RMC100 Motion Controller

and

RMCWin Software

User Manual

Version 2.31.2   December 9, 2015

Condensed Contents

Contents

Table of Contents

Disclaimer

1 Introducing the RMC100 Series

1.1 RMC100 Overview

The RMC100 series brings the benefits of modular, high-performance motion control to a wide range of industrial applications. Communications options—ranging from high-speed field buses to discrete I/O—make these controllers an excellent choice for large and small systems. Transducer types can be combined to control any hydraulic, electric, and pneumatic system. Powerful control modes—including position/pressure control, synchronized moves, gearing, splines, and teach mode—provide optimum control for motion applications.

The RMCWin software is used with the RMC100.

Features

- Two to eight axes of position or speed control
- Up to four axes of position/pressure control. See Controlling Pressure or Force.
- Isolated power input, drive outputs, discrete and analog I/O, and communications
- RS232 port for RMCWin and the RMCCOM ActiveX control. See RMCWin Overview.
- Full PID with velocity and acceleration feed forwards
- Motion and pressure profiles can be changed on the fly. See Motion Profiles.
- Gains and Feed Forwards can be changed on the fly
- 256K Flash memory for field upgrades and parameter storage. See Flash Memory.
- Trapezoidal, s-curve, and spline profiling
- Curve Tool for generating cubic splines. See Spline Overview.
- Teach mode. See Teach Mode Overview.
- Synchronization of 2-8 axes. See Synchronizing Axes.
- Electronic gearing. See Gearing Axes.
- Deterministic Event Control. See Event Control.
- Speed Control mode. See Speed Control.

Applications

- Presses
- Injection/RIM/blow molding
- Packaging equipment
- Indexing/transfer lines
- Edgers/headrigs/veneer lathes
- Pinch rollers/winders/wrappers
- Casting/forging
- Palletizers/stackers
- Flying cutoff/curve sawing
- Cyclic testing
- Robotics/animatronics
- Pneumatic press rolls
- Tube bending/forming

Communications

- PROFIBUS-DP
- Ethernet
- Modbus Plus
- Discrete I/O
  - CPU Digital I/O
  - Communication Digital I/O
  - Sensor Digital I/O
- Serial RS232/422/485
- LCD420 Terminal

Position, Pressure, Force and Velocity Transducer Interfaces

- Magnetostrictive Transducers (MDT) – Start/Stop and PWM
  See MDT Overview for Start/stop and PWM MDTs.
  See SSI Overview for SSI MDTs.
- Analog Transducers
  See Analog Overview for analog position, velocity, pressure, or force transducers.

- Quadrature Encoders
  See Quadrature Overview for encoders for servo motors.
  See Stepper Overview for encoders for stepper motors.

- Synchronous Serial Interface (SSI) – Absolute encoders and MDTs
  See SSI Overview for all transducers with SSI interfaces.

Pressure/Force Options

- Control pressure or differential force at 12- or 16-bit resolution

- Transition between position and pressure/force while in motion

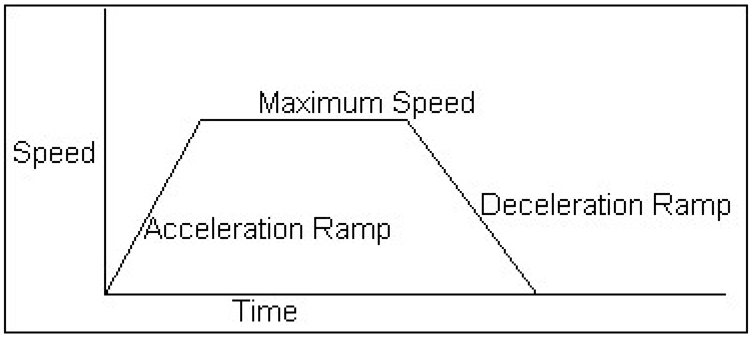
RMCLink ActiveX Control and .Net Assembly

Control the RMC from your Visual Basic, Visual C++, Java, or VBA (e.g. Excel) programs. See the RMCLink topic for details.

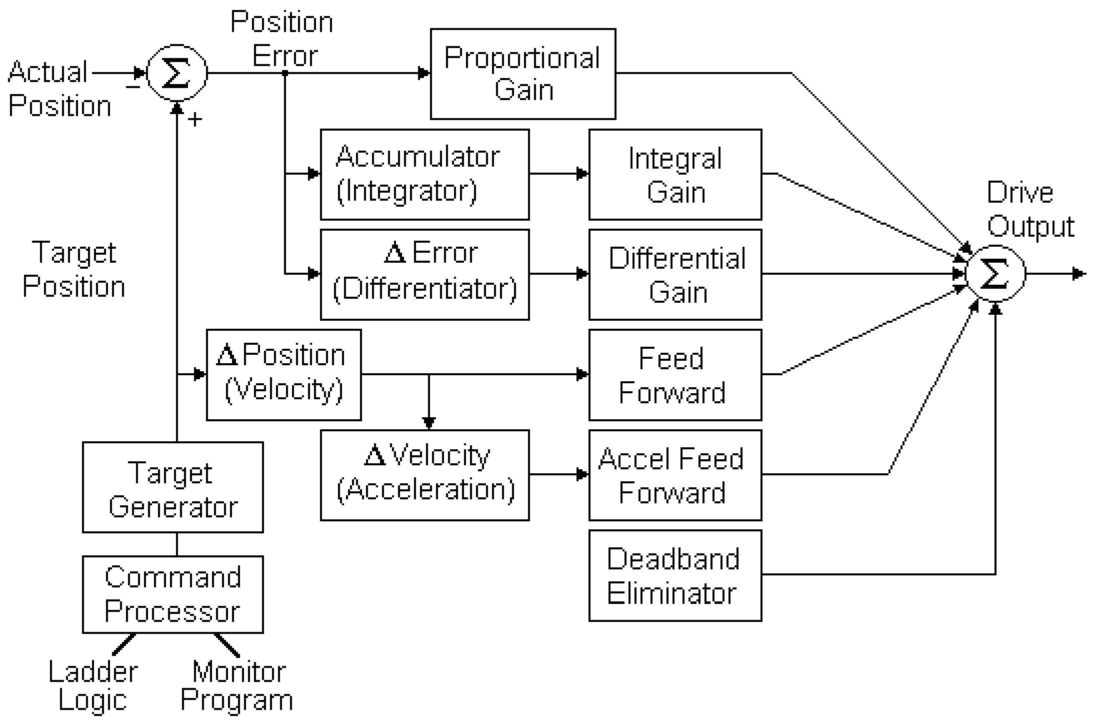
1.2 Principle of Operation

Control Loop

This motion controller is a targeting controller; the onboard microprocessor updates the Target Position and Target Speed values each control loop (1ms or 2ms, depending on the number of modules). For point-to-point moves, Target Positions are generated so the target speed follows a profile. The Mode, Acceleration, Deceleration, Speed, and Command Value (requested position) are used to generate the profile. They are specified by the user, and can be changed while the axis is moving. A trapezoidal profile is shown here.



The Actual Position, measured by any of the RMC’s position transducers, is compared with the Target Position to determine the position error. Every control loop the position error is used to calculate the closed loop components of the drive output. It is multiplied by the Proportional Gain to calculate the proportional component of the drive output. The accumulated position error is used, along with the Integral Gain, to calculate the integral portion of the drive output. The change in position error, along with the Differential Gain, is used to calculate the differential portion of the drive output.



In addition to the closed loop drive, this motion controller has two feed forward terms, made up of Extend and Retract Feed Forward, and Extend and Retract Acceleration Feed Forward. These feed forward terms give approximately the drive needed to make the axis follow the target, freeing the PID loop to correct for non-linearity in the system and changes in system load.

Drive Output

The drive generated by the motion controller is sent through optical isolation to a 12-bit digital-to-analog converter (DAC). The output from the DAC is amplified to provide a ±10 volt output to the hydraulic valve. Servo valves that need current input require a voltage-to-current converter (Delta part number VC2100). Other proportional valves are typically driven using another type of external amplifier.

2 Starting Up the RMC

2.1 Step-by-Step RMC Startup

Tip: Delta's SSn-PEn-BGn family of position/pressure simulators provide a simple way to test your program before connecting the module to a real system.

1. **Provide Power to the RMC Module**
   Before providing power to the RMC for the first time, disconnect all other wiring from the module. Use the provided three-position power connector to attach power. Each input is labeled on the front panel of the RMC. For details on the RMC power requirements, see RMC100 Specifications.

2. **Connect the RMC to the RMCWin software**
   Many of the steps that follow assume the RMCWin software package is used in conjunction with the RMC to set up the system. Therefore, RMCWin must first be connected with the RMC using either a serial port or Ethernet.

   - To set up your communication, see Connecting RMCWin to an RMC.

3. **Wire Each Transducer and Drive**
   Each transducer type includes a section on wiring the transducers. For each axis you will be using, refer to the appropriate section for wiring diagrams and descriptions:

   - MDT Wiring
   - Analog Transducer Wiring
   - Quadrature with Analog Output Wiring
   - Quadrature with Stepper Output Wiring
   - SSI Wiring

   Caution: Leave power to the motors and/or hydraulics disabled until instructed to power them on later in this procedure.

4. **Configure Each Transducer**
   Each transducer type includes a section on the steps necessary to configure axes of that type. For each axis you will be using, refer to the appropriate section for steps on configuring the transducer:

   - MDT Configuration
   - Analog Transducer Configuration

- Quadrature with Analog Output Configuration
- Quadrature with Stepper Output Configuration
- SSI Configuration

5. Test the Transducer and Drive Connections of Each Axis

Caution: Open loop operation, which this procedure uses, ignores all limits! Be prepared to remove drive power. Great care must be taken to avoid accidents when starting the RMC for the first time. The most common accident is a runaway, where the RMC tries to go to a position beyond the physical limits of an axis.

A. Ensure that the hydraulic or motor power is off, and all drive output connectors on the RMC are disconnected.

B. Start RMCWin and ensure it is Online with the RMC.

C. Connect the RMC100 drive output to the motor or hydraulic valve of the axis being setting up.

D. In RMCWin, with the cursor on the axis you want to adjust, press ALT+R to restore the null. Type 0 (zero) in the Command Value field of the axis, press ALT+SHIFT+O, and verify that the DRIVE for the axis is 0 (zero). If the NULL DRIVE is not zero, press ALT+N to clear it.

Note: Make sure the Simulate Bit in the Config word is off. Otherwise, Open Loop commands (ALT+SHIFT+O) will not affect output.

E. Turn on power to the motor or hydraulics for the axis being set up (the axis may drift due to valve null errors).

F. Next, we will output 500mV to the axis drive output. Type 500 in the Command Value field, press ALT+SHIFT+O, then type 0 (zero) in the Command Value field. Verify that the DRIVE for the axis is 500. The axis should extend. If the axis retracts, check the drive wiring polarity, hydraulic plumbing (if applicable), and valve null.

Note: The extend direction is defined as the direction in which the transducer counts increase. Watch the Transducer Counts field in RMCWin to see that the counts increase. On MDT axes, the extend direction is away from the head of the MDT. The retract direction is opposite from the extend direction.

G. Ensure that the Command Value field is still set to 0 (zero), and press ALT+SHIFT+O, then ALT+P to stop the axis.

H. Next, we will output -500mV to the axis drive output. Type -500 in the Command Value field, press ALT+SHIFT+O, then type 0 (zero) in the Command Value field. Verify that the DRIVE for the axis is -500. The axis should retract. If the axis extends, check the drive wiring polarity, hydraulic plumbing (if applicable), and valve null.

I. Ensure that the Command Value field is still set to 0 (zero), and press ALT+SHIFT+O, then ALT+P to stop the axis.

J. Repeat steps C through I for each axis in use.

6. Define the Position Units for Each Axis

Defining the position units is a feature often left out of other motion controllers; in those controllers, users must use raw transducer counts. Delta's motion controllers provide a conversion between raw transducer counts and user-definable position units.

Defining position units achieves the following two purposes:

- The mapping between raw transducer counts and position units is defined. This involves at least one scale term, and in some cases an offset.
- Select the range of valid 16-bit positions. Users can use unsigned positions (0-65535), signed positions (-32768 to 32767), or any other 16-bit range from -65536 to 65535.

The methods used for defining position units differ between transducer types in order to fit the interface most efficiently. Refer to the appropriate section or sections below:

- MDT Scaling
- SSI Scaling
- Quadrature with Analog Output Scaling
- Quadrature with Stepper Output Scaling
- Analog Transducer Scaling

7. Set the Extend and Retract Limits of Each Axis

Note: This step should be skipped for rotational axes, such as many applications of quadrature and stepper axes.

The RMC has software-enforced Extend and Retract Limits. This procedure describes setting these limits by moving the axis to each limit and entering the appropriate value in each field.

Repeat the following steps for each axis:

A. Using one of the following methods, move the axis to the extend limit:
   - Use a control box (diddle box) that can electrically drive the valve or motor.
   - Manually position the motor or cylinder.
   - Use the Drive Test procedure described in step 5 above.

B. In RMCWin, enter the value in the Actual Position field for the axis into the axis's Extend Limit parameter.

C. Using any of the above three methods, move the axis to the retract limit.

D. In RMCWin, enter the value in the Actual Position field for the axis into the axis's Retract Limit parameter.

8. Tune Each Axis

Note: We strongly recommend using RMCWin to configure, tune, and troubleshoot the system. Refer to Using Graphs of Axis Moves for information about creating plots of moves, which are very useful for tuning and diagnostics.

The next step is to tune the position, pressure, or speed control of each axis. Refer to the following topic for details on tuning:

- Tuning an Axis

At this point Auto Stop should be set to 0xE0E0 so any transducer error on the axis will cause it to stop, but other errors will not. Check the Status word for errors after each move.

9. Set up and Configure the Communications

Refer to one or more of the following topics for details on configuring the communications installed on your RMC module:

- Using the CPU Digital I/O
- Using the Sensor Digital I/O
- Using the Communication Digital I/O
- Using the Modicon Modbus Plus Communication Module
- Using PROFIBUS-DP
- Using the Ethernet Communication Module

10. Save Your Configuration Settings

There are several possible places to store the RMC configuration:

- **Flash Memory**

  All RMC settings can be saved in the RMC's Flash by issuing a single command. Refer to Flash Memory for details.

- **RMCWin Disk Files**

  All parameters and tables can be saved to and loaded from disk. The table editors and main screen each has a Save command under the File menu that can be used to save settings.

- **Programmable Controller Memory**

  If a PLC is used, the RMC settings can be stored in the PLC memory. This allows the RMC module to be replaced without losing parameters, provided that the PLC downloads the settings to the RMC on each power-up.

  The parameters can be uploaded and download when using PROFIBUS-DP, Ethernet, Modbus Plus, or the Communication Digital I/O in Command Mode. Refer to those sections for details on setting up the RMC from the PLC.

2.2 Setup Details

2.2.1 Scaling Overview

Scaling refers to converting the transducer feedback into meaningful units. The RMC100 uses the Scale and Offset parameters to convert the transducer Counts into measurement units (position, pressure, force). For example, the counts 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, the Scale and Offset must be calculated. RMC100 provides Scale/Offset Calibration utilities to assist you.

Calculating the Scale and Offset

The method of calculating the scale and offset parameters depends on the transducer type. See the scaling topic for the module you are using:

- MDT Scaling
- Quadrature Scaling
- SSI Scaling
- Analog Transducer Scaling
- Stepper Scaling
- Resolver Scaling

Scale/Offset Calibration Utilities

The following Scale/Offset Calibration utilities assist you in calculating the scale and offset parameters:

- Position Scale/Offset Calibration Utility
- Pressure Scale/Offset Calibration Utility
- Differential Force Scale/Offset Calibration Utility
- MDT Scale/Offset Calibration Utility
- SSI Scale/Offset Calibration Utility
- Quadrature Calibration Utility

Special Scaling Techniques

For specialized scaling techniques, see the Advanced Scaling topic. These techniques include scaling an axis such that speeds are expressed in minutes rather than seconds.

2.2.2 Advanced Scaling

This topic describes specialized scaling techniques. For general scaling information, see the Scaling Overview topic.

Scaling position so that speed is represented in feet per minute or revolutions per minute.

Use the following steps to scale the position so that the speeds can be expressed in units of feet per minute or RPM. Notice that when scaling speeds in feet per minute or RPM, the position units will not be in feet or revolutions. If you need position expressed in meaningful units, the speeds must be expressed in seconds rather than minutes.

1. Determine how many transducer counts you get in one foot or one revolution.
2. On the Tools menu, click Position Scale/Offset Calibration.
3. In the First Position section, enter 0 in the Actual Position box and enter 0 in the Counts box.
4. In the Second Position section, enter the number of counts per foot/revolution in the Counts box and then enter 60, 600 or 6000 in the Actual Position box.
   The selection of 60, 600 or 6000 depends on the resolution of the transducer and the maximum speed requirement. You must select a number that is less than the value entered in the Counts field. Since the number selected represents 60 feet/min (or 60 RPM), entering 600, for example, means you will be specifying speeds in increments of 0.1 foot/min (or 0.1 RPM). Entering 6000 would mean you will be specifying speeds in increments of 0.01 foot/min.
   Notice that a higher number in the Actual Position box will result in a higher resolution, which is usually desirable for controlling the axis.
5. The Scale/Offset Calibration Tool will then calculate the correct Scale (and Config) for the axis.

The following is another method of scaling the position so that the speeds can be expressed in units of feet per minute or RPM.

1. Find the correct Scale value to convert to either feet or revolutions and then multiply that by 0.6.
   The advantage of this method is that you start out working in units that make sense (feet or revolutions). The disadvantage is that you don’t know for sure when you finish what increments your speeds will be expressed in (0.1 feet/min or 0.01 feet/min, etc.).

2.2.3 Tuning

2.2.3.1 Tuning Overview

Once your system is set up and ready for use, it must be tuned in order to control it. The better tuned a system is, the closer the Actual Position follows the Target Position (the desired path of movement).

Tuning procedures differ depending on the type of system. This manual contains the following procedures that may work for the following systems. Please read the General Tuning Guidelines section before continuing to any of the tuning procedures.

- A Hydraulic Position Axis or Motor in Velocity Mode
- A Motor in Torque Mode
- A Position/Pressure System

Tuning Wizard

RMCWin provides a time-saving tuning wizard that calculates the gains based on plots of the motion. For details, see the Tuning Wizard topic.

General Tuning Guidelines

Keep these general 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.

- The tuning procedure is a reiteration of the following general steps. Use these steps throughout the tuning procedure:
    - **Make a move**. See the Go (G) command for making moves.
    - **View the plotted move**. See Plots for more information. Viewing the plot will help you determine which parameters must be changed.
    - **Change a parameter**. See Parameter Area for information on changing parameters in the RMC.
    - **Repeat these steps using the *same move*** until the parameter is at the desired value. See Using Stored Commands for repeating moves.
- Use the Sum of Errors Squared in the RMCWin plotting utility to determine how the last parameter change affected the system. The Sum of Errors Squared indicates 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.
- Remember, that in order to use the Sum of Errors Squared to compare moves, the moves must be identical.
- Begin the tuning procedure with long, slow moves and low Accelerations. This will prevent you from losing control of and potentially damaging the system.
- To obtain the greatest tuning precision, use the shortest Plot Time possible. This will spread the move across the entire plot window and allow you to see precisely how the system is responding.
- You may want to turn off some of the Auto Stop bits. The Auto Stop turns off the Drive 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. Remember to set these bits to either "Soft Stop" or "Hard Stop" when you have gained sufficient control of the axis. This may not be possible on some systems because of safety concerns.

- When changing the parameters, remember that they are not updated in the RMC until the Set Parameters (P) command is issued. They are not stored into the RMC Flash memory until the Update Flash command is issued.

2.2.3.2 Tuning a Position Axis

The following procedure may be used to tune many position axes, hydraulic axes and motors in velocity mode. Please read the Tuning Overview 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.

Tuning Procedure

1. **Do Open Loop Move**

   This step is for verifying that the system wiring and setup is correct before doing any closed loop control. Issue an Open Loop (O) command with a small drive, such as 50-150. Increase the drive until the axis begins to move. A positive drive should yield increasing counts. Issue an Open Loop command again with a negative drive. This should yield negative counts.

   Before continuing, verify that all the Gains and Feed Forwards are set to zero.

2. **Check Dead Band**

   If your system has a large dead band, you will need to set the Dead Band Eliminator value. To find your dead band, give increasing amounts of drive to the system with the Open Loop command. The value of drive at which the system starts to move is your dead band. If this value is approximately 400 or greater, the Dead Band Eliminator should probably be used. If it is less, it is left to the discretion of the designer.

3. **Adjust the Proportional Gain**

   The Proportional Gain must be adjusted to gain some control over the system for continuing the tuning procedure. Adjust the Proportional Gain by slowly increasing it and making moves. When the system gets close to final position reasonably quickly, continue to the next step. If the system begins to oscillate, decrease the gain.

4. **Adjust the Feed Forwards**

   In many hydraulic systems the feed forward parameters (Extend Feed Forward and Retract Feed Forward) are the most important parameters for position tracking during a move. These may be adjusted in 2 ways:

- Make a long move without any oscillation or overdrive. Then issue the Set Feed Forward command. This command will automatically adjust the Feed Forward parameter for the direction of that move.
- Set the Differential Gain and Integral Gain to zero and keep the Proportional Gain value from the previous step. Make long slow moves in both directions. Adjust the Extend Feed Forward and Retract Feed Forward until the axis tracks within 10% in both directions.

In non-regenerating hydraulic systems, the Extend Feed Forward will be less than the Retract Feed Forward. In regenerating systems, the opposite is true.

5. 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.

At this point, if you have gained sufficient control of the system, you may want to increase the speed, accel and decel of your moves and further adjust the proportional gain. A value of proportional gain that may seem good at low speeds and accels, may not work at higher speeds.

6. Adjust the Integral Gain

Many hydraulic systems do not require a large Integral Gain. However, it is usually desirable to have some Integral Gain (5 to 50 units) 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.

7. Adjust the Acceleration Feed Forwards

The Acceleration Feed Forward terms are 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.

Look for following errors during acceleration and deceleration. Increase the Extend and Retract Acceleration Feed Forward terms until the errors disappear.

For large masses the Acceleration Feed Forward may be in the tens of thousands.

8. 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.

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

If the drive output during the constant velocity portion of the move is smooth, the Differential Gain is perhaps not set high enough. The drive output may look "fuzzy." This indicates that the drive is responding to the minute errors of the axis. Note that not all systems allow the differential gain to be set high enough for the drive to be "fuzzy".

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.

9. **Increase System Speed**

Gradually increase the Speed and Acceleration values while making long moves. Look for following errors, overshoot, or oscillations.

- If an overdrive 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 adjusting the Proportional Gain.
- If the Drive is not high, the gains can probably be increased for better control. If the Drive 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 Dead Band value may need to be increased. However, if the oscillation is not caused by a deadband in the system then this 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.

2.2.3.3 Tuning a Torque Motor

The following procedure may be used to tune motors running in torque mode. Please read the Tuning Overview 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.

Tuning Procedure

1. **Do Open Loop Move**

This step is for verifying that the system wiring and setup is correct before doing any closed loop control. Issue an Open Loop (O) command with a small drive, such as 50-150. Increase the drive until the axis begins to move. A positive drive should yield increasing counts. Issue an Open Loop command again with a negative drive. This should yield negative counts.

Before continuing, verify that all the Gains and Feed Forwards are set to zero.

2. **Check Dead Band**

If your system has a large dead band, you will need to set the Dead Band Eliminator value. To find your dead band, give increasing amounts of drive to the system with the Open Loop command. The value of drive at which the system starts to move is your dead band. If this value is approximately 400 or greater, the Dead Band Eliminator should probably be used. If it is less, it is left to the discretion of the designer.

3. **Adjust the Differential Gain**

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. Do the following:

a. Set all the gains to zero. Issue an Open Loop (O) command of zero.

b. Increase the Differential Gain. Issue a Set Parameters (P) command to put the axis in closed loop control.

c. Repeat b until sufficient damping is obtained. There are several methods this can be done, depending on the system:

   - For small motors, rotate the motor manually to get a feel for the resistance to movement (damping). Repeat step b until the damping is significant. If the motor chatters or oscillates, decrease the gain.
   - For systems that cannot be moved manually, repeat step b until the motor starts humming (or chattering or oscillating) and then back the Differential Gain off significantly, perhaps even 50%, to avoid oscillating later while making moves.
   - For finer adjustment on large systems, repeat step b, and then momentarily (e.g. 5 milliseconds) give a drive output to the motor. The event step table can be used to do this with the Open Loop command. This jolt to the system will provide a clear indication of whether the Differential Gain is too high.

Do not set the Differential Gain too high! Remember that the point here is only to provide some damping for continuing the tuning process. The Differential Gain will be fine-tuned later. Keep in mind that motors often require a very high Differential Gain.

4. **Adjust the Proportional Gain**

Adding Proportional Gain will now improve system performance. Adjust the Proportional Gain by slowly increasing it and making moves. If the system begins to oscillate, decrease the gain.

5. Adjust the Feed Forwards

In torque motor applications, feed forward parameters (Extend Feed Forward and Retract Feed Forward) often do not require high values. Adjust these parameters by making a long move without any oscillation or overdrive. Then issue the Set Feed Forward command. This command will automatically adjust the Feed Forward parameter for the direction of that move.

6. Adjust the Integral Gain

The Integral Gain is helpful for maintaining position. It will not significantly affect tracking during short, fast moves. Increase the Integral Gain until the system starts oscillating, then back it down some. Motors may require a high Integral Gain.

7. Readjust the Differential Gain

Differential Gain tends to dampen out oscillations and help the axis track during acceleration and deceleration. This will positively affect short, fast moves.

Important: If you increase the Differential Gain, you may be able to increase the Proportional Gain somewhat without causing the system to oscillate.

If the drive output during the constant velocity portion of the move is smooth, the Differential Gain is perhaps not set high enough. The drive output may look "fuzzy." This indicates that the drive is responding to the minute errors of the axis. Note that not all systems allow the differential gain to be set high enough for the drive to be "fuzzy".

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.

8. Adjust the Acceleration Feed Forwards

The Acceleration Feed Forward terms help minimize errors during acceleration and deceleration. Increase the Extend and Retract Acceleration Feed Forward terms until the errors disappear.

9. Increase System Speed

Gradually increase the Speed and Acceleration values while making long moves. Look for following errors, overshoot, or oscillations.

- If an overdrive 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 Feed Forwards does not help, the Acceleration and Deceleration ramps are too steep for the response of the system.
- Should the system seem a little sloppy, try adjusting the Proportional Gain.

- If the Drive is not high, the gains can probably be increased for better control. If the Drive 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 Dead Band value may need to be increased.

The final tuning of the system should be made at the speed of intended operation.

2.2.3.4 Tuning a Position-Pressure System

The following procedure may be used to tune a system that transitions from position control to pressure control.

Please read the following topics before performing the tuning procedure:

- Tuning Overview
- Position/Pressure Overview
- Position/Pressure Setup
- Position/Pressure Example

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

1. **Tune the Position Gains**

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

2. Set Null Drive.

   When the axis is holding position (stopped) in closed loop, issue a Set Null Drive to Integral Drive (n) command. For best accuracy, this should be done after the axis has been in position for a while, such as 1 second.

   The null drive is the drive required to hold the axis in position. In some systems, such as hydraulics with servo valves, this value may change with time. Therefore, this step should be performed periodically.

   For the remainder of the tuning procedure, use an Event Step sequence as described in the Position/Pressure Setup topic and illustrated in the Position/Pressure Example.

3. Adjust the Proportional Gain

The Proportional Gain should be adjusted first to gain some control over the pressure before continuing the tuning procedure.

Note: If negative drive causes an increase in pressure, use negative values throughout the tuning procedure.

- Set the Proportional Gain to a small value, such as 2.
- Use an Event Step table that makes a position move with the Monitor Pressure bit set so the axis will transition to pressure control when the Pressure Set A threshold is crossed. Use a fairly long ramp time initially, such as 1000 msec.
- Once the axis is in pressure control, ramp the pressure between two pressures (this is best done in Event Steps). Gradually increase the proportional gain to minimize the following error. When you see a tendency to oscillate as the axis moves or stops, decrease the gain 10 to 30 percent.

4. Add Filter if Necessary

If the pressure feedback signal is excessively noisy, it may cause instability and will likely have been noticed already in the previous step. Adding a filter may help. This is done using the Filter TC parameter. See Filter TC for detailed information.

5. Adjust the Integral Gain

The Integral Gain helps get to the command pressure quickly.

- Use the same Event Step sequence as previously to ramp between two pressures. You may want to increase the delay between the steps to more clearly see the effect of this tuning step.
- Adjust the Integral Gain so that the pressure gets to commanded pressure quickly.
- Too much Integral Gain will cause oscillations and overshoot. If this happens, decrease the Integral Gain.

6. Adjust the Differential Gain

Differential Gain greatly enhances performance on many hydraulic systems. Differential Gain will tend to dampen out oscillations and help the axis track during acceleration and deceleration. This will positively affect short, fast moves.

- Continue ramping between two pressures.
- If the drive output is always smooth, the Differential Gain can probably be increased. The drive output should look "fuzzy." This indicates that the drive is correctly responding to the minute errors of the axis.

- A disadvantage of Differential Gain is that it amplifies measurement noise. If there is too much noise or the gain is too high, this can cause the system to chatter or oscillate. In this case, decrease the Differential Gain.

7. Readjust the Proportional Gain

Once the Differential Gain has been adjusted, the Proportional Gain may be readjusted. It 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.

- Continue ramping between two pressures.
- Slowly increase the gain. When you see a tendency to oscillate as the pressure changes or stops, reduce the gain by 10 to 30 percent.

8. Tune the Transition.

Now that both the position and pressure gains have been tuned, the transition may be tuned.

- Use the Event Step sequence to transition from position to pressure.
- If necessary, adjust the previously tuned pressure parameters for a smoother transition.
- If there is oscillation and overshoot or other undesirable behavior at the transition that cannot be avoided with tuning the gains, the Ramp Time may need to be changed, or the speed going into the transition may be need to be slower.
- Selecting Calculate Ramp Time in the Mode word allows the RMC to automatically calculates the Ramp Time. This is useful if it is difficult to determine a Ramp Time.
- The following three steps also address parameters affecting the transition.

9. Adjust the Feed Forwards

The Extend and Retract Feed Forwards provide extra drive when extending or retracting.

- Adjust the Feed Forwards to help the actual pressure track the target pressure when it is changing.
- If the actual pressure leads the target pressure, decrease the Feed Forwards.
- The Feed Forwards may need to change for different rates of pressure change. See Feed Forwards for more information.

10. Integrator Preload

Upon transition from position control to pressure control, the drive normally goes immediately toward zero. In some cases, this is undesirable. The integrator preload can be set to some value (positive or negative) to provide some drive immediately upon the transition. If your system overshoots or undershoots when entering pressure control, you may want to adjust this value. See Integrator Preload for more information.

Using the Integrator Preload is useful for systems with predictable position-to-pressure transitions. It always provides the same amount of drive.

11. Drive Transfer Percent

The Drive Transfer Percent acts similarly to the Integrator Preload. The difference is that the Integrator Preload places a constant value into the integral drive term, while the Drive Transfer Percent places a certain percentage (positive or negative) of the current drive into the integral drive term. If your system overshoots or undershoots when entering pressure control, you may want to adjust this value. See Drive Transfer Percent for more information.

Using the Drive Transfer Percent is useful for systems with unpredictable position-to-pressure transitions, such as entering with different speeds. Because it is a percentage, the transferred drive will vary with the drive required immediately prior.

12. Fine-tune the System

The final tuning of the system should be made at the speed of intended operation.

Look for following errors, overshoot, or oscillations.

- Should the system seem a little sloppy, try adjusting the Proportional Gain.
- If the actual position lags or leads the target position during the entire constant velocity section of the move, adjust the Feed Forwards.
- If an overdrive error occurs, there is not enough drive capacity to drive the axis at the requested rate of pressure change. Should this occur, increase the Ramp Time or decrease the speed of the system.
- If the Drive is not high, the gains can probably be increased for better control, depending on system stability.
- Adding or changing the Filter TC value may help if noisy feedback is suspected.
- Note that tuning pressure is often very difficult because of the large change in pressure for a small change in position.

2.2.3.5 Tuning Wizard: Overview

Tuning a motion system can be a challenging and time-consuming task. The Tuning Wizard simplifies and speeds the process by computing your gains for you.

To access the Tuning Wizard: On the **Tools** menu, click **Tuning Wizard**.

The Tuning Wizard uses the following steps:

1. **Build System Model**
   The wizard creates a mathematical model of your system. It uses standard RMCWin plots to

generate this model by evaluating the response of the system to changes in the Drive output.

2. **Choose Gains**
   You can then select gains appropriate to your system by using a simple slider indicating your preference for the desired responsiveness of the system from "Conservative" to "Aggressive". The tuning gains are automatically computed using this setting and the mathematical model that was created for your system.

Warning: The gains calculated by this wizard are not guaranteed to be correct or safe. Before applying auto-generated gains to an axis, you must first ensure that it is safe to move the axis and then be prepared to instantly stop all motion via an Emergency Stop button.

Requirements

To successfully use the Tuning Wizard, you must be able to perform the items listed below.

- Move the axis in both directions in open loop control, or closed loop control if you are re-tuning an axis.
- Take plots of both directions of motion. You can take these plots before using the Tuning Wizard, as the wizard will ask you to provide the plots. There are further requirements on the plots used to calculate the gains. For more details, see the Tuning Wizard: Obtaining Plots topic. Notice that for symmetrical systems, only one plot is necessary.

2.2.3.6 Tuning Wizard: Obtaining Plots

The Tuning Wizard uses plots of the axis motion to compute the system model used for calculating the gains. This topic provides requirements, tips and procedures for creating plots that yield the best chances of creating an accurate system model.

Plot Requirements

1. **Drive output must change during plot.**
   The plot must show the Drive output changing with corresponding movement on the axis.
2. **Drive output should not cross the Null Drive value.**
   The Drive output should not cross the Null Drive value, which is typically near zero. That is, the move should only be in one direction. If the system is symmetrical, this condition does not need to be met.

Plot Tips

In addition to the requirements listed above, these tips will help improve the chances of creating an accurate system model:

1. The plot should show a combination of rapid transitions and steady or smooth sections of the Drive output.
2. The amplitude changes of the Drive output should be large enough so that they cause significant motion on the axis.
3. The plot time should generally be 1 or 2 seconds. Systems with very large masses (>10,000

lbs), or with very slow response times, may require a longer plot time.

4. The Extra Plot Data (selected in Plot Options) should be set to Extra Precision, which is the default setting.
5. Avoid non-linear regions of the valve, which may be near 0V or 10V on some valves.

Procedures for Obtaining Plots

To obtain plots for the Tuning Wizard, Delta recommends using Event Step sequences to generate a series of moves. Some sample step sequences are included at the end of this topic.

If you do not yet have any gains for the axes, you can make Open Loop moves. If you already have some gains that keep the axis stable, you can make Closed Loop moves.

2.2.3.6.1 Open Loop Moves

Warning: In Open Loop, the safety features of the RMC100 are disabled. Take extra precautions to keep the axis from moving too fast or too far. Be prepared to stop the axis immediately with an emergency stop switch.

The sequence of steps starting at step 1 of the sample step sequence below generates an output with various ramps and small steps which should move the axis in the positive direction. The motion will be somewhat jerky due to the step changes in output voltage. The sequence can be changed to achieve slower voltage changes for systems with large masses.

The sequence of steps starting at step 20 should move the axis in the negative direction.

2.2.3.6.2 Closed Loop Moves

Note: Some of the safety features of closed loop moves may stop the axis before it has moved far enough to obtain sufficient data for tuning. You may need to temporarily disable some of the Auto Stops or increase the Following Error window. Use caution when moving the axis with disabled the Auto Stops. Remember to set the Auto Stops to a safe setting after you have completed tuning the axis.

The step sequences starting at steps 40 and 50 are examples of moves that can be used to generate plots for the tuning wizard in the positive and negative directions respectively.

NOTE: You will need to edit the positions and speeds to make the sample step sequence work for your axis.

Steps to Create Open Loop Motion and Capture a Plot

Follow these steps to capture a plot containing sufficient data to build a model:

1. In RMCWin, on the Tools menu, click Event Steps Editor.
2. Enter an event step sequence in the Event Steps Editor. Use the sample sequences shown in this topic. You may need to modify them to suit your system.

Note: The sample step sequence shown in this topic, TuneWizDemo.st1, is located in the same folder as RMCWin, which is by default C:\Program Files\RMCWin\.

3. Download the Step Table to the RMC100 by clicking the download button on the Event Step Editor toolbar ( ).

4. To start the open loop step sequence in the positive direction on the axis:

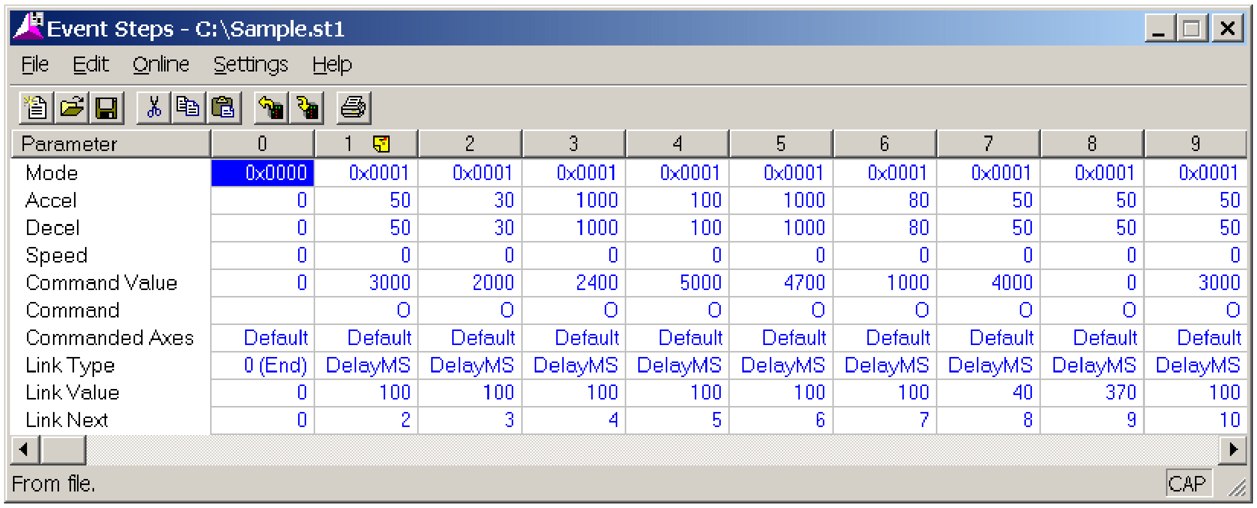
    - In the Command area in RMCWin, type "1" in the Command Value of the axis, then issue an E command to that axis.
      WARNING: This step will cause the axis to move. Be prepared to stop the axis with an emergency stop switch.
    - Open the plot of the motion by clicking the Plot Selected Axis button on the RMCWin toolbar ( ).
    - Once the plot is finished uploading, save the plot and give it a meaningful name so that you know which direction of motion it contains, for example "ExtendOL.plt"

5. To start the open loop step sequence in the other direction on the axis:

    - In the Command area in RMCWin, type "20" in the Command Value of the axis, then issue an E command to that axis.
    - Open the plot of the motion by clicking the Plot Selected Axis button on the RMCWin toolbar ( ).
    - Once the plot is finished uploading, save the plot and give it a meaningful name so that you know which direction of motion it contains, for example "RetractOL.plt"

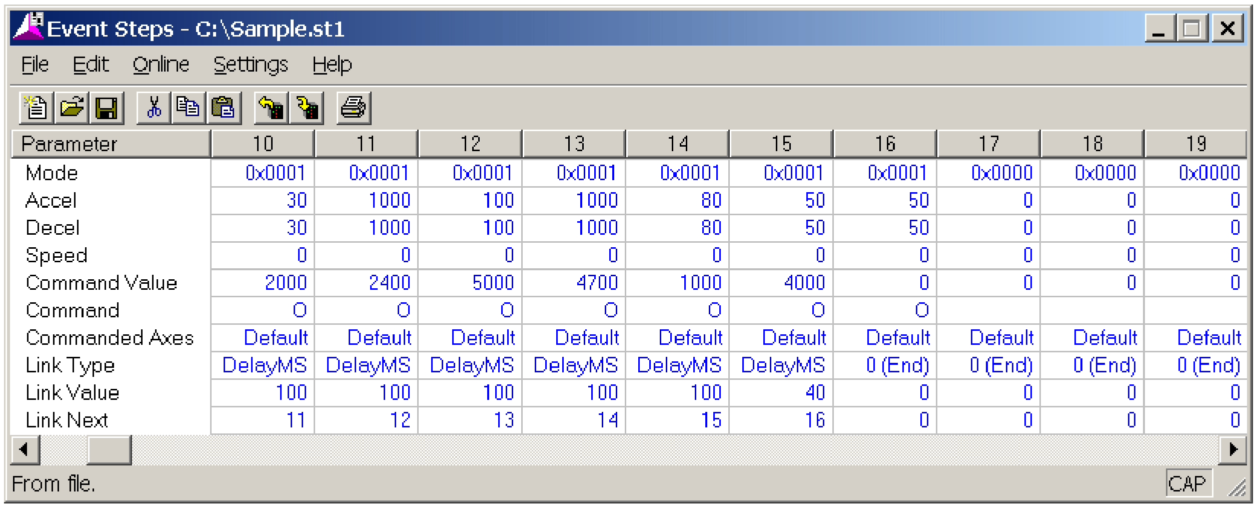
6. On the **Tools** menu, click **Tuning Wizard**. Follow the steps of the wizard. If you have a standard single-rod cylinder, be sure to select "My System is Asymmetrical". When asked to build a model from a plot, use the plots you have saved.

Sample Step Sequences

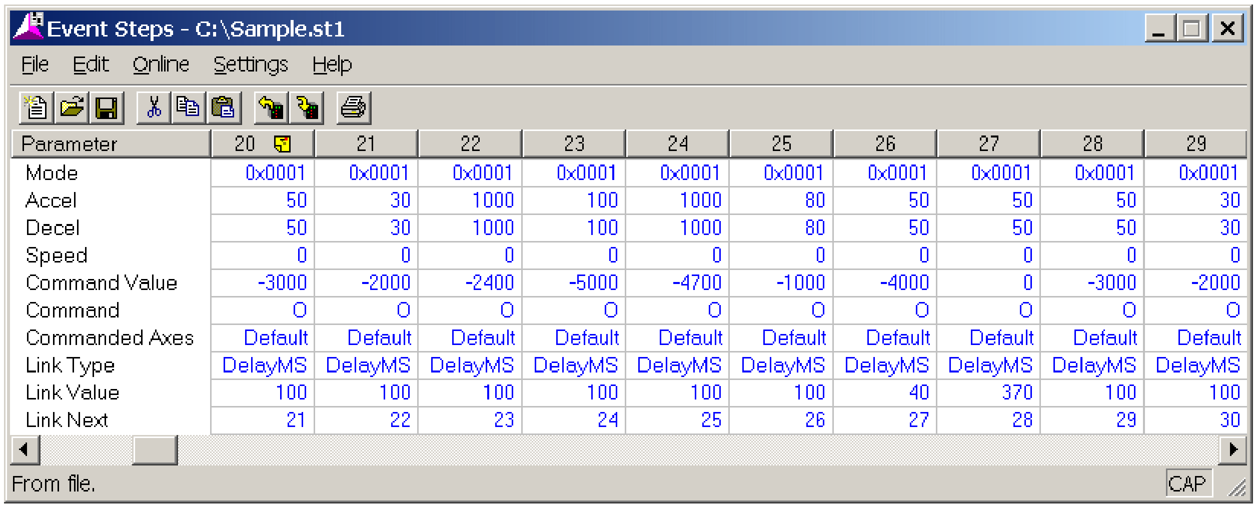


Parameter	0	1	2	3	4	5	6	7	8	9
Mode	0x0000	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001
Accel	0	50	30	1000	100	1000	80	50	50	50
Decel	0	50	30	1000	100	1000	80	50	50	50
Speed	0	0	0	0	0	0	0	0	0	0
Command Value	0	3000	2000	2400	5000	4700	1000	4000	0	3000
Command		O	O	O	O	O	O	O	O	O
Commanded Axes	Default	Default	Default	Default	Default	Default	Default	Default	Default	Default
Link Type	0 (End)	DelayMS	DelayMS	DelayMS	DelayMS	DelayMS	DelayMS	DelayMS	DelayMS	DelayMS
Link Value	0	100	100	100	100	100	100	40	370	100
Link Next	0	2	3	4	5	6	7	8	9	10

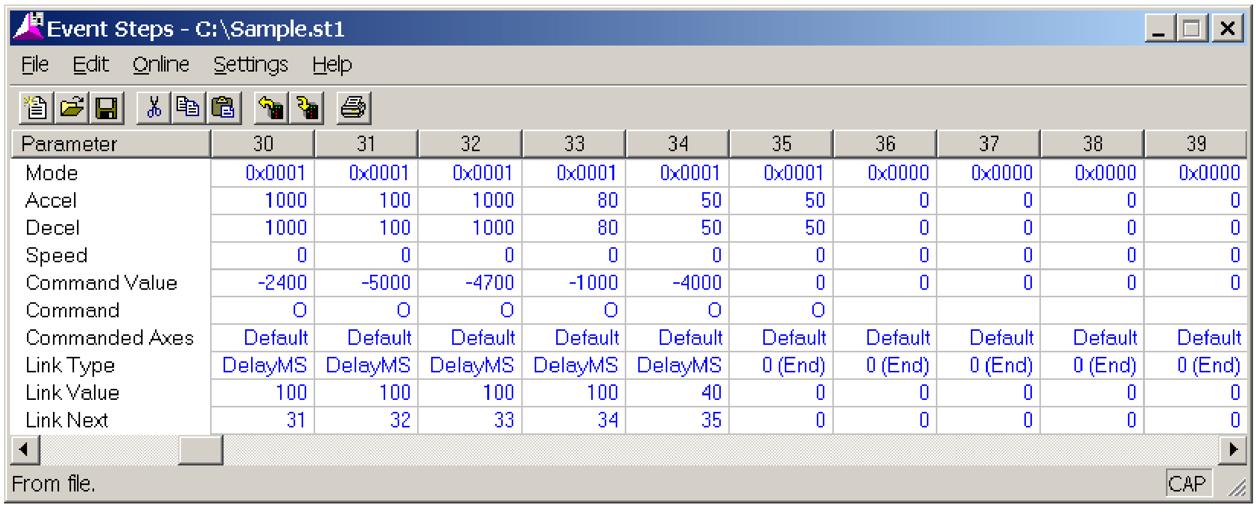
From file.



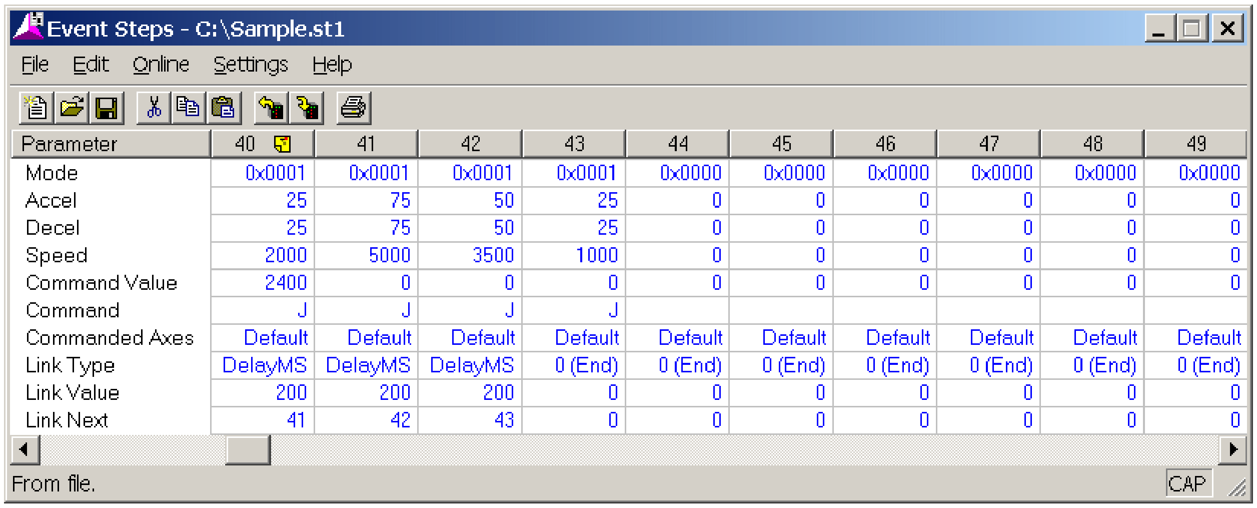
Parameter	10	11	12	13	14	15	16	17	18	19
Mode	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001	0x0000	0x0000	0x0000
Accel	30	1000	100	1000	80	50	50	0	0	0
Decel	30	1000	100	1000	80	50	50	0	0	0
Speed	0	0	0	0	0	0	0	0	0	0
Command Value	2000	2400	5000	4700	1000	4000	0	0	0	0
Command	O	O	O	O	O	O	O			
Commanded Axes	Default	Default	Default	Default	Default	Default	Default	Default	Default	Default
Link Type	DelayMS	DelayMS	DelayMS	DelayMS	DelayMS	DelayMS	0 (End)	0 (End)	0 (End)	0 (End)
Link Value	100	100	100	100	100	40	0	0	0	0
Link Next	11	12	13	14	15	16	0	0	0	0



Parameter	20	21	22	23	24	25	26	27	28	29
Mode	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001
Accel	50	30	1000	100	1000	80	50	50	50	30
Decel	50	30	1000	100	1000	80	50	50	50	30
Speed	0	0	0	0	0	0	0	0	0	0
Command Value	-3000	-2000	-2400	-5000	-4700	-1000	-4000	0	-3000	-2000
Command	O	O	O	O	O	O	O	O	O	O
Commanded Axes	Default	Default	Default	Default	Default	Default	Default	Default	Default	Default
Link Type	DelayMS	DelayMS	DelayMS	DelayMS	DelayMS	DelayMS	DelayMS	DelayMS	DelayMS	DelayMS
Link Value	100	100	100	100	100	100	40	370	100	100
Link Next	21	22	23	24	25	26	27	28	29	30



Parameter	30	31	32	33	34	35	36	37	38	39
Mode	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001	0x0000	0x0000	0x0000	0x0000
Accel	1000	100	1000	80	50	50	0	0	0	0
Decel	1000	100	1000	80	50	50	0	0	0	0
Speed	0	0	0	0	0	0	0	0	0	0
Command Value	-2400	-5000	-4700	-1000	-4000	0	0	0	0	0
Command	O	O	O	O	O	O				
Commanded Axes	Default	Default	Default	Default	Default	Default	Default	Default	Default	Default
Link Type	DelayMS	DelayMS	DelayMS	DelayMS	DelayMS	0 (End)	0 (End)	0 (End)	0 (End)	0 (End)
Link Value	100	100	100	100	40	0	0	0	0	0
Link Next	31	32	33	34	35	0	0	0	0	0

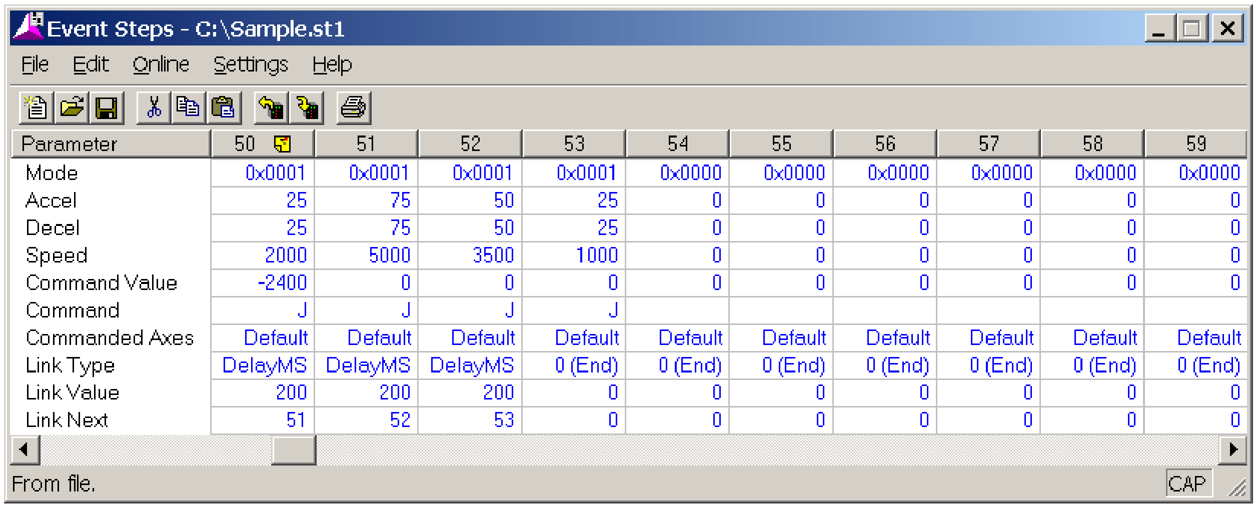


Event Steps - C:\Sample.st1

File Edit Online Settings Help

Parameter	40	41	42	43	44	45	46	47	48	49
Mode	0x0001	0x0001	0x0001	0x0001	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
Accel	25	75	50	25	0	0	0	0	0	0
Decel	25	75	50	25	0	0	0	0	0	0
Speed	2000	5000	3500	1000	0	0	0	0	0	0
Command Value	2400	0	0	0	0	0	0	0	0	0
Command	J	J	J	J						
Commanded Axes	Default	Default	Default	Default	Default	Default	Default	Default	Default	Default
Link Type	DelayMS	DelayMS	DelayMS	0 (End)	0 (End)	0 (End)	0 (End)	0 (End)	0 (End)	0 (End)
Link Value	200	200	200	0	0	0	0	0	0	0
Link Next	41	42	43	0	0	0	0	0	0	0

From file. CAP



Event Steps - C:\Sample.st1

File Edit Online Settings Help

Parameter	50	51	52	53	54	55	56	57	58	59
Mode	0x0001	0x0001	0x0001	0x0001	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
Accel	25	75	50	25	0	0	0	0	0	0
Decel	25	75	50	25	0	0	0	0	0	0
Speed	2000	5000	3500	1000	0	0	0	0	0	0
Command Value	-2400	0	0	0	0	0	0	0	0	0
Command	J	J	J	J						
Commanded Axes	Default	Default	Default	Default	Default	Default	Default	Default	Default	Default
Link Type	DelayMS	DelayMS	DelayMS	0 (End)	0 (End)	0 (End)	0 (End)	0 (End)	0 (End)	0 (End)
Link Value	200	200	200	0	0	0	0	0	0	0
Link Next	51	52	53	0	0	0	0	0	0	0

From file. CAP

3 Using RMCWin

3.1 RMCWin Overview

Description

RMCWin is a Windows 98/NT/2000/XP/Vista/7 based software package that allows you to access, display, troubleshoot, configure and control features of Delta's RMC motion control products. RMCWin allows you to adjust the parameters of the RMC and make simple movements. You can display a motion trajectory using RMCWin's graphing capability.

Connecting to the RMC

- Connecting RMCWin to an RMC
- Using the Communication Options Tab
- Working Offline
- Detecting Configuration Conflicts

Basic Topics

- Understanding the Screen
- Selecting Your View
- Accessing Context Sensitive Help
- Changing Data from the Keyboard
- Using Read-back and Write Modes
- Using Pop-up Editors
- Using the Status Bits Window
- Using the Command Log
- Using the Parameter Error List Window
- Using the I/O Bit Monitor
- Using Stored Commands
- Using Graphs of Axis Moves
- Changing the Axis Names

- Using Multiple RMCs
- Using the Scale/Offset Calibration Utilities

Table Editors

- Table Editor Basics
- Editing the Stored Command Table
- Editing the Profile Table
- Editing the Event Step Table
- Editing the Input to Event Table

Tools

- LCD Screen Editor
- Curve Tool
- Address Tool

Advanced Topics

- Downloading New Firmware
- Forcing Initialization
- Using Look-only Mode
- Using PC Mode
- Using Command-Line Options

3.2 Screen Layout

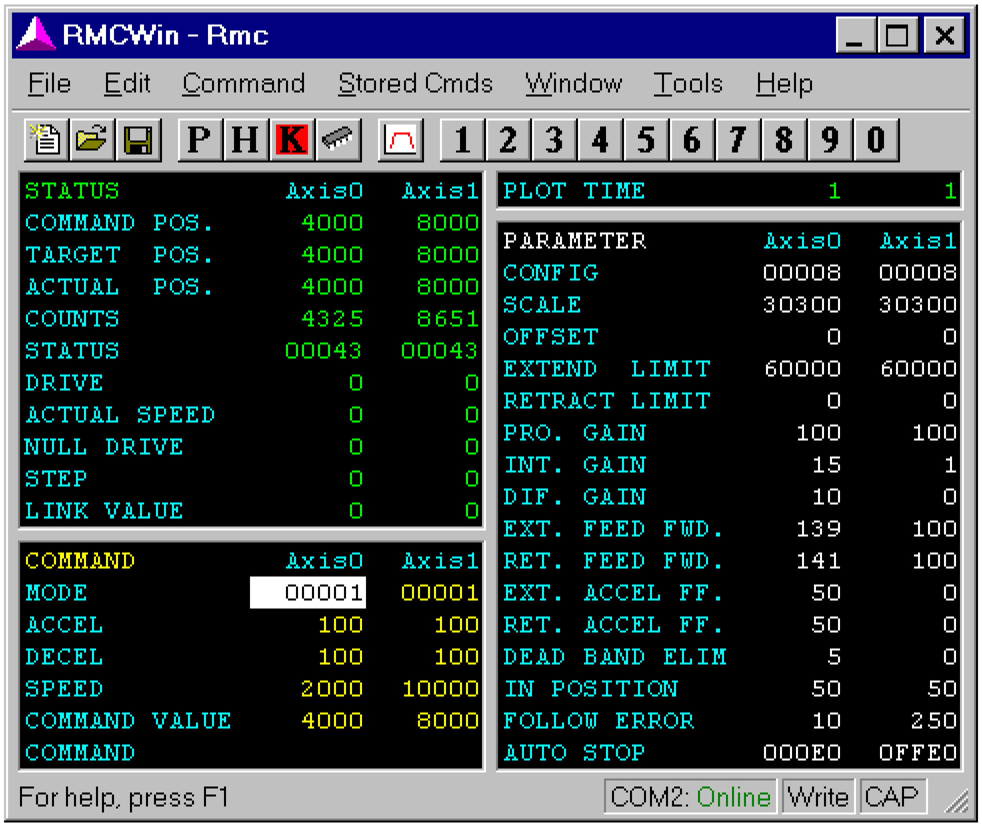
3.2.1 Understanding the Screen

The main window in RMCWin display several kinds of information. It can be divided into six sections:

- Toolbar
- Status Bar
- Status area (top-left pane)
- Command area (bottom-left pane)

- Plot Time area (top-right pane)
- Parameter area (bottom-right pane)

The following is a sample main screen:


RMCWin - Rmc
File Edit Command Stored Cmds Window Tools Help
P H K 1 2 3 4 5 6 7 8 9 0
For help, press F1
COM2: Online
Write
CAP

STATUS	Axis0	Axis1
COMMAND POS.	4000	8000
TARGET POS.	4000	8000
ACTUAL POS.	4000	8000
COUNTS	4325	8651
STATUS	00043	00043
DRIVE	0	0
ACTUAL SPEED	0	0
NULL DRIVE	0	0
STEP	0	0
LINK VALUE	0	0

COMMAND	Axis0	Axis1
MODE	00001	00001
ACCEL	100	100
DECEL	100	100
SPEED	2000	10000
COMMAND VALUE	4000	8000
COMMAND		

PLOT TIME	1	1

PARAMETER	Axis0	Axis1
CONFIG	00008	00008
SCALE	30300	30300
OFFSET	0	0
EXTEND LIMIT	60000	60000
RETRACT LIMIT	0	0
PRO. GAIN	100	100
INT. GAIN	15	1
DIF. GAIN	10	0
EXT. FEED FWD.	139	100
RET. FEED FWD.	141	100
EXT. ACCEL FF.	50	0
RET. ACCEL FF.	50	0
DEAD BAND ELIM	5	0
IN POSITION	50	50
FOLLOW ERROR	10	250
AUTO STOP	000E0	0FFE0

3.2.2 Command Area

This area is located in the lower left portion of the main window. It holds the Command fields for each axis. This area is updated only in Read-back Mode. Refer to the following sections for details on the command fields:

Mode

Acceleration

Deceleration

Speed

Command Value

Command

To the right of each of the Command field labels are the values for the command fields for each axis. You can change these values using the keyboard as described in Changing Data from the Keyboard. Changes are not sent to the RMC until the COMMAND field itself is changed. At this time, all six command fields are sent to the controller.

For details on saving and loading commands, see Changing Between Board Files.

3.2.3 Parameter Area

This area is located in the lower right portion of the main window. It holds the Parameter fields for each axis. This area is updated only in Read-back Mode. Refer to the following sections for details on the parameter fields:

Configuration Word

Scale

Offset

Extend Limit

Retract Limit

Proportional Gain

Integral Gain

Differential Gain

Extend Feed Forward

Retract Feed Forward

Extend Acceleration Feed Forward

Retract Acceleration Feed Forward

Dead Band Eliminator

In Position

Following Error

Auto Stop

To the right of each of the Parameter field labels are the values for the parameter fields for each axis. You can change these values using the keyboard as described in Changing Data from the Keyboard. Changes are sent to the RMC only when a Set Parameters Command is issued. There are several ways to send this command:

- Select a field in the axis you want initialized and press ALT+P.
- Select a field in the axis you want initialized and from the **Command** menu, click **P-Set Parameters**.
- Select the COMMAND field in the axis you want initialized, type P, and then press ENTER.

Note: The color of the Parameter values is significant. If all parameters for an axis are displayed in WHITE, then no parameters have been changed since the last time they were sent to the RMC, otherwise they are displayed in RED. This is done to remind the user to send a Set Parameters Command after updating a parameter. After this command is issued to an axis, the parameters for that axis will be displayed in WHITE.

Note: When in Read-back Mode, you will notice that RED parameters will be replaced with WHITE parameters as the current values are read from the RMC. This is done to indicate that the values displayed match those used by the RMC.

For details on saving and loading parameters, see Changing Between Board Files.

3.2.4 Plot Time Area

This area is located in the top right portion of the main window. It holds the Plot Time field for each axis. Refer to this topic for details on its use. This area is updated on startup and each time RMCWin connects to a new RMC.

To the right of the PLOT TIME heading is a column for each axis. You can change these values using the keyboard as described in Changing Data from the Keyboard. Changes are sent to the RMC immediately.

3.2.5 Status Area

This area is located in the upper left portion of the main window. It holds the Status fields for each axis. This area is updated constantly when an RMC is connected to RMCWin. Refer to the following sections for details on these parameters:

Command Position

Target Position

Actual Position

Transducer Counts

Axis Status Word

Drive

Actual Speed

Null Drive

Step

Link Value

To the right of each of the Status field labels are the values for the status fields for each axis. These values cannot be changed.

3.2.6 Status Bar

The status bar is located at the bottom of the main screen. This bar is divided into four areas:

Menu Help	All of the status bar except the three panes described below is used to display help on menu items. When no menu item is selected, it displays "For help, press F1." If a menu item is selected, or the cursor is over a toolbar button, then a brief line of help is displayed here.
Communication	This pane displays the current communication path and its state. Example communication paths include "COM1," "COM2," and "192.168.0.23". Double-click this pane to display the **Communication** tab of the **Options** dialog box, and right-click this pane to display a shortcut menu with many common communication-related commands. See Connecting RMCWin to an RMC for details. The communication path can be in one of the following states:
Closed	RMCWin is not connected to an RMC, and it is not using the communication path at all.
Offline	RMCWin is not connected to an RMC, but it is polling the communication path for an RMC to become available.
Connecting	RMCWin detected an RMC, but is in the process of connecting and validating the RMC configuration.
Online	RMCWin is currently connected to an RMC.
Loader	RMCWin is currently connected to an RMC, but the RMC is running its loader firmware. The only operation allowed in the loader is downloading new firmware. See Downloading New Firmware for details on updating the firmware.
Read/Write	This pane indicates whether the Command and Parameter areas of the main display are in read-back or write mode. Double-clicking this pane will toggle between Read-back and Write modes. Refer to Read-back versus Write Mode for a details on these modes.
CAP	This pane indicates whether the CAPS LOCK key is toggled on or off. If this pane is blank, then CAPS LOCK is not enabled. Otherwise, it will display CAP.

3.2.7 Toolbar

The follow buttons are available on the toolbar:

Note: In the descriptions below, the term current axis refers to the axis under which the currently selected cell on the main screen is visible.

	New	Creates a new board file with default parameters. Refer to Using Multiple Motion Modules for details on board files.
	Open	Opens a different board file. Refer to Using Multiple Motion Modules for details on board files.
	Save	Saves the current board file. Refer to Using Multiple Motion Modules for details on board files.
P	Set Parameters	Sends the parameters to the board for the current axis and issues a Set Parameters (P) command.
H	Halt	Issues a Halt (H) command to the current axis.
K	Kill	Issues a Disable Drive Output (K) command to all axes.
	Plot	Displays the plot for the current axis.
	Toggle Displayed Fields	Switches between displaying status and command fields and displaying plot time and parameter fields. This button is only available if the **Half View** is selected; see Selecting Your View for details on this mode.
	Save to Flash	Issues an Update Flash command, which saves all axis parameters, tables, and configuration to the Flash.
1 … 0	Stored Commands	Issues a stored command to the current axis. This is equivalent to holding CTRL and pressing a number key. Also, if the user holds down the ALT key while pressing one of these buttons, the full profile stored command is executed. This is equivalent to holding ALT and pressing a number key. See Using Stored Commands for further details.

3.3 Connecting to an RMC

3.3.1 Connecting RMCWin to an RMC

Communication Driver Status

The main window's status bar has a pane called the **Communication** pane. This pane shows the current communication path and the state of that communication path (for example, "COM1: Offline"). A path can be in any of the following states:

State	Description
Closed	RMCWin is not connected to an RMC, and it is not using the communication path at all.
Offline	RMCWin is not connected to an RMC, but it is polling the communication path for an RMC to become available.
Connecting	RMCWin detected an RMC, but is in the process of connecting and validating the RMC configuration.
Online	RMCWin is currently connected to an RMC.
Loader	RMCWin is currently connected to an RMC, but the RMC is running its loader firmware. The only operation allowed in the loader is downloading new firmware. See Downloading New RMC100 Firmware for details on updating the firmware.

Picking the Right Communication Driver

RMCWin offers numerous ways to connect to an RMC. All of these ways use one of three drivers, each of which is listed below. Click the link following each for a detailed description and wiring diagrams:

- **Serial:** This is the easiest method to use, but—unless an RS422/485 segment is inserted into the cable—the maximum cable length is typically limited to 50 feet and there is no isolation. In either case, only a single RMC can be accessed per serial port.
  See Communication Drivers: Serial Overview.

- **TCP/IP Direct to RMC-ENET:** This method requires an RMC ENET module, but otherwise gives the best performance, allows routing and addressing multiple modules, and provides isolation.
  See Communication Driver: TCP/IP Direct to RMC-ENET Overview.

Note: If your computer has a firewall, it may prevent RMCWin from finding RMC100s on your Ethernet network from the Communications tab in the Options dialog. See the Setting the Firewall to Allow RMC100 Ethernet Browsing topic for details.

- **TCP/IP-to-RS232 Bridge:** This method adds many of the benefits of Ethernet—it allows routing and addressing multiple modules and provides isolation—but at the expense of performance, it can work with all RMCs, not just those with the RMC ENET module. This is done by using a TCP/IP-to-RS232 Bridge device.
  See Communication Driver: TCP/IP-to-RS232 Bridge Overview.

Selecting and Configuring a Communication Driver

The first way to select or configure a communication driver is to use the **Communication** tab of the **Options** dialog box.

To use the Communication tab:

1. Start RMCWin.
2. On the **Tools** menu, click **Options**, and then click the **Communication** tab.
   You can also double-click the **Communication** pane of the main window's status bar, or right-click this pane and then click **Communication Options** from the shortcut menu.
3. Select the appropriate driver and options. See Using the Communication Options Tab for details.
4. Click **OK**.

Another way to change some of the communication options is to use the **Communication** pane shortcut menu. Options in this menu include:

- Open or close the communication path to switch between online and offline.
- Switch the communication path to any available serial port.
- Switch the communication path to any IP address recently used by the TCP/IP Direct to RMC-ENET driver or IP address/port pair recently used by the TCP/IP-to-RS232 Bridge driver.
- Switch the communication path to any RMC detected on the LAN. This provides a quick way to detect any RMCs on the network.

To use the **Communication** pane shortcut menu:

1. Start RMCWin.
2. On the main window's status bar, right-click the **Communication** pane.
   A shortcut menu will appear with the options described above.
3. In the shortcut menu, click the option you want to use.

3.3.2 Setting the Firewall to Allow RMC100 Ethernet Browsing

This topic is pertinent to all computers with a firewall, especially Windows XP SP2, which includes a firewall.

If you computer has a firewall, it may prevent RMCWin from finding RMC100s when browsing the Ethernet network from the Communication Options Tab. This problem will appear with Windows XP Service Pack 2 since it includes a Firewall with default settings that block the RMCWin Ethernet browsing. Other firewalls may cause the same problem. If your RMC is behind a firewall, make sure the firewall forwards port 1324 to the RMC’s IP address.

There are two methods of fixing this problem in Windows XP:

- Create an Exception for RMCWin
  This method leaves the Firewall enabled, and creates an exception only for RMCWin.
- Disable the Firewall
  This method is the easiest, but it may be undesirable since it may make your computer vulnerable to viruses and intruders.

The following instructions describe how to perform these two tasks on Windows XP:

Create an Exception

1. On the Start menu, click Control Panel.
2. Double-click Windows Firewall, and click the Exceptions tab.
3. Click Add Program, then choose RMCWin from the Programs list. If RMCWin is not listed, browse to the RMCWin.exe file.
4. Click OK to close the Add a Program dialog, then click OK to close the Windows Firewall dialog. The change takes effect immediately.

Disable the Firewall

1. On the Start menu, click Control Panel.
2. Double-click Windows Firewall.
3. Click Off, then OK. The change take effect immediately.

The methods for other firewalls vary.

3.3.3 Using the Communication Options Tab

The **Communication** tab of the **Options** dialog box is used to select and configure a communication path to an RMC.

To use the **Communication** tab:

1. Start RMCWin.
2. On the **Tools** menu, click **Options**, and then click the **Communication** tab.
   You can also double-click the **Communication** pane of the main window's status bar, or right-click this pane and then click **Communication** Options from the shortcut menu. The Communication pane is the one that displays the communication path and state (e.g. "COM1: Offline").
3. Select the appropriate driver and options.
4. Click OK.

Note: Many options available in the **Communication** tab of the **Options** dialog box can be changed more quickly using the shortcut menu on the Communication pane of the status bar. To open this shortcut menu, right-click on the **Communication** pane of the status bar (e.g. "COM1: Offline").

RMCWin can use one of three drivers for communicating between the PC and RMC. Each is described below:

- **Serial:** This is the easiest method to use, but—unless an RS422/485 segment is inserted into the cable—the maximum cable length is typically limited to 50 feet and there is no isolation. In either case, only a single RMC can be accessed per serial port. See Communication Driver: Serial Configuration for details on settings specific to this driver.

- **TCP/IP Direct to RMC-ENET:** This method requires an RMC ENET module, but otherwise gives the best performance, allows routing and addressing multiple modules, and provides isolation. See Communication Driver: TCP/IP Direct to RMC-ENET Configuration for details on settings specific to this driver.

Note: If your computer has a firewall, it may prevent RMCWin from finding RMC100s on your Ethernet network from the Communications tab in the Options dialog. See the Setting the Firewall to Allow RMC100 Ethernet Browsing topic for details.

- **TCP/IP-to-RS232 Bridge:** This method adds many of the benefits of Ethernet—it allows routing and addressing multiple modules and provides isolation—but at the expense of performance, can work on all RMCs, not just those with the RMC ENET module. It does this by using a TCP/IP-to-RS232 Bridge device. See Communication Driver: TCP/IP-to-RS232 Bridge Configuration for details on settings specific to this driver.

In addition to the driver selection and driver-specific options, there is one other option offered by this dialog box:

- **Closed (Work Offline) check box**
  Check this box to close the current communication path and work offline. When this mode is selected, no attempt will be made to connect to an RMC over the communication path. If a serial port had been open, it will be closed and therefore available for use by other applications. RMCWin will always start by trying to open the communication path, unless the **-P** command-line option is used; see Command- Line Options for details.

3.3.4 Working Offline

Working offline refers to using RMCWin without being connected to an RMC. There are two communication states that can be used to work offline: **Closed** and **Offline**. In both states, RMCWin is not connected to an RMC. However, in the **Closed** state, RMCWin is not using the communication path at all, whereas in the **Offline** state, RMCWin is polling the communication path for an RMC to become available.

RMCWin will switch automatically between **Offline** and **Online** as an RMC becomes available or is removed from the communication path. However, RMCWin can be manually placed in the **Closed** state.

To work offline, you can do either of the following:

- Leave the communication path open, but ensure that RMCWin cannot connect to the RMC. The simplest way to ensure this is to physically disconnect the RMC from the PC by removing the serial or Ethernet (CAT5) cable.

- Close the communication path. This is described below under **Opening and Closing a Communication Path**.

Opening and Closing a Communication Path

There are three ways to open and close a communication path. Each is described below:

To use the **Communication** tab in the **Options** dialog box:

1. Start RMCWin.
2. On the **Tools** menu, click **Options**, and then click the **Communication** tab.
   You can also double-click the **Communication** pane of the main window's status bar, or right-click this pane and then click **Communication Options** from the shortcut menu.
3. Click to select or clear the **Closed (Work Offline)** check box.
4. Click OK.

To use the **Communication** pane shortcut menu:

1. Start RMCWin.
2. On the main window's status bar, right-click the **Communication** pane.
   A shortcut menu will appear with the options described below.
3. On the shortcut menu, click **Close** to close the communication path, or **Open** to re-open the path.

To use a command-line option:

1. Start RMCWin with the -P command-line option (e.g. rmcwin -P). See Command- Line Options for details.
   RMCWin will start with the communication path closed. Otherwise, RMCWin always starts in the **Offline** state, and therefore scans the communication path for an RMC.

Simulating a RMC

When RMCWin connects to an RMC, it reads the physical hardware configuration and uses that configuration. However, if you are working offline, you will need a way to set up a hardware configuration to work with. The **Simulate** command offers that capability.

To simulate a RMC:

1. On the **Tools** menu, click **Module Configuration**.
2. Click **Simulate**. This command is only available when working offline.
3. You now have two options for selecting the RMC hardware configuration:

- If you know the part number of the RMC you want to simulate, then type the part number in the RMC Part Number text box. If the entry is valid, the image will change to reflect the module being simulated. The text will be red if the part number is invalid.
- You can also use the **Hardware Options** image and the **Pressure/Force Control Firmware** check box to create the RMC you want to simulate. There are three ways to edit the image:
  - o Right-click on a slot in the image, and select a module to display in that slot from the shortcut menu.
  - o Select a slot and use the UP and DOWN ARROW KEYS to switch between available

options. You can select a slot by clicking on it, or using the LEFT and RIGHT ARROW KEYS to switch between slots.

- Select a slot and press the shortcut key for the type of module you want to display in that slot. To find out what shortcut keys are available, display the shortcut menu for the slot and look at the underlined letter for each option. Press DELETE to remove a module from a slot.

Notice that the order of the slots is automatically updated to match how the configuration is shipped from Delta.

4. Click **OK**.

3.3.5 Configuration Conflict Detection

This feature simplifies replacing and cloning modules. The configuration between the currently connected module and the current board file are compared at the following times:

- When a board file is opened while a module is connected.
- When a new module is connected to RMCWin.

The following items are compared:

- All physical hardware (e.g. PROFIBUS-DP, RMC100 CPU, and two MDTs).
- All hardware configuration options. This includes all options editable from the **RMC Configuration** dialog box and none of the axis parameters. Examples of hardware configuration options include PROFIBUS-DP station address, analog channel assignments, and MDT blanking period settings.

Note: The parameters, event step table, input-to-event table, motion profile table, and status map tables in the module are not compared.

If there is a conflict when a new connection is made with an RMC, then the current board file is closed. The user is given the option of saving any changes, if there were changes. The data is then read from the RMC into an untitled board file. You may then either use the newly created board file or attempt opening another board file.

If the conflict occurs when you are already connected to an RMC and attempt to open a board file, then the **Resolve Configuration Conflict** dialog box is displayed. See Resolve Configuration Conflict Dialog Box for details.

To disable this feature:

1. On the **Tools** menu, click **Options**, and then click the **Preferences** tab.
2. Select the **Ignore configuration conflicts** check box.

3.3.6 Resolve Configuration Conflict Dialog Box

This dialog is displayed when the Configuration Conflict Detection feature determines that the configuration of a board file being opened does not match with the currently connected RMC. This feature can be disabled as described in Configuration Conflict Detection.

Note: If the connection to the RMC is lost while this dialog box is displayed, the dialog box will be removed. This is to allow for cases where you realize the board file belongs to another RMC module.

RMC Configuration List

This box lists all of the hardware currently installed in the RMC module and the associated configuration values of each piece of hardware. You may have to expand branches of the tree to find the conflict.

PC Configuration File List

This box lists the hardware and its associated configuration that is stored in the current board file. Compare this list with the RMC Configuration list to determine which configuration is desired.

There are three commands available:

- **Send configuration/parameters to RMC and save to Flash**

  This command is available only if the hardware in the RMC module matches the board file hardware, as RMCWin obviously cannot change the hardware. This command will download the configuration and parameters stored in the board file to the RMC module, store the configuration in the Flash memory, and reset the RMC to ensure that all configuration changes take effect.

- **Open but retain RMC configuration**

  This command reads the parameters, plot times, and axis names from the board file, but ignores the hardware configuration. That is, this command does not change the hardware configuration of the RMC. This changes RMCWin’s copy of the board file, so the filename will be marked as changed with an asterisk, and you will be prompted to save your changes if you close the file. However, the board file itself will not be modified unless you do choose to save it.

  If you wish to only load the parameters from a board file into your current board file and thereby retaining the filename of your current board file, click **Cancel**, and instead from the **File** menu, click **Load Parameters**. This will modify your current board file by overlaying the parameters from the selected board file. It still will not commit the changes to disk until you choose to save it.

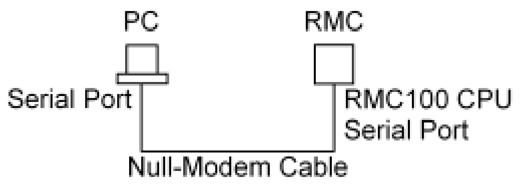
- **Cancel**

  By clicking this button, the operation to open a board file is canceled. Pressing ESC is equivalent to clicking this button.

3.3.7 Communication Drivers

3.3.7.1 Communication Driver: Serial Overview

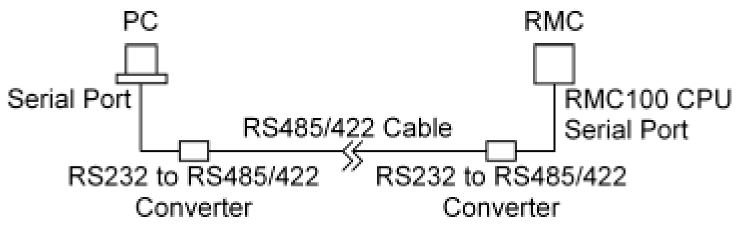
The simplest way to use RS232 for connecting RMCWin and an RMC is with a single null-modem RS232 cable connected between a PC's serial port and the RMC's "RS-232 Monitor" serial port, as shown below:



The RMC's serial port has a DTE DB9 serial connector. For further wiring details, see RS232 Wiring.

Note: RMCWin and the RMC have been tested with USB-based serial ports. In the best case, the USB-based serial ports were twice as slow as a standard serial port. One of the USB-based serial ports we tested with also did not work with Windows 2000. This problem was isolated to the USB-based serial port device driver, and therefore was not the fault of RMCWin, the RMC, the PC, nor Windows 2000. Therefore, take care in selecting a USB-based serial port. The Xircom PGSDB9 had consistent, bug-free performance on all Windows platforms, although it was three times slower than a standard serial port.

Using a simple RS232 cable has the disadvantages of no isolation on the RMC end (and typically no isolation on the PC either), and a typical cable length limitation of 50 feet. To increase the length up to 4000 ft, an RS422/485 segment can be added using two RS232-to-RS422/485 converters. Isolation can also be included in the converters:



Both methods still have the disadvantage of only being able to address a single RMC from each PC serial port. Therefore, it may be necessary to add more serial ports to a PC or switch cables to communicate with more than one RMC from a PC.

Comparison with Other Communication Drivers

- TCP/IP Direct to RMC-ENET: The direct TCP/IP method only works with an RMC-ENET and does require an Ethernet adapter in the PC, but it adds higher performance, isolation, longer cable distances, routing across intranets and the Internet, and addressing of unlimited RMCs from a single PC.
- TCP/IP-to-RS232 Bridge: The bridge method is the slowest and requires an Ethernet adapter in the PC, but it adds isolation in the bridge, longer cable distances, routing across intranets and the Internet, and addressing of unlimited RMCs from a single PC.

See also: RS232 Wiring, Serial Configuration, TCP/IP Direct to RMC-ENET, TCP/IP-to-RS232 Bridge, Connecting RMCWin to an RMC

3.3.7.2 Communication Driver: Serial Configuration

The settings for the "RS-232 Monitor" serial port on the RMC CPU are as follows:

Signals: RS232
Baud Rate: 38400
Data Bits: 8
Parity: None
Stop Bits: 1
Flow Control: None

Therefore, RMCWin does not require choosing these options to communicate with the RMC. The only setting necessary to set up on RMCWin is to select which serial port will be used. Two other options are available that affect the performance and reliability.

To configure the Serial communication driver:

1. Start RMCWin.
2. On the main window's **Tools** menu, click **Options**, and then click the **Communication** tab.
   You can also double click the **Communication** pane of the main window's status bar, or right-click this pane and then click **Communication Options** from the shortcut menu. This is the pane that displays the current communication path and state (e.g. "COM1: Offline").
3. Under **Communication Driver**, click **Serial**.
4. Select the desired settings described below.
5. Click **OK**.

The serial driver settings offered in this dialog box are listed below:

- **Serial Port**
  This list allows the user to change the serial port used to communicate between the RMC and RMCWin on the PC.
  An alternative way to change this setting is to right-click on the Communication pane of the main window's status bar, and select the serial port from the shortcut menu.

- **Communications Update Rate**
  This slider adjusts a delay that is inserted between communications over the serial port on the PC when talking to a RMC. The only purpose for this control is to decrease the load on slower PCs; the RMC can handle any of the communication speeds. If this software seems to slow down Windows, move this slider closer to **Slow**.
  Move the slider toward **Slow** to decrease the load on the PC, and toward **Fast** to increase the load on the PC.

Note: This setting is ignored while reading up a plot; no delay is used.

- **Small-Packet Mode**
  By selecting this check box, the size of the packets sent over the serial port are reduced. This is necessary on some computers that do not give Windows enough time to service the serial port queue, and therefore cause serial port overruns. Most notably this has been known to happen on some computers running Steeplechase Software’s Visual Logic Controller. If you experience problems maintaining the connection with the RMC, try checking this box.

Note: By checking this box, the communication speed will slow by roughly 40%. Therefore, only use this option if necessary.

See also: RS232 Wiring, Serial Overview, Connecting RMCWin to an RMC

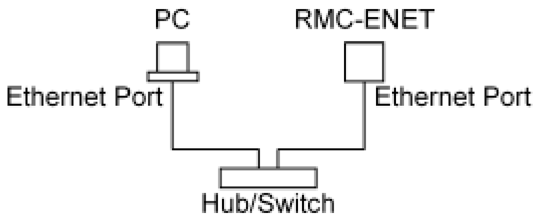
3.3.7.3 Communication Driver: TCP/IP Direct to RMC-ENET Overview

Note: This communication driver requires RMC ENET firmware dated 20001108 or later.

The **TCP/IP Direct to RMC-ENET** driver allows a PC with a TCP/IP interface (such as an Ethernet adapter or modem with TCP/IP installed) to communicate with the RMC-ENET module. Because it requires the RMC-ENET module, this method is not available for RMCs that require a different communication module such as PROFIBUS-DP.

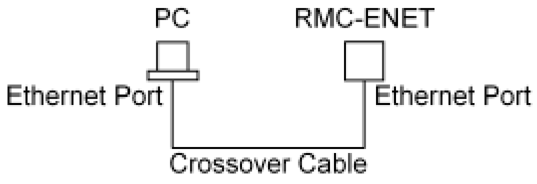
Note: If your computer has a firewall, it may prevent RMCWin from finding RMC100s on your Ethernet network from the Communications tab in the Options dialog. See the Setting the Firewall to Allow RMC100 Ethernet Browsing topic for details. Also, If your RMC is behind a firewall, make sure the firewall forwards Ethernet port 1324 to the RMC's IP address.

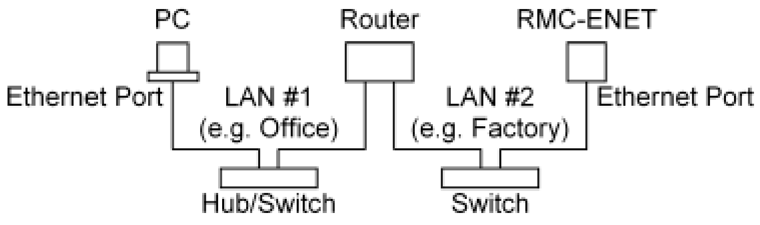
The most common way to connect the PC to the RMC-ENET is to connect each to a hub or switch using CAT5 cables:

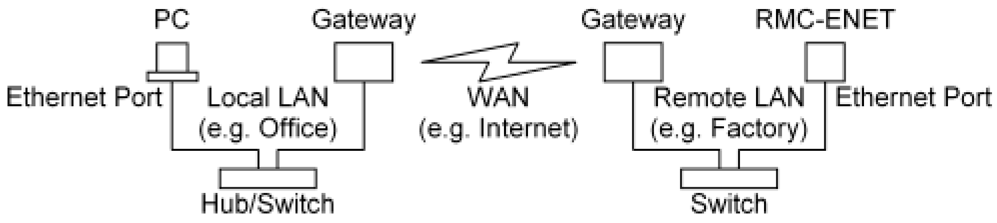


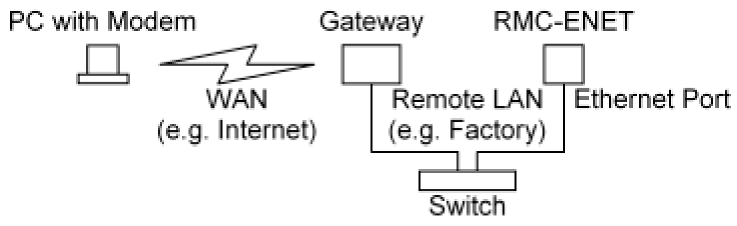
Note: A hub should only be used if the RMC-ENET is only used for configuration. If the RMC-ENET is used for control, then a switch should be used instead to eliminate collisions and thereby increase determinism.

Notice that the above wiring method is one of many ways of connecting the two devices. The following and many combinations of the following are also possible:









Comparison with Other Communication Drivers

This driver is up to 100 times faster than the other drivers, unless routing over a WAN such as the Internet. However, it cannot be used to download firmware. Following are differences when compared to each of the other drivers:

- Serial: The serial port method requires neither an RMC-ENET module nor an Ethernet adapter in the PC, but does not have as high performance, has no isolation, has shorter cable distances, and can only be used point-to-point.
- TCP/IP-to-RS232 Bridge: The bridge method does not require an RMC-ENET module, but is much slower and requires an additional bridge device.

See also: TCP/IP Direct to RMC-ENET Configuration, Serial Overview, TCP/IP-to-RS232 Bridge Overview, Connecting RMCWin to an RMC

3.3.7.4 Communication Driver: TCP/IP Direct to RMC-ENET Configuration

Note: This communication driver requires RMC ENET firmware dated 20001108 or later.

Communicating over TCP/IP requires configuring both the PC's and RMC's IP parameters, and then selecting the RMC IP address in RMCWin.

- For a discussion of choosing TCP/IP addresses, see Understanding IP Addressing.
- For instructions on setting up the RMC's TCP/IP settings, see RMC Ethernet IP Address Setup.
- For information on setting up the TCP/IP settings for your PC, refer to your Windows manuals and online help. Typically, you will not change your PC's TCP/IP settings if it is already installed. Instead, follow the instructions for configuring the RMC's TCP/IP settings, which will show the PC's settings for your convenience.

The quickest way to select the target RMC you want to communicate with is as follows:

1. Start RMCWin.
2. On the main window's status bar, right-click the **Communication** pane (e.g. "COM1: Offline"). A shortcut menu will appear. At the bottom of this shortcut menu will be listed all of the most recently accessed RMC IP addresses, plus all RMCs on the local network.

Note: If your computer has a firewall, it may prevent RMCWin from finding RMC100s on your Ethernet network. See the Setting the Firewall to Allow RMC100 Ethernet Browsing topic for details.

3. In the shortcut menu, click the IP address of the RMC you want to communicate with.

An alternative way to set up this driver is to use the **Communication** tab in the **Options** dialog box. This method allows you to manually type in an IP address and configure other driver options:

1. Start RMCWin.
2. On the main window's **Tools** menu, click **Options**, and then click the **Communication** tab. You can also double-click the **Communication** pane of the main window's status bar, or right-click this pane and then click **Communication Options** from the shortcut menu. This is the pane that displays the current communication path and state (e.g. "COM1: Offline").
3. Under **Communication Driver**, click **TCP/IP Direct to RMC-ENET**.
4. Select the desired settings described below.
5. Click **OK**.

The controls in this dialog box are listed below:

- **Target IP Address text box:**

Type the IP address of the RMC ENET module you want to communicate with. You can also use the drop-down list to select from the last several IP addresses you have entered.

- **Autobrowse Local Network check box:**

RMCWin can automatically detect all RMC ENET modules on the network that your PC's Ethernet adapter is connected to. This requires RMC ENET firmware dated 20010523 or newer. Check this box to continuously update the browse list while it is displayed.

- **Refresh button:**

This button is available if the autobrowse feature above is not selected. It will scan the network of your PC's Ethernet adapter once for RMC ENETs and update the browse list accordingly.

- **Configure button:**

If you have an RMC ENET in the browse list currently selected, then you can configure its TCP/IP settings directly by clicking this button. See RMC Ethernet IP Address Setup for details.

- **Browse list:**

The browse list is updated using one of the above two methods (autobrowse or manual refresh). It will list all RMC100s with an RMC ENET module connected to the same network as your PC's Ethernet adapter. These RMC ENET modules must have firmware dated 20010523 or newer to show up on this list. Notice that RMCs will appear on this list even if they have invalid TCP/IP settings for the network. Therefore, you may not be able to connect to all RMCs in the list without first updating the TCP/IP settings. Click **Configure** to update the TCP/IP settings of the currently-selected RMC, even if it currently has invalid TCP/IP settings.

Note: If your computer has a firewall, it may prevent RMCWin from finding RMC100s on your Ethernet network. See the Setting the Firewall to Allow RMC100 Ethernet Browsing topic for details.

- **Communications Update Rate slider:**

  This slider adjusts a delay that is inserted between transactions between RMCWin and the RMC. The only purpose for this control is to decrease the load on slower PCs; the RMC can handle any of the communication speeds. If this software seems to slow down Windows, move this slider closer to **Slow**.

  Move the slider toward **Slow** to decrease the load on the PC, and toward **Fast** to increase the load on the PC.

Note: This setting is ignored while reading up a plot; no delay is used.

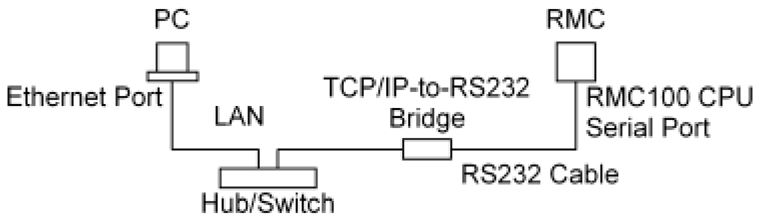
See also: TCP/IP Direct to RMC-ENET Overview, Connecting RMCWin to an RMC

3.3.7.5 Communication Driver: TCP/IP-to-RS232 Bridge Overview

Note: This communication driver requires RMC100 CPU firmware dated 20010522 or later.

This driver adds many of the benefits of Ethernet—it allows routing and addressing multiple modules and provides isolation—to all RMCs because it uses the RMC100 CPU's "RS-232 Monitor" serial port, but does so at the expense of performance. This is possible using a TCP/IP-to-RS232 Bridge device.

The following diagram shows the typical network layout. Additional RMCs could either be added through an additional bridge each to the hub or switch, or they could be plugged into additional serial ports on a multi-port bridge.



This diagram could also be drawn with any of the TCP/IP routing variants shown in the TCP/IP Direct to RMC-ENET Overview topic. Delta does not sell its own TCP/IP-to-RS232 bridge, but many companies make them. The UDS from Lantronix (www.lantronix.com) was used in testing.

Comparison with Other Communication Drivers

- Serial: The serial port is generally several times faster, and eliminates the requirement for an Ethernet adapter in the PC, but it does not provide isolation, has much shorter cable distances, cannot route across intranets and the Internet, and can only address a single RMC per serial port.
- TCP/IP Direct to RMC-ENET: The direct method is much faster, and does not require an additional bridge device, but it is limited to RMCs with the RMC-ENET module.

See also: TCP/IP-to-RS232 Bridge Configuration, Serial Overview, TCP/IP Direct to RMC-ENET Overview, Connecting RMCWin to an RMC

3.3.7.6 Communication Driver: TCP/IP-to-RS232 Bridge Configuration

Note: This communication driver requires RMC100 CPU firmware dated 20010522 or later.

The main steps to configuring a TCP/IP-to-RS232 Bridge communication path are as follows:

- **Obtain a TCP/IP-to-RS232 Bridge**

There are a number of TCP/IP-to-RS232 bridge devices available. A search on the Internet for "Ethernet to Serial Converter" should find a number of manufacturers providing these devices. The RMC has been tested with the Lantronix (www.lantronix.com) UDS-10. If multiple RMCs will be hooked up using the same method, it may be worth investing in a multi-port bridge, so that a single bridge can handle several RMCs.

- **Connect the Bridge to the Ethernet Network**

How this is done depends on the particular bridge device, but in most cases will involve using a CAT5 Ethernet cable to connect the bridge to a hub or switch on the LAN that will be used.

- **Connect the Bridge to the RMC**

The bridge will be connected to the RMC's "RS-232 Monitor" serial port using an RS232 cable. The RMC's serial port is DTE. Therefore, if the port on the bridge is DTE, then a cross-over cable should be used; otherwise a straight-through cable should be used. See RS232 Wiring for details.

- **Configure the TCP/IP-to-RS232 Bridge**

Most of the work in setting up this driver involves configuring the bridge device. These bridges are typically designed to be very flexible, and as a result can be fairly confusing to set up. The following serial settings must be used to communicate with the RMC:

Signals: RS232
Baud Rate: 38400
Data Bits: 8
Parity: None
Stop Bits: 1
Flow Control: None

In addition to these basic serial settings, there are often many other options for the serial interface. The bridge should act in Server mode. Therefore, it will not need to deal with initiating connections or setting up a specific remote IP address.

The second main set of settings you will need to configure in the bridge is the TCP/IP interface. This will include the following settings:

- **IP Address, Subnet Mask, and Default Gateway:** The same guidelines laid out in Understanding IP Addressing for selecting an RMC ENET's IP parameters apply when selecting a bridge's IP parameters. The IP address entered for the bridge will need to be entered in RMCWin, as described below.

- **TCP/UDP:** RMCWin will open a TCP connection. Therefore, if you are given the option of selecting between TCP and UDP, select TCP.

- **TCP/UDP Port:** The TCP/UDP port number set up for the bridge's serial port is really irrelevant, as long as the same value is entered in the RMCWin TCP/IP-to-RS232 Bridge driver configuration screen.

- **Configure RMCWin for the TCP/IP-to-RS232 Bridge Driver**

After you have set up the bridge, it is time to select and configure the TCP/IP-to-RS232 Bridge driver in RMCWin.

To configure the TCP/IP-to-RS232 Bridge communication driver:

1. Start RMCWin.
2. On the main window's **Tools** menu, click Options, and then click the **Communication** tab. You can also double-click the **Communication** pane of the main window's status bar, or right-click this pane and then click **CommunicationOptions** from the shortcut menu. This is the pane that displays the current communication path and state (e.g. "COM1: Offline").
3. Under **Communication Driver**, click **TCP/IP-to-RS232 Bridge**.
4. Select the desired settings described below.
5. Click **OK**.

The TCP/IP-to-RS232 Bridge driver settings offered in this dialog box are listed below:

- **Target IP Address text box:**
  Type the IP address of the TCP/IP-to-RS232 bridge device whose serial port is connected to the RMC you want to communicate with. You can also use the drop-down list to select from the last several IP addresses you have entered. Guidelines for setting up the bridge device were given above.

- **TCP Port text box:**
  The TCP/IP-to-RS232 bridge device can typically be configured to respond on any one TCP port. The port that the bridge is configured to use and the TCP port entered in this text box must match. Which port is used is usually not important, as long as it is between 1024 and 65535. In most cases, you should be able to use the default TCP port on the bridge device. Guidelines for setting up the bridge device were given above.
- **Communications Update Rate slider:**
  This slider adjusts a delay that is inserted between transactions between RMCWin and the RMC. The only purpose for this control is to decrease the load on slower PCs; the RMC can handle any of the communication speeds. If this software seems to slow down Windows, move this slider closer to **Slow**.
  Move the slider toward Slow to decrease the load on the PC, and toward Fast to increase the load on the PC.

Note: This setting is ignored while reading up a plot; no delay is used.

