, Reg1, Hm0, Hm1</td></tr>
<tr><td>Quadrature input 2 and/or 3</td><td>Reg2, Reg3, Hm2, Hm3</td></tr>
</table></td></tr>
<tr><td>RMC200 U14</td><td>Reg/Z0, Reg/Z1, D0, D1</td></tr>
<tr><td>RMC200 D24</td><td>Depending on the quadrature option:
<table>
<tr><th>Option</th><th>Registration Sources Inputs</th></tr>
<tr><td>One quadrature input (A,B,Z)</td><td>D22,D23,D18,D19</td></tr>
<tr><td>One quadrature input (A,A,B,B) with wire break detection</td><td>D22,D23,D18,D19</td></tr>
<tr><td>Two quadrature inputs, using inputs D20&D21</td><td>D18, D19</td></tr>
<tr><td>Two quadrature inputs, using inputs D22&D23</td><td>D16,D17,D18,D19</td></tr>
</table></td></tr>
</table>

Arming the Registration

Issuing the Arm Registration (52) command tells the axis to wait for a registration event. Once this command is issued, the registration is said to be **armed**. When the registration is armed, the corresponding registration armed status bit will be set (Registration 0 Armed and Registration 1 Armed). Unless the registration is armed, the registration trigger is ignored.

The Arm Registration (52) command also tells the axis which Registration Number (0 or 1) to apply the registration event to, which input the registration will be triggered from, and which edge of the input (rising or falling) will trigger the registration.

Registration Event

If the registration is armed and a registration trigger occurs, the following is done:

- The precise position at the time of the registration is **latched**. The position will be reported in the Registration 0 Position or Registration 1 Position status register.
- The corresponding latched status bit will be set (Registration 0 Latched or Registration 1 Latched).
- The corresponding armed status bit will be cleared (Registration 0 Armed or Registration 1 Armed).

Once a registration is latched, you must issue the Arm Registration (52) command again to rearm it.

Disarm Registration

To manually disarm a registration, issue the Disarm Registration (53) command. Issuing this command will clear the corresponding Registration Armed status bit (Registration 0 Armed and Registration 1 Armed).

3.2.6. Homing

Certain encoders, such as quadrature encoders, are incremental and do not provide absolute position. To use an incremental encoder in a position application, a known reference position must be established. This position is called the **Home** position. The process of determining the home position is called **Homing**. As the encoder is rotated, it increments or decrements the position from the initial home position, and thereby the position is obtained.

When an axis is homed, the Actual Position of the axis is changed. The Target and Command positions are also adjusted by the same amount. Therefore, the motion of the axis will not be affected. It is safe to home an axis when it is moving.

The RMC offers homing of quadrature, incremental SSI, and incremental Resolver axes. All are described in this topic.

How to Home an Axis

This section describes the basic steps for homing a quadrature axis using the Home input and/or the Z (Index) pulse as the home trigger. The RMC has many options for homing; for more details, see the rest of this topic.

Note: If you are using the Z pulse, it is important to specify the Z Alignment before homing. See the **Using the Z Trigger** section below.

1. **Arm the Home**

   In order to perform homing, the home must first be *armed*. This means that the RMC will be looking for the specified trigger to occur, and when it occurs, will set the Actual Position to the specified position.

   To arm the home, issue the Arm Home (50) command. This parameters for this command are:

   1. **Home Position:**

      This is the position to which the Actual Position will be set when the specified trigger occurs.

   2. **Trigger Type:**

      This is the trigger that the RMC will wait for. This is either the Home Input (select rising or falling), or the Z Input, or a combination thereof.

   3. **Repeat mode:**

      This specifies whether the homing should occur every time the trigger occurs, or only once.

4. **Home Input:**
   Specifies which input will be used as the Home input.

When the axis is armed, the Home Armed status bit will be set. This bit can be viewed in the Axis Status Registers All tab, in the Home section.

2. Move the Axis Until it Homes

Move the axis so that it reaches the specified trigger, such as the Home input or Z pulse. To move the axis, use any motion command you wish. Typically, a Move Velocity works well. When the specified trigger occurs, the Actual Position will be set to the specified position. The Target and Command Positions will be adjusted by the same amount that the Actual Position was adjusted, and the motion will therefore not be affected.

When the home has occurred, the Home Latched status bit will be set and the Home Armed status bit will be set. If the **Repeat Mode** parameter was set to **Repeat**, both the Home Latched and the Home Armed status bits will remain set. These status bits can be used in a user program to check that the home has occurred. The Home Input status bit also shows the current state of the home input.

When the home has occurred, issue a command to stop the axis or move it to some desired location.

Quadrature Homing Details

This section gives more detail on homing a quadrature axis.

Trigger Types

The following trigger types are available on quadrature axes.

#	Trigger Type	Description	Valid for		
			RMC75	RMC150	RMC200
0	H Rising	Trigger a Home on the rising edge of the Home Input.	QAx, Q1	Quad, UI/O	Q4, D24, U1
1	H Falling	Trigger a Home on the falling edge of the Home Input.	QAx, Q1	Quad, UI/O	Q4, D24, U1
2	Z	Trigger a Home on the Index (Z) Input.	QAx	Quad	Q4, D24[1], U1
3	Z And H	Trigger a Home on the Index (Z) Input if the Home Input is high.	QAx	Quad	Q4, D24[1], U1
4	Z And Not H	Trigger a Home on the Index (Z) Input if the Home Input is low.	QAx	Quad	Q4, D24[1], U1

[1]For the ABZ option only.

Using the Home Input (Trigger Types 0 - 1)

The Home input is defined as follows:

Feedback Type	Home Input
RMC75 QAx or RMC150 Quad	**Home** input on the axis connector.
RMC75 Q1 Module	**Reg** input on the axis connector.
Universal I/O Module quadrature input	Corresponding **Reg *n*** input on the module.
RMC200 Q4	The associated **Home** input or **Reg** input. Selection is made in the **Home Input** parameter of the Arm Home (50) command.
RMC200 D24 quadrature input	Any of input 16-23, depending on the quadrature option, and defined by the **Home Input** parameter of the Arm Home (50) command.

RMC200 U14 quadrature input	Selection is made in the **Home Input** parameter of the Arm Home (50) command: For channel 0: **Reg/Z0** or **D0**. For channel 1: **Reg/Z1** or **D1**.

When the home is armed and the Home Input trigger occurs, the exact counts at the time the trigger occurred are **latched**. This position is used as the physical home position. The Actual Position at this location is set to the requested **Home Position** as specified by the Arm Home (50) command. The Command Position and Target Position will be offset by the difference between the new and old Actual Positions.

On quadrature axes, the counts are latched within a very short time and are therefore typically very accurate (depending on the encoder resolution, of course). For details on the latching time, see the propagation delay specification for the I/O module. To obtain an accurate trigger, the direction and speed should be the same each time the axis is homed. This is because proximity switches have delays and hysteresis.

Using the Z input (Index Pulse) (Trigger Types 2 - 4)

The Z input is available on the RMC75 QA1 and QA2, RMC150 Quad, RMC200 Q4, RMC200 U14 and the RMC200 D24 module in certain modes.

Homing with the Index (Z) pulse is the most accurate method of homing. When the home is armed and the Index (Z) pulse of the encoder is encountered (with or without the Home Input, depending on the trigger type), the Home is latched. When this occurs, the Actual Position at the location of the Z pulse is set to the **Requested Position** as specified by the Arm Home (50) command. The Command Position and Target Position will be offset by the difference between the new and old Actual Positions.

In order to achieve the most accurate homing with the Index (Z) pulse, you must issue the Learn Z Alignment (54) command when setting up your axis. This command sets the Index (Z) Home Location parameter. Once this parameter is set correctly, you can home with the Index (Z) pulse in any direction, at any speed, and the home will always occur at the exact same position on the encoder. See the Learn Z Alignment (54) topic for details.

Linear Positioner Homing Tips

When using a quadrature encoder on a linear positioner, the axis must be homed to retain absolute positions. This is easiest if the proximity switch controlling the Home input is placed at one end of the travel (for example, in the retracted position); this will be the home position. This is a typical procedure for homing a linear positioner:

1. Arm the homing by issuing an Arm Home (50) command with a **Trigger Type** of **H Falling**.
2. **If the axis is at an unknown position**:

   Retract until the Home Input status bit is ON. This can be done with a Move Velocity command. Then, extend until the Home Input status bit is OFF. When the Home Input status bit turns off, the axis will home and the Home Latched status bit will be set.

   If the axis is already retracted:

   If the axis is already retracted, the Home Input status will already be ON and the axis needs only extend as described above. When the axis is shut down, it should be retracted until the Home Input status bit is ON. This way the axis need only extend a small amount to be homed on startup.

Disarm Home

To disarm the home manually, use the Disarm Home (51) command.

Explicit Homing

The following command set adjusts the axis position without needing a Home or Z input:

- Offset Position (47)
  This command adds the requested **Position Change** to the Actual, Target, and Command Positions.
- Set Target Position (48)
  This command sets the Target Position to the requested **Position**, and adjusts the Command Position, and Actual Position by the same amount as the Target Position changed.
- Set Actual Position (49)
  This command sets the Actual Position to the requested **Position**, and adjusts the Command Position and Target Position by the same amount as the Actual Position changed.

Why Bother?

Explicit homing is useful when the operator physically measures something on the machine and needs to make an adjustment to the positions.

Explicit homing can de done on any axis type, not just those with incremental feedback.

Homing SSI and Resolver Axes

SSI and Resolver axes give absolute positions, and therefore, homing them is not necessary. In certain applications, these axes types can be set up as incremental, and can be homed. Homing SSI axes is not supported on the RMC200.

SSI and Resolver Homing Trigger Types

The following trigger types are available on SSI and Resolver axes. The axis *must* be set to incremental.

#	Trigger Type	Description	Valid for		
			RMC75	RMC150	RMC200
0	H Rising	Trigger a Home on the rising edge of the Home Input.	SSI*		none
1	H Falling	Trigger a Home on the falling edge of the Home Input.	SSI*		
2	Z	Trigger a Home on the Index (Z) Input.	SSI*	SSI*, Resolver*	
3	Z And H	Trigger a Home on the Index (Z) Input if the Home Input is high.	SSI*		
4	Z And Not H	Trigger a Home on the Index (Z) Input if the Home Input is low.	SSI*		
5	Absolute Adjust (H Rising)	On the rising edge of the Home Input, the zero Raw Counts point of the SSI encoder is set to the requested Home Position.	SSI*		
6	Absolute Adjust (H Falling)	On the falling edge of the Home Input, the zero Raw Counts point of the SSI encoder is set to the requested Home Position.	SSI*		
7	Absolute Adjust (Immed)	The zero count point of the SSI encoder is set to the requested Home Position immediately when this command is issued.	SSI*	SSI*, Resolver*	

Using the Home Input for SSI and Resolver Axes (Trigger Types 0 - 1)

The Home input for SSI and Resolver axes is defined as follows:

Feedback Type	Home Input

Incremental SSI	Defined by the SSI Home Source parameter (RMC75 only).
Incremental Resolver	No Home input is available on Resolver inputs.

When the home is armed and the Home Input trigger occurs, the exact counts at the time the trigger occurred are **latched**. This position is used as the physical home position. For SSI and Resolver axes, the counts are only read once per control loop, so the time to latch can be up to one control loop (500µs to 4000µs).

Using the Z input for SSI and Resolver Axes (Trigger Types 2 - 4)

The Z pulse for SSI and Resolver axes is defined as follows:

Feedback Type	Home Input
Incremental SSI	The zero Raw Counts position of the encoder.
Incremental Resolver	The zero Raw Counts position of the resolver.

For incremental SSI and resolver inputs, the Index pulse is given by the point when the Raw Counts cross zero (0). This will trigger a home in the same way that the index (Z) input will. However, there is no Learn Z Alignment parameter and no need to issue the Learn Z command.

Notice that for high-turn-count multi-turn SSI encoders, the Raw Counts may cross zero very infrequently. Therefore, homing with trigger types 2-4 on an SSI encoder is typically only useful for single-turn encoders.

Using the Absolute Home Latch (Trigger Types 5 - 7)

When the home is armed and the specified home trigger occurs, the Actual Position for the SSI axis is adjusted such that the Actual Position at zero Raw Counts of the SSI or Resolver will be set to the requested **Home Position** as specified by the Arm Home (50) command.

Notice that trigger type 7 will perform the homing immediately when the command is issued. Trigger type 7 is supported by the SSI and Resolver, but trigger types 5 and 6 are supported only by the SSI.

3.2.7. System Time

The RMC motion controllers support various levels of keeping track of time. The RMC75, RMC150, and RMC200 include system time registers that keep track of time since the RMC started up. The RMC200 also contains a real-time clock and supports real-time registers.

The system time registers can be used for functions such as delay timers, calculating time between events, and comparing the time occurrence of motion controller events with the times of other devices.

Register Name	Data Type	Time Base	Rollover	Tag Name	RMC75	RMC150	RMC200
System Time - Time base starts when RMC powers up							
Time Gear Master	REAL	1.0 s	1 s	_Time			
Time, 16th Milliseconds	DINT	0.000 062 5 s	1.5 days	_Controller.SysTime_16thmsec			

			(2^{31} ms/16)				
Time, Seconds	DINT	1.0 s	68 years (2^{31} s)	_Controller.SysTime_sec			
Time, Milliseconds	DINT	0.001 s	24.8 days (2^{31} ms)	_Controller.SysTime_msec Or: _SysMS			
Time, Microseconds	DINT	0.000 001 s	35.7 minutes (2^{31} µs)	_Controller.SysTime_usec			
Time, Nanoseconds	DINT	0.000 000 001 s	1 second (1 billion ns)	_Controller.SysTime_nsec			
Time, Loop Ticks	DINT	Same as the Loop Time (0.000 125 s to 0.008 s)	3.1 - 198 days (2^{31} loop times)	_Controller.SysTime_loops Or: _SysTicks			
Real Time - Time base starts Jan. 1, 1970							
Real Time UTC, Seconds	DINT	1.0 s	February 7, 2106 (2^{32} s)	_Controller.RealTimeUTC_sec			
Real Time Local, Seconds	DINT	1.0 s	February 7, 2106 (2^{32} s)	_Controller.RealTimeLocal_sec			
Real Time, Nanoseconds	DINT	0.000 000 001 s	1 second (1 billion ns)	_Controller.RealTime_nsec			

Using the Time registers

Time Gear Master

This register is intended only for use as a master register for gearing to time, such as running a curve based on time. It smoothly increments and wraps every second. It is the only time register that has a type of REAL.

Time, Seconds
Time, Milliseconds
Time, Microseconds

These DINT registers are used for general computing of time differences in units of seconds, milliseconds, or microseconds. The user can do a simple 32-bit subtraction to compare the elapsed time.

For example, to determine when 1 second has elapsed, the current time is first stored in a variable using an expression such as:

time0 := _Controller.SysTime_msec

then the following condition determines when 1 seconds has elapsed since the time was stored:

_Controller.SysTime_msec – time0 > 1000

Since each of these registers is a 32-bit DINT, each value will wrap from +2,147,483,647 to -2,147,483,648. However, because subtracting DINT values uses 32-bit modulo math, the subtraction will yield the intended result as long as no more than 2^{31}-1 (2,147,483,647) time units have elapsed between the two times.

The user must choose which time base provides adequate granularity and rollover times. For many applications in user programs, the **Time, Milliseconds** registers works well, as the default loop time is 1 ms, and times are typically in the range covered by this register (up to 2^{31} ms).

For long times and fine granularity, use **Time, Seconds** together with **Time, Nanoseconds**, as described in **Calculating Time with Seconds and Nanoseconds** below.

Time, Loop Ticks

This DINT register has a time base that matches the current loop time. For example, if the loop time is 1 ms, its time base will be 1 ms. If the loop time is 500 µs, its time base will be 500 µs.

This register is not recommended for use because its time base can be difficult to convert to a normal time, and if a user program employing this register is ported to an RMC with a different loop time, the results may not be consistent. For calculating small time increments, use **Time, 16th Milliseconds** instead.

Time, 16th Milliseconds

This DINT register with a time base of 62.5 µs can be used in some applications where small time increments must be calculated, rather than the **Time, Loop Ticks** register. It is intended to be used for taking simple differences in timestamps, similar to the **Time, Seconds**, **Time, Milliseconds**, and **Time, Microseconds** registers.

Time, Nanoseconds

This DINT register is different from the ones above, since it wraps at 1 billion. Specifically, it cannot be used like this:

_Controller.SysTime_nsec – time0 > 1000 //Do not do this!

because rolling over at 1 billion breaks that difference.

This register is intended to be used in conjunction with **Time, Seconds** for computing accurate time differences, as described in **Calculating Time with Seconds and Nanoseconds** below.

Real Time UTC, Seconds (RMC200 Only)
Real Time Local, Seconds (RMC200 Only)
Real Time, Nanoseconds (RMC200 Only)

These DINT registers capture the real time, as determined by the real-time clock. When the RMC200 powers up, these registers take on the time stored from the onboard real-time clock. The time can also be adjusted via RMCTools. The time is in Unix time, which is seconds and nanoseconds since midnight UTC/GMT January 1, 1970.

The **Real Time UTC, Seconds** register gives the Coordinated Universal Time (UTC), which corresponds to Greenwich Mean Time (GMT). The **Real Time Local, Seconds** register gives the local real time, which is adjusted for the selected time zone and daylight savings time, if applicable. The **Real Time, Nanoseconds** register applies to both.

It is not safe to compute basic mathematical differences based on these real time values because the real time can be adjusted while the controller is running by RMCTools, or a network time protocol. See **Calculating Time with Seconds and Nanoseconds** below.