See also: TCP/IP-to-RS232 Bridge Overview, Connecting RMCWin to an RMC

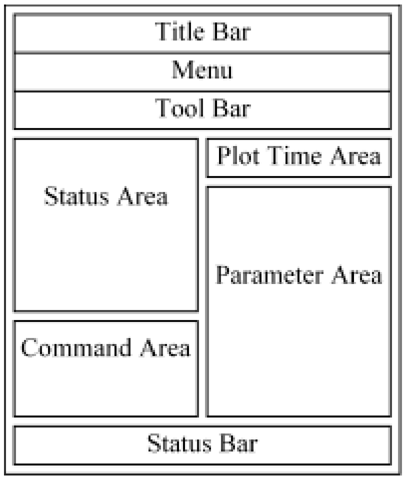
3.4 Basic Topics

3.4.1 Selecting Your View

RMCWin supports three types of views. From the **Windows** menu, choose one of the three possible views at the top of the menu. Each view is described below:

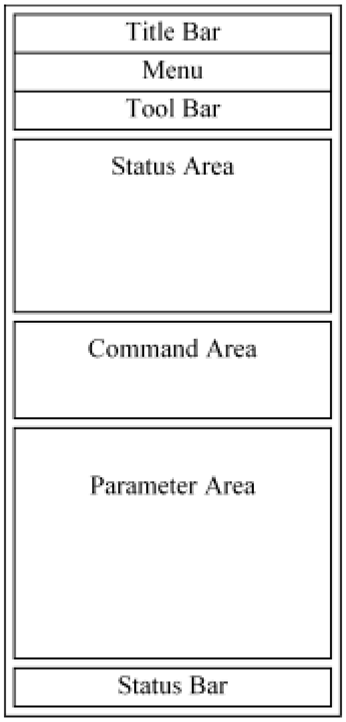
- **Full Horizontal View:**

This view displays the status, command, plot time, and parameter areas of all axes at once. The following diagram shows the positions of each area:



- **Full Vertical View:**

This view displays the status, command, and parameter areas of all axes at once. The following diagram shows the positions of each area:

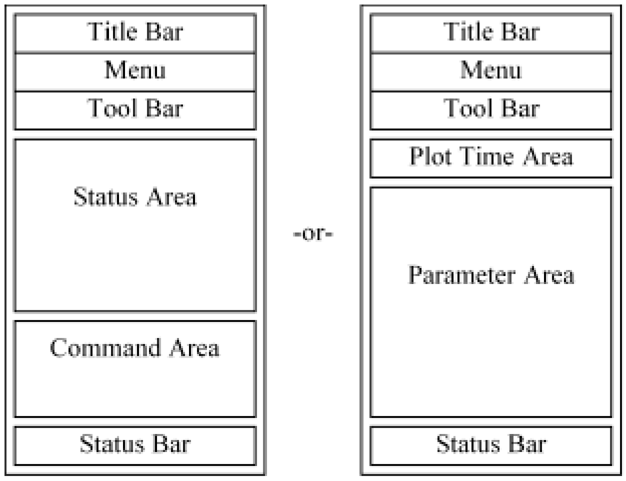


- **Half View:**

This view displays either the status and command areas or the plot time and parameter areas of all axes at once. To switch between the status/command and plot time/parameter displays, use one of the following methods:

- o Press either CTRL+LEFT ARROW or CTRL+RIGHT ARROW.
- o On the **Window** menu, click **Toggle Displayed Fields**.
- o On the toolbar, click the **Toggle Displayed Fields** button ( ).

The following diagrams show the positions of each area in this pair of views:



3.4.2 Accessing Context Sensitive Help

Each of the fields displayed on the main screen and the table editors have an associated help topic.

To display the context sensitive help:

- Right-click on the field for which you want help to display the shortcut menu, and click **Help on field**, where **field** is the name of the field.
- Or, select the cell of the field for which you want help, and press F1.

3.4.3 Changing Data from the Keyboard

The data in the COMMANDS, PLOT TIME and PARAMETER sections may be changed from the keyboard. Additionally the data in any of the table editors can be modified from the keyboard.

To enter values from the keyboard you must first select one or more cells. Selected cells are highlighted.

To select a single cell from the keyboard:

1. Use the arrow keys to highlight a different cell.

To select multiple cells:

1. Press and hold SHIFT at the first cell to be selected.
2. Use the arrow keys to change the last cell to be selected.
3. Release SHIFT.

Once one or more cells have been selected, type the desired value using one of the following formats:

- To enter decimal numbers, type the value, without a leading zero.
- To enter hexadecimal numbers, type a leading **0x** followed by the hexadecimal digits. For example, **0xFFE0**. Notice that for backward compatibility, hexadecimal values may be entered with only a leading **0**. However, they will always be displayed with the **0x**.
- To enter an ASCII command, type the letter of the command. Notice that this works only in the COMMAND field. For example, **W**.

Press ENTER to finalize your changes, or ESC or cancel your edits.

Note: In the fields that display values in hexadecimal, Pop-up Editors can be used to edit the data. This is the easiest way to ensure that these words are modified correctly.

Commands can also be issued using shortcut keys. To learn the command shortcut keys, click the **Command** menu, and look at the right column of the menu. For example, ALT+P will issue the **P - Set Parameters** command. Commands can also be issued using Stored Commands; see Using Stored Commands for more information.

Data may also be copied around the main screen. The keys used for doing so are identical to most spreadsheets.

- To cut cells to the clipboard, press CTRL+X.
  This combination is available in the table editors, but not in the main window.
- To copy cells to the clipboard, press CTRL+C.
- To paste from the clipboard to the current location, press CTRL+V. When pasting a group of cells at once, the destination cells should not be selected. Instead, the cursor should be placed in the upper-leftmost cell to which you want the block to be copied.

3.4.4 Read-back versus Write Mode

The Command and Parameter areas of the main screen can operate in either of the following two modes:

Read-back Mode

In this mode, the Command and Parameter areas will be continually read from the RMC. This mode is necessary to monitor the commands given from another source (such as the PLC) and also to determine the parameters stored on the RMC. The Command area field values are displayed in red.

Note: Because the Command and Parameter fields are constantly being updated, it is possible to have changes you are making be overwritten by a field update. To avoid this, RMCWin will not update the fields in the area of the selected cell. For example, if a parameter on axis 2 is currently selected, then RMCWin will not overwrite the parameters for axis two until the currently selected cell moves out of axis 2's parameters.

Write Mode
In this mode, the Command and Parameter areas are never automatically updated. You can freely change values without them being overwritten by automatic updates. The Command area field values are displayed in yellow.

To switch between Read-back and Write modes, use one of the following methods:

- On the **Tools** menu, click **Toggle to readback/write mode**.
- Press CTRL+T from the main screen.
- Double-click the status bar pane that says either **Read** or **Write**.

3.4.5 RMC Configuration Dialog Box

The RMC Configuration dialog box has the following areas:

RMC Part Number
This area displays the part number of the current RMC module. This part number indicates the RMC module that is currently connected, if one is connected, or if RMCWin is offline, then it contains the last RMC module edited.

Click **Simulate** when offline to change the current RMC module for editing offline; see Working Offline for details.

Slots
This area displays a list of slots in the RMC, including version information if applicable. Most RMC slot types have additional configuration options that can be set. To view or change these options, use one of the following methods:

- In the **Slots** list, click the slot of your choice, then click **Slot Options**.
- In the **Slots** list, double-click the slot of your choice.

RMC CPU Firmware

This area lists the firmware versions of either the currently connected RMC, or, if offline, the last RMC connected to. Three parts of the firmware have different versions. The Boot and Loader are seldom changed and are used for updating firmware. The Control Program is the main firmware, so you may need to provide this version to technical support.

In addition to the version information, this section is also used for updating to new firmware and backing up current firmware. The firmware is updated and backed up as two files. The first holds the Control Program. The second holds the Boot and Loader. Backing up firmware is useful in cases where you wish to ensure that you have the same firmware on two or more RMCs. Simply back up the firmware in the module you want to use to a file on your PC and then update the firmware in the other RMCs with this file. Backing up firmware is also useful in cases where you want to try out a new firmware revision while still being able to restore the current firmware.

See Downloading New RMC100 Firmware for details on both.

3.4.6 RMC100/101 CPU Options Dialog Box

The **RMC100/101 CPU Options** Dialog Box has the following tabs:

- **Deadband Eliminator**
  Select the Deadband Eliminator algorithm in this tab.

3.4.7 Using Pop-up Editors

Pop-up editors are dialog boxes that simplify editing fields in RMCWin that would otherwise be confusing to edit by hand.

There are pop-up editors for each of the following parameters: Mode, Configuration Word, and Auto Stop. These values are normally displayed in hexadecimal, but by using the pop-up editor, editing these fields becomes intuitive.

There is also a pop-up editor for editing the Link Type and Value in the Event Step table editor. This simplifies Event Step table programming by displaying all possible link types and values.

There are three ways to start a pop-up editor:

- Right-click on a cell you want to modify, and click **Popup Editor for field** from the shortcut menu, where **field** is the name of the field.
- Double-click on the cell you want to modify.
- Select the cell you want to modify and press ENTER.

Note: The Status read-back field also has a window that can be accessed in the above three ways, but this window is a read-only window that displays the current status bits. It can be cleared by pressing ESC or closing the window.

3.4.8 Using the Status Bits Window

The Status Bits window displays the bits of the Status words for each axis. It is constantly updated as the bits change in the RMC.

To display the Status Bits window, do one of the following from the main window:

- On the **Window** menu, click **Status Bits**.
- Press CTRL+B.

Additionally, the Status Bits window can be displayed by using any of the methods of displaying a Pop-up Editor.

3.4.9 Using the Command Log

Command Log Explained

When debugging problems with a system that is using the RMC, it is often difficult to determine whether the problem is caused by an action of the RMC or the Programmable Controller (P/C). To help with this problem, the Command Log is available. The Command Log holds the last 256 commands received through the communication module; commands sent by RMCWin or the event step table are not displayed. Therefore, the Command Log can be used to determine which commands were actually received by the RMC.

The "To PLC from Module" Section

This section is applicable only with PROFIBUS-DP in Compact Mode. Otherwise, the values will be zero and grayed out. This section displays status information for each axis that is available to the P/C. For each axis there are two pieces of information:

Status - This displays the Status word of the axis.

Data - The value of this field depends on the Status Area Request field of the last command sent from the P/C on this axis. It can be equal to any of the Status Parameters.

The "From PLC to Module" Section

All communication types use this section. The most recent commands are at the top of the Command Log. In addition, you will notice that the commands that changed from the previous Command Log entry are colored red.

When using a Communication Digital I/O, the commands that are issued based on the digital inputs will be recorded in the command log. Refer to the Communication Digital I/O section for specifics on these commands.

For other communication modules, these fields represent the commands and command values received from the P/C. In addition, if there are rising or falling edges on CPU or Sensor Digital I/O inputs 0 used by the Input to Event table, 00045 commands (Start Events) will be issued to the command log.

Opening and Closing the Command Log Window

To open the Command Log window, do one of the following from the main window:

- On the **Window** menu, click **Command Log**.
- Press CTRL+L.

To close the Command Log window, do one of the following:

- Press ESC.
- On the File menu, click Exit.
- Click the Close button.

Pause/Resuming the Command Log

In some applications the Command Log may be scrolling continually.

To freeze the flow of commands in the log, do one of the following:

- On the **File** menu, click **Pause Log**.
- Press P while in the Command Log screen.
- Click **Pause** (⏸) from the toolbar.

To resume the flow of commands in the log, do one of the following:

- On the **File** menu, click **Update Log**.
- Press ENTER while in the Command Log screen.
- Click **Resume Update** (▶) from the toolbar.

Scrolling in the Command Log

The scroll bars may be used to scroll through the Command Log. Scrolling up will show newer data, and scrolling down will show older data. In addition, the UP ARROW, DOWN ARROW, PAGE UP, PAGE DOWN, HOME, and END keys can be used to scroll.

Saving the Command Log

You can save the command log for later reference. The file is stored in text format and can be opened later both in RMCWin or a text editor. The default file extension is .log.

To save a command log:

1. On the **File** menu, click **Save**.

2. In the **File name** box, enter the name of the file.
3. Click **Save**.

> **Note:** As soon as the **Save** command is clicked, the Command Log is automatically paused. After saving the file, the title bar will display the filename. To return to the current Command Log, click **Resume Update**.

Opening a Command Log

You can view command logs that were previously saved.

To save a command log:

1. On the **File** menu, click **Open**.
2. In the **File name** box, enter the name of the file.
3. Click **Open**.

When a command log is opened, the command log window will stop updating and display the opened log. To resume displaying the current log, click **Resume Update**.

Changing the Command Log Properties

The command log font size and bold properties can be changed using the **Properties** dialog box. Changes made to these properties are automatically saved from session to session.

To open the **Properties** dialog box:

1. On the **File** menu, click **Properties**.
2. In the **Command Log Properties** dialog box, select your font size and check whether or not you want the normal and/or changed cells to use the bold version of the font.
3. To try the changes without closing the dialog box, click **Apply**.
4. To use the changes and close the dialog box, click **OK**.
5. To close the dialog box without saving changes, click **Cancel**.

3.4.10 Using the Parameter Error List Window

The Status word for each axis has a bit called Parameter Error. This bit is set when a problem related to user-issued commands is encountered. Because there are dozens of specific problems that can lead to this bit being set, it is important to be able to identify which of the specific parameter errors caused the bit to be set. This identification is simplified by using the Parameter Error List Window.

To start the Parameter Error List window:

- On the **Windows** menu, click **Parameter Error List**.

The window that is displayed displays all parameter errors that have been captured by RMCWin since RMCWin was started. The axis each error occurred on and a short description of the error is listed in this dialog box. To receive more in-depth help on a particular error do one of the following:

- Double-click on the error in the **Most Recent Parameter Errors** list.
- Click the error in the **Most Recent Parameter Errors** list, and then click **Help on Error**.
- Click the error in the **Most Recent Parameter Errors** list, and then press F1.

The list of errors in the Parameter Errors dialog box is built while RMCWin is connected to the RMC. The program polls the RMC frequently for status information, including current parameter error. When a new parameter error occurs on an axis, it is added to the list. Because this list is maintained within RMCWin and not in the RMC itself, restarting RMCWin will erase the list. Similarly, pressing the **Clear List** button will clear the list, but not affect the RMC in any way. It is also important to understand that because RMCWin polls to find out what errors have occurred, it is possible for an error to occur that is then cleared quickly by another valid command, causing RMCWin to miss capturing the error in its list.

3.4.11 Using the I/O Bit Monitor

The RMC has two digital inputs and two digital outputs on the main CPU. In addition, if you purchased a package with either or both of the Communication and Sensor Digital I/O modules, you will have additional digital inputs and outputs. During setup it is helpful to be able to view the status of these inputs and outputs. The I/O Bit Monitor is provided for this purpose.

To display the I/O Bit Monitor, do one of the following from the main window:

- On the **Window** menu, click **I/O Bit Monitor**.
- Press CTRL+M.

The bit monitor displays check boxes for each input and output on your RMC module. When an input is on, the box will have a check in it. When the input is off, it will be blank. Similarly, the output check boxes will be checked when the RMC is driving an output.

Recall that inputs and outputs on the Digital I/O modules can be inverted using software. For inverted inputs, the I/O Bit monitor displays them after they have been inverted. For inverted outputs, the I/O Bit monitor displays them before they have been inverted.

3.4.12 Using Stored Commands

When setting up and tuning the axes, it is usually necessary to repetitively move the axes between two or more positions. For this reason RMCWin stores 10 motion profiles for each axis; these are called stored commands. For details on setting up these profiles, see Editing the Stored Command Table. These stored commands can be used in either partial profile or full motion profile modes.

When executed in partial profile mode, only the Command and Command Value fields of the stored command are copied into the command fields of the current axis. The current axis is the axis of the currently selected cell on the main screen. Therefore, the Mode, Accel, Decel and Speed remain the same as before the move was requested. To use a stored command as a partial profile use any of these methods:

- Hold down CTRL and press the number of the stored command you wish to execute: 0 to 9. (e.g. CTRL+2 uses the partial profile of stored command 2).
- On the **Stored Cmds** menu, click the move you want to execute.
- Click the button of the desired stored command on the Toolbar.

When executed in full motion profile mode, all six of the command fields are copied from the stored command to the command fields of the current axis. To use a stored command as a full motion profile, use any of these methods.

- Hold down ALT and press the number of the stored command you wish to execute: 0 to 9. (e.g. ALT+2 uses the full motion profile of stored command 2).
- On the **Stored Cmds** menu, point to **Full Motion Profile**, and then click the move you want to execute.
- Click the button of the desired stored command on the Toolbar while holding down the ALT key.

3.4.13 Changing the Axis Names

The names of the axes are displayed throughout RMCWin. They are used only for display purposes. An axis name can have up to six characters.

To view or change all axis names, do the following:

1. On the main window's **Tools** menu, click **Options**.
2. Click the **Axis Names** tab.
3. Type in a new name for one or more axes.
4. Click **OK**.

To edit an individual axis name:

1. On the main screen, double-click the axis name.
2. Edit the axis name.
3. Click **OK**.

The axis names are attached to a single board file. If you are using multiple RMCs, then see the Using Multiple RMCs topic for details on using board files to keep track of multiple RMCs.

3.4.14 Using Multiple RMCs

RMCWin can keep track of several RMCs. The following pieces of information are associated with each RMC:

- RMC name (the filename)
- Names of each axis
- Parameters of each axis (configuration word, scale, offset, etc.)
- Command fields for each axis, except the Command itself (Mode through Command Value).
- Plot times for each axis.
- Hardware configuration of the RMC. This includes:
    - List of the physical modules included (e.g. PROFIBUS-DP, RMC100 CPU, two MDTs).
    - List of options for these modules. This includes all settings modified through the **RMC Configuration** dialog box (e.g. PROFIBUS station address, MDT blanking periods, number of SSI data bits, analog board channel assignments).
- Communication path

A board file (.bd1) stores all of the above with the exception of the communication path, which is remembered for the board file in Windows' internal Registry. Therefore, different computers can store different communication paths for the same board file.

See the following related topics:

- Creating a New Board File
- Editing Board File Information
- Changing Between Board Files
- Load Parameters Command

3.4.15 File Types

An RMC100 motion control project may use several of the editors available in RMCWin. Several editors have their own file types in which to save data. This may result in multiple files for each project.

In order to keep track of the project, it is recommended that all files for a project be kept in the same directory.

The following file types may be generated by RMCWin:

Extension	File Description	Used with…

.bd1	Board Parameter File	Board File
.plt	Plot Data File	Plots
.st1	Event Step Files	Event Step Editor
.fn1	Stored Function Files	Stored Command Editor
.pr1	Motion Profile Files	Motion Profile Editor
.i2e	Input to Event Files	Input to Event Editor
.log	Stored Command Logs	Command Log
.Crv	Curve Files	Curve Tool
.lcd	LCD Screen Files	LCD Screen Editor
.map	Status Map File	Status Map Editor

3.4.16 Creating a New Board File

Board files are used to store the following pieces of information:

- Names of each axis
- Parameters of each axis (configuration word, scale, offset, etc.)
- Command fields for each axis, except the Command itself (Mode through Command Value).
- Plot times for each axis.
- Hardware configuration of the RMC. This includes:
  - List of the physical modules included (e.g. PROFIBUS-DP, RMC100 CPU, two MDTs).
  - List of options for these modules. This includes all settings modified through the **RMC Configuration** dialog box (e.g. PROFIBUS station address, MDT blanking periods, number of SSI data bits, analog board channel assignments).

Refer to Using Multiple RMCs for general board file information.

To create a new board file:

1. On the **File** menu, click **New**.

The resulting board file will have the following characteristics:

- It will use the communication path that is currently selected.
- The parameters and commands will be reset to the defaults.
- The axis names will be set to the default axis names.

- The plot times will be set to the minimum (1ms for 1ms control loops, 2ms for 2ms control loops).
- The hardware configuration will be set to the current RMC configuration.

3.4.17 Changing Between Board Files

Board files are used to store the following pieces of information:

- Names of each axis
- Parameters of each axis (configuration word, scale, offset, etc.)
- Command fields for each axis, except the Command itself (Mode through Command Value).
- Plot times for each axis.
- Hardware configuration of the RMC. This includes:
    - List of the physical modules included (e.g. PROFIBUS-DP, RMC100 CPU, two MDTs).
    - List of options for these modules. This includes all settings modified through the **RMC Configuration** dialog box (e.g. PROFIBUS station address, MDT blanking periods, number of SSI data bits, analog board channel assignments).

Refer to Using Multiple RMCs for general board file information.

To save a board file:

1. On the **File** menu, click **Save As**.
2. In the **File name** box, enter the name of the file.
3. Click **Save**.

To load a board file:

1. On the **File** menu, click **Open**.
2. In the **File name** box, enter the name of the file.
3. Click **Open**. The file will be loaded. If the new board file uses a different communication path, then the previous communication path be closed and the new communication path will be opened.

To load just everything except the hardware configuration from a board file:

1. On the **File** menu, click **Load Parameters**.
2. In the **File name** box, enter the name of the file.
3. Click **Open**. The parameters, commands, plot times, and axis names will be loaded from the

selected board file into the currently open board file. The filename will not change on the currently open board file.

3.4.18 Editing Board File Information

The following pieces of information associated with a board file can be changed in the manner described below:

Axis Names-	Refer to the Changing the Axis Name topic for details.
Parameters -	Refer to the Parameter area topic for details.
Plot Times-	Refer to the Plot Time area topic for details.
Default Commands-	Refer to the Command area topic for details.
Module Configuration-	All of these settings are modified from the **RMC Configuration** dialog box. To display this dialog box, from the **Tools** menu, click **Module Configuration**.
Communication Path-	Although the communication path is not actually stored in the board file itself, it is remembered for every board file in the Windows Registry. See Connecting RMCWin to an RMC for details on changing this option.

3.4.19 Load Parameters Command

The board files for the RMC contain the following pieces of information:

- Sixteen parameters (e.g. configuration word, scale, auto-stop word) for each axis.
- Plot time of each axis.
- Axis name of each axis.
- Hardware configuration of the RMC. This includes:
  - List of the physical modules included (e.g. PROFIBUS-DP, RMC100 CPU, two MDTs).
  - List of options for these modules. This includes all settings modified through the **RMC Configuration** dialog box (e.g. PROFIBUS station address, MDT blanking periods, number of SSI data bits, analog board channel assignments).

When opening a board file using the **Open** command, the entire board file is loaded. However, if RMCWin is connected to an RMC and the hardware configuration in the RMC does not match that in the board file, the board file cannot be loaded. As a result, the **Resolve Configuration Conflict** dialog box is displayed to allow the user to choose an action. See Resolve Configuration Conflict Dialog Box for details.

This situation can be avoided using this command, which does not open the new board file, but instead transfers everything except the hardware configuration from the selected board file into the currently open board file. Therefore this command only loads the following:

- Sixteen parameters (e.g. configuration word, scale, auto-stop word) for each axis.
- Plot time of each axis.
- Axis name of each axis.

3.4.20 Scale/Offset Calibration Utilities

3.4.20.1 Using the Scale/Offset Calibration Utilities

The Scale/Offset Calibration utility provides an easy way to set the RMC's mapping of transducer counts to user-defined position units. This involves both the scaling, and if applicable, offsetting to convert counts to position units, and defining the position unit range.

There are several Scale/Offset Calibration Utilities:

- Position Scale/Offset Calibration Utility
- MDT Scale/Offset Calibration Utility
- SSI Scale/Offset Calibration Utility
- Quadrature Calibration Utility
- Pressure Scale/Offset Calibration Utility
- Differential Force Scale/Offset Calibration Utility

3.4.20.2 Position Scale/Offset Calibration Utility

For a description of all Scale/Offset Calibration Utilities, see Using the Scale/Offset Calibration Utilities. To use the general position Scale/Offset Calibration Utility:

1. Choose two axis positions to use for this calculation. These points should be significantly far apart to minimize errors. Generally, the extend and retract limits are fine points to choose.
2. For MDT axes, on the axis you will be calibrating, make sure the No Transducer status bit is off. You may need to issue a 'P' command to turn the bit off. This will ensure the Counts values are read correctly from the RMC.
3. Place the cursor in a field under the position axis you wish to calibrate.
4. On the **Tools** menu, click **Scale/Offset Calibration**.
5. Move the axis to the first point. This can be done by jogging the axis manually, or by using the Open Loop command.
6. Measure the physical distance to the first point on the axis in the position units that you intend to

use (for example, thousandths of inches, millimeters).

7. Under **First position**, in the **Actual position** box, type the distance measured to the first point in position units.

8. Under **First position**, click **Use Current**, which copies the COUNTS on this axis being calibrated to the **Counts** box under **First position**. You can also manually type a value in this box, but it is easiest to use the **Use Current** button.

9. Move the axis to the second point. You may want to move the **Scale/Offset Calibration** dialog box out of the way so that you can use the main RMCWin window.

10. Measure the physical distance to the second point on the axis in the position units that you intend to use.

11. Under **Second position**, in the **Actual position** box, type the distance measured to the second point in position units.

12. Under **Second position**, click **Use Current**.

13. Under **Extend/Retract Limits**, choose how you wish to have the limits set:

    - If you had the extend and retract limits set correctly, click **Use current limits, adjusted for new Scale and Offset** to adjust the limits for your new Scale and Offset.

    - If the points you used for the two positions are your limits, click **Set limits to above positions**.

    - Otherwise, click **Set limits to the following values** and type the limits in the text boxes.

13. Click **Apply**, which sets the Scale, Offset, the Prescale Divisor bits of the Configuration word, Extend Limit, and Retract Limit.

14. Click **Done**.

15. Issue a 'P' command for the axis to have the new parameters take affect.

3.4.20.3 MDT Scale/Offset Calibration Utility

For a description of all Scale/Offset Calibration Utilities, see Using the Scale/Offset Calibration Utilities. To use the MDT Scale/Offset Calibration Utility:

1. Gather the following pieces of information for an axis:

    - The gradient of the transducer and its units (e.g. 9.012µs/inch)

    - The number of recirculations for the transducer

    - The desired number of position units per inch or centimeter

    - The transducer counts at your desired zero position

    - Whether you wish to have your position units increase or decrease with increasing counts

2. For the axis you will be calibrating, make sure the No Transducer status bit is off. You may need to issue a 'P' command to turn the bit off. This will ensure the Counts values are read correctly

from the RMC.

3. Place the cursor in a field under the axis you wish to calibrate.
4. On the **Tools** menu, click **MDT Scale/Offset Calibration**.
5. Enter the above pieces of information.
6. Under **Extend/Retract Limits**, choose how you wish to have the limits set:
    - If you had the extend and retract limits set correctly, click **Use current limits, adjusted for new Scale and Offset** to adjust the limits for your new Scale and Offset.
    - Otherwise, click **Set limits to the following values** and type the limits in the text boxes.
6. Click **Apply**, which sets the Scale, Offset, the Prescale Divisor bits of the Configuration word, Extend Limit, and Retract Limit.
7. Click **Done**.
8. Issue a 'P' command for the axis to have the new parameters take affect.

3.4.20.4 SSI Scale/Offset Calibration Utility

For a description of all Scale/Offset Calibration Utilities, see Using the Scale/Offset Calibration Utilities. To use the SSI Scale/Offset Calibration Utility:

1. Gather the following pieces of information for an axis:
    - The desired number of position units per inch, millimeter, or centimeter
    - The resolution of the transducer and its units (e.g. 0.005 mm)
    - The transducer counts at your desired zero position
    - Whether you wish to have your position units increase or decrease with increasing counts
2. Place the cursor in a field under the axis you wish to calibrate.
3. From the **Tools** menu, click **SSI Scale/Offset Calibration**.
4. Enter the above pieces of information.
5. Under **Extend/Retract Limits**, choose how you wish to have the limits set:
    - If you had the extend and retract limits set correctly, click **Use current limits, adjusted for new Scale and Offset** to adjust the limits for your new Scale and Offset.
    - Otherwise, click **Enter limits above** and type the limits in the text boxes.
6. Click **Apply**, which sets the Scale, Offset, the Prescale Divisor bits of the Configuration word, Extend Limit, and Retract Limit.
7. Click **Done**.
8. Issue a 'P' command for the axis to have the new parameters take affect.

3.4.20.5 Quadrature Calibration Utility

For a list of all Scale/Offset Calibration Utilities, see Using the Scale/Offset Calibration Utilities. To use the Quadrature Calibration Utility:

1. Gather the following pieces of information for an axis:
   - The desired ratio of quadrature counts to position units. Recall that there are four quadrature counts per line or pulse. If you want 3600 position units for one revolution on a 1000-line encoder, the ratio would be 3600 counts per 4000 position units.
   - The desired 16-bit position unit range.
   - Whether you wish to have your position units increase or decrease with increasing counts
2. Place the cursor in a field under the axis you wish to calibrate.
3. From the **Tools** menu, click **Quadrature Calibration**.
4. Enter the above pieces of information.
5. Under **Extend/Retract Limits**, choose how you wish to have the limits set:
   - If you had the extend and retract limits set correctly, click **Do not change**.
   - If you wish the limits to be the minimum and maximum position unit values, click **Set to maximum range**.
   - Otherwise, click **Edit the limits directly** and type the limits in the text boxes.
6. Click **Apply**, which sets the Scale, Coordinate Limit, the Prescale Divisor bits of the Configuration word, Extend Limit, and Retract Limit.
7. Click **Done**.
8. Issue a 'P' command for the axis to have the new parameters take affect.

3.4.20.6 Resolver Scale/Offset Calibration Utility

For a description of all Scale/Offset Calibration Utilities, see Using the Scale/Offset Calibration Utilities. To use the Resolver Scale/Offset Calibration Utility:

1. Gather the following pieces of information for an axis:
   - The desired number of position units for a certain number of counts. One rotation of the resolver returns 65536 counts.
   - Whether you wish to have your position units increase or decrease with increasing counts
2. Place the cursor in a field under the axis you wish to calibrate.
3. From the **Tools** menu, click **Resolver Calibration**.

4. Enter the above pieces of information.
   - Under **Coordinate Limits**, choose the range of valid positions desired.
5. Under **Extend/Retract** Limits, choose how you wish to have the limits set:
   - If you had the extend and retract limits set correctly, click **Use current limits, adjusted for new Scale and Offset** to adjust the limits for your new Scale and Offset.
   - Otherwise, click **Enter limits above** and type the limits in the text boxes.
6. Click **Apply**, which sets the Scale, Coord. Limit, Extend Limit, and Retract Limit.
7. Click **Done**.
8. Issue a 'P' command for the axis to have the new parameters take affect.

For more details on scaling a resolver axis, see the Resolver Scaling topic.

3.4.20.7 Pressure Scale/Offset Calibration Utility

For a description of all Scale/Offset Calibration Utilities, see Using the Scale/Offset Calibration Utilities. This utility is available only on single-ended auxiliary axes. To use the Pressure Scale/Offset Calibration Utility:

1. Choose two axis pressures to use for this calculation. These pressures should be easily measured, easily maintained, and significantly far apart to minimize errors.
2. Place the cursor in a field under the axis you wish to calibrate.
3. On the **Tools** menu, click **Scale/Offset Calibration**.
4. Hold the transducer at the first pressure.
5. Measure the actual pressure at this first pressure in the pressure units that you intend to use (for example, millibars, psi).
6. Under **First pressure**, in the **Actual pressure** box, type the pressure measured to the first point in pressure units.
7. Under **First pressure**, click **Use Current**, which copies the Counts on this axis being calibrated to the **Counts** box under **First pressure**. You can also manually type a value in this box, but it is easiest to use the **Use Current** button.
8. Ramp the pressure to the second value. You may want to move the **Scale/Offset Calibration** dialog box out of the way so that you can use the main RMCWin window.
9. Measure the physical pressure at the second value in the pressure units that you intend to use.
10. Under **Second pressure**, in the **Actual pressure** box, type the pressure measured to the second point in pressure units.
11. Under **Second pressure**, click **Use Current**.
12. Click **Apply**, which sets the Scale, Offset, and the Prescale Divisor bits of the Configuration word.

13. Click **Done**.
14. Issue a 'P' command for the axis to have the new parameters take affect.

3.4.20.8 Differential Force Scale/Offset Calibration Utility

For a description of all Scale/Offset Calibration Utilities, see Using the Scale/Offset Calibration Utilities. This utility is available only on differential force axes. To use the Differential Force Scale/Offset Calibration Utility:

1. Obtain the following information:

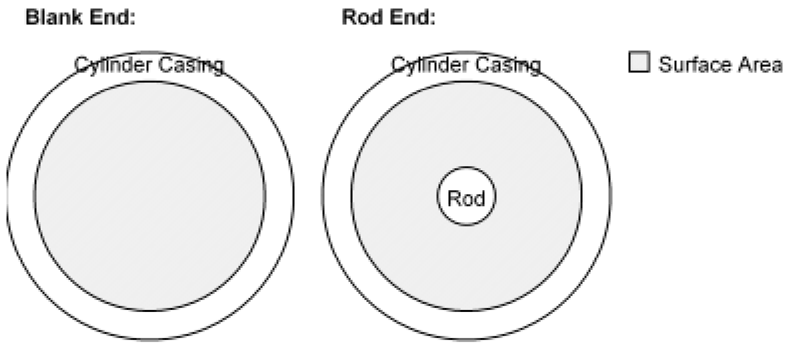
- **Pressure Gauge Scale:** This is the pressure at which the RMC will receive its maximum voltage or current. For example, if the RMC input is configured for a 0 to 5 V transducer, then you should enter the pressure at which the gauge will return a 5 V signal, even if it can return more than 5 V. For 4-20 mA gauges, give the pressure at 20 mA.

Note: The pressure gauges on each end of the hydraulic cylinder or motor should have the same range.

- **Actuator Type:** Three configurations are handled by this utility:

  - **Hydraulic Cylinder with a Single-Ended Rod**

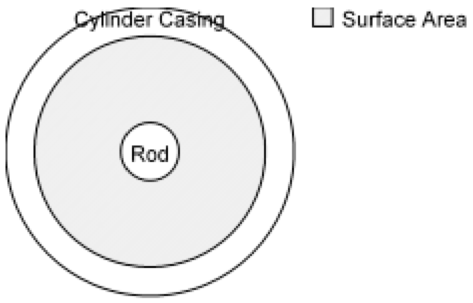
  In this configuration, the surface area on each side of the piston is multiplied by the pressure reading to give a resultant force. It is necessary to convert to force units when using a single-ended rod because equal pressures do not exert an equal force because the surface areas differ on each side of the piston, as shown below:



Note: Because the surfaces areas are different for each side of the piston in a hydraulic cylinder with a single-ended rod, SCALE A and SCALE B are not interchangeable. As a result, the first channel in the differential pair (channel 0 or 2) must be connected to the pressure sensor on the blank or cap end of the cylinder, and the second channel (channel 1 or 3) must be connected to the pressure sensor on the rod end of the cylinder. If you have wired the sensors in the opposite direction, and prefer not to change your wiring, then you will need to manually swap SCALE A and OFFSET A with SCALE B and OFFSET B.

  - **Hydraulic Cylinder with a Double-Ended Rod**

  In this configuration, the surface area on each side of the piston is multiplied by the pressure reading to give a resultant force. The following diagram shows how the surface area is calculated:

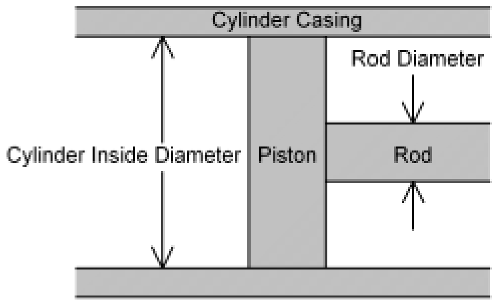


Because the surface areas on either side of the piston are equal, the scales and offsets will be equal for each pair.

- **Hydraulic Motor**

  In this configuration, the conversion from pressure to force can take many forms, since the pressure is converted to torque, which exerts a force on the system. Therefore, instead of a differential force being computed, a differential pressure is computed.

- **Cylinder Dimensions:** If your actuator is a hydraulic cylinder, then you will need to enter the cylinder's inside diameter and the rod's diameter. These values are necessary to convert the pressures read by each pressure gauge to forces by multiplying by the surface areas on each side of the piston.



- **Desired Force/Pressure Direction:** The difference force is computed by A - B, where A is the force computed from the first channel in the differential pair (channel 0 or 2), and the B is the force computed from the second channel in the differential pair (channel 1 or 3). As a result, the differential force will increase when Force A increases with respect to Force B. This is typically how applications are set up. In this case you should select the first Force Direction option in the calibration tool.

  However, in some applications, the differential force should increase when Force B increases with respect to Force A. In this case, you should select the second option. This will negate both forces (since -A - -B = B - A), so you will see Force A and Force B displayed as negative values, but the differential force should be correct.

- **Desired Force/Pressure Units:** This calibration tool is set up to easily allow you to have your force displayed in any of seven precisions (1000, 100, 10, 1, 0.1, 0.01, 0.001) of several common force units: pounds, Newtons, kilograms, metric tons (1000 kg), short (US) tons (2000 lb), long (UK) tons (2240 lb). In addition, you can manually enter a value in terms of any of the above units by selecting a base unit, and entering a conversion factor instead of selecting one of the powers of ten. See step 5 below for further details on using this section of the calibration tool.

2. Place the cursor in a field under the differential force axis you wish to calibrate.

3. On the **Tools** menu, click **Scale/Offset Calibration**.
4. Enter the above pieces of information.
5. Review the values displayed in the **Resultant Force/Pressure** area. This area shows the maximum forces, or pressures for hydraulic motors, that can be read from either channel. The values in Force Units are what will be used by the RMC and displayed in RMCWin. Notice that they cannot exceed 32,500 units. Displayed below each value in RMC Force Units is the equivalent value in the engineering units selected in the **Desired Force/Pressure Units** area.

   If the maximum forces in RMC Force Units are both below 3,250, then you may want to decrease the multiplier under **Desired Force/Pressure Units** to get finer resolution. For example, if the maximum force is listed as 900 force units and you have one force unit set to equal one (1) metric ton, then you can change one force unit to equal one tenth (0.1) of a metric ton. This will give a maximum force in RMC force units of 9,000. Notice that the maximum force in engineering until will still equal 900 metric tons.

   It is also possible to get a message to this effect:

   > "The requested scale is too large. Increase the size of each force unit."

   If this happens, then you are requesting that each force unit be less than one pressure transducer count. The solution is to change the size of each RMC force unit. For example, if you had one force unit set equal to one (1) pound (lb), and you get this message, then change one force unit to be equal to ten (10) pounds. If the message remains, continue to increase the size to 100 or 1000 pounds, or even switch from pounds to tons (short, US).
6. Click **Apply**, which sets the Scale A, Offset A, Scale B, Offset B, and the Prescale Divisor bits of the Configuration word.
7. Click **Done**.
8. Issue a 'P' command for the axis to have the new parameters take affect.

3.5 Using Plots

3.5.1 Using Graphs of Axis Moves

The RMC automatically gathers plot data for each move. A plot is triggered when any of the following commands are issued: Go, Open Loop, Relative Move, and Follow Spline Segment. The plot data is then collected for the duration and at a sampling rate as determined by the Plot Time field on the main screen. The Plot functionality in RMCWin can be used to read and display these graphs.

Click on one of the following topics for more details on using plots:

- Opening a Plot Window
- Reading Plot Data from the Motion Controller
- Selecting the Data to Plot
- Using the Plot Detail Window

- Viewing the Raw Plot Data
- Saving and Restoring Plots
- Printing a Plot

3.5.2 Opening a Plot Window

You can use one of the following methods to open a plot window from the main window:

- Click **Plot** () on the Toolbar. This opens a plot window for the current axis. The current axis is the axis of the currently selected cell on the main screen.
- Press INSERT. This opens a plot window for the current axis.
- On the **Window** menu, click on the **Plot** item of your choice. This opens a plot window for the axis indicated by the menu item name. If a plot window for this axis is already open, this command will shift the focus to the plot.

3.5.3 Reading Plot Data from the Motion Controller

If a RMC is connected to RMCWin, the data will be read from the RMC when a plot window is opened. The plot data stored in the RMC for that axis will change as new commands are issued to the axis. Therefore, it is often desirable to re-read data from the axis.

To re-read data from the RMC after opening a plot window, use one of these methods:

- On the **Data** menu, click **Upload plot from module**.
- Press INSERT.

For details on using a host controller to read plots from the RMC, see the Reading Plots from the Communication Module topic.

3.5.4 Selecting the Data to Plot

All plots read from the RMC contain the following information:

- ACTUAL POSITION
- TARGET POSITION
- DRIVE
- Status word

In addition, ACTUAL SPEED and TARGET SPEED are calculated from the ACTUAL POSITION and TARGET POSITION values and stored in the plot.

The RMC can provide two more pieces of data; the user chooses what data is collected using the **Plot Options** dialog box. To use this dialog box:

1. If you are going to change the extra graph information on only one axis, select a field in the axis.
2. On the **Tools** menu, click **Plot options**.
3. Click the option button of the data you want to include. The options are described below.
4. If you wish to change the extra graph information for all axes, select the **Set for all axes** check box.
5. Click **OK**.
6. Trigger a new graph to be stored by the module. This requires making a new move because the new information is not collected until the module begins a new graph.

The extra pieces of information that can be included are:

- **Extra Position Precision** - No additional data is displayed in the graph when this data is selected. Instead more precise positions are read from the module, which results in the better speed calculations. Use this option to get better approximations of the speed.
- **Command/Command Value** - The last command and command value received on the axis are stored for every plot entry. This is useful to verify the arrival and effects of new commands given during a move.

Note: Commands given from RMCWin are not included in the plot data.

- **Current Event/Link Value** - The current event STEP and LINK VALUE are stored for every plot entry. This is useful for debugging event sequences.
- **Raw Transducer Counts** - The raw counts read from the transducer are stored for every plot entry. Refer to Raw Transducer Counts for details.
- **Internal Speeds** - This option is generally only used under the direction of Delta technical support. A plot taken with this option will only have the basic information displayed above (Actual and Target Positions, Drive, and Status word), but instead of computing the Actual and Target Speeds from the differences in positions, it reads the RMCs Actual and Target speed values used internally. This can be useful in tracking down problems with gearing and feed forwards.

Note: The Internal Speeds option requires RMC100 CPU firmware dated 20010205 or newer.

- **Integral Drive** - This option plots the Integral Drive value in millivolts along-side the standard plot values. This is useful for determining its effect on your control system. The current rate of change in the integrator is also displayed in the Plot Detail window and Raw Plot Data.

Note: The Integral Drive option requires RMC100 CPU firmware dated 20020419 or newer.

3.5.5 Using the Plot Detail Window

A plot detail window is opened by default when a plot is read into RMCWin. This window has a title of **Data at 0.000 s** with the actual number changing depending on the hairline position. The hairline is a vertical line that can be moved using the arrow keys, page up and down keys, and by using the left mouse button. The data values displayed in the detail window represent the plot at the hairline position. Therefore, to read exact values at a position in the plot, move the hairline to the desired position on the plot and read the detail window data.

To hide the Detail Window, use one of the following:

- On the **Data** menu, click **Hide Detail Window**.
- Click the **Close** button of the Detail window.

To show the Detail Window after it has been hidden:

- On the **Data** menu, click **Show Detail Window**.

To display the individual bits on the Status word, do one of the following:

- Click on the body of the detail window. This toggles the detail window between displaying the Status word as a hexadecimal number and as the bit names. When displayed as bit names, the names of the off bits are displayed in dark gray, while the names of the on bits are displayed black for non-errors, and red for errors.
- On the **Data** menu, click **Display plot status bits**.
- Press CTRL+B while the plot status window is displayed. This will toggle the detail status bits.

To move the Detail Window, do one of the following:

- Drag the detail window by its title bar.
- On the **Data** menu, click **Move detail window**. Each time this command is clicked, the detail window will move to the next corner of the plot window.
- Press the TAB key, which executes the **Move detail window** command.

3.5.6 Viewing the Raw Plot Data

To view the plot data in numerical form rather than in graph form, you can use the Raw Data chart. This chart displays all the data in the plot at every sample. The sample that was marked by the hairline on the plot is highlighted in gray. You can use the scroll bar or arrow keys to move through the data.

To view the Raw Data chart, do one of the following:

- On the **Data** menu, click **View Raw Data**.
- Press CTRL+V while in the plot window.

To return from the Raw Data chart to the plot, do one of the following:

- On the **Data** menu, click **View Plotted Data**.

- Press ESC while in the Raw Data chart.

3.5.7 Saving and Restoring Plots

To save a plot, follow these steps:

1. Display the plot you wish to save.
2. On the **File** menu, click **Save As**.
3. In the **File name** box, enter the name of the file.
4. Click **Save**.

To view a previously saved plot, follow these steps:

1. Open a plot window. This may start loading a new plot from the RMC. It is not necessary to stop the download before opening a saved plot.
2. On the **File** menu, click **Open**.
3. In the **File name** box, enter the name of the file.
4. Click **Open**.

3.5.8 Printing a Plot

Plots can be printed out; each will take a single page. The user can set the margins, decide whether or not to print out the hairline with its detailed information, and choose to print in either landscape or portrait modes.

To set the margins:

1. On the **File** menu, click **Print Setup**.
2. Under **Margins**, enter the sizes of the margins in the four text boxes. These settings are in inches.
3. Click **OK**.

To print the hairline and the plot details at that time:

1. On the **File** menu, click **Print Setup**.
2. Select the **Print hairline and plot details** check box.
3. Click **OK**.

To select landscape or portrait modes:

1. On the **File** menu, click **Print Setup**.
2. Click **Printer Setup**.
3. Under **Orientation**, click either **Portrait** or **Landscape**.
4. Click **OK** in the **Print Setup** dialog box.
5. Click **OK** in the **Plot Print Options** dialog box.

To print a plot:

1. On the **File** menu, click **Print**.
2. Select the printer and number of copies to print.
3. Click **OK**.

3.5.9 Plot Time

The plot time field controls the time interval between plot samples. Because the number of samples for a full graph is fixed, the total graph length is also set by this parameter. In a full graph, there are 1024 samples. The plot time units are 1000/1024 milliseconds. Therefore, to request the graphs on an axis to be 4 seconds in length, this parameter should be set to 4. This will result in 1024 samples, each 4000/1024 milliseconds apart, for a total length of 4000 milliseconds.

The following two equations can be used for requesting a plot length:

Full Plot Length = PLOT TIME x 1 second

$$\text{Plot Interval} = \frac{\text{PLOT TIME} \times 1000 \text{ ms}}{1024}$$

3.5.10 Special Status Values Available In Plots

3.5.10.1 Target Speed

The target speed is displayed in a plot, both graphically and in the detail window, but it is not displayed in the main status area. Like the Actual Speed, this parameter is calculated. However, instead of being calculated off of Transducer Counts, it is calculated from the Target Position. Therefore, when viewing a plot you can compare the Actual and Target Speeds to determine how well the actual move is tracking the intended move.

Why is the Target Speed Jagged During Constant Velocity?

With older versions of the RMC100 CPU firmware and RMCWin, you may notice that during some constant velocity moves, the Target Speed is not flat. The jaggedness is caused by quantizing. That is, suppose that at constant velocity the axis is moving an average of 4.2 position units per millisecond. Because the motion controller only reads whole numbers of position units, the Target Position would appear to increase by 4 position units for four milliseconds, and every fifth millisecond it would move 5 position units. Therefore, if we were to calculate the speed based on a single interval, the speed would vary by 25% every fifth interval.

When the Target Speed is calculated, smoothing is performed so it does not vary by such a large amount, but some jaggedness is left from rounded-off position units.

3.5.10.2 Raw Transducer Counts

The **Raw Counts (lo)** field holds bits 0-15 of the Transducer Counts at the given plot time period. The **Raw Counts (hi)** field is provided for use by technical support, but also holds bits 16 and 17 of the current count. Therefore, if you are using counts above 65,535, you can combine the two fields to find the actual counts at each point on the graph.

3.5.10.3 Sum of Errors Squared

This field is computed by summing the squares of the position error for each time unit on a graph. The lower this value, the closer the actual position curve tracked the target position curve. The numerical value displayed in this field has no meaning by itself, but it can be used during tuning to check whether a tuned parameter was helpful or harmful to the tracking of the position. In order to have a valid comparison the two graphs must have the same length, and the moves need to have had the same target profile. Otherwise, the changed parameter is not isolated in the test.

3.5.10.4 Sum of Analog Errors Squared

This field is computed by summing the squares of the analog error for each time unit on a graph. The lower this value, the closer the actual analog curve tracked the target analog curve. The numerical value displayed in this field has no meaning by itself, but it can be used during tuning to check whether a tuned parameter was helpful or harmful to the tracking of the target curve. In order to make a valid comparison, the two graphs must be the same length, and the target analog profiles must be roughly the same. Otherwise, the changed parameter is not isolated in the test.

3.6 Table Editors

3.6.1 Table Editor Basics

Starting a Table Editor

To start an editor, from the **Tools** menu, click the desired editor. Notice that each editor also has a shortcut key that can be used to start it. These are displayed in the menu.

When the table editor is opened, the following steps will be taken:

1. For the tables other than the Stored Command table: if an RMC is connected to the RMCWin, the table will be read from the module.
2. If an RMC is not connected to RMCWin, the table will be read from a file. The filename used is

given by the board filename, with the appropriate extension appended.

3. If an RMC is not connected to RMCWin and the table could not be read from disk, then default values will be filled into the table.

Exiting an Editor

To close an editor, use one of these methods:

- Press ESC.
- On the **File** menu, click **Exit**.
- Click the **Close** button.

If changes have been made to the table that have not been saved to a file or downloaded to the RMC you will be prompted to do so upon exit. You will always be given the option of exiting without saving.

Editing the Tables

Once the editor window is open, you can type values in the same manner as on the main window. See Changing Data from the Keyboard for details.

Additional Editing Features in the Table Editors

To copy a column of values to many adjacent columns:

1. Enter the values you wish to have copied into adjacent columns in all fields in the left-most column in the range.
2. Select the range of cells with the keyboard or mouse. Make sure that the values you entered are in the left-most selected column.
3. On the **Edit** menu, click **Fill Right**. The left-most selected values will be copied to all other selected columns.

Resizing the Window

To resize the window, you may drag the borders of the window. After the border is released, the window will snap to the largest number of columns and rows that fit on the screen.

Resizing Columns

Resizing the window does not affect the width of the columns. There are two column widths that are stored. The first column, which gives the row titles, can have its width set independent of the other column widths. However, changing any one of the other column widths affects all columns other than the label column.

To resize the row title column:

1. Drag the right edge of the label heading. This will only affect the title column.

To resize all other columns:

1. Drag the right edge of a data column. This will affect the width of all data columns.

Saving and Restoring Tables

To save a table:

1. On the **File** menu, click **Save As**.
2. In the **File name** box, enter the name of the file.
3. Click **Save**. The file will be saved in text format.

To restore a table:

1. On the **File** menu, click **Open**.
2. In the **File name** box, enter the name of the file.
3. Click **Open**.

Uploading and Downloading Tables

Except the Stored Command table, all tables are stored on the RMC. The RMC will only use the tables stored in its memory. Therefore, the tables must be uploaded and downloaded to work on them in the table editors. If an RMC is connected to RMCWin, then the table will be uploaded when the table editor is opened.

To download a table to the Module, use one of these methods:

- On the **Online** menu, click **Download to Motion Controller**.
- Click **Download to Module** () from the toolbar.

To upload a table from the Module, use one of these methods:

- On the **Online** menu, click **Upload from Motion Controller**.
- Click **Upload from Module** () from the toolbar.

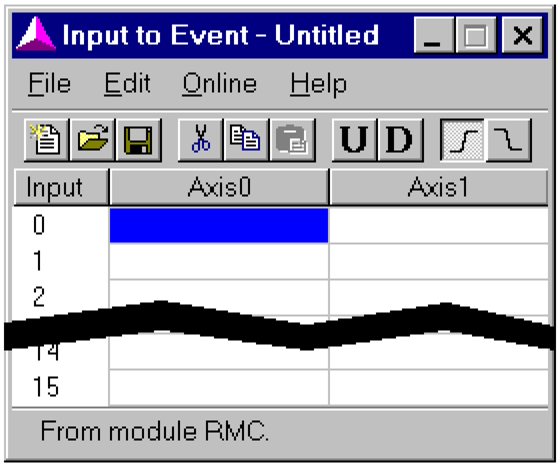
3.6.2 Editing the Input to Event Table

Input to Events Explained

Depending on the communication hardware used with the RMC module, between zero and sixteen digital inputs can be used to trigger events (start executing steps). Refer to the following table to determine which inputs can be used with the Input-to-Event feature:

Communication Type	Without Sensor DI/O	With Sensor DI/O
Comm. DI/O in Input-to-Event Mode	Comm DI/O inputs 0-15 represent input-to-event inputs 0-15.	Same as without Sensor DI/O.
Comm. DI/O in Parallel Position Mode, Parallel Event Mode, or Command Mode	No input-to-event inputs are available.	Sensor DI/O inputs 0-15 represent input-to-event inputs 0-15.
Non-Comm DI/O (PROFIBUS-DP, Modbus Plus, Ethernet, etc)	CPU inputs 0 and 1 represent input-to-event inputs 0 and 1.	Sensor DI/O inputs 0-15 represent input-to-event inputs 0-15.

The Input to Event Table is used to configure input to event behavior:



Input	Axis0	Axis1
0		
1		
2		
14		
15		

From module RMC.

When an input can be used to trigger an event, then the RMC looks into the Input to Event Table each time that input has a rising edge (it goes from non-conducting to conducting) or has a falling edge (it goes from conducting to non-conducting). Therefore, the Input to Event table is divided in half: one half for rising edges and one half for falling edges.

Each entry in the Input to Event table holds an Event Step number. The column of the entry represents an axis, and the row represents a digital input. When a digital input has a rising edge, all entries on the input's row which are not blank in the rising edge half of the I2E table are processed. An entry is process by issuing a Start Event command to the axis with the entry value as the command value (step number). When the digital input has a falling edge, the falling edge half of the I2E table is used.

Changing the Input to Event Table

Changes are made to the Input to Event table using the Input to Event Editor. Refer to Table Editor Basics for topics common to all table editors. The default extension for saved Input to Event tables is .i2e. Features specific to the Input to Event table editor are described below:

Selecting Rising Edges or Falling Edges

Only half of the input to event table is displayed at a time. To switch between the halves, click **Show Rising Edge** (⎍) or **Show Falling Edge** (⎍) from the toolbar or the **Edit** menu. The depressed toolbar button indicates the type of inputs edges currently being displayed.

Jumping to the Event Step Editor

You may want to view the Event Steps that are referenced by a cell in the Input to Event table.

To open the Event Step editor and jump to the appropriate step in that table:

- On the **Edit** menu, click **Go to Event Step**.
- Or, press CTRL+G.

3.6.3 Editing the Stored Command Table

Stored Commands Explained

The Stored Command table stores the 10 sets of commands with full profiles (MODE, ACCELERATION, DECELERATION, SPEED, COMMAND, and COMMAND VALUE) for each axis. See Using Stored Commands for details on using these stored commands to send commands to an axis using RMCWin.

Changing the Stored Command Table

These stored commands are stored in RMCWin and can be saved to disk for later retrieval. They are edited using the Stored Command table editor. Refer to Table Editor Basics for topics common to all table editors. The default extension for saved Stored Command tables is .fn1. Changes made to the Stored Command table are applied immediately.

3.6.4 Editing the Profile Table

Profiles Explained

The Profile table stores 16 profiles. For details on Motion Profiles, see Motion Profiles.

Changing the Profile Table

Changes are made to the Profile table using the Profile Editor. Refer to Table Editor Basics for topics common to all table editors. The default extension for saved Profile tables is .pr1.

3.7 Step Table Editor

3.7.1 Step Table Editor: Overview

Event Steps Explained

The Event Step table contains 256 Event Steps. For details on the use of Steps, refer to Event Control Overview. For a summary of keyboard shortcuts in this editor, see Step Table Editor: Keyboard Shortcuts.

Changing the Event Step Table

Changes are made to the Event Step table using the Event Step Editor. Refer to Table Editor Basics for topics common to all table editors. The default extension for saved Event Step tables is .st1. Features specific to the Event Step table editor are described below.

Printing the Event Step Table

You can print the current event step table, including its comments.

To set the margins:

1. On the **File** menu, click **Print Setup**.
2. Under **Margins**, enter the sizes of the margins in the four text boxes. These settings are in inches.
3. Click **OK**.

To select landscape or portrait modes:

1. On the **File** menu, click **Print Setup**.
2. Click **Printer Setup**.
3. Under **Orientation**, click either **Portrait** or **Landscape**.
4. Click **OK** in the **Print Setup** dialog box.
5. Click **OK** in the **Print Options** dialog box.

To print the table:

1. On the **File** menu, click **Print**. You can also press CTRL+P or click the print icon on the toolbar.
2. Select the printer and number of copies to print.
3. Click **OK**.

Jumping to an Event Step

Because the Event Step table has many columns, you may want to go to a specific Event Step. There are two ways of doing this:

- On the **Edit** menu, click **Goto Next Step in Sequence** or press CTRL+F. This will move the cursor to the step referred to by the current step's Link Next field.

- On the **Edit** menu, click **Go to Event Step** or press CTRL+G. This is a more general method, which prompts you to type the step number to which you want to go.

Deleting Columns

If you wish to delete a column or columns, follow these steps:

1. Click on the step number that is located at the top of each table column. This will highlight all fields in the step.
2. If you wish to select a range of steps, hold down the SHIFT key and click on the step number and the other end of the range. Now all steps between and including the two selected steps will be highlighted.
3. On the **Edit** menu, click **Delete Column x to Clipboard**.

Notice that the cells that were deleted are stored in the clipboard.

Inserting Columns

You can insert columns into the Event Step table in two ways:

To insert an empty column:

1. Select a field in the step that you want the step to be inserted in front of.
2. On the **Edit** menu, click **Insert Empty Column**.

To insert the columns in the clipboard from a delete or copy command:

1. Select a field in the step that you want the steps to be inserted in front of.
2. On the **Edit** menu, click **Insert x Columns from Clipboard**.

Automatically Updating Links

Whenever event steps are shifted left or right as described above under **Deleting Columns** and **Inserting Columns**, there is the potential of links to the shifted steps becoming broken. RMCWin is capable of adjusting links to steps that move using one of the above methods. This feature is enabled by default.

To toggle this feature on and off:

- On the **Settings** menu, click **Adjust Links on Column Operations**.

Note: The links affected by this option include those by the Link Next field, the Command Value of Start Event (E) and Teach Step (t) commands. Links from the Input to Event table can be checked as described under **Maintaining Input to Event Table Links** below.

Reporting Orphaned Links

Steps are deleted when the **Delete Columns to Clipboard** command described above is used. Also when steps are inserted into the Event Step table, the steps at the top are lost. For example, if five steps are inserted before step 10, then steps 10 to 255 are all shifted right. However, because there can only be 255 steps in the table, steps 251 to 255 are shifted off the end and lost.

Care must be taken to ensure that these lost steps weren't used. Therefore, the Event Step editor can automatically report when links to deleted steps are broken (orphaned). This feature is enabled by default.

To toggled this feature on and off, do the following:

- On the **Settings** menu, click **Report Orphaned Links**.

Maintaining Input to Event Table Links

Whenever event steps are shifted left or right as described above under **Deleting Columns** and **Inserting Columns**, there is the potential of entries in the Input to Event table to become broken. When the Maintain Input to Event Links option is enabled, the currently open Input to Event table will be checked to see if any events pointed to by the Input to Event table have moved or been deleted. If the Input to Event table editor is not open, it will temporarily be opened and load the table from the module or from the default file if no module is connected. This newly-loaded table will be checked.

To toggle this feature on and off, do the following:

- On the **Settings** menu, click **Maintain Input to Event Links**.

Adding Comments

The Event Steps editor offers the ability to enter a comment for each event step. These comments are then saved and restored with the step file. When a step has a comment associated with it, a comment icon () appears on the heading for the step. Comments are copied or moved with the step when all fields in a step are copied or moved —whether by clipboard commands such as cut, copy, and paste, or by inserting or deleting steps.

The Comment Editor is used to add, edit, and remove comments. Use one of the following methods to start the Comment Editor while in the Event Steps editor:

- On the **Edit** menu, click **Edit Comment**.
- Press CTRL+N.
- Double-click the heading of a step.

Using any of the above methods when the Comment Editor is already open will move the keyboard focus to the Comment Editor.

The Comment Editor will stay up until explicitly closed; you can freely switch between editing steps and their comments. The Comment Editor will display the comment for the step currently selected in the Event Steps editor. Changes made to comments take place immediately; therefore, you do not need to close the Comment Editor or move to another step to finalize changes before saving the file.

Note: The comments cannot be saved in the RMC, even if parameters are saved to Flash. The comments are only saved in the Event Step file (.st1). Make sure the Event Step file is saved in an accessible location for viewing the comments.

The Comment Editor responds to the following commands:

- Click **Restore** or press ALT+R to reset the comment for the current step back to its state when this step was first selected. This can also be interpreted as an undo.
- Click **Clear** or press ALT+C to erase the entire comment for the current step.
- Press CTRL+F or CTRL+G to jump to another step in the event step table without changing the keyboard focus back to the main window. See the **Jumping to an Event Step** section above for details on the differences of these commands.

- Press CTRL+N to switch keyboard focus back to the Event Steps editor.
- Click the **Close** button or press ALT+F4 to close the comment editor.

3.7.2 Step Table Editor: Keyboard Shortcuts

The following chart lists all keyboard shortcuts available in the Step Table Editor:

Press	**To**
CTRL+O	Close the current step table and open an existing file.
CTRL+S	Save the current step table.
CTRL+P	Print the current step table.
CTRL+X	Cut to the clipboard.
CTRL+C	Copy to the clipboard.
CTRL+V	Paste from the clipboard.
CTRL+F	Jump to the step linked to by the current step.
CTRL+G	Jump to any step (user is prompted).
CTRL+N	Display and switch keyboard focus to the Comment Editor.
CTRL+B	Open the Address Tool for pasting an address, if a Accel, Decel, Speed, or Command Value field is selected.
F1	Jump to the help topics.
ESC	Close the Step Table Editor.
ENTER	Start editing the current cell. If a Mode, Commanded Axes, Link Type, or Link Value field is selected, then a pop-up editor will be displayed.
DELETE	Return all selected cells to their default values.
Arrow Keys	Select a new cell.
TAB	Select the next cell to the right.
HOME	Select a new cell in the first step (0).
END	Select a new cell in the last step (255).
PAGE UP	Select a new cell in the step one screen to the left.
PAGE DOWN	Select a new cell in the step one screen to the right.

SHIFT+Selection Keys	Select cells in a range.

The following chart lists all keyboard shortcuts available in the Comment Editor:

Press	**To**
CTRL+N	Switch the keyboard focus back to the Step Table Editor.
CTRL+F	Jump to the step linked to by this step, but keep the keyboard focus in the Comment Editor.
CTRL+G	Jump to any step (user is prompted), but keep the keyboard focus in the Comment Editor.
ALT+C	Clear the current comment.
ALT+R	Reset the comment to before editing began.
ALT+F4	Close the comment editor.

3.8 LCD Screen Editor

3.8.1 LCD Screen Editor: Overview

Using the LCD420 display as documented requires the following components:

- RMCWin 1.14.0 or newer
- RMC100 CPU with an RJ-11 port
- RMC100 CPU Firmware 20001204 or newer
- LCD420 purchased after December 4, 2000 (new units have an ESC key while the old units do not)

Prior to the December 4, 2000 release, the LCD420 display had a different keypad and the firmware behaved much differently. Users of the previous version may choose to keep their older displays, in which case it is important to not update the firmware to 20001204 or newer. Otherwise, contact Delta for details on upgrading to the new LCD420 display and firmware.

The LCD Screen Editor is a window in RMCWin that is used to edit screens displayed on the LCD420 accessory for the RMC100 series motion controllers. The LCD420 is a hand-held or panel-mounted terminal with a 4-row, 20-column LCD display and a 20-key keypad.

To open the LCD Screen Editor:

1. On the **Tools** menu, click **LCD Screen Editor**.

For further information the LCD Screen Editor, select one of the following topics:

Editor Window Elements

- Editor Window Elements
- Tree Pane Details

- Screen Pane Details
- Field Pane Details
- Data Tab Details
- Format Tab Details
- Toolbar Details
- Status Bar Details

Using the LCD Screen Editor

- Using LCD Screen Files
- Uploading and Downloading LCD Screens
- Using the Clipboard
- Changing the View Options
- Keyboard Shortcuts

Using Screens

- Adding and Removing Screens
- Changing the Screen Order
- Editing Screen Text
- Selecting Insert or Overtype Mode
- Renaming Screens

Using Fields

- Adding and Removing Fields
- Moving and Resizing Fields
- Editing Field Properties
- Using Editable Fields
- Using Fields with Multiple Write Locations
- Renaming Fields

For further information on the LCD420 display itself, see LCD Display Terminal Overview and Using the LCD420 Terminal.

3.8.2 Editor Window Elements

3.8.2.1 LCD Screen Editor: Editor Window Elements

The LCD Screen Editor is divided into the following elements:

- **Tree Pane:** The tree pane is the Windows Explorer-style hierarchical tree located on the left side of the LCD Screen Editor. It represents the entire LCD screen file with all screens and fields shown in that tree. This pane is used to view the entire file at a glance, and for operations that are not contained in a single screen, such as moving a screen up or down in the list and moving a field from one screen to another. For details on the tree pane, see Tree Pane Details.

- **Screen Pane:** The screen pane is the 4-row, 20-column text box located in the upper-right quarter of the LCD Screen Editor. It represents the current screen in the current LCD screen file. This pane is used to view the screen, edit text in the screen, and to position and size fields in the screen. For details on the screen pane, see Screen Pane Details.

- **Field Pane:** The field pane is the form located in the lower-right quarter of the LCD Screen Editor. It represents the current field in the current screen. If no field is selected, this area will be empty.

This pane is used to view and edit field properties. For details on the field pane, see Field Pane Details.

- **Toolbar:** The toolbar holds buttons for commonly used commands. To help identify these buttons, hovering the pointer over a button will pop up a ToolTip and display a description in the status bar. For details on the toolbar, see Toolbar Details.
- **Status Bar:** The status bar holds a CAPS LOCK indicator, a NUM LOCK indicator, an indicator for Insert (INS) or Overtype (OVR) mode, and a line of text to display descriptions of the currently selected menu command or toolbar button. For details on the status bar, see Status Bar Details.

Changing the Layout

The following will modify the layout of these window elements:

- **Resize the panes.** Between the tree, screen, and field panes are two split bars. These raised borders between the panes can be dragged to adjust the amount of space given to each pane.
- **Resize the window.** Drag the border or the sizing handle in the lower-right corner of the window to resize the window.
- **Change the screen pane's font.** The font in the screen pane can be adjusted for optimal viewing on your monitor. See Changing the View Options for details.
- **Show or hide the toolbar and status bar.** On the **View** menu, click the **Toolbar** and **Status Bar** commands to toggle each bar. See Changing the View Options for further details.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.2.2 LCD Screen Editor: Tree Pane Details

The tree pane is the Windows Explorer-style hierarchical tree located on the left side of the LCD Screen Editor. It lists all screens and fields in the current LCD screen file. This pane is used to view the entire file at a glance and for operations that are not contained in a single screen, such as moving a screen up or down in the list and moving a field from one screen to another.

The following operations can be done from the tree pane. Follow the associated links for details on each:

- Add and remove screens. See Adding and Removing Screens.
- Change the screen order. See Changing the Screen Order.
- Add and remove fields. See Adding and Removing Fields.
- Rename screens. See Renaming Screens.
- Rename fields. See Renaming Fields.
- Cut, copy, and paste screens and fields to and from the clipboard. See Using the Clipboard.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.2.3 LCD Screen Editor: Screen Pane Details

The screen pane is the 4-row, 20-column text box located in the upper-right quarter of the LCD Screen Editor. It represents the current screen in the current LCD screen file. This pane is used to view the screen, edit text in the screen, and to position and size fields in the screen.

The following operations can be done from the screen pane. Follow the associated links for details on each:

- Edit screen text. See Editing Screen Text.
- Add and remove fields. See Adding and Removing Fields.
- Move and resize fields. See Moving and Resizing Fields.
- Display the previous or next screen. Use the PAGE UP and PAGE DOWN keys to show the previous and next screens. See Keyboard Shortcuts for a list including other shortcuts.
- Cut, copy, and paste fields and text to and from the clipboard. See Using the Clipboard.

In addition, two view options can be edited to affect how the screen pane appears: gridlines and font size. See Changing the View Options for details.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.2.4 LCD Screen Editor: Field Pane Details

The field pane is the form located in the lower-right quarter of the LCD Screen Editor. It represents the current field in the current screen. If no field is selected, this area will be empty. This pane is used to view and edit field properties.

The field pane has two tabs for editing the properties of each field: the **Data** and **Format** tabs. For details on any settings in these tabs, see the following topics:

- Data Tab Details
- Format Tab Details

Also, for information on making fields editable, see the following topic:

- Using Editable Fields

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.2.5 LCD Screen Editor: Data Tab Details

The field pane's Data tab is divided into two areas:

Data to Display

This area is used by all fields and determines what data will be displayed in this field. The first item to fill in is the **Data Area**. Listed below are all the data areas supported, and the additional options used by each.

- **Status**
  This area gives access to any of the ten status registers on any axis. These fields must be read only. This area has the following additional parameters that must be defined:

Axis: Select the axis for the status you want to display.

Field: Select the status field for the selected axis that you want to display.

- **Parameter**
  This area gives access to most of the parameters on any axis. These fields may be read only or editable. Changes made to parameters take effect immediately and do not require a separate Set Parameters (P) command to be issued. This area has the following additional parameters that must be defined:

Axis: Select the axis for the parameter you want to display.

Field: Select the parameter for the selected axis that you want to display. Notice that the Configuration Word and Auto Stop parameters cannot be displayed or edited.

Note: When parameters are changed through the LCD420 display, the Set Parameter On-the-Fly (0xD0-0xDF) commands are used internally. Not all parameters can be set on-the-fly and therefore, attempting to change them will result in a parameter error. For details on limitations of setting parameters on-the-fly, see Set Parameter On-the-Fly Commands.

- **Step Table**
  This area gives access to most fields in the Event Step table. These fields may be read only or editable. This area has the following additional parameters that must be defined:

Step: Select the step number for the field you want to display.

Field: Select the step field you want to display. Notice that the Mode, Command, Commanded Axes, Link Type, and Link Next fields cannot be displayed or edited.

- **Profile Table**
  This area gives access to most fields in the Motion Profile table. These fields may be read only or editable. This area has the following additional parameters that must be defined:

Profile: Select the profile number for the field you want to display.

Field: Select the profile field you want to display. Notice that the Mode field cannot be displayed or edited.

- **Input-to-Event Table**
  This area gives access to any Input-to-Event table entry. These fields may be read only or editable. Blank Input-to-Event table entries will be displayed as -1. This area has the following additional parameters that must be defined:

Axis: Select the axis for the input-to-event table entry you want to display.

Input: Select the input edge for the input-to-event table entry you want to display.

- **Parameter Error**
  This area gives access to the last parameter error number for any axis. For a list of parameter error numbers and descriptions on each, see Parameter Error Values. These fields must be read only. This area has the following additional parameter that must be defined:

Axis: Select the axis whose last parameter error number you want to display.

- **Any Auto Stop Error**
  This area selects a special bit field for any axis. This field will be on when the axis has any error bit on in its Status word that also has any type of auto stop enabled for it. Otherwise it will be off. This field must be read only. This area has the following additional parameter that must be defined:

Axis: Select the axis whose error bit you want to display.

Write Locations

This area is used only for editable fields. If this section is disabled then ensure that the **Editable** check box on the **Format** tab is checked. For editable bit fields, this area is available, but the list cannot be edited. In this case, the one write location will automatically match the **Data to Display** area.

For editable integer fields, you will see a list of all locations that will be written to when the LCD420 user changes the value of this field. The first item in the list will always match the location entered in the Data to Display area.

To edit this list:

- In the **Write Locations** area, click **Edit**.

For details on editing this list see Using Fields with Multiple Write Locations.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.2.6 LCD Screen Editor: Format Tab Details

The field pane's **Format** tab is divided into several areas. Some areas may not be available for some types of fields. Specifically, the **Integer (16-bit) Format** area and **Decimal Places** text box are not available for bit fields, and the **Bit On Text** and **Bit Off Text** boxes are not available for integer fields.

Width box

Type or select the width of the field. Fields can be from 1 to 8 characters wide. The width must accommodate not only all digits you want to display, but also the decimal point and negative sign if necessary.

The width can also be changed from the screen pane, or through the **Field** menu commands. It is often useful to move the fields using these commands and their associated shortcut keys while editing the width. See Moving and Resizing Fields or Keyboard Shortcuts for details.

Decimal Places box (integer fields only)

Type or select the number of decimal places to have in the value. Type 0 for no decimal point. Using a decimal point causes a decimal point to be inserted into the integer value. For example, the value 4000 with three decimal places will be displayed as 4.000 and not as 4000.000.

Bit On Text box (bit fields only)

Type or select the text you want to display when the bit represented by the field is on. The text will always be aligned left, but you can add spaces to the beginning of the text to manually center or right-align the text.

Bit Off Text box (bit fields only)

Type or select the text you want to display when the bit represented by the field is off. The text will always be aligned left, but you can add spaces to the beginning of the text to manually center or right-align the text.

Integer (16-bit) Format area (integer fields only)

RMC 16-bit integer fields use different ranges. That is, some fields are unsigned, some are signed, and some are position units, which have a user-defined range. In all cases there are 65,536 possible values, but the range that is spanned by those values changes.

For most field types, the 16-bit range is fixed. In these cases, only one option will be available in this area. However, for a few fields, the range depends on other circumstances and cannot be set automatically. One example is the Command Value field. Its range depends on the Command field. In cases like this where you are given the option of selecting the integer format, you can choose from the following:

- **Signed (-32768 to 32767):** The range is fixed for a standard signed 16-bit integer.
- **Unsigned (0 to 65535):** The range is fixed for a standard unsigned 16-bit integer.
- **Position Units for Axis x:** The position unit range for the specified axis defines the range of this field. For example, suppose that you have a Go (G) command on step 7. That step is executed on axis 2. If you added an editable field for the Command Value of step 7, then you should select

Position Units for Axis 2 as the integer format. In cases where you have multiple axes that execute the same step, you can use the position units of either axis, provided they have similar position unit ranges.

Note: If you change the sign of the Offset parameter for an axis, you may need to re-download the LCD screens to ensure that the correct position unit range is used in fields that use the position unit format for that axis.

Sample area

For integer fields, this area shows the minimum and maximum values that can be displayed in the field, given the current width, number of decimal places, and integer format. It is important to check this area, as it is common to forget to reserve space for the decimal place or negative sign.

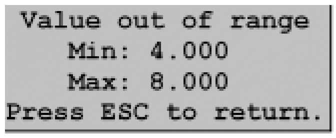
For bit fields, this area shows the On and Off text, as truncated by the field width. You can use this feedback to decide if you need to widen the field or insert spaces to center or right-align the text.

Editable area

Click to clear or select the **Editable** check box to change whether a field is editable or read only. This area is unavailable to fields that must be read only, such as status fields.

For editable integer fields, you also need to enter the range of values that can be entered in a field. Enter these values with the decimal places you have specified. For example, if you want the user to be limited to entering position units between 4000 and 8000, but this field is displayed with three decimal places, then enter limits of 4.000 and 8.000.

When an LCD420 user tries to enter a value outside of the edit limits, the value will not be used, and instead an error message will be displayed instructing the user of the field's limits. For example:



```
Value out of range
   Min: 4.000
   Max: 8.000
Press ESC to return.
```

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.2.7 LCD Screen Editor: Toolbar Details

The toolbar holds buttons for commonly used commands. To help identify these buttons, hover the pointer over a button. A ToolTip will pop up and a description will be displayed on the status bar. To show or hide the toolbar, on the **View** menu, click **Toolbar**.

The following table shows each toolbar button and the corresponding command issued:

Command		Equivalent Menu Action
🗎	**New**	On the **File** menu, click **New**.

Open	On the **File** menu, click **Open**.
Save	On the **File** menu, click **Save**.
Cut	On the **Edit** menu, click **Cut**.
Copy	On the **Edit** menu, click **Copy**.
Paste	On the **Edit** menu, click **Paste**.
Delete	On the **Edit** menu, click **Delete**.
Upload from Module	On the **Online** menu, click **Upload from Motion Controller**.
Download to Module	On the **Online** menu, click **Download to Motion Controller**.
Save to Flash	On the **Online** menu, click **Save to Flash**.
New Screen	On the **Insert** menu, click **New Screen**.
New Field	On the **Insert** menu, click **New Field**.

See Also: LCD Screen Editor Topics

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.2.8 LCD Screen Editor: Status Bar Details

The status bar is located at the bottom of the LCD Screen Editor window. This bar is divided into the following areas:

Menu Help - All of the status bar except the three panes described below is used to display help on menu items and toolbar buttons. When no menu item is selected, it displays "For help, press F1." If a menu item is selected, or the pointer is over a toolbar button, then a brief line of help is displayed here. This area is replaced by a progress bar during uploads and downloads.

CAP - This pane indicates whether the CAPS LOCK key is toggled on or off. If CAPS LOCK is enabled, this pane will display CAP. Otherwise it will be blank.

NUM - This pane indicates whether the NUM LOCK key is toggled on or off. If NUM LOCK is enabled, this pane will display NUM. Otherwise it will be blank.

INS/OVR - This pane indicates whether the LCD Screen Editor is in Insert (INS) or Overtype (OVR) mode. For further information on Insert and Overtype modes, see Selecting Insert or Overtype Mode.

To show or hide the status bar:

1. On the **View** menu, click **Status Bar**.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.3 Using the LCD Screen Editor

3.8.3.1 LCD Screen Editor: Using LCD Screen Files

The entire set of LCD screens can be saved and restored from disk files in the LCD Screen File (.lcd) format. It is not possible to save or restore individual screens or fields. Use the clipboard to copy these elements between LCD screen files. See Using the Clipboard for details.

The LCD Screen Editor always has exactly one LCD screen file open. Therefore, creating a new file, opening an existing file, or uploading a file from the RMC will overwrite the existing file. If you have made changes that have not been saved, you will be prompted to save the existing file before the operation completes. At this time, you will be given the option of canceling the operation.

The following file operations are available from the **File** menu:

New	Create a new LCD screen file with a single blank screen.
Open	Open an existing LCD screen file.
Save	Save the current LCD screen file to its current file name.
Save As	Save the current LCD screen file to a new file name.
Recently Used File	Open a recently used file. The file names of the four most-recently-used LCD screen files will be listed near the end of the **File** menu. Select the file you want to open.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.3.2 LCD Screen Editor: Uploading and Downloading LCD Screens

Note: Uploading and downloading are available only when RMCWin is online with an RMC CPU that has firmware supporting custom LCD screens. Standard firmware dated 20001204 or later has this feature.

The entire set of LCD screens can be downloaded to or uploaded from the RMC. It is not possible to upload or download individual screens or fields. Use the clipboard to copy these elements between LCD screen files. See Using the Clipboard for details.

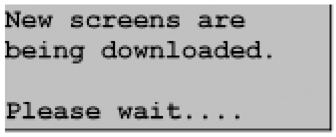
The set of screens is stored in the RMC memory. Therefore, the LCD420 display does not need to be attached to the RMC to upload and download LCD screens. However, downloading LCD screens to the RMC does not automatically save the LCD screens to Flash memory. To save the screens to Flash memory, click the Online menu, and then click **Save to Flash**, or use the corresponding toolbar button ( ). Notice that this will save all RMC parameters and tables to Flash, not just the screens. Without saving the LCD screens to Flash, the information will be lost when the RMC loses power.

To upload LCD screens from the RMC:

1. On the **Online** menu, click **Upload from Motion Controller**.
2. If you had made changes to the LCD Screen Editor's currently-open file, you will be prompted to save those changes. Click **Yes** to save the changes and proceed with the upload, click **No** to discard the changes and proceed with the upload, or click **Cancel** to cancel the upload request.
3. The screens will be uploaded from the RMC. This may take several seconds. The LCD420 can be used normally during this time.
4. After the upload is complete, if you had custom screen or field labels in your currently-open file and the uploaded screens have the same number of screens and fields, you will then be asked whether you want to keep the current screen and field labels. Click **Yes** to retain the labels, or click **No** to revert to the default labels.

To download LCD screens to the RMC:

1. On the **Online** menu, click **Download to Motion Controller**.
2. The screens will be downloaded to the RMC. This may take several seconds. During this time, the LCD420 will not respond to any input, and the following message will be displayed:



```
New screens are
being downloaded.

Please wait....
```

3. If you wish to save the screens to Flash memory, click the **Online** menu, and then click **Save to Flash**, or use the corresponding toolbar button ( ).

Note: Notice that this will save all RMC parameters and tables to Flash, not just the screens. The only Flash storage area that is not updated is for the splines.

See Also: LCD Screen Editor Topics

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.3.3 LCD Screen Editor: Using the Clipboard

In the LCD Screen Editor, the clipboard can reduce repetitive entry by allowing you to cut, copy, and paste screens, fields, and text to and from the clipboard.

The instructions below document using the **Edit** menu. You can also use shortcut menus or shortcut keys to further simplify these operations. See Keyboard Shortcuts for details.

To cut or copy a screen to the clipboard:

- In the tree pane, select the screen you want to place in the clipboard.
- On the **Edit** menu, click **Cut** or **Copy**.

To paste a screen from the clipboard:

- In the tree pane, select the screen that you want to insert the clipboard's screen in front of.
- On the **Edit** menu, click **Paste**.

To cut or copy a field to the clipboard:

- In the tree or screen pane, select the field you want to place in the clipboard.
- On the **Edit** menu, click **Cut** or **Copy**.

To paste a field from the clipboard:

- In the tree pane, select the screen into which you want to insert the clipboard's field. It will be pasted to the same location from which it was cut or copied.
- Or, to specify a location, position the insertion point in the screen pane where you want to paste the field. It will be pasted to this location.
- On the **Edit** menu, click **Paste**.

To cut or copy text to the clipboard:

- In the screen pane, select the block of text you want to place in the clipboard. This text may also contain fields.
- On the **Edit** menu, click **Cut** or **Copy**.

To paste text from the clipboard:

- In the screen pane, position the insertion point where you want to paste the text.
- On the **Edit** menu, click **Paste**.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.3.4 LCD Screen Editor: Changing the View Options

There are several options on the **View** menu that affect the appearance of the LCD Screen Editor. These settings are retained from one session to the next, but do not in any way affect the RMC and LCD420 behavior.

Status Bar and Toolbar

While most users find the status bar and toolbar to be useful, some may wish to remove them to use the their screen space for other purposes.

To show or hide the toolbar:

1. On the **View** menu, click **Toolbar**.

To show or hide the status bar:

1. On the **View** menu, click **Status Bar**.

Gridlines

The screen pane is a 4-row, 20-column text box. To help lay out text in this box, gridlines separating each character can be turned on or off.

To show or hide gridlines:

1. On the **View** menu, click **Gridlines**.

Screen Pane Font

The default font size for the screen pane was chosen to be fairly small to ensure that the window fit on all systems. You may want to increase the font size to take advantage of systems with larger display resolution.

To change the screen pane font:

1. On the **View** menu, click **Change Font**.
2. In the **Size** box, type or select the point size.
3. If necessary, click to clear or select the **Bold** check box.
4. Click **Apply** to see the effect of your changes, **OK** to accept the changes, or **Cancel** to cancel the changes.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.3.5 LCD Screen Editor: Keyboard Shortcuts

The following chart lists all keyboard shortcuts available in the LCD Screen Editor:

Press	**To**
CTRL+N	Close the current file and start a new file.
CTRL+O	Close the current file and open an existing file.
CTRL+S	Save the current file.
CTRL+X	Cut to the clipboard.
CTRL+C	Copy to the clipboard.
CTRL+V	Paste from the clipboard.
CTRL+A	In the screen pane, select all text.
CTRL+F	Add a new field at the current screen.
CTRL+W	Increase the current field's width by one character.
CTRL+Q	Decrease the current field's width by one character.
CTRL+UP ARROW	Move the current field up one line.
CTRL+DOWN ARROW	Move the current field down one line.
CTRL+LEFT ARROW	Move the current field left one character.
CTRL+RIGHT ARROW	Move the current field right one character.
INSERT	Toggle between Overtype and Insert modes.
DELETE	In the tree and screen panes, delete the current selection.
HOME	In the screen pane, jump to the beginning of the line.
END	In the screen pane, jump to the end of the line or text.
PAGE UP	In the screen pane, move to the previous screen.
PAGE DOWN	In the screen pane, move to the next screen.
F1	Jump to the help topics.
F2	In the tree pane, rename the screen or field.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.4 Using Screens

3.8.4.1 LCD Screen Editor: Adding and Removing Screens

Each LCD screen file can hold from one to sixteen screens. Therefore, you will often need to add and remove screens in each LCD screen file. In addition to the method described below, you can also add and remove screens using the clipboard. See Using the Clipboard for details.

To add a screen to an LCD screen file:

1. On the **Insert** menu, click **New Screen**. You can also use the shortcut menu on the root item in the tree pane.

To remove a screen:

1. In the tree pane, select the screen you wish to remove.
2. On the **Edit** menu, click **Delete**. You can also use the shortcut menu or shortcut key (DELETE).

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.4.2 LCD Screen Editor: Changing the Screen Order

The order of the screens in an LCD screen file is significant in the following ways:

- The first screen is the one that will be displayed when the RMC first starts up.
- The screen order determines the order used by the Previous Screen (¬) and Next Screen (®) keys on the LCD420.
- The screen order determines which keys are used to select which screens in the menu brought up by the MENU key on the LCD420.

There are three ways to move a screen up or down in the list of screens: use the **Move Screen Up/Down** menu commands, drag a screen in the tree pane, and use the clipboard.

To move a screen up or down using the **Move Screen Up/Down** commands:

1. Select the screen you want to move up or down.
2. On the **Edit** menu, click **Move Screen Up** or **Move Screen Down**. This will move the screen up or down one screen in the list. One or both of these commands may be unavailable if the screen is already at the top or bottom of the list.

To move a screen up or down by dragging the screen:

1. In the tree pane, drag the screen you want to move up or down. As you drag the screen, watch for a horizontal insertion line indicating where the screen will be inserted.

To move a screen up or down using the clipboard:

1. In the tree pane, select the screen you want to move up or down.
2. On the **Edit** menu, click **Cut**. You can also use the shortcut menu or shortcut key (CTRL+X).
3. In the tree pane, select the screen you want to insert the cut screen in front of.
4. On the **Edit** menu, click **Paste**. You can also use the shortcut menu or shortcut key (CTRL+V).

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.4.3 LCD Screen Editor: Editing Screen Text

Each screen consists of static text plus up to four fields, which change dynamically. The screen pane of the LCD Screen Editor is used to edit the static text. This text box behaves very similarly to other text boxes you have encountered in Windows applications.

Moving the Insertion Point

The following table summarizes the actions used to move the insertion point (cursor):

Press	To
UP ARROW	Moves the insertion point up one row.
DOWN ARROW	Moves the insertion point down one row.
LEFT ARROW	Moves the insertion point left one character.
RIGHT ARROW	Moves the insertion point right one character.
HOME	Moves the insertion point to the beginning of the current line.
END	Moves the insertion point to the end of the current line of text.
Click	Moves the insertion point to the character that was clicked.

Selecting a Block of Text

To select a block of text, use the SHIFT key in conjunction with any of the above. The selection extends from the original insertion point when the SHIFT key was first depressed to the current insertion point. Once text is selected you can cut or copy it to the clipboard or delete it.

You can also select all text in the screen pane. To select all text in the screen pane:

1. On the **Edit** menu, click **Select All**. You can also use the shortcut menu or shortcut key (CTRL+A).

Typing Text

To type text in the screen pane's text box, simply position the cursor and start typing. However, you will need to be aware of the difference between Insert and Overtype modes. For details on these modes, see Selecting Insert or Overtype Mode.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.4.4 LCD Screen Editor: Selecting Insert or Overtype Mode

The LCD Screen Editor is always in one of two modes: Insert or Overtype.

In Insert mode, text that is typed or pasted from the clipboard is inserted. That is, any text following the insertion point is shifted to allow for the new text. Similarly, when text is removed—whether because a selection is deleted or cut to the clipboard—it is deleted, so that any text to the right of the removed text is shifted to fill the gap left by the removed text.

In Overtype mode, text that is typed or pasted from the clipboard overwrites text in the screen pane. The text does not shift. Similarly, when a text block is removed, that block is erased, but no text shifts. This does not apply to using the DELETE and BACKSPACE keys to delete a single character. The text to the right of the insertion point will shift accordingly.

The current mode is shown in two ways. First, on the status bar, an indicator will display INS for Insert mode and OVR for Overtype mode. Second, the insertion point in the screen pane will be a blinking vertical line in Insert mode and a blinking solid block in Overtype mode.

To switch between these two modes, press the INSERT key. The LCD Screen Editor starts up in Overtype mode.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.4.5 LCD Screen Editor: Renaming Screens

The tree pane lists screens and fields. The screen names are set to Screen x by default, where x is the number of the screen (0-15). The field names are set by default based on the fields' content. For example, Axis 0 Actual Position is one possible default field name.

It is possible to rename both screens and fields. However, these names are not downloaded to the RMC. Therefore, uploading LCD screens from an RMC will revert back to the default names.

To rename a screen:

1. In the tree pane, select the screen.
2. On the **Edit** menu, click **Rename**.
   You can also use the shortcut menu, shortcut key (F2), or click again on the screen name in the tree pane to start the rename command.
3. Type in the new name.
4. Press ENTER.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.5 Using Fields

3.8.5.1 LCD Screen Editor: Adding and Removing Fields

Each LCD screen consists of static text plus up to four fields, which change dynamically. There are several ways to add and remove fields. In addition to the methods described here, the clipboard can also be used to add, remove, copy, and move fields. See Using the Clipboard for details.

Adding a Field

All methods for adding a field involve issuing the **New Field** command from the **Insert** menu. This command can be issued directly from that menu, through shortcut menus, or through the shortcut key CTRL+F. Listed below are different contexts that this command can be issued in and a description of how the field is added:

Context	**Action**
A screen or field is selected in the tree pane.	A field is added to the selected screen at the current insertion point in the screen pane.
The screen pane has the input focus, and no text is selected.	A field is created starting at the current insertion point with a default length of five characters.
The screen pane has the input focus, and a block of text is selected.	A field is created in the place of the selection. Therefore, the selection determines both the position and length of the field.
The screen pane has the input focus, no text is selected, but the insertion point is	If no text is selected, the LCD Screen Editor will check to see if the text around the insertion point is of the form ##.###, where each # represents a digit and the period represents a decimal place in the number. Therefore,

positioned by one or more '#' characters.	you can quickly add a field by typing ###.### and then pressing CTRL+F. This will add a field at the given location, with a length of seven characters, and with three decimal places.

Removing a Field

To remove a field:

1. Select the field you wish to remove in either the screen or tree pane.
2. On the **Edit** menu, click **Delete**. You can also use the shortcut menu or shortcut key (DELETE).

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.5.2 LCD Screen Editor: Moving and Resizing Fields

Fields can be positioned anywhere in the screen and can be from 1 to 8 characters wide. The screen pane is used to show the position and size of the fields. It is also used to move and resize the fields, either through commands on the **Field** menu or by pointer drag operations.

To move a field using the **Movement** commands:

1. In the tree or screen pane, select the field to move.
2. On the **Field** menu, click **Move Up**, **Move Down**, **Move Left**, or **Move Right**. You can also use the shortcut menu or shortcut keys (CTRL+UP ARROW, CTRL+DOWN ARROW, CTRL+LEFT ARROW, and CTRL+RIGHT ARROW).

To move a field by dragging:

1. Position the pointer over the field you wish to move. Ensure that the pointer changes to the move pointer.
2. Click and drag the field to its new location. A shadow will show the new location until the mouse button is released.

To change a field size using the **Increase/Decrease Width** commands:

1. In the tree or screen pane, select the field to resize.
2. On the **Field** menu, click **Increase Width** or **Decrease Width**. You can also use the shortcut menu or shortcut keys (CTRL+W and CTRL+Q).

To change a field size by dragging:

1. Position the pointer over the left or right edge of the field you wish to resize. Ensure that the pointer is a double arrow to indicate you can resize that edge.

2. Click and drag the field's edge to its new size. A shadow will show the new field size until the mouse button is released.

You can also change a field's size from the field pane's **Format** tab. See Format Tab Details.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.5.3 LCD Screen Editor: Editing Field Properties

Each LCD screen can have up to four dynamically updated fields. Fields can be read only or read/write. Examples of read only fields are actual position and drive output. Examples of read/write fields are requested speeds and positions that can be viewed and changed if desired.

The field pane has two tabs for editing the properties of each field: the **Data** and **Format** tabs. For details on any settings in these tabs, see the following topics:

- Data Tab Details
- Format Tab Details

For information on making fields editable, see the following topic:

- Using Editable Fields

See Also: LCD Screen Editor Topics

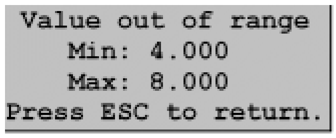
Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.5.4 LCD Screen Editor: Using Editable Fields

Fields can be read only or editable. There are three elements that control the behavior of an editable field:

- **Editable or Read Only.**
  You must first specify when a field is editable or not. This is done in the field pane's **Format** tab using the **Editable** check box. Some fields such as status fields cannot be editable. For these fields, the **Editable** check box will be unavailable.

- **Edit Limits.**
  For editable integer fields, you must specify the range of values that can be accepted. The edit limits are defined on the field pane's **Format** tab in the **Editable** area. When an LCD420 user tries to enter a value outside of the edit limits, the value will not be used, and instead an error message will be displayed instructing the user of the field's limits. For example:



```
Value out of range
    Min: 4.000
    Max: 8.000
Press ESC to return.
```

- **Write Locations.**
  A write location is a register in the RMC that is changed when a new value is entered for a field. When a field is first made editable, it will automatically have a single write location that matches the **Data to Display** area in the field pane's **Data** tab. For most fields this is all that is required. However, editable integer fields can have more than one write location.

For details on this advanced topic, see Using Fields with Multiple Write Locations.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.5.5 LCD Screen Editor: Using Fields with Multiple Write Locations

Overview

A write location is a register in the RMC that is changed when a new value is entered for a field. When a field is first made editable, it will automatically have a single write location that matches the **Data to Display** area in the field pane's **Data** tab. For most fields this is all that is required. However, editable integer fields can have more than one write location.

For example, suppose an Event Step sequence has two steps that each start a move. You want the speeds of each move to be the same, but you also want that speed to be editable from the LCD420. In this situation, you would first set up a field to display the first step's speed register. You would make this field editable and give it appropriate limits. Next, a second write location—specifically, the second step's speed register—will make editing this single field change the speed in both steps.

Write Location Limitations

There are several limitations to keep in mind when using multiple write locations:

- The first item in the Write Locations list is always the same as the **Data to Display** area of the field pane's **Data** tab. That is, it is neither possible nor desirable to have the value displayed for a field not correlate to the value that is edited. Therefore, changing the **Data to Display** area will automatically change the first **Write Locations** list item, and similarly changing the first **Write Locations** list item will automatically change the **Data to Display** area.

- When a value is entered in a field with multiple write locations, one value is written per control loop. For example, suppose a field writes to both Step 1 Speed and Step 2 Speed. When a value is entered in this field, the value will be first written to Step 1 Speed, and one control loop later (1 or 2 ms) will be written to Step 2 Speed. This is usually not important, but may be in some circumstances.

- The RMC supports up to a total of 32 fields having more than one write location. That is, you can have as many editable fields as you want, except that only 32 of these fields can have more than one write location. The **Edit Write Locations** dialog box lists the total multi-location fields used. See the discussion below on this dialog box.

- The RMC supports a total of 256 write locations among all multi-location fields. That is, the sum of all write locations of only those editable fields with more than one write location cannot exceed 256. The **Edit Write Locations** dialog box lists the total multi-location field locations used. See the discussion below on this dialog box.
- Each editable field can have up to 128 total write locations. However, the total number of write locations of all multi-location fields still must be under the 256-location limit. The **Edit Write Locations** dialog box lists the total write locations for the field. See the discussion below on this dialog box.

Editing the Write Location List

The following procedures describe how to modify the **Write Locations** list.

To start editing the write locations list:

1. Ensure that the **Data to Display** area indicates an integer field that can be edited, and that the **Editable** check box on the field pane's **Format** tab is selected.
2. In the field pane, click the **Data** tab.
3. Under **Write Locations**, click **Edit**. This will open the **Edit Write Locations** dialog box.

To add a write location:

1. Open the **Edit Write Locations** dialog box as described above.
2. Under **Write Locations**, click **New**.

This command will be unavailable if any of the limits are exceeded. See the discussion below on checking that you have not exceeded a limit.

3. Under **Location Detail**, select the appropriate settings for the desired write location. See Data Tab Details for additional information on setting up these fields.

To remove a write location:

1. Open the **Edit Write Locations** dialog box as described above.
2. In the **Write Locations** list, select the write location you wish to remove.
3. Under **Write Locations**, click **Remove**.

This command will be unavailable if you only have one write location left. You must keep at least one write location.

To edit a write location:

1. Open the **Edit Write Locations** dialog box as described above.
2. In the **Write Locations** list, select the write location you wish to remove.
3. Under **Location Detail**, change the settings for this write location. See Data Tab Details for additional information on setting up these fields.

To check the write location limits:

1. Open the **Edit Write Locations** dialog box as described above.
2. You will find the three numerical limits in the following locations. Each limit takes the form "n of m (p)," where n is the number currently used, m is the maximum number allowed, and p is the percentage of the maximum currently used:
   - **Total multi-location fields.**
     Under **Multi-Location Memory**, look at **Total Fields Used**.
   - **Total write locations in multi-location fields.**
     Under **Multi-Location Memory**, look at **Total Locations Used**.
   - **Write locations for this field.**
     Look at the **Write Locations** list heading.

To complete changes to the write location list:

1. Click **OK**.
2. If you changed the first write location in the list, you will be prompted to confirm that you want to change the **Data to Display** area of the field pane's **Data** tab. This is necessary because it is neither possible nor desirable to have the value displayed for a field not correlate to the value that is edited. Click **Yes** to accept your write-location-list edits and change the **Data to Display** area to match the first write location. Click **No** to return to the **Edit Write Locations** dialog box.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.8.5.6 LCD Screen Editor: Renaming Fields

The tree pane lists screens and fields. The screen names are set to Screen x by default, where x is the number of the screen (0-15). The field names are set by default based on the fields' content. For example, Axis 0 Actual Position is one possible default field name.