Calculating Time with Seconds and Nanoseconds

The **Time, Seconds** register (_Controller.SysTime_sec) is used together with the **Time, Nanoseconds** register (_Controller.SysTime_nsec) to calculate the time between two events, with very fine granularity over long time spans. To calculate the time between two events, a time difference calculation must be performed between the two times, of which each time is represented by a seconds DINT and a nanoseconds DINT. This calculation is somewhat complicated, so Delta has provided user functions to capture times and compute differences, as described below.

User Functions

Delta has provided several user functions to perform the calculation between two times. These user functions are available for download from Delta's forum at https://forum.deltamotion.com/. Search for **Time Functions**, download the RMCTools User Function Library (.rmcflib) file, and import it into the User Functions in RMCTools.

Time Capture User Function

This function is for capturing the current system time and storing in two DINT variables.

- **TIME_CAPTURE**(out DINT time_sec, out DINT time_nsec) : DINT
  Captures the current system time, by capturing **_Controller.SysTime_sec** and **_Controller.SysTime_nsec** into the two output parameters **time_sec** and **time_nsec**. The return value is unused and will always be zero.

Elapsed Time User Functions

These functions determine the elapsed time between the current system time and a previously captured time.

- **TIME_ELAPSED_SEC**(DINT t0_sec, DINT t0_nsec) : REAL
  Returns time elapsed since the specified timestamp in seconds with fraction. This value is imprecise because of the limitations of 32-bit floating point math.
- **TIME_ELAPSED_MSEC**(DINT t0_sec, DINT t0_nsec) : REAL
  Returns time elapsed since the specified timestamp in milliseconds with fraction. Unlike TIME_ELAPSED_SEC, this value is precise up to certain limits. For a 1 msec loop time, this value is precise up to approximately 16,000,000 msec, and for a 0.125 msec loop time, is precise up to approximately 2,000,000 msec.

General Time Difference User Functions

The user functions listed above are relatively easy to use when determining how much time has elapsed since an event. Delta also provides more advanced user functions for determining the amount of time between two events, for which the times of each has been stored:

- **TIME_DIFF**(DINT t0_sec, DINT t0_nsec, DINT t1_sec, DINT t1_nsec, out DINT dur_sec, out DINT dur_nsec) : DINT
  Calculates the difference between two times by subtracting the time t0 (seconds and nanoseconds) from t1 (seconds and nanoseconds) and returning the resulting duration in seconds and nanoseconds as output parameters. The return value is unused and will always be zero.
- **TIME_DIFF_SEC**(DINT t0_sec, DINT t0_nsec, DINT t1_sec, DINT t1_nsec) : REAL
  Returns the difference between two times in seconds by subtracting the time t0 (seconds and nanoseconds) from t1 (seconds and nanoseconds) and returning the resulting duration in seconds with fraction as a REAL. This value is imprecise because of the limitations of 32-bit floating point math.
- **TIME_DIFF_MSEC**(DINT t0_sec, DINT t0_nsec, DINT t1_sec, DINT t1_nsec) : REAL
  Returns the difference between two times in milliseconds by subtracting the time t0 (seconds and nanoseconds) from t1 (seconds and nanoseconds) and returning the

resulting duration in milliseconds with fraction as a REAL. Unlike TIME_DIFF_SEC, this value is precise up to certain limits. For a 1 msec loop time, this value is precise up to approximately 16,000,000 msec, and for a 0.125 msec loop time, is precise up to approximately 2,000,000 msec.

Converting to Seconds or Milliseconds

These user functions convert a time represented by two DINTs (seconds and nanoseconds) to a single REAL, in seconds or milliseconds. These functions are typically not necessary, since the TIME_ELAPSED and TIME_DIFF functions above can output the time to a single REAL.

- **DURATION_SEC**(DINT dur_sec, DINT dur_nsec) : REAL
  Converts from a duration in seconds and nanoseconds to seconds with fraction as a REAL.
- **DURATION_MSEC**(DINT dur_sec, DINT dur_nsec) : REAL
  Converts from a duration in seconds and nanoseconds to milliseconds with fraction as a REAL.

Example: Elapsed Time

To calculate the time that has elapsed since a certain event, first, capture the time of the event, and then use a TIME_ELAPSED function to determine the elapsed time.

The functions in this example can be obtained from Delta's forum at https://forum.deltamotion.com/. Search for **Time Functions**, download the RMCTools User Function Library (.rmcflib) file, and import it into the User Functions in RMCTools.



Example: Time Difference

To calculate the time difference between two times that each have been stored as seconds and nanoseconds, use a TIME_DIFF function.

The functions in this example can be obtained from Delta's forum at https://forum.deltamotion.com/. Search for **Time Functions**, download the RMCTools User Function Library (.rmcflib) file, and import it into the User Functions in RMCTools.

First, capturing the time that the first event occurred:



Next, capture the time that the second event occurred:



Finally, calculate the difference between the two times:



3.2.8. Event Timers

Event Timers are used to capture the precise time at which an external event occurs, such as a high-speed discrete input turning on or off, or a quadrature count changing. The time of an event will be latched at a much finer resolution than the Loop Time time of the controller. The

captured time is stored in two 32-bit registers using the standard format for the system time (seconds & nanoseconds since the start of the first motion control loop). The RMC system time is monotonically increasing and will never be adjusted when the real time is changed, so times captures can simply be subtracted from one another.

Event Timers can be used for timing-related applications such as:

- Calculating the time between pulses on a single input.
- Calculating the time between two different inputs turning on.
- Calculate a pulse width.

See the **Applications** section below for instructions on how to perform specific Event Timer applications.

Supported Modules

Module	# of Event Timers	Time Resolution
D24	4	Measurement resolution: 25 ns See the D24 topic for propogation delay specifications.

Capturing the Time of an Event

1. **Determine slot number**
   Determine the slot number of the module with the high-speed input.

Base	Slot Numbers
B5	2–4
B7	2–6
B11	2–10
B15	2–14

2.
3. **Choose an Event Timer and Event Source**
   The **Event Timer** is the internal software resource for the timer, not the actual hardware input. The Event Timers serve to keep track of the physical timing events. The number of simultaneous events that can be timed on a module is limited by the number of Event Timers available for the module.

Module	Event Timers
D24	0–3

4.
   The **Event Source** is the physical event that will be timed. This can be a quadrature count change, or a discrete input change. Certain Event Timers are supported for certain Event Sources.
5. **D24 Hardware Inputs:**

Event Timers	Event Source Names	Hardware Input
0 or 1	Quad Count: Any change in the quadrature count, useful for calculating velocity from last two counts.	Quadrature Channel 0
	In0 Rising Edge, In0 Falling Edge	Discrete Input D20
	In1 Rising Edge, In1 Falling Edge	Discrete Input D21

2 or 3	Quad Count: Any change in the quadrature count, useful for calculating velocity from last two counts.	Quadrature Channel 1
	In0 Rising Edge, In0 Falling Edge	Discrete Input D22
	In1 Rising Edge, In1 Falling Edge	Discrete Input D23

6.

7. **Choose Timer Mode**
   The following modes are available. See the **Applications** section below for more details.
   - **One-Shot**
     The Event Timer will capture the time of the first occurrence of the event after the timer is armed. This is useful when multiple pulses may occur, but only the time of the first one is desired.
   - **Continuous**
     The Event Timer will continue to capture the time of each occurrence of the event after the timer is armed. The time of the most recent event will be stored in the Event Timer registers. This is useful when multiple events may occur within a single loop time, and the time of the last event is desired. To stop capturing, use the Disarm Event Timer (106) command.
   - **Alternating**
     The Event Timer will capture the time of every other occurrence of the event after the timer is armed. To use the Alternating mode, two Arm Event Timer commands must be used on two separate Event Timers, with both Event Timers using the same module slot, event source, and timer mode (Alternating), and the two Event Timers must be in the same pair. On the D24, 0 and 1 are a pair, and 2 and 3 are a pair.
     Once Alternating mode has been armed, the first event timer will latch on the first occurrence of the event, then the second event timer will latch on the next occurrence, then the first event timer will latch on the next occurrence, and the event timers will continue to alternate in this fashion.
     To stop capturing, use the Disarm Event Timer (106) command.
     Alternating mode is useful for measuring the time between the last two counts from the quadrature encoder, which can then be used to calculate speed.
8. **Send Command**
   Send the Arm Event Timer (105) command. The commanded axis is irrelevant. This will arm the specified Event Timer, and the Event Timer Armed status bit will be set. See the **Event Timer Status Bits** section below for more details.
9. **Wait for Event**
   Once the event has occurred, the Event Timer Latched bit will be set. See the **Event Timer Status Bits** section below for more details. For the Continuous and Alternating timer modes, use the Disarm Event Timer (106) command to disarm the timer when done.
10. **View the Latched Time**
    The latched time will be stored in the following registers, as described in the **Event Timer Registers** section below.
    - Event Timer *n* Seconds
    - Event Timer *n* Nanoseconds

11. **Calculate Time Differences**
    Typically, any timing application will involve calculating the time between events. See the **Event Timer Calculations** section below for details on how to calculate the time differences.

Event Timer Status Bits

Each Event Timer has an **Event Timer Armed** status bit and an **Event Timer Latched** status bit:

- **Event Timer Armed status bit**
  The Event Timer Armed status bit will be set when the Event Timer is armed. For the **One-Shot** timer mode, once the Event Timer latches, the Armed bit will be cleared. For the **Continuous** and **Alternating** timer modes, the Armed bit will remain set until the Disarm Event Timer (106) command is sent to disable the Event Timer.
- **Event Timer Latched status bit**
  If the Event Timer is armed, the Event Timer Latched status bit will be set once the event occurs. In One-Shot mode, the Latched bit will remain set until the Event Timer is re-armed or disabled. In Continuous or Alternating mode, the Latched bit will be set only for the loop when the event time is latched, and then cleared on the next loop.

Name	Tag Name	Data Type	Address
Event Timer Armed	none	Boolean	%MD*s*.*b*.0 *s* = 129 + slot number *b* = 101 + 4 x Event Timer Number
Event Timer Latched	none	Boolean	%MD*s*.*b*.1 *s* = 129 + slot number *b* = 101 + 4 x Event Timer Number

Event Timer Registers

The time of an event is recorded in the **Event Timer Seconds** and **Event Timer Nanoseconds** registers. The Event Timer Seconds register holds the number of whole seconds, and the Event Timer Nanoseconds register holds the remaining number of nanoseconds.

Each Event Timer has one dedicated Event Timer Seconds register and one dedicated Event Timer Nanoseconds register. The time uses the standard epoch for the RMC system time (seconds & nanoseconds since the start of the first motion loop). Notice that the **Event Timer Seconds** and **Nanoseconds** registers share the same time base as the **Time, Seconds** and **Time, Nanoseconds** registers described in the System Time topic.

Name	Tag Name	Data Type	Address
Event Timer Seconds	none	DINT	%MD*s*.*b* *s* = 129 + slot number *b* = 102 + 4 x Event Timer Number
Event Timer Nanoseconds	none	DINT	%MD*s*.*b* *s* = 129 + slot number *b* = 103 + 4 x Event Timer Number

Event Timer Calculations

Event times are stored as Seconds and Nanoseconds registers that together represent a time. These two registers share the same time base. For example, a time of 13.568 seconds would be represented by a value of 13 in the Seconds register, and 568,000,000 in the Nanoseconds register.

To calculate the time between two events:

1. Determine the register addresses of the Event Timer Seconds and Event Timer Nanoseconds registers for each of the two events. This is described in the **Event Timer Registers** section above.

2. Check the Event Timer Latched status bits to ensure the timer values have latched. This is described in the **Event Timer Status Bits** section above.
3. Calculate the time difference of the two events.
   Use one of the TIME_DIFF functions provided by Delta, as described in the **Time User Functions** section below. The function will take the Event Timer Seconds and Event Timer Nanoseconds registers for each of the two events.
4. If necessary, convert the time difference to a REAL data type in the desired units of seconds, milliseconds, or nanoseconds.
   The function used to compute the time difference may perform this conversion. Otherwise, use one of the provided functions to perform the conversion.

Time User Functions

Delta has provided several user functions as described in the System Time topic to calculate the difference between two times, and to convert to a REAL value in seconds or milliseconds. The functions can be obtained from Delta's forum at https://forum.deltamotion.com/. Search for **Time Functions**, download the User Function Library (.rmcflib) file, and import it into the User Functions in RMCTools.

Applications

Following are instructions for measuring times for specific applications.

The functions in this example can be obtained from Delta's forum at https://forum.deltamotion.com/. Search for **Time Functions**, download the User Function Library (.rmcflib) file, and import it into the User Functions in RMCTools.

Time Between Pulses on a Single Input

Follow these steps to measure the time between two pulses on a single input. This will capture the two last events prior to measurement. For example, if the inputs alternated 10 times in a loop time after arming, step 1 of the user program would perform calculations on the last two input events.

1. Arm two different timers in alternating mode. The Event Source will be identical for both timers, for example In0 Rising Edge.
2. Wait for the timers to latch.
3. Calculate the time difference.

Example user program:



Time Between Pulses on Two Different Inputs

Follow these steps to measure the time between a pulse on one input, and a pulse on another input. This will capture the first pulse on each input after the Event Timer is armed.

1. Arm two different timers in one-shot mode. Choose the two inputs you wish to use for the Event Sources. Typically, both timers will be the rising edge, but your specific application may differ depending on the edge you wish to measure.
2. Wait for the timers to latch.

3. Calculate the time difference.

Example user program:



Measure Pulse Duration (Pulse Width)

Follow these steps to measure the width of a pulse. This will capture the first pulse on each input after the Event Timer is armed.

1. Arm two different timers in one-shot mode. The Event Sources will be the rising edge and falling edge of the same input.
2. Wait for the timers to latch.
3. Calculate the time difference.

Example user program:



3.2.9. Physical Limit Inputs

Each RMC axis allows for optional physical limit inputs to specify the boundaries in which the axis is allowed to operate. Each axis can have two physical limit inputs: Positive Limit Input and Negative Limit Input. Typically, these inputs are wired to limit switches.

The physical limit inputs differ from the Travel Limits, which limit the range of travel of an axis based on the feedback (position, pressure, etc.). The Travel Limits are required to be set on an axis; physical limit inputs are optional.

Setup

Use the Positive Limit Input and Negative Limit Input parameters to specify the inputs to be used for the physical limit inputs. The following options are possible:

Option	Description
none	This is the default setting.

Fault Input	**RMC75**: The Fault Input of the axis. **RMC150**: The Fault Input of the axis. Only available on the Quadrature Module. **RMC200:** n/a
Dedicated	Available on RMC75 QA and RMC150 Quadrature modules only. For the Positive Limit Input, this is the **PosLim** input. For the Negative Limit Input, this is the **NegLim** input.
general input	**RMC75**: any input from a D8 module, but only from the first 12 I/O points as listed in the Discrete I/O Monitor. **RMC150**: any general discrete input from any module. **RMC200**: any discrete input from any module.

Operation

The Positive Limit Input and Negative Limit Input *status* bits indicate the state of the Limit Inputs. When one of these inputs becomes active (as indicated by the status bits), the corresponding Positive Limit Input or Negative Limit Input *error* bit will be latched. The error bit will cause a Halt to occur if the Positive Limit Input and Negative Limit Input Auto Stops are configured to do so and the Direct Output Status bit is off.

Issuing Commands while a Limit Input is Active

While a Limit Input is active, closed loop motion commands that move the axis *away* from the valid range will set the corresponding Limit Input error bit, causing a halt if the Auto Stops are configured to do so. Motion commands that move *toward* the valid range will not set the a Limit Input error bit. This allows you to move an axis back within the limits without causing further errors.

Therefore, if the Negative Limit Input is active, a motion command that moves the axis in the *negative* direction will cause an error. A motion command that moves the axis in the *positive* direction will not cause an error. Likewise, if the Positive Limit Input is active, a motion command that moves the axis in the *positive* direction will cause an error. A motion command that moves the axis in the *negative* direction will not cause an error.

Moving Toward the Valid Travel Range

If a Limit Input is active, and the axis is in open loop, and is drifting slightly, issuing a closed loop motion command may again trigger an overtravel error. This is because the target position starts at the actual position, velocity and acceleration at the time the command is issued. Therefore, the target may continue moving away from the valid range while accelerating to turn around. This will trigger an overtravel error. To avoid this, first issue a Hold Current Position (5) command, then the motion command. Or, simply use an open loop move to move into the valid range.

Limit Input Polarity

The Limit Input Polarity parameter determines whether Positive Limit Input and Negative Limit Input are active high or active low. The Positive Limit Input status bit and Negative Limit Input status bit indicate whether or not the corresponding limit input is active. For details on voltage levels, see the specifications for the module the input is located on.

Physical Placement of Sensors for Limit Inputs

Typically, the sensors for Limit Inputs should be installed at both ends of travel on the axis. For safe operation, they should be designed such that they become active close to the end of travel, and remain active to the physical end of travel. This increases safety, because motion commands (other than Direct Output) will only be allowed in the direction toward the valid range of travel. A diagram is shown below:



Good Practice

Do not install limit sensors such that they become active close to the end of travel, but then become inactive at the end of travel, as shown below. In this case, commands in the wrong direction will be allowed.



Poor Practice

3.2.10. Feedback Resolution

Feedback Resolution specifies the smallest increment that can be measured by the feedback device (such as a position transducer or encoder). With analog feedback, the resolution can also be a function of the input circuit (Analog-to-Digital converter) on the controller. Sometimes resolution is referred to as granularity.

Why is Feedback Resolution Important?