It is possible to rename both screens and fields. However, these names are not downloaded to the RMC. Therefore, uploading LCD screens from an RMC will revert back to the default names.

To rename a field:

1. In the tree pane, select the field to rename.
2. On the **Edit** menu, click **Rename**.
   You can also use the shortcut menu, shortcut key (F2), or click again on the field name in the tree pane to start the rename command.
3. Type in the new name.
4. Press ENTER.

See Also: LCD Screen Editor Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9 Curve Tool

3.9.1 Curve Tool: Overview

The Curve Tool in RMCWin is used to view and edit spline curves that can then be run on the RMC. Spline curves are defined by a sequence of points. Each point is a position that the axis must be at when a certain time is reached. These times can be actual time or they can be based on the position of a master axis. The RMC then generates a smooth curve that fits through this sequence of points. With the introduction of Enhanced curves (requires RMC CPU firmware dated 20010208 or newer), the user has more control over how the RMC generates this curve. For example, individual line segments can be made linear, and the velocity can be set to an exact value at each point.

For more information on spline curves, see the Spline Overview topic.

To open the Curve Tool:

1. On the **Tools** menu, click **Curve Tool**.

For further information on the Curve Tool, select one of the following topics:

Screen Elements

- Screen Elements
- Graph View
- Detail Window
- Spreadsheet View
- Toolbar
- Status Bar

Using the Curve Tool

- Units of Measurement
- Using Curve Files
- Mouse Commands
- Keyboard Shortcuts

Using the Graph View

- Graph View Options

- Showing Velocity and Acceleration
- Adjusting the Scales
- Using the Grid
- Using the Scale Bars
- Changing the Orientation
- Zooming In and Out
- Scrolling

Using the Spreadsheet View

- Spreadsheet View Options
- Selecting Cells
- Editing Cells
- Deleting Cells
- Cutting and Copying Cells
- Pasting Cells
- The Insertion Point
- Resizing columns

Using Curves

- Selecting Which Curves to Display
- Selecting the Active Axis
- Copying Curves between Axes
- Importing and Exporting Curves
- Uploading and Downloading Curves
- Converting a Plot to a Curve
- Erasing a Curve
- Curve Properties and Editing Options
- Curve Axis Labels
- Curve Limits
- Standard vs. Enhanced Curves

- Auto Repeat Curves
- Enforcing Limits
- Linking Curves

Using Points

- Selecting Points
- Determining a Point's Exact Location
- Adding Points
- Deleting Points
- Point Properties
- Moving Points
- Changing a Point's Velocity
- Selecting Linear or Cubic Segments
- Expanding or Contracting Points

3.9.2 Screen Elements

3.9.2.1 Curve Tool: Screen Elements

The Curve Tool is divided into the following elements:

- **Graph View:** The Graph view is the main work area of the Curve Tool window. It displays the curves for all selected axes and allows editing the curves in a graphical format. For details on the Graph view, see Graph View.
- **Detail Window:** The Detail window can be displayed in the Graph view. It always has an associated hairline that can be positioned anywhere on the Graph view. The Detail window then displays the Time, Position, Velocity, and Acceleration of the active curve at the hairline. For details on the Detail window, see Detail Window.
- **Spreadsheet View:** The Spreadsheet view displays the currently active curve in a spreadsheet format rather than the graphical format as in the Graph view and allows editing the points of the active curve. For details on the Spreadsheet view, see Spreadsheet View.
- **Toolbar:** The toolbar holds buttons for commonly used commands. To help identify these buttons, hover the pointer over a button and a ToolTip will pop up and a description will be displayed on the status bar. For details on the toolbar, see Toolbar.
- **Status Bar:** The status bar holds the current pointer location in horizontal and vertical units, an Enforce Limits indicator, a Valid/Invalid Curve indicator, a Hints Available indicator, and a line of text to display descriptions of the currently-selected command or toolbar button. For details on the status bar, see Status Bar.

Changing the Layout

The following will modify the layout of these window elements:

- Resize the window. Drag the border or the sizing handle in the lower-right corner of the window to resize the window.
- Show, hide, or reposition the Detail window. See Detail Window for more information.
- Show, hide, or reposition the Spreadsheet view. See Spreadsheet View for more information.
- Show or hide the toolbar and status bar. On the **View** menu, click the **Toolbar**, **Status Bar**, or **Detail Window** commands to toggle each screen element.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.2.2 Curve Tool: Graph View

The Graph view is the main work area of the Curve Tool window. It displays the curves for all selected axes and allows editing the curves in a graphical format. This section of the Curve Tool window has many sub-elements; each is described below:

Background Color

- **Gray.** Points cannot have negative times. Therefore, the region of the screen for negative time is shown with a light-gray background. Even if an axis is geared to a master axis that has negative position units, the negative region in the curve tool is not used. There are two reasons for this. First, when a curve is based on a master axis, that curve always starts at zero master units. However, zero master units is defined as the position the master is at when the curve is first followed. Second, if the master axis does move backward and therefore would go negative, then the axis following the curve will hold at the first position in the curve until the axis goes forward, unless Auto Repeat is enabled. See Auto Repeat Curves for details.
- **Yellow.** Positions beyond the extend and retract limits of the active axis are shown with a yellow background. See Curve Limits for details on these limits.
- **White.** The remaining region is shown with a white background. Therefore, all points in this region have positions within the extend and retract limits of the active axis, and have times greater than or equal to zero.

Lines

- **Hairline.** If the Detail window is turned on, then you will see a black hairline at the time (or master position) currently reflected in the Detail window. This hairline can be dragged with the pointer. See Detail Window for more information.
- **Points.** Each point on all visible axes is shown with a small dot. For the active axis, the dots are red. For inactive axes, they are black.
- **Position/Time Plot.** Each visible axis will have a curve drawn between the points. The active

curve will be red; all other curves will be black.

- **Velocity Plot.** The active curve can have its velocity plotted. See Showing Velocity and Acceleration for details. The color of this curve defaults to magenta, but can be changed by the user.
- **Acceleration Plot.** The active curve can have its acceleration plotted. See Showing Velocity and Acceleration for details. The color of this curve defaults to cyan (light blue), but can be changed by the user.
- **Scale Bars.** Time (or master position), position, velocity, and acceleration can each have a scale bar. The scale bars have minor and major tick marks. The value at each major tick mark is displayed next to it. Each major tick mark can also be dragged to independently adjust each scale. Scale bars can be turned off. See Using the Scale Bars for details.
- **Zero Line.** When either velocity or acceleration is plotted and the scale bars are displayed, then a line is displayed representing zero velocity and/or acceleration. This line defaults to gray, but the color can be changed by the user. See Using the Scale Bars and Showing Velocity and Acceleration for details.
- **Velocity Limits.** When velocity is plotted, then the positive and negative velocity limits of the active axis are shown by a dotted line the same color as the velocity plot, which defaults to magenta. See Curve Limits and Showing Velocity and Acceleration for details.
- **Acceleration Limits.** When acceleration is plotted, then the positive and negative acceleration limits of the active axis are shown by a dotted line the same color as the acceleration plot, which defaults to cyan (light blue). See Curve Limits and Showing Velocity and Acceleration for details.
- **Grid.** A grid can be displayed on the Graph view. This can be in the form of dotted lines, dots, or crosses. See Using the Grid for details.
- **Selection Box.** When a range of points is selected using the pointer and a selection box as described in Selecting Points, then a dotted box with resize handles will be displayed around the selected points. This selection can then be used for Moving Points and Expanding and Contracting Points.

See Also: Graph View Topics and Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.2.3 Curve Tool: Detail Window

The Detail window allows the user to position a hairline in the Graph view and see the exact time (or master position) where the hairline is located and the position, velocity, and acceleration of the active curve at that time or master position. Therefore, the Detail window and the hairline are always linked together: if the Detail window is displayed, then the hairline will be as well.

See Units of Measurement for details on the units used by the values displayed in the Detail window.

To show or hide the Detail window:

1. On the **View** menu, click **Detail Window**.

You can also show or hide the Detail window by right-clicking in the Graph view, and then clicking **Show Detail Window** on the shortcut menu.

To reposition the hairline with the keyboard:

1. Ensure that the hairline and Detail window are displayed as described above.
2. Press CTRL+SHIFT+UP ARROW or CTRL+SHIFT+LEFT ARROW to move the hairline toward 0 time (or master position), or press CTRL+SHIFT+DOWN ARROW or CTRL+SHIFT+RIGHT ARROW to move the hairline away from 0 time (or master position).

The longer you hold down these keys the faster the hairline will move.

To reposition the hairline with the mouse:

1. Ensure that the hairline and Detail window are displayed as described above.
2. Position the pointer over the hairline, and click and drag the hairline to its new location.

To move the Detail window:

1. Drag the Detail window to its new location. It cannot be moved outside the Graph view area.

The Detail window will also automatically move out of the way of the hairline.

To temporarily show the Detail window:

1. Position the pointer at the time (or master position) where you want to display the hairline.
2. Hold down the SHIFT key and click and hold the left button.

As long as the button is held down, the hairline and Detail window will be displayed. Moving the mouse while the left button is held down will move the hairline.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.2.4 Curve Tool: Spreadsheet View

The Spreadsheet view is an area of the curve tool screen that presents the curve points in spreadsheet format rather than graphically. The Spreadsheet view can either be hidden or positioned along the top, right side, bottom, or left side. The curve that is active in the Graph view is displayed in the Spreadsheet view. Changing which axis is active in the Graph view changes which axis is displayed in the spreadsheet.

Changes made to a point's properties in the Spreadsheet view are immediately reflected in the active curve in the Graph view. Similarly, changes made in the Graph view are immediately reflected in the Spreadsheet view. The Spreadsheet view also allows points to be added to the currently active curve. The spreadsheet shows the same properties as the Point properties dialog box, but for all points on the curve; see Point Properties for more information on each property.

To show or hide the Spreadsheet View:

1. On the **View** menu, point to **Spreadsheet**, and then click the position to display the spreadsheet in or select **Hide** to remove it.

To reposition the Spreadsheet View:

1. On the **View** menu, point to **Spreadsheet**, and then click the new position for the spreadsheet.

See Also: Spreadsheet View Topics and Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan
Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.2.5 Curve Tool: Toolbar

The toolbar holds buttons for commonly used commands. To help identify these buttons, hover the pointer over a button and a ToolTip will pop up and a description will be displayed on the status bar.

The following table shows each toolbar button and the corresponding command issued:

	Command	Equivalent Menu Action
	New	On the **File** menu, click **New**.
	Open	On the **File** menu, click **Open**.
	Save	On the **File** menu, click **Save**.
	Cut	On the **Edit** menu, click **Cut**.
	Copy	On the **Edit** menu, click **Copy**.
	Paste	On the **Edit** menu, click **Paste**.
	Upload from Module	On the **Online** menu, click **Upload from Motion Controller**.
	Download to Module	On the **Online** menu, click **Download to Motion Controller**.
	Save Splines to Flash	On the **Online** menu, click **Save Splines to Flash**.
	Convert Plot to Curve	On the **Online** menu, click **Convert Plot to Curve**.

0 ... 7	**Display nth Axis Curve**	On the **View** menu, click **Display Axis**, and then click the desired axis.
Active Axis: Axis0	**Active Axis**	On the **View** menu, click **Active Axis**, and then click the desired axis.
	Zoom In	On the **View** menu, click **Zoom In**.
	Zoom Out	On the **View** menu, click **Zoom Out**.
	Fit to Screen	On the **View** menu, click **Fit to Screen**.

To show or hide the toolbar:

1. On the **View** menu, click **Toolbar**.

See Also: Curve Tool Topics

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.2.6 Curve Tool: Status Bar

The status bar is located at the bottom of the Curve Tool window. This bar is divided into the following areas:

Menu Help -	The entire status bar except the panes described below is used to display help for menu items and toolbar buttons. When no menu item is selected, it displays "For help, press F1." If a menu item is selected or the pointer is over a toolbar button, then a brief line of help is displayed there. This area will be replaced by a progress bar during uploads and downloads.
Position - X: 8384, Y: -15	This pane gives the time (or master position) and position that the pointer is over in the Graph view. This is useful to give a rough approximation of positions. The value labeled X is the horizontal location, and Y is the vertical location.
Icons -	This pane has the following three icons. Each has two states as shown below:

	Limits are Enforced
	Limits are Not Enforced

These icons indicate whether or not the curve limits are being enforced when the axis is modified through the Graph view. Double-clicking this icon will toggle this feature on and off. See Enforcing Limits for details.

Valid Curve

Invalid Curve

These icons indicate whether the curve is currently within the limits or not. The limits checked include extend position, retract position, maximum speed, maximum velocity, and master units between points. See Curve Limits for details. If the curve is not within all limits, then the curve cannot be downloaded to the RMC.

No Hints Available

Click for Hints

When the curve is invalid, then this light bulb icon will be lit, indicating that hints are available as to why the curve is invalid. Click on this icon and hold the button down to read the hints.

To show or hide the status bar:

1. On the **View** menu, click **Status Bar**.

See Also: Curve Tool Topics

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.3 Using the Curve Tool

3.9.3.1 Curve Tool: Units of Measurement

Four quantities are conveyed in the Curve Tool: time (or master position), position, velocity, and acceleration. These terms can be somewhat confusing since the curves created by the curve tool can be followed with respect to time or with respect to another axis. Therefore, this topic is devoted to helping clarify the use of these terms and define the units used by each:

Time (Master Units)
By default this quantity is labeled **Time (sms)** and refers to actual time in short milliseconds

(sms). A short millisecond is 1/1024th of a second. This term is used because, when the RMC follows a curve based on time, it processes one time unit every 976 microseconds, or 1024 time units per second. However, in many applications this quantity actually refers to a master position. Therefore, this label can be changed. For example, a general label that fits most geared curve applications would be **Master Pos (pu)**. For details on changing this label, see Curve Axis Labels.

Position (Slave Units)

By default this quantity is labeled **Position (pu)** and refers to position units of the axis following the spline. In applications in which the curve is being followed with respect to another axis, it may be useful to change the label to help differentiate it from the axis it is geared to. Therefore, this label can be changed. For example, a general label that fits most geared curve applications would be **Slave Pos (pu**). For details on changing this label, see Curve Axis Labels.

The units for this field will always be position units, as defined for the axis from the main RMCWin screen and used throughout the RMC product.

Velocity

This quantity appears on the velocity scale bar, in the Detail window, in the data tips, in the **Point Properties** dialog box, and in the **Options** dialog box under the **General** tab for setting the speed limit. The label for this quantity cannot be changed.

The speed units are position units per 1024 time units. Therefore, if a curve is followed with respect to time, each time unit is 1/1024th of a second, so the speed units are position units per second.

If the curve is followed with respect to another axis, then the speed units are less clear: the speed will be slave position units per 1024 master position units. For example, a 1:1 gear ratio would have a speed of 1024.

Acceleration

This quantity appears on the acceleration scale bar, in the Detail window, in the data tips, in the **Point Properties** dialog box, and in the **Options** dialog box under the **General** tab for setting the acceleration limit. The label for this quantity cannot be changed.

The acceleration units are position units per 1024 time units per 1024 time units. Therefore, if a curve is followed with respect to time, each time unit is 1/1024th of a second, so the acceleration units are position units per second per second.

If the curve is followed with respect to another axis, then the acceleration units are less clear: the acceleration will be slave position units per 1024 master position units per 1024 master position units.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.3.2 Curve Tool: Using Curve Files

Curves can be saved and restored from disk files in the Curve (.crv) format. Curves from all axes are saved and restored, even if they are currently not visible. It is not possible to save individual curves or points.

The Curve Tool always has exactly one open curve file. Therefore, creating a new file or opening an existing file will overwrite the current curve data, and uploading a curve from an RMC axis will overwrite the curve on that axis in the open file. If you have made changes that have not been saved, you will be prompted to save the current file before the operation completes or to cancel the operation.

The following file operations are available from the **File** menu:

- **New**
  Create a new Curve file with no initial curves.
- **Open**
  Open an existing Curve file.
- **Save**
  Save the current Curve file to its current file name.
- **Save As**
  Save the current Curve file to a new file name.
- **Recently Used File**
  Open a recently used file. The file names of the four most-recently-used Curve files will be listed near the end of the File menu. Select the file you want to open.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.3.3 Curve Tool: Mouse Commands

In addition to the mouse commands listed below, any movement of the pointer over the Curve Tool will cause the current pointer location to be updated in the status bar.

Mouse Commands

Operation	**Pointer Over**	**Result**
Hover	Point	Display a data tip with the point's coordinates, velocity, and acceleration.
Click	Point	Select the point.
Click	Spreadsheet Cell	Select the cell and cancel any previous selection.

SHIFT+Click	Point	Select all points between the first point selected and the one clicked.
SHIFT+Click	Spreadsheet Cell	Select all cells between the previously selected cell and the one clicked.
CTRL+Click	Point	Toggle whether the point is selected or not.
CTRL+Click	Spreadsheet Cell	Add the cell to the selection.
Double-Click	Point	Open the **Point Properties** dialog box.
Double-Click	Graph View	Add a point to the active curve.
Double-Click	Spreadsheet Cell	Enter edit mode for that cell.
Right Click	Anywhere	Open a shortcut menu for the selected object(s).
Drag	Graph View	Create a selection box. When the button is released, all points within the selection box will be selected.
Drag	Selection box resize handle	Expand and/or collapse all selected points.
Drag	Hairline	Reposition the hairline for the Detail window.
Drag	Scale bar major mark	Independently adjust the time (or master position), position, velocity, or acceleration scale.
Drag	Point or selection box	Drag the selected point or points.
Drag	Spreadsheet Cell	Selects a range of cells.
ALT+Drag	Point or selection box	Drag the selected point(s), but keeps the position constant.
CTRL+Drag	Point or selection box	Drag the selected point(s), but keeps the time (or master position) constant.
SHIFT+Drag	Graph View	Position the hairline for the Detail window. If the Detail window is not displayed, it will be temporarily displayed until the button is released.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.3.4 Curve Tool: Keyboard Shortcuts

Shortcuts available in both Graph and Spreadsheet views:

Press	To
CTRL+N	Close the current file and start a new file.
CTRL+O	Close the current file and open an existing file.
CTRL+S	Save the current file.
CTRL+X	Cut the active curve to the clipboard.
CTRL+C	Copy the active curve to the clipboard.
CTRL+V	Paste a curve from the clipboard over the active curve.
CTRL+A	Select all points on the active curve.
CTRL+Arrow Keys	Move all selected points in the direction pressed.
INSERT	Prompt the user for coordinates of a new point to insert.
CTRL+DELETE	Erase the active curve.
F1	Jump to the help topics.
ALT+S	Display or hide the Spreadsheet view.
F6	Switch between the Graph view and the Spreadsheet view.

Shortcuts available in the Graph view:

Press	To
CTRL+SHIFT+Arrow Keys	Move the hairline, if the Detail window is displayed.
Arrow Keys	Change the current selected point(s).
HOME	Select the first point on the active curve.
END	Select the last point on the active curve.
DELETE	Delete all selected points.
ESC	Cancel a drag or de-select all points.
ENTER	Display properties for the currently selected point(s).

ALT+ENTER	Display properties for the currently selected point(s).
TAB	Switch the active curve to the next displayed axis.
SHIFT+TAB	Switch the active curve to the previous displayed axis.
*	Fit all curves to Screen.
+	Zoom In.
-	Zoom Out.
C	Pan the location under the pointer to the center of the screen.

Shortcuts available in the Spreadsheet view:

Press	**To**
SHIFT+Arrow Keys	Select a range of cells.
Arrow Keys	Select a new cell.
HOME	Select the first cell in the spreadsheet.
END	Select the last cell in the spreadsheet.
TAB	Select the next cell.
SHIFT+TAB	Select the previous cell.
DELETE	Delete cells or selected points; see Deleting Cells for details.
ESC	Cancel an edit when editing or clear the selection when not editing.
ENTER	Start editing the currently selected cell or complete an edit.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.4 Using the Graph View

3.9.4.1 Curve Tool: Graph View Options

The **Graph** tab of the **Options** dialog box allows the user to customize the Graph view.

To view or change the Graph view options:

1. On the **Tools** menu, click **Options**.

2. Click the **Graph** tab.
3. Change any options on this page.
4. Click **OK**.

The **Graph** tab has the following sections:

- **Additional Plot Values**
  This section controls the plotting of velocity and acceleration. From this section you can turn on or off the velocity and acceleration plots, set the color of the velocity and acceleration plots, and set the zero line color. The zero line is a line drawn at zero velocity and acceleration. See Showing Velocity and Acceleration for details.
- **Graph Scale**
  This section allows you to set the scale of the time (or master position), position, velocity, and acceleration plots. Increasing each value will zoom in. Decreasing a value will zoom out. See Adjusting the Scales for details.
- **Grids**
  This section controls whether there is a grid, and if there is a grid, what type of grid is displayed. See Using the Grid for details. This area also lets you turn the scale bars on or off. See Using the Scale Bars for details.
- **Orientation**
  This section controls whether time (or master position) is displayed as increasing from left to right or from top to bottom. You can use this setting to better match your application's orientation. For example, parison profiles are typically viewed vertically, so using the vertical orientation may be clearer in that application. See Changing the Orientation for details.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.4.2 Curve Tool: Showing Velocity and Acceleration

For all selected curves, the curve is drawn as position with respect to time (or master units). You can also have the Curve Tool plot the velocity and/or acceleration with respect to time (or master position). These plots are shown on top of the position plot.

A zero line is drawn at the midline of the Graph view. This line is not affected by scrolling the Graph view.

The velocity and acceleration plots always apply to the active curve. Therefore, switching to another active axis will cause these plots to be redrawn for the new axis.

Velocity and acceleration plots can be turned on and off. You can also select the colors used for these plots and the color of the zero line. These steps are described below.

There are two ways of changing the velocity and acceleration scales. You can manually enter a scale value, as described in Adjusting the Scales, and you can drag the scale bars, as described in Using the Scale Bars.

In addition to the velocity plot and scale bar, dotted lines are drawn at the positive and negative speed limits in the same color as the velocity plot. Similar lines are drawn at the acceleration limits. See Curve Limits for details.

To show or hide a velocity or acceleration plot:

1. On the **Tools** menu, click **Options**.
2. Click the **Graph** tab.
3. Under **Additional Plot Values**, use the **Show Velocity on Graph** and **Show Acceleration on Graph** check boxes to turn these plots on or off.
4. Click **OK**.

   You can also turn these plots on or off by right-clicking in the Graph view, and then clicking **Show Velocity** or **Show Acceleration** on the shortcut menu.

To change the colors used by the velocity or acceleration plot:

1. On the **Tools** menu, click **Options**.
2. Click the **Graph** tab.
3. Under **Additional Plot Values**, click the **Change Color** button to the right of the color swatch you want to change. The following list describes what is affected by each color:
   - **Velocity Color:** Velocity scale bar, plot, and limit lines.
   - **Acceleration Color:** Acceleration scale bar, plot, and limit lines.
   - **Zero Line Color:** Zero line for velocity and acceleration.
4. Select the color you want to use.
5. Click **OK** in the **Color** dialog box.
6. Click **OK** in the **Options** dialog box.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.4.3 Curve Tool: Using the Grid

The Graph view has the following options for displaying a grid:

- **No Grid.** No grid is displayed.
- **Dots.** Light gray dots are drawn at each division.
- **Crosses.** Light gray crosses are drawn at each minor division and dark gray crosses are drawn at each major division.

- **Lines.** Light gray, dotted lines are drawn at each minor division and dark gray, dotted lines are drawn at each major division.

The position and time scales determine the grid spacing. The spacing of the dots, crosses, or lines corresponds to the major and minor tick marks on the scale bars. Both the grid and tick mark spacing is automatically chosen by the Curve Tool to give natural divisions (for example, 100, 200, 500, 1000). Therefore, as the scales are changed, the units per grid will change if necessary to keep reasonable grid spacing.

To change the grid type:

1. On the **Tools** menu, click **Options**.
2. Click the **Graph** tab.
3. Under **Grid**, use the **Show Grid** check box to turn the grid on or off.
4. If **Show Grid** is checked, click the **Dots**, **Crosses**, or **Lines** option button to select a grid type.
5. Click **OK**.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.4.4 Curve Tool: Adjusting the Scales

Each quantity (time, position, velocity, and acceleration) has an adjustable scale. Use the scale bars and grid to help keep track of the units. See Using the Scale Bars and Using the Grid for details.

Use any of the following methods to adjust the scale:

- Enter the scales manually in the **Options** dialog box. This method is described below.
- Drag a scale bar's major tick mark. See Using the Scale Bars for details.
- Use the **Zoom In**, **Zoom Out**, and **Fit to Screen** commands. See Zooming In and Out for details.

To enter new scale values in the **Options** dialog box:

1. On the **Tools** menu, click **Options**.
2. Click the **Graph** tab.
3. Under **Graph Scale**, type new values for the scales you want to change.

Increasing a value will zoom in. Decreasing a value will zoom out.

4. Click **OK**.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.4.5 Curve Tool: Using the Scale Bars

Each quantity being plotted can have an associated scale bar displayed. These scale bars help give the user an idea of the approximate positions, time (or master positions), velocities, and accelerations shown in the Graph view. These scale bars can be turned off to clean up the Graph view. Also, the major tick marks on each scale bar can be dragged to adjust its scale.

The scale bars are drawn on top of the Graph view, but are otherwise transparent. That is, you can still edit a curve that extends beyond the scale bars.

To show or hide the scale bars:

1. On the **Tools** menu, click **Options**.
2. Click the **Graph** tab.
3. Under **Grids**, use the **Show Scale Bars** check box to show or hide the scale bars.

This affects the scale bars of all plotted quantities, including velocity and acceleration, if they are turned on. See Showing Velocity and Acceleration for details.

4. Click **OK**.

You can also turn the scale bars on or off by right-clicking in the Graph view, and then clicking **Show Scale Bars** on the shortcut menu.

To adjust a scale by dragging a major tick mark:

1. Ensure that scale bars are shown, using the steps above if necessary to show them.
2. Position the pointer over a major tick mark on the scale bar of the quantity whose scale you want to adjust. The major tick marks are raised to indicate they can be dragged. The pointer will also change when over a major tick mark.
3. Drag the major tick mark toward the base point to zoom out and away from the base point to zoom in. The base point is defined as follows for each quantity:

   - **Position and Time Scales:** The base point is where the position and time (or master position) scale bars cross.
   - **Velocity and Acceleration:** The base point is where the scale bar you are dragging crosses the zero line.

You will find that it is usually easiest to drag the tick mark that starts closest to the base point when you want to zoom in and to drag the tick mark that starts farthest from the base point when you want to zoom out.

If while dragging a major tick mark, you are unhappy with your changes, press ESCAPE to cancel the scale change.

4. Release the button to finalize the scale change.

Note: This is not the only way to change the scales. You can also change the scale using the **Options** dialog box (see Adjusting the Scales for details) or using the **Zoom In**, **Zoom Out**, and **Fit to Screen** commands described in Zooming In and Out.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.4.6 Curve Tool: Changing the Orientation

Graphs are usually displayed with the independent variable horizontal—increasing from left to right—and the dependent variable vertical—increasing from bottom to top. This is how the Curve Tool defaults to display the curve: time (or master position) is the independent variable, and the position (or slave position) is the dependent variable.

However, some applications such as parison drops are accustomed to a different orientation, with the time (or master position) increasing from top to bottom and the position increasing from left to right.

The Curve Tool supports both orientations. The first is called **Horizontal Time**, and the second is called **Vertical Time**.

To change the Graph view orientation:

1. On the **Tools** menu, click **Options**.
2. Click the **Graph** tab.
3. Under **Orientation**, click either the **Horizontal Time** or **Vertical Time** option button.
4. Click **OK**.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.4.7 Curve Tool: Scrolling

It is often difficult or undesirable to fit the entire curve or set of curves in the Graph view at once. Therefore, you will need to scroll at times to view other parts of your curves. The Curve Tool supports the standard scroll bars, but also offers several special ways to scroll to help improve your productivity.

Scrolling in the time (or master position) direction in the Graph View will affect the velocity and acceleration plots if they are enabled (see Showing Velocity and Acceleration for details), but scrolling in the position direction will not affect these plots, since their zero line is fixed at the center of the Graph view window.

It is expected that the user knows how to use the standard Windows scroll bars, but the additional scrolling methods are described below:

- **Centering on the Pointer in the Graph View**
  If you can see the location you want to work on, then you can center on that location with a single keystroke.

  To center on the pointer:

  1. Position the pointer over the location on the Graph view that you want centered.
  2. Press C. This will scroll the display to center on the pointer.

- **Scroll While Zooming in the Graph View**
  Similar to centering on the pointer, you can also use the **Zoom In** and **Zoom Out** commands from the keyboard to zoom in or out and center on the pointer at the same time. See Zooming In and Out for details.

- **Auto Scrolling in Both Views**
  When you are dragging points or the selection box in the Graph view, or dragging to select cells in the Spreadsheet view, then approaching the edges of the Curve Tool will start the view scrolling automatically. This is useful to move points farther than or select cells beyond what can be seen with the current view.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.4.8 Curve Tool: Zooming In and Out

There are three ways to change the position and time (or master position) scales:

- Enter the scales manually in the **Options** dialog box. See Adjusting the Scales for details.
- Drag a scale bar's major tick mark. See Using the Scale Bars for details.
- Use the **Zoom In**, **Zoom Out**, and **Fit to Screen** commands. This method is described below.

The Zoom In, Zoom Out, and Fit to Screen commands all adjust both the time (or master position) and position scales simultaneously. You must use one of the other methods listed above if you want to adjust the scales independently, or if you want to adjust the velocity or acceleration scales.

To zoom in or out on the center of the screen:

1. On the **View** menu, click **Zoom In** or **Zoom Out**, or use the corresponding toolbar buttons 🔍+ and 🔍− respectively.

While zooming in and out with the above method is useful, a more powerful way to zoom and pan simultaneously is to combine the pointer and keyboard or mouse wheel.

To zoom in or out centered on the pointer:

1. Position the pointer on the portion of the Graph view you want to center on.
2. Press the PLUS SIGN (+) on the numeric keypad to zoom in or the MINUS SIGN (-) on the numeric keypad to zoom out. You can also hold down the CTRL key and scroll the mouse wheel up to zoom in or down to zoom out.
3. Each time the Graph view is zoomed in this manner, the pointer will be positioned in the center of the Curve Tool so further zooms will not cause unexpected panning.

To fit all visible curves on the screen:

1. Ensure that you have all curves that you want to fit on the screen marked as visible. See Selecting Which Curves to Display for details.
2. On the **View** menu, click **Fit to Screen**, or use the corresponding toolbar button . You can also use the ASTERISK (*) key on the numeric keypad to issue this command.

See Also: Curve Tool Topics

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.5 Using the Spreadsheet View

3.9.5.1 Curve Tool: Spreadsheet View Options

The **Spreadsheet** tab of the **Options** dialog box allows the user to customize the Spreadsheet view.

To view or change the spreadsheet options:

1. On the **Tools** menu, click **Options**.
2. Click the **Spreadsheet** tab.
3. Change any options on this page.
4. Click **OK**.

The **Spreadsheet** tab has the following sections:

- **Location**
  This section allows the user to show or hide the spreadsheet and to select the position of the spreadsheet. If the spreadsheet is not displayed, the position options are not available. To set the spreadsheet position check the **Show Spreadsheet** option.
- **Column Widths**
  This section allows the user to set the width of the columns on the spreadsheet. When the spreadsheet is displayed along the left or right side, it is in "vertical" mode and all four columns can be sized individually. When the spreadsheet is displayed along the top or bottom, the spreadsheet is in "horizontal" mode and all columns have the same width. The values in this

section are displayed in inches. Click **Set All To Default** to set all column widths to the default values.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.5.2 Curve Tool: Selecting Cells

The spreadsheet allows for single cell selections as well as selecting ranges of cells. While there are many combinations of cell selections, not all features are available with all selections. For more details on what features are available when selecting ranges see Editing Cells, Deleting Cells, Cutting and Copying Cells, and Pasting Cells.

To select an individual cell:

1. Click on the cell in the Spreadsheet view with the mouse. Do not hold down any keys while selecting the cell.

   You can also use the keyboard's arrow keys to select a cell. Again, do not hold down any other keys (such as CTRL or SHIFT) while performing this action.

To select a range of cells:

1. Move the pointer to the desired starting cell and click and hold the left mouse button down while dragging the pointer to the last cell to be selected.

   You can also use the keyboard's arrow keys or a combination of the keyboard and mouse to select a range of cells. To use both the keyboard and mouse, select the starting cell using the mouse and hold the SHIFT key down and click the last cell. To use the arrow keys, move the selected cell to the desired location without holding down any other keys. Now hold down the SHIFT key and move to the last cell in the range.

To select multiple individual cells or cell ranges:

1. Hold down the CTRL key while selecting a cell or cell range with the mouse. Continue selecting cells or cell ranges while holding down the CTRL key.

To select columns or rows:

1. Click on the header column or row. To select more than one column or row hold the left mouse button down while dragging the mouse across the header columns or rows. To select multiple individual columns or rows, use the CTRL key as described above while clicking on the headers.

When a cell is selected, that point is considered selected in the Graph view. Selecting all the properties for a single point selects the entire point in the Spreadsheet view. This mode is handled differently when deleting; see Deleting Cells for more information.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.5.3 Curve Tool: Editing Cells

The Spreadsheet view allows for editing individual cells as well as editing cell ranges. In order to edit a cell or range of cells, the cell or cells must first be selected; see Selecting Cells for details. The cell that is editable is highlighted by a black box. When editing a cell range or multiple ranges, the value in the editable cell will be used for all the selected cells.

To edit a cell:

1. Select the cell(s) to be edited.
2. Press ENTER to start editing.
   You can also double-click the cell or press the F2 key to start editing.
3. Edit the cell's value.
4. Press ENTER when done.

To quickly enter a value in a cell:

1. Select the cell(s) to be edited.
2. Type a new value for the cell.
3. Press ENTER when done.

To cancel an edit already in progress:

1. Press the ESC key.

Changing the selected cell or pressing the ENTER key accepts the edit and updates the point in the Graph view. Pressing ESC cancels any edit in progress. The point's properties are not changed when the edit is canceled.

When editing a range, all cells in the range are set to the single value accepted into the editable cell. For example, to set all the velocities to 100, select all the velocity cells, type '100', and then press ENTER. All the cells in the range will be updated.

Changing a point's time, or master position, property may cause the point to be moved in the order. For example, if there are three points, at time 100, 200, and 300, changing the point whose time value is 200, to a value of 400 will change the order of the points to be 100, 300, 400. The Graph view will immediately reflect the new order. This method can be used to rearrange points along the curve.

Editing a velocity cell causes the cell to have a fixed velocity, see Changing a Point's Velocity for more information.

Note: Since the time, or master position, property cannot be set to a value within 10 units of another point, editing a range will fail if more than one time cell is included in the range. Additionally, if the Interval Type property is included in a selection range along with any other property types the edit will fail since the Interval Type property does not accept numeric values and the other property types do not accept letters.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.5.4 Curve Tool: Deleting Cells

In the Spreadsheet view, individual cells and entire points can be deleted. Only the Velocity property can be deleted individually since a point's time, position, and interval type cannot be deleted apart from deleting the entire point. Deleting the velocity value sets the point to dynamic velocity; see Changing a Point's Velocity for more details on Fixed and Dynamic Velocities.

To delete a velocity cell:

1. Select the velocity cell or range of velocity cells.
2. On the **Edit** menu, click **Delete**.

You can also press the DELETE key or use the shortcut menu.

To delete an entire point:

1. Select all the properties for that point. The easiest way to do this is to click the header for that point.
2. On the **Edit** menu, click **Delete**. (The point will be removed from the curve.)

To delete an entire point using the shortcut menu:

1. Right click on the header cell for the point.
2. On the shortcut menu, click **Delete**. (The point will be removed from the curve.)

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.5.5 Curve Tool: Cutting and Copying Cells

Cell values can be copied to the clipboard for pasting into other cells or into external programs. Cutting cell values initially copies the values to the clipboard and then deletes the cells; see Deleting Cells for more information. Only cells which can be deleted can be cut.

Cutting and copying work on individual cells and cell ranges. When cutting or copying with multiple cell ranges selected, the ranges must be the same in one or more dimensions of the selection. That is, they must all contain the same number of rows, but can contain a different number of columns. Or they must all contain the same number of columns, but can contain a different number of rows. If the selection is invalid, cut and copy will not be available.

To cut or copy cell values:

1. Select the cell(s) to be copied; see Selecting Cells.
2. On the **Edit** menu, click **Cut** or **Copy** as desired.

If the **Cut** and **Copy** commands are not available on the menu, try changing the selection and repeating the steps above. If **Copy** is available and **Cut** is not, then the selection probably includes cells which cannot be deleted. Either change the selection or use **Copy**.

Note: When copying the Interval Type property, a 0 is copied for "Cubic" and a 1 is copied for "Linear". This only appears when pasting to an external application.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.5.6 Curve Tool: Pasting Cells

Values copied to the clipboard either from the spreadsheet or from external programs can be pasted into the spreadsheet. The values must be valid for the curve selected and for the properties being pasted into. Values can be pasted into individual cells or into selection ranges. If the dimensions of the values to be pasted do not match the selection range, a warning message will be displayed. To paste the values regardless of the dimensions select a single cell as the starting point of the paste.

To paste values into the Spreadsheet:

1. Copy the values to be pasted; see Cutting and Copying Cells for more information.
2. Select the cell(s) to receive the values copied; see Selecting Cells for more information.
3. On the **Edit** menu, click **Paste**.

If the paste command is not available, then the selection is not valid for pasting or there is no data in the clipboard to be pasted. Try changing the selection and repeating the steps above.

If the dimensions of the data in the clipboard and the selection in the spreadsheet will require additional points to be added, then the values will be pasted into the Insertion Point—see The Insertion Point for more information—otherwise the values will be overwritten; see Editing Cells for more details.

Note: When pasting the Interval Type property, a 0 is used for "Cubic" and a 1 is used for "Linear". This is only used when pasting from an external application.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.5.7 Curve Tool: The Insertion Point

The Spreadsheet view always shows one more row or column than there are points in the Graph view. This is called the "Insertion Point." This column allows for new points to be entered through the spreadsheet. Since at a minimum a point must have a time, or master position, and a position value, these properties must be entered before a point is actually added to the curve.

Note: The time, or master position, value determines where in the curve the point will be placed. To insert a point between two existing points, enter a time value which falls between the two existing points' time values. For example, if there is a point at 100 and a point at 300, adding a point at time value 200 will insert the point at that location on the curve.

The Insertion Point is indicated by an "asterisk" (✳) in the header for that point.

See Also: Curve Tool Topics

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.5.8 Curve Tool: Resizing columns

The columns in the spreadsheet can be resized to increase or decrease the width of the column. When the Spreadsheet is displayed vertically along the right or left side of the curve tool, the columns in the spreadsheet can be resized individually. That is, each column can have a different size.

When the Spreadsheet is displayed horizontally along the bottom or top of the curve tool, the columns in the spreadsheet must all be the same size. That is, resizing one column will cause all the columns to be resized to the same width.

To change a column's width:

1. On the **Tools** menu, click **Options**.
2. In the **Options** dialog box, click the **Spreadsheet** tab.
3. Type new widths for the column(s) to be changed in inches, or click **Set All To Default**, to restore the column widths to the default settings.
4. Click **OK** to apply the new settings.

If the value specified is smaller than the minimum column width allowed, an error message will be displayed and the column width will be set to the minimum value.

To Change a column's width using the mouse:

1. Move the pointer over the right edge of the cell whose width you want to change.
2. When the pointer becomes a ↔, drag the cell divider left or right to decrease or increase the column's width.

See Also: Curve Tool Topics

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.6 Using Curves

3.9.6.1 Curve Tool: Selecting Which Curves to Display

All axes in an RMC can have curves associated with them except for auxiliary pressure axes. However, because it can be confusing to display multiple curves simultaneously, you can control which axes are displayed in the Graph view.

You can view or change which axis curves are displayed using either the toolbar or the menu. Making a curve not visible does not delete the curve. It can be made visible again and will be saved to disk with any visible curves.

To view or change which axis curves are displayed using the toolbar:

1. Find the toolbar buttons with numbers (0 through 7).

These numbers represent axes 0 through 7 in the RMC. Depressed buttons represent axes that are currently being displayed. Raised buttons represent axes that are currently not being displayed. Unavailable buttons are either non-existent axes or axes that cannot support curves.

2. Click the button of an axis to add or remove it from the axes that are displayed.

To view or change which axis curves are displayed using the menu:

1. On the **View** menu, click **Display Axis**.

All axes are listed in the pop-up menu. Checked axes are currently being displayed. Unchecked axes are not being displayed. Unavailable axes are either non-existent or cannot support curves.

2. Click the name of an axis to add or remove it from the axes that are displayed.

See Also: Curve Tool Topics

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.6.2 Curve Tool: Selecting the Active Axis

Only one axis is active at any given time. The name of the active axis is displayed in the toolbar, as in the following example toolbar section:

The active curve is special in the following ways:

- Its name is displayed in the toolbar under **Active Axis**. See Toolbar for details.
- The active curve is plotted in red, whereas all other curves are plotted in black.
- The velocity and acceleration plots, if enabled, correspond only to the active axis. See Showing Velocity and Acceleration for details.
- The extend and retract limits of the active axis are indicated by displaying the out-of-limits regions with a yellow background. See Curve Limits for details.
- The Detail window shows values from the active axis only. See Detail Window for more information.
- The active axis is the one used by the **Copy, Paste, Cut, Erase Curve, Upload from Motion Controller, Download to Motion Controller,** and **Convert Plot to Curve commands**.
- The keyboard methods of selecting and moving points only affect the active axis. See Selecting Points and Moving Points for details.
- Only points on the active axis can be selected by dragging a selection box. See Selecting Points for details.
- The **Valid/Invalid Curve** and **Hint** icons on the status bar reflect the active axis only. See Status Bar for details.
- Only the active curve is displayed in the Spreadsheet view. See Spreadsheet View for details.

The following methods can be used to change the active axis:

- On the toolbar, in the **Active Axis** list, select the name of the axis you want to make active. If the axis you want to make active is not listed, then you must either first make it visible—as described in Selecting Which Axes to Display—or use the menu method described below.
- On the **View** menu, click **Active Axis**, and then click the name of the axis you want to make active. If the axis you select was not being displayed, then it will be made visible.
- When in the Graph view, press TAB to make the next visible axis the active axis, or SHIFT+TAB to make the previous visible axis the active axis.
- Click on any point on the axis you want to make active.

See Also: Curve Tool Topics

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.6.3 Curve Tool: Copying Curves between Axes

The clipboard can be used in the Curve Tool to copy curves between axes as well as import and export curves to other applications such as Microsoft Excel. For further details on importing and exporting curves, see Importing and Exporting Curves.

To copy a curve to the clipboard:

1. Select the axis of the curve you want to copy to the clipboard, as described in Selecting the Active Axis.
2. On the **Edit** menu, click **Cut** or **Copy**.

You can also use the corresponding toolbar buttons (and) or the CTRL+X and CTRL+C shortcut keys to issue these commands.

To paste a curve from the clipboard:

1. Select the axis that you want to paste the curve in the clipboard to, as described in Selecting the Active Axis.
2. On the **Edit** menu, click **Paste**.

You can also use the corresponding toolbar button () or the CTRL+V shortcut key to issue this command.

If there is an existing curve on this axis, then you will be prompted whether you want to overwrite that curve. If you answer yes, then the existing curve will be replaced by the one on the clipboard.

To copy cells from the Spreadsheet view to the clipboard, see Cutting and Copying Cells for more information.

See Also: Curve Tool Topics

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.6.4 Curve Tool: Importing and Exporting Curves

It is possible to import and export curves to and from the Curve Tool. It is most common to use this feature in conjunction with Microsoft Excel. Importing and exporting is done with the clipboard. The steps below describe Microsoft Excel specifically, but other applications will behave similarly.

To export a curve to Microsoft Excel:

- In the Curve Tool, do the following:

1. Select the axis of the curve you want to export, as described in Selecting the Active Axis.
2. Click the Graph view to make it the active view.
3. On the **Edit** menu, click **Cut** or **Copy**.

   You can also use the corresponding toolbar buttons (and) or the CTRL+X and CTRL+C shortcut keys to issue these commands.

   Both will place the curve on the clipboard, but Cut will erase the curve from the Curve Tool.

   You can also copy cells from the Spreadsheet view to the clipboard. See Cutting and Copying Cells for more information.

- Open a Microsoft Excel worksheet and do the following:

1. Select the top-left cell you want to paste the curve into.
2. On the **Edit** menu, click **Paste**.

   You can also use the corresponding toolbar button () or the CTRL+V shortcut key to issue this command.

   The curve will be pasted in two columns. The first column will hold the times (or master positions) of all points in the curve. The second column will hold the positions of all points in the curve. Notice that velocities and linear/cubic segment information is not exported.

To import a curve from Microsoft Excel:

- Open a Microsoft Excel worksheet and do the following:

1. Type two columns of numbers. Each row represents a point, with the first column being the time (or master position) and the second column being the position.
2. Select this two-column block of cells.
3. On the **Edit** menu, click **Copy** to place the data on the clipboard.

   You can also use the corresponding toolbar button () or the CTRL+C shortcut key to issue this command.

- In the Curve Tool, do the following:

1. Select the axis of the curve you want to import into, as described in Selecting the Active Axis.
2. Click the Graph view to make it the active view.
3. On the **Edit** menu, click **Paste**.

   You can also use the corresponding toolbar button () or the CTRL+V shortcut key to issue this command.

   If there is an existing curve on this axis, then you will be prompted whether you want to overwrite that curve. If you answer yes, then the existing curve will be replaced by the one on the clipboard.

You can also paste data from an external program into the Spreadsheet view. See Pasting Cells for more information.

See Also: Curve Tool Topics

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.6.5 Curve Tool: Uploading and Downloading Curves

Note: Uploading and downloading are available only when RMCWin is online with an RMC CPU that has firmware supporting curves. Firmware dated 19971204 or later supports this feature.

Note: Downloading enhanced curves requires RMC CPU firmware dated 20010208 or later. See Standard vs. Enhanced Curves for details.

The Curve Tool allows downloading or uploading curves to or from the RMC. It will only upload or download a curve to a single axis. Downloading a curve to an axis from RMCWin will erase all existing curves on that axis in the RMC. You should not download a curve while that axis is running a spline. Uploading a curve from an axis that has more than one spline segment will upload the first, or oldest, curve. It is only possible to have multiple spline segments per axis by downloading them through the communication module such as Ethernet, Modbus Plus, Serial, or PROFIBUS. See Downloading Splines to the RMC for details.

Each curve that is downloaded is stored in the RAM memory. Downloading a curve does not automatically save the curve to Flash memory. The Curve Tool offers a command to save all curves in the RMC to Flash memory. This command is described below. Without saving the curves to Flash, the curves will be lost when the RMC loses power.

To upload a curve from the RMC:

1. Select the axis you want to upload, as described in Selecting the Active Axis.
2. On the **Online** menu, click **Upload from Motion Controller**, or use the corresponding toolbar button ().
3. The curve currently in the RMC for the active axis will be uploaded to the Curve Tool.

To download a curve to the RMC:

1. Select the axis of the curve you want to download, as described in Selecting the Active Axis.
2. On the **Online** menu, click **Download to Motion Controller**, or use the corresponding toolbar button ().
3. If the curve you are downloading is an Enhanced curve, but the RMC does not have firmware that supports Enhanced curves, then you will be given two options. First, you can convert the curve to a Standard curve and proceed with the download. Second, you can cancel the download.
4. The curve for the active axis will be downloaded to the RMC.

5. If you want to save this and all other splines to Flash memory, then proceed with the steps below for saving all splines to Flash.

To save all curves in the RMC to Flash:

1. On the **Online** menu, click **Save Splines to Flash**, or use the corresponding toolbar button ( ).
2. Wait until the Flash operation is complete, as is indicated by the progress window. If power to the RMC is lost while the Flash operation is in progress, the RMC will lose both old and new curves.

See Also: Curve Tool Topics

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.6.6 Curve Tool: Converting a Plot to a Curve

Many applications that require the Curve Tool also require training the Curve Tool as to what the curves should look like. For example, some applications use a joystick to move some axes through a sequence, and then want to have the RMC play back that sequence. Other applications may have an existing control system that needs to be retrofitted with an RMC-equipped system, so the RMC should copy the existing movements.

The RMC excels in these applications. You will first need to capture a plot of the move you want to convert to a curve, then convert the plot to a curve, and finally clean up the curve. Each phase is described below:

To capture a plot for converting to a curve:

1. Set the Plot Time for the axis to indicate the number of seconds that the move you want to capture will take. Round up to the next largest whole number (or even number if you have RMC CPU firmware with a 2 ms control loop).
2. Initiate the plot capture at the same time you start the motion on the axis.

The most common way to start the plot capture in this situation is to issue a Start Graph (y) command on the axis that you want to capture.

You may want to use the Event Step table to issue this command to tightly couple the start of the graph with the start of the motion.

To convert a plot to a curve starting with the Plot window open:

1. Open a Plot window for the axis of the curve you want created, as described in Opening a Plot Window.
2. Either upload a plot from the RMC, or open a saved plot, as described in Reading Plot Data from the Motion Controller and Saving and Restoring Plots.
3. On the **Data** menu, click **Convert Plot to Curve**.

The Curve Tool will be opened or brought to the foreground. You will be prompted for the conversion interval.

4. In the **Convert Plot to Curve** dialog box, type the number of milliseconds that you want between each point in the curve.

The shorter this interval, the more difficult it will be to edit the curve later and the higher the accelerations will be, but the more accurately you will follow the curve. The longer this interval, the easier it will be to edit the curve, but it may not following the original plot as well. It is worth trying several conversion intervals to find the optimal interval.

5. Click **OK**.

To convert a plot to a curve starting with the Curve Tool open:

1. Open the Curve Tool.
2. Select the axis that has a plot you want to convert to a curve, as described in Selecting the Active Axis.
3. On the **Online** menu, click **Convert Plot to Curve**, or use the corresponding toolbar button ().

The current plot will be read from the active axis. This will take a few seconds.

4. In the **Convert Plot to Curve** dialog box, type the number of milliseconds that you want between each point in the curve.

The shorter this interval, the more difficult it will be to edit the curve later and the higher the accelerations will be, but the more accurately you will follow the curve. The longer this interval, the easier it will be to edit the curve, but it may not follow the original plot as well. It is worth trying several conversion intervals to find the optimal interval.

5. Click OK.

Tips for Cleaning up a Curve

- Avoid the temptation of specifying a short time between each point in the **Convert Plot to Curve** dialog box. This can make it nearly impossible to edit the curve and will often make the move jerkier by having so many segments with different accelerations.
- Delete points in sections that have few inflections. This will help make these segments smoother with very little affect on the accuracy of the curve.
- If a section of the curve is supposed to be linear, then delete all points in that interval, and instead set the point at the beginning of that section to be linear. See Selecting Linear or Cubic Segments for details.
- You might want to try drawing your curve from scratch, using the uploaded curve only as a guideline. This is the best way to get the most accurate and smooth curve. In the steps below, we will assume that axis 0 is the axis that you read the plot from and on which you want the final curve, and that axis 1 is the temporary axis:
    - Convert a plot to a curve on axis 0. You can use a relatively short conversion interval when using this method to get the most accurate copy of the curve.
    - Copy this curve from axis 0 to axis 1. See Copying Curves between Axes for details on

copying curves from one axis to another.

- Ensure that both axes 0 and 1 are displayed. See Selecting Which Curves to Display for details.
- Select axis 0 as the active axis, as described in Selecting the Active Axis.
- Delete the curve on axis 0, as described in Erasing a Curve.
- Add points to axis 0 to create a curve that matches the profile of the uploaded curve on axis 1.
- You will need to create the new curve slightly above or below the uploaded curve to avoid grabbing points on axis 1 accidentally.
- Add as few points as you can to match the uploaded curve. Drag the points as necessary to help match the curve.
- Press CTRL+A to select all points in the axis 0 curve.
- Drag the entire axis 0 curve until it is over the uploaded curve on axis 1.
- Drag individual points in the axis 0 curve to match the axis 1 curve if differences show up after overlaying the curves.
- Delete the curve on axis 1.

See Also: Curve Tool Topics

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.6.7 Curve Tool: Erasing a Curve

To erase a curve:

1. Select the axis of the curve you want to erase, as described in Selecting the Active Axis.
2. On the **Edit** menu, click **Erase Curve**.

You can also use the shortcut key (CTRL+DELETE) to issue this command.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.6.8 Curve Tool: Curve Properties and Editing Options

The **General** tab of the **Options** dialog box allows you to view or change curve properties and editing options.

To view or change these options:

1. On the **Tools** menu, click **Options**.
2. Click the **General** tab.
3. Change any options on this page.
4. Click **OK**.

The General tab has the following settings:

- **Axis Settings**

This section has several settings for each axis. Axes that cannot have curves—such as axes that do not exist in the current RMC configuration and auxiliary pressure axes—will be unavailable. The following settings are available for each axis:

  - **Axis Labels (Time and Position)**

  Any text can be typed into these labels for Time and Position. These labels are used whenever the curve tool refers to one of these dimensions. See Curve Axis Labels for details.

  - **Limits (Speed and Accel)**

  Each axis has a maximum speed and acceleration. Use these text boxes to change these limits. For details on these values, see Curve Limits.

  - **Enhanced Curve**

  Use the check boxes in this column to select whether the curve on the given axis is Standard or Enhanced. See Standard vs. Enhanced Curves for details.

  - **Auto Repeat**

  These checkboxes enable or disable the Auto Repeat feature for the curve on each axis. This feature is only available for Enhanced curves. See Auto Repeat Curves for details.

- **Editing Options**

The following options apply to editing all curves:

  - **Enforce Axis Limits**

  Check this box to enforce extend, retract, velocity, acceleration, and time (or master position) difference limits when editing curves in the Graph view. See Enforcing Limits for details.

  - **Link Curves Together**

  Check this box to link together points with identical time (or master position) values on all visible axes. This feature is useful when the curves from two or more axes need to be synchronized. See Linking Curves for details.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.6.9 Curve Tool: Curve Axis Labels

Any text can be typed into these labels for Time and Position. These labels are used whenever the curve tool refers to one of these dimensions. For example, the Detail window, data tips, and the **Point Properties** dialog box all use these labels.

By default these labels are **Time (sms)** and **Pos (pu)**. These labels work well if the curve will be followed with respect to time and not geared to another axis. In this case, sms refers to short milliseconds. There are 1024 sms in one second. This term is used because the RMC's control loop is processed 1024 times per second.

If the curve will be followed with respect to another axis, then the default labels do not work as well. In that case, labels such as **Master Pos (pu)** and **Slave Pos (pu)** more accurately describe the situation.

To change a curve's axis labels:

1. On the **Tools** menu, click **Options**.
2. Click the **General** tab.
3. Find the text boxes on the row of the axis you want to change under the columns labeled **Axis Labels**, **Time** and **Position**.
4. Type the labels you want to use in these text boxes.
5. Click **OK**.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.6.10 Curve Tool: Curve Limits

Each curve has a number of limitations. Some limit values can be changed, while others cannot. Most of these limitations can be temporarily ignored by disabling the Enforce Limits option as described in Enforcing Limits. Each limit is described below.

The status bar has three icons related to limits. See Status Bar for a description of these icons.

Note: The curve cannot be downloaded to the RMC if any of the limits have been violated, even if limits are not being enforced.

Curve Limits

- **Time (or master position) Between Points**
  A point cannot be closer than ten (10) time units or farther than 65,535 time units from an adjacent point. These limits are fixed. Placing points closer than 10 time units is the one limitation that cannot be overridden by disabling the **Enforce Limits** option.

- **Position**
  The curve cannot extend beyond the axis Extend and Retract Limits. These limits cannot be changed from within the Curve Tool, but they can be changed from the main RMCWin window, even while the Curve Tool is open. These limits can be ignored while editing a curve by disabling the **Enforce Limits** option.

The Extend and Retract Limits for the active axis are indicated in the Graph view by showing the area beyond the limits with a yellow background.

- **Velocity**
  Each curve has a maximum speed assigned to it. This maximum speed can be changed from the **General** tab of the **Options** dialog box in the Curve Tool as described below. The maximum speed must be between 0 to 65,535. The default is 65,535 speed units. See Units of Measurement for a definition of speed units. This limit can be ignored while editing a curve by disabling the **Enforce Limits** option.

If the velocity plot is visible, then the positive and negative limits will be drawn as dotted lines in the same color as the velocity plot.

- **Acceleration**
  Each curve has a maximum acceleration assigned to it. This maximum acceleration can be changed from the **General** tab of the **Options** dialog box in the Curve Tool as described below. The maximum acceleration must be between 0 to 400,000. The default is 400,000 acceleration units. See Units of Measurement for a definition of acceleration units. This limit can be ignored while editing a curve by disabling the **Enforce Limits** option.
  If the acceleration plot is visible, then the positive and negative limits will be drawn as dotted lines in the same color as the acceleration plot.

To change the speed and/or acceleration limits of a curve:

1. On the **Tools** menu, click **Options**.
2. Click the **General** tab.
3. Find the text boxes on the row of the axis you want to change under the columns labeled **Speed** and **Accel Limits**.
4. Type the values you want using the units described above.
5. Click **OK**.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.6.11 Curve Tool: Standard vs. Enhanced Curves

The RMC supports two types of curves: Standard and Enhanced. Each is described below.

Standard Curves

Standard curves were introduced with RMC CPU firmware dated 19971204. This gave the RMC the capability of following curves defined by cubic splines. The user defines the position and time (or master position) of each point in the curve. The velocities of the endpoints are fixed at zero (0). The velocities of all other points and accelerations of all points are computed by the RMC to find the smoothest curve through all points.

There are several deficiencies with Standard curves:

- Because the accelerations are computed at the endpoints, it is likely that they will be non-zero and therefore result in an acceleration discontinuity and relatively high jerk at each endpoint.

- It is difficult to accurately approximate a linear segment.
- It is difficult to set and maintain a given velocity at a given point.
- The velocities at the endpoints could not be changed from zero (0).

These issues prompted the introduction of Enhanced curves.

Enhanced Curves

Enhanced curves were introduced with RMC CPU firmware dated 20010208. They greatly enhance the capabilities of the RMC in following an arbitrary curve, and further enhance the smoothness of the curve.

The following items are introduced with enhanced curves:

- The accelerations at the endpoints of a curve are always zero (0). Therefore, there is no acceleration discontinuity and therefore much lower jerk at the end points. The acceleration is continuous over the entire curve, even with the additional features listed below.
- Segments can be easily designated as linear segments, which makes them exactly linear. This greatly simplifies camming and flying cut-off applications.
- Any point, including the endpoints, can have its velocity set to a specific value. This, too, simplifies camming and flying cut-off applications.
- Curves can be set up to auto-repeat, meaning that the curve restarts automatically when it reaches its end. See Auto Repeat Curves for details.

The drawback of Enhanced curves is that they can currently only be downloaded to the RMC through RMCWin, and not through communications modules such as Ethernet, PROFIBUS, and Modbus Plus.

You should always use Enhanced curves unless you are running firmware prior to 20010208 and do not wish to upgrade, or you need to download curves through the communication module.

To toggle a curve between Enhanced and Standard:

1. On the **Tools** menu, click **Options**.
2. Click the **General** tab.
3. Find the check box on the row of the axis you want to change under the column labeled **Enhanced Curve**. To make the curve for this axis Enhanced, set this check box. To make this curve Standard, clear this check box.
4. Click **OK**.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.6.12 Curve Tool: Auto Repeat Curves

The Auto Repeat feature affects what happens when the end of a curve is reached. Auto Repeat is available only for Enhanced curves. See Standard vs. Enhanced Curves for details. With respect to Auto Repeat, Standard curves behave the same as an Enhanced curve with Auto Repeat disabled. The behaviors with and without Auto Repeat are described below:

- **Auto Repeat Disabled:**

When the last point in a curve is reached, the axis will stop and hold position at the final curve point. The In Position bit will turn on once the Actual Position comes within the In Position window from the Target Position. The curve will no longer be followed, even if the curve was geared to a master axis and the master axis goes backward.

If the curve is geared to a master axis and the master axis goes backward beyond the first point in the curve, the axis will hold position at the first point in the curve. If the master axis goes forward again, then the curve will begin to be followed from the start. The In Position bit will never come on in this situation because the curve is not completed.

- **Auto Repeat Enabled:**

When the axis reaches either end of the curve, it will automatically continue at the other end of the curve. There will be no position discontinuities, as described below.

Offset Positions on an Auto Repeat

When a curve wraps around from the start to end or vice versa, the positions of the first and last points in the curve are compared. If they are equal, then the curve continues without further event. If they are not equal, then the RMC offsets the position of the axis by the difference between the first and last points' positions. This offset is only allowed on axes with rotary feedback devices such as quadrature and rotary SSI. Attempting to auto repeat a curve with different start and end positions on an axis with a linear feedback device such as MDT, Analog, or linear SSI will fail with a parameter error.

To enable or disable Auto Repeat for a curve:

1. On the **Tools** menu, click **Options**.
2. Click the **General** tab.
3. Find the check box on the row of the axis you want to change under the column labeled **Auto Repeat**. To enable Auto Repeat, set this check box. To disable Auto Repeat, clear this check box.
4. Click **OK**.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.6.13 Curve Tool: Enforcing Limits

There are a number of limitations that each curve must satisfy. For example, each must not exceed position, velocity, and acceleration limits. For a detailed discussion on these limits, see Curve Limits.

The curve must satisfy all of these limits before it can be downloaded to the RMC. However, there are times when it is easier to allow the curve to violate one or more of these limits while it is being edited. Thus, the Curve Tool allows you to enable and disable limit enforcement.

The status bar will display one of the following two icons:

Limits are Enforced

Limits are Not Enforced

In addition to this icon, the status bar also has two other icons related to the curve limits. See Status Bar for a description of these icons.

To enable or disable limit enforcement using the status bar:

1. Double-click the **Limits are Enforced** or **Limits are Not Enforced** status bar icon to toggle enforcement on and off.

To enable or disable limit enforcement using the shortcut menu:

1. Right-click in the Graph or Spreadsheet view.
2. On the shortcut menu, click **Enforce Axis Limits**.

To enable or disable limit enforcement using the **Options** dialog box:

1. On the **Tools** menu, click **Options**.
2. Click the **General** tab.
3. Under **Editing Options**, set or clear the **Enforce Axis Limits** check box.
4. Click **OK**.

See Also: Curve Tool Topics

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.6.14 Curve Tool: Linking Curves

Some applications require synchronizing the curves from two or more axes. The Link Curves feature keeps track of updating all visible curves together. Turning on this feature causes the following changes to take place when editing a curve:

- When a point is added to a curve, the following steps will be taken:
  - If the point being added is within five pixels in the time (or master position) direction from a point on another visible curve, then the point to be added will have its time (or master position) adjusted to match the existing point on the linked curve. This makes it easier to add synchronized points using the mouse.
  - If the point being added is not within five pixels in the time (or master position) direction from the active curve, then the point is added to the active curve, and a point is also added to curves of other visible axes that do not already have a point at that time or master position. The position used for these new points will either match the existing curve, or if it is off the end of the existing curve, will match the position of the closest point on that curve.
- When a point is dragged on one curve, points on other visible curves that have the same time (or master position) value will also be adjusted time-wise. The positions will not be changed on the other curves.
- When a point is deleted on one curve, points on other visible curves that have the same time (or master position) value will also be deleted.

To turn the Link Curves feature on or off:

1. On the **Tools** menu, click **Options**.
2. Click the **General** tab.
3. Under **Editing Options**, set or clear the **Link Curves Together** check box.
4. Click **OK**.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.7 Using Points

3.9.7.1 Curve Tool: Selecting Points

Points need to be selected before doing several operations, such as expanding/contracting, deleting, and editing properties. Currently selected points are outlined with a blue square in the Graph view, and outlined in a black box or highlighted with a blue background in the Spreadsheet view. There are a number of ways to select points. Each is described below:

To select a single point with the mouse in the Graph view:

1. Click on the point in the Graph view.

To select a single point with the mouse in the Spreadsheet view:

1. Click on the header for the point in the Spreadsheet view. You can also select all the property cells for a point, see Selecting Cells for details.

To select multiple points using the mouse and selection box:

1. Click and drag to select a region of the Graph view.

All points that fall within this region will become selected. The selection box will be resized to fit just the selected points. The selection box can be used for moving all the points together or expanding and contracting the points.

To select a range of points using the mouse in the Graph view:

1. Click on the first point in the range.
2. Hold down the SHIFT key and click on the last point in the range.

These two points and all points with time (or master position) values that fall between the two will be selected.

To select a range of points using the mouse in the Spreadsheet view:

1. Click on the header for the first point.
2. Hold down the SHIFT key and click on the header for the last point.

These two points and all the point in between these two will be selected.

See Selecting Cells for more information.

To add or remove a point from the list of selected points in the Graph view:

1. Hold down the CTRL key, and click on a point.

If the point was previously selected, it will be de-selected. If it was not previously selected, then it will become selected.

To add or remove a point from the list of selected points in the Spreadsheet view:

1. Hold down the CTRL key and click on the header for that point.

See Selecting Cells for details.

To select all points in the active curve:

1. On the **Edit** menu, click **Select All**.

You can also use the CTRL+A shortcut key to issue this command.

To select points with the keyboard in the Graph view:

1. Press any of the following keys to change the selection to the point indicated. Holding down the SHIFT key with any of these keys will select a range of points instead of just the single point:

Press	To Select
HOME	First point in the curve.
END	Last point in the curve.
LEFT ARROW UP ARROW	Previous point.
RIGHT ARROW DOWN ARROW	Next point.

2.
3. See Also: Curve Tool Topics

4. Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan
5.

3.9.7.2 Curve Tool: Determining a Point's Exact Location

Displaying a curve graphically is convenient for viewing the entire curve. However, there are times when you may have to know or edit a point's exact location. This can be done by using the following:

- **Data Tips:**
  Hover the pointer over a point displayed in the Graph view until a data tip pops up. This data tip displays the point's time (or master position), position, velocity, and acceleration. The Curve Tool must be the active window for data tips to work. See Units of Measurement for a description of the units displayed.

- **Point Properties:**
  You can display a point's properties using a number of methods. See Point Properties for details. This method can also be used for editing properties.

- **Spreadsheet View:**
  You can display the active curve in the Spreadsheet view to see a point's time, position, velocity and interval type properties. This method also allows you to edit the point's properties. See Spreadsheet View for details on displaying the Spreadsheet view.

The following methods will not give you a point's exact location:

- **Status Bar:**
  The status bar has a pane that displays the current location of the pointer. However, this always shows the pointer location, which may or may not match up with a point's location even if it appears the pointer is over the point.

- **Detail Window:**
  The Detail window is useful for determining the exact position, velocity, and acceleration at a given time (or master position), but it is not always possible to position it to have the exact same time (or master position) as a point on the screen. See Detail Window for more information.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.7.3 Curve Tool: Adding Points

To add a point to your curve, use one of the following methods. The keyboard method has the advantage of allowing you to add points at a precise location. In any of these methods, it is possible that adding a point at the requested location will be invalid. If this is the case, you will be instructed why the request is invalid.

If you have the **Link Curves Together** option enabled, then adding a point on one axis will add a point to all other visible curve. See Linking Curves for details and exceptions.

To add a point with the mouse in the Graph view:

1. Select the axis you want to add a point to, as described in Selecting the Active Axis.
2. Double-click at the location you want to add the point. A point will be added to the active curve at this location.

To add a point with the shortcut menu in the Graph view:

1. Select the axis you want to add a point to, as described in Selecting the Active Axis.
2. Right-click at the location you want to add the point to display a shortcut menu.
3. On the shortcut menu, click **Insert Point**. A point will be added to the active curve at the location that was clicked to display the shortcut menu.

To add a point using the keyboard:

1. Press the INSERT key.

This will display an **Insert Point** dialog box.

2. Select the axis you want to add the point to.
3. Type in Time (or master position) and Position values.
4. Click **OK**.

You can also add points using the Spreadsheet view, see Spreadsheet View and The Insertion Point for details.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.7.4 Curve Tool: Deleting Points

One or more points can be deleted from a curve. It is possible that deleting the requested point would make the active curve invalid. If this is the case, you will be instructed why the request is invalid.

If you have the **Link Curves Together** option enabled, then deleting a point will also delete all other visible points on other axes with identical time (or master position) values. See Linking Curves for details.

To delete a point in the Graph view:

1. Select the point or points you wish to delete in the Graph view, as described in Selecting Points.
2. On the **Edit** menu, click **Delete**.
   You can also issue the **Delete** command by pressing the DELETE key or by right-clicking on the point or points you wish to delete, and clicking Delete on the shortcut menu.

To delete a point in the Spreadsheet view:

1. Click on the header for the point you wish to delete.
2. On the **Edit** menu, click **Delete**.
   You can also issue the **Delete** command by pressing the DELETE key or by clicking **Delete** on the shortcut menu.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.7.5 Curve Tool: Point Properties

Each point in a curve has the following properties:

Property	**Description**
Time	The time or master position for the point.
Position	The slave position for the point.
Velocity	Each point can have either a fixed or dynamic velocity. By default all points except end points have dynamic velocities, which means that RMCWin computes the best velocity for the point to yield the smoothest overall curve. End points default to have a fixed velocity of zero. However, if you need a point to have a specific velocity, you can change it to be a fixed velocity.
Interval Type	This property defines what shape the interval that follows this point will have. If you select **Cubic**, then it will be defined by a cubic function. If you select **Linear**, then it will be a straight line from this point to the next. By default all intervals are **Cubic**.

Note: Linear intervals and fixed velocities are not available in Standard curves. See Standard vs. Enhanced Curves for details.

To view or change point properties, use the **Point Properties** dialog box:

1. Select the point or points whose properties you wish to view or edit, as described in Selecting Points.
2. On the **View** menu, click **Point Properties**.
   You can also double-click on a point, right-click on a point and then click **Properties** on the shortcut menu, or press ALT+ENTER or ENTER to open the **Point Properties** dialog box.
3. Edit the properties if you wish.
   If you are editing a single point and want to adjust all following points by the same amount the selected point's time (or master position) and position change, then set the **Shift Following Points** check box.
4. Click **OK**.

You can also change a point's properties in the Spreadsheet view; see Spreadsheet View and Editing Cells for details.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.7.6 Curve Tool: Moving Points

There are a number of ways to move points in the Graph view. The only method that sets the position and time (or master position) exactly is the Point Properties dialog box, described below. All other methods involve using the mouse and are therefore limited by the resolution of the screen. Points can also be moved using the Spreadsheet view; see Spreadsheet View for details.

Each method below moves all selected points the same amount. If you want to instead expand or contract a set of points, see Expanding and Contracting Points.

To move a point using the **Point Properties** dialog box:

1. Select the point or points that you want to change, as described in Selecting Points.
2. On the **View** menu, click **Properties**.
   You can also double-click on a point, right-click on a point and then click **Properties** on the shortcut menu, or press ALT+ENTER or ENTER to open the **Point Properties** dialog box.
3. Type new values in the Time and Position text boxes.
   If multiple points are selected, then you will not be able to edit the Time value, since each point must have a unique time (or master position).
4. If you are editing a single point and want to adjust all following points by the same amount that the selected point's time (or master position) and position change, then set the **Shift Following Points** check box.
5. Click **OK**.

To move points by dragging the selection box:

1. Select a range of points by creating a selection box, as described in Selecting Points.
2. Drag the body of the selection box by placing the pointer over the center region of the selection box and dragging.
   To hold time (or master position) constant during the drag, hold down the CTRL key while dragging.
   To hold position constant during the drag, hold down the ALT key while dragging.

To move points by dragging a point or points:

1. Click and drag a point.
   If the point is already one of several points selected, then all points will be dragged together.
   To hold time (or master position) constant during the drag, hold down the CTRL key while dragging.
   To hold position constant during the drag, hold down the ALT key while dragging.

To move points using the keyboard:

1. Select the point or points you want to drag, as described in Selecting Points.
2. Press CTRL+LEFT ARROW, CTRL+RIGHT ARROW, CTRL+UP ARROW, or CTRL+DOWN ARROW to move the point in the corresponding direction on the screen. Hold down the keys to move the point faster.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.7.7 Curve Tool: Selecting Linear or Cubic Segments

A curve can consist of a mixture of cubic (curved) and linear intervals. By default each interval is cubic, but you can freely change an interval's type between Cubic and Linear. Each point controls the interval that follows it.

Note: Linear intervals are not available in Standard curves. See Standard vs. Enhanced Curves for details.

Note: It is not recommended to put two or more linear segments in a row because they will cause an acceleration spike (not displayed in the Curve Tool) at the transition and therefore a significant jerk in the motion.

To change an interval type:

1. Select the point or points that precede the intervals you want to change, as described in Selecting Points.
2. On the **View** menu, click **Properties**.
   You can also double-click on a point, right-click on a point and then click **Properties** on the shortcut menu, or press ALT+ENTER or ENTER to open the **Point Properties** dialog box.
3. Under **Advanced**, select either **Cubic** or **Linear** from the **Interval Type** list.

4. Click **OK**.

You can also change a point's interval type in the Spreadsheet view; see Spreadsheet View and Editing Cells for details.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.9.7.8 Curve Tool: Changing a Point's Velocity

Each point can have either a fixed velocity or dynamic velocity. By default all points have dynamic velocities, which means that RMCWin computes the best velocity for the point to yield the smoothest overall curve. However, if you need a point to have a specific velocity, you can change it to be a fixed velocity.

Note: Fixed velocities are not available in Standard curves. See Standard vs. Enhanced Curves for details.

To set a point's velocity:

1. Select the point or points that you want to change, as described in Selecting Points.
2. On the **View** menu, click **Properties**.
   You can also double-click on a point, right-click on a point and then click **Properties** on the shortcut menu, or press ALT+ENTER or ENTER to open the **Point Properties** dialog box
3. Under **Advanced**, set the **Set Velocity** check box.
4. Enter the desired velocity in the **Velocity** text box.
5. Click **OK**.

You can also set a point's velocity in the Spreadsheet view; see Spreadsheet View and Editing Cells for details.

To give a point a dynamic velocity:

1. Select the point or points that you want to change, as described in Selecting Points.
2. On the **View menu**, click **Properties**.
   You can also double-click on a point, right-click on a point and then click **Properties** on the shortcut menu, or press ALT+ENTER or ENTER to open the **Point Properties** dialog box.
3. Under **Advanced**, clear the **Set Velocity** check box.
4. Click **OK**.

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl"

endspan

3.9.7.9 Curve Tool: Expanding or Contracting Points

It is often useful to expand or contract a range of points with respect to either time (or master position) or position, such that the ratio of the times or positions between the points does not change. The Curve Tool offers this feature.

To expand or contract a range of points:

1. Select a range of points by creating a selection box, as described in Selecting Points.
2. Drag one of the eight resize handles on the selection box.

The opposite resize handle will be the anchor. For example, dragging the right-center resize handle will keep the left-most point in the selection constant, and dragging the top-left resize handle will keep the bottom-most point's position constant and the right-most point's time (or master position) constant (assuming time is shown horizontally).

See Also: Curve Tool Topics

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

3.10 Address Tool

3.10.1 Address Tool: Overview

The Address Tool calculates RMC register addresses for fields that are displayed in RMCWin. RMC register addresses are required for many communication types (Ethernet, Serial, Modbus Plus, and PROFIBUS-DP in Message Mode) and for some commands in the Event Step table (Add, Subtract, and MulDiv).

Fields in RMCWin that have corresponding RMC register addresses include all fields in the main window (status, commands, plot time, and parameters) and the Event Step, Motion Profile, Status Map, and Input-to-Event table editors.

Opening the Address Tool

To display the Address Tool, in the main window, click the **Window** menu, and then click **Address Tool**. In addition to being opened directly, the Address Tool is also opened automatically when adding a bookmarked address (see Bookmarking Addresses) or pasting an address into the Event Step table (see Using with the Event Step Editor).

Address Tool Screen Layout

The Address Tool has the following sections:

- **Address Format list.** Use this list to select the format of the addresses displayed in the Address Tool. For example, if you are programming an Allen-Bradley PLC-5 and you want to know what address to use to read Axis 0's Target Position, then you should select the Allen-Bradley address format. Notice that addresses pasted to the Event Step table will automatically be converted to the RMC Commands format.
- **Current Field.** The name and address of the currently selected field is automatically updated in this area of the Address Tool as you switch between different windows and fields in RMCWin. For example, to find the address for Axis 0's Target Position, you would open the Address Tool, and

then click on the value for Axis 0's Target Position in the main window. The Address Tool will be updated to display the name and address of this field.

- **Bookmark list.** Addresses can be bookmarked for later use. See Bookmarking Addresses for details.
- **Close button.** Click this button to close the Address Tool.
- **Bookmark and Delete buttons.** These buttons are used to maintain bookmarks of addresses for later use. See Bookmarking Addresses for details.
- **Paste button.** This button is used to paste bookmarked addresses into the Event Step Editor. Notice that this button is only available if exactly one address is selected in the bookmark list, and the current field is an Accel, Decel, Speed, or Command Value in the Event Step table editor. See Using with the Event Step Editor for details.

The Address Tool can be resized and minimized as desired. It can also be set up to stay on top of all other RMCWin windows, as described in Keeping the Address Tool in the Foreground.

3.10.2 Address Tool: Bookmarking Addresses

The name and address of the current field is continuously updated in the Address Tool. That is, as you select different fields with corresponding addresses, the current field's name and address will change in the Address Tool.

Whether you are using the Address Tool to give you addresses to use in your PLC program or in the Event Step table, you will often want to remember addresses that are used commonly. The Address Tool provides bookmarking to keep track of as many addresses as you want. In addition, bookmarked addresses can be easily entered in the Event Step table, as described in Using with the Event Step Editor.

To add a bookmark for a field's address:

1. Select the cell of the field whose address you want to bookmark.
2. In the Address Tool, click **Bookmark**.

Notice that clicking the Bookmark button brings the Address Tool to the foreground, which is often undesirable. Therefore, we recommend using one of the following shortcuts when bookmarking addresses:

- If the field that you want to bookmark is in the main RMCWin window, then on the **Window** menu, click **Add Bookmarked Address**.
- From any window whose fields are reflected in the Address Tool, click CTRL+SHIFT+B to add a bookmark for the current field. If the Address Tool is not already open, then it will be opened and the current field will be added to the Bookmark list.

Notice that if the address is already in the bookmark list, then the existing entry will be moved to the top of the list instead of adding a duplicate entry.

To delete one or more bookmarks from the address list:

1. Open the Address Tool.

2. In the **Bookmarked Fields** list, select the fields that you want to remove from the list.
3. Click **Delete**.

To paste a bookmarked address to the Event Step table, see Using with the Event Step Editor.

Note: If you find yourself bookmarking many addresses, you may find it useful to increase the size of the Address Tool so that you do not have to scroll up and down in the bookmark list to find the address you need. Simply drag the border of the Address Tool to change its size.

3.10.3 Address Tool: Using with the Event Step Editor

The Address Tool can be used normally (see Address Tool Overview) to obtain addresses of any Event Step table field. However, it can also be used to simplify entering addresses into the Event Step table for commands that use RMC register addresses. Currently, the Add, Subtract, and MulDiv commands each require source and destination RMC register addresses. Manually entering these addresses is time consuming and error prone.

Therefore, the Address Tool's bookmark feature has been augmented to allow pasting bookmarked addresses into an Event Step table field. Here is how this feature is intended to be used:

1. Bookmark the field or fields you will be pasting into the Event Step table as described in Bookmarking Addresses. It is usually quickest to bookmark several fields ahead of time so that this step does not need to be repeated for each address.
2. In the Event Step table editor, select the field that will hold the bookmarked address. For example, to enter a bookmarked address into the source address field of an Add command, you would select the Deceleration field for that step. Notice that if you have already entered the command (+) in the command field, then the address fields will be displayed with parentheses to indicate that they are addresses.
3. Press CTRL+B to start pasting an address into the field. This will bring the Address Tool to the foreground with a bookmarked address selected. This is equivalent to clicking **Paste Bookmarked Address** from the **Edit** menu in the Event Step table editor.

Notice that if you are in an Event Step table editor field that is recognized as an address (indicated by parentheses) because the command has already been entered, then you can press ENTER or double-click the field to start pasting an address into the field.

4. Use the UP and DOWN ARROW KEYS to select the address you want to paste from the bookmark list. This is equivalent to selecting an address by clicking it with the mouse.
5. Press ENTER to paste the value to the Event Step table editor and bring it to the foreground. This is equivalent to clicking **Paste**. Notice that the address pasted will automatically be converted to the RMC Commands format if you have another format selected in the Address Tool.

This sequence has been carefully designed to avoid requiring manually switching between foreground windows. That is, pasting an address is as simple as pressing CTRL+B, UP or DOWN ARROW KEYS, and then ENTER, without using the mouse at all. Of course, the mouse still can be used in all of the above steps.

3.10.4 Address Tool: Keeping the Address Tool in the Foreground

Because the Address Tool updates as you select fields in other windows, you will often want to ensure that the Address Tool remains visible, even when you are working in other windows. One way to do this is to move and/or resize all windows so that the Address Tool will never be behind any windows. However, an easier solution in many cases is to use the Always on Top feature of the Address Tool. When this feature is turned on, the Address Tool will never go behind other windows, even when it is not active.

To turn on or off the Always on Top feature:

1. Right-click the Address Tool title bar to display its shortcut menu.
2. On the shortcut menu, click **Always on Top**.

Note: Most RMCWin windows have the Always on Top feature, which is enabled in the same way for each window. Notice that no guarantee can be made as to which window will be on top out of multiple windows designated as Always on Top.

3.11 Advanced Topics

3.11.1 Downloading New RMC100 Firmware

If new features have been added or problems fixed in the RMC firmware, then it is necessary to update the firmware to take advantage of these improvements. You should only use the firmware download feature at the instruction of technical support to address a specific need.

Note: You cannot update the RMC100 firmware when connected to the RMC via the **TCP/IP Direct to RMC-ENET** communication driver. You will need to use either the **Serial** or **TCP/IP-to-RS232 Bridge** driver.

To download new firmware:

1. Obtain the new firmware from technical support.
2. Run RMCWin.
3. Turn off the system being controlled by the RMC. This is necessary because the RMC must be reset for the new firmware to take effect. During this time, the RMC cannot control motion.
4. On the **Tools** menu, click **Module Configuration**.
5. If you want to be able to restore the current firmware, you may want to make a backup. Click **Backup FW** and follow the instructions that follow to do this.
6. Click **Update FW**.
7. After reading the warning, click **Yes** if you feel it is a safe time to have the RMC reset. Otherwise, click **No**.
8. You will be asked to find the file you will use as the new firmware. Enter the name of the firmware file provided by technical support, and click **Open**. This can be on a floppy disk or on the hard

disk.

9. If the firmware you requested to download is not supported by the current Boot/Loader firmware, a notification will appear. Click OK if you have the Boot/Loader firmware and give the Boot/Loader firmware filename in the **Open** dialog box.

10. The version number of the firmware to be downloaded will be displayed. Verify that the firmware date(s) is newer than the current firmware you are using. If so, click **Yes**, otherwise click **No** to select another firmware file.

11. The firmware will then be downloaded.

3.11.2 Downloading New Serial/Ethernet Firmware

If new features have been added or problems fixed in the Serial or Ethernet communication modules' firmware, then it is necessary to update the firmware to take advantage of these improvements. You should only use the firmware download feature at the instruction of technical support to address a specific need.

Note: You cannot update the RMC ENET firmware when connected to the RMC via the TCP/IP Direct to RMC-ENET communication driver. You will need to use either the **Serial** or **TCP/IP-to-RS232 Bridge** driver.

To download new firmware:

1. Obtain the new firmware from technical support.

2. Start RMCWin.

3. Turn off the system being controlled by the RMC. This is necessary because the RMC must be reset for the new firmware to take effect. During this time, the RMC will lose communication through the communications module.

4. On the **Tools** menu, click **Module Configuration**.

5. In the **Slots** list, click the **Ethernet** or **Serial** module.

6. Click **Slot Options**.

7. Click the **Firmware** tab.

8. Click **New Firmware**.

9. After reading the warning, click **OK** if you feel it is a safe time to have the communications be interrupted. Otherwise, click **Cancel**.

10. You will be asked to find the file you will use as the new firmware file. In the **File name** box, enter the name of the firmware file provided by technical support, and click **Open**. This file can be on a floppy or on the hard drive.

11. The version numbers of the firmware to be downloaded will be displayed. Verify that the firmware date is the version you expected. If so, click **Yes**, otherwise, click **No** to select another firmware file.

12. The firmware will then be downloaded.

13. When the download is complete, click **OK** from the progress dialog box. Click **Cancel** from the **Options** dialog box unless you want to make changes to other configuration items.

    Ethernet Firmware Files

    The RMC Ethernet protocols are divided into two different firmware files. When updating the RMC100 Ethernet firmware, you must choose the file that contains the protocol you need. The files are called Protocol Group E and Protocol Group F:

- **Protocol Group E**

    a. EtherNet/IP
    b. Allen-Bradley CSP (a.k.a. DF1 over Ethernet)
    c. Modbus/TCP
    d. Omron FINS
    e. AutomationDirect.com HEI

- **Protocol Group F**
    a. ISO-over-TCP for Siemens S7 controllers
    b. CAMP for Simatic 505 with CTI Ethernet
    c.

New RMC100-ENET modules will be shipped with Protocol Group E. However, both groups are available for download from the Delta website, and both are supported.

3.11.3 Options: Preferences

The Preferences tab of the Options dialog box contains miscellaneous user preferences for RMCWin.

To view or change these preferences, do the following:

1. On the main window's **Tools** menu, click **Options**.
2. Click the **Preferences** tab.
3. Select the preferences you want to use.
4. Click **OK**.

The Preferences tab includes the following options:

- **Do not show confirmation warnings check box**

Select this check box to suppress confirmation-warning messages. This includes all warnings when closing a window to save the contents to either the RMC or the disk. Select this check box only if you know the program very well and understand when your work is saved.

- **Look-only check box**

In Look-only mode, the user cannot make changes to the module. This prevents the user from changing any parameters or issuing any commands. It is still possible to view all data. Notice that this setting is saved so it will be remembered the next time RMCWin is started.

- **Limit Instances to One per Board File check box**

  Clear this option if you want to open multiple RMCWin copies at once but do not care which board file is used. For example, if you wish to share between two event step tables, you can clear this option, start a second RMCWin, open an Event Step table editor under each, and copy and paste between them.

  Set this option in all other situations. The primary reason for this is that it prevents accidentally starting multiple RMCWin copies, for example, by clicking the RMCWin icon too many times.

  A more important reason to set this option is that several PC-based human-machine-interface (HMI) software packages allow starting RMCWin from a button, but if the focus is switched back to the full-screen HMI there is no way to retrieve the copy of RMCWin hidden behind the HMI. Clicking the button with this option cleared starts a new copy of RMCWin, but if the serial port is being used, then the new copy would be useless since the first, hidden copy would retain control of the serial port. With this option set, clicking on the button on the HMI will trigger the original RMCWin instance to be brought to the foreground instead of starting a second instance.

  Some systems require access to two or more RMCs and provide a separate button for each on the HMI. Each button starts RMCWin with a different board file and a different communication path. Therefore, with this option set, it is possible to have each of these various board files and communication paths open under different copies of RMCWin, but you are still protected from starting multiple copies with the same board file and communication path.

- **Ignore Configuration Conflicts check box**

  When this box is checked, RMCWin’s configuration-conflict detection feature is disabled. For details on this feature, see Configuration Conflict Detection.

3.11.4 Forcing Initialization

It is possible to have RMCwin automatically send the parameters it has stored in a file to the RMC. In most cases this should not be necessary because there are two other methods of ensuring a module is initialized. Both are more desirable:

1. The parameters can be stored in the Flash. A module uses the parameters stored in the Flash when it starts up.
2. The parameters can be stored in the Programmable Controller and downloaded when the RMC is started up. This is usually desirable when a Programmable Controller is available to ensure that the RMC is always configured correctly even if a module is swapped out.

To enable Forced Initialization, you must start RMCWin using the -F command-line parameter. This can be added as a parameter by editing the shortcut which starts RMCWin or by typing the parameter after the program name when starting it from a command prompt. For more details, see Command-line Options.

3.11.5 Using Look-only Mode

In Look-only mode, the user cannot make changes to the module. This prevents the user from changing any parameters or issuing any commands. It is still possible to view all data.

To toggle look-only mode:

1. On the **Tools** menu, click **Options**, and then click the **Preferences** tab.
2. Select or clear the **Look-only** check box to enable or disable look-only mode.
3. Click **OK**.

Notice that this setting is saved so that it will be remembered the next time that RMCWin is started.

3.11.6 Using PC Mode

If RMCWin is going to be used for demonstration purposes without actually being attached to an RMC, you can use PC mode. The difference between PC mode and regular operation is that in PC mode, RMCWin starts with the communication path closed. This means that you cannot connect to an RMC.

If RMCWin is not in PC mode, it can still run without being connected to an RMC. However, it will still constantly scan the communication path, thus reserving resources (such as the serial port) that could be freed for other applications.

To run RMCWin in PC mode, you must start RMCWin using the -P command-line parameter. This can be added as a parameter by editing the shortcut which starts RMCWin or by typing the parameter after the program name when started from the command-line.

3.11.7 Command-Line Options

RMCWin can be started with various command-line options. These options can be used to ensure that the program starts with the correct settings. Notice that most users will have no need for these options because they are saved from one execution of RMCWin to another. Options given on the command-line override the previously saved settings.

The following options are available:

-1, -2, -3, -4	Select COM1, COM2, COM3 or COM4 as the serial port to use. This will override any communication path setting for the current board file.
-Dn	Generate a debug log file named Debug.txt in the current directory. The logging level is controlled by n, which can be from 0 (no logging) to 5 (most logging). There is no logging enabled by default.
-F	Force initialization onto module. See Forcing Initialization for details.
-L	Start in Look-only mode. See Using Look-only Mode for details.
-M	Allow multiple instances of RMCWin to run at once with the same board file open. The copy of RMCWin that is started with this command and all subsequent RMCWin instances started can run at the same time. This will continue until the Limit Instances to One per Board File check box on the

	Preferences tab of the Options dialog box is set.
-P	Start RMCWin with the communication path closed. This can be used for demonstrations or offline editing.
-R	Start in Read-back mode. See Read-back versus Write Mode for details.
-W	Start in Write mode. See Read-back versus Write Mode for details.
Board file	The parameter after the filename that does not start with a ’r;-’ or ’r;/’ gives the board filename that is used on startup (e.g. rmcwin carriage.bd1). See Using Multiple Motion Modules for details on board files.

4 Controller Features

4.1 Event Control Overview

The Event Control feature allows you to execute a sequence of commands without intervention from the Programmable Controller (P/C). This lets the module respond to events within the control-loop time (1ms or 2ms) rather than the scan rate of the P/C. It also reduces the controller programming required.

Event Control consists of a series of Steps that are linked together in sequences. The Steps consist of a command area containing the instruction to be executed and a link area that specifies the next Step number and its trigger. There are a total of 256 Steps that can be shared by all axes.

Steps

Each Step contains a command with its associated parameters, plus the information necessary to link to the next Step in a sequence. The Step format is as follows:

Command Info:

Mode	MODE word
Accel	Acceleration
Decel	Deceleration
Speed	Requested Speed
Command Value	Requested position or command value
Command	Any valid ASCII command
Commanded Axes	Axis (or axes) to send command to

Link Info:

Link Type	Condition that triggers execution of next step
Link Value	Parameter associated with Link Type
Link Next	Number of next Step in sequence

Storing Steps

Steps are stored in the motion controller's memory. This step table can be saved in the motion controller's Flash using the Update Flash command. However, in some applications this information should be stored in the Programmable Controller so that it can be downloaded to the motion controller on initialization. Refer to Event Step Transfer for details on storing the steps.

Changing Event Steps

Refer to Editing the Event Step Table for details on using the Event Step editor.

Basic Event Step Flow

After a valid event step sequence is loaded into the motion controller, event control can be used. This section describes the basic flow:

To start a sequence of events, use one of these methods:

- Issue a Start Event command to trigger an event from RMCWin or the Programmable Controller.
- Use the Input to Event Table to trigger an event sequence from a digital input.

A sequence of events will stop when one of the following occurs:

- A Quit Events command has been issued from RMCWin or the Programmable Controller.
- A step is executed which has a Link Type of 0.
- A Halt or Disable Drive Output command is carried out.
- Another Start Event command is carried out. This stops the current sequence and starts the new one.

The following additional details should also be kept in mind:

- The Mode, Accel, Decel, Speed and/or Command Value fields need not be filled out if the command does not use them.
- The command field can be left blank in a step if only the link type is being used. This is useful for chaining steps together to make a sequence wait for multiple conditions to all be true.
- The command in an event will be executed as soon as a step is reached in the event sequence.
- The next Step in the sequence will be executed as soon as the conditions from the Link Type and Link Value are met.
- Multiple axes can be made to follow identical patterns simultaneously by using the same steps on each.
- Steps may be conditionally branched using the Poll (?) command.
- Pop-up editors may be used for simplified editing of the Mode, Commanded Axes, Link Type and Link Next fields. For information on using these editors, see Using Popup Editors.

Example

In the following example three steps are executed. They cause the axis to make a move, wait and then make another move:

	Step 15	**Step 16**	**Step 17**
Mode	0x0001	0x0001	0x0001
Accel	100	100	100
Decel	100	100	100
Speed	10000	10000	10000
Command Value	15500	10000	3000

Command	G		G
Commanded Axes	Default	Default	Default
Link Type	BitsON	DelayMS	DelayMS
Link Value	00001	500	0
Link Next	16	17	0

Step 15 issues a Go (G) command to 15.5 inches (15500). The BitsON (B) link type with a link value of 0x0001 causes the motion controller to look for the least significant bit in the STATUS word (the In Position bit). When the In Position bit turns on, indicating the move is complete, Step 16 (Link Next – 16) is executed.

Step 16 has no command so no command is issued. The DelayMS (D) link type with a link value of 500 waits 500 milliseconds before jumping to Step 17.

Step 17 issues a Go (G) command to 3 inches. The DelayMS link type with a value of 0 (zero) causes the axis to go immediately to Step 0, which stops the sequence.

4.2 Flash Memory

The RMC has two blocks of Flash memory that can be independently updated. The first section contains all of the following information:

- All axes' parameters
- The profile table
- The entire step table
- The input to event table
- Communication configuration data
- LCD screens

The second section contains the following:

- Spline tables

Determining if the Flash Data was Used

Prior to issuing the first Set Parameters (P) command, the Initialized bit in the STATUS word will be cleared. While this bit is cleared, the Parameter error bit in the STATUS word of the first axis will indicate whether the first Flash section was valid. If the Parameter error bit is set, then the checksum was invalid and all data stored in the Flash will use default values. If the Parameter error bit is not set, then the checksum was correct and the data stored in the Flash will be used. Similarly, the Parameter error bit in the Status word of the second axis will indicate whether the second Flash section (the splines) was valid.

Storing Data in the Flash

All data in each section is stored simultaneously. Therefore, it is not possible to store just the step table in the Flash without storing all the other data in that section listed above. This will not be a problem as long as you ensure that all data you want to be stored is set correctly before saving the data in the Flash.

To update the first Flash section, issue the Update Flash (U) command to any axis, or use the **Save to Flash** toolbar button in RMCWin's main or LCD Screen Editor windows.

To update the second (spline) Flash section, use the **Save Splines to Flash** toolbar button in RMCWin's Curve Tool window. Notice that this section cannot be updated without using RMCWin.

While either Flash section is being updated, the green CPU LED will flash. During this time motion control will continue, but removing the power from the module will result in all parameters being stored in the Flash to be lost.

The disadvantage of storing configuration data in the motion controller is that, when one module is replaced with another, the parameters and profiles must be loaded into the new module. Because of this, all parameters, profiles and event step table data must be stored either in the control program or in RMCWin files so they can be later transferred to the motion controller when needed.

4.3 Gearing Axes

Gearing is used when one axis (the geared axis) must move incrementally and proportionately to another axis (the gear master). This topic describes a method of gearing one axis to another by using a gear ratio. There are three other types of gearing available, described in the following topics:

- Sine Move Command
- Follow Spline Segment Command
- Follow Axis with Offset - see the SoftPID Position Move Command

The gear master axis can either be controlled by the RMC (internal) or just a position input from an encoder or transducer that is not under the control of the RMC (external, also called an half axis). A numerator and denominator specified in the geared axis's command area determines the ratio of the movement between the gear master and the geared axis.

The only limitation for gearing axes together is that axes must not be geared in a loop (that is, if axis 0 is geared to axis 1, axis 1 cannot also be geared to axis 0).

The Gear Master Axis:
An axis does not need to do anything to become the master of a gearing relationship. That is, the axis may be controlled in open or closed loop, or it may even be only an input (in the case of a velocity or position reference such as a joystick). The geared axis will select its gear master axis.

Setting up a Geared (Slave) Axis:
The slave axis starts gearing when issued a Go command with the command parameters set up as follows. It continues gearing until another motion command is issued to the same axis.

1. **Specify the Gear Master and Type.**

In the Mode command parameter, do the following:

- **Set the Gear Bit** (bit 12, 0x2000 hex). If this bit is not set, then the Go command will not initiate a geared move. It will instead do a point-to-point or speed control move.
- **Use the Gear Master Select bits** (bits 4-6) to select the desired master axis. Multiple slaves can be geared to the same master.
- **Set the Gear Type Bit** (bit 14, 0x4000 hex) to be geared to the master's Actual Position. Clear this bit to gear to the master's Target Position.

2. Specify the Gear Ratio.

The slave-to-master gear ratio is specified by the Command Value command parameter divided by the Speed command parameter, as shown in the below three equivalent equations:

$$\text{Ratio} = \frac{\text{Command Value}}{\text{Speed}} = \frac{\text{\# master gear teeth}}{\text{\# slave gear teeth}}$$

$$\text{Ratio} = \frac{\text{Command Value}}{\text{Speed}} = \frac{\text{Slave Target Speed}}{\text{Master Speed}}$$

$$\text{Ratio} = \frac{\text{Command Value}}{\text{Speed}} = \frac{\text{Change in Slave Target Position}}{\text{Change in Master Position}}$$

* The master speed and position used may be based on either the Actual or Target Position, as selected by the Gear Type bit in the Mode word.

Note: When the gear ratio is clutched between two ratios--such as from non-geared to 2:5 gearing--the gear ratio will be ramped at a user-specified rate (see below). This clutching will take place by ramping the gear ratio numerator from its original value to the final ratio numerator. However, the interim numerator values must always be integers. Therefore, it is recommended that relatively large values be used for the gear ratio components. For example, instead of specifying values of 2 and 5 for a 2:5 ratio, you might specify 2000 and 5000. This allows clutching through up to 2000 gear ratios instead of only two, for a much smoother clutch.

Example:

To obtain a gear ratio of 1:2, such that the slave axis moves at half the speed of the master axis, you will enter a Command Value that is one half the value of the Speed. A Command Value of 1000, and a Speed of 2000 would work. As described in the above note, values of 1 and 2 would also work, but there would be no clutching: the axis would go directly from a 0:2 to a 1:2 gear ratio. By using the equivalent 1000:2000 gear ratio, clutching from 0:2000 to 1000:2000 can be made in up to 1000 small gear steps.

3. Specify the Gear Ratio Clutching.

Whenever a Gear command is given to an axis that is already geared, the gear ratio is ramped from the current ratio to the requested ratio. Unless the gear ratio is ramping to or from zero, only the numerator (given in the Command Value command parameter) can change; the denominator (given in the Speed command parameter) must remain constant.

When a Gear command is given to an axis that is not currently geared, then its initial gear ratio is computed such that the computed geared speed of the axis will match its current speed.

Note: RMC CPU firmware prior to 20020222 always starts a previously non-geared axis with a zero gear ratio.

The ramping up or down of the gear ratio may be specified in several ways. The Acceleration/Deceleration Mode Select bits in the Mode word determine the method:

Mode 0: Reserved; do not use.

Mode 1 (Rate): Acceleration indicates the number of counts that the numerator is increased or decreased each millisecond until it reaches the requested ratio. The Deceleration field is not used in this mode.

Mode 2 (Distance): This mode uses the following two parameters to specify the range of master positions over which the gear ratio will be ramped:

Acceleration: Indicates the total distance spanned by the master positions. The span must be less than 32767 position units for proper operation.

Deceleration: Indicates the starting master position for the slave's ramp.

Speed: The sign of this field, which is the denominator of the new gear ratio, indicates the direction that the range of master positions span from the starting master position.

Note: A negative number x may need to be entered as 65536 - x. For example, -1000 is entered as 64536.

Once the final gear ratio is reached, the slave will be locked into that gear ratio. Therefore, if the master backs up after the final gear ratio is reached, the slave will move at that ratio.

If the master backs up *before* the final gear ratio is reached, the slave will follow the clutching ramp profile backwards. If the master moves before the clutching start point, and then back into the clutching area, the slave will resume the clutching at the correct position as specified by the command.

Example 1 of Mode 2:

Acceleration: 600
Deceleration: 1000
Speed: Positive

The slave will change the gear ratio beginning when the master is at 1000 position units, and reach the requested gear ratio when the master is at 1600 position units.

Note that if the axis is given this gearing command when the master axis is at a position greater than 1600, the gearing will start instantaneously.

Example 2 of Mode 2:

Acceleration: -200
Deceleration: 500
Speed: Negative

The slave will change the gear ratio beginning when the master is at 500 position units, and reach the requested gear ratio when the master is at 300 position units.

Note that if the axis is given this gearing command when the master axis is at a position less than 300, the gearing will start instantaneously.

Mode 3 (Time): The Acceleration field indicates the time in milliseconds that the ramp will take. The Deceleration field is not used in this mode.

4. **Issue the Command.**

After the command parameters (Mode, Acceleration, Deceleration, Speed, and Command Value) have been set up as described in the preceding steps, the Go (G) command can be issued.

Gear Ratio Status

State Bits A and B in the Status word are used to indicate the state of the gear ratio on the slave axis in the following manner:

Gear Ratio State	Bit 5 (State Bit B)	Bit 4 (State Bit A)
At zero	0	0
Increasing	0	1
At Requested Value	1	0
Decreasing	1	1

Limitations

Because both the Command Value and Speed fields, which make up the gear ratio, are limited to –32,768 and +32,768, the number of teeth on the master and slave gear are likewise limited. However the Speed field (denominator) of the slave axis cannot be zero. Also, be careful when setting the number of master teeth larger than the number of slave teeth. It is possible to cause the slave speeds to exceed 65,535 position units per second, which would cause the speed to wrap and incorrect feed forward values to be calculated.

Example 1:

	Axis 0 (Master)	Axis 1 (Slave)
Mode	0x0281	0x2001
Acceleration	100	50
Deceleration	100	50
Speed	10000	10000
Command Value	0	5000
Command	G	G

In the gear master axis's Mode word, the Rotational and S-Curve bits are set to indicate that the axis will control speed using S-Curved ramps. See Speed Control for details on this feature. Also in its Mode word, the Acceleration and Deceleration Mode is selected as type 1. As a result the axis will move continuously in the positive direction at 10,000 position units per second.

In the Mode word of the geared axis, Gearing mode is selected with axis 0 as the master and will use Acceleration/Deceleration Mode 1. The Gear Type bit is cleared indicating that the axis will gear to the master's Target Position. It will move at half the speed of the master because the gear ratio is 5000 divided by 10,000. Assuming that the gear ratio is 0 when the slave is issued this command, it will take 100 milliseconds to ramp the gear ratio from 0/10,000 to 5000/10,000 at 50/10,000 per millisecond.

Example 2:

	Axis 0 (Master)	**Axis 1 (Slave)**
Mode	0x0281	0x2002
Acceleration	100	1000
Deceleration	100	1000
Speed	10000	10000
Command Value	0	10000
Command	G	G

In this example, the master is given the exact command as the previous example.

The slave axis's Mode word is the same, except that Acceleration/Deceleration Mode 2 is used, which ramps the axis based on the master's distance. The final ratio is 1:1 (10,000/10,000). The ramp up to this ratio will occur while the master moves from 1000 to 2000 position units (as given by the starting position in the Deceleration field, the master move length given in the Acceleration field, and the master move direction given by the sign on the Speed field).

As the master moves the 1000 position units, the slave will move 500 position units since its average speed is one half of the master speed while ramping. This means that if the master is at a position of 1000 and moving toward 2000 and the slave is at a position of 1500 when the slave receives the command, both axes will reach 2000 at the same time. This is useful for flying cutoff saw applications.

4.4 LED Indicators

Understanding the CPU LED Indicators

The CPU module, labeled RMC100, has five LEDs. Each is labeled as follows:

CPU Status This LED can be in the following states:

Red	On power up, this LED will be red until the firmware is started.
Solid Green	This indicates that the module is running correctly.
Flashing Green	This indicates that the Flash is being written to or erased. Do not power down the RMC while the LED is

	in this state, or the Flash write will fail.
In 0	When this LED is RED, the CPU digital input 0 is a logical 1.
In 1	When this LED is RED, the CPU digital input 1 is a logical 1.
Out 0	When this LED is RED, the CPU digital output 0 is conducting.
Out 1	When this LED is RED, the CPU digital output 1 is conducting.

For further details on the CPU digital inputs and outputs, see Using the CPU Digital I/O.

Other Module LED Indicators

Many of the additional RMC modules have LED indicators. Look in the table of contents or index for the appropriate module's LED indicator topic.

Analog Modules LED Indicators

Ethernet LED Indicators

MDT LED Indicators

SSI LED Indicators

Quadrature LED Indicators

Stepper LED Indicators

Motion Profiles

What is a Motion Profile?

A profile is a speed, acceleration, deceleration and mode used during a move. The profile specifies how the axis will move to the requested position. Commands that will change the profile are:

Set Mode - Changes the MODE bits.

Change Acceleration - Changes the acceleration rate.

Change Deceleration - Changes the deceleration rate

Set Speed (Unsigned) and Set Speed (Signed) - Changes the speed.

Note: The profile does not include the requested or starting positions.

What is the Motion Profile Table?

The Motion Profile Table stores sixteen motion profiles (Mode, Acceleration, Deceleration, and Speed) in its Flash memory. It is used only for the following communication types:

- PROFIBUS-DP in Compact Mode
- Communication DI/O in Command Mode
- Communication DI/O in Parallel Position Mode

Why use the Motion Profile Table?

In the communication types that use the Motion Profile Table, the PLC can issue only a Command and Command Value in a single command cycle. Therefore, the Go command can only give the Requested Position as the command value, and to set the entire motion profile and give the Go command would require five commands. While this is flexible, it is inefficient.

By using the Motion Profile Table, a complete move including all four motion profile fields can be set with a single Go Using Profile command, speeding up both execution and development time.

How is the Motion Profile Table Used?

When using PROFIBUS-DP or the Communication DI/O in Command Mode, the Go/Set Pressure Using Profile and Open Loop Using Profile commands select one of the 16 motion profiles in the table and use that profile to issue the command.

When using the Communication DI/O in Parallel Position Mode, every command given uses one of the motion profiles. For details, see Using Parallel Position Mode.

How is the Motion Profile Table Changed?

There are two ways to the change the motion profiles table:

- Use the Motion Profile Table Editor. On the **Tools** menu in RMCWin, click **Profile Editor** to start this editor; see Editing the Profile Table for details.
- When using PROFIBUS-DP or the Communication DI/O in Command Mode, use the Set Profile commands to change the table from the PLC.

How do I Save the Motion Profile Table?

There are three places where the Motion Profile Table can be saved:

- Flash memory. Use the Update Flash command to store the current profile table in the Flash memory.
- Disk file. The table can be saved to disk from within the Motion Profile Table Editor.
- PLC memory. When using PROFIBUS-DP or the Communication DI/O in Command Mode, use the Set Profile and Get Profile commands to send the table to and from the PLC.

What are the Default Motion Profile Values?

When the motion controller starts, it first does a checksum on the profile table in the Flash memory. If the Flash-stored profile table checksum is correct, it is used. If the checksum is incorrect, then the following default profiles are used:

Profile	**0**	**1**	**2**	**3**
MODE	0	0	0	0
ACCEL	1000	2000	5000	8000
DECEL	1000	2000	5000	8000
SPEED	1000	2000	5000	8000

Profile	**4**	**5**	**6**	**7**
MODE	0	0	0	0
ACCEL	10000	12000	15000	18000
DECEL	10000	12000	15000	18000
SPEED	10000	12000	15000	18000

Profile	**8**	**9**	**10 (0A)**	**11 (0B)**

MODE	1	1	1	1
ACCEL	100	100	100	200
DECEL	100	100	100	200
SPEED	20000	25000	30000	35000

Profile	**12 (0C)**	**13 (0D)**	**14 (0E)**	**15 (0F)**
MODE	1	1	1	1
ACCEL	200	200	200	100
DECEL	200	200	200	100
SPEED	40000	45000	50000	10000

4.5 Reference Axis Filtering

When positions and velocities are used as references for moves made on other axes, whether for electronic gearing, geared splines, or geared sine moves, it is often desirable to filter the reference input to reduce the noise induced mechanically, electrically, or through quantization error.

The RMC100 provides position filtering on axes with position or velocity inputs through a special reference state. The reference state is defined as follows:

- The Actual Position status field reflects the actual reading from the transducer, and the Target Position reflects the filtered position, after applying the Filter Time Constant, Reference Deadband, Velocity Limit, and Acceleration Limit parameters described below. Axes gearing to this reference axis should select to gear to the reference axis's Target Position to use the filtered position.
- The drive output is in open loop. It can be changed normally using Open Loop (O) commands. In cases where the RMC100 is controlling an axes used as a reference by another axis, the effect of Actual Position noise on the geared axis can be eliminated by gearing to the reference axis's Target Position instead of the Actual Position.
- While in the reference state, all motion commands for this axis are disabled. For example, Go (G) and Relative Move (J) commands will generate an Invalid Command parameter error but otherwise be ignored.

An axis enters and exits the reference state through one of two means:

- Analog inputs configured as Position Reference and Velocity Reference axes are always in the reference state and have the filter parameters applied. The filter parameters are included in the sixteen axis parameters, and can be edited like any other axis parameter. The Reference (W) command can also be used to change these parameters.
- All axes with position or velocity inputs can use the Reference (W) command to place the axis in the reference state. The Reference (W) command parameters define the filter parameters. The axis remains in the reference state until a Set Parameters (P) command is issued, which will place the axis back into closed loop, disable the position filter, and re-enable motion commands.

The filter is controlled by the following four parameters. The effects of each are shown in examples at the end of this topic:

- **Filter Time Constant:** This parameter controls the time constant itself for the filter. It is entered in milliseconds. The cut-off frequency for the filter in Hertz is found by 1/2pt, where t is the time constant is seconds. See Filter Time Constant for details on the parameter itself.

Note: The Velocity Limit and Acceleration Limit are used only in conjunction with the Filter Time Constant. If the Filter Time constant is zero, they will have no effect.

- **Velocity Limit:** This parameter limits the rate that the Target Position can change. See Velocity Limit for details on the parameter itself.

- **Acceleration Limit:** This parameter limits the rate that the Target Velocity can change. It is used to avoid a velocity discontinuity after an Actual Position step jump. See Acceleration Limit for details on the parameter itself.

- **Reference Deadband:** This parameter is used to eliminate hunting (rocking back and forth between values) of the Target Position when the input is at rest. This parameter specifies the number of position units that the Actual Position must move before it affects the Target Position. See Reference Deadband for details on the parameter itself.

Note: The Reference Deadband should not be used unless the Filter Time Constant, Velocity Limit, and Acceleration Limit are used.

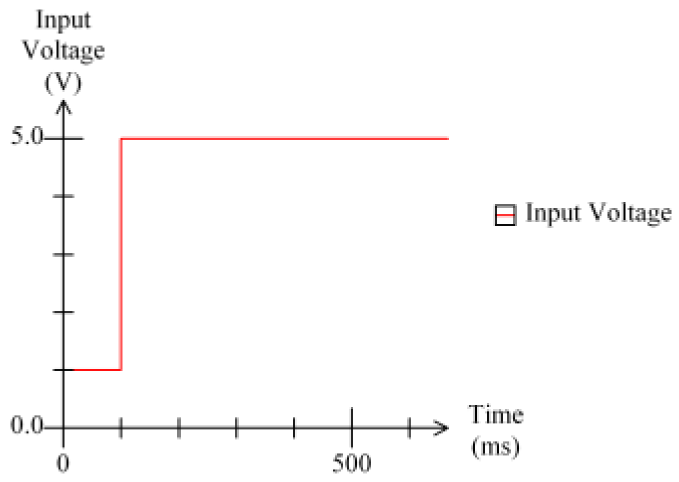
The position parameters on analog reference axes are saved with the other axis parameters to the Flash memory when the Update Flash (U) command is issued. However, the position filter settings on all other axis types cannot be saved to Flash memory. Therefore, non-analog reference axes can never start in the reference state on power-up.

The following examples demonstrate how these parameters affect the Target Position on a reference axis.

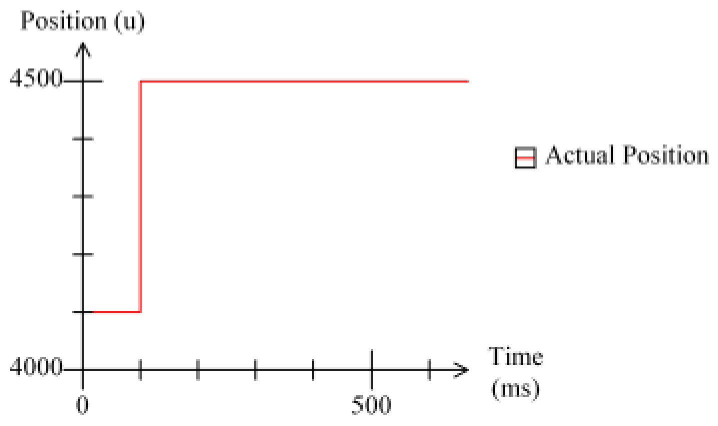
Example 1: Analog Reference with a Step Jump

In this application, the user wants to specify the command position through an analog signal. The user wants the RMC to take care of both ramping the target position smoothly from one command position to the next and controlling to the target position.

Suppose the input voltage is as follows:

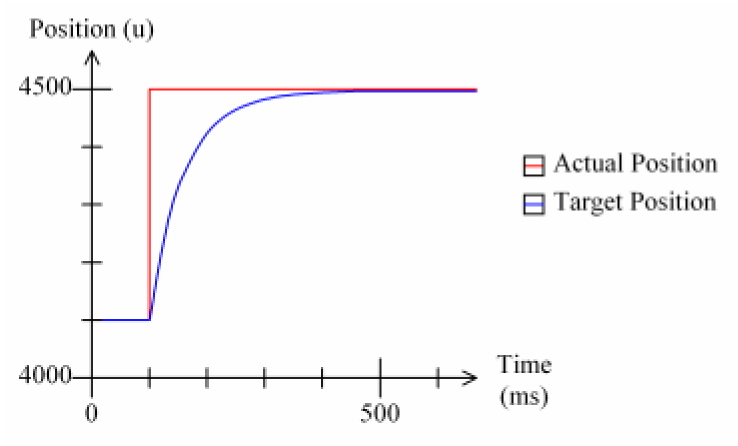


This would result in an Actual Position with the same shape, although scaled to the user's units. Without a reference filter enabled, the Target Position will match this Actual Position curve:



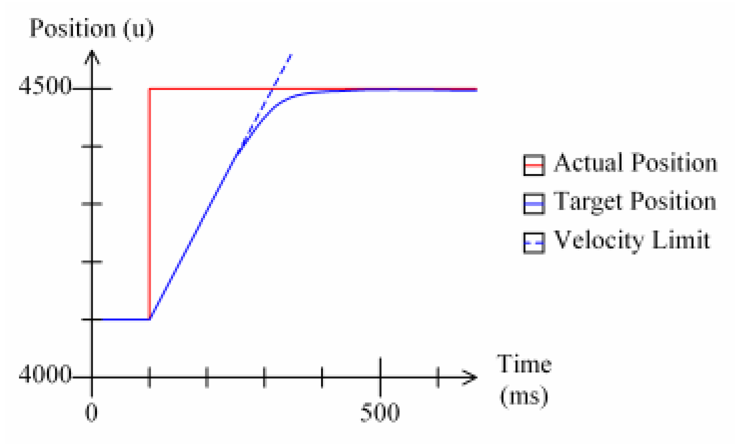
So, without the reference filter, the RMC would not ramp the reference input at all, and jumps in the input current or voltage would be translated directly to jumps in the reference position and jumps in any axes geared to this reference.

To enable reference filtering, the Filter Time Constant parameter must be set to a non-zero value. The Target Position will now appear as follows, as compared to the Actual Position:

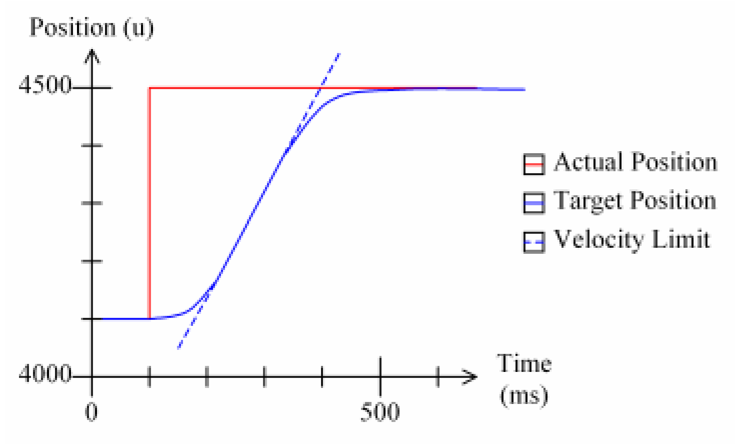


This is a great improvement in that the Target Position no longer has a large step jump. However, two problems remain. First, the Target Velocity at the time of the step jump depends on the amount the Actual Position changed at the step jump. Second, there is a sharp change in the Target Velocity at the time the step jump occurs.

The first problem can be addressed by introducing the Velocity Limit filter parameter. This parameter will limit the Target Velocity in our example as shown:



This limits the velocity to a user-specified maximum, but it still leaves us with a sharp change in the Target Velocity at the time of the step jump. The Acceleration Limit can be used to address this issue:



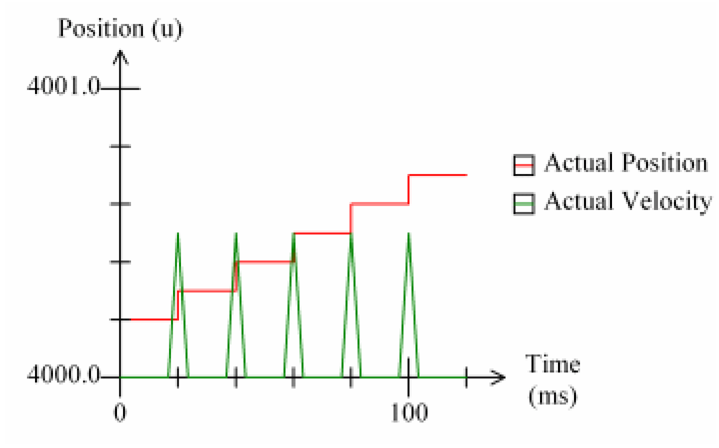
In the above graph, the Target Position is smoothed out at the time of the step jump. This curve should be usable as a reference for RMC100 geared moves. In addition, the Reference Deadband parameter can be used to remove slight ripple in the Target Position while the input voltage is at rest.

Example 2: Quadrature Reference with Quantization Error

Quadrature references will not see large step jumps in the input as is possible for analog inputs. However, the quantization error in applications with low rates of quadrature counts per second can lead to spikes in the reference speed.

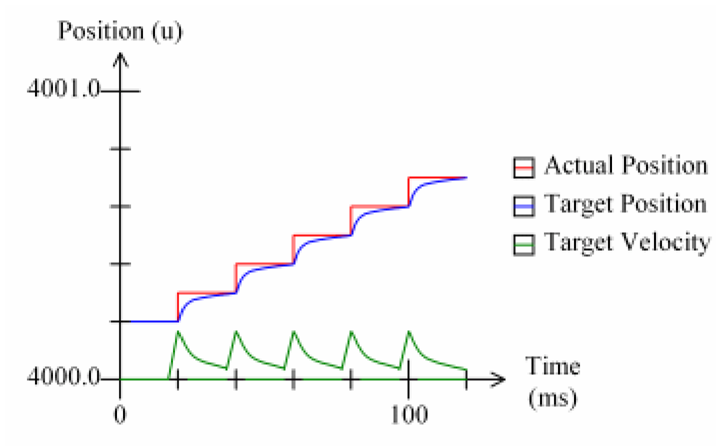
Note: This example demonstrates how the reference filter can be used to reduce the effect of quantization error. However, increasing the resolution of the quadrature encoder (for example, from a 1000-line to a 4000-line encoder) can dramatically reduce quantization error as well.

The Actual Position curve shown below demonstrates the effect of quantization error when moving at slow speeds:



While the effect of quantization on the positions themselves does not appear very great--after all, the position is accurate to one-half a quadrature count--its effect on velocities is much more dramatic.

By applying a Filter Time Constant, the Target Position and Velocity are improved as shown:



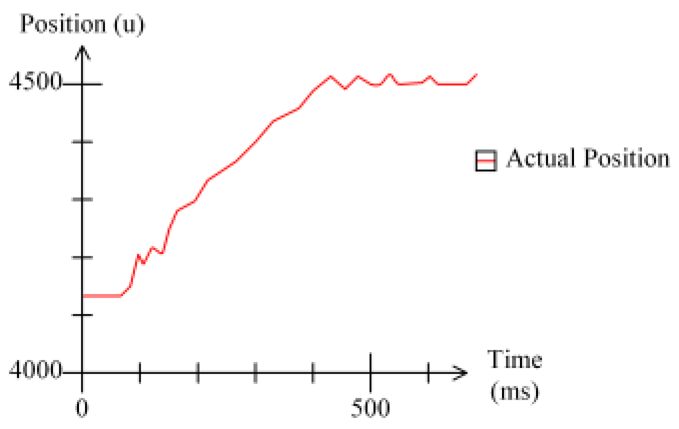
The Velocity and Acceleration Limits are necessary only if the quadrature reference axis moves too quickly at times.

The Reference Deadband can be used to hold the Target Position steady when the reference axis is at rest.

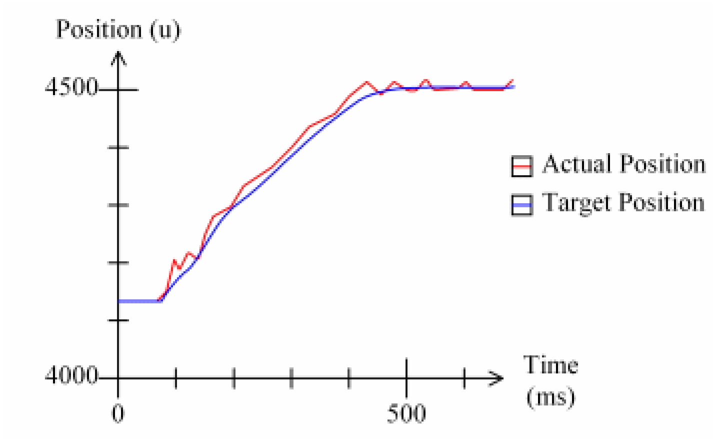
Example 3: Analog Joystick Reference

In this final example, the user wants to use a 0 to 10V input voltage as a position reference for other axes. Unlike the first analog input example, the input voltage in this application will not typically have a step jump.

A typical Actual Position based on the input voltage might look like this:



Gearing to a manually-guided voltage such as a joystick often results in rough motion due to mechanical jitter and friction in the potentiometer itself. Therefore, the jaggedness of the previous graph can be reduced by applying the Filter Time Constant.



This may be all that is necessary in this application. However, if it is possible for the user to move the joystick too quickly, then the Velocity and Acceleration Limits can be used as in the first example to prevent over-speed conditions. The Reference Deadband can be used to hold the Target Position steady when the reference input is at rest.

4.6 Speed Control

The RMC100 provides two modes of closed loop speed control:

- **Speed Control with Position Loop**
  This mode performs closed loop control on the axis position (not speed). To achieve speed control, the RMC100 generates a target that moves at the commanded speed. This means that if the Actual Position of the axis falls behind the Target Position, the RMC100 will attempt to catch up to the moving Target Position, requiring the axis to move at a speed much higher than the commanded speed. This may be undesirable in certain applications.

- **Speed Control with Velocity Loop**
  This mode performs closed loop control on the axis speed. If the Actual Speed of the axis falls behind the Target Speed, the RMC100 will attempt to catch up to the Target Speed.

Note: Speed Control with Velocity Loop is supported in RMC100 CPU firmware dated 20030515 or later.

Using Speed Control with Position Loop

Speed control with Position Loop is used in the same way as position control with the following exceptions:

- The Rotational bit must be set in the Mode word.
- The Command Value field for the Go command indicates the direction of the move, rather than the Requested Position.

Using Speed Control with Velocity Loop

Speed control with Velocity Loop is used in the same way as Speed Control with Position Loop with the following addition:

- The Velocity Loop bit must be set in the Mode word.

Speed Control Commands

There are three main commands to use in speed control:

1. **Go Command**

When a Go command is issued with the Rotational bit in the Mode word set, a Speed Control with Position Loop move is initiated. The Command Value holds the requested direction. For further details, see Go Command.

When a Go command is issued with the Rotational bit and the Velocity loop bits in the Mode word set, a Speed Control with Position Loop move is initiated. The Command Value holds the requested direction. For further details, see Go Command.

Advantages:

This command can start a speed control move. All command fields can be set with a single command.

Disadvantages:

When used with a communication type that allows only a Command and Command Value each command cycle (PROFIBUS-DP in Compact Mode and Communication DI/O using Command Mode or Parallel Position Mode), the speeds and accelerations are limited to those in the Motion Profile Table.

2. **Set Speed (Signed) Command**

This command takes a number between –32,768 and 32,767 as the Command Value and sets both the requested direction (positive indicates an extend or clockwise move, and negative indicates a retract or counter-clockwise move) and the requested speed. The Rotational bit in the Mode word must be set prior to issuing this command.

Advantages:

Both the speed and direction can be changed with a single command.

Disadvantages:

Only speeds between 0 and 32,767 can be set with this command. Also, this command cannot set the Rotational bit in the Mode word, so another command must do this.

3. Set Speed (Unsigned) Command

This command works the same as the Set Speed (Signed) command except that only the requested speed is changed; the requested direction cannot be changed with this command. Therefore, the Command Value is an unsigned number between 0 and 65,535 and holds the requested speed. The Rotational bit in the Mode word must be set prior to issuing this command.

Advantages:

Allows the speed to be set to any valid value (0 to 65,535 position units/second).

Disadvantages:

The direction cannot be changed with this command. Also, this command cannot set the Rotational bit in the Mode word, so another command must do this.

Speed Control with Cyclic Operations

In speed control the positions will increment or decrement until the position wraps. However, position can be reset on the fly. This is desired if there is a machine cycle that is less than 65536 position units long. If a machine cycle is 30000 units long it would be nice to reset the position to 0 when the axis gets above 29999. This way the same positions will occur at the same place in the cycle. This can be accomplished using either a Home Input to reset the positions or an Offset Positions command. The Home Input must be used if the absolute positions must always be the same or if the cycle distance is not an integer number of position units. For other cases, the Offset Positions command provides a cheaper and easier to implement way of maintaining the relative distance between events. In either case the Event Step Table must be used to either re-arm the Home Input or to reset the positions.

4.7 Rotational Mode

Rotational mode is specifically designed for cyclic axes. When an axis is in this mode, the positions wrap around from the Retract Limit to the Extend Limit. One of the most common uses for Rotational mode is speed control.

To enter or exit rotational mode, use the Set Mode (M) command or issue any of the following motion commands with the Rotational Bit (bit 9) set or cleared:

- Go (G)
- Follow Spline Segment (f)
- Open Loop (O)
- Sine Move (~)

Note: Only the following commands can exit rotational mode: Go (G) command for Quick, Synchronized, and Point-to-Point Moves, or a Sine Move (~).

Attempts to change the Rotational mode bit through the Set Mode (M) command will be ignored if the axis is doing a Point-to-Point, Quick, Synchronized, or Speed Control move.

Axes remain in Rotational mode through all Auto Stops, Halt (H) commands, Disable Drive (K) commands, and Set Parameters (P) commands.

Example:

To reset an axis position every 3600 position units, set the Retract Limit to 0 and the Extend Limit to 3599. When the axis moves beyond zero in the negative direction, it will wrap around to 3599. Notice that setting the limits to 0 and 3600 would result in 3601 position units per turn.

4.8 Spline Overview

Why Use Cubic Splines?

Some applications require that the motion controller move between a number of positions that vary for each sequence of moves. An example of this is curve sawing. In this application a cant (a section of a log) is scanned, and an optimizing computer determines the best path for the saws through the cant. The motion controller then must follow this smooth path through the cant.

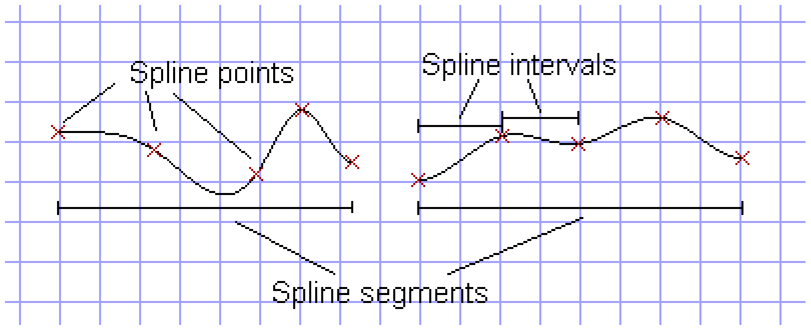
The most basic way to follow this curve would be to select various positions along the cant and calculate where the saws need to be at each length on the cant. Then the programmable controller could give commands to go to the various positions. The problem with using this method is that for each Go (G) command, the motion controller will ramp up to a constant velocity, and then ramp down to a stop for each point. This would not give a smooth curve through the entire cant.

Another method would be to carefully control the speed of a move to the extend or retract limit so that the saws follow the curve. The problem with this method is that a large number of calculations must be done in the programmable controller.

To greatly simplify the work required by the programmable controller and also provide smooth moves between points along a curve, the RMC has been equipped with cubic splines. When using cubic splines, the programmable controller needs only to send the positions where the axis should be located at various times along the curve. The RMC takes care of creating a curve that is smooth for position and velocity and continuous for acceleration.

Basic Vocabulary

Refer to these example splines for the following definitions.



Spline Point:

The user defines several spline points for a single curve. In the example shown above, the X's mark the points set by the user. The horizontal direction in this graph is time and the vertical direction is the axis position.

Spline Interval:

The user controls the number of master units between the spline points. Master units can be any of three quantities:

- **Master Axis Position Units.** If the curve is followed with a master axis selected, then each master unit will correspond to one position unit on the master axis. See the Follow Spline (f) and Follow Spline Relative commands for details on following a spline with a master axis. This is the most common way of following a spline.
- **Time.** If the curve is followed without a master axis selected, and no Digital I/O module counter is enabled (see Using Counters), then each master unit will correspond to 1/1024th of a second.
- **Counter Ticks.** If the curve is followed without a master axis selected, and a Digital I/O module counter is enabled (see Using Counters), then each master unit correlates to one counter tick. This method is largely historical. It is usually better to use the Master Axis Position Units as master units, but this method was introduced before the RMC's Quadrature module was available.

The interval between spline points cannot be greater than 65,535 master units or less than 10 units.

Spline Segment:

The RMC can hold several spline curves in its memory at a time; each is called a segment. There is no exact limit for how many points can be in a segment, but there is a limit for the total number of points among all segments in the motion controller. This limit is 1023 points per axis on a 2-axis RMC, 511 points per axis on a 3- to 4-axis RMC, and 255 points per axis on RMCs with five or more axes.

Standard vs. Enhanced Splines

The RMC supports two types of curves: Standard and Enhanced. Each is described below.

Standard Curves

Standard curves were introduced with RMC CPU firmware dated 19971204. This gave the RMC the capability of following curves defined by cubic splines. The user defines the position and time of each point in the curve. The velocities of the endpoints are fixed at zero (0). The velocities of all other points and accelerations of all points are computed by the RMC to find the smoothest curve through all points.

There are several deficiencies with Standard curves:

- Because the accelerations are computed at the endpoints, it is likely that they will be non-zero and therefore result in an acceleration discontinuity and relatively high jerk at each endpoint.
- It is difficult to accurately approximate a linear segment.
- It is difficult to set and maintain a given velocity at a given point.
- The velocities at the endpoints could not be changed from zero (0).

These issues prompted the introduction of Enhanced curves.

Enhanced Curves

Enhanced curves were introduced with RMC CPU firmware dated 20010208. They greatly enhance the capabilities of the RMC in following an arbitrary curve, and further enhance the smoothness of the curve.

The following items are introduced with enhanced curves:

- The accelerations at the endpoints of a curve are always zero (0). Therefore, there is no acceleration discontinuity and therefore much lower jerk at the end points. The acceleration is continuous over the entire curve, even with the additional features listed below.

- Segments can be easily designated as linear segments, which makes them exactly linear. This greatly simplifies camming and flying cut-off applications.

- Any point, including the endpoints, can have its velocity set to a specific value. This, too, simplifies camming and flying cut-off applications.

- Curves can be set up to auto-repeat, meaning that the curve restarts automatically when it reaches its end. See Auto Repeat Curves for details.

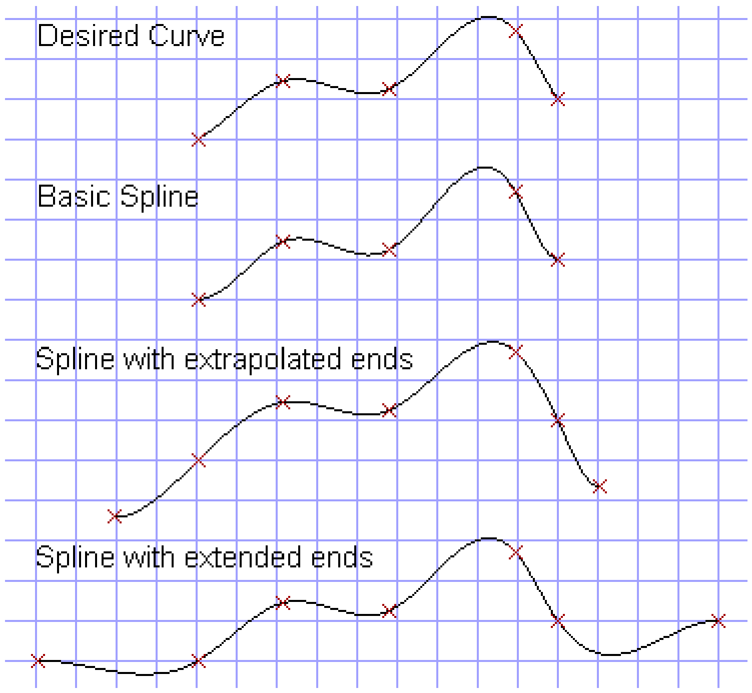
The drawback of Enhanced curves is that they can currently only be downloaded to the RMC through RMCWin's Curve Tool, and not from a Programmable Controller through communication modules such as Ethernet, PROFIBUS, and Modbus Plus. See Curve Tool Overview for details on the Curve Tool.

You should always use Enhanced curves unless you are running firmware prior to 20010208 and do not wish to upgrade, or you need to download curves through the communication module.

What If I Don't Want Zero Velocity at My Endpoints?

Each spline segment in a Standard curve will begin and end at zero velocity. This is desirable so that when a motion controller's axis is given the command to follow a spline curve, the axis does not instantly jerk to the new velocity. However, there are applications where the curve does not begin with zero velocity.

If the curve is downloaded from RMCWin, then you can switch to use an Enhanced curve which allows the velocity of each point to be set. However, if the curve is being downloaded over Ethernet, Modbus Plus, PROFIBUS or some other communication module, it is recommended that the user add one or more points to the beginning and end of the curve. The following curves demonstrate how this can be done:



The first curve is the actual curve that we want to match. The second curve is the one that would be generated by the RMC if no leading or trailing points were used. The velocities do not match well at the ends in this curve. In the third curve, one point is added to the front and one point is added to the end. In this curve, the endpoints were placed the same distance and time from the original endpoints as the second points from the ends. In the fourth curve, the new endpoints are added at the same position as the original endpoints, but are a fixed time from the original endpoints. This method works best when the time between the original and new endpoints are at least twice the distance between the original first and second points.

The best method to use depends on the application, and is not limited to the above methods.

Using Cubic Splines

There are three ways to get the points in a cubic spline into the RMC:

- Download curves from the Curve Tool. See Curve Tool Overview for details.
- Issue the commands described below over the communication module.
- Use the Spline Download Area. For a discussion of the tradeoffs between these three methods and further details on using the Spline Download Area, see Downloading Splines to the RMC.

Only the first method can generate Enhanced curves, the other two will generate Standard curves.

Using the Cubic Spline Commands

There are five commands that apply to cubic splines. Click on any of the command names below for further details on each:

Add Spline Point This is used to add a point to the end of a new segment.

Set Spline Interval This is used to control the time/counter ticks between points.

End Spline Segment This is used to indicate the end of a segment and perform final calculations on the curve.

Clear Spline Segments This is used to clear one or more segments from the motion controller's memory.

Follow Spline Segment This causes the axis to follow a single spline segment. The motion controller updates the spline every millisecond. The axis must already be at the position of the first point for this command to succeed.

Follow Spline Relative This causes the axis to follow a single spline segment relative to the current axis position. This command is similar to the Follow Spline (f) command but does not require the axis to be at the position of the first spline point.

In addition to these five commands, there are two other ways to download splines. They are described under Downloading Splines to the RMC. When using other methods of download splines—using the Spline Download Area and using RMCWin—the first three commands listed above are not used, but the final two still apply.

A Cubic Spline Example

Suppose that a cant (a section of a log) is scanned for a curve sawing application. For simplicity, we assume the cant was scanned at only seven points. We also assume that we're using a Sensor Digital I/O to keep track of the quadrature counter on the belt moving the cant. Therefore, we'll also assume that the cant was scanned at a uniform distance between readings of 200 quadrature counts.

Therefore, suppose the optimizer gave the following readings:

Counts Since Start of Cant	**Position of Saws (in thousandths of inches)**
0	60000
200	61000
400	61500
600	61250
800	61000
1000	60750
1200	61000

In this application it is not safe to assume that the saws should be at zero velocity at the ends of the cant. Therefore, we'll use one of the methods described above for getting non-zero velocities at the ends: we will add a point to the beginning and end of these sets of points. So the new data will look like this:

Counts Since Start of Cant	**Position of Saws (in thousandths of inches)**

-400	60000
0	60000
200	61000
400	61500
600	61250
800	61000
1000	60750
1200	61000
1600	61000

The optimizer must then send this spline segment to the motion controller. Assuming that the optimizer cannot use RMCWin to download the spline, there are two methods of doing this. The first is to use the Add Spline Point, Set Spline Interval, and End Spline Segment commands. The second is to use the Spline Download Area. An example of the steps required to do each is shown below:

Downloading with the Spline Download Area:

You should look over Downloading Splines to the RMC for details used in this example. To use the Spline Download Area for this example, you need only write two blocks of data to the RMC's Spline Download Area. First, you would write the following data to give the count and time intervals. Addresses are given in Modbus (RTU, TCP, and Plus) format. See Downloading Splines to the RMC for other communication module addressing formats.

Address	Value	Description
12289	1	Format of interval data (one value per interval)
12290	400	Interval between points 0 and 1
12291	200	Interval between points 1 and 2
12292	200	Interval between points 2 and 3
12293	200	Interval between points 3 and 4
12294	200	Interval between points 4 and 5
12295	200	Interval between points 5 and 6
12296	200	Interval between points 6 and 7
12297	400	Interval between points 7 and 8

Now that the RMC knows the number of points we will download and the intervals between these points, we can download the point positions:

Address	Value	Description
14337	9	Number of points in the spline segment.
14338	60000	Point 0 position
14339	60000	Point 1 position
14330	61000	Point 2 position
14331	61500	Point 3 position
14332	61250	Point 4 position
14333	61000	Point 5 position
14334	60750	Point 6 position
14335	61000	Point 7 position
14336	61000	Point 8 position

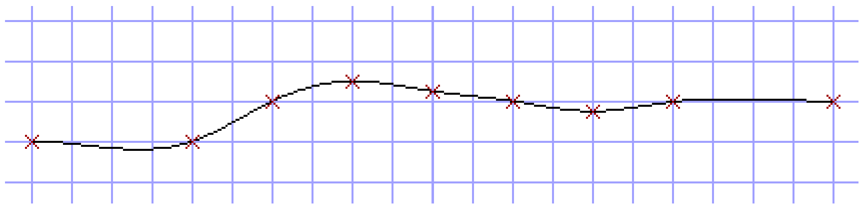
Downloading with Add Spline Point, Set Spline Interval, and End Spline Segment Commands: The following commands would be issued in order:

Command	Command Value	
C	0	Clear any pre-existing splines (only needed on initialization).
T	400	Set the interval for the points that follow to 400 master units.
X	60000	Send leading point.
T	200	Set the interval for the points that follow to 200 master units.
X	60000	Send 1st scanned value.
x	61000	Send 2nd scanned value.
X	61500	Send 3rd scanned value.
x	61250	Send 4th scanned value.
X	61000	Send 5th scanned value.
x	60750	Send 6th scanned value.
T	400	Set the interval for the points that follow to 400 master units.

X	61000	Send 7th scanned value.
x	61000	Send trailing point.
T	0	Signal end of the segment.

In this example, notice that the case of the Add Spline Point (X) command is toggled each time it is used. In order for the motion controller to process a command, it must detect that the command or command value has changed. Therefore, by toggling the case of the 'X' command, each point will be processed even if two points have the same value (as is the case with the last two points).

The following graph demonstrates what the loaded spline will look like, with the vertical gridlines being 100 counts apart and the horizontal gridlines being 1000 position units (1 inch) apart.



Next, suppose an electric eye in front of the saws detects when a cant is about to be cut. A cant will trigger this electric eye a fixed number of counts before it reaches the blades. In our example, this eye is 1000 counts in front of the saws. Because of the leading point added to the spline, our spline begins 400 counts before the cant itself, so we must delay 600 counts from the time the electric eye detects the beginning of the cant to the time that we begin following the spline.

The programmable controller can take care of this time delay, however it is also possible to process this input using the Input to Event and Event Step tables in the RMC. To do this, the output from the electric eye can be run into the motion controller's CPU input 0. CPU output 0 will then be run back to the programmable controller to indicate when the cut is completed. The entry in the Input to Event table for this input will hold the Event Step 1. The Event Step table would look something like this.

	Step 1	**Step 2**	**Step 3**	**Step 4**
Command Value	0x0001	0x0001	0	0x0001
Command	]	0	f	[
Link Type	InputRise	DelayTicks	BitsON	DelayMS
Link Value	0	600	00001	20
Link Next	2	3	4	1

In the first step, we reset the output to the programmable controller and wait for the input from the electric eye to trigger. After this occurs, we move to step 2 and delay for 600 counts. After the delay, event step 3 will be executed, which issues an 'f' command that makes the motion controller begin following the spline. After the In Position bit of the STATUS word is set, then step 4 is executed. At this point the CPU output 0 is set high and held high for 20 milliseconds to let the programmable controller know that the cant has been cut. The sequence is then repeated.

4.9 Synchronizing Axes

Axis synchronization is achieved by setting either the Sync A or Sync B bit in the MODE word on the axes to be synchronized and then issuing a Go or Relative Move command to the last axis in the sync group. Up to two groups of axes can be synchronized together. If both groups are used, then one group must use the Sync A bit, and the other uses Sync B. Each group can contain as many axes as desired.

The following requirements exist when synchronizing a group of axes:

- All synchronized axes must be initialized, as indicated by the Parameter Initialized bit in the STATUS word. Do this by issuing the Set Parameters command.
- The low eight MODE bits of each synchronized axis must match exactly.
- Synchronized commands (Go and Relative Move) must be issued to the last synchronized axis. The last synchronized axis is the highest numbered axis in the group and is displayed in the right-most column in RMCWin.
- Synchronized commands issued to any synchronized axis other than the last axis are ignored.
- The MODE, ACCELERATION, DECELERATION, SPEED, and COMMAND VALUE fields must be set in all axes to be synchronized before any synchronized commands are issued.

Internal to the RMC, the synchronization is accomplished as follows:

- For each group of axes the axis with the longest travel distance is designated the master. If two or more synchronized axes tie for having the longest travel distance, then the first of those axes is designated the master. The other synchronized axes are designated as slaves.
- The maximum speed of the master axis is limited so none of the synchronized axes will go above their requested SPEED.
- The master then moves toward its destination using its own ACCELERATION, DECELERATION, and the speed calculated in the above step.
- At the same time, the TARGET POSITIONS of the slave axes are ratioed to the target position of the master based on the distance each is traveling.
- If any synchronized axis is halted during the move, all the other axes will halt as well.
- If a new Go or Relative Move command is given to a slave axis which changes or removes the Sync bit used in the MODE word, then that slave will perform the new move, and all of the axes it was previously synchronized with will continue their synchronized move.
- If a new Go or Relative Move command is given to the master axis which changes or removes the Sync bit used in the MODE word, then the master will perform the new move, but all of its previous slaves will stop.
- If a new Go or Relative Move command is given to the last synchronized axis which retains the Sync bit in the MODE word, then all synchronized axes will start a new move from their current positions.

Note: Because a new Go or Relative Move command will cause the travel distance ratios to be recalculated, any change in the speeds or travel distances of any of the axes may result in a speed discontinuity in the new slave axes, although the position will always be continuous.

It is normally not possible to use RMCWin to issue commands to several axes simultaneously. However, when a command is issued to an axis that has a sync bit set, the software will automatically issue commands to all other axes that have the same sync bit set. This can only be done while in Write mode. Similarly, if a Stored Command is a synchronized command and is executed, all like-synchronized Stored Commands for the other axes will be executed simultaneously.

4.10 Teach Mode Overview

Many applications require that the operator be able to teach the system the desired positions in a sequence of moves. The RMC uses the Teach Step command to support these applications. This feature requires the use of Event Control; see Event Control Overview if you are not familiar with this feature.

The Teach Step Command

When the Teach Step command is issued to an axis, the current Target Position for the axis is copied into the Command Value field of the event step given by the command. Therefore, the move being taught must be located in the Event Step table.

Using Teach Mode

The following general steps are used in setting up an application utilizing teach mode:

1. Set up the Event Step table to hold the desired sequence of moves.

   At this point, the moves that are to be taught do not need to have correct requested positions, but for safety, approximate values should be used.

2. Download the Event Step table to the RMC.
3. Jog the axis to the first desired position.
4. Issue a Teach Step command to fill the requested position of the first move to be taught.
5. Repeat steps 3 and 4 for as many taught moves as desired.
6. Execute the newly taught Event Step sequence.

Teach mode can be, and has been, implemented using even the simplest communication mode: the Communication Digital I/O using Input to Event Mode. In this case it is necessary to place the Teach Step commands in the Event Step table as well, and trigger them using discrete inputs.

4.11 VC2100 and VC2124 Voltage-to-Current Converters

The VC2100 and VC2124 two-axis voltage-to-current converters transform ±10V signals into current signals capable of driving hydraulic servo valves or similar loads. They also provide a convenient way to set the full scale current to match valve requirements, limit maximum current and set optimum working ranges.

The VC2100 and VC2124 provide full scale range (FSR) output current for each channel is switch-selectable from ±10mA to ±100mA in 10mA steps.

The VC2100 requires a ± 15VDC power supply. The VC2124 requires a 24VDC power supply.

See Delta's website at www.deltamotion.com for more details on these voltage-to-current converters.

VC2100 Output Characteristics

This table shows the minimum output drive voltage and maximum load resistance for various output currents and power supply voltages.

Output	±15V ±5% Supplies			±12V ±5% Supplies		
Current	Vout		Maximum	Vout		Maximum
mA	Typ	Min	Load W	Typ	Min	Load W
10	13.7	12.7	1265	10.7	9.8	980
20	13.2	12.2	608	10.2	9.3	465
30	12.7	11.7	388	9.7	8.8	293
40	12.2	11.2	279	9.2	8.3	208
50	11.7	10.7	213	8.7	7.8	156
60	11.2	10.2	169	8.2	7.3	122
70	10.7	9.7	138	7.7	6.8	97
80	10.2	9.2	114	7.2	6.3	79
90	9.7	8.7	96	6.7	5.8	65
100	9.2	8.2	82	6.2	5.3	53

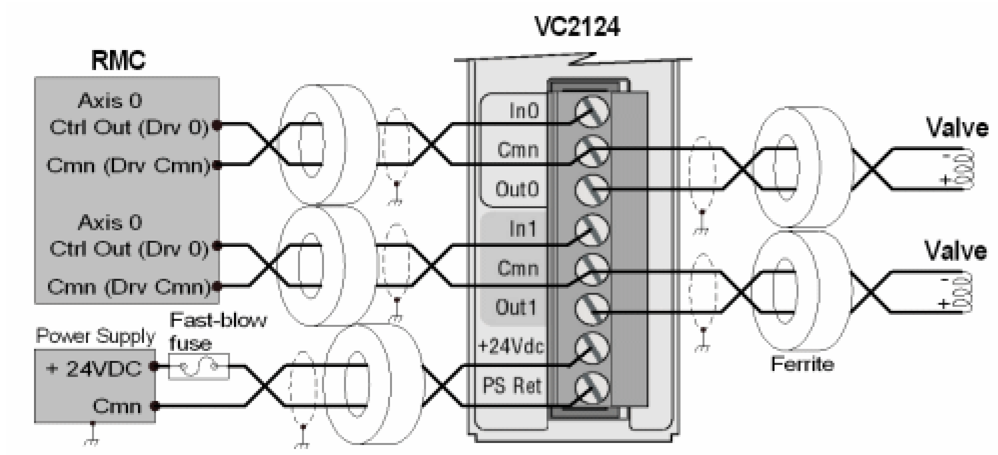
The VC2100 can drive a short circuit to common—an internal 50W current-limiting resister limits the output current. The output amplifier will shut down under severe overload (such as driving a short to a power supply).

Wiring

VC2124

Fuse 24Vdc input with 5A maximum, UL-listed, fast-blow fuse. One fuse suffices for up to 10 VC2124s. For maximum protection, use one 500mA fuse per VC2124.

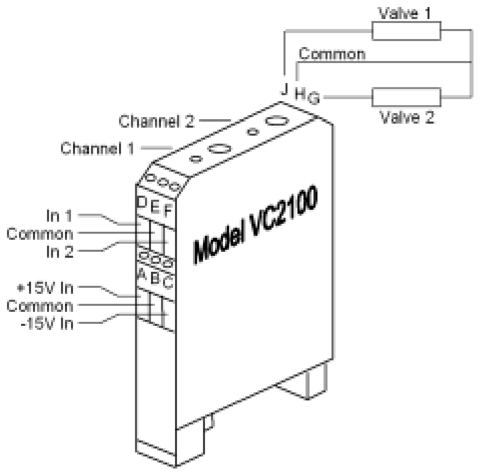
For noise immunity, use twisted, shielded pairs for all connections (twisted pair with overall shield is acceptable). For best noise immunity and CE compliance, keep wires from the RMC to the VC2124 as short as possible and less than 98ft (30m), and place ferrites on all cables as close to the VC2124 as possible. Sample ferrite part numbers from Steward: 28A2029-0A0 or 0A2, 28A5131-0A2, 28A0593-0A2, 28A0807-0A2, 28A3851-0A2, 28A2024-0A0 or 0A2.



VC2100

Fuse the ±15Vdc inputs with 5A maximum, UL-listed, fast-blow fuses. For maximum protection, use two 500mA fuses per VC2100.

For noise immunity, use twisted, shielded pairs for all connections (twisted pair with overall shield is acceptable). For best noise immunity, keep wires from the RMC to the VC2100 as short as possible and less than 98ft (30m), and place ferrites on all cables as close to the VC2124 as possible. Sample ferrite part numbers from Steward: 28A2029-0A0 or 0A2, 28A5131-0A2, 28A0593-0A2, 28A0807-0A2, 28A3851-0A2, 28A2024-0A0 or 0A2.



Terminal	Function
A	+15 volt supply in
B	Power supply common
C	-15 volt supply in
D	Voltage Input 1
E	Common
F	Voltage Input 2
G	Current Output 2

H	Common
J	Current Output 1

4.12 Position/Pressure Control

4.12.1 Position-Pressure Overview

The RMC100 excels at smoothly transitioning from position to pressure (or force) control while in motion. This requires two axes: one for the position control and one for the pressure control. When set up for position/pressure control, the RMC100 effectively controls these two axes as one unit.

For information on only controlling pressure or force, see the Controlling Solely Pressure or Force topic.

For complete instruction on how to set up, tune and control a position/pressure system, read the following topics:

1. This topic.
2. Position/Pressure Setup
3. Tuning a Position/Pressure System
4. Position/Pressure Example

Required Equipment

To control position to pressure transitions, an RMC100 with the following options is required:

- Pressure control option (the designation is RMC101)
- An analog module (for the pressure transducer inputs). An H, G, or A module will work. For details on configuring the module for pressure control and for your particular transducer type refer to Using Analog Channels as Differential Force Inputs, Using Analog Channels as Pressure Inputs and Configuring the Analog Transducer Type.
- A module for position inputs. This module must have an analog drive output (not a stepper output). This may be an MDT, SSI, QUAD, or even the same analog module in the previous item if it has a drive output. Most hydraulic systems use an MDT for position feedback.

Types of Pressure Control

The RMC100 offers two types of pressure control:

- **Set Pressure**
  This is the standard method of control. When the user issues a Set Pressure command with a certain value, the RMC attempts to change the pressure to that value. Note that if the pressure feedback is double-ended, this becomes force control.
- **Pressure Limit**
  This type of pressure control is only available upon request as it is intended for specific applications. It allows the user to do position moves while pressure is being limited to a given value.

Three Basic Modes of Operation

There are three basic modes that must be understood in order to control pressure or force:

- **Position Control Mode:**
  In this mode, the position axis is entirely controlling the drive output. The status fields of the pressure axis will still be updated, but the pressure axis will have no affect on the position drive output. Position axes which do not have assigned pressure axis will always operate in this mode.

- **Pressure Monitoring Mode:**
  In this mode, the position axis is controlling the drive output, but is monitoring the pressure (or force) during the move. As soon as the pressure reaches a minimum pressure threshold (given by the Pressure Set A field), the axis will enter Pressure Regulating Mode. To enter this mode, a move command (either Go (G) or Relative Move (J)) with the Monitor Pressure bit set in the Mode word must be issued.

- **Pressure Regulating Mode:**
  In this mode, the pressure axis is controlling the pressure (or force). The position of the axis will have no effect on the control of the axis except in the following cases:

  - If the Actual Position of the axis moves outside the Extend or Retract limits, the axis goes into open loop mode with the Null Drive output. The axis does not return to pressure monitoring mode unless the pressure falls below another pressure threshold (given by the Pressure Set B field).

  - If the Actual Position reaches the commanded position, the Target Pressure will decrease in order to prevent the Actual Position from exceeding the Command Position. This assumes that the axis is compressing something that may spring back if the exerted force is decreased. To get the Target to increase, re-issue the Set Pressure Command.

The axis may leave pressure regulating mode and return to Position Control by giving a move command without the Monitor Pressure Mode bit set.

A typical position/pressure application uses the above modes in the following order:

1. A move is made in position control mode to the starting position.
2. A move is made in the direction of increasing pressure with the monitor pressure bit set (pressure monitoring mode).
3. When the pressure reaches Pressure Set A, the axis enters pressure regulating mode. Pressure can now be controlled.
4. To exit pressure control, a move is made with the Monitor Pressure bit cleared.

To perform position/pressure control on your system:

- **Set up the RMC100**
  See Position/Pressure Setup for a step-by-step illustrated procedure on how to set up the RMC100 for position/pressure control. Included are instructions for using the Step Table and the commands required for position/pressure control.

- **Tune the system**

See Tuning a Position/Pressure System for a step-by-step procedure on how to tune your system.

- **Example**
  See Position/Pressure Example for a fully detailed, step-by-step, illustrated example of setting up and tuning position/pressure control.

4.12.2 Position-Pressure Setup

This section provides detailed information on how to set up the RMC for position/pressure control.

Read this topic before continuing to Tuning a Position/Pressure System.

The Position/Pressure Overview topic should be read before this topic.

For a practical example of setting up and tuning a system, see Position/Pressure Example.

Steps for Setting Up Position/Pressure Control:

1. **Configure the Axes**
   Once the RMC is correctly wired to the transducers, the modules must be correctly configured.

   a. **Configure the Position Axis**
   The following procedure is for configuring MDT modules. If you have a different module, look up its configuration procedure in the on-line help.

   i. **Set blanking period (if required)**

   - If you have a clevis-mounted transducer, see MDT configuration to change the blanking period from the default 21 ms to 5 ms.

   ii. **Configuration word**

   - Double-click the Config Word for the MDT axis.
   - Select the transducer type you have. See MDT configuration for more details on the Configuration word.

   iii. **Set scale and offset**

   - On the **Tools** menu, click **MDT Scale/Offset Calibration**.
   - In the **Desired Position Units** field, enter the desired measurement units. Many position/pressure applications use 1000 units per inch.
   - In the **Transducer** field, enter the transducer data. This information is available from the transducer specifications.
   - The **Offset** field allows you to set your zero point at any point along the transducer. Move the cylinder to where you want your zero point to be. Click the button in the **Offset** field to automatically enter the counts into the **counts** box.

- If you wish your measurement to be in the opposite direction of the transducer counts, select **decreasing position units** in the **Increasing counts equals** field.
- Set the desired extend and retract limits in the **Extend/Retract Limits** field.
- Click **Apply** or **Done** when you are finished.
- Issue a Set Parameters (P) command to initialize the axis with these settings.

b. Configure the Analog Axis

The following procedure is for configuring the analog axis for pressure control. For further details on configuring the module for pressure control and for your particular transducer type refer to Using Analog Channels as Differential Force Inputs Using Analog Channels as Pressure Inputs and Configuring the Analog Transducer Type.

i. Module Configuration

- On the **Tools** menu, click **Module Configuration**.
- In the **Slots** list, select an analog module.
- Click **Slot options**.
- Click the **Channels 0-1** or **Channels 2-3** tab depending on which channel your pressure transducer is connected to.
- Click an option button to set the correct channels for either auxiliary pressure if you are using single-ended pressure, or auxiliary force if you are using double-ended (differential) pressure.
- Click Update RMC.
- The Update Module Configuration dialog box will be displayed to indicate the progress. If the RMC could not be reset automatically, you may be prompted to reset the RMC manually.
- In the RMC Configuration dialog box, click Close.

ii. Configuration word

- Double-click the Config Word (in the parameter area of the main screen) for the pressure axis.
- In the Input type field, select the type of feedback of your pressure transducer.

iii. Set scale and offset

- On the **Tools** menu, click **Scale/Offset Calibration**.
- If the axis is pressure control:
    - Enter the desired Actual Pressure for two different pressure counts (the counts are the feedback from the transducer). The scale and offset will be automatically calculated.

- If the axis is force control:

  Enter the following information:

  - Maximum gauge reading
  - Actuator Type
  - Cylinder Dimensions
  - Desired Force Units
  - Desired Force Direction

  This information allows the Actual Pressure in the Status area of the main screen to display the net force on the cylinder.

- Click **Apply** or **Done** when you are finished.

2. Assign the Pressure Axis to a Position Axis

The RMC performs pressure control using two or more inputs from transducers and one drive output to the hydraulic valve. Any position axis with analog drive output (not stepper output) can be used to provide the drive output and position input. To enable pressure control on this position axis, you must assign a pressure-enabled analog axis to the position axis. In this configuration, the analog inputs—whether using two transducers monitoring differential force, or a single transducer monitoring single-ended pressure—are then used together with the position axis to control both position and pressure.

To assign a pressure axis to a position axis:

a. Double-click the Config Word for the position axis. This will open the Config Word dialog box.

b. In the Pressure Axis field, click the desired Pressure Axis.

c. Click OK.

d. Issue a Set Parameters (P) command to cause these bits to take effect.

e. If you want the axis assignment to be remembered after the RMC power is cycled, then issue an Update Flash (U) command . This needs to be done only once.

3. Set Up the Event Step Table

Setting up and tuning position/pressure control requires issuing multiple commands numerous times. Doing this from a PLC or by typing commands in RMCWin is very awkward. To simplify the process, Event Steps in RMCWin should be used for setting up and tuning position/pressure control.

To perform position/pressure control, the following steps must be set up in the Event Step Table:

1. Issue a P command so the axis will hold position.
2. Issue a Set Null Drive to Integral Drive (n) command. This updates the null drive (drive needed to hold position), which is critical for pressure control.

3. Move the axis to the correct starting position. Normally, the axis should be at a position where the pressure is below the desired entry pressure (Pressure Set A).

4. Set up the pressure control mode. This sets up the parameters that the pressure axis will use when it enters pressure regulating mode. This is done by issuing the Set Pressure (^) command.
   Note: this step does not put the axis into pressure regulating mode.

   This step sets up the following parameters:

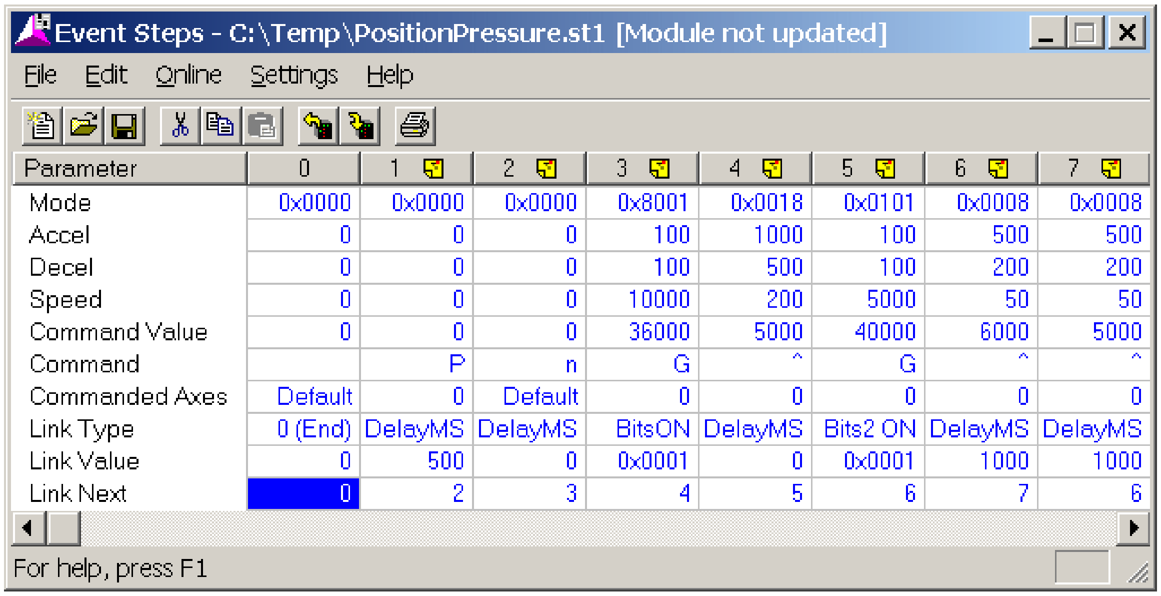
Mode:	Sets Ramp Type, Ramp Time and Integrator Mode
Pressure Set A:	Pressure at which Pressure control begins.
Pressure Set B:	Pressure at which Pressure control ends.
Ramp Time:	Time to ramp to commanded pressure.
Command Pressure:	The pressure the system will go to when regulating pressure.

5.

5. Move in the direction of increasing pressure with the monitor pressure bit set (pressure monitoring mode). When the pressure reaches Pressure Set A, the axis enters pressure regulating mode. The axis enters pressure control according to the parameters previously set up by the Set Pressure command.

6. Control the pressure as desired.

The following sample Event Step Table illustrates these steps:


Event Steps - C:\Temp\PositionPressure.st1 [Module not updated]
File Edit Online Settings Help
For help, press F1

Parameter	0	1	2	3	4	5	6	7
Mode	0x0000	0x0000	0x0000	0x8001	0x0018	0x0101	0x0008	0x0008
Accel	0	0	0	100	1000	100	500	500
Decel	0	0	0	100	500	100	200	200
Speed	0	0	0	10000	200	5000	50	50
Command Value	0	0	0	36000	5000	40000	6000	5000
Command		P	n	G	^	G	^	^
Commanded Axes	Default	0	Default	0	0	0	0	0
Link Type	0 (End)	DelayMS	DelayMS	BitsON	DelayMS	Bits2 ON	DelayMS	DelayMS
Link Value	0	500	0	0x0001	0	0x0001	1000	1000
Link Next	0	2	3	4	5	6	7	6

Explanation of each step in the Event Step table:

Step 0:
This step is normally linked to after finishing a sequence. If the event control is at step 0, it usually means the event control has stopped. This step does nothing.

Step 1:
Issues a P command so the axis will hold position. Waits for 1/2 sec so the drive output will stabilize for the next step.

Step 2:
Issues a Set Null Drive to Integral Drive. This updates the null drive (drive needed to hold position), which is critical for pressure control.

Step 3:
This step moves to 36.0 in. at 10 in./sec (assuming it is set up for a resolution of 0.001 in., then 36000 = 36 in. and 10000 = 10 in.) It then waits until bit 1 (the In Position bit) is on before continuing to the next step.

Step 4:
This step sets up the pressure parameters that will be used in step 5. The Mode word is set up to Calculate Ramp Time and Integrator Active Only at Pressure. Pressure Set A is 1000, Pressure Set B is 500, and the Command Pressure is 5000. This step waits 0 msec and goes to step 5. Note that the Ramp Time, set to 200, is not used since Calculate Ramp Time has been selected.

Step 5:
This step moves toward 40 in. with the Monitor Pressure bit set in the Mode word. Once the pressure reaches Pressure Set A (as set in step 4), it will begin controlling pressure. It will go to a pressure of 5000 (as set in step 4). This step waits until bit 0 (At Pressure) is on before continuing to step 6.

Step 6:
This step changes the Command Pressure to 6000, with the same Pressure Set A and B as in step 4 and a Ramp Time of 50. The pressure will ramp to 6000, and then the step waits 1000 msec before continuing to step 7.

Step 7:
This step changes the Command Pressure to 5000, with the same Pressure Set A and B and Ramp Time values as in step 6. The pressure will ramp to 5000, and then the step waits for 1000 msec before continuing to step 5. Steps 5 and 6 will repeat until a Quit Events (Q) command is issued. Ramping the pressure up and down in this manner is very useful when tuning the pressure. It is important that the Ramp Time be fairly short, because parameters suitable for slow changes (long Ramp Times) may make the system unstable for fast changes.

Notes on using event control for position/pressure:

- To begin an Event Step sequence, enter the desired event number in the Command Value field and issue a Start Events (E) command by typing "E" in the Command field on the main screen.
- Before any commands may be issued to an axis, a Set Parameters (P) command must be issued. This initializes the axis.
- Event control is only allowed on position axes. However, when a pressure command is entered in an Event Step on a position axis that has an assigned

pressure axis, then the command is issued to the pressure axis instead of the position axis.

- For a list of pressure commands that will be sent to the pressure axis, refer to Command. Notice, however, that the Set Mode (M), Open Loop (O), and Set Parameters (P) commands will not be sent to the pressure axis because they are also valid on the position axis.
- When editing the event table you will have to keep in mind that for pressure commands, you should enter values for Pressure Set A in the Acceleration field, values for Pressure Set B in the Deceleration field, and values for Ramp Time in the Speed field.

Once you have set up your Event Steps table, continue to Tuning a Position/Pressure System.

4.12.3 Position-Pressure Example (Part 1)

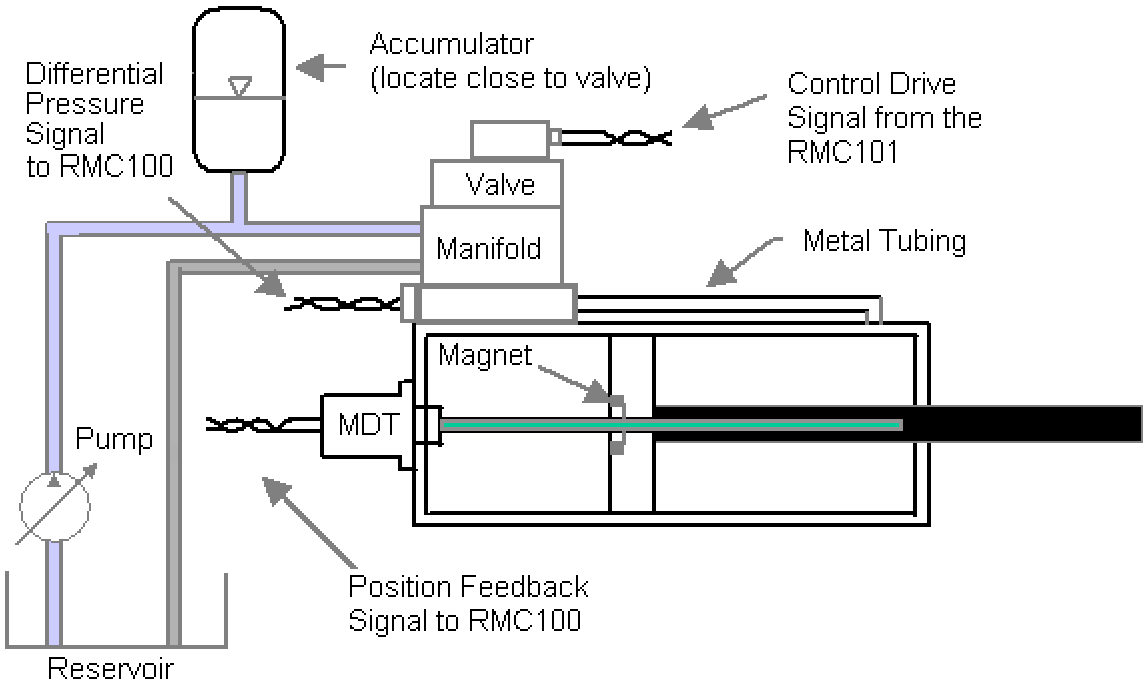
This is a complete step-by-step example of setting up and tuning a position/pressure hydraulic axis. Even if your system does not have pressure control, it may be useful to read the position portions of this example.

This example is divided into 3 parts with subparts:
(each section is a separate help topic)

1. Setup (this topic)
2. Position Tuning
3. Pressure Setup
4. Pressure Tuning
5. Transition Tuning

This example uses an RMC101-M1-A1-ENET, a common configuration for position/pressure control. The system itself is a hydraulic cylinder with MDT feedback (start/stop rising edge) and analog double-ended pressure feedback, making force control possible. If the pressure feedback were single-ended, only pressure control would be possible.

1. **System Design**
   A typical hydraulic system for position/pressure control looks like this:



Note the good design practices:

- Metal tubing between valve and cylinder.
- An accumulator close to valve.
- Use of a zero-overlap valve.

2. Wiring

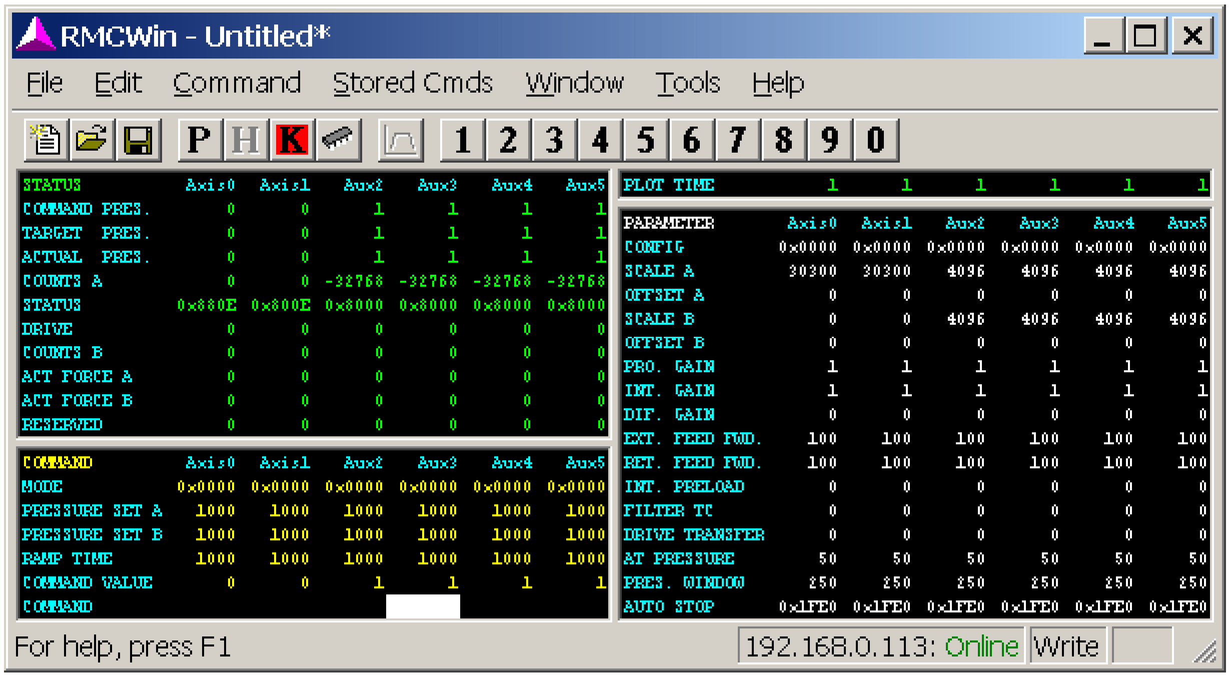
It is important to wire the transducer correctly. See MDT Wiring and Analog Wiring for details.

For this example the system is wired as follows:

- The pressure transducers are wired to the Analog module channels 0 and 1
- The position feedback is wired to MDT channel 0
- The MDT Drv 0 is wired to the valve.

3. RMCWin View

When connected to this RMC, the main RMCWin window looks as follows:


RMCWin - Untitled*
File Edit Command Stored Cmds Window Tools Help
P H K 1 2 3 4 5 6 7 8 9 0

STATUS	Axis0	Axis1	Aux2	Aux3	Aux4	Aux5
COMMAND PRES.	0	0	1	1	1	1
TARGET PRES.	0	0	1	1	1	1
ACTUAL PRES.	0	0	1	1	1	1
COUNTS A	0	0	-32768	-32768	-32768	-32768
STATUS	0x880E	0x800E	0x8000	0x8000	0x8000	0x8000
DRIVE	0	0	0	0	0	0
COUNTS B	0	0	0	0	0	0
ACT FORCE A	0	0	0	0	0	0
ACT FORCE B	0	0	0	0	0	0
RESERVED	0	0	0	0	0	0

COMMAND	Axis0	Axis1	Aux2	Aux3	Aux4	Aux5
MODE	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
PRESSURE SET A	1000	1000	1000	1000	1000	1000
PRESSURE SET B	1000	1000	1000	1000	1000	1000
RAMP TIME	1000	1000	1000	1000	1000	1000
COMMAND VALUE	0	0	1	1	1	1
COMMAND						

PLOT TIME	1	1	1	1	1	1

PARAMETER	Axis0	Axis1	Aux2	Aux3	Aux4	Aux5
CONFIG	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000
SCALE A	30300	30300	4096	4096	4096	4096
OFFSET A	0	0	0	0	0	0
SCALE B	0	0	4096	4096	4096	4096
OFFSET B	0	0	0	0	0	0
PRO. GAIN	1	1	1	1	1	1
INT. GAIN	1	1	1	1	1	1
DIF. GAIN	0	0	0	0	0	0
EXT. FEED FWD.	100	100	100	100	100	100
RET. FEED FWD.	100	100	100	100	100	100
INT. PRELOAD	0	0	0	0	0	0
FILTER TC	0	0	0	0	0	0
DRIVE TRANSFER	0	0	0	0	0	0
AT PRESSURE	50	50	50	50	50	50
PRES. WINDOW	250	250	250	250	250	250
AUTO STOP	0x1FE0	0x1FE0	0x1FE0	0x1FE0	0x1FE0	0x1FE0

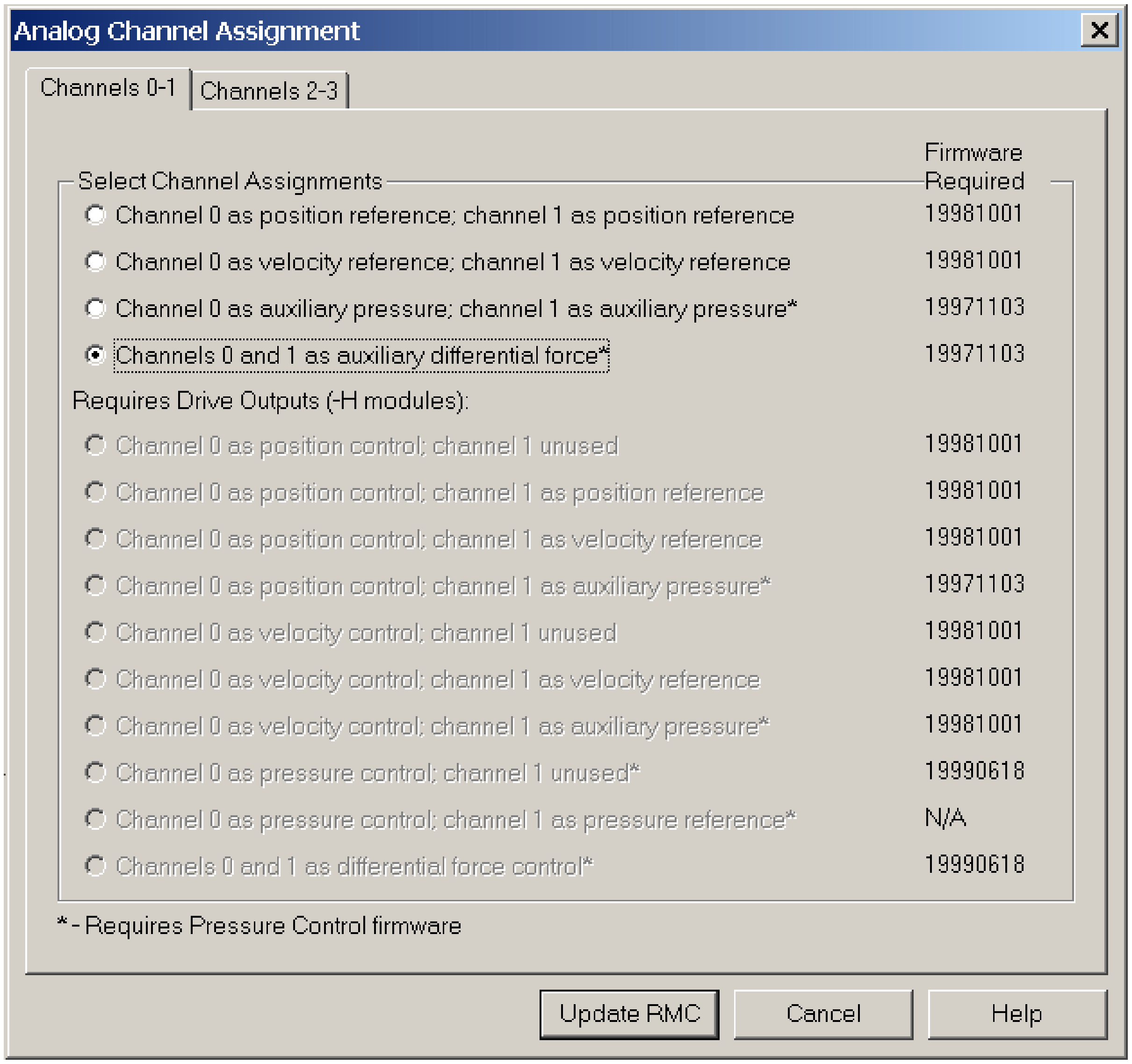
For help, press F1 192.168.0.113: Online Write

- Axes 0 and 1 are the two MDT axes.
- Axes 2-5 are the 4 channels of the Analog module.

4. Configuring the pressure/force axes.

To configure the pressure axis, the following steps are performed:

- On the **Tools** menu, click **Module Configuration**.
- In the **Slots** field, double-click **Analog**. This opens the following window:


Analog Channel Assignment

Channels 0-1 | Channels 2-3

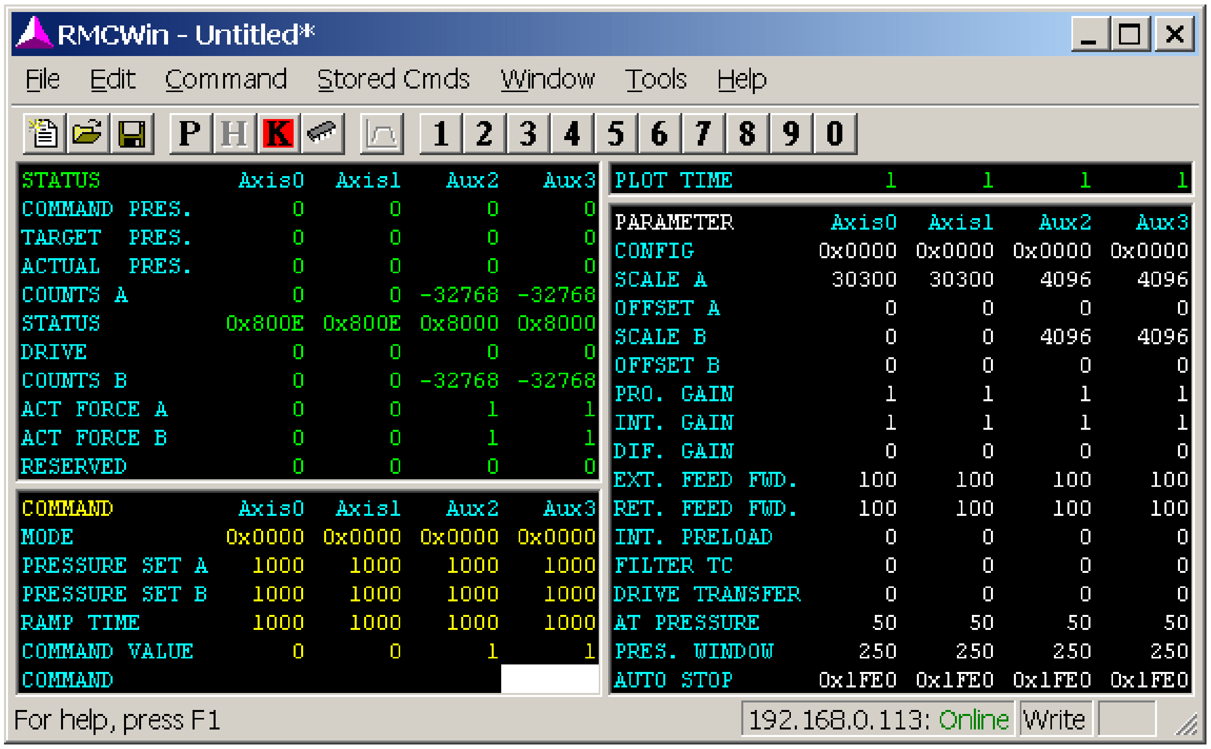
Select Channel Assignments

	Firmware Required
( ) Channel 0 as position reference; channel 1 as position reference	19981001
( ) Channel 0 as velocity reference; channel 1 as velocity reference	19981001
( ) Channel 0 as auxiliary pressure; channel 1 as auxiliary pressure*	19971103
(●) Channels 0 and 1 as auxiliary differential force*	19971103
Requires Drive Outputs (-H modules):	
( ) Channel 0 as position control; channel 1 unused	19981001
( ) Channel 0 as position control; channel 1 as position reference	19981001
( ) Channel 0 as position control; channel 1 as velocity reference	19981001
( ) Channel 0 as position control; channel 1 as auxiliary pressure*	19971103
( ) Channel 0 as velocity control; channel 1 unused	19981001
( ) Channel 0 as velocity control; channel 1 as velocity reference	19981001
( ) Channel 0 as velocity control; channel 1 as auxiliary pressure*	19981001
( ) Channel 0 as pressure control; channel 1 unused*	19990618
( ) Channel 0 as pressure control; channel 1 as pressure reference*	N/A
( ) Channels 0 and 1 as differential force control*	19990618

* - Requires Pressure Control firmware

- Since the system is double-ended pressure, select the auxiliary differential force option.
- Do the same on the **Channels 2-3** tab.
- Click **Update RMC**.
- When the RMC is finished updating, click **Close**.

The RMCWin window now looks like this:


RMCWin - Untitled*
File Edit Command Stored Cmds Window Tools Help
P H K 1 2 3 4 5 6 7 8 9 0

STATUS	Axis0	Axis1	Aux2	Aux3
COMMAND PRES.	0	0	0	0
TARGET PRES.	0	0	0	0
ACTUAL PRES.	0	0	0	0
COUNTS A	0	0	-32768	-32768
STATUS	0x800E	0x800E	0x8000	0x8000
DRIVE	0	0	0	0
COUNTS B	0	0	-32768	-32768
ACT FORCE A	0	0	1	1
ACT FORCE B	0	0	1	1
RESERVED	0	0	0	0

COMMAND	Axis0	Axis1	Aux2	Aux3
MODE	0x0000	0x0000	0x0000	0x0000
PRESSURE SET A	1000	1000	1000	1000
PRESSURE SET B	1000	1000	1000	1000
RAMP TIME	1000	1000	1000	1000
COMMAND VALUE	0	0	1	1
COMMAND				

PLOT TIME	1	1	1	1

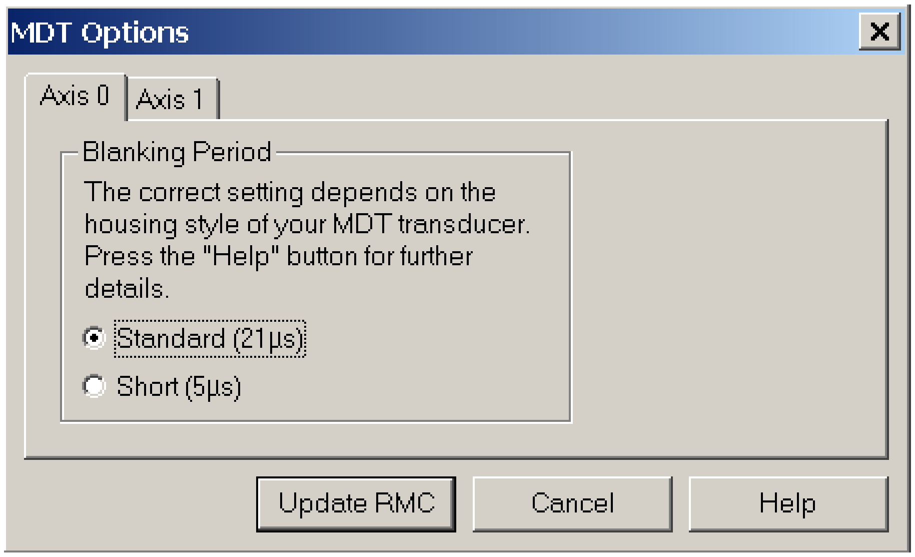
PARAMETER	Axis0	Axis1	Aux2	Aux3
CONFIG	0x0000	0x0000	0x0000	0x0000
SCALE A	30300	30300	4096	4096
OFFSET A	0	0	0	0
SCALE B	0	0	4096	4096
OFFSET B	0	0	0	0
PRO. GAIN	1	1	1	1
INT. GAIN	1	1	1	1
DIF. GAIN	0	0	0	0
EXT. FEED FWD.	100	100	100	100
RET. FEED FWD.	100	100	100	100
INT. PRELOAD	0	0	0	0
FILTER TC	0	0	0	0
DRIVE TRANSFER	0	0	0	0
AT PRESSURE	50	50	50	50
PRES. WINDOW	250	250	250	250
AUTO STOP	0x1FE0	0x1FE0	0x1FE0	0x1FE0

For help, press F1 | 192.168.0.113: Online | Write

- Notice that there are only 4 axes now, because each force axis uses two channels on the Analog module.
- Axis 0 is the MDT axis, and Aux 2 is the pressure axis.

5. **Configuring the MDT position axis.**
   To scale the MDT position axis, the following steps are performed:

- On the **Tools** menu, click **Module Configuration**.
- In the **Slots** field, double-click on MDT, which opens the following window:



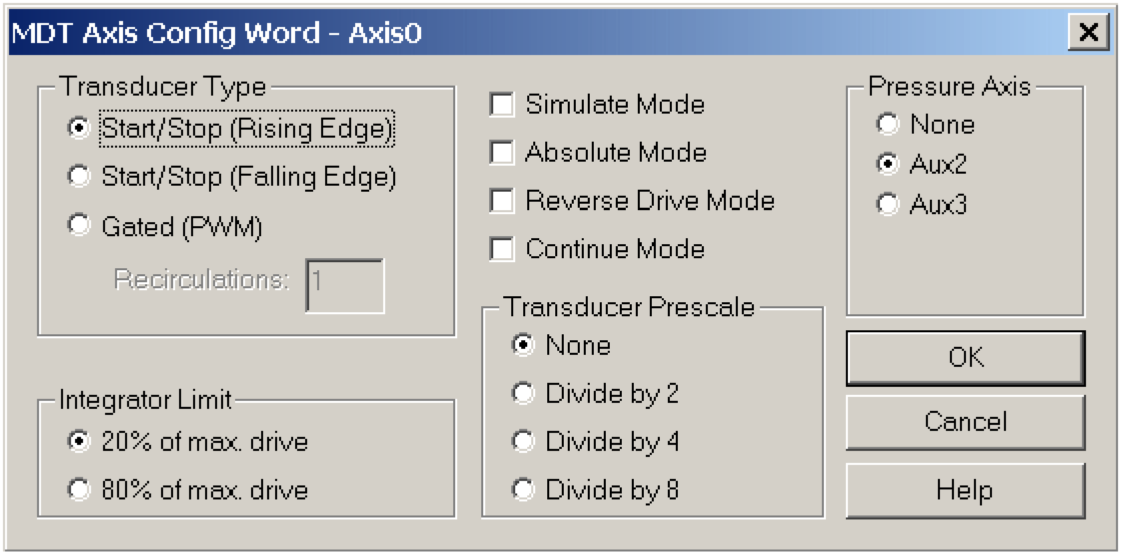
Blanking Period

The correct setting depends on the housing style of your MDT transducer. Press the "Help" button for further details.

● Standard (21µs)

○ Short (5µs)

- It is already set to 21 ms, which is correct. If the MDT transducer is a clevis-mount type, choose 5 ms.
- Click **Update RMC**.
- Double-click the Axis 0 Config Word, which opens the following window:



MDT Axis Config Word - Axis0

Transducer Type
- (●) Start/Stop (Rising Edge)
- ( ) Start/Stop (Falling Edge)
- ( ) Gated (PWM)

Recirculations: 1

- [ ] Simulate Mode
- [ ] Absolute Mode
- [ ] Reverse Drive Mode
- [ ] Continue Mode

Pressure Axis
- ( ) None
- (●) Aux2
- ( ) Aux3

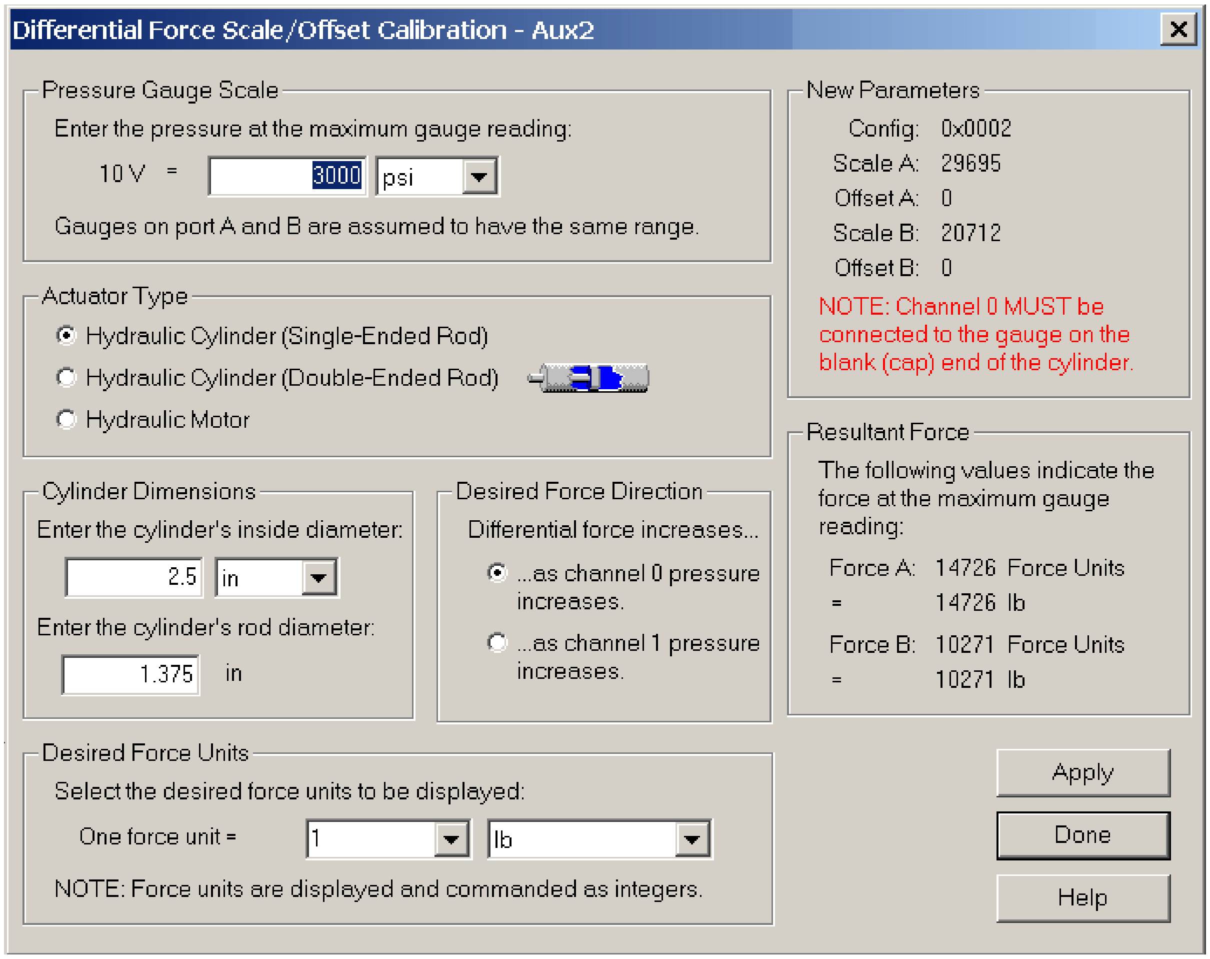
Transducer Prescale
- (●) None
- ( ) Divide by 2
- ( ) Divide by 4
- ( ) Divide by 8

Integrator Limit
- (●) 20% of max. drive
- ( ) 80% of max. drive

- In the **Transducer Type** field, select **Start/Stop (Rising Edge)**.
- In the **Pressure Axis** field, select **Aux 2** as the pressure axis. This assigns the pressure axis to this position axis. This is necessary for the RMC to be able to control position and pressure on axis 0.
- Leave all the other settings as they are.
- Click **OK**.
- Issue a Set Parameters (P) command to Axis 0.

6. **Scaling the pressure/force axis.**
   To configure the pressure axis, the following steps are performed:

- Click on any Aux 2 field.
- On the **Tools** menu, click **Scale/Offset Calibration**. The following window opens:


Differential Force Scale/Offset Calibration - Aux2

Pressure Gauge Scale

Enter the pressure at the maximum gauge reading:

10 V = 3000 psi

Gauges on port A and B are assumed to have the same range.

Actuator Type

(●) Hydraulic Cylinder (Single-Ended Rod)

( ) Hydraulic Cylinder (Double-Ended Rod)

( ) Hydraulic Motor

Cylinder Dimensions

Enter the cylinder's inside diameter: 2.5 in

Enter the cylinder's rod diameter: 1.375 in

Desired Force Direction

Differential force increases...

(●) ...as channel 0 pressure increases.

( ) ...as channel 1 pressure increases.

Desired Force Units

Select the desired force units to be displayed:

One force unit = 1 lb

NOTE: Force units are displayed and commanded as integers.

New Parameters

Config:	0x0002
Scale A:	29695
Offset A:	0
Scale B:	20712
Offset B:	0

NOTE: Channel 0 MUST be connected to the gauge on the blank (cap) end of the cylinder.