There are two main reasons why feedback resolution is important:

a. **Positioning Accuracy**
The controller cannot hold a position if it can't accurately determine how close it is to the desired position. It is generally necessary to have resolution that is several times better than the desired accuracy. Notice, however, that high resolution is a requirement for, but is not equivalent to high accuracy.

b. **Quantization Noise**
A less obvious, but equally important reason is quantization noise. Since velocity is the change in position per unit of time, the velocity resolution is dependent on the position resolution and the controller loop time. If the controller has a 1 millisecond loop time, the velocity resolution will be 1000 times worse than the position resolution. The acceleration measurement will be 1000 times worse than the velocity measurement. The differential gain and double differential gain use the actual velocity and actual acceleration respectively, to increase the system stability. Excessive quantizing noise severely limits the effectiveness of these gains.

Maximum Feedback Resolution for the RMC

The maximum resolution available on the RMC for various feedback types is listed below:

- **MDT**
For MDT Start/Stop or PWM feedback with a typical transducer having a gradient (or calibration constant) of 9 µs/in, the resolution of obtained by the RMC is:
  - RMC75 MA Module: 0.0005 inch with 1 recirculation
  - RMC150 MDT Module: 0.001 inch with 1 recirculation
  - RMC200 S8 and U14: 0.0005 inch with 1 recirculation
- **SSI**
The SSI transducer sends the position information digitally, so the only limit is in the transducer or encoder. One micron resolution is common for linear SSI transducers, and 8192 counts per turn is common for rotary encoders. Other resolutions are readily available.
The number of supported SSI bits is:
  - RMC75 MA module: 8 to 32 SSI bits

- RMC150 Universal I/O module: 8 to 32 SSI bits
- RMC150 SSI module: 8 to 31 SSI bits
- RMC200 S8 and U14: 8 to 32 SSI bits

Delta recommends that the SSI Counts value should not exceed 24 bits (16,777,216). See the **Exceeding 24 Bits** section below.

- **Load Cell**

  The analog-to-digital converter on the LC8 load cell input is 24 bits. The effective resolution is lower, at approximately 17 bits, depending on the LC8 module filter settings and external noise. Given that the input range is ±34.5 mV, and an approximate resolution is 17 bits, the effective resolution is 540 nanovolts.

  Note:
  Typically, noise on the load cell wires will exceed these small resolution values, and will have a larger impact on system performance.

- **Analog**

  The RMC Analog-to-Digital converter support the following number of bits:

  - RMC75 AA, A2, and AP2: 16 bits
  - RMC150 H, G, and Universal I/O: 16 bits
  - RMC150 A: 12 bits
  - RMC200 A8 and U14: 18 bits

  The effective resolution of the Analog-to-Digital converted signal is increased by the following items:

  1. **Oversampling**
     The analog-to-digital converters are read multiple times per sample, increasing the effective resolution of the Analog-to-Digital converted signal:
     1. AA, A2, AP2, H, and G: eight times oversampling per sample, increasing the effective resolution of the Analog-to-Digital converted signal. For example, the eight times oversampling causes a 16-bit input's effective resolution to be 19 bits (one part in 524,288) over the full ±10V range.
     2. RMC150 Universal I/O Module: sampled at 60kHz, which is a minimum of 15 times oversampling.
     3. RMC200 A8 and U14 module: sampled internally at 200 kHz. The A8 is effectively 20 bits, and the U14 is effectively 21 bits.
  2. **Input Range Gain**
     Choosing the ±5V or 4-20mA ranges on the RMC150 H or G modules or the 4-20 mA range on the U14 module changes the gain of the analog input, increasing the resolution over the requested range.

  Due to the factors listed above, the effective resolution of the RMC analog inputs is as shown in the table below.

Module	±10 V	±5 V	4-20 mA
AA AA, A2, AP2	38.1 µV		0.15 µA
Analog (H)	38.1 µV	19.1 µV	0.076 µA
Analog (G)	38.1 µV		
Analog (A)	610 µV	305 µV	1.221 µA
Universal I/O	38.1 µV		0.15 µA
A8	20.1 µV		0.0805 µA

U14	9.77 µV		0.0316 µA

Note:
Typically, noise on the analog signals will exceed these small resolution values, and will have a larger impact on system performance. This is especially true on 18-bit inputs such as the A8 and U14 modules.

- **Quadrature**
  Quadrature encoder resolution is specified in pulses (also called lines) per revolution on rotary encoders or pulses per inch (or meter) on linear encoders. The RMC quadrature input will decode the pulses to generate four counts per pulse. So the internal resolution on the RMC will be four times the resolution of the encoder given in pulses. For example, encoders with 2000 pulses will have 8000 counts.

  The maximum quadrature encoder frequency is limited:

Module	Max quadrature frequency (counts/sec):
RMC75 QA, Q1	8,000,000
RMC150 Quad	4,000,000
RMC150 UI/O	8,000,000
RMC200 Q4	12,000,000, depending on input type
RMC200 S8	8,000,000
RMC200 D24	1,000,000, depending on input type
RMC200 U14	8,000,000, depending on input type

  Delta recommends that the quadrature Counts value should not exceed 24 bits (16,777,216). See the **Exceeding 24 Bits** section below.

- **Resolver**
  The RMC150 Resolver module can be configured for 14 or 16 bits per revolution. One revolution of the resolver will always consist of 65,536 counts on the RMC, regardless of the resolver resolution parameter. With 16-bit resolution, the counts will increment by one; with 14-bit resolution the counts will increment by four.

Exceeding 24 Bits

The quadrature and SSI inputs can handle Counts values up to 32 bits. However, Delta recommends that you design your system and programming such that the quadrature and SSI Counts value do not exceed 24 bits (16,777,216). The RMC can still interface with SSI devices that have more than 24 bits, but you should make sure the counts will not exceed 16,777,216. These limitations do not apply to voltage, current, or MDT feedback types, since their values never exceed 24 bits.

Delta recommends that the Counts value not be allowed to exceed 24 bits (16,777,216) because this causes the Actual Position to lose resolution. This occurs because the Actual Position is stored as a 32-bit floating point number, which is limited to 24 bits of precision. The resolution of a floating-point number depends on how large the number is. For example, a floating point number can precisely represent any integer that fits in 24 bits (-16,777,216 to +16,777,216) or it can represent numbers at a 0.001 resolution in the range of -16,777.216 to +16,777.216.

To determine when an axis' Actual Position will lose resolution, look at the Counts register. As long as the Counts register stays within 24 bits (-16,777,216 to +16,777,216), then the Actual Position register will approximately match the resolution of the transducer. However, as the Counts move outside that range, the Counts register and the Actual Position register will lose resolution.

For example, if the Counts are 16,777,220 (slightly larger than 24 bits), and the transducer counts (the counts directly from the transducer are represented exactly in the

Raw Counts register, which may be different from the Counts register, but it does indicate the change in counts exactly) change by one to 16,777,221, the Counts value will still read 16,777,220. It will not change until the counts have changed by 2, to 16,777,222. This will cause the Actual Position to become "jerky" and will affect the control. This problem is doubled for each power-of-two increase in the Counts value. Notice that the Counts and Actual Position registers are still accurately keeping track of the position change, but they are losing resolution.

Notice that in order to preserve accuracy on incremental feedback types, the RMC internally maintains a 32-bit integer accumulator. This ensures that the position does not drift due to loss of resolution.

Rotary axes use the Count Unwind value, which will keep the Counts within a defined range. Therefore, as long as the Count Unwind is kept below 16,777,216, this additional loss in resolution will not occur.

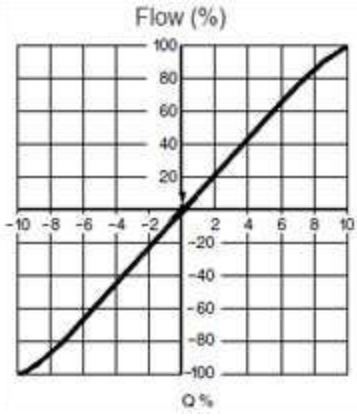
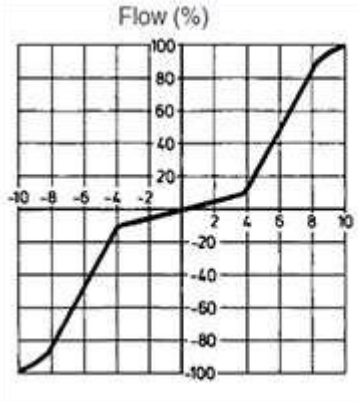
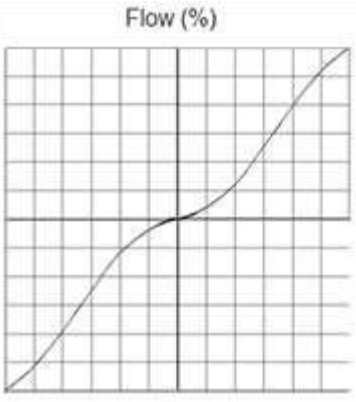
For linear axes, you can keep the Counts register from exceeding 24 bits by homing the axis, or using the Set Actual Position (49) or Offset Position (47) commands. With quadrature encoder inputs, the usable range with full resolution can be doubled by setting the zero value of the Actual Position to the middle of travel. For absolute linear SSI feedback, you can also use the Count Offset parameter to move the usable Counts range closer to zero.