Resultant Force

The following values indicate the force at the maximum gauge reading:

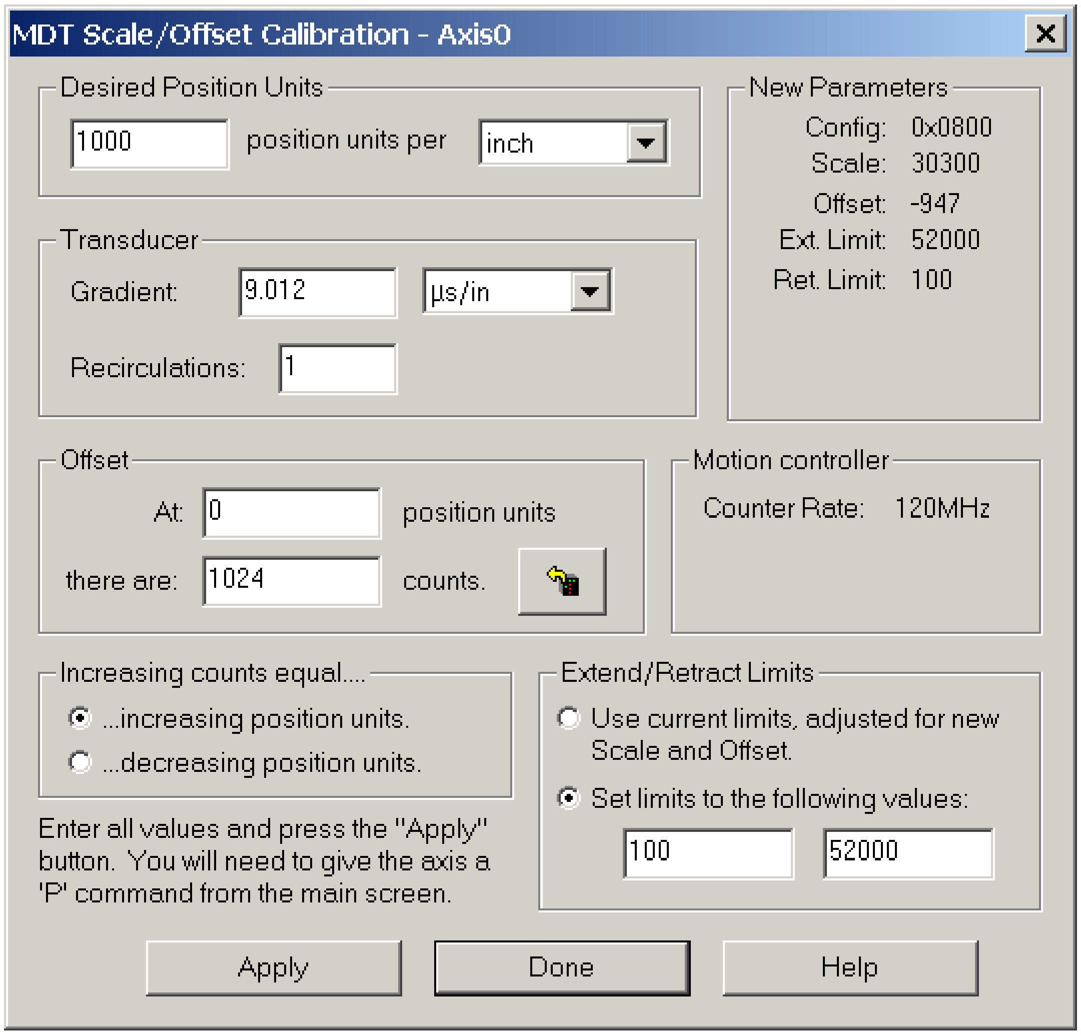
Force A:	14726	Force Units
=	14726	lb
Force B:	10271	Force Units
=	10271	lb

- The information is entered as shown in the picture:
    - o The pressure transducer has a max reading of 3000 psi, so "3000" and "psi" are entered in the **Pressure Gauge Scale**.
    - o The cylinder is a single-ended rod, so this option is selected in the **Actuator Type** field.
    - o The cylinder dimensions are 2.5 in. inside diameter and 1.375 in. rod diameter.
    - o The Force units are to be displayed in lbs, so "1" and "lbs" are entered in the **Desired Force Units** field.
    - o The **Desired Force Direction** is selected such that the force increases when channel 0 pressure increases.
    - o Note that channel 0 must be connected to the gauge on the blank (cap) side of the cylinder.
- Click **Apply** and **Done**.
- Issue a Set Parameters (P) command to initialize the axis with these values.
- Note that if the pressure were single-ended, the Scale/Offset dialog would be different.

7. **Scaling the MDT position axis.**

To scale the MDT position axis, the following steps are performed:

- Click on any Axis 0 field.
- On the **Tools** menu, click **MDT Scale/Offset Calibration**. The following window opens:



MDT Scale/Offset Calibration - Axis0

Desired Position Units: 1000 position units per inch

Transducer
Gradient: 9.012 µs/in
Recirculations: 1

New Parameters
Config: 0x0800
Scale: 30300
Offset: -947
Ext. Limit: 52000
Ret. Limit: 100

Offset
At: 0 position units
there are: 1024 counts.

Motion controller
Counter Rate: 120MHz

Increasing counts equal....
[✓ ...increasing position units.]
...decreasing position units.

Extend/Retract Limits
Use current limits, adjusted for new Scale and Offset.
[✓ Set limits to the following values:] 100 52000

Enter all values and press the "Apply" button. You will need to give the axis a 'P' command from the main screen.

- The position feedback is to be measured in thousandths of inches, so 1000 pos units per inch is entered in the **Desired Position Units** field.
- The transducer gradient information in the **Transducer** field is found on the transducer.
- To set the offset, the following steps are performed
    - Retract the cylinder all the way. This can be done by issuing an Open Loop (O) command with a small amount of drive.
    - When the cylinder is all the way retracted, the transducer shows 1024 counts in this example.
    - If this is the desired zero point, click the button in the **Offset** field and it automatically enters

1024 into the counts field.

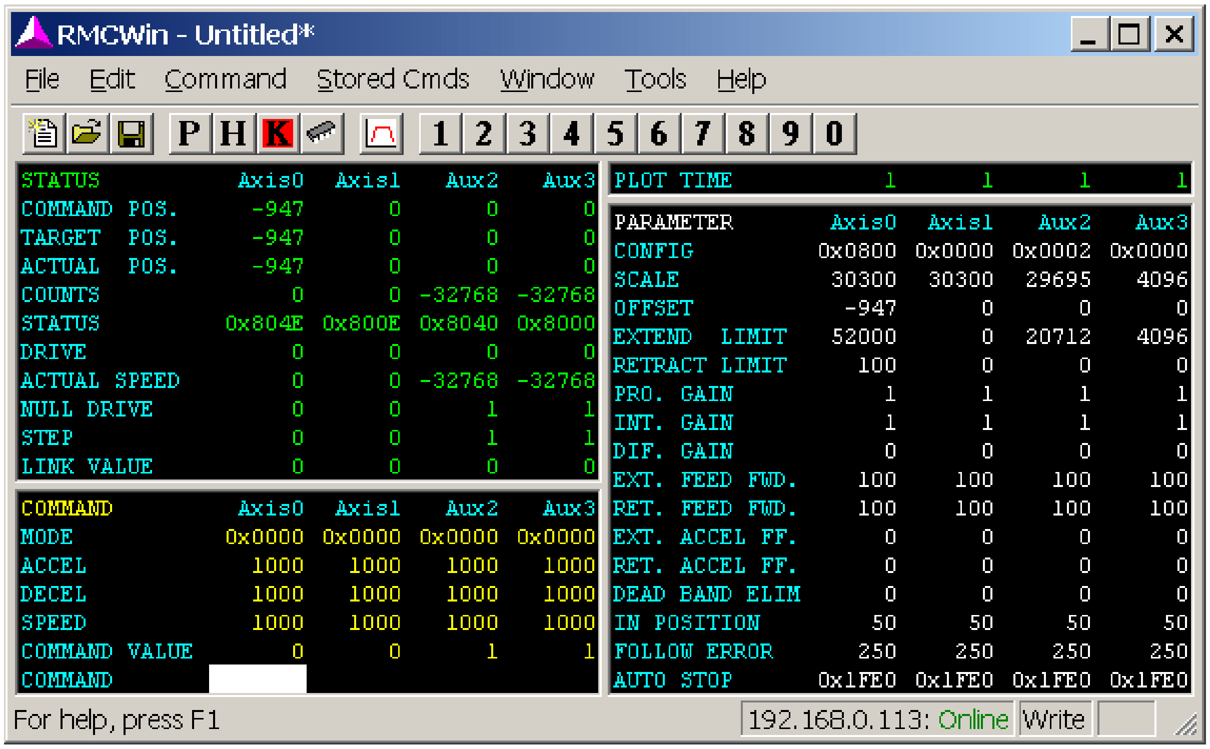
- o **…increasing counts** is selected in the **Increasing counts equal…** field.

- In the **Extend/Retract Limits** field, click **Set limits to the following values:**. Set the desired extend and retract limits to 0.100 and 52 inches. This system can extend 52.700 inches, but it is undesirable to ever reach either end.

- Click **Apply** and **Done**.

- Issue a Set Parameters (P) command to Axis 0.

The main RMCWin window looks now like this:



```
STATUS            Axis0   Axis1    Aux2    Aux3
COMMAND  POS.      -947       0       0       0
TARGET   POS.      -947       0       0       0
ACTUAL   POS.      -947       0       0       0
COUNTS                0       0  -32768  -32768
STATUS           0x804E  0x800E  0x8040  0x8000
DRIVE                 0       0       0       0
ACTUAL SPEED          0       0  -32768  -32768
NULL DRIVE            0       0       1       1
STEP                  0       0       1       1
LINK VALUE            0       0       0       0

COMMAND           Axis0   Axis1    Aux2    Aux3
MODE             0x0000  0x0000  0x0000  0x0000
ACCEL              1000    1000    1000    1000
DECEL              1000    1000    1000    1000
SPEED              1000    1000    1000    1000
COMMAND VALUE         0       0       1       1
COMMAND

PLOT TIME               1       1       1       1

PARAMETER           Axis0   Axis1    Aux2    Aux3
CONFIG             0x0800  0x0000  0x0002  0x0000
SCALE               30300   30300   29695    4096
OFFSET               -947       0       0       0
EXTEND   LIMIT      52000       0   20712    4096
RETRACT LIMIT         100       0       0       0
PRO.  GAIN              1       1       1       1
INT.  GAIN              1       1       1       1
DIF.  GAIN              0       0       0       0
EXT.  FEED  FWD.      100     100     100     100
RET.  FEED  FWD.      100     100     100     100
EXT.  ACCEL  FF.        0       0       0       0
RET.  ACCEL  FF.        0       0       0       0
DEAD BAND ELIM          0       0       0       0
IN POSITION            50      50      50      50
FOLLOW ERROR          250     250     250     250
AUTO STOP          0x1FE0  0x1FE0  0x1FE0  0x1FE0

For help, press F1            192.168.0.113: Online  Write
```

This example is continued in the next topic.

4.12.4 Position-Pressure Example (Part 2)

This is part 2 of the complete step-by-step example of setting up and tuning a position/pressure axis.

Part 2: Tuning position.

Now the system is set up and ready to tune for position. The procedure in Tuning a Position Axis is followed:

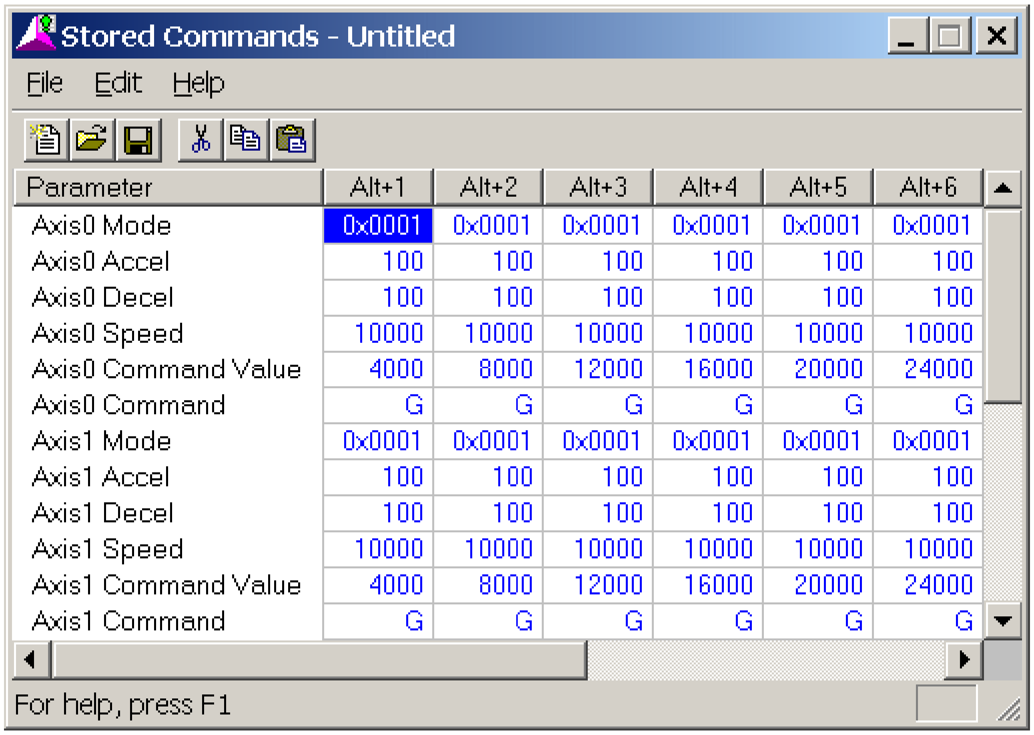
- Issue Open Loop (O) commands with small positive and negative drives to see if the cylinder goes in the right direction.
- If the cylinder moves in the wrong direction, the wiring to the drive may be swapped, or Reverse Drive mode can be selected in the Config word.
- Check the deadband. This system has a very small deadband, approx 5 millivolts, so we will not worry about it.
- Set the gains and feed forwards to zero and issue a Set Parameters (P) command.
- Double-click the Axis 0 Auto Stop word (in the Parameters field).
    - o Set the Overdrive, Integrator Windup and Overdrive bits to Status only. Note that is possible only on systems where safety is not important.

o Set the proportional gain to a small value, such as 5. Issue a Set Parameters (P) command.

o Set the Plot Time for Axis 0 to 4.

o Now make a move. The tuning process will consist of repeatedly making moves. To easily make repeated moves, use Stored Commands.

- o To edit and view the Stored Commands, click **Stored Command Editor** on the **Stored Commands** menu. The following window appears:

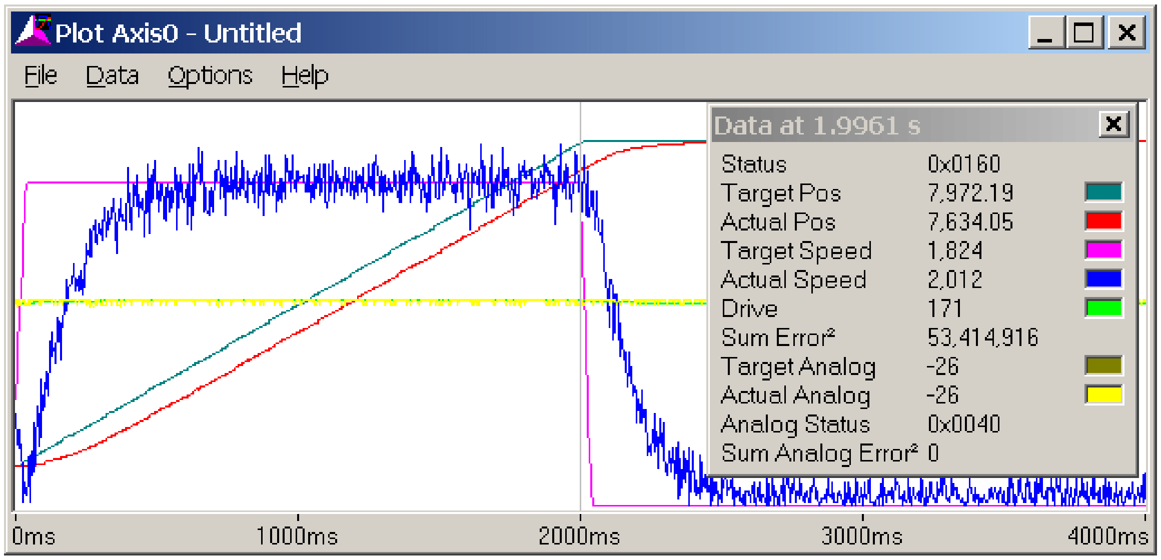


Parameter	Alt+1	Alt+2	Alt+3	Alt+4	Alt+5	Alt+6
Axis0 Mode	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001
Axis0 Accel	100	100	100	100	100	100
Axis0 Decel	100	100	100	100	100	100
Axis0 Speed	10000	10000	10000	10000	10000	10000
Axis0 Command Value	4000	8000	12000	16000	20000	24000
Axis0 Command	G	G	G	G	G	G
Axis1 Mode	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001
Axis1 Accel	100	100	100	100	100	100
Axis1 Decel	100	100	100	100	100	100
Axis1 Speed	10000	10000	10000	10000	10000	10000
Axis1 Command Value	4000	8000	12000	16000	20000	24000
Axis1 Command	G	G	G	G	G	G

- o For example, **ALT-1** on axis 0 will issue a Go cmd to 4000 at a speed of 2 in/sec, with acceleration rates of 10 in/(sec*sec). These are common initial values for moving hydraulic systems. Thes values are editable from this window.
- o To use the stored commands, click on any Axis 0 field, then press e.g. ALT-1 to move to 4

in. and ALT-5 to move to 20 in. The command will be issued to the axis where the cursor is located.

- Using the Stored Commands, a move is made in this example from 2 in. to 8 in. The move will take less time than what the Plot Time is set to so that the entire move can be viewed.
- After making the move, press the Insert key. This opens the Plot window with the last move. It looks like this:

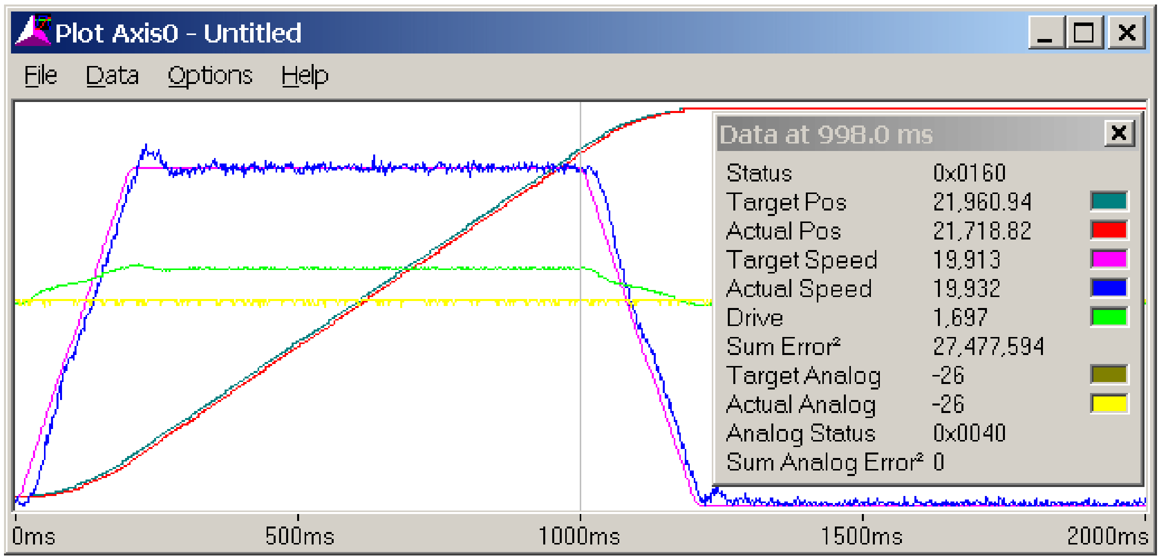


Data at 1.9961 s

Status	0x0160	
Target Pos	7,972.19	🟦
Actual Pos	7,634.05	🟥
Target Speed	1,824	🟪
Actual Speed	2,012	🟦
Drive	171	🟩
Sum Error²	53,414,916	
Target Analog	-26	🟫
Actual Analog	-26	🟨
Analog Status	0x0040	
Sum Analog Error²	0	

  - The detail window in the plot may be opened by clicking **Show Detail Window** on the **Data** menu. The detail window provides much useful information.
  - In this plot, the colored lines represent:
    - Red Actual Position
      Dark Blue-Green Target Position (Desired Position)
      Blue Actual Speed
      Pink Target Speed
      Green Drive Output (volts)
      Yellow Pressure
  - Note the Sum Error Squared. This number gives an indication of the position error throughout the entire move. If the same move is repeated, this value can be compared to previous moves to determine whether the control has improved or worsened. If the number decreases, the control has improved.

- If the system controls fairly well, which this one does, increase the speed. Here it is increased to 20.000 in/sec. The plot time can also be reduced to 2.
- Increase the proportional gain until some oscillation appears, then back off 10-20%. On this system that point is about 70. The plot looks like this:

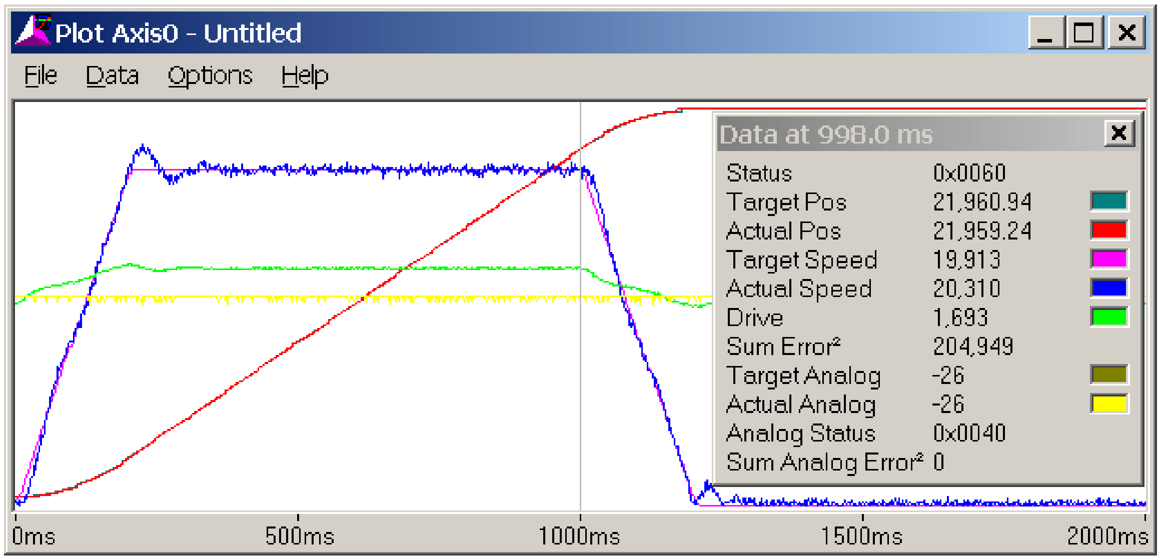


Data at 998.0 ms

Status	0x0160	
Target Pos	21,960.94	🟦
Actual Pos	21,718.82	🟥
Target Speed	19,913	🟪
Actual Speed	19,932	🟦
Drive	1,697	🟩
Sum Error²	27,477,594	
Target Analog	-26	🟫
Actual Analog	-26	🟨
Analog Status	0x0040	
Sum Analog Error²	0	

- o The Sum Error Squared decreased from 53 million to 27 million.
- o The position lags during the constant speed portion of the move, which the next step will address.

- Make a fairly long move without any oscillations. After the move has completed, issue a Feed Forward Adjust (F) command. The Extend Feed Forward value is automatically updated to 84 in this case.
- Make a move in the opposite direction. Issue a Feed Forward Adjust (F) command. The Retract Feed Forward value is automatically updated to 84 in this case.
- Make a move. The axis should track much better now:

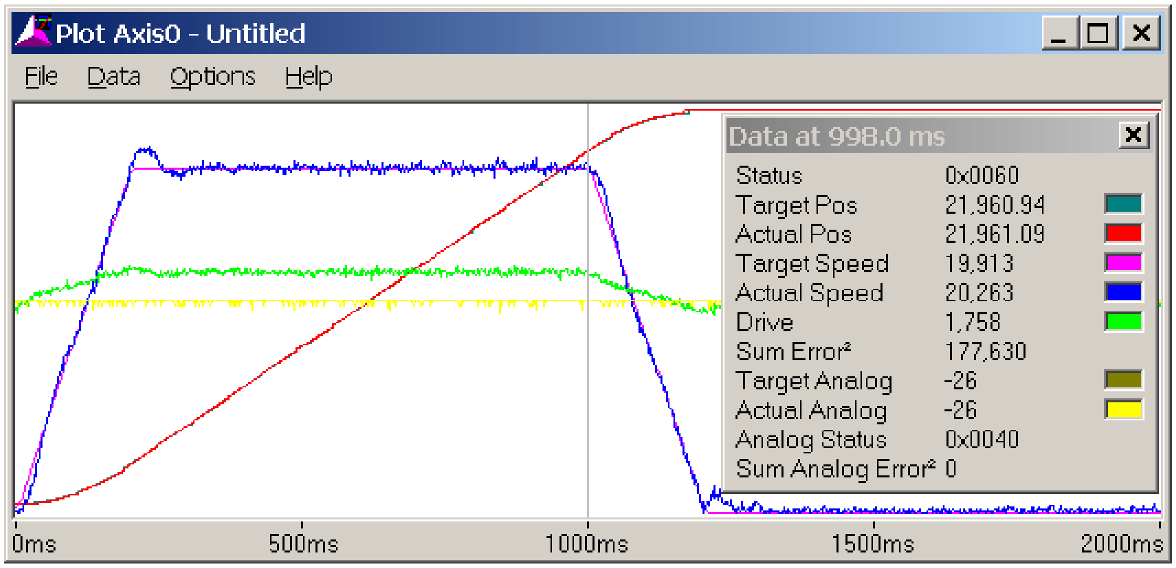


Data at 998.0 ms

Status	0x0060	
Target Pos	21,960.94	🟦
Actual Pos	21,959.24	🟥
Target Speed	19,913	🟪
Actual Speed	20,310	🟦
Drive	1,693	🟩
Sum Error²	204,949	
Target Analog	-26	🟫
Actual Analog	-26	🟨
Analog Status	0x0040	
Sum Analog Error²	0	

- o The Sum Error Squared decreased from 27 million to 204,000!

- The next step is the integral gain. Gradually increase it as long as it does not adversely affect the system. Generally, the Integral gain does not need to be extremely high. In this example, 150 was deemed enough. The plot (not shown) did not change much from the previous one. This does not mean the integral gain won't help! It is important for dynamic changes that may be encountered during system operation and may not be present during the tuning.
- Increase the Differential Gain. In this example, it was increased to 250. When the Differential gain is high enough, the drive output may look fuzzy:

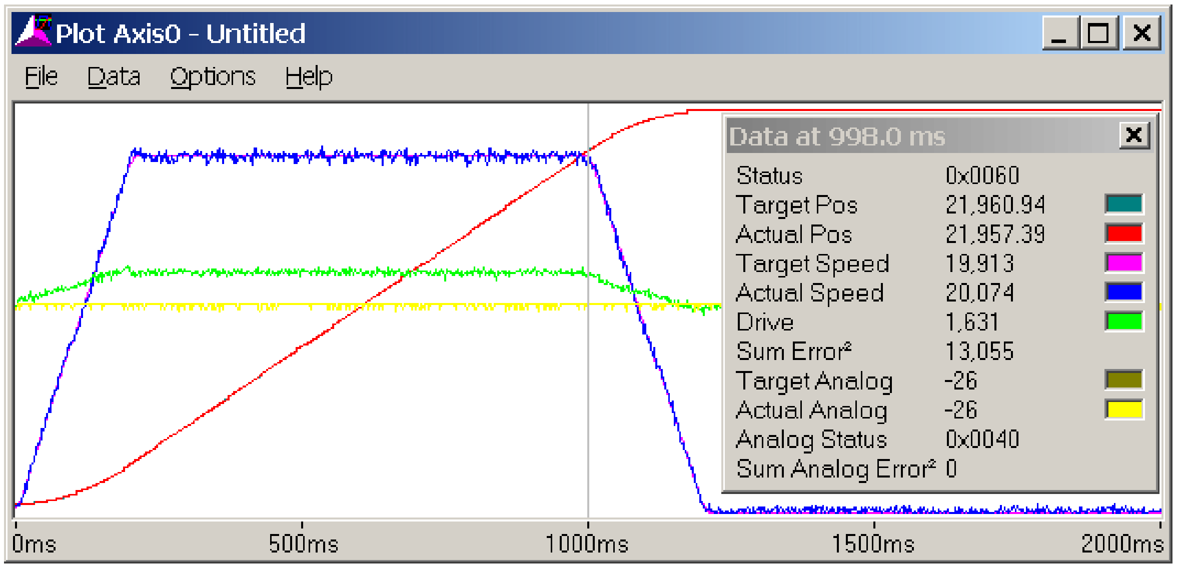


Data at 998.0 ms

Status	0x0060	
Target Pos	21,960.94	🟦
Actual Pos	21,961.09	🟥
Target Speed	19,913	🟪
Actual Speed	20,263	🟦
Drive	1,758	🟩
Sum Error²	177,630	
Target Analog	-26	🟫
Actual Analog	-26	🟨
Analog Status	0x0040	
Sum Analog Error²	0	

  - If the system starts oscillating or chattering, decrease the Differential Gain.
  - The Sum Error Squared decreased to 177,000.

- Increase the Acceleration Feed Forwards to get rid of the following errors during acceleration and deceleration.
- After increasing the Acceleration Feed Forwards to 200, the plot looks like this:

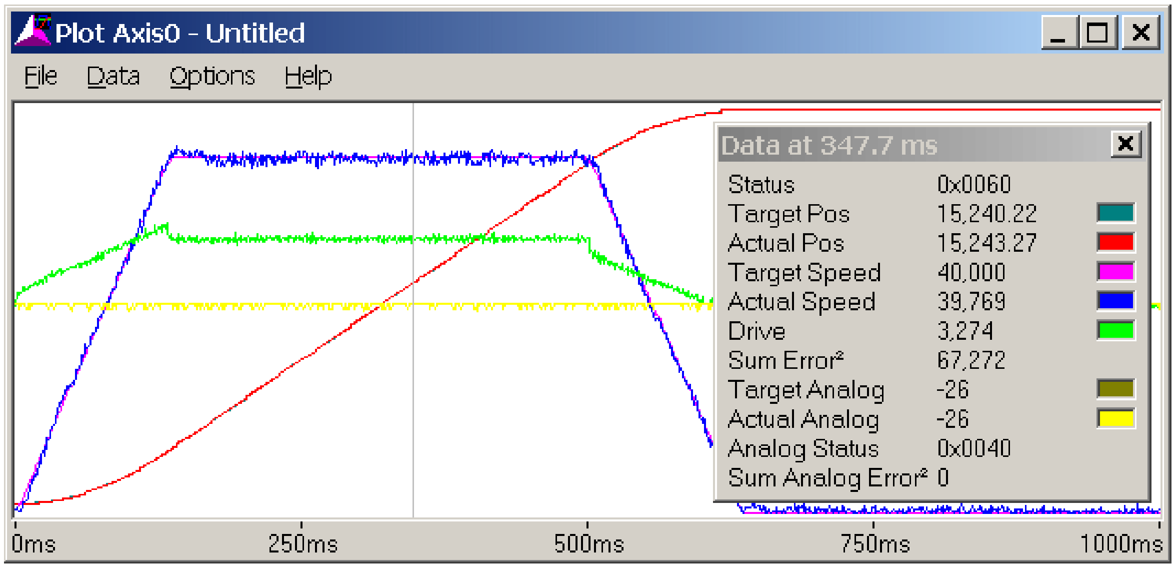


Data at 998.0 ms

Status	0x0060	
Target Pos	21,960.94	🟦
Actual Pos	21,957.39	🟥
Target Speed	19,913	🟪
Actual Speed	20,074	🟦
Drive	1,631	🟩
Sum Error²	13,055	
Target Analog	-26	🟫
Actual Analog	-26	🟨
Analog Status	0x0040	
Sum Analog Error²	0	

- o The Sum Error Squared decreased to 13,000!
- o The overshoot disappeared.
- o The Extend and Retract Feed Forwards are generally not the same for hydraulic systems.

- The position is now tuned. To check that this will work at higher speeds, increase the Accel and Decel to 300 and the speed to 40000. The plot (plot time = 1 sec) looks like this:



Data at 347.7 ms

Status	0x0060	
Target Pos	15,240.22	🟦
Actual Pos	15,243.27	🟥
Target Speed	40,000	🟪
Actual Speed	39,769	🟦
Drive	3,274	🟩
Sum Error²	67,272	
Target Analog	-26	🟫
Actual Analog	-26	🟨
Analog Status	0x0040	
Sum Analog Error²	0	

- o The parameters work well for the higher speed (no lagging, overshoot or oscillation), and will be left unchanged.
- o The Sum Error Squared increased, but that is only because of the higher speed, which inherently results in greater position errors.

This example is continued in the next topic.

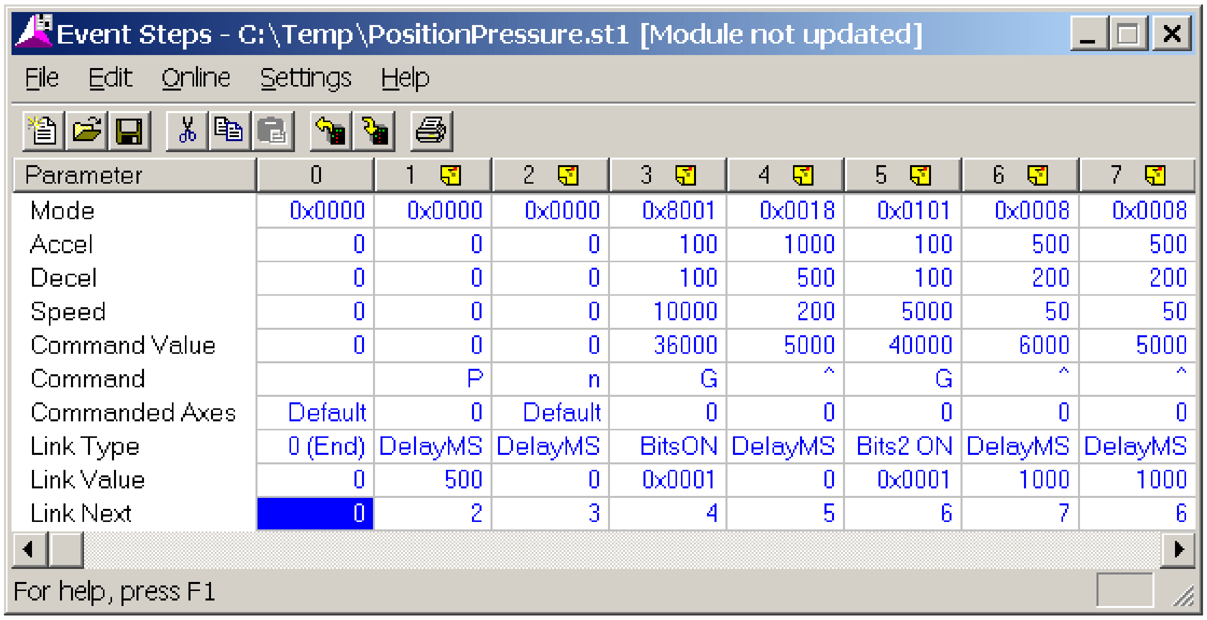
4.12.5 Position-Pressure Example (Part 3)

This is part 3 of the complete step-by-step example of setting up and tuning a position/pressure axis.

Part 3: Setting up for Pressure Control

Now that the position control is tuned, the pressure can be tuned. First, the step table must be set up.

- Set up a step table to simplify entering pressure control. The step table is as follows:


Event Steps - C:\Temp\PositionPressure.st1 [Module not updated]
File Edit Online Settings Help
For help, press F1

Parameter	0	1	2	3	4	5	6	7
Mode	0x0000	0x0000	0x0000	0x8001	0x0018	0x0101	0x0008	0x0008
Accel	0	0	0	100	1000	100	500	500
Decel	0	0	0	100	500	100	200	200
Speed	0	0	0	10000	200	5000	50	50
Command Value	0	0	0	36000	5000	40000	6000	5000
Command		P	n	G	^	G	^	^
Commanded Axes	Default	0	Default	0	0	0	0	0
Link Type	0 (End)	DelayMS	DelayMS	BitsON	DelayMS	Bits2 ON	DelayMS	DelayMS
Link Value	0	500	0	0x0001	0	0x0001	1000	1000
Link Next	0	2	3	4	5	6	7	6

Explanation of the steps:

TIP: To view the options selected in the Mode word of each Event Step, open the Event Step table, enter the step exactly as it appears in the figure, and then double-click on the Mode word for that step.

Step 0:
This step does nothing.

Step 1:
Issues a P command so the axis will hold position. Waits for 1/2 sec so the drive output will stabilize for the next step.

Step 2:
Issues a Set Null Drive to Integral Drive (n) command. This updates the null drive (the drive needed to hold position), which is critical for pressure control.

Step 3:
This step moves to 36.0 in. at 10 in./sec (assuming it is set up for a resolution of 0.001 in., then 36000 = 36 in. and 10000 = 10 in./sec) It then waits until bit 1 (the In Position bit) is on before continuing to the next step.

Step 4:
This step sets up the pressure parameters that will be used in step 5. The Mode word is set up to Calculate Ramp Time and Integrator Active Only at Pressure. Pressure Set A (the point at which pressure control begins) is 1000, Pressure Set B (the point at which pressure control ends) is 500, and the Command Pressure is 5000. This step waits 0 msec and goes to step 5. Note that the Ramp Time, set to 200, is not used since Calculate Ramp Time has been selected.

Step 5:
This step moves toward 40 in. with the Monitor Pressure bit set in the Mode word. Once the pressure reaches Pressure Set A (as set in step 4), it will begin controlling pressure. It will go to a pressure of 5000 (as set in step 4). This step waits until bit 1 (At Pressure) is on before continuing to step 6.

Step6:
This step changes the Command Pressure to 6000, Pressure Set A to 500, Pressure Set B to 500, and sets the Ramp Time to 50. The pressure will ramp to 6000, and then the step waits 500 msec before continuing to step 7.

Step 7:
This step changes the Command Pressure to 5000, with the same Pressure Set A and B and Ramp Time values as in step 6. The pressure will ramp to 5000, and then the step waits for 500 msec before continuing to step 7. Steps 6 and 7 will repeat continuously.

This example is continued in the next topic.

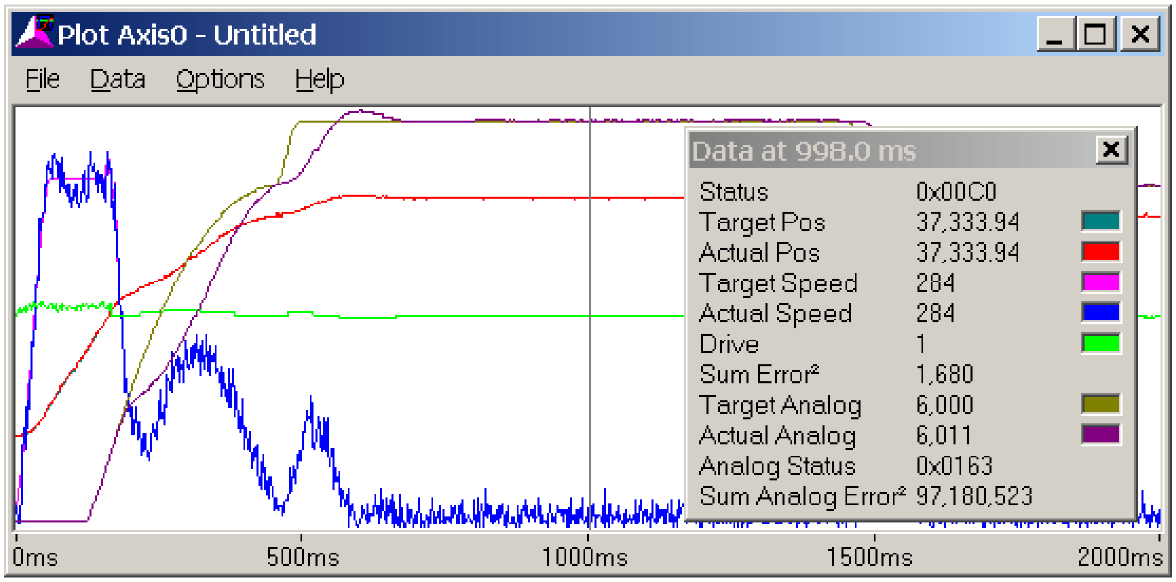
4.12.6 Position-Pressure Example (Part 4)

This is part 4 of the complete step-by-step example of setting up and tuning a position/pressure axis.

Part 4: Tuning Pressure

The pressure is tuned with the following steps:

- Set the Proportional Gain on the pressure axis (Aux 2) to a small value, such as 2. Set all other Aux 2 gains and feed forwards to 0 and issue a P command.
- Run the event sequence # 1. This will make a move that goes into pressure control and then ramps between pressures. The plot looks like this:

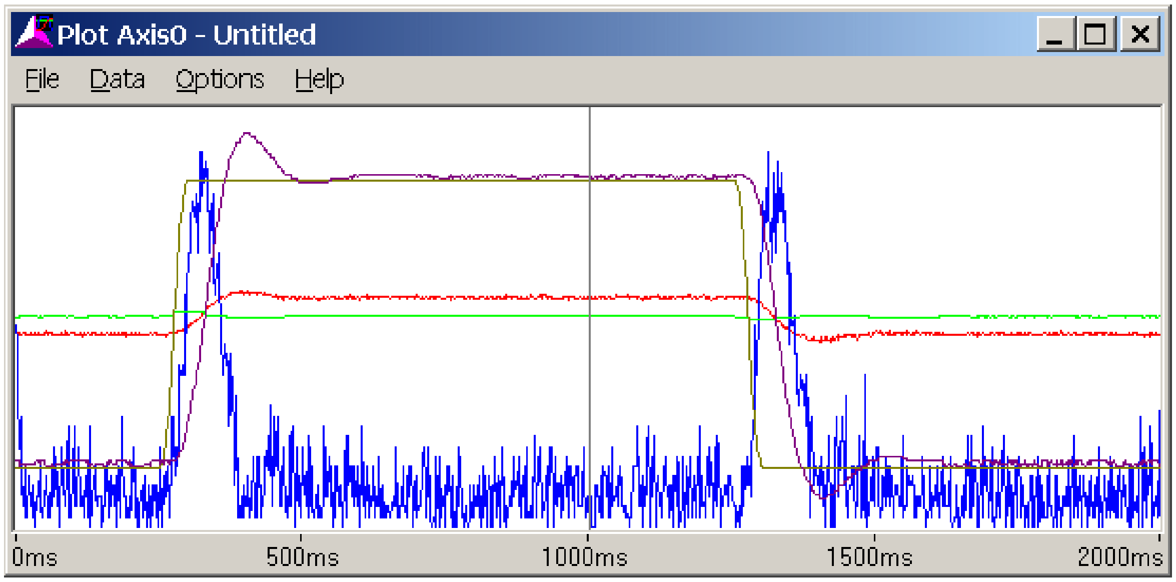


Data at 998.0 ms

Status	0x00C0	
Target Pos	37,333.94	🟦
Actual Pos	37,333.94	🟥
Target Speed	284	🟪
Actual Speed	284	🟦
Drive	1	🟩
Sum Error²	1,680	
Target Analog	6,000	🟫
Actual Analog	6,011	🟪
Analog Status	0x0163	
Sum Analog Error²	97,180,523	

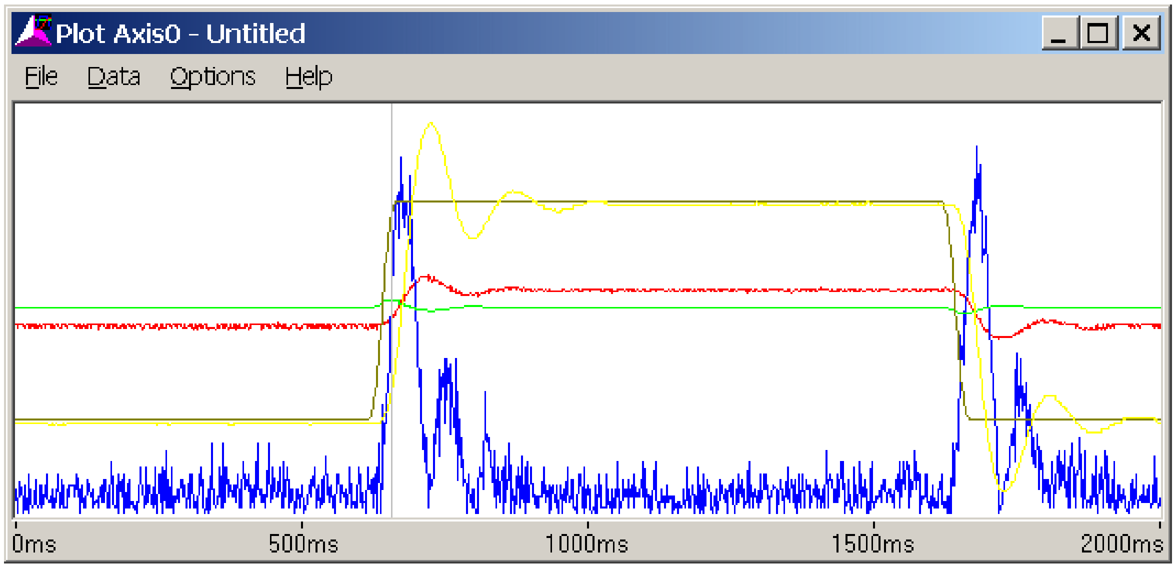
- o Note how the pressure (yellow) begins increasing and when it reaches 1000, pressure regulation begins. The white line is the target pressure.
- o Note the large undershoot as pressure regulation begins.

- Once the axis is in pressure control, type 'y' in the Axis 0 command field. This will start a plot. Press the Insert key to view the plot:



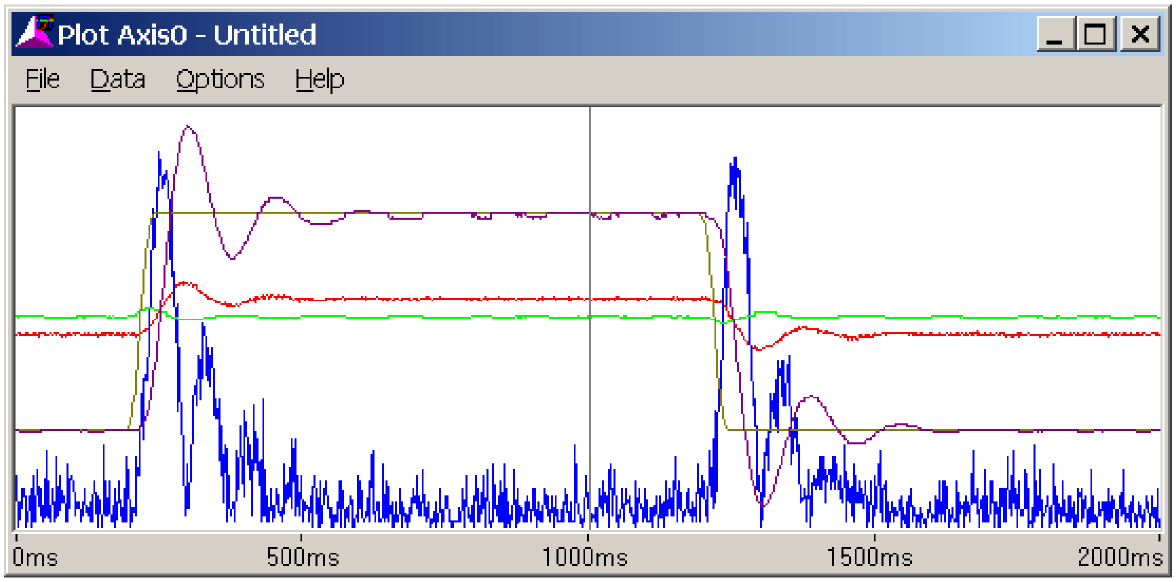
- o The plot shows the pressure ramping up and down. The actual pressure lags the target pressure and does not reach the target pressure during the constant pressure portions of the plot.
- o In this discussion, the Sum Error Squared has been left out because the visual inspection will be enough. However, keep in mind that the Sum Error Squared is a powerful aid for pressure also, and can be used if you wish.

- To change the gains now, simply enter the gain, issue a "P" command, and start a plot. There is no need to run event step 1, because the pressure is ramping up and down endlessly. The gain will take affect as soon as the "P" command is issued. To view the plot, type 'y' in the command field to start the plot, and press the Insert key to view the plot.
- After increasing the Proportional Gain to 4, it looks like this:



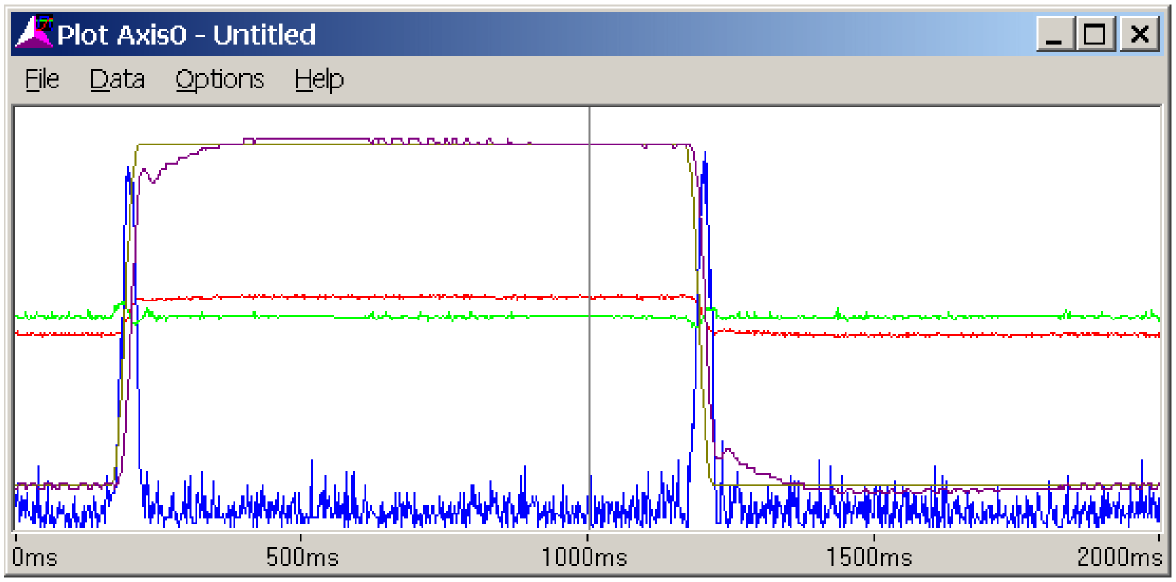
  - o Note how it responds faster, but has some oscillation.

- The next step is to adjust the Integral Gain. Setting it to 15 results in this:



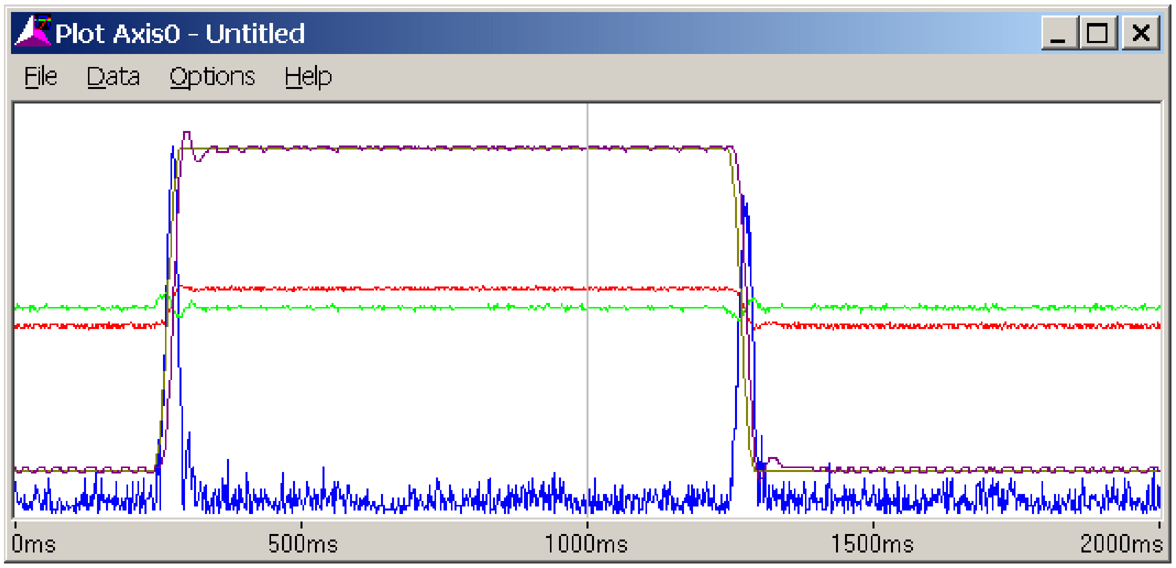
  - o Even though adding integral gain caused the oscillation to worsened a little, it is still necessary. The next step will fix the oscillation.

- The next step is to adjust the Differential Gain. Gradually increasing it to 250 results in this:



  - o It removed the oscillations.

- Now that the Differential Gain has been added, the Proportional Gain can be further increased. After trying several values, 8 seems the best for tightest control and least oscillation:



  - o Notice that the pressure reaches the command pressure faster.
  - o Note that there is a small amount of overshoot, but that is because the pressure is ramping so quickly. In normal operation the pressure would have to ramp slower for no overshoot.

This example is continued in the next topic.

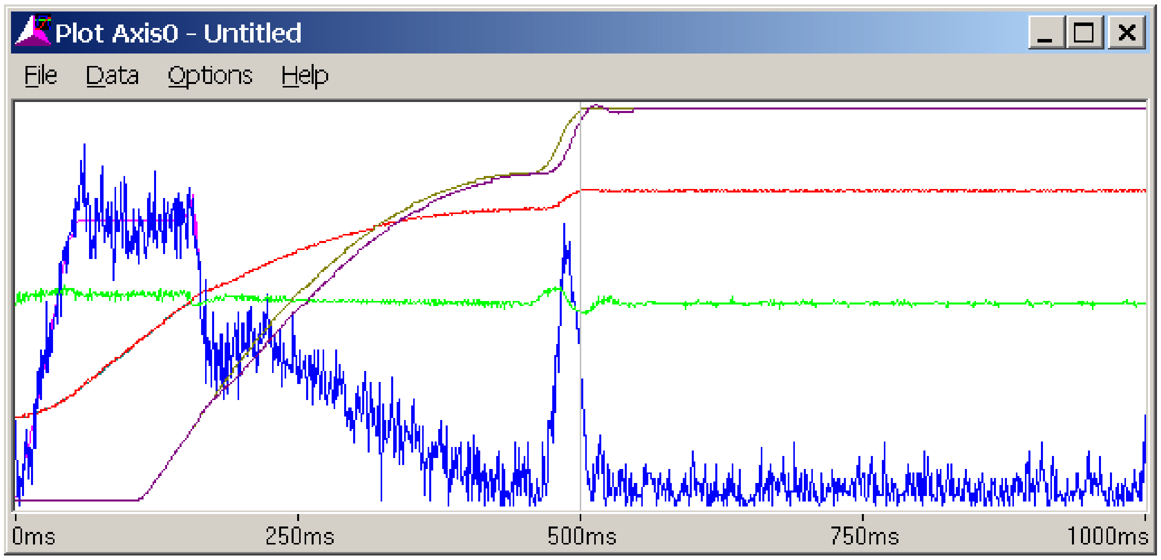
4.12.7 Position-Pressure Example (Part 5)

This is Part 5 of the complete step-by-step example of setting up and tuning a position/pressure axis.

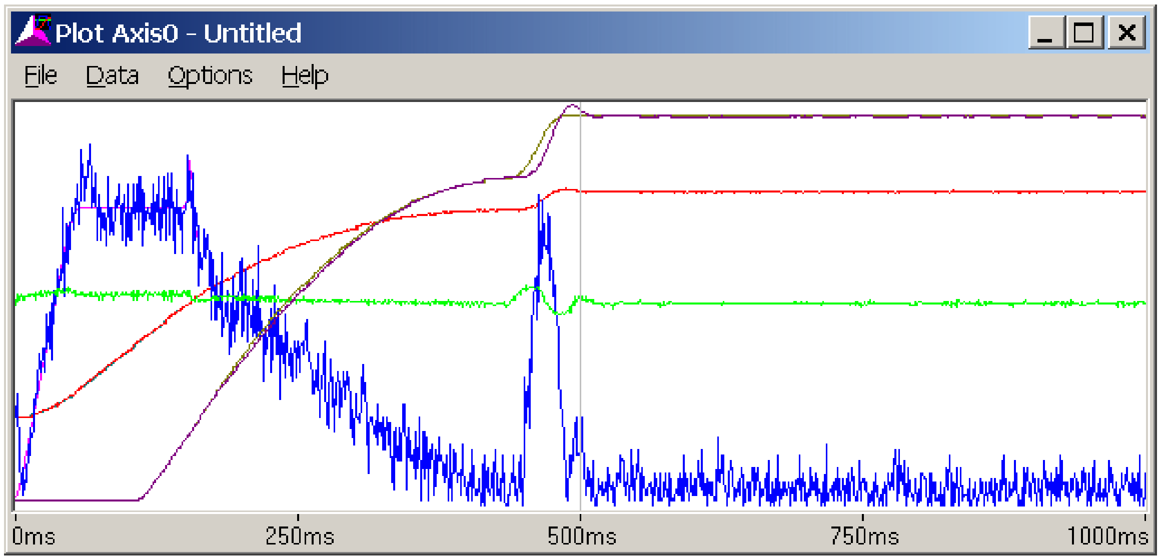
Part 5: Tuning the position-to-pressure transition.

Now that the position has been tuned and the pressure has been tuned, we can focus on the transition from position to pressure.

- Run Event Step 1 again. It looks like this:



  - o The pressure lags after pressure control begins.
  - o Add Feed Forward to keep pressure from lagging. In this case, the Feed Forwards were set to 13:

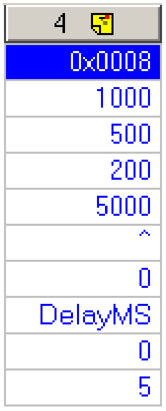


- Notice that there is some overshoot when the pressure ramps up and down. This is because the rate of change in pressure is high. During normal operation, the rate should be lower. The system is tuned for a high rate because a system stable at a high rate will generally be stable at a lower rate. The converse is not true, i.e. a system tuned for a low rate will not necessarily be stable at a higher rate.

- If the rate of change in pressure is changed (i.e. the ramp time or commanded pressure is changed) the feed forwards may also need to be changed.

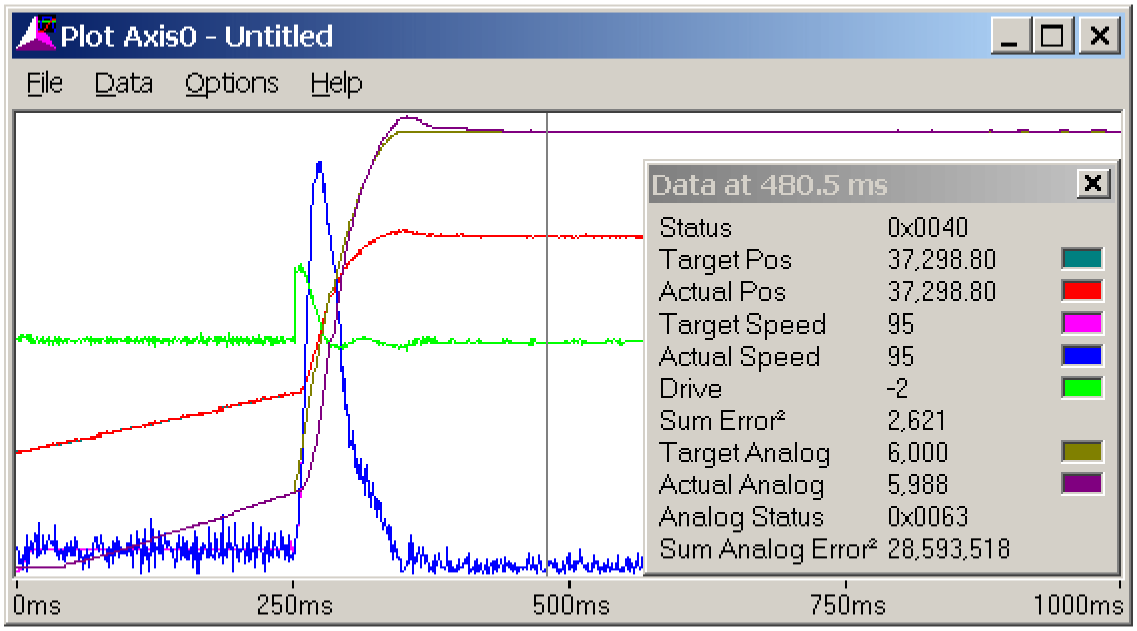
- The tuning of the transition in this case was unusually simple because "Calculate ramp time" was selected in the Mode word of Event Step 4 (step 4 sets up the pressure before the transition). The RMC automatically calculates a ramp time that will work well. However, some applications require a specific ramp time.

- To try the "Set ramp time" option, double-click the Mode word of Event Step 4 and in the **Ramp Time** field click **Use ramp time value**. Click **OK**. Set the ramp time to 200 (in the Speed field). Step 4 now looks like this:



4
0x0008
1000
500
200
5000
^
0
DelayMS
0
5

- Download the Step table to the RMC by clicking the button in the Event Steps window. Run Event Step 1. The plot looks like this:



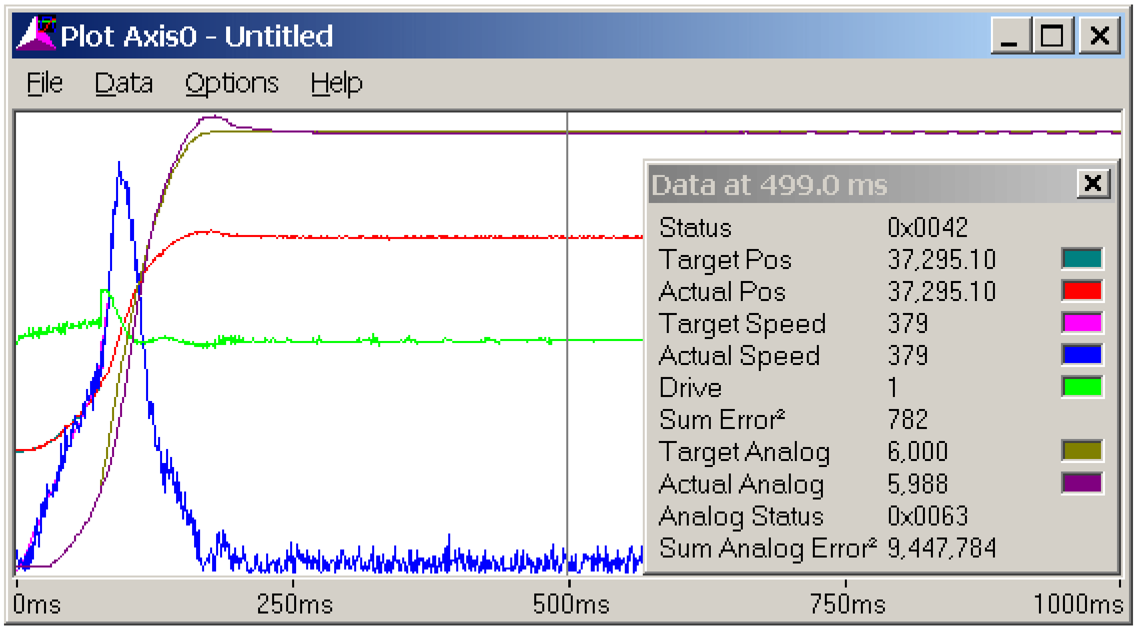
Data at 480.5 ms

Status	0x0040	
Target Pos	37,298.80	🟦
Actual Pos	37,298.80	🟥
Target Speed	95	🟪
Actual Speed	95	🟦
Drive	-2	🟩
Sum Error²	2,621	
Target Analog	6,000	🟫
Actual Analog	5,988	🟪
Analog Status	0x0063	
Sum Analog Error²	28,593,518	

- o Note how the target pressure begins rising immediately once the pressure reaches Pressure Set A, causing the actual pressure to lag. Because of the short ramp time, the pressure overshoots when it reaches the commanded pressure.

- To keep the pressure from lagging when entering pressure control in this case, the speed entering pressure control can changed to closer match the axis speed once pressure control has been entered. This will result in a smaller change in speed at the transition, and less pressure lag.

- Currently, the entering speed is 1000 (1 in/sec, from Step 6, Speed field). If we change this to 10000 and run Event Step 1, the plot looks like this:



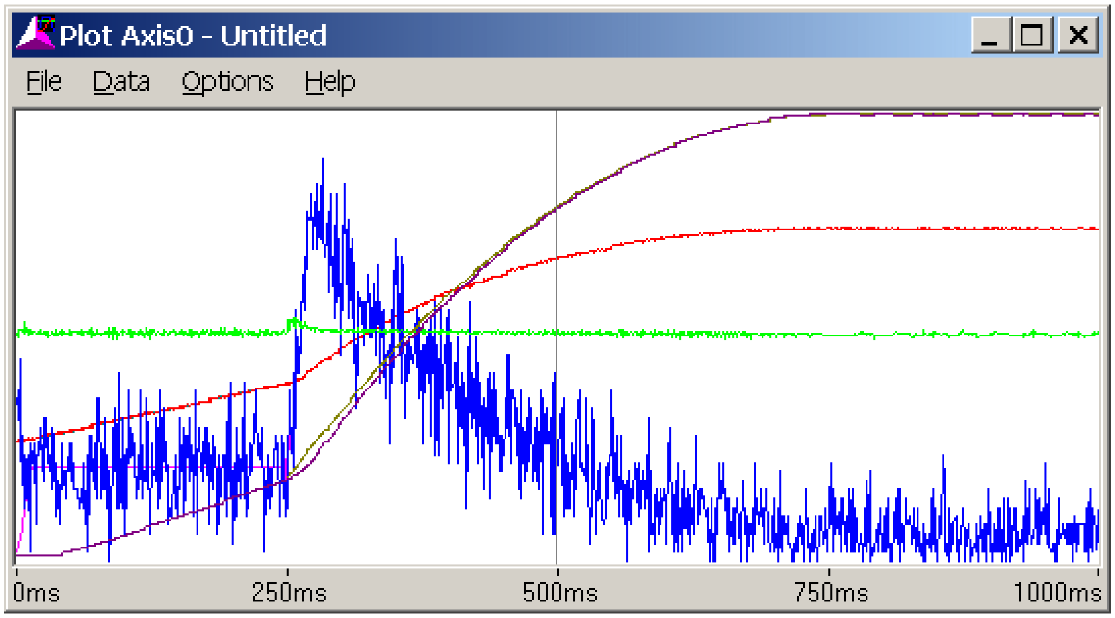
Data at 499.0 ms

Status	0x0042	
Target Pos	37,295.10	🟦
Actual Pos	37,295.10	🟥
Target Speed	379	🟪
Actual Speed	379	🟦
Drive	1	🟩
Sum Error²	782	
Target Analog	6,000	🟫
Actual Analog	5,988	🟪
Analog Status	0x0063	
Sum Analog Error²	9,447,784	

  - o The speed discontinuity upon the transition is much smaller. The speed can be further changed if desired.

  - o The pressure still overshoots. The simplest solution for this is to use a longer ramp time. It may be possible to further tune the system, but it is likely that the system response simply can't handle such fast pressure changes.

- o Lengthening the ramp time to 1000 msec (Step 4, Speed field), changing the Speed in Step 5 back to 1000 and running Event Step 1 results in the following plot:



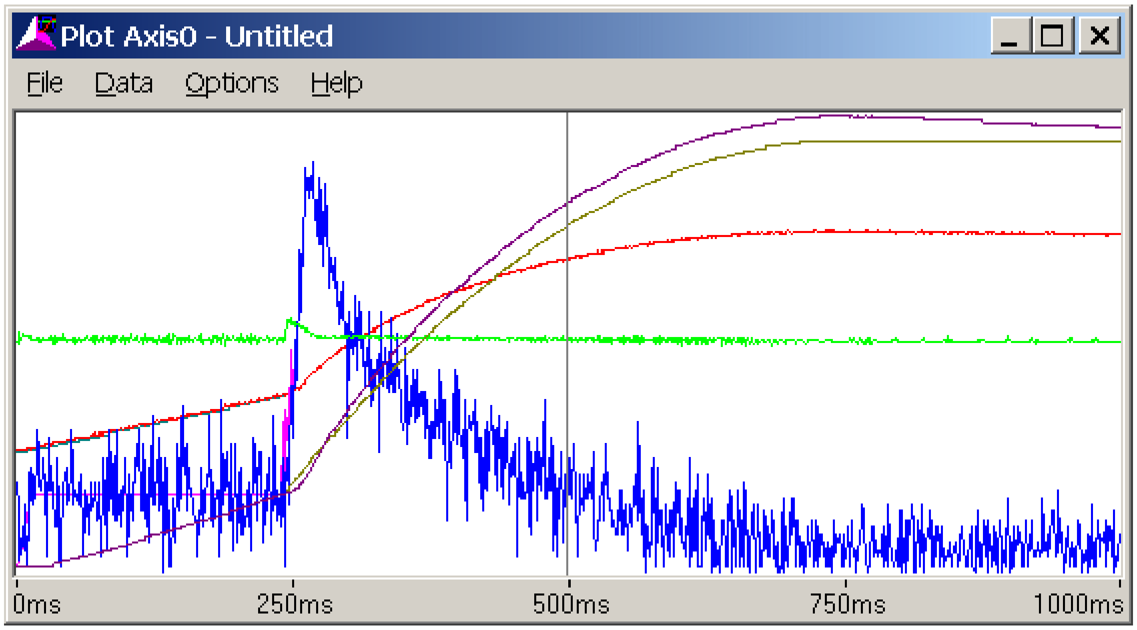
- o The overshoot disappeared because the pressure did not change as quickly. There is some lag upon the transition, but can likely be corrected by adjusting the entering speed.

- Another method of eliminating the lag immediately after the transition is to use the following two parameters:
  - o Integrator Preload
  - o Drive Transfer Percent

  Both of these parameters add drive to the Integral Drive when pressure control begins. The difference between these two is that Integrator Preload adds a set value to the drive, while Drive Transfer Percent adds a percentage of the drive immediately prior to the transition.

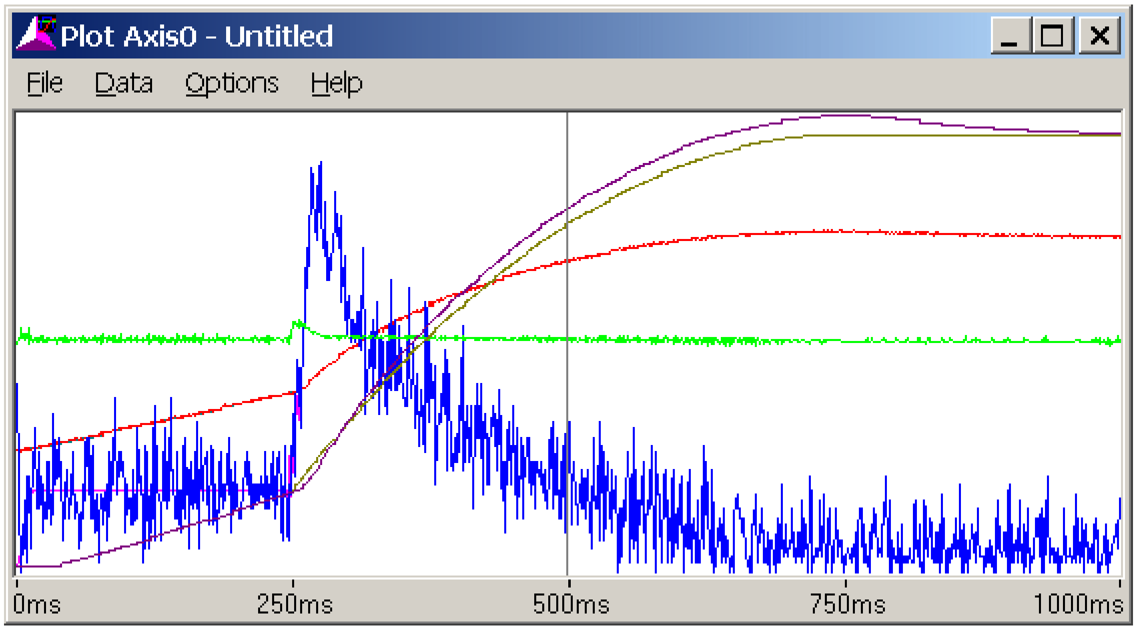
- The Drive Transfer Percent is useful when the entering conditions vary, such as the speed. The Integral Preload is useful when the entering conditions are the same every time.
  Warning! These parameters add drive to the Integral Drive. In order for this drive to "unwind" (go towards zero), the Integral Gain must be greater than zero! If the Integral Drive cannot unwind, the system will not control properly.

- Setting the Integral Preload to 200 (200 mV of drive), issuing a P cmd, and then running Event Step 1 results in the following plot:

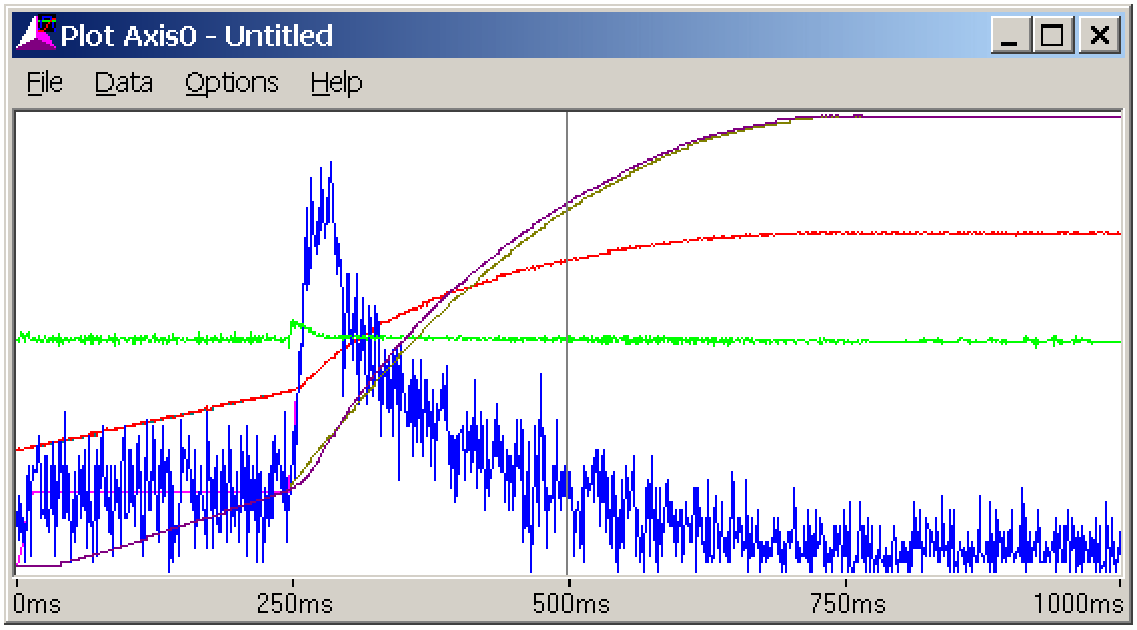


- o Note that the pressure lags much less immediately after the transition, but the pressure begins to lead because the Integral Drive has not yet unwound. This is probably because "Integrator Active Only at Pressure" was selected in the mode word of Step 4. This causes the integrator to be inactive until the pressure reaches the commanded pressure. The integrator cannot unwind until it reaches commanded pressure.

- Selecting "Always Active" in the Integrator Mode field of the Mode word in Step 4 (do this by double-clicking the Mode word in Step 4) and running event Step 1 results in the following plot:

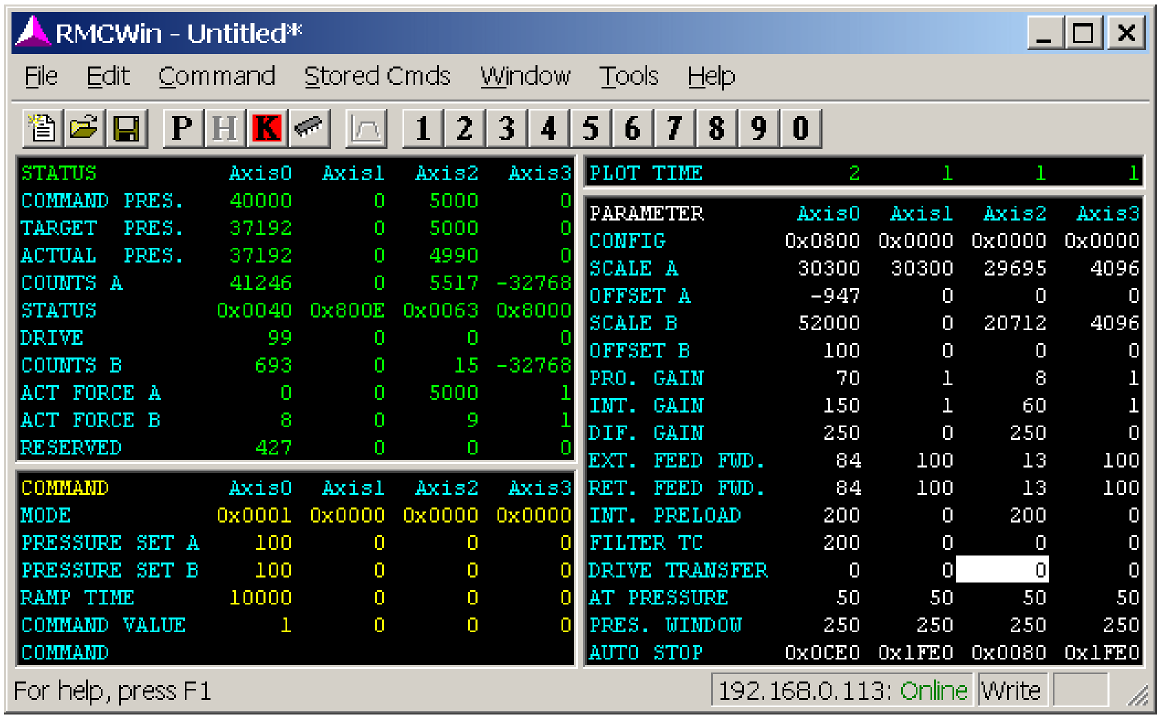


- o Selecting "Always Active" did help a little, but not much. We will try changing the Integral Gain from 15 to 60 to see if it helps. Doing this and then running Event Step 1 results in the following plot:



- The results are much better. Notice how the pressure leads a little, but it eventually tapers off and the pressure is right on when it reaches the commanded pressure.
- Increasing the Integral gain makes a system more prone to oscillation. It may be necessary to reduce it a little.

- The system is now fairly well tuned. The RMCWin window now looks like this:



STATUS	Axis0	Axis1	Axis2	Axis3
COMMAND PRES.	40000	0	5000	0
TARGET PRES.	37192	0	5000	0
ACTUAL PRES.	37192	0	4990	0
COUNTS A	41246	0	5517	-32768
STATUS	0x0040	0x800E	0x0063	0x8000
DRIVE	99	0	0	0
COUNTS B	693	0	15	-32768
ACT FORCE A	0	0	5000	1
ACT FORCE B	8	0	9	1
RESERVED	427	0	0	0

COMMAND	Axis0	Axis1	Axis2	Axis3
MODE	0x0001	0x0000	0x0000	0x0000
PRESSURE SET A	100	0	0	0
PRESSURE SET B	100	0	0	0
RAMP TIME	10000	0	0	0
COMMAND VALUE	1	0	0	0
COMMAND				

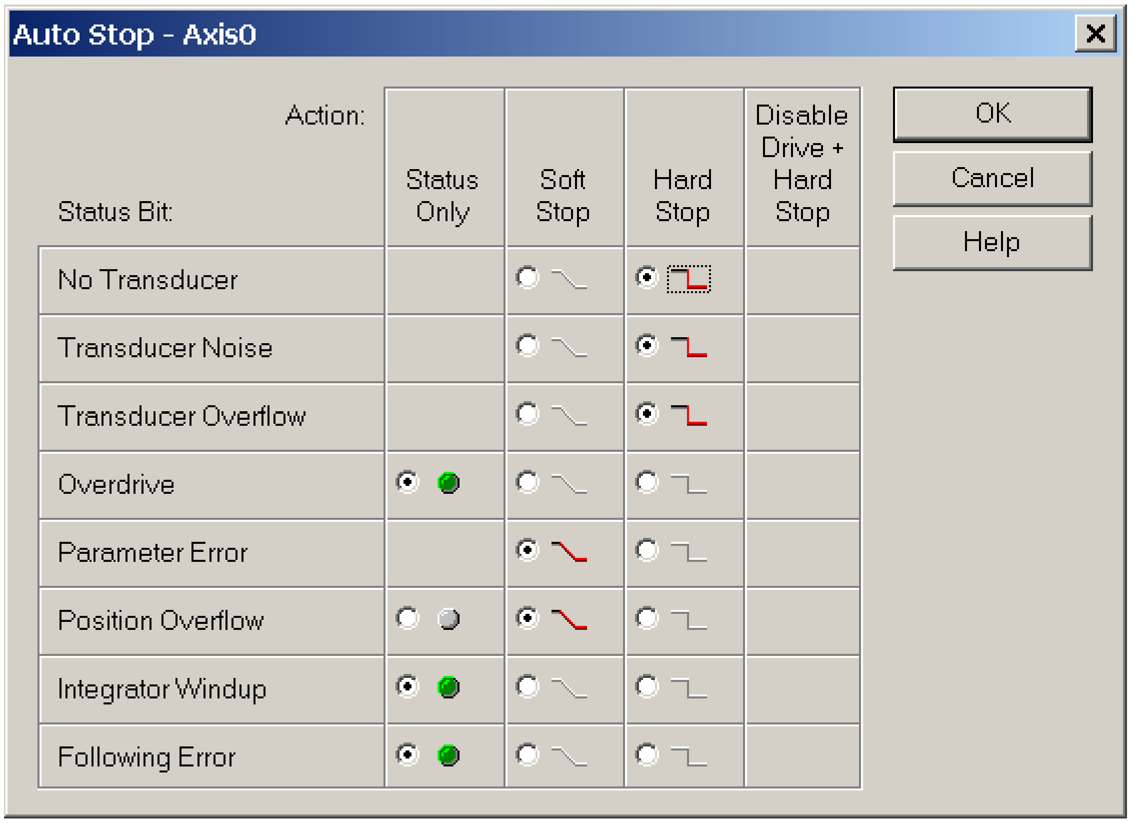
PLOT TIME	2	1	1	1

PARAMETER	Axis0	Axis1	Axis2	Axis3
CONFIG	0x0800	0x0000	0x0000	0x0000
SCALE A	30300	30300	29695	4096
OFFSET A	-947	0	0	0
SCALE B	52000	0	20712	4096
OFFSET B	100	0	0	0
PRO. GAIN	70	1	8	1
INT. GAIN	150	1	60	1
DIF. GAIN	250	0	250	0
EXT. FEED FWD.	84	100	13	100
RET. FEED FWD.	84	100	13	100
INT. PRELOAD	200	0	200	0
FILTER TC	200	0	0	0
DRIVE TRANSFER	0	0	0	0
AT PRESSURE	50	50	50	50
PRES. WINDOW	250	250	250	250
AUTO STOP	0x0CE0	0x1FE0	0x0080	0x1FE0

For help, press F1 | 192.168.0.113: Online | Write

- Now that the system is tuned, it is important to enable any Auto Stops that were set to Status

Only. Double-click the Axis 0 Auto Stop field. The following window appears:



Action: / Status Bit:	Status Only	Soft Stop	Hard Stop	Disable Drive + Hard Stop
No Transducer		○	◉	
Transducer Noise		○	◉	
Transducer Overflow		○	◉	
Overdrive	◉	○	○	
Parameter Error		◉	○	
Position Overflow	○	◉	○	
Integrator Windup	◉	○	○	
Following Error	◉	○	○	

- On most systems, it is preferable to have any axis errors cause a Soft or Hard Stop for safety. Additionally, Soft Stops are often desirable because they slowly stop the axis. A Hard Stop immediately puts the drive output to 0 volts, which in some cases can cause a sudden (and potentially dangerous) jerk in the system. Carefully consider your system requirements before determining how to set the Auto Stop bits. In this case, all the Auto Stops are set to Soft Stop. Clicking OK exits the window and issuing a P command initializes the axis with these parameters.

- The In Position, At Pressure and Following Error parameters should also be set properly.

  - The In Position parameter specifies the size of the window in which the axis is considered to be in position. Once the axis is in this window, the In Position bit is turned on. This is useful for determining whether the axis has reached the commanded position. In this example, the axis is considered to be in position when it is within 0.050 inches, so the In Position parameter is set to 50.

  - The At Pressure parameter is similar to the In Position parameter, but is for the differential pressure axis. In this example, it is set to 50, which is 50 lbs.

  - The Following Error parameter determines how large the difference between the Target Position and Actual Position can get before the Following Error bit is set in the Status word. Normally, this parameter is set to greater than the worst-case following error to avoid unnecessarily causing an error. In this example, the Following Error parameter is set to 250 (0.25 in.).

- The system is now ready to be used for more complicated commands, such as speed control, gearing, splines, synchronized moves, etc.

5 Communications

5.1 Digital I/O

5.1.1 Digital I/O Specifications

The following specifications apply to the digital inputs and outputs on the CPU, Communication Digital I/O and Sensor Digital I/O modules:

Inputs (CPU 0-1 and DI/O 0-17)

Description	CPU: Differential (Sink or Source) DI/O: Sinking (Sourcing driver)
Logic Polarity	CPU: True High DI/O: Configurable (True High default)
Isolation	2500 VAC RMS optically isolated
Signal compatibility	6 mA max at 5 VDC, 10 mA max at 24 VDC
Threshold voltage	2.75 VDC typical, 3 VDC maximum
Threshold current	2.7 mA typical, 3.2 mA maximum
Max input voltage	26.4 VDC
Filtering	Digital I/O Inputs 0-15: 500 µs Digital I/O Inputs 16-17: 250 µs CPU Inputs 0-1: none
8-bit edge counter	Input 17
Quadrature counter	Inputs 16 (A input) and 17 (B input)

Standard Outputs (DI/O module 0-7, CPU module 0-1)

Description	Solid State Relay

	CPU: Independent
	DI/O: Common high or low side
Logic polarity	CPU: True High
	DI/O: Configurable (True high default)
Isolation	2500 VAC RMS
Maximum voltage	±30 V (DC or peak AC)
Maximum current	±100 mA
Max. propagation delay	1.5 ms
Logic 1	Low impedance (50 W maximum)
Logic 0	High impedance (<1 mA leakage current at 250 V)

5.1.2 Digital I/O Wiring

Digital Outputs

The outputs from the Digital I/O's are solid state relays (SSRs). When they are ”off” they have high impedance, and when ”on” they have low impedance (50 W maximum, 25 W typical). Because the outputs are isolated, the user must power them externally. The maximum current and voltage for the outputs is 100 mA and 30 V.

5.1.2.1 CPU Digital Outputs

Each CPU digital output has a + and - connection. Both lines must be connected for the output to function. Because both sides of the output are provided, the switches may be independently connected in a high side or low side configuration (that is, with the load (input) on the source or sinking side of the output). See the wiring diagrams below.

5.1.2.2 DI/O Digital Outputs

There are nine pins on the ”OUTPUTS” section of the Digital I/O. The bottom pin is marked ”Output Cmn” and is common to one side of all the output relays. The other side of the eight SSRs are numbered 0-7.

The switches can be wired in a high-side or low-side configuration. A high-side configuration ties ”Output Cmn” to the output power source; the SSRs control power to the load. A low-side configuration ties ”Output Cmn” to GND.

Digital Output Wiring Diagrams

External Fuses should be used to protect the SSRs if there is a possibility of over-current. When switching inductive loads, it is important to place a diode or tranzorb across the load to protect the switch when transitioning from an ”on” to an ”off” state. Otherwise, the collapsing magnetic field can cause a reverse voltage spike in excess of the 30 V rating of the SSR. See figures below for detail.

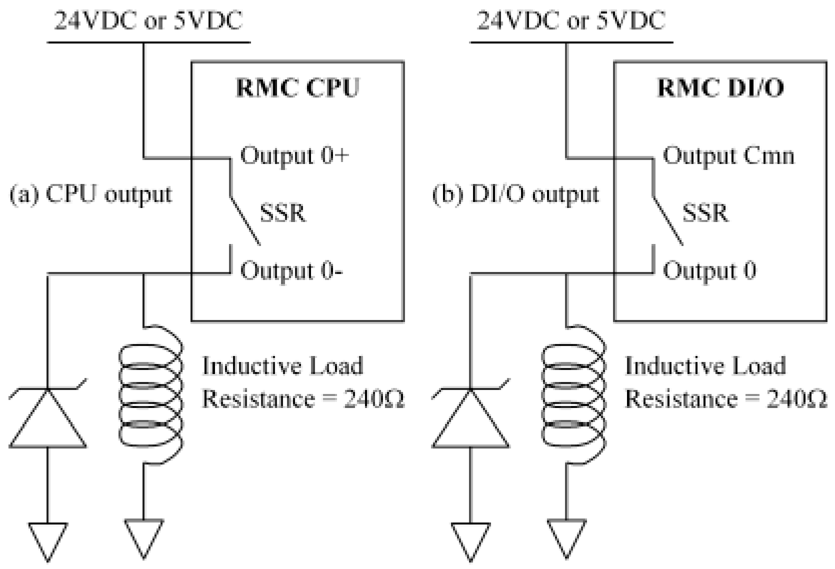


Figure #1: SSR switching inductive load; high-side configuration

To calculate the maximum current through the SSR in the above diagram, we assume zero SSR resistance:

Max. current = 24V / 240W = 100mA

Max. current = 5V / 240W = 20.8mA

In the 24V case, the maximum current is right at the maximum allowed by the SSRs. The outputs may be overpowered if the coil resistance is reduced further. To calculate the typical current through the SSR, we use the typical SSR resistance of 25W:

Typical current = 24V / (240W + 25W) = 90.6mA

Typical current = 5V / (240W + 25W) = 18.9mA

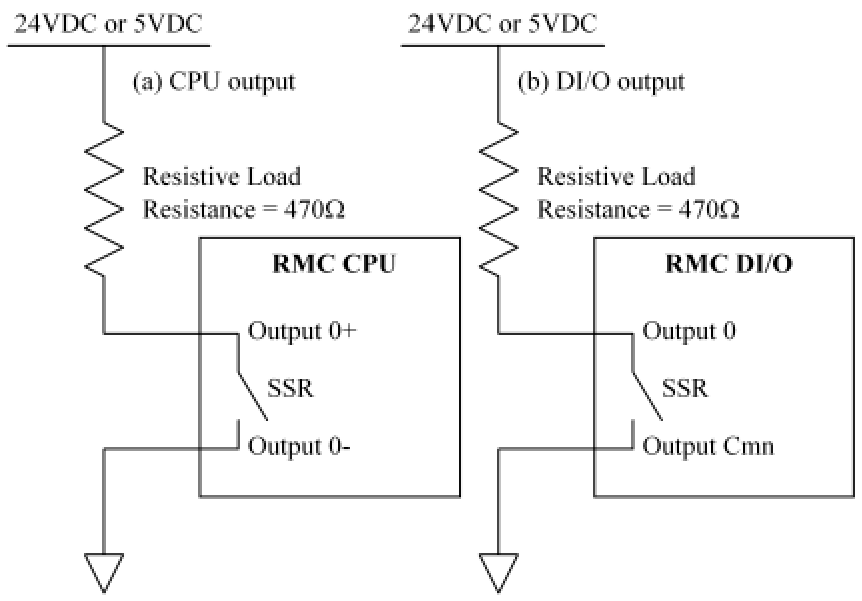


Figure #2: SSR switching resistive load; low-side configuration

To calculate the maximum current through the SSR in the above diagram, we assume zero SSR resistance:

Max. current = 24V / 470W = 51.1mA

Max. current = 5V / 470W = 10.6mA

Notice that both maximum current are well within the ratings. To calculate the typical current through the SSR, we use the typical SSR resistance of 25W:

Typical current = 24V / (470W + 25W) = 48.5mA

Typical current = 5V / (470W + 25W) = 10.1mA

Digital Inputs

The Digital Inputs are compatible with signal levels ranging from 5V to 24V. The Digital inputs draw 6mA maximum with a 5V input and 10mA with a 24V input.

5.1.2.3 CPU Inputs

Each of the CPU's two inputs have both a + and - connection. This allows the input to be connected as either a sinking or sourcing input, which means that most P/C outputs can be connected directly to the CPU inputs. See the wiring diagrams below.

5.1.2.4 DI/O Inputs

The DI/O inputs consist of a single common and eighteen individual input connections.

Note: The numbered inputs must be positive relative to the "Input Cmn" pin common. That is, these are sinking inputs.

Most P/C outputs can be connected to the DI/O inputs directly. The exception is open collector (sinking) outputs. See the discussion below for using sinking outputs.

Note: Because the inputs are designed for use with 5V outputs, the threshold is 2.75VDC. This is a small percentage of the 24V output. As a result, it is important that the inputs have very little noise. If noise is a problem, we recommend that the wiring be rearranged so the noise in the wires is reduced. If this is not feasible, try using the software input filters, or you may want to consider using a voltage divider on the inputs. This is described below.

Digital Input Wiring Diagrams

The following are some input wiring diagrams:

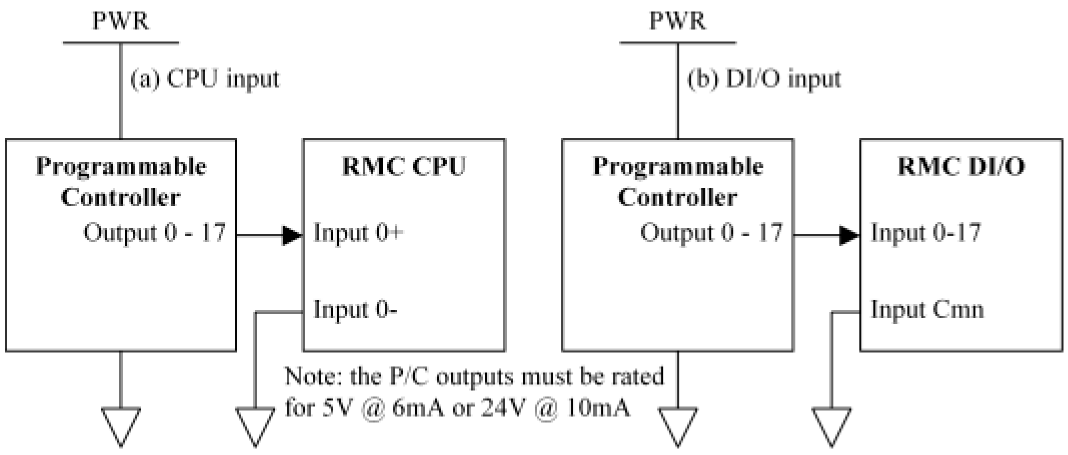


Figure #3: Direct connection to Programmable Controller

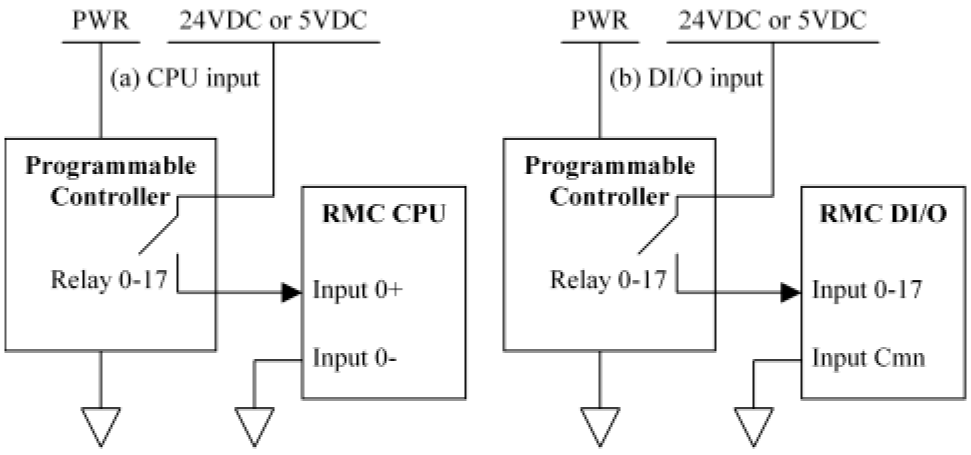


Figure #4: Relay Connection from Programmable Controller

The RMC's CPU inputs can also be connected directly to open collector (sinking) outputs. DI/O inputs cannot be connected directly, as described below.

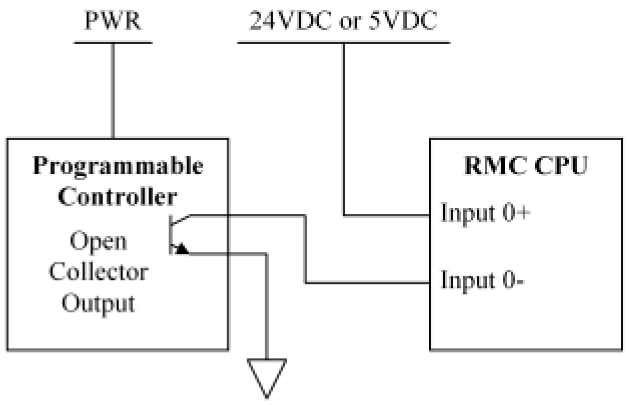


Figure #5: Open Collector Outputs to RMC CPU Inputs

The RMC's DI/O inputs are not optimized for use with open collector (sinking) outputs. The difficulty arises from the fact that sinking outputs have a common ground, but the DI/O also needs a common ground. Therefore, pull-up resisters must be used, as shown in the following diagram (resister values are described below):

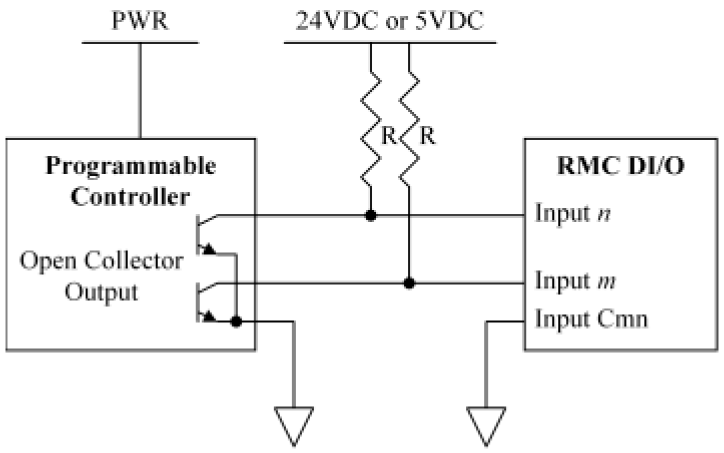


Figure #6: Open Collector Outputs to RMC DI/O Inputs

For 24VDC power, R should be a 3.3 kOhm, ½ watt resister. The output must be capable of switching 7.5mA when closed. When open collector output is open, the DI/O input sees 7V @ 5.1mA.

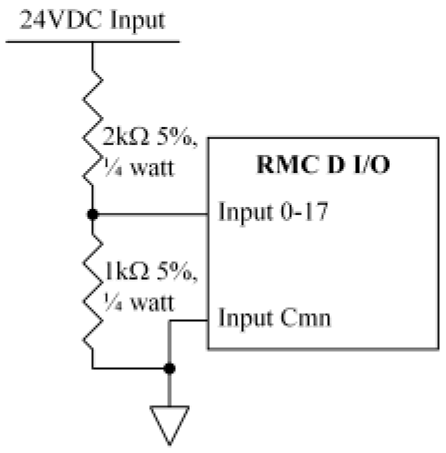
For 5VDC power, R should be a 560 Ohm, 1/8 watt resister. The output must be capable of switching 9.0mA when closed. When open collector output is open, the DI/O input sees 3.1V @ 3.4mA.

Dividing the Input Voltage

Because the inputs are designed for use with 5V outputs, the threshold is 2.75VDC. This is a small percentage of the 24V outputs. As a result, it is important that the inputs have very little noise. This section describes using a voltage divider to raise the apparent input threshold.

Note: Before resorting to dividing the input voltage, take all possible measures to eliminate noise from the inputs. Also, you may want to employ the software input filters.

To divide the inputs, attach resisters to each input as shown in the following diagram:



This configuration will reduce noise susceptability by a factor of about five.

See also:

General Wiring Information

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

5.1.3 Using Counters

Both the Communication Digital I/O and Sensor Digital I/O modules are equipped with quadrature and edge counters. Only one counter on the entire motion control module can be enabled at a time. That is, if an edge counter on the Communication Digital I/O is enabled, then no counters on the other module can be enabled.

Edge Counters Explained

An edge counter counts the rising and falling edges of an incoming square wave. Therefore, if the input wave has a frequency of 1000Hz (has a rising edge 1000 times a second), then an edge counter will register 2000 counts per second (because it counts both rising and falling edges). An edge counter cannot determine direction as a quadrature counter can.

On this motion controller, the input for an edge counter is input 17 on both digital I/O modules.

Note: Because of the 250µs filter on input 17, the maximum input frequency is 2000Hz. This translates to 4000 counts per second using the edge counter.

Quadrature Counters Explained

A quadrature counter counts transitions on two square waves. The inputs, labeled A and B, are 90 degrees out of phase. If the input A phase is leading the B phase, then this counter increases, and if the B phase leads the A phase, then the counter decreases. This type of counter is often used on belts to determine both the speed and direction of a belt. If the A and B signals have a frequency of 1000Hz (they each have 1000 rising edges a second), then the quadrature counter will register 4000 counts per second (because it counts both rising and falling edges on both signals).

On the RMC, inputs A and B are inputs 16 and 17 respectively on both digital I/O module.

Note: Because of the 250µs filter on inputs 16 and 17, the maximum input frequency is 2000Hz. This translates to 8000 counts per second using the quadrature counter.

Using Counters

There are currently two primary uses for counters on the motion controller:

1. They can be used to delay based on an input when using the Event Step table using the DelayTicks (d) link type.
2. They can be used to move through Splines.

Enabling a Counter

Counters are configured as described below. Notice that for the Communication Digital I/O, you must not be using Parallel Position mode. The current counter configuration is stored in the Flash and is only read from the Flash on power-up.

To enable or disable a counter:

1. On the **Tools** menu, click **Module Configuration**.
2. In the **Slots** list, click the DI/O item you wish to edit. This may be either a Communication or Sensor DI/O.
3. Click **Slot options**.
4. Under **Counter type**, click the appropriate option. Notice that this area will be unavailable if the Communication DI/O is in Parallel Position mode as counters are not supported in that case.
5. Click **Update RMC**.
6. The **Update Module Configuration** dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module manually.

5.1.4 CPU Digital I/O

5.1.4.1 Using the CPU Digital I/O

The CPU is equipped with two digital inputs and two digital outputs. The use of these inputs and outputs depends on the type of communication module located to the left of the CPU. If the communication module is a Communication Digital I/O, then these inputs are reserved as described in the section on the Communication Digital I/O.

Otherwise, they can be used in the following ways.

- The inputs can be used by the Input to Event table.
- The inputs can be used to control transitions between Steps in the Event Step table through the Single Input (I, i, O, and o) and Multiple Inputs link types.
- The outputs can be set using the Set Outputs and Reset Outputs commands.

For details on additional discrete I/O options, see Using the Communication Digital I/O and Using the Sensor Digital I/O.

It is highly recommended that the I/O Bit Monitor be used for debugging wiring of these discrete inputs.

5.1.5 Sensor Digital I/O

5.1.5.1 Using the Sensor Digital I/O

The Digital I/O module provides a simple communications interface between the motion controller and the system controller or Programmable Controller (P/C). RMC modules are available with two types of Digital I/O. When a digital I/O is installed to the left of the CPU, it is called the Communication Digital I/O. When a digital I/O is installed to the right of the CPU, it is called the Sensor Digital I/O. Both share the same specifications; refer to Digital I/O Specifications for this information. However, the Sensor and Communication Digital I/O's have different features. This topic discusses the features of the Sensor Digital I/O.

The Sensor Digital I/O has eighteen digital inputs and eight digital outputs. The inputs on the Sensor Digital I/O can be used for the following:

- The inputs can be used to trigger event sequences using the Input to Event table. That is, rising edges and falling edges on these inputs can be set up to start any event sequence on one or more axes. Refer to Input to Event table for details. This ability is available unless the Communication DI/O is used in Input to Event mode.
- The inputs can be used to control transitions between Steps in the Event Step table through the Single Input (I, i, O, and o) and Multiple Inputs link types. In this case, the transition is allowed or disallowed by the link type, but an event sequence must already have been started before the link types will be evaluated. This differs from using the Input to Event table, which actually begins an event sequence.
- Inputs 16 and 17 can be used as either an edge or quadrature counter. See Using Counters for details.
- When used in combination with a Communication DI/O using Parallel Event mode, the inputs can be used as part of that mode. Refer to Parallel Event mode for details.

The outputs of the Sensor DI/O can be used for the following:

- The outputs can be set using the Set Outputs and Reset Outputs commands.

The use of this I/O is configured using the Sensor Digital I/O dialog box. To open this dialog box:

1. On the **Tools** menu, click **Module Configuration**.
2. In the **Slots** list, click the **Sensor Digital I/O** item. This item will be available only if the Sensor Digital I/O module is installed.
3. Click **Slot options**.

The Sensor Digital I/O dialog box has the following areas:

- **Invert Inputs check boxes**
  To invert inputs, select the check boxes of the inputs you want inverted. When digital inputs are displayed with the I/O Bit Monitor they are displayed after they have been inverted.
- **Invert Outputs check boxes**
  To invert outputs, select the check boxes next to the outputs you want inverted. When digital outputs are displayed with the I/O Bit Monitor they are displayed before they have been inverted.
- **Counter Type Selection**
  The Sensor Digital I/O is equipped with quadrature and edge counters. Only one of these two counters may be used at a time. Similarly, only one of the counters on the Sensor or Communication Digital I/O modules can be enabled at a time. Refer to Using Counters for details.

  This section will be unavailable if the Communication DI/O is configured in Parallel Event mode because this mode uses all inputs on both the Communication DI/O and Sensor DI/O.

This dialog box has two available commands:

- **Update RMC**
  This command sends the Sensor Digital I/O options to the RMC, issues an Update Flash command, and resets the RMC to make all changes take effect.
- **Cancel**
  This command closes the dialog box and discards changes.

5.1.6 Communication Digital I/O

5.1.6.1 Using the Communication Digital I/O

The Digital I/O module provides a simple communications interface between the motion controller and the system controller or Programmable Controller (P/C). RMC modules are available with two types of Digital I/O. When a digital I/O is installed to the left of the CPU, it is called the Communication Digital I/O. When a digital I/O is installed to the right of the CPU, it is called the Sensor Digital I/O. Both share the same specifications, refer to Digital I/O Specifications for this information. However, the Sensor and Communication Digital I/O’s have different features. This topic discusses the features of the Communication Digital I/O.

The Communication Digital I/O has eighteen digital inputs and eight digital outputs. The use of these digital inputs is configured using the **Communication DI/O Options** dialog box. To open this dialog box:

1. On the **Tools** menu, click **Module Configuration**.
2. In the **Slots** list, click the **Comm Digital I/O** item. This item will only be available if a Communication Digital I/O module is installed.
3. Click **Slot options**.

The **Communication DI/O Options** dialog box has the following areas:

- **Invert Inputs**

  To invert inputs, check the boxes next to the inputs you want inverted. When digital inputs are displayed with the I/O Bit Monitor they are displayed after they have been inverted.

- **Invert Outputs**

  To invert outputs, check the boxes next to the outputs you want inverted. When digital outputs are displayed with the I/O Bit Monitor they are displayed before they have been inverted.

- **Counter Type**

  The Communication Digital I/O is equipped with quadrature and edge counters. Only one of these two counters may be used at a time. Similarly, only one of the counters on the Sensor DI/O module can be enabled at a time. Refer to Using Counters for details.

Note: In Parallel Position and Parallel Event modes, the inputs used by the counters are reserved for other uses. Therefore, counters cannot be used in these modes.

- **DI/O Mode**

  The Communication Digital I/O can operate in several modes. To configure the mode of your choice, do the following:

  1. In the **DI/O Mode** list, click the appropriate mode. Use the drop-down arrow.
  2. Click **Mode options** to display the options window corresponding to the selected mode.
  3. After making changes to the mode options window, click **OK** in the Mode Options dialog box.
  4. Click **Update RMC**.
  5. The **Update Module Configuration** dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module manually.

  The available modes are described in the following sections:

  - Parallel Position Mode
  - Input to Event Mode
  - Command Mode
  - Parallel Event Mode

- **Error Indication**

Note This section is only used by RMC CPU firmware dating 20011112 or later.

By default, the **Ready** discrete output (CPU output 0) matches the state of the **Run/Stop** input (CPU input 0). However, with 20011112 or later firmware, this output can also be used to indicate that an error has occurred on a group of axes. To select the axes you want to monitor for errors, check the boxes next to each axis you want to monitor.

If an error bit turns on in the Status Word for any of these selected axes, and that error bit has been set up to trigger a Soft or Hard Stop by the Auto Stop word, then the **Ready** output will turn off even if the Run/Stop output is on. Therefore, the **Ready** output will be on if the **Run/Stop** output is on and there are no Auto Stop errors on any of the selected axes.

This dialog box has two available commands:

- **Update RMC**

  This command sends the Communication Digital I/O options to the RMC, issues an Update Flash command, and resets the RMC to make all changes take effect.

- **Cancel**

  This command closes the options dialog and discards any changes made.

5.1.6.2 Features Shared by All Modes

There are several details that are common between the Digital I/O modes (Parallel Position, Input to Event, Command, and Parallel Event). These are described here.

Inverting Inputs and Outputs

It is possible to invert any Communication Digital I/O input or output. Refer to Communication Digital I/O for instructions on inverting the bits.

CPU Input 0 - Run/Stop

Note: This special functionality is valid only if there is a DI/O module in the comm (leftmost) slot.

This input is edge-sensitive. On the rising edge (0 to 1 transition), a Set Parameters command is sent to each axis. On the falling edge (1 to 0 transition), a Disable Drive Output command is sent to each axis. This input should be set low when drive or hydraulic power is turned off.

CPU Output 0 - Ready

Note: This special functionality is valid only if there is a DI/O module in the comm (leftmost) slot.

With RMC CPU firmware dating prior to 20011112, this input echoes the **Run/Stop** input described above. That is, when the axes are initialized because of a rising edge of the **Run/Stop** input, this output will turn on, and it will go off when the **Run/Stop** input goes off.

With RMC CPU firmware dating 20011112 or later, the user can also use this output to indicate that an error occurred on a group of axes. If an error bit turns on in the Status Word for any of the selected axes, and that error bit has been set up to trigger a Soft or Hard Stop by the Auto Stop word, then the **Ready** output will be off even if the **Run/Stop** output is on. Therefore, the **Ready** output will be on if the **Run/Stop** output is on *and* there are no Auto Stop errors on any of the selected axes.

To select the axes that use this error indication, use the **Communication DI/O Options** dialog box, as described in Using the Communication Digital I/O. By default no axes are selected, so the behavior matches that of pre-20011112 firmware.

CPU Output 1 - Command Acknowledge

Note: This special functionality is valid only if there is a DI/O module in the comm (leftmost) slot.

This bit toggles each time a new command is received. The criteria for a new command varies on the mode used; refer to the appropriate mode description for details. This output deglitches inputs and simplifies programming.

Digital Inputs

The digital inputs are reserved by the current mode you are using. Refer to Input to Event Mode, Parallel Position Mode, Command Mode, or Parallel Event Mode for details.

Digital Outputs

The Communication Digital I/O has eight digital outputs. When using Input to Event Mode, Parallel Position Mode, or Parallel Event Mode, these inputs can be used in the following three ways. Refer to Command Mode for details on their use in that mode.

- **In Position**: As many outputs as installed axes are used to indicate when the In Position bit of the Status word is set. These bits start at bit 0. Therefore, output 0 represents the In Position bit of axis 0, output 1 represents the In Position bit of axis 1, etc.
- **Auto Stop Error**: If there are four or fewer axes in the motion controller, then one additional bit per axis is reserved to indicate when an axis has stopped due to an error. This bit is set when an error occurs—as marked in the Status word—which is also set in the Auto Stop word. These bits start at bit 4. Therefore, output 4 represents the Auto Stop Error bit for axis 0, output 5 represents the Auto Stop Error bit for axis 1, etc.
- **User Controlled**: Any bits still unused can be set and reset using the Set Outputs and Reset Outputs commands. Attempts to use these commands on bits reserved for In Position or Auto Stop will be ignored. In Input to Event and Parallel Event modes, bits that default to being used for In Position and Auto Stop Errors may be overridden to become user controlled. To do this, click **Mode options** from the **Communication DI/O Options** dialog box.

The following charts show the bit-use defaults for some RMC configurations:

Output	**RMC100-M1**	**RMC100-M2**
0	In Position (Axis 0)	In Position (Axis 0)
1	In Position (Axis 1)	In Position (Axis 1)
2	User Controlled	In Position (Axis 2)
3	User Controlled	In Position (Axis 3)
4	Auto Stop (Axis 0)	Auto Stop (Axis 0)
5	Auto Stop (Axis 1)	Auto Stop (Axis 1)

6	User Controlled	Auto Stop (Axis 2)
7	User Controlled	Auto Stop (Axis 3)

Output	**RMC100-M3**	**RMC100-M4**
0	In Position (Axis 0)	In Position (Axis 0)
1	In Position (Axis 1)	In Position (Axis 1)
2	In Position (Axis 2)	In Position (Axis 2)
3	In Position (Axis 3)	In Position (Axis 3)
4	In Position (Axis 4)	In Position (Axis 4)
5	In Position (Axis 5)	In Position (Axis 5)
6	User Controlled	In Position (Axis 6)
7	User Controlled	In Position (Axis 7)

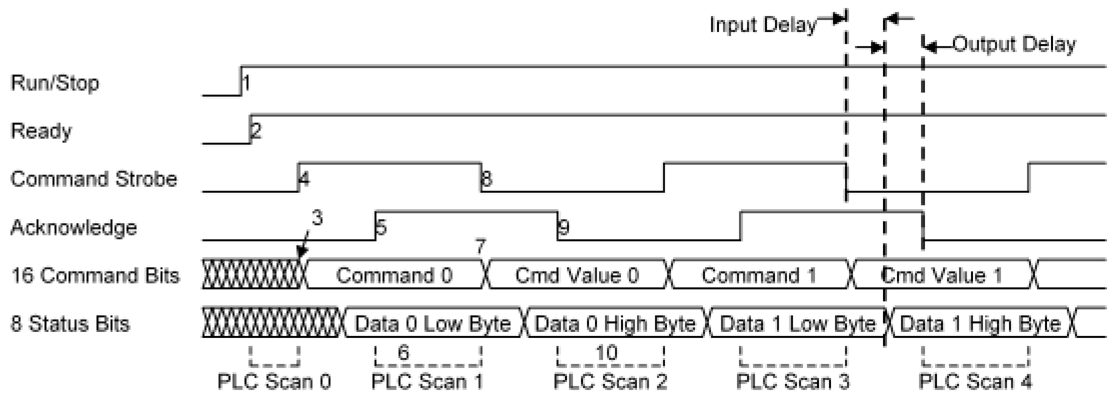
5.1.6.3 Using Command Mode

This mode allows any motion controller command to be sent to the motion controller using the Communication Digital I/O. In addition, it allows the user to request a wide range of status information back, including ACTUAL POSITION. This mode is more complicated than the other modes and requires a PLC to communicate with the RMC, but it is much more capable than the other Communication Digital I/O modes.

Refer to Features Shared by All Modes for details for input and output assignments that are common to all modes.

Basic Operation

In this mode, commands and status information are sent between the motion controller and the programmable controller using the following sequence of steps:



1. Raise the CPU input 0. This input is also called **Run/Stop**. CPU input 1 should start low.
2. Wait for the CPU output 0 to raise. This input is called **Ready**.
3. Place a 16-bit command word on digital inputs 0-15. This word gives the command type and also information for which data is requested back. See Command Words for Command Mode for details on using this word.
4. Raise the **Command Strobe**.
5. Wait for CPU output 1 to toggle. This output will be called **Acknowledge** from here forward.
6. Read digital outputs 0-7, which will hold bits 0-7 of the data just requested.
7. If the command sent in the previous steps requires a command value (such as a requested position), place the 16-bit command value on digital inputs 0-15.
8. Lower the **Command Strobe**.
9. Wait for **Acknowledge** to toggle. At this point the command will be have been executed.
10. Read digital outputs 0-7, which will hold bits 8-15 of the requested data. Although the requested data may be changing constantly, the entire 16 bits is latched when the command is first executed. It is this latched data that is outputted byte by byte.

    These steps should be processed by the PLC so that a complete new command can be sent every two PLC scans. To achieve this, the following steps should be taken:

1st scan:
Perform step 1.

2nd scan:
Perform steps 2 through 4 to send first half of first command.

3rd scan:
Perform steps 5 through 8 to send the second half of first command.

4th scan:
Perform steps 9 and 10 to finish receiving the data requested by the first command.

Perform steps 3 and 4 to send first half of second command.

5th scan:

Perform steps 5 through 8 to send the second half of second command.

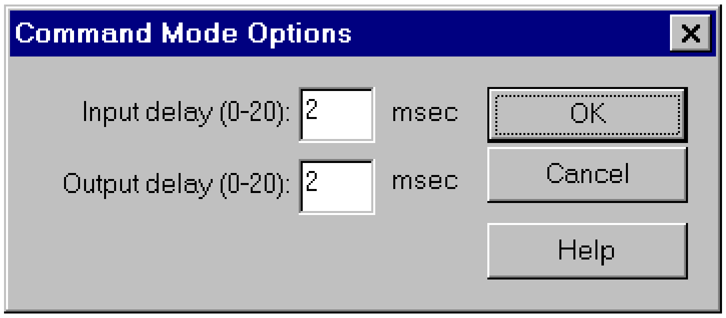
6th scan:
Perform steps 9 and 10 to finish receiving the data requested by the second command.

Perform steps 3 and 4 to send first half of third command.

… (Repeat the last two scans for each new command)

Configuring Command Mode

This is done using the **Command Mode Options** dialog box:



Command Mode Options

Input delay (0-20): 2 msec

Output delay (0-20): 2 msec

To use this dialog box:

1. On the **Tools** menu, click **Module Configuration**.
2. In the **Slots** list, click the **Comm Digital I/O** item.
3. Click **Slot options**.
4. In the **DI/O mode** list, click **Command**.
5. Click **Mode options**.
6. Select the desired options.
7. Click **OK**.
8. Click **Update RMC**.
9. The **Update Module Configuration** dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module manually.

Input Delay

Digital inputs 0-15 are not stored in the RMC module until the **Command Strobe** input (CPU in 1) transitions from low to high or from high to low. However, it is possible that the inputs are not stable for some time after that. Therefore, the Input Delay value can be set to make the motion controller wait a number of milliseconds after the **Command Strobe** input toggles before reading inputs 0-15. This is especially important if relays are used to drive these inputs.

Output Delay

Digital outputs 0-7 are set by the motion controller before the **Acknowledge** output (CPU out 1) is toggled. However, it takes a small amount of time for these outputs to settle, and additionally, the inputs reading these lines may take more time to register the outputs. Therefore, the Output Delay value can be set to make the motion controller wait a number of milliseconds after setting the digital outputs before toggling the **Acknowledge** output.

5.1.6.4 Using Input to Event Mode

This mode should be used when the positions to which an axis will be moved are known and can be pre-programmed. It takes advantage of the Input to Event and Event Step tables. Refer to Features Shared by All Modes for details on input and output assignments that are common to all modes.

Basic Operation

When used in this mode, each digital input on the Communication Digital I/O corresponds to a row of the Input to Event table. The row has an entry for each axis. Each entry is used to specify a step the axis will execute in the step when the input is activated. If an axis is not to respond to the input then its entry should be -1. Refer to that topic for further details. One input can trigger as many as eight axes to start executing different sequences of steps at the same time. Each axis can be thought of having its own sequence of steps that may be run independently of other axes' step sequences. Modifications to the normal Input to Event table behavior are discussed below in the **Single-axis inputs** and **Non-linkable inputs** sections.

Each time a transition is made on DI/O 0-15, the **Acknowledge** bit (CPU output 1) will be toggled. This bit will also toggle when the **Run/Stop** bit (CPU input 0) rises or falls.

Configuring Input to Event Mode

This is done using the **Input to Event Mode Options** dialog box. To use this dialog box:

1. On the Tools menu, click **Module Configuration**.
2. In the **Slots** list, click the **Comm Digital I/O** item.
3. Click **Slot options**.
4. In the **DI/O mode** list, click **Input to Event**.
5. Click **Mode** options.
6. Select the desired options.
7. Click **OK**.
8. Click **Update RMC**.
9. The **Update Module Configuration** dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module manually.

Single-axis Inputs

As described in the Input to Event table topic, a single input transition normally affects all axes that have an entry for that input transition. However, you may also configure each individual input to affect only a single axis. To make an input affect only a single axis, select the **Single-axis input** check box. For information on selecting an axis, see the section below titled Axis Select Bits.

An example application where this is useful would be where the same operation can be done on one of several axes using the same button and input. In this application, one might allow the user to set the Axis select bits using a rotary or toggle switch, and then make the input coming from the button a **Single-axis input**. Then, in the Input to Event table, the input transition row would have Step numbers for each axis, but only one would be executed depending on the **Axis select bits** at the time of the transition.

Non-linkable Inputs

This area operates similarly to the **Single-axis inputs** area described above. By selecting the check box next to an input, you are modifying the behavior of the Input to Event table for both transitions of that input. In this case, **Non-linkable inputs** will result in the command in the Event Step referenced by the Input to Event table to be executed on the axis, but the Event Step sequence will not be started, nor will any current Event Step sequence on that axis be interrupted.

User-defined Outputs

Note: This feature is available only in firmware version 19980414 and later.

As described in Features Shared by All Modes, many of the digital outputs are pre-defined. However, under Input-to-Event mode, it is often useful to reserve one or more of these outputs to be triggered explicitly by the event step table (for example, when a sequence of events finish, an output can be set high). Check the boxes of the output numbers you wish to reserve for this purpose. Depending on the number of axes your RMC model offers, one or more of the outputs may be unavailable but checked. These outputs are not predefined and therefore are always user-defined.

To Set the Input Filter

Note: This feature is available only in firmware version 19980716 and later.

Use this text box to set the number of milliseconds that inputs 0-15 of the Digital I/O must be settled before any events will be triggered. This can be used to filter out electronic noise, however be sure to take all possible measures to reduce electronic noise in the system before resorting to using this filter.

Axis Select Bits

This read-only text displays the bits that are being used for selecting an axis. These bits are used only when an input is marked as a **Single-axis Input** (see above). The following charts show how to select the different axes:

Axis Select Bits when no Counter is Used:

CPU	**DI/O**	**DI/O**	
Input 1	**Input 17**	**Input 16**	**Axis**
0	0	0	0
0	0	1	1
0	1	0	2*
0	1	1	3*
1	0	0	4*
1	0	1	5*
1	1	0	6*

1	1	1	7*

Axis Select Bits when Edge Counter is Used:

CPU Input 1	**DI/O Input 16**	**Axis**
0	0	0
0	1	1
1	0	2*
1	1	3*

Axis Select Bits when Quadrature Counter is Used:

CPU Input 1	**Axis**
0	0
1	1

* When an axis is selected that is not present on the motion module, the single-axis input is ignored.

5.1.6.5 Using Parallel Position Mode

In this mode, the user can give simple Go and Open Loop commands using discrete inputs. Refer to Features Shared by All Modes for details on input and output assignments that are common to all modes.

Basic Operation

Before any moves may be made, you must do the following to store the proper parameters. It only needs to be done once because the parameters and profiles will be read from the Flash on power up:

1. Tune the axes by adjusting the parameters.
2. Set up the Profile table to hold all the motion profiles you plan to use (including MODE, ACCEL, DECEL and SPEED). Refer to the charts above for determining which profiles you will use.
3. Save the parameters and profiles to Flash using the Update Flash command. This command can be sent from RMCWin.

You must do the following on each power up to initialize all the axes:

1. Energize the **Run/Stop** bit (CPU input 0).

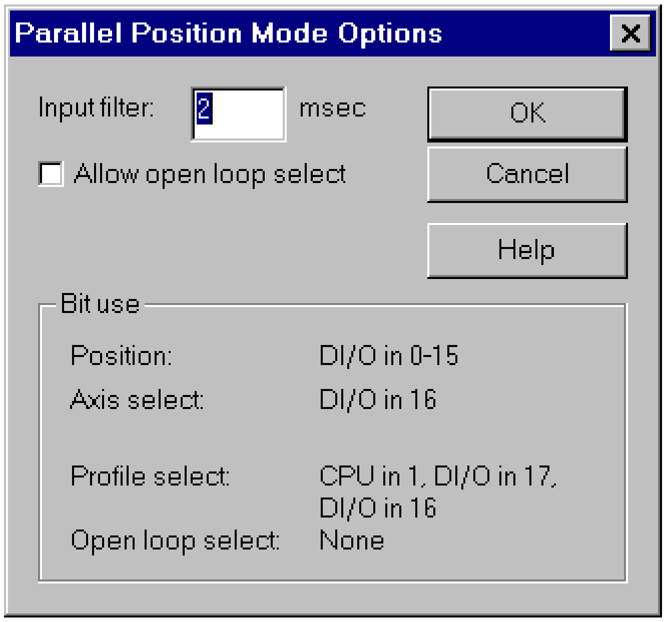
2. Wait for the **Ready** bit (CPU output 0) to go to a logical 1.

   You must continue to hold the **Run/Stop** bit high. If a move is in progress when you make the **Run/Stop** go low, the move will stop immediately. If a move is requested when the **Run/Stop** is low, the request will be ignored.

   To request a move, you must select the axis and profile using Digital inputs 16-17 and CPU input 1. You must also set the position or drive on Digital Inputs 0-15. These must all be done simultaneously. The new value is not processed until all axis, profile and position/drive select bits have been held at the same values for the duration set by the Input Filter field described below. At this time if any of the values are different from the last command, then a new command is issued.

   If the CPU input 1 has been reserved to select Open Loop mode and this bit is set, then when a new command value is given, an Open Loop command will be issued with the command value read from Digital I/O inputs 0-15. Otherwise, a Go command will be issued, also using the command value read from Digital I/O inputs 0-15.

Configuring Parallel Position Mode
This is done using the **Parallel Position Mode Options** dialog box:


Parallel Position Mode Options

Input filter: 2 msec

☐ Allow open loop select

Bit use

Position:	DI/O in 0-15
Axis select:	DI/O in 16
Profile select:	CPU in 1, DI/O in 17, DI/O in 16
Open loop select:	None

To use this dialog box:

1. On the **Tools** menu, click **Module Configuration**.
2. In the **Slots** list, click the **Comm Digital I/O** item.
3. Click **Slot** options.
4. In the **DI/O mode** list, click **Parallel Position**.
5. Click **Mode options**.

6. Select the desired options.
7. Click **OK**.
8. Click **Update RMC**.
9. The **Update Module Configuration** dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module manually.

   Each option is described below:

To Set the Input Filter

When a new position is written to input bits 0 through 15, all sixteen position/drive bits plus the axis and profile select bits must be updated simultaneously. Because this will never happen exactly simultaneously, a filter is necessary. Use this field to set the number of milliseconds that all eighteen inputs of the Digital I/O, plus CPU input 1 must hold their values before the new values are considered a new command.

Allow Open Loop Select

If four or fewer axes are installed on the motion controller, then the user can use a bit to select whether an axis moves in open or closed loop mode. If this check box is selected, then CPU input 1 will be reserved for this purpose. If this box is not checked, then all moves will be closed loop and this bit will be used to select the profile and/or axis. With more than four axes, this option is not available because the CPU input 1 bit must be used to select the upper axes.

If this feature is disabled or CPU input 1 is low, the Digital I/O inputs 0 through 15 are used as a 16-bit position input. This is the normal operation.

If this feature is enabled and CPU input 1 is high, the motion controller will act in Open Loop mode. Digital I/O inputs 0 through 15 specify the output value over a -10.000 to +10.000 volt range using signed binary notation. For position and zero values, convert the desired output in millivolts to binary, while for negative values, subtract the desired output in millivolts from 65536 and convert the result to binary. For example, an input of 0010011100010000b will make an output of +10.000 volts, while an input of 1101100011110000b will make an output of -10.000 volts. You can use this mode to jog an axis.

Bit Use Area

This read-only area displays the bits used for the various options. It is originally set based on the number of axes available and is modified as the **Allow Open Loop Select** check box is checked and unchecked. Refer to the chart below for exact bit assignments:

Note: If CPU input 1 has been reserved for open loop select, then profiles 4 through 7 will be in Open Loop mode. Refer to the Open Loop command for details.

Axis and Profile Select Bits for 2-axis RMCs:

CPU	**DI/O**	**DI/O**		
Input 1	**Input 17**	**Input 16**	**Axis**	**Profile**
0	0	0	0	0
0	0	1	1	1

0	1	0	0	2
0	1	1	1	3
1	0	0	0	4
1	0	1	1	5
1	1	0	0	6
1	1	1	1	7

Axis and Profile Select Bits for 4-axis RMCs:

CPU	DI/O	DI/O		
Input 1	**Input 17**	**Input 16**	**Axis**	**Profile**
0	0	0	0	0
0	0	1	1	1
0	1	0	2	2
0	1	1	3	3
1	0	0	0	4
1	0	1	1	5
1	1	0	2	6
1	1	1	3	7

Axis and Profile Select Bits for 6-axis RMCs:

CPU	DI/O	DI/O		
Input 1	**Input 17**	**Input 16**	**Axis**	**Profile**
0	0	0	0	0
0	0	1	1	1
0	1	0	2	2
0	1	1	3	3
1	0	0	4	4
1	0	1	5	5

Axis and Profile Select Bits for 8-axis RMCs:

CPU	DI/O	DI/O		
Input 1	**Input 17**	**Input 16**	**Axis**	**Profile**
0	0	0	0	0
0	0	1	1	1
0	1	0	2	2
0	1	1	3	3
1	0	0	4	4
1	0	1	5	5
1	1	0	6	6
1	1	1	7	7

5.1.6.6 Using Parallel Event Mode

Note: This mode is available only in firmware version 19980706 and later.

This mode is intended for use with devices that can provide parallel outputs, such as PLCs and thumb-wheel switches. Commands may be given to up to four axes at a time, which is useful for PLCs whose scan times are too long to use Parallel Position mode—which can issue only one command per scan—and Command mode—which can issue a command every two scans.

This mode utilizes the RMC’s Event Control feature; you should be familiar with this feature before using this mode. Each command consists of an 8-bit Event Control step number for the receiving axis to begin executing and a **Trigger** input.

When used with thumb-wheels, some or all step-number inputs should be tied to the thumb-wheel, and a push-on/push-off button should be tied to the **Trigger** input.

Refer to Features Shared by All Modes for details on input and output assignments that are common to all modes.

Basic Operation

This mode is unique in that it can utilize both the Communication DI/O and Sensor DI/O. Each of the DI/O modules can command two axes. When a command is given in Parallel Event mode, an event sequence begins on the commanded axis at the Event Step number given on the inputs. The following input/output assignments are used:

CPU DI/O:

Input 0	Run/Stop. Described in Features Shared by All Modes.
Input 1	Unused.
Output 0	Ready. Described in Features Shared by All Modes.
Output 1	Acknowledge. Described in Features Shared by All Modes.

Communication DI/O:

Inputs 0-7	Axis 0 Command Event Step (in binary)
Inputs 8-15	Axis 1 Command Event Step (in binary)
Input 16	Axis 0 Command Trigger
Input 17	Axis 1 Command Trigger
Outputs 0-7	Described in Features Shared by All Modes.

Sensor DI/O:

Inputs 0-7	Axis 2 Command Event Step (in binary)
Inputs 8-15	Axis 3 Command Event Step (in binary)
Input 16	Axis 2 Command Trigger
Input 17	Axis 3 Command Trigger
Outputs 0-7	User-controlled with Set Outputs and Reset Outputs commands.

The CPU inputs, CPU outputs, and DI/O outputs are described in Features Shared by All Modes.

The following three items must be true for a new command to be issued:

- The **Run/Stop** (CPU input 0) line must be high.
- The **Trigger** input for the axis must have changed states (toggled) from the last command issued on the axis.
- The eight event step inputs and the trigger input for the axis must all be stable for the number of milliseconds given by the **Input Filter** value, described below.

Once all of these conditions are satisfied for an axis, a Start Events (E) command is issued to the axis, using the Event Step number representing the axis's DI/O inputs as the first event in the sequence.

All communications occurring on one axis's input lines are independent of the communications taking place on other axes' input lines. Therefore, multiple commands can be issued at once.

Configuring Parallel Event Mode

This is done using the Parallel Event Mode Options dialog box. To use this dialog box:

1. On the **Tools** menu, click **Module Configuration**.
2. In the **Slots** list, click the **Comm Digital I/O** item.
3. Click **Slot options**.
4. In the **DI/O mode** list, click **Parallel Event**.
5. Click **Mode options**.
6. Select the desired options.
7. Click **OK**.
8. Click **Update RMC**.
9. The **Update Module Configuration** dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module manually.

To Set the Input Filter

When a new event number is written to an axis, all eight event-number bits plus the trigger bit for the same axis must be updated simultaneously. This will never happen exactly at the same time, so an input filter must be used. The input filter setting is used to configure the number of milliseconds over which all nine inputs for an axis must hold their values before the new value is considered a command.

User-defined Outputs

As described in Features Shared by All Modes, many of the digital outputs are pre-defined, as described in that topic. However, under Parallel Event mode, it is often useful to reserve one or more of these outputs to be triggered explicitly by the event step table (for example, when a sequence of events finish, an output can be set high). Check the boxes of the output numbers you wish to reserve for this purpose. Depending on the number of axes your RMC model offers, one or more of the outputs may be unavailable but checked. Since these outputs are not predefined the user must define them.

5.1.6.7 Technical Brief: Using the RMC Discrete I/O Command Mode

Abstract

The RMC-DI/O is capable of sophisticated motion control using small and inexpensive Programmable Controllers with simple discrete I/O. An RMC with a DI/O communication interface is capable of four discrete I/O interfaces: Command Mode, Input to Event Mode, Parallel Position Mode, and Parallel Event Mode. Of these four communication modes, Command Mode is the most flexible; it allows changing parameters and retrieving positions, speeds, errors and other status information from the motion controller using simple discrete I/O.

This technical brief will compare the four discrete I/O interfaces of the RMC100 series product-line, describe implementing Command Mode, and finally provide a sample application using Command Mode.

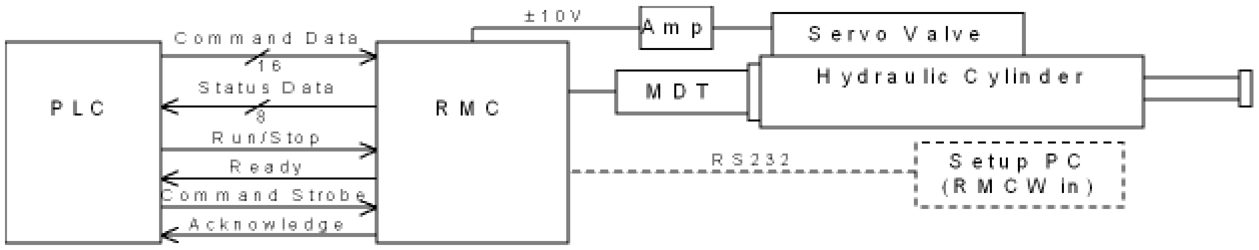
DI/O Communication Mode Comparison

The following chart lists the advantages and disadvantages of each communication mode. Each word or phrase in bold print appears in RMCWin's online help index.

Interface Mode	Advantages	Disadvantages
Command Mode	• Any RMC command can be issued • Any status information can be retrieved (including Actual Position, Actual Speed, Drive, Error Bits, and other Status Bits)	• Requires a PLC • Requires 2 PLC scans per command • Gives commands to only one axis per command cycle
Parallel Event Mode	• Up to four axes may be commanded at once • PLC or thumb-wheel switches may be used • Allows use of Event Control feature	• Requires parallel inputs • Returns only a Halted Bit—indicating an error has occurred—and an In Position Bit per axis • Sequence must be pre-programmed in Event Control
Input to Event Mode	• Does not require a PLC • Multiple axes may be given commands from a single input	• Returns only a Halted Bit—indicating an error has occurred—and an In Position Bit per axis • Issues only Start Event commands
Parallel Position Mode	• Any position can be moved to in a single PLC scan • Any open loop drive can be triggered in a single PLC scan • Multiple profiles can be selected for any move	• Requires a PLC • Returns only a Halted Bit—indicating an error has occurred—and an In Position Bit per axis • Issues only Go and Open Loop commands • Gives commands to only one axis per scan

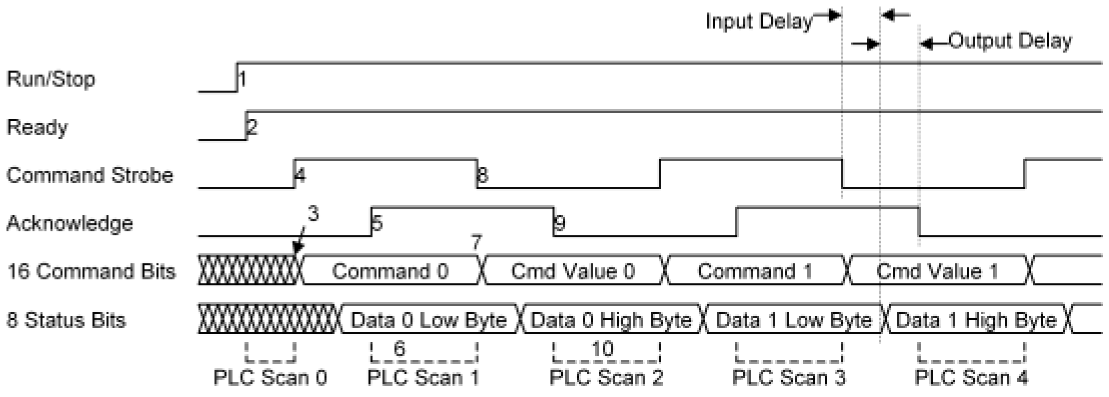
Implementation

The following diagram shows the electrical control connections of a single-axis hydraulic system using RMC-DI/O in Command Mode (the PC and its RS232 cable are needed only during setup):



This document discusses only the connections between the Programmable Controller and the RMC. For details on the transducer and drive wiring, look up Wiring Notes in the RMCWin index.

The RMC accepts commands of up to 32 bits and returns 16-bit data words. However, because it has only half that number of inputs and outputs, an extra input and output on the RMC are used to strobe half of the data at a time. The following timing chart and accompanying list of steps illustrate the sequencing of Command Mode:



This process begins assuming that Run/Stop and Command Strobe are set low by the PLC. In the steps below inputs and outputs are labeled as ”r;CPU” or ”r;DI/O”. These labels refer to the CPU and DI/O modules of the RMC product, and not of the PLC:

1. Raise the Run/Stop (CPU input 0) line on the RMC. This is done on startup of the PLC.
2. Wait for the Ready (CPU output 0) line on the RMC to go high in response to the Run/Stop line. This begins the first PLC scan.
3. Place the 16-bit Command on the RMC’s Command Bits (DI/O inputs 0-15).
4. Raise the Command Strobe (CPU input 1) line on the RMC. This ends the first PLC scan.
5. Wait for the Acknowledge (CPU output 1) line on the RMC to go high. This begins the next PLC scan.
6. Read the low byte of the data requested by the command from the Status Bits (DI/O outputs 0-7) on the RMC.
7. Place the 16-bit Command Value on the RMC’s Command Bits.
8. Lower the Command Strobe on the RMC. Then ends the next PLC scan.
9. Wait for the Acknowledge line on the RMC to go low. This begins the next PLC scan.
10. Read the high byte of the data requested by the command from the Status Bits on the RMC.

The process then repeats from step 3 with the next command.

In the above timing chart, notice two time durations are marked: Input Delay and Output Delay. These intervals can be set using RMCWin to any value between zero and twenty milliseconds.

Input Delay indicates how long the RMC waits after seeing the Command Strobe toggle before reading the command data. Change this setting to account for the PLC outputs’ and the RMC inputs’ settling time.

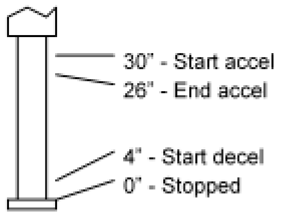
Output Delay indicates how long the RMC waits after writing the requested data before toggling the Acknowledge line. Change this setting to account for the RMC outputs’ and the PLC inputs’ settling time.

To configure Command mode, do the following:

- Connect the serial port to the module being configured.
- On the Tools menu, click Module Configuration.
- In the Slots list, click the Communication DI/O item, and then click Slot options.
- In the DI/O mode list, select Command mode.
- If you need to invert any inputs or outputs to match your hardware, select the appropriate check boxes in the Invert inputs and Invert outputs areas.
- If you wish to use the Communication DI/O counter feature, click either Edge or Quadrature under Counter Type. See Using Counters in the RMCWin online help for more details.
- Click Mode options.
- Enter the Input delay and Output delay parameters as described above.
- Click OK.
- Click Update RMC.
- The Update Module Configuration dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module manually.

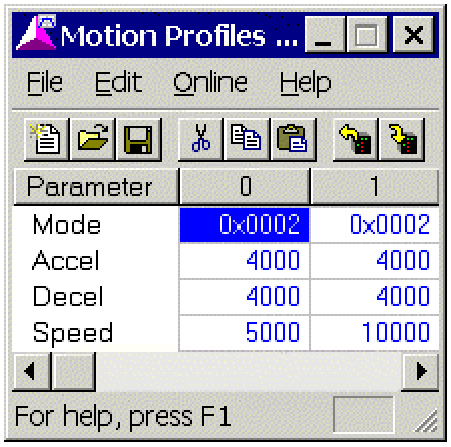
Sample Application

In this application, a single hydraulic cylinder will be moved over a 30 in stroke. It will extend at a rate of 5 in/sec and retract at a rate of 10 in/sec. At each end of the move, there will be a four-inch acceleration/deceleration zone:



The operator display will include the current actual position.

This application can easily be done using an RMC100-M1-DI/O module using Command Mode. The simplest way to implement this application is to begin by loading the profile table with the two motion profiles used:



Parameter	0	1
Mode	0x0002	0x0002
Accel	4000	4000
Decel	4000	4000
Speed	5000	10000

The 'r;2' in the Mode field indicates that the acceleration and deceleration are given as distances. For details on the motion profile table, look up Profiles in the RMCWin online help.

This profile table and the tuning parameters can either be stored in the RMC's Flash memory, or stored in the Programmable Controller and downloaded using the Set Parameter commands. We'll assume they are stored in the Flash. See the Set Profile and Set Parameter topics in the RMCWin online help for details on setting these values from the Programmable Controller.

Reference

Throughout this technical note, references are made to RMCWin online help index entries. To obtain the RMCWin software package, contact Delta Computer System's web site (www.deltacompsys.com).

5.1.6.8 Technical Brief: Using the RMC Discrete I/O Input to Event Mode

Abstract

The RMC-DI/O is capable of sophisticated motion control using small and inexpensive Programmable Controllers with simple discrete I/O. An RMC with a DI/O communications interface is capable of four discrete I/O interfaces: Command Mode, Input to Event Mode, and Parallel Position Mode. Of these four communication modes, Input to Event mode is the simplest for many repetitive-process applications; many applications may not require a programmable controller.

This technical brief will compare the four discrete I/O interfaces of the RMC100 product-line, describe implementing Input to Event Mode, and finally provide a sample application using Input to Event Mode.

DI/O Communication Mode Comparison

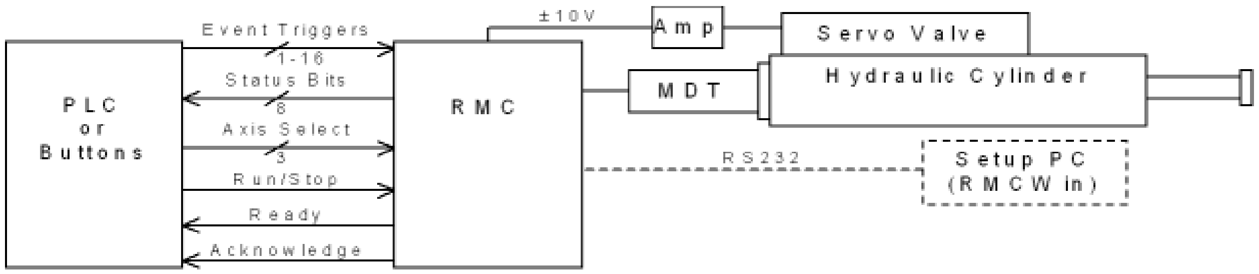
The following chart lists the advantages and disadvantages of each communication mode. Each word or phrase in bold print appears in RMCWin's online help index.

Interface Mode	Advantages	Disadvantages
Input to	• Does not require a PLC	• Returns only a Halted

Event Mode	• Multiple axes may be given commands from a single input • Allows use of Event Control feature	Bit—indicating an error has occurred—and an In Position Bit per axis • Issues only Start Event commands
Parallel Event Mode	• Up to four axes may be commanded at once • PLC or thumb-wheel switches may be used • Allows use of Event Control feature	• Requires parallel inputs • Returns only a Halted Bit—indicating an error has occurred—and an In Position Bit per axis • Sequence must be pre-programmed in Event Control
Parallel Position Mode	• Any position can be moved to in a single PLC scan • Any open loop drive can be triggered in a single PLC scan • Multiple profiles can be selected for any move	• Requires a PLC • Returns only a Halted Bit—indicating an error has occurred—and an In Position Bit per axis • Issues only Go and Open Loop commands • Gives commands to only one axis per scan
Command Mode	• Any RMC command can be issued • Any status information can be retrieved (including Actual Position, Actual Speed, Drive, Error Bits, and other Status Bits)	• Requires a PLC • Requires 2 PLC scans per command • Gives commands to only one axis per command cycle

Implementation

The following diagram shows the electrical control connections of a single-axis hydraulic system using RMC-DI/O in Input to Event Mode (the PC and its RS232 cable are needed only during setup):



This document discusses only the connections between the Programmable Controller and the RMC. For details on the transducer and drive wiring, look up Wiring Notes in the RMCWin index.

Input to Event mode requires use of the Event Control feature of the RMC. You should first familiarize yourself with Event Control and the Event Step table. The RMCWin online help is a good reference for this information.

Input to Event mode does not require multiple inputs to be switched simultaneously. For this reason, many applications can use the RMC without a controlling PLC. The following points describe the operation of Input to Event mode:

- Parameters and tables used by the RMC are configured using the RMCWin software and stored in the RMC Flash memory.
- All axes are initialized by raising the Run/Stop (CPU input 0) line on the RMC, and all axes are immediately stopped by the falling of the Run/Stop line. Therefore, the Run/Stop input is often connected to the system's emergency-stop button.
- The Ready (CPU output 0) line echoes the Run/Stop input line to notify the controlling system when the RMC is initialized.
- When the Run/Stop line is raised and a Trigger (DI/O inputs 0-15) input rises or falls, the Input to Event table is used to determine which axes receive an event sequence.

The Input to Event table has a row for each edge (rising and falling) of each Trigger input (32 rows total). Each of these rows has a field for each axis; each field may either be left blank or hold an Event Step number (0 to 255).

When an edge occurs on a Trigger input, each axis's field in the row corresponding to the input edge that took place is checked. For each field containing a valid value, the axis starts executing the event sequence beginning with the Event Step number given by the Input to Event table entry.

- Each time the Run/Stop or a Trigger input transitions, the Acknowledge (CPU output 1) line toggles. Stand-alone applications generally ignore this, but PLC-controlled applications may wish to use this signal.
- The eight Status outputs (DI/O outputs 0-7) default to being used for the following:
-

DI/O Output #	**2-4 Axis RMC's**	**5-8 Axis RMC's**
0	Axis 0 In Position	Axis 0 In Position
1	Axis 1 In Position	Axis 1 In Position
2	Axis 2 In Position	Axis 2 In Position
3	Axis 3 In Position	Axis 3 In Position
4	Axis 0 Stop on Error	Axis 4 In Position
5	Axis 1 Stop on Error	Axis 5 In Position
6	Axis 2 Stop on Error	Axis 6 In Position
7	Axis 3 Stop on	Axis 7 In

Error Position

•

Using the RMCWin software, these outputs may be marked to be user-controlled instead of being used for the above default assignments. User-controlled outputs are set and cleared from the Event Step table.

The following features of Input to Event mode add complexity to the mode, and are necessary only for a small number of applications:

- Using RMCWin, any of the Trigger inputs can be marked as Single-axis inputs. When an edge occurs on a Single-axis input, only one field in the Input to Event table is used. This field is given by the row of the input edge that took place and the axis selected by the three Axis Select inputs (CPU input 1, DI/O inputs 16-17).
- **Note:** This feature is not required in most cases where only a single axis is to be affected by an input. Most often this can be done by leaving all axes' event step numbers blank in the Input to Event table except the one axis that is to be affected.

The following table shows how to select an axis with these inputs:

Desired Axis	**DI/O Input 17**	**DI/O Input 16**	**CPU Input 1**
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

- Using RMCWin, any of the Trigger inputs can be marked as Non-linked inputs. When an edge occurs on one of these inputs, the Input to Event table is still referenced to find the axes that have Event Step numbers. However, instead of starting an event sequence on the axis, only the command from the event step is given to the axis and any pre-existing event sequence continues execution.

The following general steps must be taken to set up a system using Input to Event mode:

1. **Design the System**

   Designing the system begins with selecting the appropriate method of communication. First, decide whether one of the RMC's field bus solutions fits your application, and if you decide to use digital I/O, then decide which of the communication modes fits your application.

   If you decide to use digital I/O using Input to Event mode, then you must design the wiring of the system, the event step table, and the input to event table.

2. **Program the Event Step Table**

Programming the Event Step table is described in the RMCWin online help. The event step table will hold the majority of the controlling logic. Be sure to save your Event Step table both in the RMC Flash memory and on disk from RMCWin.

3. **Program the Input to Event Table**

Programming the Input to Event table is described in the RMCWin online help. This table serves the purpose of mapping edges of inputs to event sequences in the Event Step table. Be sure to save your Input to Event table both in the RMC Flash memory and on disk from RMCWin.

4. **Configure the RMC Communication**

The following steps are required to configure the Communication DI/O from RMCWin:

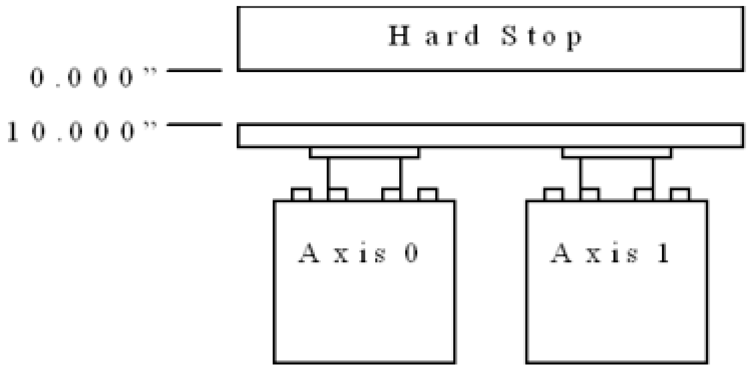
- Connect the serial port to the module being configured.
- On the Tools menu, click Module Configuration.
- In the Slots list, select the Communication DI/O item, and then click Slot options.
- In the DI/O mode list, select Input to Event mode.
- If you need to invert any inputs or outputs to match your hardware, select the appropriate check boxes in the Invert inputs and Invert outputs areas.
- If you wish to use the Communication DI/O counter feature, click either Edge or Quadrature under Counter Type. See Using Counters in the RMCWin online help for more details.
- Click Mode options.
- If you want to define one or more inputs as either single axis or non-linked, then use the check boxes in the Single-axis inputs and Non-linked inputs areas.
- If desired, you may select to control any of the outputs by checking the appropriate User-controlled outputs check boxes.
- Click OK.
- Click Update RMC.
- The Update Module Configuration dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module manually.

5. **Wire, Test, and Tune the System**

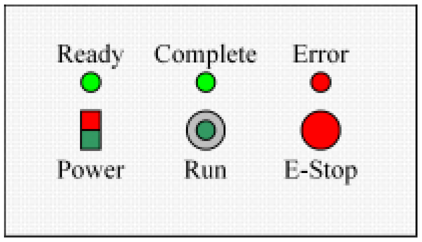
Wiring and testing should follow your design. Tuning the system is described in the RMCWin online help. Be sure to save your tuning parameters both in the RMC Flash memory and on disk from RMCWin.

Sample Application

In this application a user wants to use a motion controller to precisely control a pair of hydraulic cylinders to the 1/1000th of an inch resolution. The cylinders are part of a press and are tied together at the surface of the press. The user wishes to have the cylinders begin at 10.000”, extend to 0.250”, and retract back to 10.000”, as shown in the following diagram:



The following simple control panel will be used for this application:



The buttons and indicators are to operate as follows:

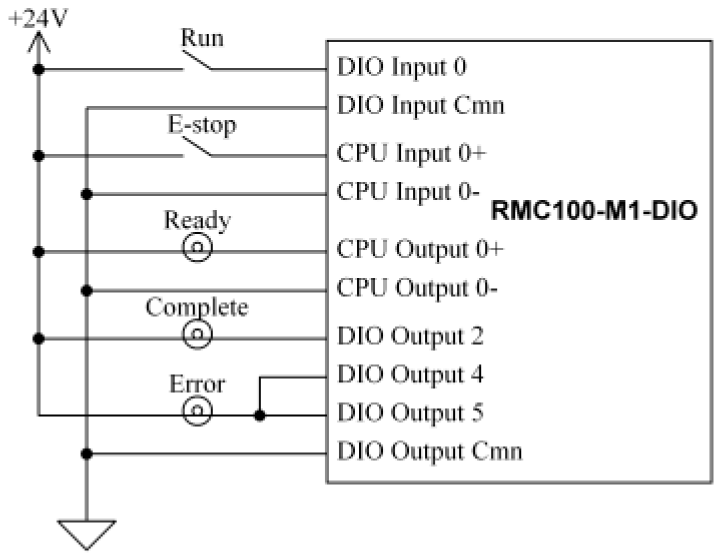
- The E-stop button—closed when pulled, and open when pushed—must be pulled in order for the system to run. When pressed, the system will halt immediately.
- The normally-open Run button starts the extension and retraction cycle of the cylinders.
- The green Ready light indicates when power is supplied and the E-stop is not pushed.
- The green Complete light indicates when the cycle is completed.
- The red Error light indicates that the cylinders stopped because of an error (for example, perhaps the cylinders lagged behind the target positions by too much).

1. Design the System

This application requires two axes, and we will assume that the position feedback is given by magnetostrictive displacement transducers (MDT’s). Therefore, the appropriate module to use is the RMC100-M1-DI/O.

The RMC controls based on the transducer or encoder counts received. However, counts rarely correspond to meaningful engineering units. Therefore, the RMC allows the user to convert counts into meaningful position units by using Scale and Offset parameters. Refer to RMCWin’s online documentation for details. In this application, the position units used will be thousandths of an inch. Therefore, speeds will be given in thousandths of an inch per second.

The following wiring diagram would allow the above control panel to work with the RMC in Input to Event mode:



Simply by wiring the system, many of the controls are already handled:

- The E-stop button enables or halts the axes as wired into the Run/Stop (CPU input 0) input of the RMC.
- The Ready light works as specified because the Ready (CPU output 0) output on the RMC is on when the system is not halted by the Run/Stop.
- The Error light works as specified because it combines the Stop on Error status outputs (DI/O outputs 4 and 5) of the two axes. Therefore, when either axis has an error, the light will illuminate.

2. Program the Event Step Table

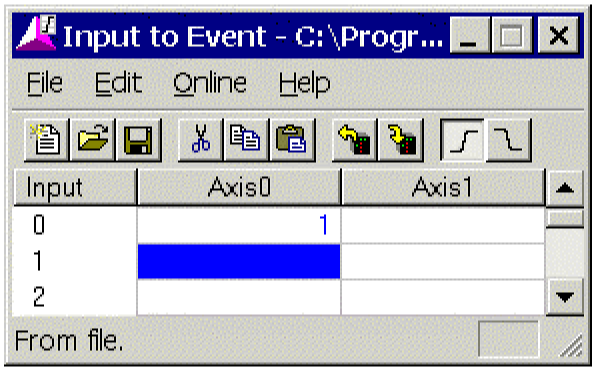
We now need to write an event sequence that will generate the desired sequence on the two axes and turn on DI/O output 2 to illuminate the Complete light at the end. A complete listing of available commands can be obtained from RMCWin's online help. The following event sequence will be used to control both axes:

Each step is described below, beginning with the first step that will be executed:

1. A Reset Outputs (]) command is issued. Its hexadecimal Command Value indicates which bits to reset. Therefore, bit 2 (hexadecimal 00004) represents DI/O output 2. The DelayMS (D) link type and its 0 link value indicate to delay 0 milliseconds and then go to step 2.
2. Both axes are commanded to Go (G) to 250 position units. The AxesInPos (A) link type waits until axes 0 and 1 are both in position, and then goes to step 3.
3. Both axes are commanded to Go (G) to 10,000 position units. The AxesInPos (A) link type waits until axes 0 and 1 are both in position, and then goes to step 4.
4. A Set Outputs ([) command is issued, which turns on DI/O output 2. The event sequence then ends by going back to event step 0.
5. This step ends the event step sequence because it has no link type.

3. Program the Input to Event Table

The final step is to cause DI/O input 0 to trigger the event sequence shown. This is done with the following simple Input to Event table:



Input	Axis0	Axis1
0	1	
1		
2		

This single table entry causes axis 0 to start the event sequence beginning with step 1 whenever DI/O input 0 has a rising edge.

4. Configure the RMC Communication

The steps required for this procedure are described in the Implementation section of this Technical Brief. For this application, you should not need to use (1) inverted inputs, (2) inverted outputs, (3) counters, (4) single-axis inputs, (5) non-linked inputs, or (6) user-controlled outputs.

5. Wire, Test, and Tune the System

The system should be wired as described in the design above. Test the functionality of the final system, and finally tune the system as described in the RMCWin online help.

Reference

Throughout this technical note, references are made to RMCWin online help index entries. To obtain the RMCWin software package, contact Delta Computer System's web site (www.deltacompsys.com).

5.1.6.9 Technical Brief: Using the RMC Discrete I/O Parallel Position Mode

Abstract

The RMC-DI/O is capable of sophisticated motion control using small and inexpensive Programmable Controllers with simple discrete I/O. An RMC with a DI/O communication interface is capable of four discrete I/O interfaces: Command Mode, Input to Event Mode, and Parallel Position Mode. Of these four communication modes, Parallel Position mode is easiest to use for applications in which an axis must be able to move to numerous—perhaps calculated—positions.

This technical brief will compare the four discrete I/O interfaces of the RMC100 series product-line, describe implementing Parallel Position Mode, and finally provide a sample application using Parallel Position Mode.

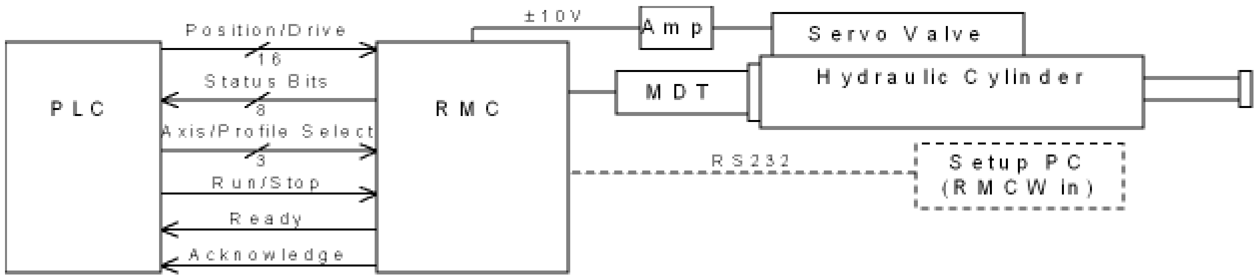
DI/O Communication Mode Comparison

The following chart lists the advantages and disadvantages of each communication mode. Each word or phrase in bold print appears in RMCWins online help index.

Interface Mode	Advantages	Disadvantages
Parallel Position Mode	• Any position can be moved to in a single PLC scan • Any open loop drive can be triggered in a single PLC scan • Multiple profiles can be selected for any move	• Requires a PLC • Returns only a Halted Bitindicating an error has occurredand an In Position Bit per axis • Issues only Go and Open Loop commands • Gives commands to only one axis per scan • Is limited in number of speed/acceleration profiles
Parallel Event Mode	• Up to four axes may be commanded at once • PLC or thumb-wheel switches may be used • Allows use of Event Control feature	• Requires parallel inputs • Returns only a Halted Bitindicating an error has occurredand an In Position Bit per axis • Sequence must be pre-programmed in Event Control
Command Mode	• Any RMC command can be issued • Any status information can be retrieved (including Actual Position, Actual Speed, Drive, Error Bits, and other Status Bits)	• Requires a PLC • Requires 2 PLC scans per command • Gives commands to only one axis per command cycle
Input to Event Mode	• Does not require a PLC • Multiple axes may be given commands from a single input • Allows use of Event Control feature	• Returns only a Halted Bitindicating an error has occurredand an In Position Bit per axis • Issues only Start Event commands

Implementation

The following diagram shows the electrical control connections of a single-axis hydraulic system using RMC-DI/O in Parallel Position Mode (the PC and its RS232 cable are needed only during setup):



This document discusses only the connections between the Programmable Controller and the RMC. For details on the transducer and drive wiring, look up Wiring Notes in the RMCWin index.

The following points describe the operation of Parallel Position mode:

- Parameters and tables used by the RMC are configured using the RMCWin software and stored in the RMC Flash memory.
- All axes are initialized by raising the Run/Stop (CPU input 0) line on the RMC, and all axes are immediately stopped by the falling of the Run/Stop line on the RMC. Therefore, an emergency-stop button often controls the Run/Stop input.
- The Ready (CPU output 0) line matches the Run/Stop input line to give feedback to the controlling system that the RMC is ready to take commands.
- When the Run/Stop line is set, the Position/Drive or Axis/Profile Select (described below) lines are monitored for changes. When any of these inputs change, and they remain stable for a user-configured duration (between 2 and 20 milliseconds), a new command is issued to the RMC.
- The Axis/Profile Select (CPU input 1 and DI/O inputs 16-17) lines select both the profile of the new command, and the axis the command is sent to.

The profile is selected from one of the first eight profile entries in the RMCs Profile table. The following chart indicates how the inputs are used to select the profile:

CPU Input 1	DI/O Input 17	Input 16
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

The bits used to select the axis that receives the command change depending on the total number of axes in the module. The following charts indicate how the inputs are used to select the axis:

Two axes:

DI/O Input 16	Axis #
0	0

1	1

Three or four axes:

DI/O Input 17	Input 16	Axis #
0	0	0
0	1	1
1	0	2
1	1	3

Four or more axes:

CPU Input 1	DI/O Input 17	DI/O Input 16	Axis #
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

- The user may select from RMCWin whether profiles 4-7 are used as open or closed loop commands. Profiles 0-3 are always used as closed loop commands.
- When a closed loop profile is selected, the Position/Drive lines (DI/O inputs 0-15) give the 16-bit requested position. The move begins immediately after the inputs are stabilized (described above). The axis commanded and the profile used are selected by the Axis/Profile Select lines.
- When an open loop profile is selected, the Position/Drive lines (DI/O inputs 0-15) give the 16-bit drive in millivolts. The move begins immediately after the inputs are stabilized (described above). The axis commanded and the profile used are selected by the Axis/Profile Select lines.
- Each time the RMC executes a new command, it updates the Status Bits (described below) and toggles the Acknowledge line (CPU output 0).

It is important to wait for the Acknowledge line to toggle before using the Status Bits; otherwise, the bits may reflect the status of a previous move. For example, suppose the In Position bit is set from a previously completed move. If the Acknowledge line has not toggled before the PLC uses the In Position bit, the newly commanded move will look as though it completed immediately.

- The eight Status outputs (DI/O outputs 0-7) are used for the following:

DI/O Output #	2-4 Axis RMCs	5-8 Axis RMCs
0	Axis 0 In Position	Axis 0 In Position
1	Axis 1 In Position	Axis 1 In Position
2	Axis 2 In Position	Axis 2 In Position

3	Axis 3 In Position	Axis 3 In Position
4	Axis 0 Stop on Error	Axis 4 In Position
5	Axis 1 Stop on Error	Axis 5 In Position
6	Axis 2 Stop on Error	Axis 6 In Position
7	Axis 3 Stop on Error	Axis 7 In Position

The following general steps must be taken to set up a system using Parallel Position mode:

1. Design the System

Designing the system begins with selecting the appropriate method of communication. First, decide whether one of the RMCs field bus solutions fits your application, and if you decide to use digital I/O, then decide which of the communication modes fits your application.

1. If you decide to use digital I/O using Parallel Position mode, then you must design the wiring of the system and the event step table.

2. Program the Profile Table

Programming the Profile table is described in the RMCWin online help. The profile table holds the speeds, accelerations, and decelerations used by the moves. Be sure to save your Profile table both in the RMC Flash memory and on disk from RMCWin.

3. Configure the RMC Communication

The following steps are required to configure the Communication DI/O from RMCWin:

- Connect the serial port to the module being configured.
- On the Tools menu, click Module configuration.
- In the Slots list, select the Communication DI/O item, and then click Slot options.
- In the DI/O mode list, select Parallel Position mode.
- If you need to invert any inputs or outputs to match your hardware, select the appropriate check boxes in the Invert inputs and Invert outputs areas.
- Click Mode options.
- In the Input Filter box, enter the number of milliseconds you wish to have the RMC wait for the inputs to settle. You may need to change this more than once if you are not sure how long you should delay. If you dont need the speed, it may be wise to be conservative and select 20 ms.
- If you wish to do open loop moves as well as closed, select the Allow open loop select check

box.

- Click OK.
- Click Update RMC.
- The Update Module Configuration dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module manually.

4. Wire, Test, and Tune the System

Wiring and testing should follow your design. Tuning the system is described in the RMCWin online help. Be sure to save your tuning parameters both in the RMC Flash memory and on disk from RMCWin.

Sample Application

The customer needs a single axis to move between two positions. The first position is the retracted position for the cylinder and is fixed; this position is defined at 0 inches. The second position is calculated by the PLC and may be anywhere from 10 to 20 inches, depending on the measured size of the raw materials coming down a production line.

We will assume that the customer decided 5 inches per second was the maximum safe extension speed, and that 10 inches per second was the maximum safe retracting speed.

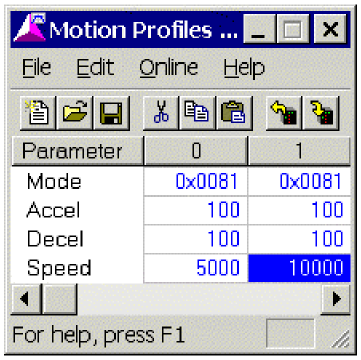
1. Design the System

The RMC controls based on the transducer or encoder counts received. However, counts rarely correspond to meaningful engineering units. Therefore, the RMC allows the user to convert counts into meaningful position units by using Scale and Offset parameters. Refer to RMCWins online documentation for details. In this application, the position units used will be thousandths of an inch. Therefore, speeds will be given in thousandths of an inch per second.

Because only one axis of MDT feedback is needed, the smallest RMC module availablean RMC100-M1-DI/O modulewill be used.

2. Program the Profile Table

Because two speeds are required, two motion profiles will be used. As shown in the charts for selecting profiles and axes above, profiles 0, 2, 4, and 6 are available for axis 0. Therefore, profiles 0 and 2 will be used. The following profile table entries defines the speeds required by the customer for each move (remember that profile 1 is unused in this example):



Parameter	0	1
Mode	0x0081	0x0081
Accel	100	100
Decel	100	100
Speed	5000	10000

Once this Profile table and the desired configuration parameters are stored in the RMC Flash memory, the following steps are used to make the moves.

3. **Configure the RMC Communication**

   The steps required for this procedure are described in the Implementation section of this Technical Brief. You should not need to invert any inputs or outputs, nor should you need to allow open loop select. You may need to change the Input Filter setting depending on your I/O speed.

4. **Wire, Test, and Tune the System**

   The system should be wired as described in the design above. Test the functionality of the final system, and finally tune the system as described in the RMCWin online help.

   When the system is completed, the following steps can be taken to make the moves:

- Initializing the axis:
    1. Set the Run/Stop (CPU input 0) line high.
    2. Wait for the Ready (CPU output 0) line to be high.
- Moving the axis to its home position:
    1. When returning to the home position, the user wants to use profile 2 (which goes 10 inches per second). Therefore, CPU input 1 and DI/O input 16 will be reset, and DI/O input 17 will be set high. Notice that this step also selects axis 0.
    2. Place the value 1500 (for 1.5 or 1500 thousandths of an inch) in binary on DI/O inputs 0-15.
    3. Wait for the Acknowledge (CPU output 1) line to toggle, indicating the command has been received.
    4. Wait for the In Position bit of axis 0 (DI/O output 0) to be high, indicating the axis has reached its commanded position.
- Moving the axis to its calculated extension position:
    1. When extending, the user wants to use profile 0 (which goes 5 inches per second). Therefore, CPU input 1, DI/O input 17, and DI/O input 16 will all be reset. Notice that this step also selects axis 0.
    2. Place the calculated requested position (in thousandths of inches) in binary on DI/O inputs 0-15.
    3. Wait for the Acknowledge (CPU output 1) line to toggle, indicating the command has been received.
    4. Wait for the In Position bit of axis 0 (DI/O output 0) to be high, indicating the axis has reached its commanded position.
- Stopping the axis in an emergency:
    1. Lower the Run/Stop (CPU input 0) line.

Reference

1. Throughout this technical note, references are made to RMCWin online help index entries. To obtain the RMCWin software package, contact Delta Computer Systems web site (www.deltacompsys.com).

5.1.6.10 Technical Brief: Using the RMC Discrete I/O Parallel Event Mode

Abstract

The RMC-DI/O is capable of sophisticated motion control using small and inexpensive Programmable Controllers with simple discrete I/O. An RMC with a DI/O communication interface is capable of four discrete I/O interfaces: Command Mode, Input to Event Mode, Parallel Position Mode, and Parallel Event Mode. Of these communication modes, Parallel Event mode is the best fit when the motion can be pre-programmed using the RMC's Event Control feature and parallel inputs can be provided to the RMC—such as through a PLC or thumb-wheel switch.

This technical brief will compare the discrete I/O interfaces of the RMC100 series product-line, describe implementing Parallel Event Mode, and finally provide a sample application using Parallel Event Mode.

DI/O Communication Mode Comparison

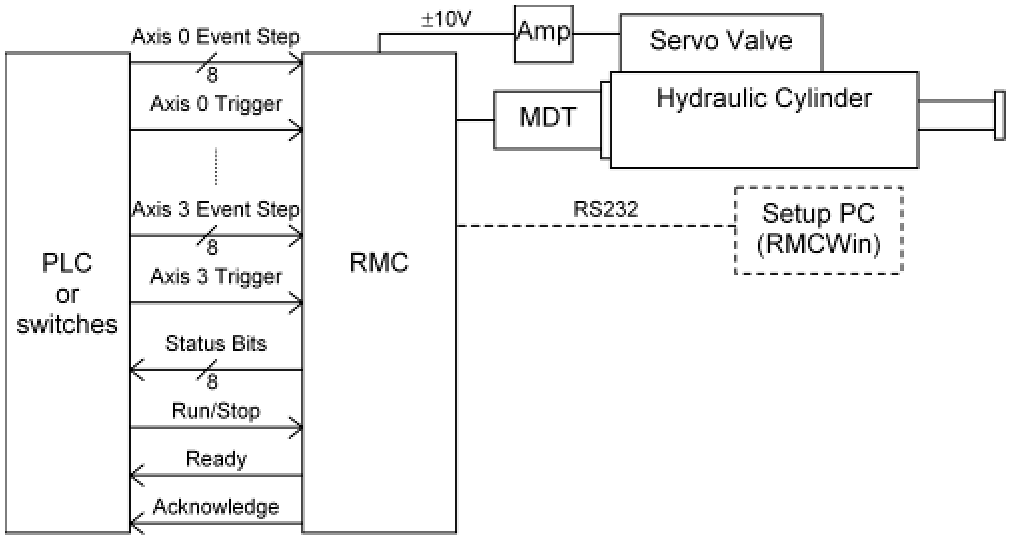
The following chart lists the advantages and disadvantages of each communication mode. Each word or phrase in bold print appears in RMCWin's online help index.

Interface Mode	Advantages	Disadvantages
Parallel Event Mode	• Up to four axes may be commanded at once • PLC or thumb-wheel switches may be used • Allows use of Event Control feature	• Requires parallel inputs • Returns only a Halted Bit—indicating an error has occurred—and an In Position Bit per axis • Sequence must be pre-programmed in Event Control
Parallel Position Mode	• Any position can be moved to in a single PLC scan • Any open loop drive can be triggered in a single PLC scan • Multiple profiles can be selected for any move	• Requires a PLC • Returns only a Halted Bit—indicating an error has occurred—and an In Position Bit per axis • Issues only Go and Open Loop commands • Gives commands to only one axis per scan • Is limited in number of speed/acceleration profiles
Command Mode	• Any RMC command can be issued • Any status information can be retrieved	• Requires a PLC • Requires 2 PLC scans per command • Gives commands to only

	(including Actual Position, Actual Speed, Drive, Error Bits, and other Status Bits)	one axis per command cycle
Input to Event Mode	• Does not require a PLC • Multiple axes may be given commands from a single input • Allows use of Event Control feature	• Returns only a Halted Bit—indicating an error has occurred—and an In Position Bit per axis • Issues only Start Event commands

Implementation

The following diagram shows the electrical control connections of a single-axis hydraulic system using RMC-DI/O in Parallel Event Mode (the PC and its RS232 cable are needed only during setup):



This document discusses only the connections between the Programmable Controller and the RMC. For details on the transducer and drive wiring, look up Wiring Notes in the RMCWin index.

The following points describe the operation of Parallel Event mode:

- Parallel Event mode uses the following digital inputs and outputs. The following table is intended to match input and output number with the names associated with each. Their uses will be described below.
-

CPU DI/O:	
Input 0	Run/Stop
Input 1	Unused
Output 0	Ready
Output 1	Acknowledge
Communication DI/O:	
Inputs 0-7	Axis 0 Event Step

Inputs 8-15	Axis 1 Event Step
Input 16	Axis 0 Trigger
Input 17	Axis 1 Trigger
Outputs 0-7	Status Bits

Sensor DI/O (required only if more than two axes are used):

Inputs 0-7	Axis 2 Event Step
Inputs 8-15	Axis 3 Event Step
Input 16	Axis 2 Trigger
Input 17	Axis 3 Trigger
Outputs 0-7	Unused

- Parameters and tables used by the RMC are configured using the RMCWin software and stored in the RMC Flash memory.
- All axes are initialized by raising the Run/Stop (CPU input 0) line on the RMC, and all axes are immediately stopped by the falling of the Run/Stop line on the RMC. Therefore, an emergency-stop button often controls the Run/Stop input.
- The Ready (CPU output 0) line matches the Run/Stop input line to give feedback to the controlling system that the RMC is ready to take commands.
- When the Run/Stop line is set, the Trigger inputs for all axes are monitored. When the Trigger switches state (either from on to off or vice versa), then all nine inputs for the axis (Trigger plus Event Step) are monitored; if they remain stable for a user-configured duration (between 2 and 20 milliseconds) a new command is issued to the RMC.
- When a command is received, the Event Step number given in binary on that axis's eight Event Step inputs is read and the axis begins an event sequence with that event step.
- Each time the Run/Stop changes, or a new command is executed on one or more axes, the Acknowledge (CPU output 0) line toggles.

  It is important to wait for the Acknowledge line to toggle before using the Status Bits; otherwise, the bits may reflect the status of a previous move. For example, suppose the In Position bit is set from a previously completed move. If the Acknowledge line has not toggled before the PLC uses the In Position bit, the newly commanded move will look as though it completed immediately.
- The eight Status outputs (DI/O outputs 0-7) are used for the following:

  1.

DI/O Output #	2-4 Axis RMC's	5-8 Axis RMC's
0	Axis 0 In Position	Axis 0 In Position
1	Axis 1 In Position	Axis 1 In Position
2	Axis 2 In Position	Axis 2 In Position
3	Axis 3 In Position	Axis 3 In Position
4	Axis 0 Stop on	Axis 4 In Position

	Error	
5	Axis 1 Stop on Error	Axis 5 In Position
6	Axis 2 Stop on Error	Axis 6 In Position
7	Axis 3 Stop on Error	Axis 7 In Position

2.

Using the RMCWin software, these outputs may be marked to be user-controlled instead of being used for the above default assignments. User-controlled outputs are set and cleared from the Event Step table.

The following general steps must be taken to set up a system using Parallel Event mode:

1. Design the System

Designing the system begins with selecting the appropriate method of communication. First, decide whether one of the RMC's field bus solutions fits your application, and if you decide to use digital I/O, then decide which of the communication modes fits your application.

If you decide to use digital I/O using Parallel Event mode, then you must design the wiring of the system and the event step table.

2. Program the Event Step Table

Programming the Event Step table is described in the RMCWin online help. The event step table will hold the majority of the controlling logic. Be sure to save your Event Step table both in the RMC Flash memory and on disk from RMCWin.

3. Configure the RMC Communication

The following steps are required to configure the Communication DI/O from RMCWin:

1. Connect the serial port to the module being configured.
2. On the Tools menu, click Module configuration.
3. In the Slots list, select the Communication DI/O item, and then click Slot options.
4. In the DI/O mode list, click Parallel Event mode.
5. If you need to invert any inputs or outputs to match your hardware, select the appropriate check boxes in the Invert inputs and Invert outputs areas.
6. Click Mode options.
7. In the Input Filter box, enter the number of milliseconds you wish to have the RMC wait for

the inputs to settle. You may need to change this more than once if you are not sure how long you should delay. If you don't need the speed, it may be wise to be conservative and select 20ms.

8. If desired, you may select to control any of the outputs by selecting the appropriate User-controlled outputs check boxes.
9. Click OK.
10. Click Update RMC.
11. The Update Module Configuration dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module manually.

4. Wire, Test, and Tune the System

Wiring and testing should follow your design. Tuning the system is described in the RMCWin online help. Be sure to save your tuning parameters both in the RMC Flash memory and on disk from RMCWin.

Sample Application

The customer needs a single axis to move between two positions. The user will select one of ten different positions to which the cylinder will extend and then retract back to a home position. The home position will be defined at 0". The ten extend positions are required by different products that may be produced and are known ahead of time (half-inch steps between 4" and 9.5"). The user wishes to use a thumb-wheel to select the set (sequence) and a button to initiate the motion.

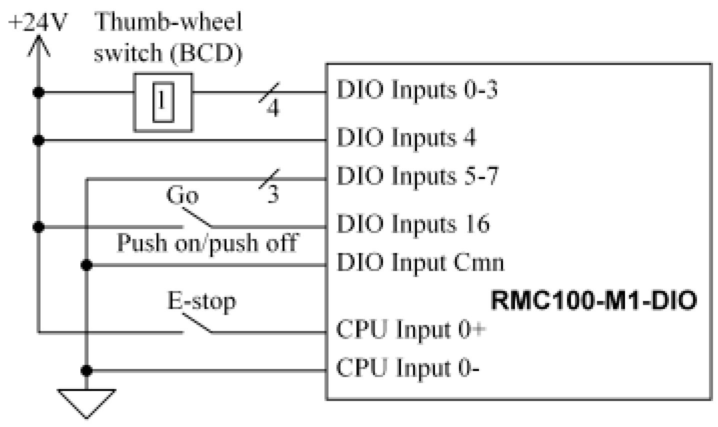
We will assume that the customer decided 5 inches per second was the maximum safe extension speed, and that 10 inches per second was the maximum safe retracting speed.

1. Design the System

The RMC controls based on the transducer or encoder counts received. However, counts rarely correspond to meaningful engineering units. Therefore, the RMC allows the user to convert counts into meaningful position units by using Scale and Offset parameters. Refer to RMCWin's online documentation for details. In this application, the position units used will be thousandths of an inch. Therefore, speeds will be given in thousandths of an inch per second.

Because only one axis of MDT feedback is used, the smallest RMC module available—an RMC100-M1-DI/O module—will be used.

Using Parallel Event mode, the thumb-wheel can be used to trigger an event sequence and will select the starting event step number. The following wiring diagram is used:



Notice that the eight Event Step number bits are wired to hold the following values:

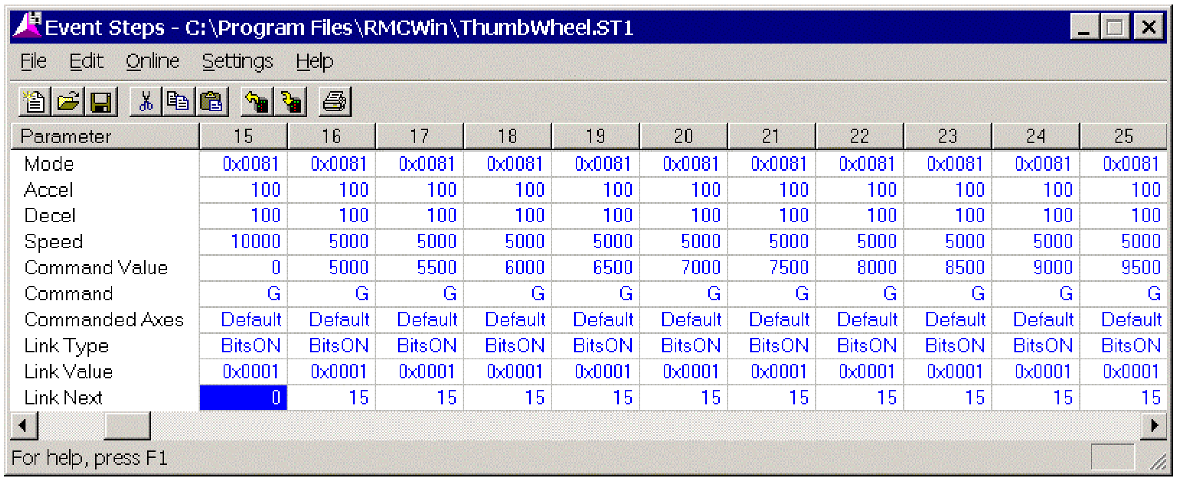
Bit #:	7	6	5	4	3	2	1	0
Value:	0	0	0	1	X	X	X	X

X = Set by thumb-switch

When this binary number is converted to decimal, the selectable step numbers range is 16-25. The reason bit 4 is tied high is to avoid using event step 0, without which the range would be 0-15. Event step 0 is used—by convention—as a step which does nothing.

2. Program the Event Step table

We will use the steps selected by the thumb-wheel (16-25) to perform the first move. Each of those ten steps will wait until the axis is in position, and then skip to step 15, which will retract back to the home position, and then return to step 0 when in position. The following screen shot demonstrates this table:


Event Steps - C:\Program Files\RMCWin\ThumbWheel.ST1
File Edit Online Settings Help
For help, press F1

Parameter	15	16	17	18	19	20	21	22	23	24	25
Mode	0x0081	0x0081	0x0081	0x0081	0x0081	0x0081	0x0081	0x0081	0x0081	0x0081	0x0081
Accel	100	100	100	100	100	100	100	100	100	100	100
Decel	100	100	100	100	100	100	100	100	100	100	100
Speed	10000	5000	5000	5000	5000	5000	5000	5000	5000	5000	5000
Command Value	0	5000	5500	6000	6500	7000	7500	8000	8500	9000	9500
Command	G	G	G	G	G	G	G	G	G	G	G
Commanded Axes	Default	Default	Default	Default	Default	Default	Default	Default	Default	Default	Default
Link Type	BitsON	BitsON	BitsON	BitsON	BitsON	BitsON	BitsON	BitsON	BitsON	BitsON	BitsON
Link Value	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001	0x0001
Link Next	0	15	15	15	15	15	15	15	15	15	15

This table must be downloaded to the module, and then stored in the RMC's Flash memory.

3. Configure the RMC Communication

The steps required for this procedure are described in the Implementation section of this Technical Brief. You should not need to invert any inputs or outputs, nor should you need to select any additional user-controlled outputs.

4. **Wire, Test, and Tune the System**

The system should be wired as described in the design above. Test the functionality of the final system, and finally tune the system as described in the RMCWin online help.

Reference

Throughout this technical note, references are made to RMCWin online help index entries. To obtain the RMCWin software package, contact Delta Computer System's web site (www.deltacompsys.com).

5.2 Ethernet

5.2.1 RMC Ethernet Module Overview

Uses for the RMC ENET Module

The RMC ENET module allows the RMC to communicate with other Ethernet devices, including many PLCs and PCs. This port can be used in a number of ways, as listed below:

- **Control or Monitoring from a Third Party PLC or Software Package**
  This is the most common and powerful use for the RMC ENET module. The RMC can emulate the Ethernet protocols of most major PLCs, including PLCs from the following manufacturers: Allen-Bradley, Automationdirect.com, Modicon, Omron, Siemens, and SoftPLC. As a result, any PLC or PC-based control or HMI package that can access data in one of these PLCs should be able to access data in the RMC. See Using the RMC ENET with Programmable Controllers for details.

- **Communication with RMCWin**
  The RMC ENET module allows communication with RMCWin in place of the serial port connection. Using Ethernet in place of a serial cable to communicate between the RMC and RMCWin has the following advantages:

  - Much higher performance than RS232 in most cases (using low-speed connections such as modems will slow this down).
  - Longer cable distances are allowed.
  - Eliminates running the extra serial cable if it would otherwise not be used.
  - Allows addressing many RMCs from a single PC without switching serial cables or adding more serial ports to the PC.

  See Using the RMC ENET with RMCWin for more details.

- **Control or Monitoring from a Custom Device or Software**
  Some advanced users will want to write their own control or HMI package on the PC. There are several options available for such users:

  - For Windows-based applications, the RMC ENET ActiveX Control provides a simple interface for communicating with the RMC over Ethernet. Visual Basic, Java, and Visual C++ all support ActiveX Controls. See the RMCLink topic for details.
  - For any applications with access to a TCP/IP API (such as Winsock for Windows and BSD Sockets for Unix), any of the protocols supported by the RMC can be used. See Using

Sockets to Access the RMC ENET for details.

Note: The RMC ENET does not support any of the native protocols built into Windows. That is, the RMC does not support Web browsers, FTP, e-mail, and browsing through Network Neighborhood.

Configuring the RMC Ethernet Module

Setting up the RMC Ethernet module requires entering only a few TCP/IP parameters: configuration type, IP address, subnet mask, and gateway address. See RMC Ethernet IP Address Setup for details on entering these values. Users who have set up networks before will find it straight-forward to select these values. First-time TCP/IP users will need to read the following topics for suggestions on values to use:

- Understanding IP Addressing
- Setting up an Isolated TCP/IP Control Network

Note: It is not necessary to select the application protocol or device with which you will be communicating. The RMC will automatically respond to all supported devices.

Ethernet Module LEDs

LAN LED

The LAN LED indicates when network traffic is being sent or received:

- **Off:** The Ethernet link is down or there is no current activity on the network.
- **Flashing Green:** The Ethernet link is up and send or receive activity has been detected.

LINK LED

The LINK LED indicates whether the physical Ethernet link is up or down.

- **Off:** The Ethernet link is down.
- **Steady Green:** The Ethernet link is up.

Ethernet Module Firmware

The RMC Ethernet Protocols are divided into two different firmware files. When updating the RMC100 Ethernet firmware, you must choose the file that contains the protocol you need. The files are called Protocol Group E and Protocol Group F:

- **Protocol Group E**

  a. EtherNet/IP
  b. Allen-Bradley CSP (a.k.a. DF1 over Ethernet)
  c. Modbus/TCP
  d. Omron FINS
  e. AutomationDirect.com HEI

- **Protocol Group F**

-
    a. ISO-over-TCP for Siemens S7 controllers
    b. CAMP for Simatic 505 with CTI Ethernet

New RMC100-ENET modules will be shipped with Protocol Group E. However, both groups are available for download from the Delta website, and both are supported.

RMC ENET Supported Numbers of TCP and CIP Resources

TCP Connections:

4 TCP connections (all inbound)

Examples:

- RMCWin
- Each EtherNet/IP I/O connection is set up using an EtherNet/IP TCP connection
- Any other PLC communication

CIP Connected Messaging:

The RMC ENET limits CIP connections as follows:

- Max of 8 total CIP connections
- Max of 4 I/O CIP connections

Examples:

- EtherNet/IP I/O Data connection with a ControlLogix
- EtherNet/IP Input Data connection with a ControlLogix
- ControlLogix message to RMC ENET (cache enabled)

5.2.2 Using the RMC ENET with Programmable Controllers

Ethernet Compatibility

It is important to understand that the RMC ENET module will not be able to communicate with every device designated as communicating via Ethernet and TCP/IP. These two sets of standards—Ethernet and TCP/IP—specify the electrical specifications and routing protocols, but do not specify the content or format of the data that is transmitted. The data format is called the Application Protocol. For a complete list and diagram of protocols supported by the RMC Ethernet module, see RMC Ethernet Protocols.

This issue is closely analogous to serial ports. Two devices must agree on signal levels, baud rate, and data, stop, and parity bits. However, these serial devices still cannot communicate if they do not understand one another's data. Example: Try connecting a serial cable between a PC running TISOFT and an Allen-Bradley SLC 5/05. TISOFT expects one protocol, while the Allen-Bradley expects another.

It is important to know which devices the RMC supports. We support the following devices, and any other device that communicates with the same protocol as one of these:

- Allen-Bradley Ethernet PLC-5 (1785-L20E, -L40E, or -L80E)
- Allen-Bradley PLC-5 with the PLC-5 Ethernet Interface Module (1785-ENET)
- Allen-Bradley Ethernet SLC 5/05
- Allen-Bradley ControlLogix with Ethernet Interface Module (1756-ENET)
- Allen-Bradley SoftLogix 5
- Allen-Bradley RSLinx
- Automationdirect.com DirectLogic 205 (240, 250) with an H2-ECOM Ethernet module
- Automationdirect.com DirectLogic 405 (430, 440, 450) with an H4-ECOM Ethernet module
- Modicon Quantum with the 140 NOE 211 00 Ethernet TCP/IP module
- Omron CS1 and CV Programmable Controllers with an ETN01 Ethernet module.
- Siemens Simatic TI505 with the CTI 2572 Ethernet TCP/IP module
- Siemens S7-300 with the CP 343-1 TCP module
- Siemens S7-400 with the CP 443-1 TCP module
- SoftPLC Corporation's SoftPLC

It is Delta's goal to support all major Ethernet devices. However, we need your feedback to determine which Ethernet devices are being used, so please call Delta Computer Systems, Inc. to discuss additional Ethernet device support.

PC-Based Control and HMI Packets

Any existing software package that supports reading from or writing to one of the above PLCs should be able to read from and write to the RMC. Most such packages will support at least one of these devices. See Using Other Ethernet Packages with the RMC ENET for further details.

Communicating with Ethernet TCP/IP Masters

For further details on communicating between the RMC and a PLC Ethernet TCP/IP master, select one of the following topics:

- Using Allen-Bradley Controllers with the RMC ENET
- Using Automationdirect.com PLCs with the RMC ENET
- Using the Modicon Quantum with the RMC ENET
- Using Omron PLCs with the RMC ENET
- Using the Siemens Simatic TI505 with the RMC ENET

- Using the SoftPLC with the RMC ENET

5.2.3 Using the RMC ENET with RMCWin

RMCWin 2.0 and newer can communicate directly with the RMC ENET module over Ethernet. This requires RMC ENET firmware dated 20010523 or later.

The following topics relate to getting RMCWin to connect to an RMC ENET:

- Connecting RMCWin to an RMC
- Communication Drivers: TCP/IP Direct to RMC-ENET
- RMC Ethernet IP Address Setup

5.2.4 Ethernet Setup Topics

5.2.4.1 RMC Ethernet IP Address Setup

Note: This section assumes you have read and understood the Understanding IP Addressing topic. That topic describes the meaning of the IP address, subnet mask, and default gateway parameters on this screen.

Each RMC ENET has three basic settings needed for TCP/IP communication to work: the IP address, the subnet mask, and the default gateway. The RMC ENET module allows these to be set manually or dynamically using BOOTP or DHCP. Manual configuration is recommended because the benefits of BOOTP are negligible with the RMC, and DHCP defaults to leasing an IP address. Leasing an IP address is not acceptable for industrial systems, because when the lease expires, the device will no longer be able to communicate.

There are two ways to change the TCP/IP settings in the RMC ENET module.

The first method requires an Ethernet adapter in your PC, and requires that the RMC be on the same network as your PC. Notice that the RMC ENET it not required to have valid TCP/IP settings. Therefore, you can configure and set up your RMC ENET module without ever using the serial port!

The second method requires first establishing a connection to the RMC module. Therefore, it requires either using the serial port or already having valid TCP/IP settings in the RMC ENET. It should be used when you do not have an Ethernet adapter in your PC, you are already connected to the RMC ENET module, or the RMC you want to configure is not on the same network as your PC.

To set up the RMC ENET TCP/IP settings over Ethernet:

1. Start RMCWin.
2. On the Tool menu, click Options.
3. Click the Communication tab.

4. Under Communication Drivers, click TCP/IP Direct to RMC-ENET.

5. Under Settings, either ensure that Autobrowse Local Network is checked or click Refresh.

   You should see all RMCs on the network that your PC's Ethernet adapter is connected to. Notice that you will not see RMCs with RMC ENET firmware dated prior to 20010523, nor will you see TCP/IP-to-RS232 bridges even if they are connected to RMCs.

6. In the browse list, select the RMC ENET you want to configure.

   If you are not sure which RMC in the list corresponds to the physical RMC you want to configure, you may have to disconnect the Ethernet cable from the RMC, wait for it to disappear from the browse list if you are using autobrowse (otherwise click Refresh), then reconnect the cable and see which device comes back onto the list.

7. Click Configure.

8. Click the option button of the desired configuration method.

9. If manual configuration is selected, type the IP address, subnet mask, and default gateway parameters, each in dotted decimal format (e.g. 192.168.0.5). The default gateway parameter is optional.

10. Click OK.

    RMCWin will change the TCP/IP settings in the RMC ENET module. When the update is complete, the browse list will be updated.

To set up the RMC ENET TCP/IP settings using Slot Options:

1. Start RMCWin.

2. Connect RMCWin to the RMC you want to configure. Ensure that the connection status is indicated to be "online" in the status bar. In order to connect over Ethernet, you will need valid TCP/IP settings.

3. On the Tools menu, click Module Configuration.

4. In the Slots list, click the Ethernet item.

5. Click Slot options.

6. In the Ethernet Options dialog box, click the TCP/IP tab.

7. Click the desired configuration method option button.

8. If manual configuration is selected, type the IP address, subnet mask, and default gateway parameters, each in dotted decimal format (e.g. 192.168.0.5). The default gateway parameter is optional.

9. Click Update RMC.

10. The Update Module Configuration dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module manually.

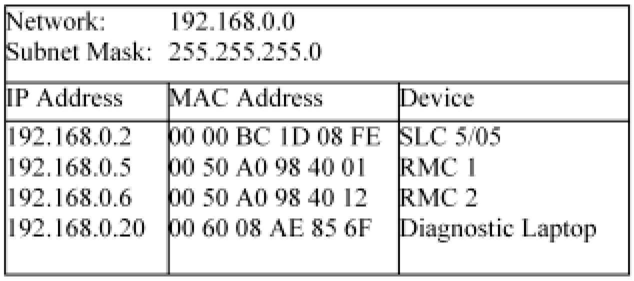
11. To check that the change has taken place, you can use the PING utility from any PC that has it and is on the same network.