3.2.11. Valve Linearization

Valve linearization refers to compensating for non-linear hydraulic valves.

Valve Types

Linear and **non-linear** refer to the flow versus command signal profile of a valve. Typically, Delta recommends using linear valves. Linear valves are nearly always of very high quality, provide excellent response and are very easy to set up. Very high-quality, non-linear valves may be useful in certain situations that require high flow together with very fine positioning or pressure accuracy.

Linear	**Non-linear**	
	Single-knee	**Curvilinear**
The flow is proportional to the command signal input and does not require linearization.  Notice that it is common for the flow profile to roll off as it approaches 100%. This is common, and the valve is still considered linear.	The flow has a single sharp 'kink' and can be linearized via the Single-Point method. 	These flow profiles can be linearized via curves. 

Linearization Types

Valve linearization is applied only when the axis is in closed loop control. The RMC supports valve linearization via two methods:

- **Single-Point**
  Used only for valves with a single "knee" or "kink" in the flow versus command signal diagram
- **Curves**
  Used for valves with any flow diagram, especially curvilinear valves.

Single-Point Valve linearization

The RMC provides single-point valve linearization to compensate for valves with one sharp "knee" or "kink" in the flow versus command signal diagram, as shown above. The flow diagram is assumed to include the points (0,0), (-100,-100), and (100,100).

To Apply Single-Knee Valve Linearization:

1.

1. In the Axis Tools, in the Axis Parameters Pane, on the **All** tab, expand the **Output** section.
2. In the **Valve Linearization Type** cell, choose **Single-Point**
3. Set the axis parameters as follows:

RMC75/150:

Knee Command Voltage: The input voltage value of the knee. This voltage can be obtained from the valve data sheet. For example, in the Single-Knee Valve diagram above, on the horizontal axis, the voltage is 4 volts.

Knee Flow Percentage: The flow percentage of the valve at the knee. This value can be obtained from the valve data sheet. For example, in the Single-Knee Valve diagram above, on the vertical axis, the flow percentage is 10%.

RMC200:

Knee Command Input: The input value of the knee. This percentage value can be obtained from the valve data sheet. For example, in the **Single-Knee Valve** diagram above, on the horizontal axis, the voltage of 4 volts would equal an input value of 40%.

Knee Flow Output: The flow percentage of the valve at the knee. This value can be obtained from the valve data sheet. For example, in the **Single-Knee Valve** diagram above, on the vertical axis, the flow percentage is 10%.

5. Download the changes and update Flash.

Creating a Curve for Valve Linearization

To create a curve for valve linearization, make a curve that matches the flow profile of the valve, with the x-axis being the input signal in percent, and the y-axis being the flow output in percent. For overlapped-spool valves, see **Curve Valve Linearization and Deadband** below.

Most valve flow profiles are given as positive flows for both positive and negative x-values. The curve must have negative flows for negative x-values.

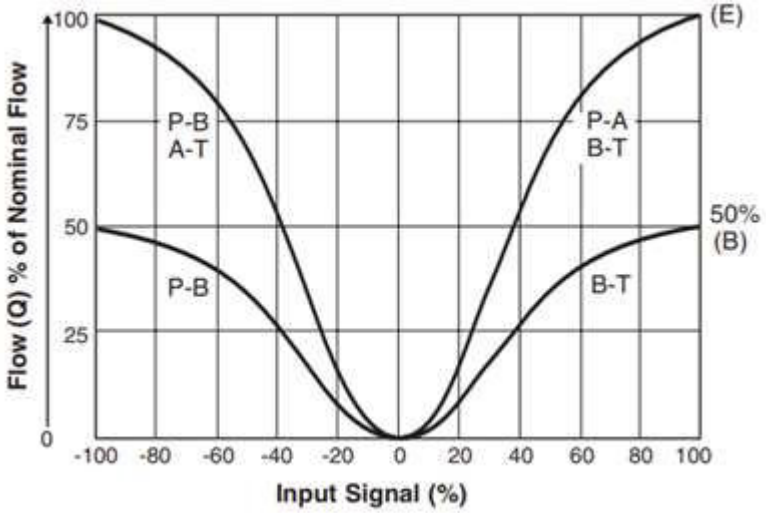
The curve must meet the following requirements:

- Curve Properties:
  - **Curve Type** must be **Valve Linearization**.
  - **Endpoint Behavior** must be **Natural Velocity**.
  - **Interpolation** must be **Linear** or **Cubic**.
- The first point must be (-100, -100).
- The last point must be (100, 100).
- The curve must contain the point (0,0).
- The x-values must be within the range -100 and 100.
- The y-values must be within the range -100 and 100, including the interpolated portions of the curve between points.
- The curve must be positive monotonic. This means the curve is always increasing or flat. Any flat section, if present, should only be around the point (0,0). See Curve Valve Linearization and Deadband below.

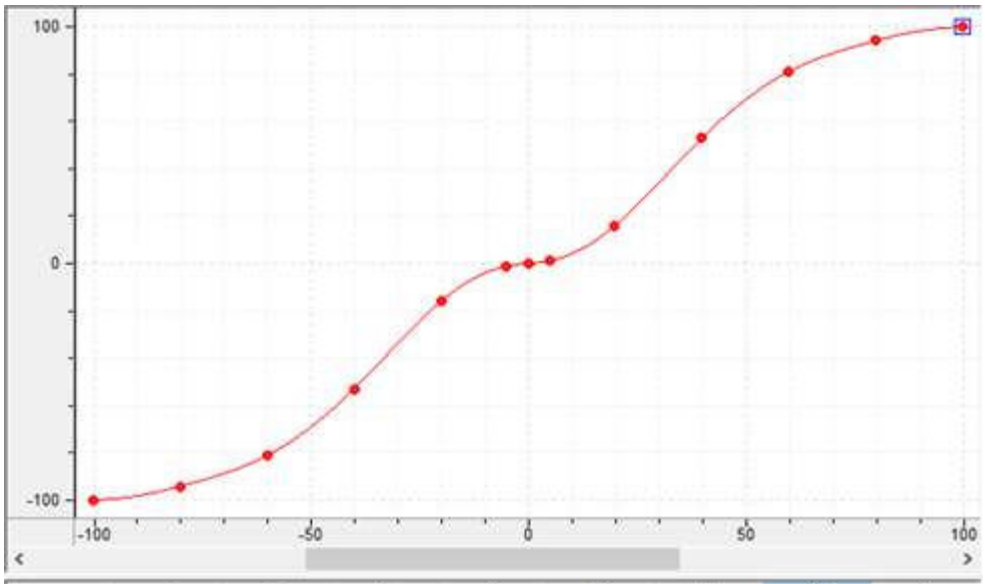
Tip: If the y-values of the flow profile are in gpm or lpm, you will need to convert them to percent when creating the curve.

Example:

Consider profile E in this valve flow profile diagram:

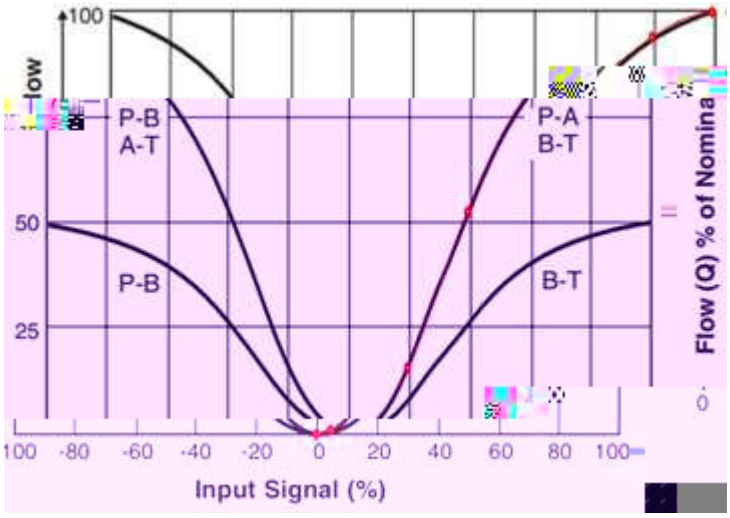


The following curve was created for this flow profile. Points were chosen at 20, 40, 60, and 80, plus an additional point close to zero (5.1, 1.1) so that the curve follows the profile accurately in that region. The negative values mirror the positive values.



	6	7	8	9	10	11	12	✳
X Value	0.0	5.1	20.0	40.0	60.0	80.0	100.0	
Y Value	0.0	1.1	16.0	53.0	81.0	94.0	100.0	
Velocity	0.136	0.375	1.531	1.909	0.965	0.492	0.125	0.0
Type	Standard	Standard	Standard	Standard	Standard	Standard	Standard	Standard

One way to verify that the curve matches is to take a screen shot and overlay the curve on the flow profile using an image editing program:



Curve Valve Linearization and Deadband

Deadband refers to the range for an overlapped spool where the flow is zero until the command signal reaches a certain value.