Selecting a Configuration Method

The first decision to be made is the method you will use to configure the IP address of your device. Here are the three options selected in the TCP/IP tab of the Ethernet Options dialog box:

- **Manually specify an IP address**

Using this method, the administrator keeps a record of all IP addresses assigned for each network, as well as the subnet mask and default gateway of the network. To use this method, click Specify an IP address, and enter each of the three parameters. If you are creating a stand-alone network, see Setting up a Stand-alone TCP/IP Control Network for suggestions on values to use, otherwise consult your network administrator. An example record for a network might look like this:



Network: 192.168.0.0
Subnet Mask: 255.255.255.0

IP Address	MAC Address	Device
192.168.0.2	00 00 BC 1D 08 FE	SLC 5/05
192.168.0.5	00 50 A0 98 40 01	RMC 1
192.168.0.6	00 50 A0 98 40 12	RMC 2
192.168.0.20	00 60 08 AE 85 6F	Diagnostic Laptop

- **Obtain IP address, subnet mask, and default gateway from a BOOTP server.**

This option causes the RMC to utilize the BOOTstrap Protocol (BOOTP) to determine its IP address, subnet mask, and default gateway. When the RMC powers up, it broadcasts a request for its IP parameters. If a BOOTP server is available on the network, then the BOOTP server will look into a database of mappings from MAC addresses to IP addresses for the MAC address of the RMC that sent the request. If it finds a match, it will give the RMC an IP address, default gateway, and subnet mask. If no BOOTP server responds—either because no BOOTP server exists, or the RMC's MAC address was not found in the BOOTP server's database—the RMC's Ethernet communication channel will not be usable.

If you wish to use BOOTP, contact a network administrator for details on purchasing or obtaining a shareware BOOTP server.

- **Obtain IP address, subnet mask, and default gateway from a DHCP server.**

This option utilizes the Dynamic Host Configuration Protocol (DHCP) to determine the IP parameters in the same way that BOOTP is used for the above option. In fact, DHCP is an extension of BOOTP and adds two features: dynamic selection of IP addresses from a pool of addresses, and lease times for addresses.

The first feature works well for many office situations because each PC can use DHCP, so there is no need for the user to enter a database of each individual IP address and MAC address. Instead, a range of IP addresses is specified, and IP addresses are assigned as they are requested. However, this does not work in industrial situations because most PLCs require that devices refer to one another by IP address. This is not possible if the IP addresses of the devices change from time to time.

The second feature is also not very usable in industrial situations. It allows each address assignment to be assigned for a limited amount of time (in hours or even days). This is intended to be used in conjunction with dynamic IP address assignment to avoid running out of addresses in the pool.

Therefore, the enhanced features of DHCP over those of BOOTP are not useful. However, DHCP still supports the one-to-one mapping of MAC addresses to IP addresses provided by BOOTP. Therefore, DHCP is offered as an alternative only to allow you to purchase either a DHCP or BOOTP server if you choose to use such a protocol.

IP Address
Subnet Mask
Default Gateway
These three fields are described in Understanding IP Addressing. IP Address and Subnet Mask are required parameters if you are manually configuring the RMC. Enter them in dotted decimal notation (e.g. 192.168.0.5 and 255.255.255.0). The Default Gateway parameter is optional. If you choose not to use it, leave it blank, and the RMC will not be able to communicate with devices on networks other than its own. Otherwise, enter a value in dotted decimal notation (e.g. 192.168.0.1).

LAN Settings

On the right side of the RMC TCP/IP Configuration dialog box are the TCP/IP settings for the LAN your PC's Ethernet adapter is connected to. These settings are derived from the IP address and subnet mask set up for your PC's Ethernet adapter.

If the RMC you are configuring will be running on the same network as the PC, then you should use the same subnet mask as the PC, and you should enter a TCP/IP address between the minimum and maximum addresses.

Note: The LAN Settings section is provided for convenience, but it is still necessary to take steps to ensure that the IP address you choose is unique and will remain unique. This typically involves referring to a list of IP addresses reserved on the network and recording the address that is used for the RMC on this list.

5.2.4.2 Setting up a Stand-alone TCP/IP Control Network

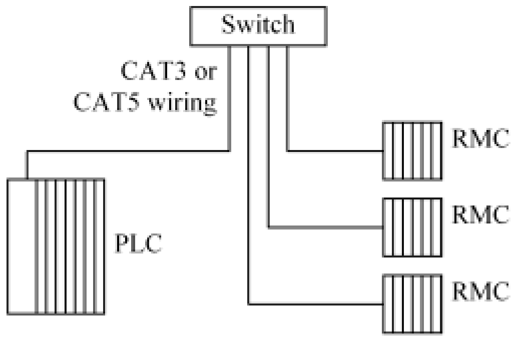
Note: This section is intended only for new networks that will not be connected via a router to another network. If you are going to be adding an RMC to an existing network or you will be creating a new network that will be connected via a router to another network, consult your network administrator.

Note: This section assumes you have read and understood the Understanding IP Addressing topic. Please read that topic first.

Setting up a stand-alone network takes the following steps:

1. **Wire the Network**

The RMC uses the IEEE 802.3 for 10BaseT hardware standard. This means it runs at 10Mbps on twisted pair wiring rated Category 3 or higher, and uses RJ45 connectors. Twisted pair networks generally use a star topology, which means that each device is wired to a single switch device:



Care should be taken to use a high-quality switch that will support your temperature, noise, vibration, and other environmental requirements. It is also important to use a switch rather than a hub to avoid collisions, which reduce the determinism of the network. Both hubs and Category 3 or 5 (commonly called CAT3 or CAT5) cabling are readily available from network supply companies.

2. **Select a network address and subnet mask.**

   By convention, the network address 192.168.0 is never sent over the Internet. Therefore, it is a good choice for the network address of a stand-alone control network. Because this network address is 24 bits long, our subnet mask will be 255.255.255.0. This leaves 254 local addresses (remember that addresses 0 and 255 are reserved) for an IP address range of 192.168.0.1 to 192.168.0.254.

3. **Assign local addresses for each device.**

   We now have a range of 254 IP addresses to assign. The only real requirement when assigning these addresses is to ensure that you do not assign two devices the same IP address. Therefore, it is imperative that you record your IP address assignments for use later when you need to add or replace device. By convention, local address 1 (IP address 192.168.0.1 in our new network) is used as the default gateway. We do not have a gateway/router at this time, but it is a good idea to leave that address unassigned for now.

   So, assign your first device (perhaps a PLC) 192.168.0.2, assign your second device (perhaps an RMC) 192.168.0.3, etc.

4. **Enter the network parameters into each device.**

   The method of assigning the network parameters varies for each type of device. Use the IP address you have assigned, a subnet mask of 255.255.255.0, and leave the default gateway blank for each device. See RMC Ethernet IP Address Setup for details on editing these parameters on the RMC, and consult the manuals of the other devices on your network for details on setting up their TCP/IP parameters.

5.2.4.3 RMC Ethernet Firmware Screen

The Firmware tab on the Ethernet Options dialog holds the following information:

- **Communication Program Version**
  This field gives the version of the main Ethernet program currently running in the RMC Ethernet module. This field will display "Debugger" if no firmware is currently loaded. To update your firmware, see Downloading New Serial/Ethernet Firmware.

- **Boot Version**

This field gives the version of the Boot firmware in the RMC Ethernet module.

- **Loader Version**
  This field gives the version of the internal Loader firmware used for updating the main Ethernet program.

- **Hardware Revision**
  The hardware revision of the RMC Ethernet firmware is displayed. Unlike the above versions, this cannot be field upgraded without replacing the module.

- **Ethernet (MAC) Address**
  Every Ethernet device manufactured is required by the IEEE Standards Organization to have a unique MAC address. This address is also called a LAN MAC address or Ethernet ID. This address is given in the form of six hexadecimal bytes, which can be displayed in several formats: run together (e.g. 0050A0984001), separated by hyphens (e.g. 00-50-A0-98-40-01), or separated by spaces (e.g. 00 50 A0 98 40 01). However, all formats are equivalent. Network administrators may require this value when tracking network problems or configuring BOOTP or DHCP protocols. All devices manufactured by Delta Computer Systems, Inc. begin with "00 50 A0".

The Firmware tab of the Ethernet Options dialog has the following commands available:

- **Update**
  This command initiates the sequence to update the main Ethernet program in the RMC Ethernet module. Use this button only when directed to do so by Delta technical support.

- **Update B/L**
  This command initiates the sequence to update the Boot and Loader firmware in the RMC Ethernet module. Use this button only when directed to do so by Delta technical support.

5.2.4.4 RMC Ethernet Statistics

The Ethernet Statistics window displays a number of diagnostic counters that are useful for monitoring Ethernet performance and troubleshooting problems. This window can be displayed using the following steps:

1. On the main RMCWin window's Window menu, click Ethernet Statistics. Notice that this command is only available for RMC modules with an ENET module.

Note: Using these diagnostic counters requires RMC ENET firmware version 20000915 or later. Also, reading these counters through RMCWin's TCP/IP driver requires RMC ENET firmware version 20010831 or later.

This list of counters is divided up into three logical groups to simplify understanding the values. However, all groups are read from the module and updated together. That is, refreshing the counters while displaying one group of counters also refreshes the other counters.

There are five buttons in this window:

- **Continuous Update checkbox**
  When this box is checked, the counters will be read from the RMC ENET continuously about four times per second. When this box is not checked, the counters are only updated when the Refresh button is clicked.

- **Refresh**
  Pressing this button will read all counters from the module. This operation takes place immediately and only happens once each time the button is pressed. This button is not available if the Continuous Update checkbox is checked.

- **Clear**
  Pressing this button will clear all counters in the module. This is often useful to see the affect of a change to the system. For example, suppose you change your controlling PLC's ladder logic and want to see its affect on the transfer rate. You may want to first start the ladder logic and then clear the statistics to avoid confusing old data with the new.

- **Save**
  Save all the current statistics data to a file.

- **Help**
  Display this topic in the online help.

- **Close**
  Close the Ethernet Statistics window.

Each counter is described below:

General

Note: Each of these counters is limited at 4,294,967,296, and if it reaches this value, then it will wrap back to zero and resume counting from there. This generally will only occur on the Total Bytes Received and Total Bytes Sent counters.

- **Total Bytes Received**
  This is the count of all bytes in all packets received by the RMC whether directed to either the RMC's MAC address or the broadcast MAC address (FF FF FF FF FF FF). Packets discarded due to the Rx Errors described below are excluded from this count, but all other packets are included, even if discarded for other reasons. NOTE: When this counter reaches 4,294,967,296, it will wrap back to zero.

- **Total Valid Directed Pkts Rcvd**
  This is the count of all packets received by the RMC that were directed to the RMC's MAC address, and were processed. This does not include packets received but discarded due to protocol errors such as invalid protocol, IP address, or destination TCP/UDP port.

- **Total Valid Broadcast Pkts Rcvd**
  This is the count of all packets received by the RMC that were directed to the broadcast MAC address, and were processed.

- **Total Discarded Directed Pkts**
  This is the count of all packets received by the RMC that were directed to the RMC's MAC address, but were discarded due to invalid protocol, IP address, or TCP/UDP port.

- **Total Discarded Broadcast Pkts**
  This is the count of all packets received by the RMC that were directed to the broadcast MAC address, but were discarded due to invalid protocol, IP address, or TCP/UDP port.

- **Total Bytes Sent**
  This is the count of bytes sent by the RMC over the Ethernet. NOTE: When this counter reaches 4,294,967,296, it will wrap back to zero.

- **Total Directed Pkts Sent**
  This is the count of packets the RMC sent to a specific MAC address. This is typically done by the RMC only in response to a request from another device.

- **Total Broadcast Pkts Sent**
  This is the count of packets the RMC sent to the broadcast MAC address. This is typically done only when the RMC is requesting a BOOTP or DHCP server to respond with its IP address.

- **CPU Load % (Last) (requires 20010831 or newer RMC ENET firmware)**
  This value gives the percent of CPU time that is currently being used. The closer this number approaches 100, the slower it will respond to incoming requests.

- **CPU Load % (Max) (requires 20010831 or newer RMC ENET firmware)**
  This value holds the maximum CPU load percentage that has been reached since the RMC ENET was started.

Rx Errors (Receive Errors)

Occasional occurrences of the following errors are expected. However, it is important to ensure that you do not get a lot of these errors. Also, if you are experiencing network instability, check these counters for clues to what might be going wrong.

- **Missed Packets (Ethernet Controller)**
  This is the count of packets missed because the Ethernet controller chip's internal buffer filled up temporarily.

- **Missed Packets (RMC ENET)**
  This is the count of packets missed because the RMC ENET's internal buffer filled up temporarily. This is differentiated from Missed Packets (Ethernet Controller) for internal troubleshooting purposes.

- **Runts Received**
  This is a count of all packets received under the 64-byte minimum specified by the Ethernet IEEE 802.3 specification. This indicates a problem in other devices on the network and could result in network performance problems.

- **Extra Data Received**
  This is a count of all packets received over the 1518-byte maximum specified by the Ethernet IEEE 802.3 specification. This indicates a problem in other devices on the network and could result in network performance problems.

- **Bad CRC Received**
  This is a count of all packets received with a problem with their Frame Check Sequence (FCS), which is a 32-bit Cyclic Redundancy Check (CRC). This could indicate electrical noise or an error in another device on the network.

Tx Errors (Transmit Errors)

- **Single Tx Collisions (20010608 RMC ENET firmware or older)**
  This counts the number of times the RMC ENET sent a packet, had a collision, but succeeded on the first retry. This retry happens within 500ms from the collision. High numbers of single collisions will degrade network performance. The simplest solutions are (1) reducing the number of devices on the network, and (2) using an Ethernet switch instead of an Ethernet hub to reduce the size of the collision domain.
  In 20010724 RMC ENET firmware, this field was replaced by a collision histogram. See below.

- **Multiple Tx Collisions (20010608 RMC ENET firmware or older)**
  This counts the number of times the RMC ENET sent a packet and had more than one collision.

Therefore, it took more than one retry to send the packet. The RMC's Ethernet controller will give up trying to send the packet after 16 attempts.
In 20010724 RMC ENET firmware, this field was replaced by a collision histogram. See below.

- **Tx Retries**
  This is the total of all transmit retries due to collisions. Therefore it is the sum of all the Single Tx Collisions and the retries for each Multiple Tx Collision. For example, suppose there were 32 single-collisions transmits, 2 two-collision transmits, and 1 three-collision transmits . The Single Tx Collisions counter would hold 32, the Multiple Tx Collisions counter would hold 3, and the Tx Retries counter would hold 39 ([32 x 1] + [2 x 2] + [1 x 3]).

- **Jabber**
  This counts the number of times the RMC ENET transmitter erroneously transmits for longer than 26 ms. This condition should never occur. This is a safeguard from having the RMC's Ethernet controller jam the network.

- **Underrun**
  This counts the number of times the RMC ENET was sending a packet but stopped in the middle. This should never occur.

- **Late Collisions**
  This indicates the number of times the RMC ENET detected a collision after the first 512 bits have been transmitted. A late collision may indicate an illegal network configuration such as too great a distance between hubs or devices.

- **Tx Collisions (nx) (requires 20010831 RMC ENET firmware or newer)**
  Each of these sixteen numbers gives the number of packets that had exactly that many collisions. For example, the "1 collision" entry includes counts all frames that had exactly one collision, not one or more. This can be used to determine the longest delay incurred because of collisions.

5.2.4.5 RMC Ethernet Activity Log

The Ethernet Activity Log displays the recent events that occurred in the RMC-ENET module. Notice that it does not include events that occurred in the main RMC100 CPU module. This window can be displayed using the following steps:

1. On the main RMCWin window's Window menu, click Ethernet Activity Log. Notice that this command is only available for RMC modules with an ENET module.

Note: Using the Activity Log requires RMC ENET firmware version 20010831 or later.

This log displays events that occurred in the RMC-ENET module, one per line. The time that each event occurred is displayed in hours, minutes, seconds, and milliseconds relative to the time that the RMC-ENET module was started.

There are four buttons in this window:

- **Refresh**
  Click this button to re-read the Activity Log. It may take several seconds for the refresh to complete and the log to be updated.

- **Clear Log**
  Click this button to clear the activity log. Notice that clearing the log does not reset the timestamps for future events. That is, times for future events will still be stamped with the time since the RMC-ENET module was started, and not since the log was cleared.

- **Save**
  Save all the current activity data to a file.
- **Help**
  Display this topic in the online help.
- **Close**
  Close the Ethernet Activity Log window.

5.2.5 Ethernet Informational Topics

5.2.5.1 Understanding IP Addressing

IP Address

A fundamental part of setting up a TCP/IP network is setting up IP addresses. An IP address is a 32-bit number that is generally displayed in dotted decimal format, in which each octet (8 bits) of the address is displayed in decimal format, and each value is separated by period (e.g. 192.168.0.5). A less common, but often useful, way of displaying the address is in hexadecimal. The hexadecimal equivalent of 192.168.0.5 is C0A80005. Every computer on an intranet (one or more networks connected together) must have a unique IP address.

Subnet Mask

To facilitate communicating between multiple interconnected networks, the IP address is broken into two parts. One part is the network address, and the other part is the local address. Each network has a unique network address, and every device on that network has the same network address portion in its IP address. The local address uniquely identifies a computer within a network. It is expected that local addresses will be duplicated on different networks, but the entire IP address (network address + local address) is always unique.

The method for determining which portion of the IP address is the network address and which portion is the local address is to use a value called a subnet mask. A subnet mask is also a 32-bit number often displayed in dotted decimal format. Each bit of the subnet mask that is a 1 means that the corresponding bit of the IP address is part of the network address. Each bit of the subnet mask that is a 0 means that the corresponding bit of the IP address is part of the local address.

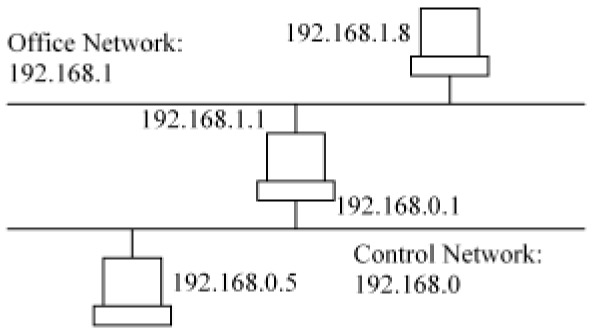
Example:

IP Address	192.168.0.5	`C0A80005`
Subnet Mask	255.255.255.0	`FFFFFF00`
Network Address	192.168.0	`C0A800`
Local Address	5	`05`

Therefore, from this example, we see that a device with an IP address of 192.168.0.5 and subnet mask of 255.255.255.0 will have a network address of 192.168.0 and a local address of 5. Other devices on this network must have the same network address but different local addresses. Therefore, some possible IP addresses for other nodes on the network include 192.168.0.6, 192.168.0.1, and 192.168.0.25. There are two reserved local addresses: a local address with all zero bits refers to the network (e.g. 192.168.0.0), and a local address with all one bits is the broadcast address for the network (e.g. 192.168.0.255).

Default Gateway

Suppose the device given in the above example must communicate with a device on a connected network with an IP address of 192.168.1.8. Because the device is not on the same network there is no electrical connection between the computers so it cannot send its data directly. Instead it must go through an IP router. An IP router is a device that sends packets it receives from one network that are intended for devices on another network to the other network. Here is the example intranet:



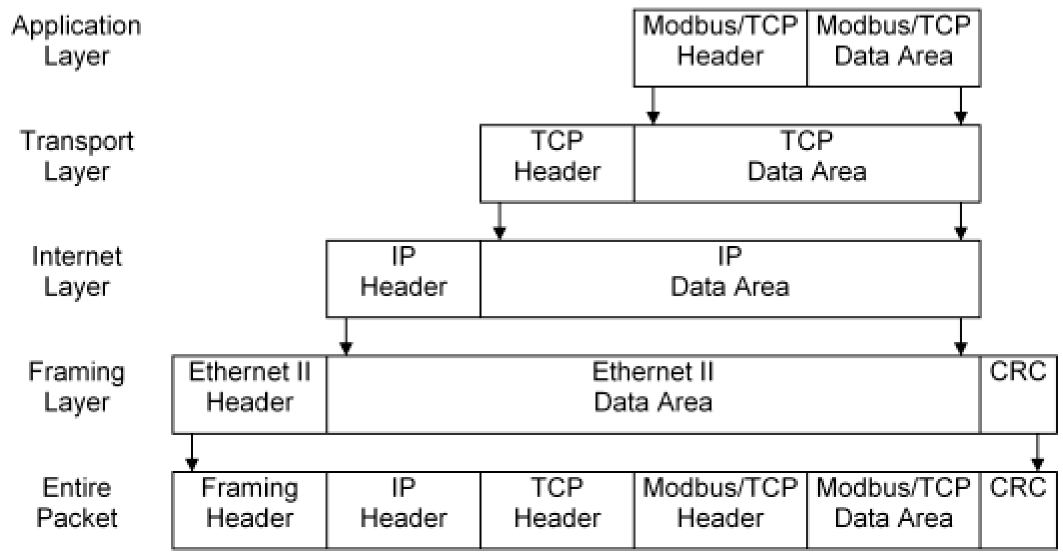
How does 192.168.0.5 send a message to 192.168.1.8? The answer is that it must use a third parameter called the default gateway. This parameter is the IP address of the router who will take care of getting the packet to its destination. The rule for most devices is to send packets to devices with the same network address directly over its network, but to send packets to devices with a different network address to the default gateway. In the above network, the device at 192.168.0.5 would have a default gateway of 192.168.0.1, and the device at 192.168.1.8 would have a default gateway of 192.168.1.1.

The default gateway parameter is optional if the device will be on a network that is not connected to any other networks, or if you have an intranet but do not want to allow the device to communicate with devices on networks other than its own.

5.2.5.2 RMC Ethernet Protocols

Note: The RMC Ethernet protocols are divided into two different firmware files. When updating the RMC100 Ethernet firmware, you must choose the file that contains the protocol you need. For more details, see Downloading New Serial/Ethernet Firmware.

Networking is often viewed conceptually as layers of protocols. Each layer contains a header, used to fulfill the purpose of that layer, and data. These layers are set up such that each layer contains the header and data from the next higher layer within its data area, as shown in the following diagram:

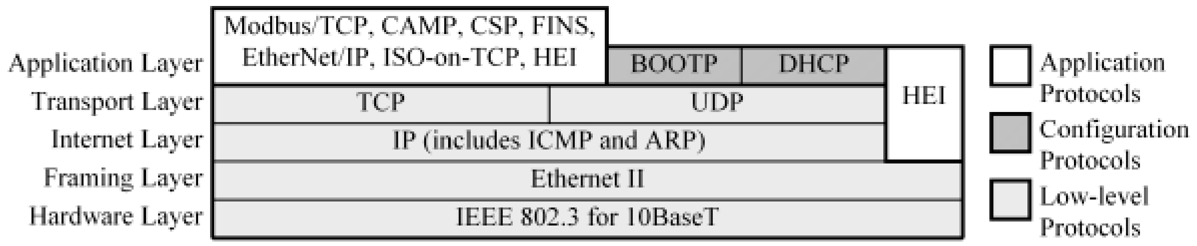


This diagram shows the four conceptual layers of TCP/IP: application, transport, internet, and framing. A fifth layer—the hardware layer—is often added below these four layers, but is left out of this diagram because it is more of a specification of how the data is sent rather than another protocol header. When a device is sending a packet the packet is assembled from the top layer down, but when receiving a packet, it must be processed from the bottom layer up.

Here is how the RMC might look at an incoming packet with this structure:

1. Hardware Layer: A full packet is received and passed to the Framing Layer.
2. Framing Layer: The CRC (cyclic redundancy check) is verified. If this fails, the packet is discarded. Next, the destination MAC address in the framing header is compared with the RMC's MAC address. If the addresses do not match and the destination address was not a special broadcast address, the packet is discarded. Otherwise it is passed to the Internet Layer.
3. Internet Layer: The IP address in the IP header is compared with the RMC's user-selectable IP address. If it does not match, the packet is discarded. Otherwise, it is passed to the Transport Layer.
4. Transport Layer: The transport layer provides a number of services, but minimally must specify the port that the data should be sent to. A port is an abstract connection point on a device that allows for multiple connections to exist on a single device. It also helps determine which application protocol will follow. The packet may be discarded here too if the destination port is not one that the RMC supports.
5. Application Layer: In our example, the application protocol is Modbus/TCP, so the Modbus/TCP header contains data such as the RMC register address to begin reading or writing from, the number of registers to access, and whether the operation is a read or write. The Modbus/TCP data area holds the actual words to be written.

Here is a diagram demonstrating all protocols supported by the RMC and the layers to which they belong:



Each protocol is briefly described below:

- **ARP (Address Resolution Protocol)**
  Ethernet packets can either be broadcast (received by all devices on the network) or sent to a single MAC address. However, applications generally address computers by IP address rather than MAC address. Therefore, this protocol is used to determine the MAC address of the computer owning a given IP address.

- **BOOTP (BOOTstrap Protocol)**
  This protocol is used to allow a central database to be maintained with all IP addresses on a network. This single computer is the BOOTP server. When a BOOTP client (such as the RMC, if configured to use BOOTP) starts up, it broadcasts asking a BOOTP server to tell it what its IP address should be. The BOOTP server looks up the MAC address of the BOOTP client in a database and sends a reply with the corresponding IP address.

- **CAMP (Common ASCII Message Protocol)**
  This is an open protocol developed by Control Technology, Inc. who manufacturers the CTI 2572 which allows the Siemens Simatic TI505 controller to communicate via Ethernet. Information is available on this protocol from CTI's web-site: http://www.controltechnology.com.

- **CSP (Client/Server Protocol)**
  This is a proprietary protocol developed by Allen-Bradley, Inc. Variants of this are used on Allen-Bradley's SLC 5/05 and PLC-5 controllers. This protocol is also used by the Allen-Bradley's SoftLogix 5 and RSLinx and SoftPLC Corporation's SoftPLC. Allen-Bradley does not publish the specifications for this protocol.

- **DHCP (Dynamic Host Configuration Protocol)**
  This protocol is an enhanced version of BOOTP. However, for industrial applications, the enhancements (lease times and dynamic assignment of IP addresses) are generally not usable. The RMC supports both BOOTP and DHCP so that the user may use either type of server.

- **Ethernet II**
  This is the most common framing layer protocol used by Ethernet devices. Other alternatives include IEEE 802.3 SNAP, IEEE 802.5 (Token Ring), and IEEE 802.4 (Token Bus). The RMC only supports Ethernet II framing, and therefore will not work on networks using any of the other framing types. All PLCs currently support Ethernet II framing although the Modicon Quantum allows selecting either Ethernet II (the default) or IEEE 802.3 SNAP.

- **EtherNet/IP**
  This is an open application protocol, maintained by ODVA (http://www.odva.org). EtherNet/IP is used by Ethernet modules for several PLC's including Allen Bradley, Schneider Electric, and Omron. The RMC100 supports EtherNet/IP Messaging and EtherNet/IP cyclic I/O.

- **FINS**
  This is an open application protocol developed and used by Omron Electronics Inc. This protocol is available over a number of media, including Ethernet and serial. Additional information is available in the CS1 Communications Reference Manual, available on Omron's web site: http://www.omron.com/oei.

- **HEI (Host Engineering Inc)**
  This is a proprietary protocol controlled by Host Engineering Inc (http://www.hosteng.com). This application protocol can be run on top of UDP/IP, IPX, and Ethernet II. The RMC can respond to HEI requests over UDP/IP and Ethernet II.

- **ICMP (Internet Control Message Protocol)**
  This protocol—running on top of the Internet Protocol—is used for sending error messages between routers and also provides the ping service. Only the ping service portion of this protocol is used by the RMC. Ping is a utility provided on most operating systems that simply asks if a device can be found with a given IP address or name.

- **IEEE 802.3 for 10BaseT**
  This is a standard for sending Ethernet data at 10Mbps through twisted pair wiring rated Category 3 (CAT3) or greater. The connectors used by this standard are RJ45. Media converters are readily available from network supply companies to convert these signals to other IEEE 802.3 standards such as 10Base2 (BNC or coaxial cable), 10BaseFL (fiber optic), 10Base5 (AUI), and 100BaseT (100Mbps twisted pair using CAT5 wiring).

- **IP (Internet Protocol)**
  This is the main Internet Layer protocol and is used for sending packets between two computers in a network or across two or more networks. It also allows for fragmentation of packets. This is a situation where a packet is larger than a network's maximum packet size, so it is broken into smaller packets that are re-assembled by the recipient.

- **ISO-on-TCP**
  This application protocol was designed to allow using protocols based on the ISO 8073 Transport Protocol to work on the TCP/IP stack of protocols. This protocol is used by the Siemens S7-300 and S7-400 PLCs. IEEE maintains this protocol. The RMC supports a protocol on top on the ISO-on-TCP protocol called Fetch/Write that is also supported by the S7 PLCs.

- **Modbus/TCP**
  This is an open protocol developed and used by Modicon of Schneider Electric. Its standard is published on Modicon's web-site: http://www.modicon.com/openmbus.

- **TCP (Transmission Control Protocol)**
  This is the main Transport Layer protocol in the TCP/IP stack. It provides for maintaining a reliable connection between two devices.

- **UDP (User Datagram Protocol)**
  This Transport Layer protocol is used by some application protocols instead of TCP. UDP is lighter weight than TCP and works well in situations where acknowledgements are built into the application protocol or are not required.

5.2.6 Controlling and Monitoring the RMC over Ethernet

5.2.6.1 Allen-Bradley Controllers

5.2.6.1.1 Using Allen-Bradley Controllers with the RMC Ethernet Module

Allen-Bradley has several Ethernet options for its PLCs. The SLC 5/05 and Ethernet PLC-5E controllers each have a built-in Ethernet port. The ControlLogix uses the 1756-ENET or 1756-ENBT communication module to use Ethernet, and PLC-5 controllers other than the PLC-5E can have Ethernet added using a PLC-5 Ethernet Interface Module. Allen-Bradley also sells a PC-based controller called the SoftLogix 5, which can also communicate with the RMC.

Note: Ethernet communication with the ControlLogix Ethernet (e.g. 1756-ENET and 1756-ENBT) modules requires RMC ENET firmware dated 20000420 or later.

In addition to Allen-Bradley's controllers, SoftPLC Corporation manufactures a PC-based, PLC-5 compatible controller that can also communicate with the RMC. See Using the SoftPLC with the RMC Ethernet Module for details.

All of these PLCs can the same ladder logic block to communicate over the Ethernet: the Message (MSG) block. This block takes a number of parameters, which are briefly described below. For a complete description of the parameters, refer to Allen-Bradley's Instruction Set Reference Manual for the appropriate PLC.

Note: With the introduction of the 1756-ENET/B and 1756-ENBT/A, the ControlLogix now also supports I/O connections through EtherNet/IP, which greatly improves performance and determinism. When using the ControlLogix, you will most likely want to use both the MSG block and I/O. EtherNet/IP is discussed in detail in a number of topics starting with Using EtherNet/IP with the RMC ENET.

The Allen-Bradley PLCs can read or write from registers in compatible remote devices such as other Allen-Bradley PLCs or the RMC. The RMC emulates a PLC-5 with 248 integer files (N7 and N8-N255) with 256 elements each (0-255), all of which are accessible over the Ethernet from the Allen-Bradley PLCs. See the RMC Register Map (Allen-Bradley) for details on those registers and their addresses.

Note: Although the RMC Register Map is spread out over 248 integer files, reads and writes that extend beyond the end of an RMC register file will continue into the next file or files. This is particularly useful on the ControlLogix, which allows reading large amounts of data with a single MSG block. By reading 2048 integers from N9:0 in the RMC, the entire Event Step table can be read into the ControlLogix, even though it uses N9 through N17.

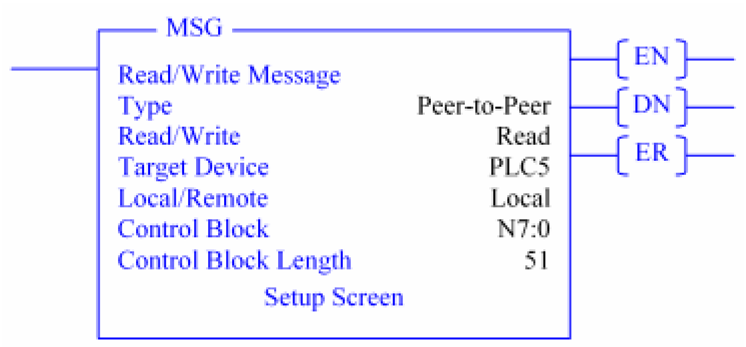
If you need help setting up your network, either consult your network administrator, or for simple stand-alone networks, see Setting up a Stand-alone TCP/IP Control Network.

MSG parameters:

The MSG block has slightly different parameters depending on the controller and programming software you are using. The parameters used by RSLogix 5 version 3.2.0.0, RSLogix 500 version 3.01.02.00, and RSLogix 5000 version 2.27.00 for the PLC-5, SLC 5/05, and ControlLogix controllers respectively are described below. The SoftLogix 5 parameters are similar.

SLC 5/05 MSG Block Parameters:

The SLC 5/05 MSG block is displayed as the following:



MSG

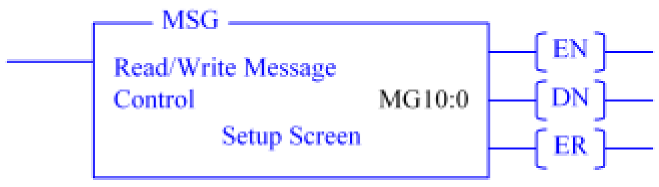
Read/Write Message	
Type	Peer-to-Peer
Read/Write	Read
Target Device	PLC5
Local/Remote	Local
Control Block	N7:0
Control Block Length	51

- **Type:** This parameter is always set to Peer-To-Peer for Ethernet communication channels.
- **Read/Write:** This parameter should be set to **Read** to read registers from the RMC, and to **Write** to write registers to the RMC.
- **Target Device:** This parameter has possible values of 500CPU, 485CIF, and PLC5. This should be set to PLC5 for communicating with the RMC.
- **Local/Remote:** This parameter has possible values of Local and Remote. It should be set to Local for communicating with the RMC.
- **Control Block:** This parameter points to a block of 51 integer-file registers. Set this to a block of registers, and then use the Setup Screen option in the MSG ladder logic block to modify those register values:

- **This Controller:** This section holds parameters for the SLC 5/05.
    - **Communication Command:** This parameter will be set to either PLC5 Read or PLC5 Write. It is not changed from within the Command Block; it is changed in the MSG block itself.
    - **Data Table Address:** Enter the address of the first Allen-Bradley PLC register to read RMC registers into, or to write to RMC registers from.
    - **Size in Elements:** Enter the number of RMC registers to read or write in this field. The range enforced by the SLC is 1 to 256 integers. Reads or writes that extend beyond the end of a register file will continue into the next register file. For example, reading 256 elements from N9:128 will read N9:128 to N9:255, then N10:0 to N10:127.
    - **Channel:** Set this to the channel number of the Ethernet channel. For the SLC 5/05, this should be channel #1.
- **Target Device:** This section holds parameters for the target device.
    - **Message Timeout:** Indicate the number of seconds to wait for the RMC to respond before determining that the attempt failed. This can be set as low as a few seconds.
    - **Data Table Address:** Enter the address of the first RMC register to read or write in this field. See the RMC Register Map (Allen-Bradley) for help on addresses.
    - **Ethernet (IP) address:** Set this to the IP address of the RMC you wish to communicate with.
    - **MultiHop:** This parameter should be set to **No**.

PLC-5 MSG Block Parameters:

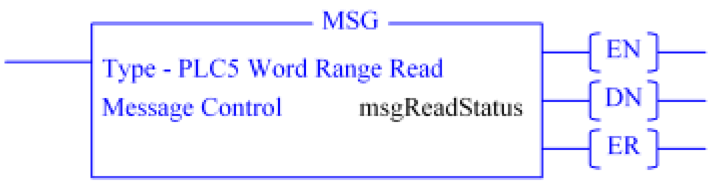
The PLC-5 MSG block is displayed as follows:



- **Control:** This parameter points to a block of 51 N-file (integer) registers or two (2) MG-file (message) registers. Set this to an unused block of registers, and then use the **Setup Screen** option in the MSG ladder logic block to modify those register values:
- **This PLC-5:** This section holds parameters for the PLC-5.
  - **Communication Command:** From this drop-down list, select **PLC-5 Typed Read** to read values from the RMC, or **PLC-5 Typed Write** to write values to the RMC.
  - **Data Table Address:** Enter the address of the first Allen-Bradley PLC register to read RMC registers into, or to write to RMC registers from.
  - **Size in Elements:** Enter the number of RMC registers to read or write in this field. Transfers are limited to 1000 bytes for PLC-5 Typed Reads and Writes. Therefore, this limit is 500 integers, 250 floats, etc. Notice that this limit is larger than the number of elements in the RMC's N-files. Reads or writes that extend beyond the end of a register file will continue into the next register file. For example, reading 300 elements from N9:0 will read N9:0 to N9:255, then N10:0 to N10:43.
  - **Port Number:** Set this to the Ethernet channel number. For the PLC-5, this should be channel #2.
- **Target Device:** This section holds parameters for the target device.
  - **Data Table Address:** Enter the address of the first RMC register to read or write in this field. See the RMC Register Map (Allen-Bradley) for help on addresses.
  - **MultiHop:** This parameter should be set to **No**.
  - **Ethernet (IP) address:** Set this to the IP address of the RMC you wish to communicate with.

ControlLogix MSG Block Parameters:

The ControlLogix MSG block is displayed as follows:



To edit the parameters of the message block, select the MSG block, and click on the button with the ellipses to the right of the message tag name (msgReadStatus in the example above).

This will bring up a dialog with two tabs. Each is described below:

- **Configuration** tab:
  - **Message Type:** From this drop-down list, select **PLC5 Word Range Read** to read values from the RMC, or **PLC5 Word Range Write** to write values to the RMC.
  - **Source Element (reads only):** Enter the address of the first RMC register you want to read. See the RMC Register Map (Allen-Bradley) for help on addresses.

- **Source Tag (writes only):** Enter the tag in the ControlLogix that you want to send to the RMC. In most cases, the tag should be either an array or structure composed entirely of INT data types. This is because both the RMC registers and the INT data type hold 16-bit binary numbers. One issue to keep in mind is that the INT data types are always treated like signed numbers with a range of -32768 to 32767, while the RMC registers are sometimes signed, sometimes unsigned (0 to 65,535) and sometimes somewhere in between.
- **Number of Elements:** Enter the number of RMC registers to read or write in this field. You can transfer from 1-32767 registers at a time per MSG block. Reads or writes that extend beyond the end of a register file will continue into the next register file or files. For example, reading 2048 elements from N9:0 will read N9:0 to N9:255, then N10:0 to N10:255, on through N17:255.
- **Destination Tag (reads only):** Enter the tag in the ControlLogix into which you want to read the RMC data. See the **Source Tag** bullet above for suggestions on the data type of the tag.
- **Destination Element (writes only):** Enter the address of the first RMC register you want to write. See the RMC Register Map (Allen-Bradley) for help on addresses.

- **Communication tab:**

- **Path:** You must enter the path from the Logix5550 CPU to the RMC. This path must, at the minimum, include going over the rack to the 1756-ENET card and then going over the Ethernet to the RMC. Therefore, the format of the path usually is as follows:

  1, [1756-ENET slot number], 2, [RMC IP Address]

  Therefore, if the 1756-ENET is in slot 2 (the third slot on the rack), and the RMC is at address 192.168.0.5, then the following path would be used:

  1, 2, 2, 192.168.0.5

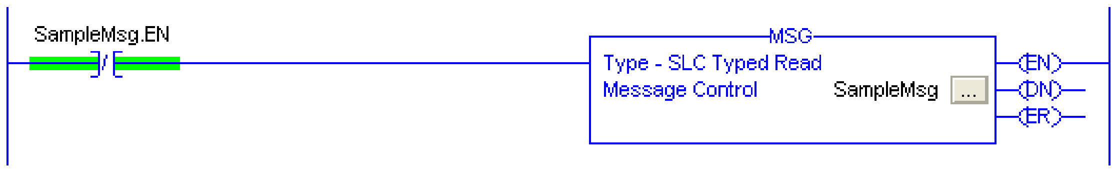
- **Communication Method:** Select the **CIP** option.
- **Cache Connections:** This checkbox is not used for the message type selected.

Using the MSG Block in Ladder Logic

The Allen-Bradley MSG block takes multiple ladder scans to complete. Therefore, it is important to enable the MSG block for the correct amount of time. Specifically, the MSG block must be energized until the message control's enable (EN) bit turns on. Delta has found some aspects of this to be difficult and therefore has provided the following ladder samples:

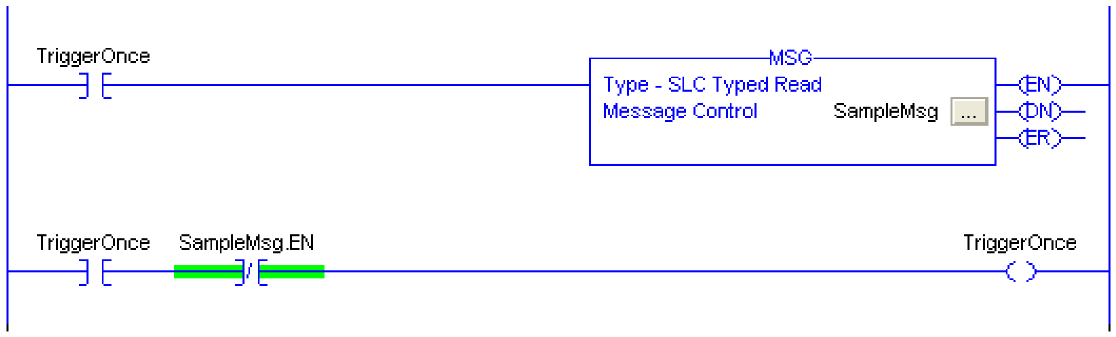
Read or Write Continuously

Using the Examine If Open instruction as shown below fulfills two requirements of continuous MSG transactions. First, it will keep the block energized until the EN turns on, and second, it de-energizes the MSG block once the transactions is started so that when the transaction is completed (EN goes low again), the MSG block sees a rising edge on its input, thus repeating the transaction:



Read or Write Once

This sample takes care to keep the MSG block energized until the MSG block starts, as indicated by the enable (EN) bit turning on. Once this happens, the application-controlled TriggerOnce coil is turned off. The message control's Done (DN) or Error (ER) bits can be used to process the results of the transaction.



5.2.6.1.2 RMC Register Map (Allen-Bradley)

Tip: RMCWin's Address Tool provides an easy way to identify addresses in the RMC. Simply open the Address Tool and then move the cursor to any field in RMCWin that represents an RMC Register, and the Address Tool will display the address in the address format of your choice. See Address Tool for details.

The RMC module has 64K (65536) 16-bit registers that can be read from or written to over Ethernet, Serial, Modbus Plus, and PROFIBUS-DP. Each register is assigned an address. However, under the different communication methods, different addressing schemes are used. This topic describes using Allen-Bradley PLC addressing. For details on addressing from other modules refer to the following topics:

- RMC Register Map (Automationdirect.com)
- RMC Register Map (Modbus/TCP and Modbus/RTU)
- RMC Register Map (Omron FINS)
- RMC Register Map (Siemens TI505)
- RMC Register Map (Siemens S7)
- RMC Register Map (Modbus Plus)
- RMC Register Map (PROFIBUS-DP Message Mode)

Allen-Bradley offers several Ethernet and serial solutions for its ControlLogix, SLC, PLC-5, and SoftLogix 5 controllers. In addition, SoftPLC emulates the PLC-5 and therefore also uses Allen-Bradley's Ethernet protocol. Over this protocol, the RMC's registers are broken into a number of integer files. Each integer file used (N7, N9-N18, L19, N20-N255) is configured to be the maximum size allowed on a SLC 5/05 file: 256 elements: Nf:0-255. For details on reading and writing these registers, see the following topics:

- Using Allen-Bradley Controllers with the RMC ENET
- Using SoftPLC's SoftPLC with the RMC ENET
- Using DF1 (Full- and Half-Duplex) with the RMC SERIAL

Status Registers:
These registers can only be read; writes are ignored.

Allen-Bradley and SoftPLC	**Register Description**
N7:0	Axis 0 Command Position
N7:1	Axis 0 Target Position
N7:2	Axis 0 Actual Position
N7:3	Axis 0 Transducer Counts. **Note:** See the Transducer Counts (32-bit) Registers section below for a way to read 32-bit counts instead of 16-bit counts.
N7:4	Axis 0 Status Word
N7:5	Axis 0 Drive
N7:6	Axis 0 Actual Speed
N7:7	Axis 0 Null Drive
N7:8	Axis 0 Event Step
N7:9	Axis 0 Link Value
N7:10-19	Same as above but for axis 1
N7:20-29	Same as above but for axis 2
N7:30-39	Same as above but for axis 3
N7:40-49	Same as above but for axis 4
N7:50-59	Same as above but for axis 5

N7:60-69	Same as above but for axis 6
N7:70-79	Same as above but for axis 7

Command Registers:
These registers can be read or written.

Allen-Bradley and SoftPLC	Register Description
N7:80	Axis 0 Mode Word
N7:81	Axis 0 Acceleration
N7:82	Axis 0 Deceleration
N7:83	Axis 0 Speed
N7:84	Axis 0 Command Value
N7:85	Axis 0 Command
N7:86-91	Same as above but for axis 1
N7:92-97	Same as above but for axis 2
N7:98-103	Same as above but for axis 3
N7:104-109	Same as above but for axis 4
N7:110-115	Same as above but for axis 5
N7:116-121	Same as above but for axis 6
N7:122-127	Same as above but for axis 7

Parameter Registers:
These registers can be read or written. Changes to these registers do not take effect until a Set Parameters (P) command is executed.

Allen-Bradley and SoftPLC	Register Description
N7:128	Axis 0 Configuration Word
N7:129	Axis 0 Scale
N7:130	Axis 0 Offset

N7:131	Axis 0 Extend Limit
N7:132	Axis 0 Retract Limit
N7:133	Axis 0 Proportional Gain
N7:134	Axis 0 Integral Gain
N7:135	Axis 0 Differential Gain
N7:136	Axis 0 Extend Feed Forward
N7:137	Axis 0 Retract Feed Forward
N7:138	Axis 0 Extend Acceleration Feed Forward
N7:139	Axis 0 Retract Acceleration Feed Forward
N7:140	Axis 0 Dead Band Eliminator
N7:141	Axis 0 In Position Window
N7:142	Axis 0 Following Error
N7:143	Axis 0 Auto Stop
N7:144-159	Same as above but for axis 1
N7:160-175	Same as above but for axis 2
N7:176-191	Same as above but for axis 3
N7:192-207	Same as above but for axis 4
N7:208-223	Same as above but for axis 5
N7:224-239	Same as above but for axis 6
N7:240-255	Same as above but for axis 7

Event Step Table Registers:

These registers can be read or written. When using the Allen-Bradley addressing scheme with these registers, you must keep in mind that the Event Step Table is split over eight register files (this is done because the SLC 5/05 only supports 256 words per file). Use the following table to determine the register file for a given event step:

Event Step (n)	**Register File (f)**	**Step Offset (r)**
0-31	N9	(n - 0) x 8
32-63	N10	(n - 32) x 8

64-95	N11	(n - 64) x 8
96-127	N12	(n - 96) x 8
128-159	N13	(n - 128) x 8
160-191	N14	(n - 160) x 8
192-223	N15	(n - 192) x 8
224-255	N16	(n - 224) x 8

Note: On Allen-Bradley PLCs, reads and writes that extend beyond the end of an RMC register file will continue into the next file or files. This is particularly useful on the ControlLogix, which allows reading large amounts of data with a single MSG block. For example, by reading 2048 integers starting at N9:0 in the RMC, the entire Event Step table can be read into the ControlLogix.

The register map for addressing the fields in the event step table is as follows:

Allen-Bradley and SoftPLC	Register Description
N9:0	Step 0 Mode Word
N9:1	Step 0 Acceleration
N9:2	Step 0 Deceleration
N9:3	Step 0 Speed
N9:4	Step 0 Command Value
N9:5	Step 0 Command/Commanded Axes
N9:6	Step 0 Link Type/Link Next
N9:7	Step 0 Link Value
Nf:r + 0	Step n (0-255) Mode Word
Nf:r + 1	Step n (0-255) Acceleration
Nf:r + 2	Step n (0-255) Deceleration
Nf:r + 3	Step n (0-255) Speed
Nf:r + 4	Step n (0-255) Command Value
Nf:r + 5	Step n (0-255) Command/Commanded Axes
Nf:r + 6	Step n (0-255) Link Type/Link Next

Nf:r + 7	Step n (0-255) Link Value

Input to Event Table Registers:
These registers can be read or written.

Allen-Bradley and SoftPLC	Register Description
N17:0	Event Step for Axis 0 on Input 0 Rising Edge
N17:1	Event Step for Axis 1 on Input 0 Rising Edge
N17:2	Event Step for Axis 2 on Input 0 Rising Edge
N17:3	Event Step for Axis 3 on Input 0 Rising Edge
N17:4	Event Step for Axis 4 on Input 0 Rising Edge
N17:5	Event Step for Axis 5 on Input 0 Rising Edge
N17:6	Event Step for Axis 6 on Input 0 Rising Edge
N17:7	Event Step for Axis 7 on Input 0 Rising Edge
N17:8 + n	Event Step for Axes n (0-7) on Input 1 Rising Edge
	:
N17:120 + n	Event Step for Axes n (0-7) on Input 15 Rising Edge
N17:128 + n	Event Step for Axes n (0-7) on Input 0 Falling Edge
	:
N17:248 + n	Event Step for Axes n (0-7) on Input 15 Falling Edge

Status Map Registers:
This block of registers is only used by the Modbus Plus and PROFIBUS interfaces. Therefore, these registers are unused by this Ethernet protocol.

Allen-Bradley and SoftPLC	Register Description
N18:0-63	Status Map Entries

Plot Type Registers:

The plot type registers can be read or written. The values that are read indicate the extra plot information in the current graph. The values written to these registers tell the controller which extra plot information to obtain on the next plot. For these registers, the following values are used:

- 0: Extra position precision
- 1: Command and Command Value
- 2: Event Step and Link Value
- 3: Raw Transducer Counts
- 4: Internal Target and Actual Speeds
- 5: Integral Drive

For more information on these four types of plot information, see Selecting the Data to Plot and Reading Plots from the Communication Module.

Allen-Bradley and SoftPLC	**Register Description**
N18:64	Axis 0 plot type
N18:65	Axis 1 plot type
N18:66	Axis 2 plot type
N18:67	Axis 3 plot type
N18:68	Axis 4 plot type
N18:69	Axis 5 plot type
N18:70	Axis 6 plot type
N18:71	Axis 7 plot type

Digital (Discrete) I/O Registers:

These registers indicate the current state of the digital inputs and outputs. These registers may only be read; writes will be ignored, as this product does not support forcing inputs or outputs.

Because different PLCs label bit numbers differently, the following chart is provided to show the mapping between the devices:

	MSB								
RMC bit #	15	14	13	12	11	10	9	8	7
Allen-Bradley bit #	15	14	13	12	11	10	9	8	7

The bit numbers listed in the table below are in RMC format (0 is LSB, 15 is MSB):

Allen-

Bradley and SoftPLC	**Register Description**
N18:72	CPU Digital Inputs 0 and 1 in LSBs of low byte, Outputs 0 and 1 in LSBs of high byte
N18:73	Unused
N18:74	Unused
N18:75	Sensor Digital I/O Inputs 0-15
N18:76	Sensor Digital I/O Inputs 16-17 (stored to two LSBs)
N18:77	Sensor Digital I/O Outputs 0-7 in high byte (low byte unused)
N18:78	Unused
N18:79	Unused

Plot Time Registers:

The Plot Time interval is configurable on the RMC. This interval indicates the number of control loops between each sample in a plot. Therefore, if the control loop is 0.976ms (e.g. RMC100-M1), this value indicates roughly the number of milliseconds between samples. If the control loop is 1.953ms (e.g. RMC100-M4), this indicates half of the number of milliseconds between samples.

These registers may be read or written. When read, they indicate the plot interval of the currently gathered plot. When written, they set the plot interval the RMC should use for the next plot it will gather. When the RMC starts the next plot, it copies the requested plot interval into the currently used plot interval.

Allen-Bradley and SoftPLC	**Register Description**
N18:80	Axis 0 plot time interval
N18:81	Axis 1 plot time interval
N18:82	Axis 2 plot time interval
N18:83	Axis 3 plot time interval
N18:84	Axis 4 plot time interval
N18:85	Axis 5 plot time interval
N18:86	Axis 6 plot time interval
N18:87	Axis 7 plot time interval

Last Parameter Error Registers:

Note: To use these registers through Ethernet, you must have RMC100 CPU control firmware dated 19990715 or later and Ethernet firmware dated 19990702 or later.

Each of these read-only registers holds the number of the last parameter error generated on an axis. This is useful for determining the cause of the Parameter Error bit in the status word. For a description of the values read from these registers, see Parameter Error Values.

Allen-Bradley and SoftPLC	Register Description
N18:88	Last parameter error on axis 0
N18:89	Last parameter error on axis 1
N18:90	Last parameter error on axis 2
N18:91	Last parameter error on axis 3
N18:92	Last parameter error on axis 4
N18:93	Last parameter error on axis 5
N18:94	Last parameter error on axis 6
N18:95	Last parameter error on axis 7

Firmware Date Registers:

Note: To use these registers through Ethernet, you must have RMC100 CPU control firmware dated 19990715 or later and Ethernet firmware dated 19990702 or later.

These read-only registers hold information about the firmware versions in the RMC100 CPU module. The Boot and Loader firmware versions have no effect on the actual performance of the RMC and therefore can usually be ignored.

Allen-Bradley and SoftPLC	Register Description
N18:96	Boot firmware month (MSB) and day (LSB)
N18:97	Boot firmware year
N18:98	Loader firmware month (MSB) and day (LSB)
N18:99	Loader firmware year
N18:100	Control firmware month (MSB) and day (LSB)
N18:101	Control firmware year
N18:102	Control firmware Beta Code. This will be 0 for standard release firmware, 'B' for Beta

	firmware, or 'SI' for Superimposed firmware.
N18:103	Feature code. This register is mainly reserved for internal use but does have two bits that may be useful to some users: • If bit 1 (value 0x0002) is set, the control loop is 2 ms, otherwise the control loop is 1 ms. • If bit 0 (value 0x0001) is set, a sensor DI/O is present, otherwise there is no sensor DI/O.

Transducer Counts (32-bit) Registers:

Note: To use these registers through Ethernet, you must have RMC Ethernet firmware dated 20020115 or later.

Each of these read-only registers holds the transducer counts for an axis. Notice that these registers are 32-bit registers. The low 16 bits will match the Counts register read from registers N7:3, N7:13, etc. However, when the transducer counts go below zero (for incremental transducers) or above 65,535 counts, some information is lost. By using these registers, all bits of the counts are available.

Allen-Bradley and SoftPLC	Register Description
L19:0	32-bit Transducer Counts for axis 0
L19:1	32-bit Transducer Counts for axis 1
L19:2	32-bit Transducer Counts for axis 2
L19:3	32-bit Transducer Counts for axis 3
L19:4	32-bit Transducer Counts for axis 4
L19:5	32-bit Transducer Counts for axis 5
L19:6	32-bit Transducer Counts for axis 6
L19:7	32-bit Transducer Counts for axis 7

Reserved Registers:

Reading these values will return zero, and writes are ignored.

Allen-Bradley and SoftPLC	Register Description
N20:0-N47:255	Unused

Spline Download Area:

These registers are write only. Reading them will return zero. This area is used to download intervals and points in a spline. This is a much more efficient alternative to using individual New Spline Point and Set Spline Interval/End Segment commands. For details on using this Spline Download Area, see Downloading Splines to the RMC.

Allen-Bradley and SoftPLC	Register Description
N48:0-N63:255	Spline Download Area

Plot Registers:

These registers can only be read; writes are ignored.

Note: Reading plots is not a trivial task; for further details, see Reading Plots from the Communication Module.

Allen-Bradley and SoftPLC	Register Description
N64:0-N87:255	Plot data for axis 0
N88:0-N111:255	Plot data for axis 1
N112:0-N135:255	Plot data for axis 2
N136:0-N159:255	Plot data for axis 3
N160:0-N183:255	Plot data for axis 4
N184:0-N207:255	Plot data for axis 5
N208:0-N231:255	Plot data for axis 6
N232:0-N255:255	Plot data for axis 7

5.2.6.1.3 Using EtherNet/IP with the ControlLogix

The ControlLogix PLCs support EtherNet/IP through the 1756-ENET/B,1756-ENBT/A and other communication modules. EtherNet/IP combines the traditional messaging services with higher-performance and more deterministic I/O services. Messaging can be used to send arbitrary data between two devices using a request/response protocol. I/O is used to set up a specialized high-speed connection between two or more devices using the producer consumer model.

Messaging through EtherNet/IP uses the standard Message (MSG) block, as described in Using Allen-Bradley Controllers with the RMC Ethernet Module. I/O connections are described in detail over a number of topics starting with Using EtherNet/IP with the RMC ENET. Refer to these topics for more details.

5.2.6.2 Automationdirect.com's DL205/305

5.2.6.2.1 Using Automationdirect.com PLCs with the RMC ENET

DirectLogic PLCs from Automationdirect.com (formerly PLC Direct) can be used to control the RMC over Ethernet. Currently two families of DirectLogic PLCs have Ethernet modules available to them: the DL205 and the DL405. The DL205 requires the H2-ECOM Ethernet module, while the DL405 requires the H4-ECOM. Both Ethernet modules are also sold through Automationdirect.com.

Note: Ethernet communication with the DirectLogic PLCs and ECOM modules requires RMC ENET firmware dated 20000913 or later.

As of this writing, the DL205 family includes the D2-230, D2-240, and D2-250 CPUs. The DL405 family includes the D4-430, D4-440, and D4-450 CPUs. The following chart lists the limitations of the various CPUs when used with the RMC and Ethernet.

Family	CPU	Limitations
DL205	D2-230	No Ethernet support. Cannot be used with the RMC ENET.
	D2-240	Severely limited support. Review Register Map before using.
	D2-250	Cannot use Plots.
DL405	D4-430	Cannot use Plots and the Spline Download Area.
	D4-440	Cannot use Plots.
	D4-450	Cannot use Plots.

Note: The documentation below assumes the user to be familiar with DirectLogic PLC programming, and instead focuses on how to initiate reads and writes to an RMC.

The standard method of initiating data transfers from a DirectLogic 205/405 PLC is to use the RX and WX instructions. These instructions send data over raw Ethernet packets without using TCP/IP at all. This results in better performance, but limits the PLC-to-PLC communication to a single network (packets can not be routed to other networks or the Internet). Because IP is not used, IP addresses cannot be used. In their place, each device has a unique Module ID ranging from 1 to 90.

Setting the RMC Module ID

Automationdirect.com ships a setup program called NetEdit with the ECOM manual. It can also be downloaded from their supplier's site: http://www.hosteng.com. This program can be used to set the Module ID of any ECOM or RMC module.

To set the RMC Module ID:

1. Ensure that your PC has TCP/IP installed.
2. Connect your PC and the RMC to the same network. Generally this involves plugging each into the same Ethernet switch or hub.
3. Start NetEdit on your PC.
4. Under **Protocol**, select **UDP/IP**. The program will take approximately 1 second to scan the network for devices.
5. You should now see one or more devices in the **Module** list. The devices beginning with **00 50 A0** are RMCs. Devices beginning with **00 E0 62** are most likely ECOM modules.
6. Click the RMC in the **Module** list. The rest of the NetEdit application should update to hold the information for the RMC. The Name will default to **Delta RMC**, even though the **Type** under **Module Information** will be **H2-ECOM** for compatibility.
7. In the **Module ID** text box under **Configuration,** enter the Module ID you wish to use. This ID must be unique for all devices on your network. This value must be between 1 and 90 to be usable by other PLCs.
8. If you wish, you can also set your IP Address and IP Netmask (Subnet Mask) fields at this time. They are not required to be set up to use the RMC with the ECOM modules.

Note: The Name and Description fields in the RMC can be changed, but because they serve no purpose for the RMC, they will return to their default values when power is cycled.

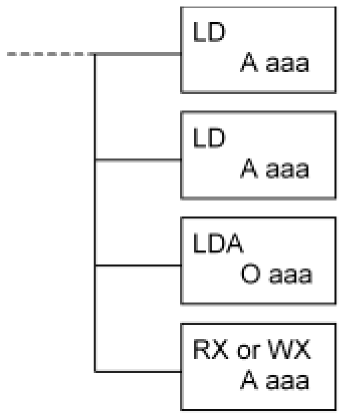
9. Click **Update**. This will save the Module ID, IP Address, and Subnet Mask in the RMC Flash memory. Do not power down the RMC until the RMC's CPU LED stops blinking (this takes approximately seven seconds).

For further details on using NetEdit, see its documentation in the ECOM manual from Automationdirect.com.

Reading and Writing RMC Data from the DL205/405

The ECOM manual describes reading and writing data from other PLCs in **Chapter 3: RLL Programming for Communications** of its original version (4/98). Use the exact same procedure for reading and writing from the RMC; the RMC emulates an H2-ECOM. Refer to the RMC Register Map (Automationdirect.com) topic for a list of addresses in the RMC that can be read or written.

The following is the format of the RX and WX routines:



The first three instructions load three values onto the accumulator stack, and the RX or WX instruction takes these values off the accumulator stack and uses them for preparing the data to send over the network. Therefore, you have some flexibility as to how to put the values on the accumulator stack—for example, in the LD instructions you can use K to specify constants or V to specify a register holding the value to put on the stack. For the most part these additional methods are not described in this manual but can be found in the DL205 PLC User Manual and DL405 PLC User Manual.

Note: Recall that DirectLogic V-memory addresses are given in octal. Refer to the DL205/405 PLC User Manual for a description of octal.

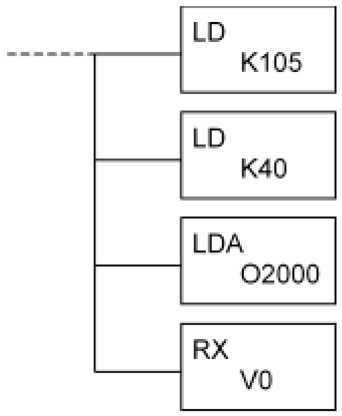
Here is a description of the values placed on the stack:

Value	Description
First (LD)	This 16-bit BCD value is divided into three fields. The first digit indicates the base number. In most cases this must be 0 to indicate the CPU base. The second digit indicates the ECOM slot number. Slots start with 0 for the first slot to the right of the CPU and increase by one for each slot to the right. The last two digits indicate the Module ID of the remote device. Enter the RMC's Module ID here.
Second (LD)	This 16-bit BCD value indicates the number of bytes to read or write. For RMC registers, this value must be an even number between 2 and 128, which represent 1 to 64 words.
Third (LDA)	This indicates the location in the local PLC to read the data into, and write data out from. The address is a V-memory address, even though the prefix is O (for octal) instead of V for V-memory.
Fourth (RX or WX)	The parameter on the RX or WX instruction indicates the location in the remote PLC (the RMC in our case). See RMC Register Map (Automationdirect.com) for a map of addresses. Notice that V0-V177 and V1000-V1177 may be displayed as TA0-TA177 and CTA0-CTA177, which are called aliases. You can disable displaying aliases by selecting the **View** menu in

DirectSOFT32, **Options** menu item, **Global** tab, and clearing the **Display Aliases** checkbox.

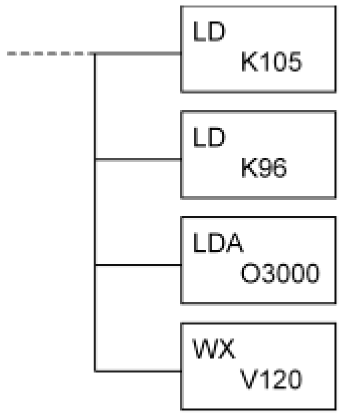
Example 1

The user has an ECOM in slot 1 of the CPU base and an RMC with Module ID 5. The user wants to read all ten status words for each of the first two axes of the RMC and store those twenty words into V2000-V2023. This is done as follows:



Example 2

The user has an ECOM in slot 1 of the CPU base and an RMC with Module ID 5. The user wants to write all six command words to all eight axes on the RMC, using the commands in V3000-V3057. This is done as follows:



Indirect Addressing

When communicating with the RMC with its large memory map, there may be times where it is desirable to read continuous blocks of data. For example, since only 64 words can be read at a time, to read the entire 2048-word Event Step table would require 32 reads. The first read would start at V2000, the second would start at V2100, and so on. You can take advantage of V-memory pointers (P type) for the RX and WX parameter. Again, this is out of the scope of RMC documentation and the user should refer to the DL205/405 PLC User Manual for details.

5.2.6.2.2 RMC Register Map (Automationdirect.com)

Tip: RMCWin's Address Tool provides an easy way to identify addresses in the RMC. Simply open the Address Tool and then move the cursor to any field in RMCWin that represents an RMC Register, and the Address Tool will display the address in the address format of your choice. See Address Tool for details.

The RMC module has 64K (65536) 16-bit registers that can be read and written over various communication types. This topic describes the mapping of a subset of those registers to Automationdirect.com DirectLogic 205/405 addresses. For the addresses of these registers for other communication methods such as Modbus Plus, PROFIBUS-DP, and other Ethernet protocols, see the following topics:

- RMC Register Map (Allen-Bradley)
- RMC Register Map (Modbus/TCP and Modbus/RTU)
- RMC Register Map (Omron FINS)
- RMC Register Map (Siemens TI505)
- RMC Register Map (Siemens S7)
- RMC Register Map (Modbus Plus)
- RMC Register Map (PROFIBUS-DP Message Mode)

The various Automationdirect.com DL205/405 PLCs have different ranges of V-memory that can be accessed via the Ethernet. The Ethernet-capable DL205/405 PLCs only support between 1280 and 15616 V-memory words and therefore cannot access 64K registers. For each block of registers below, the DL205/405 PLC CPUs that support that block are listed under **PLC Support**.

Status Registers:

PLC Support: D2-240, D2-250, D4-430, D4-440, and D4-450

These registers can only be read; writes are ignored.

Note: V0-V177 may be displayed as TA0-TA177, called aliases. You can disable displaying aliases by selecting the **View** menu in DirectSOFT32, **Options** menu item, **Global** tab, and clearing the **Display Aliases** checkbox.

V-memory Address	Register Description
V0	Axis 0 Command Position
V1	Axis 0 Target Position
V2	Axis 0 Actual Position
V3	Axis 0 Transducer Counts
V4	Axis 0 Status Word

V5	Axis 0 Drive
V6	Axis 0 Actual Speed
V7	Axis 0 Null Drive
V10	Axis 0 Event Step
V11	Axis 0 Link Value
V12-V23	Same as above but for axis 1
V24-V35	Same as above but for axis 2
V36-V47	Same as above but for axis 3
V50-V61	Same as above but for axis 4
V62-V73	Same as above but for axis 5
V74-V105	Same as above but for axis 6
V106-V117	Same as above but for axis 7

Command Registers:

PLC Support: D2-240, D2-250, D4-430, D4-440, and D4-450

These registers can be read or written.

Note: V0-V177 may be displayed as TA0-TA177, called aliases. You can disable displaying aliases by selecting the View menu in DirectSOFT32, Options menu item, Global tab, and clearing the Display Aliases checkbox.

V-memory Address	Register Description
V120	Axis 0 Mode Word
V121	Axis 0 Acceleration
V122	Axis 0 Deceleration
V123	Axis 0 Speed
V124	Axis 0 Command Value
V125	Axis 0 Command
V126-V133	Same as above but for axis 1

V134-V141	Same as above but for axis 2
V142-V147	Same as above but for axis 3
V150-V155	Same as above but for axis 4
V156-V163	Same as above but for axis 5
V164-V171	Same as above but for axis 6
V172-V177	Same as above but for axis 7

Parameter Registers:

PLC Support: D2-240, D2-250, D4-430, D4-440, and D4-450

These registers can be read or written. Changes to these registers do not take effect until a Set Parameters (P) command is executed.

Note: V1000-V1177 may be displayed as CTA0-CTA177, called aliases. You can disable displaying aliases by selecting the View menu in DirectSOFT32, Options menu item, Global tab, and clearing the Display Aliases checkbox.

V-memory Address	Register Description
V1000	Axis 0 Configuration Word
V1001	Axis 0 Scale
V1002	Axis 0 Offset
V1003	Axis 0 Extend Limit
V1004	Axis 0 Retract Limit
V1005	Axis 0 Proportional Gain
V1006	Axis 0 Integral Gain
V1007	Axis 0 Differential Gain
V1010	Axis 0 Extend Feed Forward
V1011	Axis 0 Retract Feed Forward
V1012	Axis 0 Extend Acceleration Feed Forward

V1013	Axis 0 Retract Acceleration Feed Forward
V1014	Axis 0 Dead Band Eliminator
V1015	Axis 0 In Position Window
V1016	Axis 0 Following Error
V1017	Axis 0 Auto Stop
V1020-V1037	Same as above but for axis 1
V1040-V1057	Same as above but for axis 2
V1060-V1077	Same as above but for axis 3
V1100-V1117	Same as above but for axis 4
V1120-V1137	Same as above but for axis 5
V1140-V1157	Same as above but for axis 6
V1160-V1177	Same as above but for axis 7

Event Step Table Registers:

PLC Support: D2-250, D4-430, D4-440, and D4-450: Full 256-step table
D2-240: Only the first 160 steps (up to V4377)

These registers can be read or written.

Note: In the table below, the form V2000+n*10 is used. The 10 is also octal. Therefore, the last digit will always be the same for a given field on every step. For example, the Mode word is V2000 for step 0, V2010 for step 1, V2020 for step 2, and so on up to V5770 for step 255.

The register map for addressing the fields in the event step table is as follows:

V-memory Address	Register Description
V2000	Step 0 Mode Word
V2001	Step 0 Acceleration
V2002	Step 0 Deceleration
V2003	Step 0 Speed

V2004	Step 0 Command Value
V2005	Step 0 Command/Commanded Axes
V2006	Step 0 Link Type/Link Next
V2007	Step 0 Link Value
V2000+n*10	Step n (0-255) Mode Word
V2001+n*10	Step n (0-255) Acceleration
V2002+n*10	Step n (0-255) Deceleration
V2003+n*10	Step n (0-255) Speed
V2004+n*10	Step n (0-255) Command Value
V2005+n*10	Step n (0-255) Command/Commanded Axes
V2006+n*10	Step n (0-255) Link Type/Link Next
V2007+n*10	Step n (0-255) Link Value

Input to Event Table Registers:

PLC Support: D2-250, D4-430, D4-440, and D4-450 (D2-240 excluded)

These registers can be read or written.

V-memory Address	**Register Description**
V6000	Event Step for Axis 0 on Input 0 Rising Edge
V6001	Event Step for Axis 1 on Input 0 Rising Edge
V6002	Event Step for Axis 2 on Input 0 Rising Edge
V6003	Event Step for Axis 3 on Input 0 Rising Edge
V6004	Event Step for Axis 4 on Input 0 Rising Edge
V6005	Event Step for Axis 5 on Input 0 Rising Edge
V6006	Event Step for Axis 6 on Input 0 Rising Edge
V6007	Event Step for Axis 7 on Input 0 Rising Edge
V6010 + n	Event Step for Axes n (0-7) on Input 1 Rising Edge

:	:
V6170 + n	Event Step for Axes n (0-7) on Input 15 Rising Edge
V6200 + n	Event Step for Axes n (0-7) on Input 0 Falling Edge
:	:
V6370 + n	Event Step for Axes n (0-7) on Input 15 Falling Edge

Status Map Registers:

PLC Support: D2-250, D4-430, D4-440, and D4-450 (D2-240 excluded)

This block of registers is only used by the Modbus Plus and PROFIBUS interfaces. Therefore, these registers are unused by Automationdirect.com Ethernet.

V-memory Address	Register Description
V6400-V6437	Status Map Entries

Plot Type Registers:

PLC Support: D2-250, D4-430, D4-440, and D4-450 (D2-240 excluded)

The plot type registers can be read or written. The values that are read indicate the extra plot information in the current graph. The values written to these registers tell the controller which extra plot information to obtain on the next plot. For these registers, the following values are used:

- 0: Extra position precision
- 1: Command and Command Value
- 2: Event Step and Link Value
- 3: Raw Transducer Counts
- 4: Internal Target and Actual Speeds
- 5: Integral Drive

For more information on these four types of plot information, see Selecting the Data to Plot and Reading Plots from the Communication Module.

V-memory Address	Register Description
V6500	Axis 0 plot type
V6501	Axis 1 plot type

V6502	Axis 2 plot type
V6503	Axis 3 plot type
V6504	Axis 4 plot type
V6505	Axis 5 plot type
V6506	Axis 6 plot type
V6507	Axis 7 plot type

Digital (Discrete) I/O Registers:

PLC Support: D2-250, D4-430, D4-440, and D4-450 (D2-240 excluded)

These registers indicate the current state of the digital inputs and outputs. These registers may only be read; writes will be ignored, as this product does not support forcing inputs or outputs.

The bit numbers listed in the table below are in RMC format (0 is LSB, 15 is MSB):

V-memory Address	Register Description
V6510	CPU Digital Inputs 0 and 1 in LSBs of low byte, Outputs 0 and 1 in LSBs of high byte
V6511	Unused
V6512	Unused
V6513	Sensor Digital I/O Inputs 0-15
V6514	Sensor Digital I/O Inputs 16-17 (stored to two LSBs)
V6515	Sensor Digital I/O Outputs 0-7 in high byte (low byte unused)
V6516	Unused
V6517	Unused

Plot Time Registers:

PLC Support: D2-250, D4-430, D4-440, and D4-450 (D2-240 excluded)

The Plot Time interval is configurable on the RMC. This interval indicates the number of control loops between each sample in a plot. Therefore, if the control loop is 0.976ms (e.g. RMC100-M1), this indicates roughly the number of milliseconds between samples. If the control loop is 1.953ms (e.g. RMC100-M4), this indicates half of the number of milliseconds between samples.

These registers may be read or written. When read, they indicate the plot interval of the currently gathered plot. When written, they set the plot interval the RMC should use for the next plot it will gather. When the RMC starts the next plot, it copies the requested plot interval into the currently used plot interval.

V-memory Address	**Register Description**
V6520	Axis 0 plot time interval
V6521	Axis 1 plot time interval
V6522	Axis 2 plot time interval
V6523	Axis 3 plot time interval
V6524	Axis 4 plot time interval
V6525	Axis 5 plot time interval
V6526	Axis 6 plot time interval
V6527	Axis 7 plot time interval

Last Parameter Error Registers:

PLC Support: D2-250, D4-430, D4-440, and D4-450 (D2-240 excluded)

Note: To use these registers, you must have RMC100 CPU control firmware dated 19990715 or later.

Each of these read-only registers holds the number of the last parameter error generated on an axis. This is useful for determining the cause of the Parameter Error bit in the status word. For a description of the values read from these registers, see Parameter Error Values.

V-memory Address	**Register Description**
V6530	Last parameter error on axis 0
V6531	Last parameter error on axis 1
V6532	Last parameter error on axis 2
V6533	Last parameter error on axis 3
V6534	Last parameter error on axis 4
V6535	Last parameter error on axis 5
V6536	Last parameter error on axis 6
V6537	Last parameter error on axis 7

Firmware Date Registers:

PLC Support: D2-250, D4-430, D4-440, and D4-450 (D2-240 excluded)

Note: To use these registers, you must have RMC100 CPU control firmware dated 19990715 or later.

These read-only registers hold information about the firmware versions in the RMC100 CPU module. The Boot and Loader firmware versions have no effect on the actual performance of the RMC and therefore can usually be ignored.

V-memory Address	Register Description
V6540	Boot firmware month (MSB) and day (LSB)
V6541	Boot firmware year
V6542	Loader firmware month (MSB) and day (LSB)
V6543	Loader firmware year
V6544	Control firmware month (MSB) and day (LSB)
V6545	Control firmware year
V6546	Control firmware Beta Code. This will be 0 for standard release firmware, 'B' for Beta firmware, or 'SI' for Superimposed firmware.
V6547	Feature code. This register is mainly reserved for internal use but does have two bits that may be useful to some users: • If bit 1 (value 0x0002) is set, the control loop is 2 ms, otherwise the control loop is 1 ms. • If bit 0 (value 0x0001) is set, a sensor DI/O is present, otherwise there is no sensor DI/O.

Reserved Registers:

PLC Support: D2-250, D4-430, D4-440, and D4-450 (D2-240 excluded)

Reading these values will return zero, and writes are ignored.

V-memory Address	Register Description
V6550-V3777	Unused

Spline Download Area:

PLC Support: D2-250, D4-440, and D4-450 (D2-240 and D4-430 excluded)

These registers are write only. Reading them will return zero. This area is used to download intervals and points in a spline. This is a much more efficient alternative to using individual New Spline Point and Set Spline Interval/End Segment commands. For details on using this Spline Download Area, see Downloading Splines to the RMC.

V-memory Address	Register Description
V10000-V17777	Spline Download Area

Plot Registers:

Due to the limited addressing supported by the Automationdirect.com DirectLogic 205/405 PLCs, plots cannot be read through the Ethernet.

5.2.6.3 EtherNet/IP Controllers

5.2.6.3.1 Using EtherNet/IP with the RMC ENET

EtherNet/IP is an open application protocol, maintained and distributed by ODVA (http://www.odva.org). EtherNet/IP combines the traditional messaging services with higher-performance and more deterministic I/O services. Messaging can be used to send arbitrary data between two devices using a request/response protocol. I/O is used to set up a specialized high-speed connection between two or more devices using the producer consumer model.

The RMC is a passive EtherNet/IP device. It does not establish its own I/O connections, nor does it initiate messaging transactions. Therefore, an active EtherNet/IP device or client is required to control the RMC or request data from the RMC.

The RMC ENET supports both EtherNet/IP messaging and I/O. The RMC100 ENET statements of conformance and EDS files are available for download from the Delta website at www.deltamotion.com.

Note: EtherNet/IP messaging requires RMC ENET firmware dated 20000420 or newer. EtherNet/IP I/O requires RMC ENET firmware dated 20010724 or newer.

Using EtherNet/IP I/O with the RMC

The RMC can support I/O connections with up to four devices (PLCs, etc.). An I/O connection sends specific data cyclically between the devices involved in the connection. This data is sent on an interval called the Requested Packet Interval (RPI). The RMC supports an RPI from 5.0 ms up to 3200.0 ms.

The Input data sent by the RMC includes one synchronization register followed by ten status words for each axis in the module. The Output data sent to the RMC includes one synchronization register followed by six command words for each axis in the module. Reading and writing any registers other than the Status and Command registers requires use of messaging. See Setting Up an EtherNet/IP I/O Connection and Controlling the RMC over EtherNet/IP I/O for details.

Using EtherNet/IP Messaging with the RMC

EtherNet/IP supports messaging in addition to I/O connections. Messaging can be used to read and write parameters, the Event Step table, splines, plots, status, and commands. In the ControlLogix, MSG (message) blocks are used to read and write data in the RMC using EtherNet/IP messaging. See Using Allen-Bradley Controllers with the RMC ENET for details.

EtherNet/IP Clients

The RMC is a passive EtherNet/IP device. It does not establish its own I/O connections, nor does it initiate messaging transactions. Therefore, an active EtherNet/IP device or client is required to control the RMC or request data from the RMC.

RMC ENET Supported Numbers of TCP and CIP Resources

TCP Connections:

4 TCP connections (all inbound)

Examples:

- RMCWin
- Each I/O connection is set up using an EtherNet/IP TCP connection
- Any other PLC communication

CIP Connected Messaging:

The RMC ENET limits CIP connections as follows:

- Max of 8 total CIP connections
- Max of 4 I/O CIP connections

Examples:

- I/O Data connection with a ControlLogix
- Input Data connection with a ControlLogix
- ControlLogix message to RMC ENET (cache enabled)

5.2.6.3.2 Setting Up an RMC EtherNet/IP I/O Connection

This topic describes the concepts involved in setting up an EtherNet/IP I/O connection. For step-by-step procedures for Allen-Bradley PLCs, see Configuring an RMC EtherNet/IP I/O Connection for the ControlLogix. For details on controlling the RMC once a connection has been made, see Controlling the RMC over EtherNet/IP I/O.

Setting up an EtherNet/IP I/O connection involves the following concepts:

- Using the Correct EDS File

- Connection Type
- Requested Packet Interval (RPI)
- Multicast vs. Point-to-Point (Unicast)
- Using a Generic EDS File

Using the Correct EDS File

Many vendors' EtherNet/IP configuration tools support importing the RMC's EDS file and then use the contents of this file to correctly configure a number of internal settings. However, some configuration tools—most notably RSLogix 5000 as of this writing—do not use EDS files to configure EtherNet/IP I/O connections. For tools such as these, see the Using a Generic EDS section below, or for RSLogix 5000, see the Configuring an RMC EtherNet/IP I/O Connection for the ControlLogix topic.

There are two EDS files currently available for RMC100 Ethernet module. Current versions are available for download from Delta's website www.deltamotion.com. Each EDS file is packaged in a compressed (.zip) file with its corresponding icon. Use the following table to select the correct EDS file to download:

EDS File	Version[1]	Firmware	Download Location
rmc100-enet_v1.eds	1.x	20020315-20130222	http://deltamotion.com/files/eds/rmc100-enet_v1_eds.zip
rmc100-enet_v2.eds	2.1	20151208 or newer	http://deltamotion.com/files/eds/rmc100-enet_v2_eds.zip

[1]This revision refers to the EtherNet/IP Identity Object revision, not the firmware revision. This revision only changes when the EtherNet/IP functionality is changed.

Once the correct EDS compressed file has been downloaded, extract the EDS file and icon to a temporary folder and import them into your EtherNet/IP configuration tool.

Connection Type

The RMC supports two types of I/O connections:

- **Input/Output**
  This connection is bidirectional: the originator (PLC or HMI) produces data consumed by the RMC and the target (RMC) produces data that is consumed by the originator. This connection type is also called an Exclusive Owner connection or the *controlling connection*. Each RMC can have no more than one Input/Output connection open at a time.

- **Input Only**
  For this connection type, only the target (RMC) produces data, which is consumed by the originator (PLC or HMI). The originator will only send a heartbeat packet, sometimes at a reduced interval (less frequently than the RPI) used to allow the RMC to identify when the connection is broken.

Of these two, the Input/Output connection type is by far the most commonly used. The Input Only is generally only used when multiple I/O connections are used. See the Establishing Multiple I/O

Connections with a Single RMC topic for details.

When using the RMC's EDS file in your EtherNet/IP configuration tool, you should be able to select the I/O connection type from a list. If you are using a Generic EDS File (as required by RSLogix 5000 prior to version 20.00), see the Using a Generic EDS File section below.

Requested Packet Interval (RPI)

EtherNet/IP I/O sends data between the communicating devices at the **Requested Packet Interval** (RPI). The RPI is configured in the EtherNet/IP controller (for example, RSLogix 5000), not in RMCTools. The RMC supports the following RPIs, based on the number of simultaneous I/O connections established (see the Establishing Multiple I/O Connections with a Single RMC topic):

I/O Connections	Minimum RPI	Typical RPI	Maximum RPI
1 (typical)	5 ms	20 ms	3,200 ms
2	7 ms	20 ms	3,200 ms
3	9 ms	20 ms	3,200 ms
4	12 ms	20 ms	3,200 ms

Delta recommends using the slowest RPI that meets the requirements of your application in order to reduce network requirements. See the EtherNet/IP Performance Overview topic for more details.

Multicast vs. Point-to-Point (Unicast)

The RMC supports both multicast and unicast (point-to-point) I/O connections. Traditionally, most EtherNet/IP I/O connections have been multicast, since Allen-Bradley controllers did not support unicast I/O connections until RSLogix 5000 version 18.00.00 was released in March 2010. However, multicast connections are much more demanding on the network bandwidth and require special switchgear (either routers or special switches with IGMP snooping/querying capabilities) to limit the spread of multicast traffic across the network. The RMC only requires multicast I/O connections when multiple I/O connections will be established at once, which is quite rare. Therefore, in almost all cases, unicast I/O connections should be selected whenever supported by the EtherNet/IP I/O controller.

Using a Generic EDS File

Some PLC EtherNet/IP configuration tools do not support importing third-party EDS files. In these cases, additional connection configuration information usually must be entered into a generic device template. The following chart provides the correct values for various connection settings that may be required by your EtherNet/IP configuration tool:

Setting	Connection Type	
	Input/Output	Input Only
Input Connection Point[1,2]	1	

Output Connection Point[1,2]	2	32
Config Data Assembly Instance[2]	4	
Connection Path[2]	20 04 24 04 2C 02 2C 01	20 04 24 04 2C 20 2C 01
Transport Class	1[3]	
Trigger Type	Cyclic[3]	
O->T Format	32-bit Run/Idle[3]	Heartbeat
O->T Fixed/Variable	Fixed[3]	
O->T Connection Type	Point-to-point[3]	
O->T Priority	Scheduled[3]	
O->T Size[4]	2, 4, …, 98 bytes	0 bytes
T->O Format	Modeless[3]	
T->O Fixed/Variable	Fixed[3]	
T->O Connection Type	Point-to-point or Multicast	
T->O Priority	Scheduled[3]	
T->O Size[5]	2, 4, …, 162 bytes	
O->T RPI	5ms to 3200ms	
T->O RPI	5ms to 3200ms	
Configuration Data Size	0 or 1 byte	0 bytes

[1] The term ‘Assembly Instance’ is used by some applications instead of ‘Connection Point’.

[2] The Input and Output Connection Points and Config Data Assembly Instance are implied by the Connection Path. Therefore, configuration tools will generally either ask for the Connection Points and Instances or the Connection Path but not both.

[3] This value is the common default value for this setting and the option to change this field may not be offered by the configuration tool.

[4] The O->T size does not include the 2-byte transport header and, for Input/Output connections, the 4-byte Run/Idle header.

[5] The T->O size does not include the 2-byte transport header.

Unfortunately, which fields are required and how they are presented varies significantly from one tool to the next. For this reason, most EtherNet/IP configuration tools are moving toward supporting importing EDS files to reduce the complexity of setting up an I/O connection.

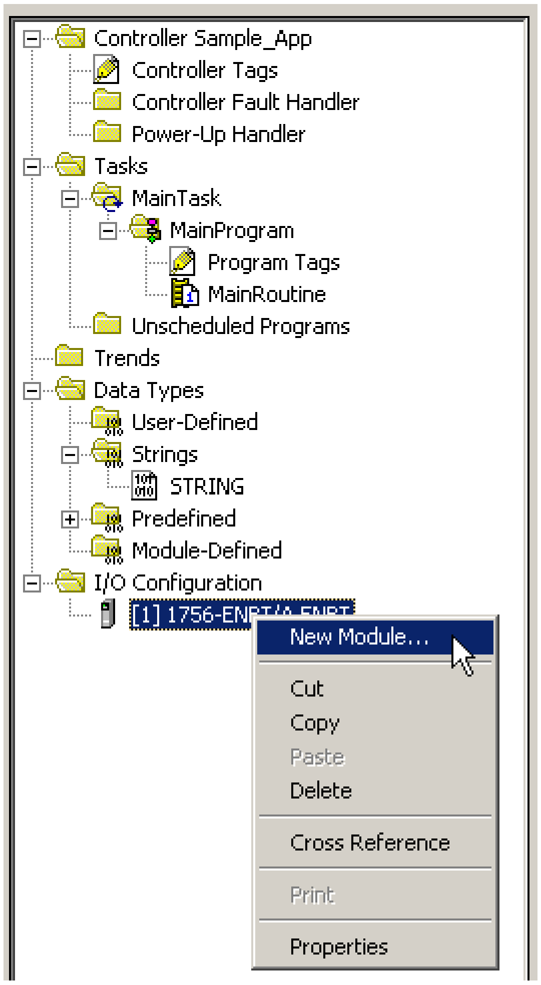
The following chart shows the fields required to set up an EtherNet/IP generic device connection using Rockwell Automation's RSLogix 5000. For step-by-step instructions on setting up an Input/Output connection using RSLogix 5000, see the Configuring an RMC EtherNet/IP I/O Connection for the ControlLogix topic.

	Connection Type	
Setting	**Input/Output**	**Input Only**
Comm Format	Data – INT	Input Data – INT
Input Assembly Instance	1	1
Input Size	1-81 (16-bit)	1-81 (16-bit)
Output Assembly Instance	2	32
Output Size	1-49 (16-bit)	--
Configuration Assembly Instance	4	4
Configuration Size	0 (8-bit)	0 (8-bit)
Requested Packet Interval (RPI)	5.0ms - 3200.0ms	5.0ms - 3200.0ms
Use Unicast Connection	Yes or No	No

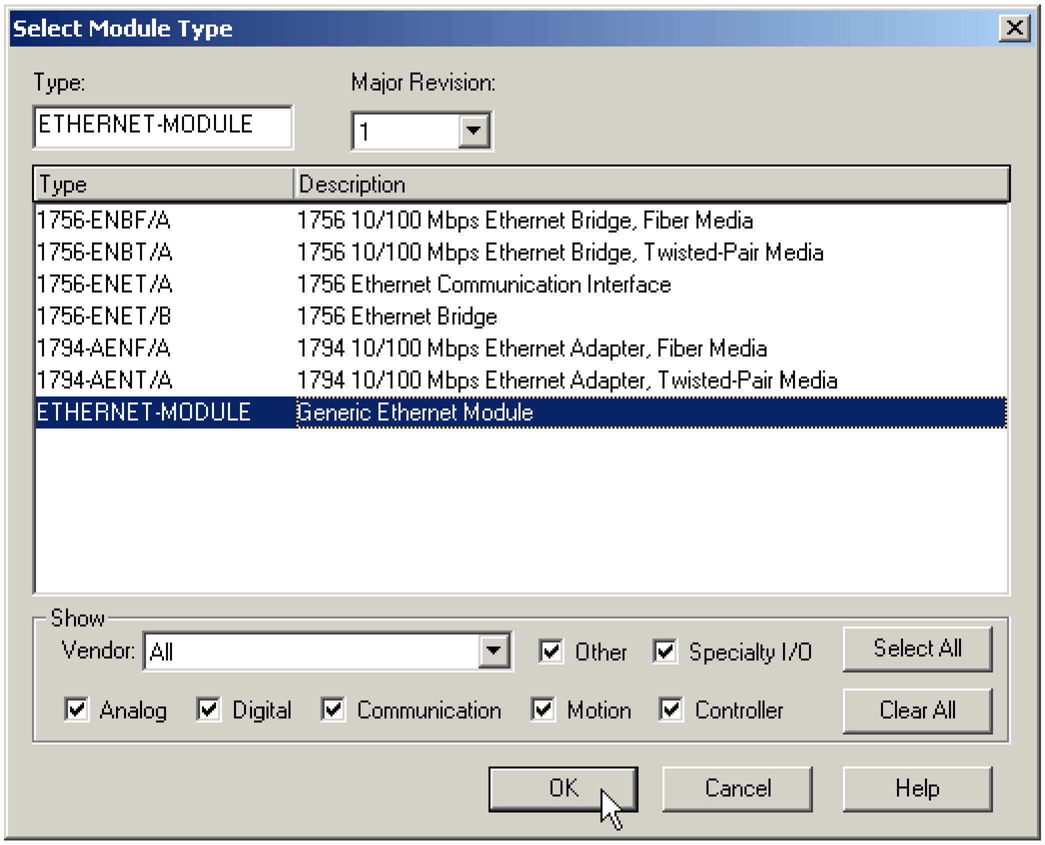
5.2.6.3.3 Configuring an RMC EtherNet/IP I/O Connection for the ControlLogix

Allen-Bradley's RSLogix 5000 software is used to set up an I/O connection between an RMC and a ControlLogix PLC. This topic describes how to set up this connection.

1. Start RSLogix 5000 and open the project to which you want to add an RMC I/O connection.
2. In the Controller Organization window, add a 1756-ENET/B or 1756-ENBT/A module under the **I/O Configuration** item. If the ControlLogix Ethernet module that you want to use already exists, then skip this step. Otherwise, refer to the Ethernet module's manual for details on adding the module.
3. In the Controller Organization window, right click on the 1756-ENET/B or 1756-ENBT/A under which you want to add the RMC. The following shortcut menu will be displayed:



4. In the shortcut menu that appears, click **New Module**. The following dialog box will be displayed:



Select Module Type

Type: ETHERNET-MODULE

Major Revision: 1

Type	Description
1756-ENBF/A	1756 10/100 Mbps Ethernet Bridge, Fiber Media
1756-ENBT/A	1756 10/100 Mbps Ethernet Bridge, Twisted-Pair Media
1756-ENET/A	1756 Ethernet Communication Interface
1756-ENET/B	1756 Ethernet Bridge
1794-AENF/A	1794 10/100 Mbps Ethernet Adapter, Fiber Media
1794-AENT/A	1794 10/100 Mbps Ethernet Adapter, Twisted-Pair Media
ETHERNET-MODULE	Generic Ethernet Module

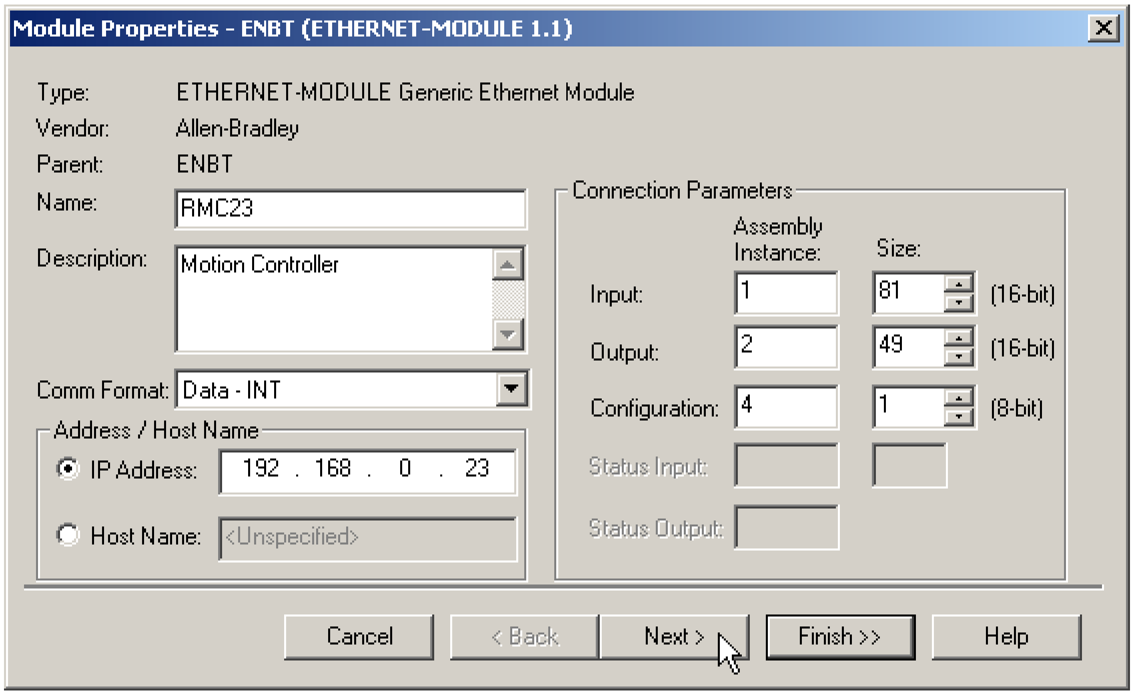
Show

Vendor: All

☑ Other ☑ Specialty I/O

☑ Analog ☑ Digital ☑ Communication ☑ Motion ☑ Controller

5. Click the ETHERNET-MODULE type and click **OK**. The following dialog box will be displayed:



Module Properties - ENBT (ETHERNET-MODULE 1.1)

Type: ETHERNET-MODULE Generic Ethernet Module

Vendor: Allen-Bradley

Parent: ENBT

Name: RMC23

Description: Motion Controller

Comm Format: Data - INT

Address / Host Name

● IP Address: 192 . 168 . 0 . 23

○ Host Name: <Unspecified>

Connection Parameters

	Assembly Instance:	Size:	
Input:	1	81	(16-bit)
Output:	2	49	(16-bit)
Configuration:	4	1	(8-bit)
Status Input:			
Status Output:			

6. Fill in the fields in this dialog box as follows:

General:

Name	Type a valid module name for the RMC.
Description	Type a description.
Comm Format	Select one of the following formats: **Data - INT** Input and Output Data will be allocated. Only a single connection originator (ControlLogix CPU) can use this format for any given RMC. **Input Data - INT** Input Data only will be allocated. Up to four connection originators can use this format simultaneously.
Address/ Host Name	Enter the IP address or host name of the RMC. The RMC must have its IP address set up to match this address. See RMC Ethernet IP Address Setup for details. To use a host name, the DNS server must be configured independently from the RMC. The RMC has no built-in DNS support.

Connection Parameters:

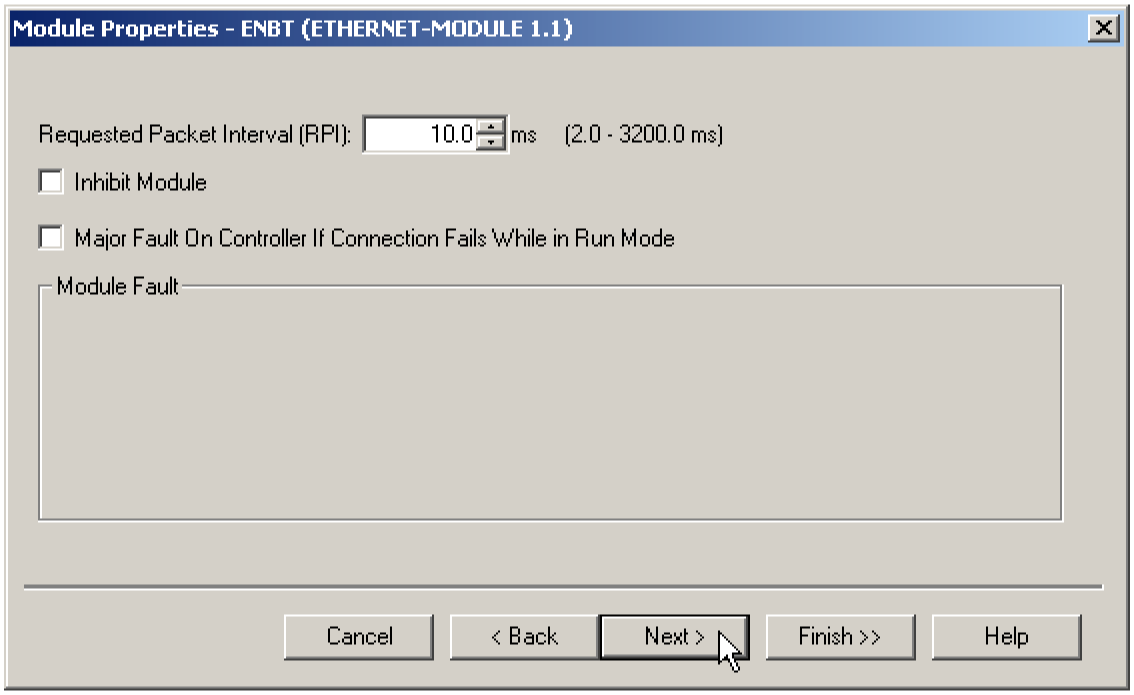
	Assembly Instance	**Data Size**	**Size**
Input	1	16-bit	1 plus 10 per axis (11, 21, …, 81)
Output	2*	16-bit	1 plus 6 per axis (7, 13, …, 49)*
Configuration	4	8-bit	0 or 1

* If the "**Input Data - INT**" Comm Format is selected, the size for the Output will be unavailable, and the Assembly Instance should be set to 32.

See Controlling the RMC over EtherNet/IP I/O for a description of the contents of the Input, Output, and Configuration data areas.

Note: The Status Input and Output assembly options are not used by the RMC.

7. Click **Next**. The following dialog box will be displayed:


Module Properties - ENBT (ETHERNET-MODULE 1.1)

Requested Packet Interval (RPI): 10.0 ms (2.0 - 3200.0 ms)

☐ Inhibit Module

☐ Major Fault On Controller If Connection Fails While in Run Mode

Module Fault

8. Type a Requested Packet Interval (RPI) between 5.0 and 3200.0 ms in steps of 1.0 ms. The RMC ignores fractions of a millisecond and cannot support an RPI below 5.0 ms.

9. Set the **Inhibit Module** and **Major Fault On Controller if Connection Fails While in Run Mode** check boxes as required by your application.

10. Click **Finish**.

    The above steps will allocate two or three tags in the Controller Tags database in RSLogix 5000. These tags correspond to the Input, Output, and Configuration data set up for the module under the Connection Parameters section above. The type of each tag is a special module-defined type created by RSLogix. Each special type has a Data field that holds the actual data. The following table summarizes the tags created for each module:

Tag Name	**Type of Data**	**Description**
[Name]:I	INT[size]	Input Data.
[Name]:O	INT[size]	Output Data. This tag exists only for **Data - INT** and not **Input Data - INT** connections.
[Name]:C	SINT[400]	Configuration Data. Notice that a full 400 bytes is allocated by the ControlLogix regardless of how many are actually configured to be sent to the RMC.

See Controlling the RMC over EtherNet/IP I/O for a description of the contents of the Input, Output, and Configuration data areas.

5.2.6.3.4 Establishing Multiple I/O Connections with a Single RMC

Each RMC can support I/O connections with up to four EtherNet/IP clients such as the ControlLogix 1756-L1. Each connection in an RMC must use the same RPI and Input Data size. Also, only one of these can use a *controlling* connection. The rest must use *Input Only* connections. When multiple connections to the same RMC are used, all connections must use multicast (not unicast).

A **controlling** connection is one that has both input and output data. The output data (from the client) contains commands that the RMC will process. The output data also includes PLC status information that is used for detecting when the client switches from Run to Program mode. See Handling Broken I/O Connections for details.

An *Input Only* connection is one that has only input data. The only output data sent to the RMC is a heartbeat frame used to detect connection time outs.

For more information on I/O connections see Controlling the RMC over EtherNet/IP I/O.

When using the EDS file, the controller connection is established when the **Input/Output** connection is selected, and an **Input Only** connection is established when the Input Only connection is selected. When using a ControlLogix generic Ethernet device, a controlling connection is set up by selecting the **Data - INT** format and Output Assembly Instance 2, and an input only connection is set up by selecting the Input **Data - INT** format and Output Assembly Instance 32.

Example:

Suppose three EtherNet/IP clients want to access the data for the first four axes of an RMC. As required by the RMC, only one of these controllers will have a controlling connection. The fastest RPI required in this application is 10.0 ms.

The controlling EtherNet/IP client will have a controlling connection. Because this connection will access four axes, the Input size should be 41 words, and the Output size should be 25 words. An RPI of 10 ms is used.

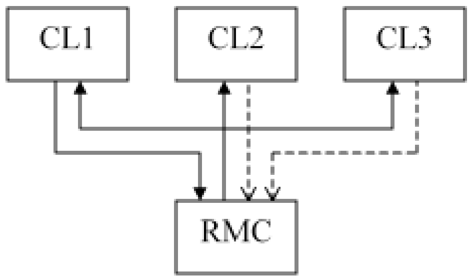
The other EtherNet/IP clients can now also be configured to access the same RMC, but they must use Input Only connections. They must also use the same RPI (10 ms) and input size (41 words).

Suppose that one of these EtherNet/IP clients also wanted to monitor the fifth axis, even though it is not controlling that axis. This would require increasing the Input Size from 41 to 51 words on all controllers accessing this RMC's input data.

5.2.6.3.5 Controlling the RMC over EtherNet/IP I/O

As described in the Setting Up an EtherNet/IP I/O Connection topic, each RMC can have I/O connections open with up to four EtherNet/IP clients (such as the ControlLogix PLC). Only one of these connections can produce data for the RMC to consume. All of these connections will consume data produced by the RMC.

The following diagram shows three clients accessing a single RMC:



Notice how the RMC produces one data frame that is consumed by all three clients using what is called a multicast. All three clients produce data frames consumed by the RMC, but only one has data for the RMC (CL1 in this example). The other two are heartbeat frames to time out old connections. These heartbeat connections are shown as dotted lines in the above diagram.

Therefore, excluding the heartbeat frames, the RMC only receives one set of data and generates one set of data each RPI, regardless of how many I/O connections it is handling. So, what data is included in these frames?

Output Data (from Client):

Offsets and sizes are in 16-bit words (INT on the ControlLogix).

Offset	Size	Description
0	1	**Sync Out Register.** See Using the Sync Registers below for details. This register must be used to issue commands to the RMC.
1	6	**Axis 0 Command.** These six registers correspond to the Mode, Acceleration, Deceleration, Speed, Command Value, and Command registers for axis 0. See Using the Sync Registers below for details on issuing commands.
7	6	**Axis 1 Command.** Same as for axis 0.
13	6	**Axis 2 Command.** Same as for axis 0.
19	6	**Axis 3 Command.** Same as for axis 0.
25	6	**Axis 4 Command.** Same as for axis 0.
31	6	**Axis 5 Command.** Same as for axis 0.
37	6	**Axis 6 Command.** Same as for axis 0.
43	6	**Axis 7 Command.** Same as for axis 0.

In addition to the above Output Data, the controlling client also sends a four-byte prefix, which indicates whether the client is in Run or Program mode. This is used as described in Handling Broken I/O Connections.

Input Data (to Client):

Offsets and sizes are in 16-bit words (INT on the ControlLogix).

Offset	Size	Description
0	1	**Sync In Register.** See Using the Sync Registers below

		for details.
1	10	**Axis 0 Status.** These ten registers correspond to the ten status registers displayed in RMCWin for an axis.
11	10	**Axis 1 Status.** Same as for axis 0.
21	10	**Axis 2 Status.** Same as for axis 0.
31	10	**Axis 3 Status.** Same as for axis 0.
41	10	**Axis 4 Status.** Same as for axis 0.
51	10	**Axis 5 Status.** Same as for axis 0.
61	10	**Axis 6 Status.** Same as for axis 0.
71	10	**Axis 7 Status.** Same as for axis 0.

The Input Data is updated and sent at the RPI by the RMC regardless of the state of the Sync Registers. As described below under Using the Sync Registers, the Sync Registers only apply when issuing commands.

Using the Sync Registers

The RMC ENET does not look at the command registers in the Output Data it receives until the Sync Out Register changes. When this register changes, then all commands for all available axes are processed.

The Sync In Register will change to match the Sync Out Register after the command(s) have been received by the RMC and the status registers in the Input Data have been updated.

Sync Registers are used for the following reasons:

- To avoid splitting up of commands issued to the RMC.

  Example:

  The ControlLogix does not synchronize its I/O with the PLC scan. Therefore, in the time it takes to place values in the command words for one or more axes, the Output Data could be sent out to RMC, mixing some old and some new data. By having the RMC ignore command changes until the Sync Out Register changes, this problem is avoided.

- To coordinate when the status in the Input Data has been updated to reflect a command that was issued.

  Example:

  Suppose an axis is in position, and therefore the In Position bit is set in the axis's Status word. The PLC sends a Go (G) command to this axis to move it to another position. As soon as this command is received by the RMC, the In Position bit will be cleared, and it will not be set again until the axis reaches the new position. However, if the In Position bit in the Input Data is checked before the client receives Input Data reflecting the Go command, then the In Position bit will still be set, possibly leading the PLC program to think the axis is in-position prematurely.

  Here is the recommended sequence for issuing commands:

1. Wait until the Sync In and Sync Out Registers match.
   If they do not match, then this means that another command or set of commands is in progress.
2. Clear any old commands from the Command registers for each axis.

Otherwise, when the Sync Out Register is changed, the commands would be re-issued.

3. Write all required command fields to the Output Data for all commands you want to issue.
   You can issue up to one command per axis. Leave the Command field set to 0 for each axis that you do not want to issue a command to.

4. Change the Sync Out Register.
   The easiest way to do this is to add one to it. Some PLCs report an error when a register overflows (e.g. 32,767 to -32,768), so you will instead want to add one and mask it with a value such as 16,383. This will make the register count from 0 to 16,383, and then wrap back down to 0 without an error. Take care to ensure that you only update the Sync Out Register once so that the commands do not get re-issued. Therefore, you may need to do intermediate processing in a separate local register and copy the result into the Sync Out Register.

5. Wait for the Sync In Register to match the Sync Out Register.
   It is important to wait until this occurs before using the status bits in the Input Data. See the example above for how problems can occur if this step is ignored.

Configuration Data

In addition to the Input and Output data that is sent every RPI, configuration data can be sent when the connection is established. The RMC defines only the first configuration byte. Any additional configuration bytes must be zero or the connection attempt will be rejected. This byte is defined as follows:

Configuration Data:

Offsets and sizes are in 8-bit bytes (SINT on the ControlLogix)

Offset	Size	Description
0	1	**Broken Connection Action.** This byte controls the RMC's handling of a broken controlling I/O connection, or a transition of the controlling EtherNet/IP client from Run to Stop/Program. The options for this field are described in Handling Broken I/O Connections. If no configuration data is sent when the connection is established, then this value is assumed to be -1.

5.2.6.3.6 Handling Broken I/O Connections

It is important in many industrial applications to detect faults quickly. One such fault is losing communication to EtherNet/IP I/O. EtherNet/IP supports a variable timeout value, which is expressed in terms of Requested Packet Intervals (RPIs). For example, the ControlLogix establishes its EtherNet/IP I/O connections with a timeout of 32 RPIs. Therefore, an RPI of 5.0 ms will have a timeout of 32 x 5.0 ms or 160 ms.

When either device in an I/O connection does not receive a packet from the other device for the timeout interval, it closes the connection and typically indicates this condition to the main program. The method of indicating this condition depends on the actual device. This topic describes the methods used by the RMC and ControlLogix.

Handling Broken I/O Connections in the RMC

The following conditions are defined as a broken connection in the RMC ENET:

- The controlling connection is broken due to a timeout. This will occur when the cable is disconnected, when excessive collisions cause the connection to timeout, or when the client is powered off or reset.
- The controlling client intentionally closes the connection. This can happen when an I/O connection is removed, when a new program is downloaded to the client, or when the connection is reconfigured to have a different RPI or data size.
- Although not strictly a broken connection, the condition of the controlling client switching from Run to Program mode also triggers this handling.

Note: All of these conditions only apply to the controlling client and its connection. No broken connection actions will be done when an Input Only connection is broken.

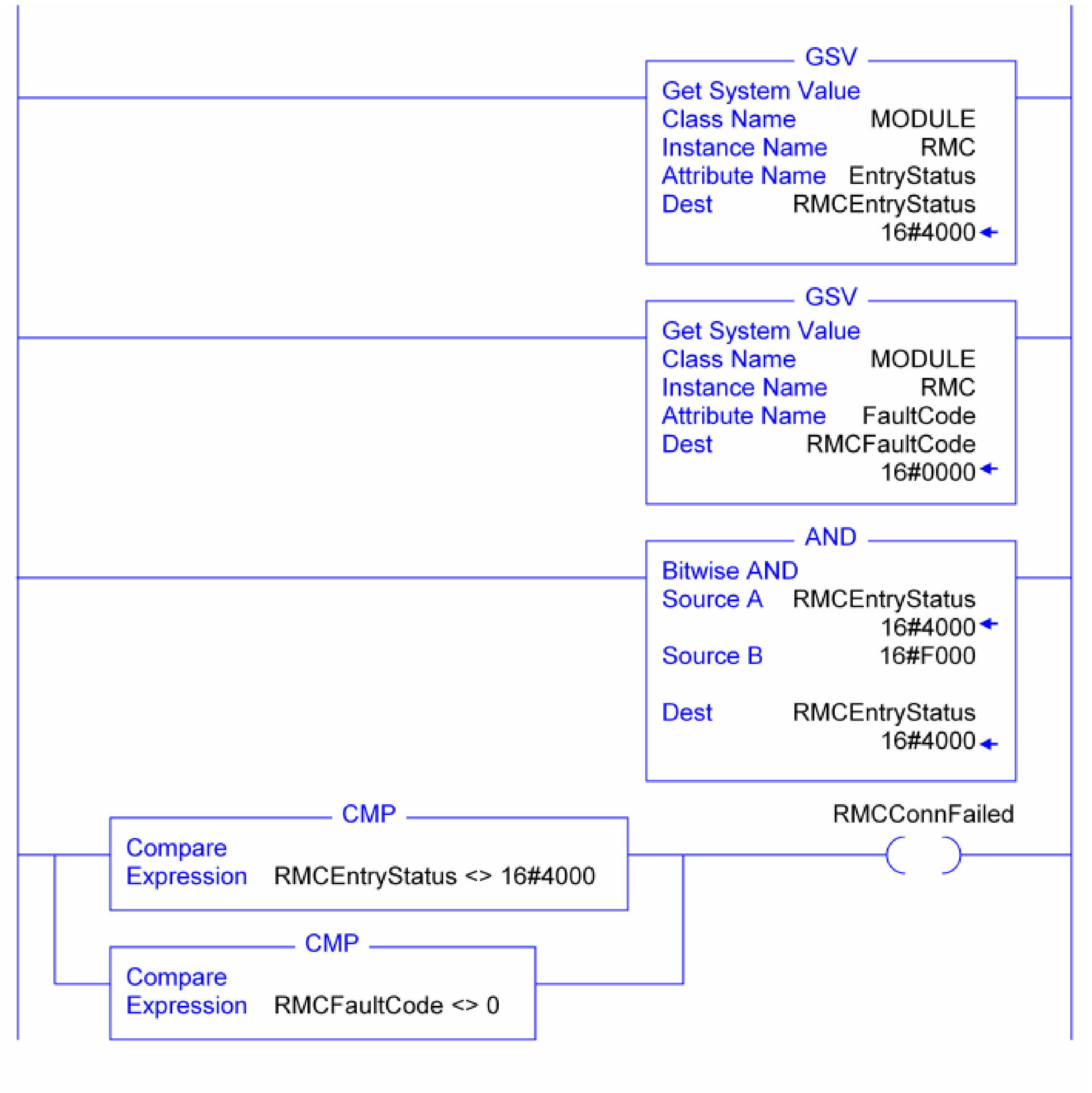
The action taken when any of these conditions occur is selectable by the Broken Connection Action byte of the Configuration Data described in Controlling the RMC over EtherNet/IP I/O. If this byte is omitted, the connection will behave as though the Broken Connection Action byte was specified as -1. The following values define the recovery action taken:

Value	Action
-1	Every axis without the Continue bit set in its Configuration word is halted and its event step sequence stops if it was running. All such axes will receive the Halt (H) command unless they are in open loop, in which case they will receive the Disable Drive Output (K) command. For more flexibility, use Broken Connection Action values of 0 to 31 as described below.
0 to 15	The Simulate Rising Edge ({) command is issued to axis 0 with a command value equal to the value of the Broken Connection Action byte (0 to 15). This allows the user to define an event sequence to run on any or all of the axes. Therefore, event sequences can be stopped and other special recovery sequences can be run. Review the Simulate Rising Edge command and the Input-to-Event table for details.
16 to 31	Same as 0 to 15, except that the Simulate Falling Edge (}) command is issued with a command value equal to the value of the Broken Connection Action byte minus sixteen (0 to 15).

Handling Broken I/O Connections in the ControlLogix

The ControlLogix has two methods of handling a broken connection with an EtherNet/IP device such as the RMC:

- In the **Module Properties** dialog box for the RMC, on the **Connection** tab, the **Major Fault On Controller If Connection Fails In Run Mode** check box can be checked to fault the ControlLogix when the EtherNet/IP connection to the RMC is broken.
- The Get System Value (GSV) block can be used to read the status of the connection in ladder logic. The following ladder logic demonstrates this:



The core of this ladder segment is reading the EntryStatus and FaultCode attributes from the RMC MODULE object using the GSV blocks. The MODULE objects are internal to the ControlLogix and represent external modules. In the Instance Name field of the GSV blocks, type the name you selected for the particular RMC module.

If the connection to the module is running, then the high four bits of the EntryStatus will be equal to 4 and the FaultCode will be equal to 0. This is described in the RSLogix 5000 online help's "Accessing the MODULE Object" topic.

The above ladder masks off the low 12 bits of the EntryStatus using an AND block, and then sets the RMCConnFault coil to indicate whether or not the connection is faulted.

5.2.6.3.7 EtherNet/IP Performance

5.2.6.3.7.1 EtherNet/IP Performance Overview

The following factors affect the determinism and performance of an EtherNet/IP I/O connection:

- **Load on the RMC ENET**
  The RMC ENET load is described in Evaluating the Load on the RMC ENET.
- **Load and bandwidth of the 1756-ENET or 1756-ENBT module**
  The 1756-ENET load is described in Evaluating the Load on the 1756-ENET, and the 1756-ENBT load is described in Evaluating the Load on the 1756-ENBT.
- **LAN Component Capabilities (10/100 Mbps, Half/Full-Duplex, Hubs/Switches)**
  The effects of these capabilities are described in Predicting the Effect of Collisions.
- **Utilization of the LAN Bandwidth**
  Utilization is defined and its effects are evaluated in Predicting the Effect of Collisions.

In addition to these specific considerations, the Setting Up Large EtherNet/IP Networks topic gives suggestions for reducing collisions and the load on network components in large EtherNet/IP networks.

5.2.6.3.7.2 Evaluating the Load on the RMC ENET

The RMC ENET's bandwidth is limited to 500 frames/second. This bandwidth is shared among the one to four I/O simultaneous connections supported by the RMC. See Establishing Multiple I/O Connections with a Single RMC for details on having multiple connections. The important point from this topic is that all connections for each RMC must have the same RPI.

To compute the bandwidth requirement on the RMC ENET use the following formula:

Frames/Second = (1 + connections) / RPI

This value should be kept below 90% of the RMC ENET bandwidth, or 450 frames/second. Using too close to the maximum bandwidth would result in decreased determinism and degraded performance of messaging tasks such as using ControlLogix MSG blocks and connecting via RMCWin.

The following table gives the minimum RPI allowed to keep the bandwidth under 90% for one to four connections:

I/O Connections	**Min RPI**
1	5.0 ms
2	7.0 ms
3	9.0 ms
4	12.0 ms

Example:

Suppose you will be establishing one I/O connection to an RMC, and the RPI will be 15.0 ms. Use the formula above to compute the RMC ENET load:

Frames/Second	=	(1 + connections) / RPI
	=	(1 + 1) / 0.015s
	=	133

This is only 27% of the RMC ENET bandwidth. Therefore, a lot of processing time will be available for handling non-I/O traffic such as from RMCWin and ControlLogix MSG blocks.

5.2.6.3.7.3 Evaluating the Load on the 1756-ENET

Rockwell Automation's **EtherNet/IP Performance and Application Guide** (Pub. No. ENET-AP001C-EN-P) describes how to compute the 1756-ENET loading in detail. When using the worksheets in that manual, consider each RMC connection a "Rack Optimized Connection." That is, the Frames/Second required for an RMC in the 1756-ENET is computed the same as Rack Optimized Connections:

Frames/Second = (2 x connections) / RPI

The 1756-ENET module's bandwidth is limited to 900 frames/second. Rockwell recommends that no more than 90% (810 frames/second) be allocated to I/O connections. From our independent testing, we recommend allocating no more than 80% (720 frames/second) for I/O connections.

If a 1756-ENET will be used only for controlling RMCs and each RMC has the same RPI, then the following chart can be used to determine the minimum RPI allowed for the number of RMCs:

RMCs	**Min RPI**
1	5.0 ms*
2	6.0 ms
3	9.0 ms
4	12.0 ms
5	14.0 ms
6	17.0 ms
7	20.0 ms
8	23.0 ms
9	25.0 ms
10	28.0

ms

* The 1756-ENET has enough bandwidth for a single connection with an RPI as low as 3.0 ms, but the RMC does not support that low of an RPI.

If the 1756-ENET is controlling other non-RMC I/O, the bandwidth required by these other connection will also need to be taken into consideration. The **EtherNet/IP Performance and Application Guide** covers these more advanced configurations.

In addition to the 1756-ENET and RMC ENET load limitations, the effect of collisions needs to be considered. See Predicting the Effect of Collisions for details.

Example:

Suppose one ControlLogix will be controlling three RMCs. The intended RPI is 6.0 ms. Therefore the bandwidth in frames/second is computed as follows:

Frames/Second	=	(2 x connections) / RPI
	=	(2 x 3) / 0.006s
	=	1000

This is over both the 720 frames/second we recommend and the 810 frames/second that Rockwell recommends. Therefore, you have three options:

- Increase the RPI of one or more of the RMCs until the bandwidth is below 720. Raising each to 9.0 ms does this.
- Replace the 1756-ENET with the 1756-ENBT. The bandwidth on the 1756-ENBT (see Evaluating the Load on the 1756-ENBT) is limited to 5000 frames/second, and easily handles this load.
- Use two 1756-ENET modules: one will control two RMCs with RPIs of 6.0 ms (667 frames/second) and the second will control one RMC with an RPI of 6.0 ms (333 frames/second). Both are within the recommended limits.

Example:

Suppose one ControlLogix will be controlling three RMCs again, but this time one RMC needs an RPI of 5.0 ms, and the other two RMCs need an RPI of 15.0 ms. To compute the bandwidth requirement on the ControlLogix, compute the bandwidth required at each RPI, and sum them together:

Frames/Second	=	(2 x connections) / RPI + (2 x connections) / RPI
	=	(2 x 1) / 0.005s + (2 x 2) / 0.015s
	=	400 + 267
	=	667

This load is under the recommended 80% bandwidth (720 frames/second) of the 1756-ENET. Therefore, this network should work.

5.2.6.3.7.4 Evaluating the Load on the 1756-ENBT

The 1756-ENBT has a total bandwidth of 5000 frames/second. If we reserve 10% of this bandwidth for non-I/O communications (RSLogix 5000, etc.), then we are left with 4500 frames/second. This is enough bandwidth to control eleven RMCs at the RMCs' minimum RPI of 5 ms. As will be discussed in the next topic, collisions on the RMC-to-switch collision domains preclude this many RMCs being placed on the same network with such a low RPI in most applications.

The load on the 1756-ENBT is computed in the same way as the 1756-ENET. However, for the purpose of controlling RMCs, the collisions will be the limiting factor in network performance and determinism when using a 1756-ENBT. See Predicting the Effect of Collisions for details. For additional considerations when setting up large EtherNet/IP networks, see Setting Up Large EtherNet/IP Networks.

5.2.6.3.7.5 Predicting the Effect of Collisions

Collisions

With half-duplex (full-duplex will be described below) Ethernet, collisions occur when two or more devices attempt to send a frame at the same time. When frames collide, the frame must be retried, so each device delays (backs-off) a random amount of time and then retries. For each successive collision on a frame, the maximum back-off time doubles. If a frame collides sixteen successive times, the frame will be discarded.

Note: RMCWin provides a histogram of number of frames that have collided a given number of times. See RMC Ethernet Statistics for details.

Collision Domains

Ethernet networks are divided into collision domains. A collision domain is the set of all devices that can collide with one another. The more devices in a collision domain: the higher the probability of a collision. Each collision domain extends through any hubs or repeaters and stop at switches, routers, or end-devices (PCs, RMCs, PLCs). This is best illustrated with some examples:

- A network has four RMCs and one 1756-ENET connected to a hub. This network has a single collision domain with five devices competing for its bandwidth.
- A network has four RMCs and one 1756-ENET connected to a switch. This network has five collision domains, each with two devices competing for its bandwidth. These smaller collision domains will yield a more deterministic and higher performance network. This is why we recommend using a switch instead of a hub.
- A network has four RMCs and one 1756-ENET connection to a switch. The switch is then connected through a router to the rest of the plant network. The control network will have six collision domains: the five in the previous example plus one between the switch and router. All traffic on the plant network will be in separate collision domains.

Full-Duplex Ethernet

Full-duplex Ethernet occurs between exactly two devices (usually a switch and an end-device such as the 1756-ENBT) when both devices support full-duplex (the RMC100 does not support full-duplex Ethernet). Each device in a full-duplex segment can send and receive data at the same time, thereby doubling the bandwidth, and more importantly, eliminating collisions. The 1756-ENBT supports full-duplex Ethernet. Therefore, use of a full-duplex-capable switch is highly recommended to eliminate collisions between the 1756-ENBTs and the switch, which are the most heavily trafficked segments. The 1756-ENET and RMC ENET do not support full-duplex, and therefore will be susceptible to collisions.

Network Utilization

The probability of a collision occurring depends primarily on the utilization of the network in a collision domain. Utilization is the relationship between the available bandwidth (10 or 100 Mbps) and the actual amount of data vying for that bandwidth. The amount of data is a function of the number and size of frames being sent. We will represent utilization as the percent of time the wire is active.

Utilization vs. Collisions

The probability of collisions in a given collision domain depends on the network utilization. As is described below, the maximum delay per frame depends on how many collisions a frame experiences, therefore a good approach is to choose how long of a delay is acceptable and then determine how many collisions per frame are acceptable.

The maximum acceptable collisions per frame, and therefore the maximum acceptable delay per frame will vary from application to application. Typically the absolute limit will be the timeout time for an I/O connection. The ControlLogix sets up its I/O connections to time out if no packets are received within 32 RPIs. For example, for an RPI of 5.0 ms, the connection will reset if a packet is not received in 32 x 5.0 ms or 160 ms. Therefore, the number of collisions per frame should be kept below 11 (using the chart below). When a connection times out, the connection is closed and may take several seconds to be re-established. During this time the PLC and RMC will execute any broken connection handling they have implemented.

Notice that with half-duplex Ethernet, there can be no guarantee that a frame will reach its destination by any given time. However, the probability of such an event may become so low that they are effectively masked by other more-probable events such as a cable being cut or a plant fire. Again, it should be noted that with full-duplex Ethernet, collisions do not exist, and as such, the Ethernet media becomes deterministic.

The following table shows the percentage of the frames that collided n or more times at various network utilizations in lab tests at Delta. These tests were done with a single ControlLogix 1756-ENBT/A module requesting data from multiple RMCs with RPIs of 5.0 ms using a switch. The same utilization with a hub would result in higher collisions since more than two devices can compete for the same bandwidth at once:

			Utilization		
Collisions:	**7.6%**	**15.1%**	**22.7%**	**30.2%**	**37.8%**
1 or more	0.14%	1.0%	3.0%	5.6%	8.9%
2 or more	0.028%	0.23%	0.80%	1.7%	3.0%
3 or more	0.0020%	0.023%	0.12%	0.35%	0.74%
4 or more	0.000025%	0.0013%	0.011%	0.052%	0.14%
5 or more	0.0%	0.000049%	0.00066%	0.0044%	0.017%
6 or more	0.0%	0.000001%	0.000041%	0.00029%	0.0020%
7 or more	0.0%	0.0%	0.000002%	0.000017%	0.00038%
8 or more	0.0%	0.0%	0.0%	0.0%	0.000069%

9 or more	0.0%	0.0%	0.0%	0.0%	0.000005%

The above statistics were captured on a network using a switch. Networks using a hub will experience higher rates of collisions for the same utilization because the probability of a collision increases with more devices.

Delays Incurred by Collisions

The time delayed for each frame depends on the baud rate (10 or 100 Mbps) and frame size. The following chart shows the total minimum and maximum delays incurred by a frame having from zero to sixteen collisions assuming the maximum frame size utilized by a RMC I/O frame. Notice that only the 10 Mbps section of the chart applies to the switch-to-RMC segment of the LAN since the RMC ENET is 10 Mbps.

	10 Mbps Delays		**100 Mbps Delays**	
Collisions	**Min (ms)**	**Max (ms)**	**Min (ms)**	**Max (ms)**
0	0.000	0.198*	0.000	0.020*
1	0.013	0.502	0.001	0.050
2	0.026	0.909	0.003	0.091
3	0.038	1.52	0.004	0.152
4	0.051	2.54	0.005	0.254
5	0.064	4.38	0.006	0.438
6	0.077	7.86	0.008	0.786
7	0.090	14.6	0.009	1.46
8	0.102	27.9	0.010	2.79
9	0.115	54.3	0.012	5.43
10	0.128	107.	0.013	10.7
11	0.141	160.	0.014	16.0
12	0.154	212.	0.015	21.2
13	0.166	265.	0.017	26.5
14	0.179	317.	0.018	31.7
15	0.192	370.	0.019	37.0
16	0.205	423.	0.020	42.3

* Even with no collisions, a frame may have to wait for the wire to be clear before being sent. This is called deferral.

Example:

Suppose the Ethernet statistics for the RMC ENET show that the highest number of collisions for any frame is six (6). How long of a delay is this? Given that the RMC ENET runs at 10 Mbps, the above chart shows that a frame will six collisions would have been successfully delivered between 0.077 ms and 7.86 ms from when it was intended to be sent.

Computing Utilization for RMC/ControlLogix Ethernet Networks

In order to predict the probability of collisions on collision domains of an Ethernet network used by EtherNet/IP between a ControlLogix 1756-ENET or ENBT and RMCs, we must compute the utilization on the various collision domains. The first step in computing utilization is to compute the bandwidth requirement for the collision domain. That is, how many frames are vying for the bandwidth each second?

There are two types of frames used in an I/O connection with an RMC: frames consumed by the

RMC, and frames produced by the RMC. Each RMC will produce exactly one frame each RPI, but it will consume one frame per connection each RPI. The bandwidth requirement on each collision domain reached by a consumed or produced frame is 1/RPI.

When a hub is used, there is only one collision domain, so all produced and consumed frames affect the required bandwidth for that domain.

When a switch is used, frames are routed intelligently only to the ports that need them. Frames consumed by the RMC are sent directly from the 1756-ENET/ENBT to the RMC, and thereby only affect only the collision domains of the two devices involved (1756-ENET/ENBT to switch, and switch to RMC). Frames produced by the RMC are multicast. For most inexpensive switches treat multicast packets the same as broadcast packets by forwarding the frames to all ports on the switch. Switches that support IGMP (Internet Group Management Protocol) or IGMP snooping do better by keeping track of which ports have devices that care about the multicast frame and only forwarding to those ports. However, notice that most of these switches require a router to be present in the network for the IGMP snooping to work. Therefore, if your system will not always have a router present, then you should purchase a switch that supports IGMP snooping without a router present. Compare the first two examples to see the difference this can make on utilization.

Note: Delta always recommends using a switch over a hub. The price difference is negligible and the determinism boost is significant. When using a protocol that makes heavy use of multicasting (as EtherNet/IP does), Delta also recommends using a switch that supports IGMP snooping, preferably without requiring an additional router. EtherNet/IP is the only protocol that the RMC ENET currently supports that uses multicast.

The bandwidth required in frames/second for each collision domain is computed by adding up frames/RPI for each different RPI on that collision domain.

Example:

Using the above information, we will give examples of computing the bandwidth required for the most common collision domains in an Ethernet network. For each of these examples assume there are two ControlLogix PLCs (CLX1 and CLX2), and four RMCs (RMC1 through RMC4). Assume that RMC1 and RMC2 are controlled by CLX1 and monitored by CLX2, RMC3 is controlled by CLX1 only, and RMC4 is controlled by CLX2 only. The RPI is 10 ms for RMC1 and RMC2, 5 ms for RMC3, and 7 ms for RMC4. This is summarized in the following chart:

	CLX1	**CLX2**
RMC1	10.0 ms	10.0 ms
RMC2	10.0 ms	10.0 ms
RMC3	5.0 ms	--
RMC4	--	7.0 ms

For the first two collisions domains types, we will assume all devices are connected to a switch. For the third, we will assume all devices are connected to a hub.

- The collision domain from the switch to an RMC ENET.

When the switch does not support IGMP, this collision domain will receive all frames consumed by the RMC, plus all frames produced by any RMC on the network.

When the switch does support IGMP, then this collision domain will receive all frames consumed and produced by the RMC. Multicast packets sent by other RMCs are filtered out by the IGMP-capable switch.

Example (IGMP not supported by switch):

In the example above, there are four RMC/switch collision domains: one each for RMC1 to RMC4. All four of these segments will include the four frames produced by the four RMCs. The bandwidth required for these is as follows:

Produced Frames/Second = 2 / 0.010s + 1 / 0.005s + 1 / 0.007s = 543

Next, we need to add the frames consumed by each RMC. This can be different for each RMC/switch collision domain.

RMC1 Frames/Second = 543 + 2 / 0.010s0 = 743

RMC2 Frames/Second = 543 + 2 / 0.010s = 743

RMC3 Frames/Second = 543 + 1 / 0.005s = 743

RMC4 Frames/Second = 543 + 1 / 0.007s = 686

Example (IGMP supported by switch):

In the example above, there are four RMC/switch collision domains: one each for RMC1 to RMC4. The bandwidth of each is computed based on the number of frames consumed by the RMC (one or two in this example) and the number of frames produced by the RMC (always one) each RPI:

RMC1 Frames/Second = (2 + 1) / 0.010s = 300

RMC2 Frames/Second = (2 + 1) / 0.010s = 300

RMC3 Frames/Second = (1 + 1) / 0.005s = 400

RMC4 Frames/Second = (1 + 1) / 0.007s = 286

- The collision domain from the switch to the 1756-ENET/ENBT. It should be noted first that if a 1756-ENBT is used and the switch supports full-duplex Ethernet, then there will be no collisions, and therefore this collision domain need not be considered.

When the switch does not support IGMP, this collision domain sees all frames produced by the ControlLogix for an RMC, plus all frames produced by any RMC on the network.

When the switch does support IGMP, this collision domain sees all frames produced by the ControlLogix for any RMC, plus all frames produced by any RMC destined for this ControlLogix.

Example (IGMP not supported by switch):

The ControlLogix/switch collision domains include all frames produced by any RMC on the network. This was computed for the previous collision domain: 543 frames/second.

In addition to these frames, the ControlLogix/switch collision domains also include frames produced by the ControlLogix and consumed by the RMCs. CLX1 produces frames for RMC1, RMC2, and RMC3. CLX2 produces frames for RMC1, RMC2, and RMC4. Therefore, the bandwidth for the CLX1/switch and CLX2/switch collision domains are as follows:

CLX1 Frames/Second = 543 + 2 / 0.010s + 1 / 0.005s = 943

CLX2 Frames/Second = 543 + 2 / 0.010s + 1 / 0.007s = 886

Notice that if a 1756-ENBT is used and full-duplex is used, then no collisions will occur. If half-duplex is used, but 100 Mbps is used, then the utilization will be 1/10th that of 10 Mbps.

Example (IGMP supported by switch):

There are two ControlLogix/switch collision domains. Each sees frames produced for RMCs by its ControlLogix and frames consumed by that ControlLogix. Therefore, this will be two frames per RMC at each RPI:

CLX1 Frames/Second = 4 / 0.010s + 2 / 0.005s = 800

CLX2 Frames/Second = 4 / 0.010s + 2 / 0.007s = 686

- The collision domain for a network using a hub instead of a switch.

This collision domain will receive all frames consumed and produced by all RMCs.

Example:

This hub collision domain includes all frames. Therefore, we add the bandwidth required by RMC-produced frames (543 frames/second) to the bandwidth required by all frames consumed by RMCs:

Hub Frames/Second = 543 + 4 / 0.010s + 1 / 0.005s + 1 / 0.007s = 1,286

As stated above, utilization is a function of the available bandwidth (10 or 100 Mbps) and the amount of data vying for the bandwidth. The amount of data includes the number and size of frames being sent.

The available bandwidth for a 10 Mbps Ethernet network with 1,792-bit frames and the mandatory 96-bit inter-frame gap is 5,296 frames per second. The 1,792 number is the maximum frame size used in an RMC I/O connection. The available bandwidth on a 100 Mbps Ethernet network is ten times that of 10 Mbps, or 52,966 frames/second.

Finally, use the above frames/second results to compute the utilization by dividing the actual bandwidth requirement by the maximum bandwidth and multiplying by 100%:

Utilization = 100% x actual bandwidth / maximum bandwidth

As noted above, the maximum bandwidths for 10 and 100 Mbps networks are 5,296 and 52,966 frames/second respectively.

Example:

Using the same setup as the previous example and assuming a switch is used, the utilization of the six collisions domains are as follows:

RMC1-3/switch Utilization = 100% x 743 / 5,296 = 14.0%

RMC4/switch Utilization = 100% x 686 / 5,296 = 13.0%

The ControlLogix segments can run at either 10 or 100 Mbps (1756-ENBT only). If 10 Mbps is used, then the utilization would be as follows:

CLX1/switch Utilization = 100% x 943 / 5,296 = 17.8%

CLX2/switch Utilization = 100% x 886 / 5,296 = 16.7%

If 100 Mbps is used, then the utilization would be as follows:

CLX1/switch Utilization = 100% x 943 / 52,966 = 1.8%

CLX2/switch Utilization = 100% x 886 / 52,966 = 1.7%

Again, if full-duplex is used with the 1756-ENBT, there will be no collisions in on the ControlLogix collision domains.

Controlling Collisions

If you are experiencing or anticipate unacceptable levels of collisions, then you have several options for reducing the number of collisions:

- Replace hubs with switches.
- Replace hubs and non-IGMP capable switches with IGMP capable switches.
- Replace half-duplex components with full-duplex components.
- Replace 10 Mbps with 100 Mbps components.
- Increase the RPI.
- Split the network into two or more smaller networks.

For further details, see Setting Up Large EtherNet/IP Networks.

How not to Control Collisions

Do NOT set the Ethernet switch port to the RMC100 to full-duplex. This is why:

1. A collision is defined on a half-duplex 10/100baseT Ethernet segment as happening when the Tx wire pair and Rx wire pair are active at the same time. Notice that there is no electrical contention

on 10/100baseT during a collision like there was on a truly shared media like Coax (10base2) Ethernet.

2. When a device says that it is Half Duplex, it is saying that its internal Ethernet controller cannot handle receiving data on the Rx pair at the same time it transmits on the Tx pair.
3. Ethernet has a protocol for half-duplex communication to incur minimal delay on this conceptually shared media.
4. There are 3 pieces to this scheme: (1) before sending data on a half-duplex segment, the device (switch or RMC) must wait until its partner is not sending, (2) if the Rx pair is idle, then the send begins, but the sender monitors for a collision during the first 512 bits, (3) collision during this phase is perfectly NORMAL and triggers a quick retry (see our help topic on just how short this is).
5. So when both sides (switch and RMC in this case) play by the rules, then only one side sends at a time, and when they happen to start talking at the same time, then a timely retry is done automatically.

So, what happens when the switch decides to NOT play by the rules? (i.e. the switch thinks the link is full duplex, but the RMC is still limited to half duplex)

1. The switch will receive any data sent by the RMC without watching to see if the Rx lines are busy when it sends on the Tx lines.
2. Because the switch sends data to the RMC without looking for a collision, the packet may be discarded by the RMC's Ethernet chips, while the switch is blissfully unaware that the packet was thrown away, so it WILL NEVER BE RETRIED! Further, it may or may not interrupt the packet going from the RMC to the switch, causing missing packets in the other direction.

In review, all that is accomplished by setting the switch to Full Duplex is it turns off REPORTING of collisions on the switch side, and makes matters worse by trading normal (quickly retried) collisions for worse errors that do not resend the packet. Because EtherNet/IP is a robust protocol, it can handle a few missed I/O packets without a hiccup. However, if one of the packets that was thrown away was from an MSG block, then it could be multiple seconds before it gets retried by the protocol! A properly configured Ethernet system should never see Jabber, Underrun, Late Collisions, Bad CRC, Extra Data, or Runt errors.

5.2.6.3.7.6 Setting Up Large EtherNet/IP Networks

Most examples given thus far show only small EtherNet/IP networks. They do so for simplicity. Larger EtherNet/IP networks are possible as well. This topic discusses a few issues that come to the forefront with large systems. The terms "large" and "small" are ambiguous, but similarly, the concepts really apply to both large and small networks. An example of a small systems is one to four RMCs connected to a single 1756-ENET or 1756-ENBT. An example of a large system is 40 RMCs connected to one or, ideally, more 1756-ENBT modules.

- **Reduce Bandwidth Usage to Actual Requirements.**
  Suppose you need to control 40 RMCs from a single ControlLogix 1756-L1. If you use a single 1756-ENBT for this task, the bandwidth required for this system at a 5 ms RPI is calculated as follows:

Frames/Second	=	(2 x connections) / RPI
	=	(2 x 40) / 0.005 s
	=	16,000

This is well over the allowed 4500 frames/second on the 1756-ENBT. However, if the system can get by with a slower RPI, the bandwidth drops dramatically. For example, increasing the RPI from 5 to 20 ms reduces the bandwidth requirement as follows:

Frames/Second	=	(2 x connections) / RPI
	=	(2 x 40) / 0.020 s
	=	4,000

In many cases, the reduction in bandwidth comes at no cost to the system performance because it may be likely that the ControlLogix is unable to scan its ladder logic more frequently than every 30 ms, in which case most 5 ms updates are ignored.

Notice that in some applications, the RPI required for different RMCs may be different. For example, suppose that three of this group of forty require a 5 ms RPI, but the rest can get by with a 25 ms RPI. Since the RPIs are set independently for each RMC, the bandwidth can be further reduced as follows:

Frames/Second	=	(2 x connections) / RPI + (2 x connections) / RPI
	=	(2 x 3) / 0.005 s + (2 x 37) / 0.025 s
	=	1,200 + 2,960
	=	4,160

- **Divide the Network.**
  You can further improve the stability and determinism of your system by dividing up the system into multiple networks. For example, we could reduce the near-maximum load of 4,000 frames/second to a comfortable 2,000 frames/second by adding a second 1756-ENBT module. By keeping the two networks separate (that is, use a separate switch for each), the collisions are also reduced drastically. For a large Ethernet system, the additional cost of a second, third, or even fourth 1756-ENBT module (around $1000 each as of this writing) is often insignificant compared to the total cost of the system and gives much higher reliability.

- **Upgrade to Smarter Switches.**
  EtherNet/IP utilizes IP multicasting, and as such uses a protocol called IGMP (Internet Group Management Protocol). Most low-cost switches do not utilize IGMP to control which ports care about the multicast packets, but instead broadcast multicast packets to all ports. This increases

the load on all segments of the network, which will increase collisions on half-duplex segments and increase the possibility of overruns. By choosing switches that support IGMP (often called IGMP Snooping), you can reduce the load on your network components and thereby increase your determinism. However, notice that most of these switches require a router to be present in the network for the IGMP snooping to work. Therefore, if your system will not always have a router present, then you should purchase a switch that supports IGMP snooping without a router present.

5.2.6.4 Modicon Quantum

5.2.6.4.1 Using Modicon PLCs with the RMC Ethernet Module

As of this writing, Modicon does not have Ethernet built in to any of its PLCs. However, it does have an Ethernet TCP/IP module which can be added to the Quantum PLC backplane. Modicon also is rumored to be working on a Momentum PLC with Ethernet TCP/IP capabilities. Our understanding is that both PLCs will use the same method to communicate over Ethernet.

The module required for communicating via Ethernet with the Modicon Quantum is called the Modicon Quantum Ethernet TCP/IP Module (140 NOE 211 00). It fits directly into the Modicon Quantum backplane.

The Modicon Quantum Ethernet TCP/IP requires the following versions of Modicon products:

- Quantum Executive 2.0
- Modsoft 2.31
- Concept 2.0
- Modlink 2.0

If you are adding Ethernet TCP/IP to an existing Modicon Quantum PLC with the older Executive, you will need to upgrade to the new Quantum Executive, which may require converting existing Modsoft and Concept projects.

Follow the instructions in the Modicon Quantum Ethernet TCP/IP Module User Guide (840 USE 107 00) for installing and configuring the module.

If you need help setting up your network, either consult your network administrator, or for simple stand-alone networks, see Setting up a Stand-alone TCP/IP Control Network.

The MaSTeR (MSTR) ladder logic block is used to read and write registers in remote devices over Ethernet. The RMC has 65536 registers that can be read and/or written. Refer to the RMC Register Map (Modbus/TCP and Modbus/RTU) topic for a map of the register addresses. Refer to the following topics for details and examples on using the MSTR block:

- Using the MSTR Modicon Ladder Logic Block
- MSTR Block Read Operation
- MSTR Block Write Operation
- MSTR Block Error Codes

5.2.6.4.2 RMC Register Map (Modbus/TCP and Modbus/RTU)

Tip: RMCWin's Address Tool provides an easy way to identify addresses in the RMC. Simply open the Address Tool and then move the cursor to any field in RMCWin that represents an RMC Register, and the Address Tool will display the address in the address format of your choice. See Address Tool for details.

The RMC module has 64K (65536) 16-bit registers that can be read from or written to over Ethernet, Modbus Plus, and PROFIBUS-DP. Each register is assigned an address. However, under the different communication methods, different addressing schemes are used. This topic describes the addressing over Modbus/TCP and Modbus/RTU. For details on addressing from other modules refer to the following topics:

- RMC Register Map (Allen-Bradley)
- RMC Register Map (Automationdirect.com)
- RMC Register Map (Omron FINS)
- RMC Register Map (Siemens TI505)
- RMC Register Map (Siemens S7)
- RMC Register Map (Modbus Plus)
- RMC Register Map (PROFIBUS-DP Message Mode)

The Modicon Quantum requires the Ethernet TCP/IP (140 NOE 211 00) module in order to communicate over Ethernet with the RMC. For details on using this device see Using Modicon PLCs with the RMC Ethernet Module. For details on using the Modbus/RTU serial protocol, see Using Modbus/RTU with the RMC SERIAL.

Note: When entering register addresses into an MSTR block, do not add 40000 or 400000 to indicate holding registers. MSTR blocks expect the addresses to start at 1, rather than 40001 or 400001.

Other devices such as the Control Microsystems SCADAPack with the 5905 Ethernet Gateway can also communicate with Modbus/TCP.

Status Registers:
These registers can only be read; writes are ignored.

Modbus Address	Register Description
1	Axis 0 Command Position
2	Axis 0 Target Position
3	Axis 0 Actual Position
4	Axis 0 Transducer Counts

5	Axis 0 Status Word
6	Axis 0 Drive
7	Axis 0 Actual Speed
8	Axis 0 Null Drive
9	Axis 0 Event Step
10	Axis 0 Link Value
11-20	Same as above but for axis 1
21-30	Same as above but for axis 2
31-40	Same as above but for axis 3
41-50	Same as above but for axis 4
51-60	Same as above but for axis 5
61-70	Same as above but for axis 6
71-80	Same as above but for axis 7

Command Registers:
These registers can be read or written.

Modbus Address	Register Description
81	Axis 0 Mode Word
82	Axis 0 Acceleration
83	Axis 0 Deceleration
84	Axis 0 Speed
85	Axis 0 Command Value
86	Axis 0 Command
87-92	Same as above but for axis 1
93-98	Same as above but for axis 2
99-104	Same as above but for axis 3
105-110	Same as above but for axis 4
111-116	Same as above but for axis 5

117-122	Same as above but for axis 6
123-128	Same as above but for axis 7

Parameter Registers:

These registers can be read or written. Changes to these registers do not take effect until a Set Parameters (P) command is executed.

Modbus Address	Register Description
129	Axis 0 Configuration Word
130	Axis 0 Scale
131	Axis 0 Offset
132	Axis 0 Extend Limit
133	Axis 0 Retract Limit
134	Axis 0 Proportional Gain
135	Axis 0 Integral Gain
136	Axis 0 Differential Gain
137	Axis 0 Extend Feed Forward
138	Axis 0 Retract Feed Forward
139	Axis 0 Extend Acceleration Feed Forward
140	Axis 0 Retract Acceleration Feed Forward
141	Axis 0 Dead Band Eliminator
142	Axis 0 In Position Window
143	Axis 0 Following Error
144	Axis 0 Auto Stop
145-160	Same as above but for axis 1
161-176	Same as above but for axis 2
177-192	Same as above but for axis 3
193-208	Same as above but for axis 4
209-224	Same as above but for axis 5

225-240	Same as above but for axis 6
241-256	Same as above but for axis 7

Event Step Table Registers:
These registers can be read or written.

Modbus Address	**Register Description**
257	Step 0 Mode Word
258	Step 0 Acceleration
259	Step 0 Deceleration
260	Step 0 Speed
261	Step 0 Command Value
262	Step 0 Command/Commanded Axes
263	Step 0 Link Type/Link Next
264	Step 0 Link Value
257+n*8	Step n (0-255) Mode Word
258+n*8	Step n (0-255) Acceleration
259+n*8	Step n (0-255) Deceleration
260+n*8	Step n (0-255) Speed
261+n*8	Step n (0-255) Command Value
262+n*8	Step n (0-255) Command/Commanded Axes
263+n*8	Step n (0-255) Link Type/Link Next
264+n*8	Step n (0-255) Link Value

Input to Event Table Registers:
These registers can be read or written.

Modbus Address	**Register Description**
2305	Event Step for Axis 0 on Input 0 Rising Edge
2306	Event Step for Axis 1 on Input 0 Rising Edge

2307	Event Step for Axis 2 on Input 0 Rising Edge
2308	Event Step for Axis 3 on Input 0 Rising Edge
2309	Event Step for Axis 4 on Input 0 Rising Edge
2310	Event Step for Axis 5 on Input 0 Rising Edge
2311	Event Step for Axis 6 on Input 0 Rising Edge
2312	Event Step for Axis 7 on Input 0 Rising Edge
2313 + n	Event Step for Axes n (0-7) on Input 1 Rising Edge
:	:
2425 + n	Event Step for Axes n (0-7) on Input 15 Rising Edge
2433 + n	Event Step for Axes n (0-7) on Input 0 Falling Edge
:	:
2553 + n	Event Step for Axes n (0-7) on Input 15 Falling Edge

Status Map Registers:

This block of registers is only used by the Modbus Plus and PROFIBUS interfaces. Therefore, these registers are unused by this Ethernet protocol.

Modbus Address	Register Description
2561-2592	Status Map Entries

Plot Type Registers:

The plot type registers can be read or written. The values that are read indicate the extra plot information in the current graph. The values written to these registers tell the controller which extra plot information to obtain on the next plot. For these registers, the following values are used:

- 0: Extra position precision
- 1: Command and Command Value
- 2: Event Step and Link Value
- 3: Raw Transducer Counts
- 4: Internal Target and Actual Speeds
- 5: Integral Drive

For more information on these four types of plot information, see Selecting the Data to Plot and

Reading Plots from the Communication Module.

Modbus Address	Register Description
2625	Axis 0 plot type
2626	Axis 1 plot type
2627	Axis 2 plot type
2628	Axis 3 plot type
2629	Axis 4 plot type
2630	Axis 5 plot type
2631	Axis 6 plot type
2632	Axis 7 plot type

Digital (Discrete) I/O Registers:

These registers indicate the current state of the digital inputs and outputs. These registers may only be read; writes will be ignored, as this product does not support forcing inputs or outputs.

Because different PLCs label bit numbers differently, the following chart is provided to show the mapping between the devices:

	MSB								
RMC bit #	15	14	13	12	11	10	9	8	7
Modicon bit #	1	2	3	4	5	6	7	8	9

The bit numbers given in the table below are in RMC format (0 is LSB, 15 is MSB):

Modbus Address	Register Description
2633	CPU Digital Inputs 0 and 1 in LSBs of low byte, Outputs 0 and 1 in LSBs of high byte
2634	Unused
2635	Unused
2636	Sensor Digital I/O Inputs 0-15
2637	Sensor Digital I/O Inputs 16-17 (stored to two LSBs)

2638	Sensor Digital I/O Outputs 0-7 in high byte (low byte unused)
2639	Unused
2640	Unused

Plot Time Registers:

The Plot Time interval is configurable on the RMC. This interval indicates the number of control loops between each sample in a plot. Therefore, if the control loop is 0.976ms (e.g. RMC100-M1), this indicates roughly the number of milliseconds between samples. If the control loop is 1.953ms (e.g. RMC100-M4), this indicates half of the number of milliseconds between samples.

These registers may be read or written. When read, they indicate the plot interval of the currently gathered plot. When written, they set the plot interval the RMC should use for the next plot it will gather. When the RMC starts the next plot, it copies the requested plot interval into the currently used plot interval.

Modbus Address	Register Description
2641	Axis 0 plot time interval
2642	Axis 1 plot time interval
2643	Axis 2 plot time interval
2644	Axis 3 plot time interval
2645	Axis 4 plot time interval
2646	Axis 5 plot time interval
2647	Axis 6 plot time interval
2648	Axis 7 plot time interval

Last Parameter Error Registers:

Note: To use these registers through Ethernet, you must have RMC100 control firmware dated 19990715 or later and Ethernet program dated 19990702 or later.

Each of these read-only registers holds the number of the last parameter error generated on an axis. This is useful for determining the cause of the Parameter Error bit in the status word. For a description of the values read from these registers, see Parameter Error Values.

Modbus Address	Register Description
2649	Last parameter error on axis 0

2650	Last parameter error on axis 1
2651	Last parameter error on axis 2
2652	Last parameter error on axis 3
2653	Last parameter error on axis 4
2654	Last parameter error on axis 5
2655	Last parameter error on axis 6
2656	Last parameter error on axis 7

Firmware Date Registers:

Note: To use these registers through Ethernet, you must have RMC100 control firmware dated 19990715 or later and Ethernet program dated 19990702 or later.

These read-only registers hold information about the firmware versions in the RMC100 CPU module. The Boot and Loader firmware versions have no effect on the actual performance of the RMC and therefore can usually be ignored.

Modbus Address	Register Description
2657	Boot firmware month (MSB) and day (LSB)
2658	Boot firmware year
2659	Loader firmware month (MSB) and day (LSB)
2660	Loader firmware year
2661	Control firmware month (MSB) and day (LSB)
2662	Control firmware year
2663	Control firmware Beta Code. This will be 0 for standard release firmware, 'B' for Beta firmware, or 'SI' for Superimposed firmware.
2664	Feature code. This register is mainly reserved for internal use but does have two bits that may be useful to some users: • If bit 1 (value 0x0002) is set, the control loop is 2 ms, otherwise the control loop is 1 ms. • If bit 0 (value 0x0001) is set, a sensor DI/O is present, otherwise there is no sensor DI/O.

Reserved Registers:

Reading these values will return zero, and writes are ignored.

Modbus Address	Register Description
2665-12288	Unused

Spline Download Area:

These registers are write only. Reading them will return zero. This area is used to download intervals and points in a spline. This is a much more efficient alternative to using individual New Spline Point and Set Spline Interval/End Segment commands. For details on using this Spline Download Area, see Downloading Splines to the RMC.

Modbus Address	Register Description
12289-16384	Spline Download Area

Plot Registers:

These registers can be read only; writes are ignored.

Note: Reading plots is not a trivial task; for further details, see Reading Plots from the Communication Module.

Modbus Address	Register Description
16385-22528	Plot data for axis 0
22529-28672	Plot data for axis 1
28673-34816	Plot data for axis 2
34817-40960	Plot data for axis 3
40961-47104	Plot data for axis 4
47105-53248	Plot data for axis 5
53249-59392	Plot data for axis 6

59393-65536	Plot data for axis 7

5.2.6.5 Omron CS1 and CV PLCs

5.2.6.5.1 Using Omron PLCs with the RMC ENET

Overview

The CS1 and CV families of PLCs from Omron Electronics Inc. can be used to control the RMC over Ethernet. These PLCs require the ETN01 Ethernet Unit and use Omron's FINS protocol to communicate with the RMC.

Note: Ethernet communication with the Omron PLCs requires RMC ENET firmware dated 20001023 or later.

Note: The documentation below assumes the user is familiar with Omron PLC programming, and instead focuses on how to initiate reads and writes to an RMC.

The FINS protocol uses a three-stage addressing system: network address, node number, and unit number. These three address elements have the following purposes:

- **Network Address:**
  This value identifies the network on which the target node resides. The PLC looks up this number in its Local and Remote Network Tables to determine which local unit number should be used to send out the request. For Ethernet systems, typically only the Local Network Table is used to map a network address (for example, 1) to the unit number on the ETN01 Ethernet Unit (for example, 0). See your Omron PLC and/or CX-Programmer documentation for details on setting up the Local and Remote Network Tables.

- **Node Number:**
  This value identifies the node on the network given by the network address. For Ethernet networks, this number must be converted to an IP address. This conversion is done by the ETN01 Ethernet Unit. Three methods are available. In the Automatic Address Generation method, the IP address is derived by combining the ETN01's IP Network Address (e.g. 192.168.0.0) with the Node Number (e.g. 5) to get an IP Address (e.g. 192.168.0.5). In the IP Address Table method, each Node Number maps to an IP Address via an entry in an IP Address Table. In the Combined method, the IP Address Table is used first, but if the Node Number is not found in the table, then the Automatic Address Generation method is used. See the Omron ETN01 Ethernet Unit manual for details on these methods.

- **Unit Number:**
  This value identifies the unit number on the rack of the remote node identified by the Network Address and Node Number.

The RMC does not require manually setting these values in the RMC. Only the IP address needs to be set in the RMC; see RMC Ethernet IP Address Setup for details. See the discussion below on setting up the RECV and SEND instructions for details on what values to use for the Network, Node, and Unit numbers when communicating with an RMC.

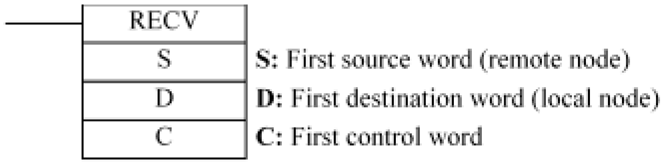
The RMC has 64K registers. When accessed by Omron PLCs, this data appears in Data Memory (DM) registers D0 to D16383, and Extended Data Memory (EM) registers En_0 to En_6143, where n is the bank number from 0 through 7. For CPUs that do not support some or all of these

Extended Data Memory Banks, the only functionality that will be lost is reading up complete plot data. The CV500 PLC also cannot read data beyond D8191, however, the only feature that is lost by this limitation is access to the Spline Download Area. See RMC Register Map (Omron FINS) for details.

The Omron CS1 and CV PLCs use the Network Receive (RECV) and Network Send (SEND), instructions to read and write registers in remote devices such as the RMC over FINS. The Omron PLC instruction manuals are very well written and should suffice as a reference for setting up this communication. However, a brief synopsis of each of these instructions is provided below, followed by an example.

Network Receive: RECV(098) for CS1, RECV(193) for CV

This instruction is used to read data from the RMC's registers into the PLC's registers. It has the following format:


RECV
S
D
C
S: First source word (remote node)
D: First destination word (local node)
C: First control word

RECV should be energized for only one scan to start the communication. After that, the Communications Port Enabled Flag (A202.00 to A202.07 on the CS1 family and A502.00 to A502.07 on the CV family) should be used to determine when the operation is complete, as described in the Omron documentation. Each parameter is described below:

S First Source Word in Remote Node. Give the starting address to read from in the RMC. This must be either a DM (Dnnnnn) or EM (Eb_nnnnn) address. The current extended data memory bank on the RMC is always set to bank 0. Refer to the RMC Register Map (Omron FINS) topic for exact addresses.

D First Destination Word in Local Node. Give the starting address in the PLC to store the data read from the RMC. This is typically a DM address.

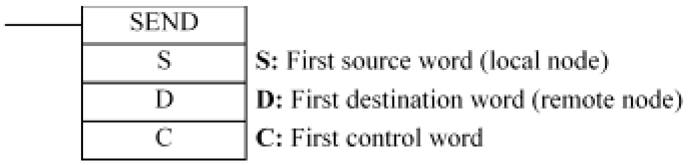
C First Control Word. This instruction uses a block of five control words. These words are used to control the network communication itself. These five words are defined as follows:

Word	MSB (Bits 8-15)	LSB (Bits 0-7)
C	**Number of words to read (1 to 512).** The PLC supports reads up to 990 words, but the RMC is limited to 512 words.	
C+1	**Always 00 for RMCs.**	**Remote Network Address.** For RMCs, this value should be set to the address of the RMC's Ethernet network.
C+2	**Remote Node Number.** For RMCs, this value should be set to whichever Node Number will be mapped to the RMC's IP address.	**Remote Unit Number.** For RMCs, this value should be 0.
C+3	**Port Number: 00 to 07.** The Port Number is used to	**No. of Retries: 00 to 0F (0 to 15).**

	allow simultaneous communications in the PLC. Use a different number for each communication that may be requested simultaneously.	For RMC communications, this value should be between 2 and 5.
C+4	**Timeout: 0001 to FFFF (0.1 to 6553.5 seconds).** The default setting of 0000 sets a monitoring time of 2 seconds. For RMC communications, this value should be set to 0001 or 0002 for 0.1 or 0.2 second timeouts.	

Network Send: SEND(090) for CS1, SEND(192) for CV

This instruction is used to write data from the PLC's registers to the RMC's registers. It has the following format:


SEND
S
D
C
S: First source word (local node)
D: First destination word (remote node)
C: First control word

SEND should be energized for only one scan to start the communication. After that, the Communications Port Enabled Flag (A202.00 to A202.07 on the CS1 family and A502.00 to A502.07 on the CV family) should be used to determine when the operation is complete, as described in the Omron documentation. Each parameter is described below:

S First Source Word in Local Node. Give the starting address in the PLC of the data to write to the RMC. This is typically a DM address.

D First Destination Word in Remote Node. Give the starting address in the RMC of the place to write the data. This must be a DM address (Dnnnnn). None of the RMC's EM banks can be written to. Refer to the RMC Register Map (Omron FINS) topic for exact addresses.

C First Control Word. This instruction uses a block of five control words. These words are used to control the network communication itself. These five words are defined as follows:

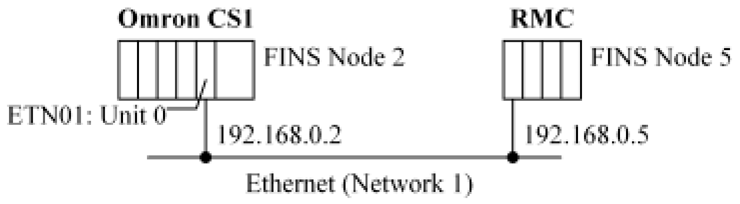
Word	MSB (Bits 8-15)	LSB (Bits 0-7)
C	**Number of words to write (1 to 512).** The PLC supports writes up to 990 words, but the RMC is limited to 512 words.	
C+1	**Always 00 for RMCs.**	**Remote Network Address.** For RMCs, this value should be set to the address of the RMC's Ethernet network.
C+2	**Remote Node Number.** For RMCs, this value should be set to whichever Node Number will be mapped to the RMC's IP address.	**Remote Unit Number.** For RMCs, this value should be 0.

C+3	**Port Number: 00 to 07.** The Port Number is used to allow simultaneous communications in the PLC. Use a different number for each communication that may be requested simultaneously.	**No. of Retries: 00 to 0F (0 to 15).** For RMC communications, this value should be between 2 and 5.
C+4	**Timeout: 0001 to FFFF (0.1 to 6553.5 seconds).** The default setting of 0000 sets a monitoring time of 2 seconds. For RMC communications, this value should be set to 0001 or 0002 for 0.1 or 0.2 second timeouts.	

CS1 Example

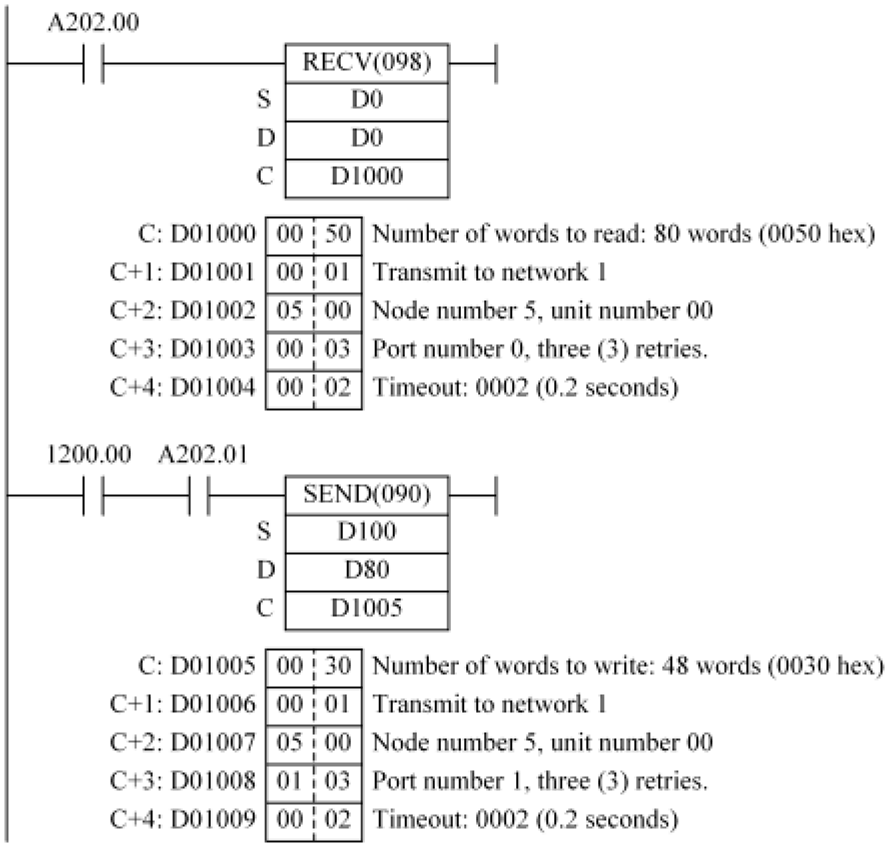
The user has an Omron CS1 PLC with a CS1W-ETN01 Ethernet Unit and an RMC-ENET. The Omron PLC has just one network, which is the Ethernet network accessible through an ETN01 Ethernet Unit. This network has Network Address 1. The ETN01 Ethernet Unit has Unit Number 0. The Omron PLC has IP address 192.168.0.2, and the RMC has IP address 192.168.0.5. The Omron ETN01 is set up to use Automatic Address Generation, so the RMC has Node Number 5, and the ETN01 has node number 2.

The following diagram demonstrates the configuration:



The Omron PLC must have one entry in its Local Network Table. This entry assigns Network Address 1 (the Ethernet network) to Unit Number 0 (the ETN01 Ethernet Unit).

The user wishes to continuously read the 80 status words starting at address D0 in the RMC and store them at D0 in the PLC. The user also wishes to write 48 words from D100-D147 to the RMC's command registers D80-D127 whenever the 1200.00 coil is set. The following ladder logic does this:



A202.00
RECV(098)
S D0
D D0
C D1000
C: D01000 00 50 Number of words to read: 80 words (0050 hex)
C+1: D01001 00 01 Transmit to network 1
C+2: D01002 05 00 Node number 5, unit number 00
C+3: D01003 00 03 Port number 0, three (3) retries.
C+4: D01004 00 02 Timeout: 0002 (0.2 seconds)
1200.00 A202.01
SEND(090)
S D100
D D80
C D1005
C: D01005 00 30 Number of words to write: 48 words (0030 hex)
C+1: D01006 00 01 Transmit to network 1
C+2: D01007 05 00 Node number 5, unit number 00
C+3: D01008 01 03 Port number 1, three (3) retries.
C+4: D01009 00 02 Timeout: 0002 (0.2 seconds)

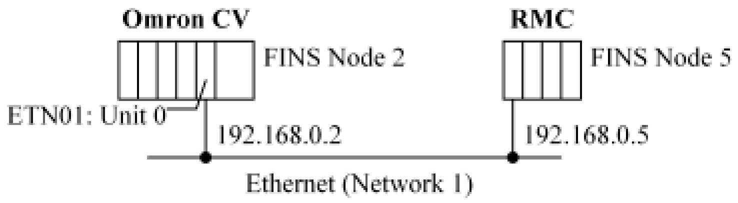
In this example, the RECV(098) instruction will be triggered each time the Communication Port 0 Enabled Flag (A202.00) is set. This flag will be set any time port 0 is not busy. Therefore, the PLC will read these registers from the RMC continuously as fast as it can. Depending on the load of the PLC, this will read the RMC's status as often as every 18 ms.

The SEND(090) instruction uses port 1. It is important that this is a different port number from the RECV(098) instruction. It is possible to use the same port number for two network instructions, but this reduces performance and requires additional synchronization ladder logic. Examples of doing so are given in the Omron documentation.

CV Example

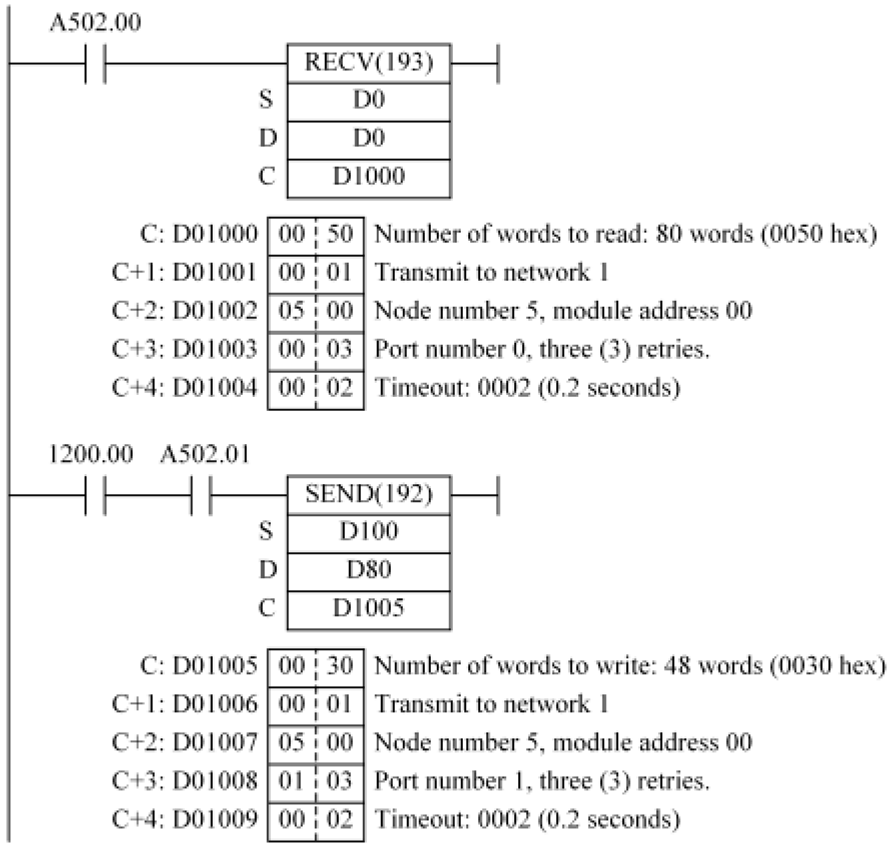
The user has an Omron CV PLC with a CV500-ETN01 Ethernet Unit and an RMC-ENET. The Omron PLC has just one network, which is the Ethernet network accessible through an ETN01 Ethernet Unit. This network has Network Address 1. The ETN01 Ethernet Unit has Module Address (equivalent to Unit Number in the CS1) 0. The Omron PLC has IP address 192.168.0.2, and the RMC has IP address 192.168.0.5. The Omron ETN01 is set up to use Automatic Address Generation, so the RMC has Node Number 5, and the ETN01 has node number 2.

The following diagram demonstrates the configuration:



The Omron PLC must have one entry in its Local Network Table. This entry assigns Network Address 1 (the Ethernet network) to Module Address 0 (the ETN01 Ethernet Unit).

The user wishes to continuously read the 80 status words starting at address D0 in the RMC and store them at D0 in the PLC. The user also wishes to write 48 words from D100-D147 to the RMC's command registers D80-D127 whenever the 1200.00 coil is set. The following ladder logic does this:


A502.00
RECV(193)
S D0
D D0
C D1000
C: D01000 00 50 Number of words to read: 80 words (0050 hex)
C+1: D01001 00 01 Transmit to network 1
C+2: D01002 05 00 Node number 5, module address 00
C+3: D01003 00 03 Port number 0, three (3) retries.
C+4: D01004 00 02 Timeout: 0002 (0.2 seconds)
1200.00
A502.01
SEND(192)
S D100
D D80
C D1005
C: D01005 00 30 Number of words to write: 48 words (0030 hex)
C+1: D01006 00 01 Transmit to network 1
C+2: D01007 05 00 Node number 5, module address 00
C+3: D01008 01 03 Port number 1, three (3) retries.
C+4: D01009 00 02 Timeout: 0002 (0.2 seconds)

In this example, the RECV(193) instruction will be triggered each time the Communication Port 0 Enabled Flag (A502.00) is set. This flag will be set any time port 0 is not busy. Therefore, the PLC will read these registers from the RMC continuously as fast as it can. Depending on the load of the PLC, this will read the RMC's status as often as every 18 ms.

The SEND(192) instruction uses port 1. It is important that this is a different port number from the RECV(193) instruction. It is possible to use the same port number for two network instructions, but this reduces performance and requires additional synchronization ladder logic. Examples of doing so are given in the Omron documentation.

5.2.6.5.2 RMC Register Map (Omron FINS)

Tip: RMCWin's Address Tool provides an easy way to identify addresses in the RMC. Simply open the Address Tool and then move the cursor to any field in RMCWin that represents an RMC Register, and the Address Tool will display the address in the address format of your choice. See Address Tool for details.

The RMC module has 64K (65536) 16-bit registers that can be read from or written to over Ethernet, Modbus Plus, and PROFIBUS-DP. Each register is assigned an address. However, under the different communication methods, different addressing schemes are used. This topic describes the addressing over Omron's FINS protocol, as used by the Omron CS1 and CV PLCs. For details on addressing from other modules refer to the following topics:

- RMC Register Map (Allen-Bradley)
- RMC Register Map (Automationdirect.com)
- RMC Register Map (Modbus/TCP and Modbus/RTU)
- RMC Register Map (Siemens TI505)
- RMC Register Map (Siemens S7)
- RMC Register Map (Modbus Plus)
- RMC Register Map (PROFIBUS-DP Message Mode)

The Omron CS1 and CV PLCs require the ETN01 Ethernet module in order to communicate with the RMC. For details on using this device, see Using Omron PLCs with the RMC ENET.

Other devices that communicate with FINS over Ethernet may also be able to communicate with the RMC.

Status Registers:
These registers can only be read; writes are ignored.

Omron Address	Register Description
D00000	Axis 0 Command Position
D00001	Axis 0 Target Position
D00002	Axis 0 Actual Position
D00003	Axis 0 Transducer Counts
D00004	Axis 0 Status Word
D00005	Axis 0 Drive
D00006	Axis 0 Actual Speed
D00007	Axis 0 Null Drive

D00008	Axis 0 Event Step
D00009	Axis 0 Link Value
D00010-D00019	Same as above but for axis 1
D00020-D00029	Same as above but for axis 2
D00030-D00039	Same as above but for axis 3
D00040-D00049	Same as above but for axis 4
D00050-D00059	Same as above but for axis 5
D00060-D00069	Same as above but for axis 6
D00070-D00079	Same as above but for axis 7

Command Registers:
These registers can be read or written.

Omron Address	**Register Description**
D00080	Axis 0 Mode Word
D00081	Axis 0 Acceleration
D00082	Axis 0 Deceleration
D00083	Axis 0 Speed
D00084	Axis 0 Command Value
D00085	Axis 0 Command
D00086-D00091	Same as above but for axis 1
D00092-D00097	Same as above but for axis 2
D00098-D00103	Same as above but for axis 3
D00104-	Same as above but for axis 4

D00109	
D00110-D00115	Same as above but for axis 5
D00116-D00121	Same as above but for axis 6
D00122-D00127	Same as above but for axis 7

Parameter Registers:

These registers can be read or written. Changes to these registers do not take effect until a Set Parameters (P) command is executed.

Omron Address	Register Description
D00128	Axis 0 Configuration Word
D00129	Axis 0 Scale
D00130	Axis 0 Offset
D00131	Axis 0 Extend Limit
D00132	Axis 0 Retract Limit
D00133	Axis 0 Proportional Gain
D00134	Axis 0 Integral Gain
D00135	Axis 0 Differential Gain
D00136	Axis 0 Extend Feed Forward
D00137	Axis 0 Retract Feed Forward
D00138	Axis 0 Extend Acceleration Feed Forward
D00139	Axis 0 Retract Acceleration Feed Forward
D00140	Axis 0 Dead Band Eliminator
D00141	Axis 0 In Position Window
D00142	Axis 0 Following Error
D00143	Axis 0 Auto Stop
D00144-D00159	Same as above but for axis 1

D00160-D00175	Same as above but for axis 2
D00176-D00191	Same as above but for axis 3
D00192-D00207	Same as above but for axis 4
D00208-D00223	Same as above but for axis 5
D00224-D00239	Same as above but for axis 6
D00240-D00255	Same as above but for axis 7

Event Step Table Registers:
These registers can be read or written.

Omron Address	**Register Description**
D00256	Step 0 Mode Word
D00257	Step 0 Acceleration
D00258	Step 0 Deceleration
D00259	Step 0 Speed
D00260	Step 0 Command Value
D00261	Step 0 Command/Commanded Axes
D00262	Step 0 Link Type/Link Next
D00263	Step 0 Link Value
D00256+n*8	Step n (0-255) Mode Word
D00257+n*8	Step n (0-255) Acceleration
D00258+n*8	Step n (0-255) Deceleration
D00259+n*8	Step n (0-255) Speed
D00260+n*8	Step n (0-255) Command Value
D00261+n*8	Step n (0-255) Command/Commanded Axes
D00262+n*8	Step n (0-255) Link Type/Link Next

D00263+n*8	Step n (0-255) Link Value

Input to Event Table Registers:
These registers can be read or written.

Omron Address	Register Description
D02304	Event Step for Axis 0 on Input 0 Rising Edge
D02305	Event Step for Axis 1 on Input 0 Rising Edge
D02306	Event Step for Axis 2 on Input 0 Rising Edge
D02307	Event Step for Axis 3 on Input 0 Rising Edge
D02308	Event Step for Axis 4 on Input 0 Rising Edge
D02309	Event Step for Axis 5 on Input 0 Rising Edge
D02310	Event Step for Axis 6 on Input 0 Rising Edge
D02311	Event Step for Axis 7 on Input 0 Rising Edge
D02312 + n	Event Step for Axes n (0-7) on Input 1 Rising Edge
:	:
D02424 + n	Event Step for Axes n (0-7) on Input 15 Rising Edge
D02432 + n	Event Step for Axes n (0-7) on Input 0 Falling Edge
:	:
D02552 + n	Event Step for Axes n (0-7) on Input 15 Falling Edge

Status Map Registers:
This block of registers is only used by the Modbus Plus and PROFIBUS interfaces. Therefore, these registers are unused by this Ethernet protocol.

Omron Address	Register Description
D02560-D02591	Status Map Entries

Plot Type Registers:

The plot type registers can be read or written. The values that are read indicate the extra plot information in the current graph. The values written to these registers tell the controller which extra plot information to obtain on the next plot. For these registers, the following values are used:

- 0: Extra position precision
- 1: Command and Command Value
- 2: Event Step and Link Value
- 3: Raw Transducer Counts
- 4: Internal Target and Actual Speeds
- 5: Integral Drive

For more information on these four types of plot information, see Selecting the Data to Plot and Reading Plots from the Communication Module.

Omron Address	Register Description
D02624	Axis 0 plot type
D02625	Axis 1 plot type
D02626	Axis 2 plot type
D02627	Axis 3 plot type
D02628	Axis 4 plot type
D02629	Axis 5 plot type
D02630	Axis 6 plot type
D02631	Axis 7 plot type

Digital (Discrete) I/O Registers:

These registers indicate the current state of the digital inputs and outputs. These registers may only be read; writes will be ignored, as this product does not support forcing inputs or outputs.

The Omron PLCs and the RMC use the same bit numbering. Therefore, bit numbers in the table below use 0 for the least-significant bit and 15 for the most-significant bit:

Omron Address	Register Description
D02632	CPU Digital Inputs 0 and 1 in LSBs of low byte, Outputs 0 and 1 in LSBs of high byte
D02633	Unused
D02634	Unused
D02635	Sensor Digital I/O Inputs 0-15

D02636	Sensor Digital I/O Inputs 16-17 (stored to two LSBs)
D02637	Sensor Digital I/O Outputs 0-7 in high byte (low byte unused)
D02638	Unused
D02639	Unused

Plot Time Registers:

The Plot Time interval is configurable on the RMC. This interval indicates the number of control loops between each sample in a plot. Therefore, if the control loop is 0.976ms (e.g. RMC100-M1), this indicates roughly the number of milliseconds between samples. If the control loop is 1.953ms (e.g. RMC100-M4), this indicates half of the number of milliseconds between samples.

These registers may be read or written. When read, they indicate the plot interval of the currently gathered plot. When written, they set the plot interval the RMC should use for the next plot it will gather. When the RMC starts the next plot, it copies the requested plot interval into the currently used plot interval.

Omron Address	Register Description
D02640	Axis 0 plot time interval
D02641	Axis 1 plot time interval
D02642	Axis 2 plot time interval
D02643	Axis 3 plot time interval
D02644	Axis 4 plot time interval
D02645	Axis 5 plot time interval
D02646	Axis 6 plot time interval
D02647	Axis 7 plot time interval

Last Parameter Error Registers:

Note: To use these registers through Ethernet, you must have RMC100 CPU control firmware dated 19990715 or later.

Each of these read-only registers holds the number of the last parameter error generated on an axis. This is useful for determining the cause of the Parameter Error bit in the status word. For a description of the values read from these registers, see Parameter Error Values.

Omron Address	Register Description

D02648	Last parameter error on axis 0
D02649	Last parameter error on axis 1
D02650	Last parameter error on axis 2
D02651	Last parameter error on axis 3
D02652	Last parameter error on axis 4
D02653	Last parameter error on axis 5
D02654	Last parameter error on axis 6
D02655	Last parameter error on axis 7

Firmware Date Registers:

Note: To use these registers through Ethernet, you must have RMC100 CPU control firmware dated 19990715 or later.

These read-only registers hold information about the firmware versions in the RMC100 CPU module. The Boot and Loader firmware versions have no effect on the actual performance of the RMC and therefore can usually be ignored.

Omron Address	Register Description
D02656	Boot firmware month (MSB) and day (LSB)
D02657	Boot firmware year
D02658	Loader firmware month (MSB) and day (LSB)
D02659	Loader firmware year
D02660	Control firmware month (MSB) and day (LSB)
D02661	Control firmware year
D02662	Control firmware Beta Code. This will be 0 for standard release firmware, 'B' for Beta firmware, or 'SI' for Superimposed firmware.
D02663	Feature code. This register is mainly reserved for internal use but does have two bits that may be useful to some users: • If bit 1 (value 0x0002) is set, the control loop is 2 ms, otherwise the control loop is 1 ms. • If bit 0 (value 0x0001) is set, a sensor DI/O is present, otherwise there is no sensor DI/O.

Reserved Registers:

Reading these values will return zero, and writes are ignored.

Omron Address	Register Description
D02664-D12287	Unused

Spline Download Area:

These registers are write only. Reading them will return zero. This area is used to download intervals and points in a spline. This is a much more efficient alternative to using individual New Spline Point (X) and Set Spline Interval/End Segment (T) commands. For details on using this Spline Download Area, see Downloading Splines to the RMC.

Note: The Omron CV500 and CVM1-CPU01 PLCs are limited to addressing D0 to D8191, and therefore cannot use the Spline Download Area.

Omron Address	Register Description
D12288-D16383	Spline Download Area

Plot Registers:

These registers can only be read; writes are ignored.

Note: Reading plots is not a trivial task; for further details, see Reading Plots from the Communication Module.

Omron Address	Register Description
E0_00000-E0_06143	Plot data for axis 0
E1_00000-E1_06143	Plot data for axis 1
E2_00000-E2_06143	Plot data for axis 2
E3_00000-E3_06143	Plot data for axis 3
E4_00000-E4_06143	Plot data for axis 4
E5_00000-E5_06143	Plot data for axis 5

E6_00000-E6_06143	Plot data for axis 6
E7_00000-E7_06143	Plot data for axis 7

Note: Omron PLCs can only access as many Extended Data Memory banks as they have. Therefore, many Omron PLCs will only be able to access a limited number of the banks listed above, or may not be able to access the plot registers at all.

5.2.6.6 Rockwell Software RSView32

5.2.6.6.1 Using Rockwell Software's RSView32 with the RMC Ethernet Module

RSView32 is a PC-based Human-Machine Interface (HMI) software package published by Rockwell Software. Because of the RMC's support for Allen-Bradley PLC Ethernet protocols, the RMC can be used directly from RSView32 to provide information such as axis positions to the user directly instead of directing all the information through the PLC.

Note: For RSView32 Direct Driver support for the RMC, you will need RMC Ethernet firmware dated 20000501 or later.

RSView32 communicates through RSLinx, also from Rockwell Software, to the devices. Therefore, to set up a communication path from RSView32 to the RMC, you must first set up a driver between RSLinx and the RMC. Once the driver is configured, RSView32 can be configured to use it to connect to one or more RMCs. Each step will be described below. These steps were documented with RSLinx 2.10.166.0 and RSView32 6.20.49.

Step 1: Configure a Driver for the RMC in RSLinx

1. Start RSLinx.
2. On the Communications menu, click Configure Drivers.
3. Under Configured Drivers, check to see if there is an AB_ETH driver already installed. If there is an AB_ETH driver already installed, do the following:
    a. Under Configured Drivers, click on the driver to which you wish to add the RMC. Each driver can hold many devices.
    b. Click Configure.

   If there is no AB_ETH driver, do the following:
    a. Under Available Driver Types, select Ethernet to PLC-5/SLC-5/5820-El.
    b. Click Add New.
    c. In the Add New RSLinx Driver dialog box, choose whatever name you want. This example will assume you enter ABETH-RMC. Click OK.

   Either of the above procedures will bring up a Configure driver for Ethernet to PLC-5/SLC-5/5820-El dialog box.
4. In the Current Mapping list, click a line without an address associated to it.
5. In the IP Address or hostname text box, enter the RMC TCP/IP address. For example,

192.168.0.23.

6. Click Accept.

7. Click OK.

Step 2: Test the RSLinx Driver

1. Start RSLinx.

2. On the Communications menu, click RSWho.

3. Click on the plus (+) sign to the left of the driver configured in the steps above (e.g. AB_ETH-RMC).

4. If the driver is working correctly, you should see a node under the driver with the RMC IP address, a device type of SLC-5/05, and a processor name of RMC100. If this does not work, then ensure you have the IP addresses set up correctly on the PC and RMC, and that you have RMC Ethernet firmware dated 20000501 or later. You can test the basic Ethernet communication by going to a DOS prompt and typing ping 192.168.0.23, but using the IP address of your RMC.

5. Close the RSWho window. Leaving RSWho open will consume network bandwidth.

Step 3: Configure the TCP/IP Channel in RSView32

This step is not required if you already have an RSView32 channel set up for the RSLinx driver configured in step 1.

1. Start RSView32 and open your project.

2. Open the RSView32 Channel configuration window. To do this, from the Project Manager window, select the Edit Mode tab, and double-click the Channel item from under the System node.

3. In the Channel list, click an unused channel.

4. In the Network Type list, click TCP/IP.

5. Under Primary Communication Driver, click the RSLinx driver configured in the steps above (e.g. AB_ETH-RMC).

6. Click OK.

7. Close the Project window.

Step 4: Add the RMC Node to RSView32

1. Start RSView32 and open your project.

2. Open the RSView32 Node configuration window. To do this, from the Project Manager window, select the Edit Mode tab, and double-click the Node item from under the System node.

3. In the Node window, ensure that you can see both the edit form at the top and the spreadsheet at the bottom. Use the View menu's Form and Spreadsheet commands to display both.

4. In the spreadsheet at the bottom of the Node window, click an empty line. This ensures that you add a new node rather than edit an existing node.

5. Under Data Source, click the Direct Driver option.

6. In the Name text box, type the name you want to use for this node (for example, RMC100).

7. In the Channel drop-down list, click the TCP/IP channel created above.

8. For the Station text box, type the IP address of the RMC. Alternatively, you can click the ellipse (¼) button to start RSWho and select the RMC100 node graphically.

9. In the Type drop-down list, click SLC 5 (Enhanced). Using RSWho in the above step will set this field automatically.

10. Click Accept.

11. Close the Node window.

Step 5: Adding an Example Tag for the RMC Node

At this point, RSView32 recognizes the RMC as a SLC-5/05. You can add tags that use this device in the same way you add tags to any other Allen-Bradley PLC or SLC. This final step shows how to add a tag to the actual position of axis 0.

1. Start RSView32 and open your project.

2. Open the RSView32 Tag Database configuration window. To do this, from the Project Manager window, select the Edit Mode tab, and double-click the Tag Database item from under the System node.

3. In the Node window, ensure that you can see both the edit form at the top and the spreadsheet at the bottom. Use the View menu's Form and Spreadsheet commands to display both.

4. Click New.

5. Under Tag, in the Name text box, type the name you wish to use. For example, RMCAxis0ActPos.

6. In the Type drop-down list, click Analog.

7. Under Data Source, select the Device option for the Type.

8. For the Node Name text box, click the ellipse (¼) button to bring up the Node Browser window.

9. In the Node Browser window, click the RMC node added in the steps above and click OK.

10. In the Address text box, type the address of the register in the RMC. This will always be in the form N[file]:[element]. The first axis's actual position is at address N7:2. For a complete list of registers and their addresses, see the RMC Register Map (Allen-Bradley).

11. All other features on the edit form, such as scan classes and security, are outside the scope of RMC documentation. Refer to RSView32 documentation for details on using these features.

12. Click Accept.

13. Click Close.

5.2.6.6.2 RMC Register Map (Allen-Bradley)

Tip: RMCWin's Address Tool provides an easy way to identify addresses in the RMC. Simply open the Address Tool and then move the cursor to any field in RMCWin that represents an RMC Register, and the Address Tool will display the address in the address format of your choice. See Address Tool for details.

The RMC module has 64K (65536) 16-bit registers that can be read from or written to over Ethernet, Serial, Modbus Plus, and PROFIBUS-DP. Each register is assigned an address. However, under the different communication methods, different addressing schemes are used. This topic describes using Allen-Bradley PLC addressing. For details on addressing from other modules refer to the following topics:

- RMC Register Map (Automationdirect.com)
- RMC Register Map (Modbus/TCP and Modbus/RTU)
- RMC Register Map (Omron FINS)
- RMC Register Map (Siemens TI505)
- RMC Register Map (Siemens S7)
- RMC Register Map (Modbus Plus)
- RMC Register Map (PROFIBUS-DP Message Mode)

Allen-Bradley offers several Ethernet and serial solutions for its ControlLogix, SLC, PLC-5, and SoftLogix 5 controllers. In addition, SoftPLC emulates the PLC-5 and therefore also uses Allen-Bradley's Ethernet protocol. Over this protocol, the RMC's registers are broken into a number of integer files. Each integer file used (N7, N9-N18, L19, N20-N255) is configured to be the maximum size allowed on a SLC 5/05 file: 256 elements: Nf:0-255. For details on reading and writing these registers, see the following topics:

- Using Allen-Bradley Controllers with the RMC ENET
- Using SoftPLC's SoftPLC with the RMC ENET
- Using DF1 (Full- and Half-Duplex) with the RMC SERIAL

Status Registers:

These registers can only be read; writes are ignored.

Allen-Bradley and SoftPLC	**Register Description**
N7:0	Axis 0 Command Position
N7:1	Axis 0 Target Position
N7:2	Axis 0 Actual Position
N7:3	Axis 0 Transducer Counts. **Note:** See the Transducer Counts (32-bit) Registers section below for a way to read 32-bit counts instead of 16-bit counts.
N7:4	Axis 0 Status Word
N7:5	Axis 0 Drive

N7:6	Axis 0 Actual Speed
N7:7	Axis 0 Null Drive
N7:8	Axis 0 Event Step
N7:9	Axis 0 Link Value
N7:10-19	Same as above but for axis 1
N7:20-29	Same as above but for axis 2
N7:30-39	Same as above but for axis 3
N7:40-49	Same as above but for axis 4
N7:50-59	Same as above but for axis 5
N7:60-69	Same as above but for axis 6
N7:70-79	Same as above but for axis 7

Command Registers:
These registers can be read or written.

Allen-Bradley and SoftPLC	Register Description
N7:80	Axis 0 Mode Word
N7:81	Axis 0 Acceleration
N7:82	Axis 0 Deceleration
N7:83	Axis 0 Speed
N7:84	Axis 0 Command Value
N7:85	Axis 0 Command
N7:86-91	Same as above but for axis 1
N7:92-97	Same as above but for axis 2
N7:98-103	Same as above but for axis 3
N7:104-109	Same as above but for axis 4
N7:110-115	Same as above but for axis 5
N7:116-121	Same as above but for axis 6

N7:122-127	Same as above but for axis 7

Parameter Registers:

These registers can be read or written. Changes to these registers do not take effect until a Set Parameters (P) command is executed.

Allen-Bradley and SoftPLC	Register Description
N7:128	Axis 0 Configuration Word
N7:129	Axis 0 Scale
N7:130	Axis 0 Offset
N7:131	Axis 0 Extend Limit
N7:132	Axis 0 Retract Limit
N7:133	Axis 0 Proportional Gain
N7:134	Axis 0 Integral Gain
N7:135	Axis 0 Differential Gain
N7:136	Axis 0 Extend Feed Forward
N7:137	Axis 0 Retract Feed Forward
N7:138	Axis 0 Extend Acceleration Feed Forward
N7:139	Axis 0 Retract Acceleration Feed Forward
N7:140	Axis 0 Dead Band Eliminator
N7:141	Axis 0 In Position Window
N7:142	Axis 0 Following Error
N7:143	Axis 0 Auto Stop
N7:144-159	Same as above but for axis 1
N7:160-175	Same as above but for axis 2
N7:176-191	Same as above but for axis 3
N7:192-207	Same as above but for axis 4
N7:208-223	Same as above but for axis 5
N7:224-239	Same as above but for axis 6

N7:240-255	Same as above but for axis 7

Event Step Table Registers:

These registers can be read or written. When using the Allen-Bradley addressing scheme with these registers, you must keep in mind that the Event Step Table is split over eight register files (this is done because the SLC 5/05 only supports 256 words per file). Use the following table to determine the register file for a given event step:

Event Step (n)	Register File (f)	Step Offset (r)
0-31	N9	(n - 0) x 8
32-63	N10	(n - 32) x 8
64-95	N11	(n - 64) x 8
96-127	N12	(n - 96) x 8
128-159	N13	(n - 128) x 8
160-191	N14	(n - 160) x 8
192-223	N15	(n - 192) x 8
224-255	N16	(n - 224) x 8

Note: On Allen-Bradley PLCs, reads and writes that extend beyond the end of an RMC register file will continue into the next file or files. This is particularly useful on the ControlLogix, which allows reading large amounts of data with a single MSG block. For example, by reading 2048 integers starting at N9:0 in the RMC, the entire Event Step table can be read into the ControlLogix.

The register map for addressing the fields in the event step table is as follows:

Allen-Bradley and SoftPLC	Register Description
N9:0	Step 0 Mode Word
N9:1	Step 0 Acceleration
N9:2	Step 0 Deceleration
N9:3	Step 0 Speed
N9:4	Step 0 Command Value
N9:5	Step 0 Command/Commanded Axes
N9:6	Step 0 Link Type/Link Next

N9:7	Step 0 Link Value
Nf:r + 0	Step n (0-255) Mode Word
Nf:r + 1	Step n (0-255) Acceleration
Nf:r + 2	Step n (0-255) Deceleration
Nf:r + 3	Step n (0-255) Speed
Nf:r + 4	Step n (0-255) Command Value
Nf:r + 5	Step n (0-255) Command/Commanded Axes
Nf:r + 6	Step n (0-255) Link Type/Link Next
Nf:r + 7	Step n (0-255) Link Value

Input to Event Table Registers:
These registers can be read or written.

Allen-Bradley and SoftPLC	**Register Description**
N17:0	Event Step for Axis 0 on Input 0 Rising Edge
N17:1	Event Step for Axis 1 on Input 0 Rising Edge
N17:2	Event Step for Axis 2 on Input 0 Rising Edge
N17:3	Event Step for Axis 3 on Input 0 Rising Edge
N17:4	Event Step for Axis 4 on Input 0 Rising Edge
N17:5	Event Step for Axis 5 on Input 0 Rising Edge
N17:6	Event Step for Axis 6 on Input 0 Rising Edge
N17:7	Event Step for Axis 7 on Input 0 Rising Edge
N17:8 + n	Event Step for Axes n (0-7) on Input 1 Rising Edge
	:
N17:120 + n	Event Step for Axes n (0-7) on Input 15 Rising Edge
N17:128 + n	Event Step for Axes n (0-7) on Input 0 Falling Edge
	:

N17:248 + n	Event Step for Axes n (0-7) on Input 15 Falling Edge

Status Map Registers:

This block of registers is only used by the Modbus Plus and PROFIBUS interfaces. Therefore, these registers are unused by this Ethernet protocol.

Allen-Bradley and SoftPLC	Register Description
N18:0-63	Status Map Entries

Plot Type Registers:

The plot type registers can be read or written. The values that are read indicate the extra plot information in the current graph. The values written to these registers tell the controller which extra plot information to obtain on the next plot. For these registers, the following values are used:

- 0: Extra position precision
- 1: Command and Command Value
- 2: Event Step and Link Value
- 3: Raw Transducer Counts
- 4: Internal Target and Actual Speeds
- 5: Integral Drive

For more information on these four types of plot information, see Selecting the