If the valve has deadband, follow these steps for valve linearization via curves:

1. Create a curve that matches the valve flow profile and select that curve for the axis' valve linearization. See **Creating a Valve Linearization Curve with Deadband** below.
2. If the axis hunts when it is in position, you can mitigate it as follows:
    1. Set the Output Deadband axis parameter to the deadband of the valve.

2. Set the Deadband Tolerance axis parameter to a small value. This will ratio the Output Deadband when the Actual Position is close to the Command Position to prevent hunting. The axis typically holds position within a tolerance of about half of the Deadband Tolerance.

Creating a Valve Linearization Curve with Deadband

A linearization curve for a valve with deadband must follow the requirements listed in **Creating a Curve for Valve Linearization** above. It can be especially challenging to create the flat spot around zero, since the curve tends to become non-monotonic.

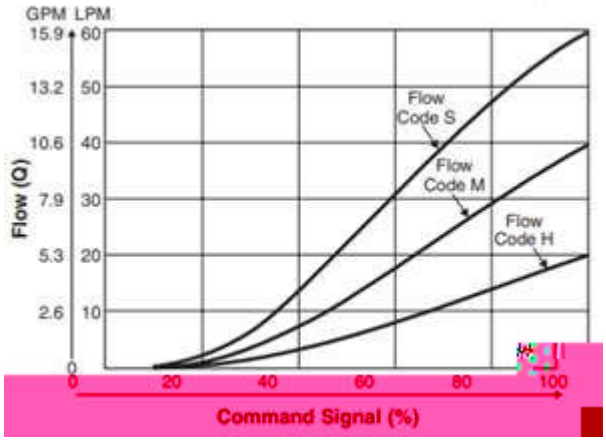
To make a curve flat around zero:

1. For the two points next to zero on the x-axis (the points to the left and right of zero), set the y-values to zero.
2. If using a cubic interpolation:
   For the point to the *left of zero*, and for the point *at zero*, in the spreadsheet editor, set the point type to **Const Vel**.
   Or, in **Curve Properties**, you can set the **Auto Constant Velocity Behavior** to **Enabled**.

Tip: The actual deadband of the valve may vary from the profile given in the datasheet. You may need to manually determine the deadband of the valve and adjust the curve accordingly.

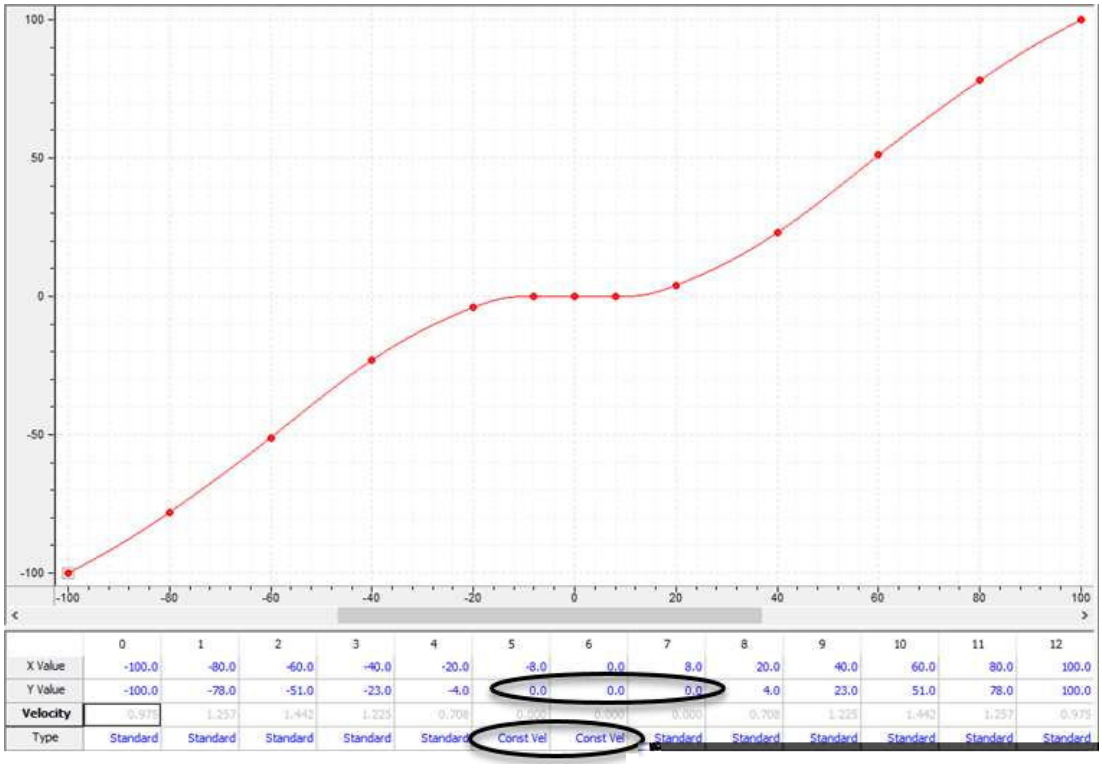
Example

Consider profile S in this valve flow profile diagram:



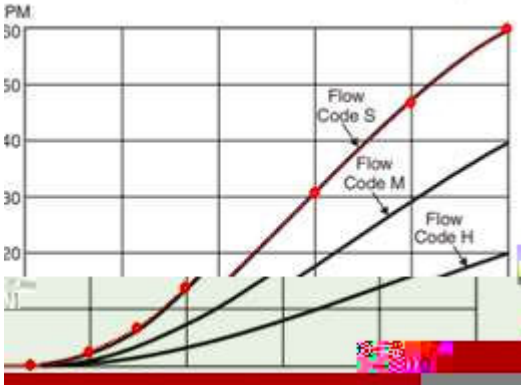
The curve below was created for this flow profile. Points were chosen at 8, 20, 30, 40, 60, and 80 so that the curve follows the profile accurately. The y-points were converted to percent. The negative values mirror the positive values.

Notice that the y-values for points 5, 6 and 7 are zero, and the Type for points 5 and 6 is Constant Velocity. The Constant Velocity type is necessary to maintain the monotonicity of the flat section of a cubic-interpolated curve.



	0	1	2	3	4	5	6	7	8	9	10	11	12
X Value	-100.0	-80.0	-60.0	-40.0	-20.0	-8.0	0.0	8.0	20.0	40.0	60.0	80.0	100.0
Y Value	-100.0	-78.0	-51.0	-23.0	-4.0	0.0	0.0	0.0	4.0	23.0	51.0	78.0	100.0
Velocity	0.975	1.257	1.442	1.225	0.708	0.000	0.000	0.000	0.708	1.225	1.442	1.257	0.975
Type	Standard	Standard	Standard	Standard	Standard	Const Vel	Const Vel	Standard	Standard	Standard	Standard	Standard	Standard

Overlaying the curve on the flow profile using an image editing program shows that the curve closely follows the profile:



3.3. Axes

3.3.1. Axis Types

The axis types listed in this topic are available on the RMC.

What is an axis?

An axis is a combination of inputs and/or outputs in one single entity, and includes all the necessary motion features for those inputs and/or outputs, which can include feedback scale and offset, filtering, control algorithms, output scaling, etc.

The major types of axes available in the RMC are:

- **Control Axis:**
  Has one Control Output and control either zero, one or two feedback quantities, such as Position, Velocity, Acceleration, Pressure, or Force.
  A control axis with no feedback quantities is called an **Output Only** axis.
- **Reference Axis:**
  Has an input for transducer feedback and does not have a Control Output. A reference axis is not capable of controlling a system. Reference axes are commonly used in gearing, synchronization, or monitoring applications.
- **Virtual axis:**
  Has only a virtual target position (with velocity and acceleration) with no feedback or control loop. Typically used as a gear or cam master.
- **Cascading Outer Loop**:
  A control axis that provides a virtual Control Output, and does not use a physical Control Output. For more details, see the Cascade Control topic.

Counting Axes

Any axis in the RMC counts as one single axis, regardless of the number of inputs or outputs. For example:

- A position-force application may require one analog position input, two analog pressure inputs, and 1 output, and is considered one axis.
- A reference axis with only one input is considered one axis.
- A virtual axis has no feedback or output and is considered one axis.

Supported Number of Axes per Controller

	RMC75	RMC150	RMC200 CPU20L	RMC200 CPU40
Max Control Axes[1]	2	8	18	50
Max Total Axes, including Virtual, Reference, and Outer Loop	4	16	48	128

[1] For the RMC75 and RMC150, Control Axes includes Output Only axes. For the RMC200, Control Axes does not include Output Only axes.

Note: It is possible to add more analog inputs on the RMC75 than can be assigned to axes. However, it is still possible to view the voltage of the extra analog inputs using the Analog Input Registers. Inputs that are unassigned to axes can have no status bits, error bits, scaling, filtering, etc.

Axes and Loop Time

The number of axes for each RMC is limited depending on the specified Loop Time. See the Loop Time topic for details.

Axes Physical Feedback Types

The RMC supports the physical feedback types listed below. Notice that any type of feedback (e.g. torque, temperature, etc.) can be controlled if the transducer is compatible with the RMC.

Any of these feedback types can be used as part of a Control Axis, Reference Axis, or Cascading Outer Loop axis.

Feedback Type	Transducer Type	Supported Transducer Output Types
Position Notice that a position feedback control axis can also control velocity.	position	MDT
		SSI
		Quadrature
		Resolver
		Analog Voltage or Current
Velocity	tachometer	Analog Voltage or Current
Acceleration (single- or dual-input)	1 or 2 accelerometers	Analog Voltage or Current
Pressure	pressure sensor	Analog Voltage or Current
Force (single-input)	load cell (±10V, 0-10V, ±5V, 4-20mA)	Analog Voltage or Current
Force (single-input)	load cell (mV/V)	Millivolts/Volt
Force (dual-input, differential)	2 pressure sensors	Analog Voltage or Current

Custom Feedback

An axis' feedback can be defined as custom, which requires creating a user program to continuously calculate the feedback value. This allows for using the sum, difference, or average of other transducers as feedback. It also provides for switching feedback on-the-fly, redundant feedback, and feedback linearization. For details, see Custom Feedback.

Dual Loop Control

The RMC supports dual-loop control. This is typically used for controlling two quantities, such as position and force, with a single actuator. See the Dual-Loop Axes topic for details.

Feedback Type	Description
Position-Pressure	Used for controlling both position and pressure with one actuator. Typically used with hydraulic cylinders.
Position-Force	Used for controlling both position and force with one actuator. Typically used with hydraulic cylinders.
Position-Acceleration	Used for advanced control such as active damping, which provides precision control of difficult systems, such as pneumatic systems.
Velocity-Pressure	Used for controlling both velocity and pressure with one actuator.
Velocity-Force	Used for controlling both velocity and force with one actuator.
Velocity-Acceleration	Used for advanced control applications.

Defining Axes

The user has full control over how the inputs and outputs on the Axis Module and the inputs on the Expansion Modules are assigned to axes.
See the Defining Axes topic for details on assigning axes to the hardware.

Other Axis Types

In addition to the axis types listed above, further distinction can be made between axes:

- Rotary vs. Linear
- Absolute vs. Incremental

3.3.2. Defining Axes

When creating axes, the user must define the axis type and define which physical hardware is assigned to each axis. For details on the axis types supported by the RMC, see the Axis Types Overview topic.

Changing the Axis Definitions

When the RMC powers up for the first time, it will create default axes. You may need to change the axis definitions to fit your application. To change the axis definitions, use the Axis Definitions Dialog:

- In the Project pane, expand the **Axes** folder and double-click **Axis Definitions**.
- To remove an axis, click the **Remove** button. To add a new axis, click **New**. To change an existing axis, click **Change**.
- To change the axis order, use the **Move Up** and **Move Down** buttons. The axes will adhere to this order throughout the entire RMC. The Axis Tools and Command Tool will display the axes in this order, and the register addresses and tag names will be defined by this order.

Note: If the controller contains a Universal I/O module, you must first set up it's channels prior to defining axes.

You will need to define the following items for each axis:

Item	Options
Axis Type	**Control:** Has a physical Control Output and zero to two inputs **Reference:** Has only an input **Virtual:** Has no feedback or output. Has only a target. **Cascading Outer Loop:** Used for Cascade Control
Control Loops (Control Axes and Cascading outer Loop Axes Only)	**None:** For axes with no feedback, only a Control Output **Single-Loop:** For controlling a single quantity, such as position, velocity, acceleration (single- or dual-input), pressure, or force (single- or dual-input). **Dual loop:** For controlling two quantities with a single Control Output, such as position-pressure, position-force, position-acceleration, velocity-pressure, velocity-force, velocity-acceleration.
Output (Control Axes Only)	**Type:** Analog output.

	Using: Choose which physical Control Output on the RMC to use.
Feedback Types (for axes with feedback)	**Type:** • Position • Velocity • Acceleration (single- or dual-input) • Pressure • Force (single- or dual-input) **Using:** Choose which physical inputs will be assigned to the axis, or select **Custom**.

Axis Labels

In addition to the axis number assigned by RMCTools, each axis can be assigned a name by the user. To change the name of an axis, in the Axis Definitions Dialog, select an axis and click **Rename**.

Notice that the connectors on the front of the RMC75 labeled "Axis 0" or "Axis 1" do not necessarily have to belong to the internal Axis 0 or Axis 1 in the RMC75. The connectors on the RMC150 also do not necessarily indicate which internal axis they belong to.

3.3.3. Control and Reference Axes

3.3.3.1. Axis Type: Control

A control axis has a Control Output and controls either zero, one or two quantities, such as Position, Pressure, or Force. The controlled quantity is provided by a feedback input. A control axis is capable of controlling a system because it has a Control Output.

There are three types of control axes:

1. One-Input Control Axis

A control axis that controls a single quantity, such as position or pressure, is called a *one-input control axis*.

One-Input Control Axis Types	Required Inputs	Required Control Outputs	Transducer Types
Position	1	1	MDT, SSI, Analog, Quadrature, Resolver
Velocity	1	1	Analog Voltage or Current
Pressure	1	1	Analog Voltage or Current
Force	1 or 2	1	Analog Voltage, Current, Load Cell

Note:
A force input requires 1 analog or mV/V input if a load cell is used, or 2 analog inputs for differential force on a hydraulic cylinder. In the differential case, it is still considered one input, since it is only one quantity.

2. Two-Input Control Axis

A control axis that controls two quantities, such as position *and* pressure, is called a *two-input control axis*. Notice that the secondary inputs must always be analog or mV/V inputs.

Two-Input Control Axis Types	Required Primary Inputs	Required Secondary Analog or mV/V Inputs	Required Control Outputs
Position-Pressure	1	1	1
Position-Force	1	1 or 2	1
Position-Acceleration	1	1 or 2	1
Velocity-Pressure	1	1	1
Velocity-Force	1	1 or 2	1
Velocity-Acceleration	1	1 or 2	1

Note:
A force input requires 1 analog or mV/V input if a load cell is used, or 2 analog inputs for differential force on a hydraulic cylinder. In the differential case, it is still considered one input since it is only one quantity, although it requires two physical analog inputs.

3. Output Only Axis (No Input)

An Output Only axis has a Control Output and does not have any inputs. Control Outputs are available only on an axis module.

Creating a Control Axis

For details on defining axes, see the following topics:

Defining Axes

Axis Definitions: Dialog

Axis Definitions: Edit

3.3.3.2. Axis Type: Reference

A **reference axis** is an axis that has only an input, such as position, pressure, or force. A reference axis does not have a Control Output and cannot control anything. A reference axis is commonly called a "half-axis".

Reference axes are typically used for monitoring some quantity, such as:

- Position
- Velocity
- Acceleration
- Pressure
- Force

Creating a Reference Axis

To create a Reference axis, you must add a new reference axis or change an existing axis. For details on defining axes, see the following topics:

Defining Axes

Axis Definitions Dialog

Axis Definitions: Edit

Using Reference Axes

Reference axes are typically used for monitoring some quantity. The following are examples of specific uses with the RMC:

- Gear some axis to the reference input.
- Use the value of the reference to affect the flow of the User Program.
- Use the value of the reference in a Expression command in the User Program to perform mathematical calculations.

3.3.3.3. Axis Type: Cascading Outer Loop

A Cascading Outer Loop axis is used as the outer loop of cascaded axes. A cascading outer loop axis is identical to a control axis, but it provides a virtual Control Output, and does not use a physical Control Output. This is only for advanced applications. For details, see the Cascade Control and control axis topics.

For descriptions of all available RMC axis types, see the Axis Types Overview topic.

There are two types of cascading outer loop axes:

1. One-Input Control Axis

A control axis that controls a single quantity, such as position or pressure, is called a *one-input control axis*.

One-Input Control Axis Types	Required Inputs	Transducer Types
Position	1	MDT, SSI, Analog, Quadrature, Resolver
Velocity	1	Analog Voltage or Current
Pressure	1	Analog Voltage or Current
Force	1 or 2	Analog Voltage or Current, or mV/V

Note:
A force input requires 1 analog or mV/V input if a load cell is used, and 2 analog inputs for differential force on a hydraulic cylinder. In the differential case, it is still considered one input, since it is only one final quantity.

2. Two-Input Control Axis

A control axis that controls two quantities, such as position *and* pressure, is called a *two-input control axis*. Notice that the secondary inputs must always be analog or mV/V inputs.

Two-Input Control Axis Types	Required Primary Inputs	Required Secondary Analog or mV/V Inputs
Position-Pressure	1	1
Position-Force	1	1 or 2

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP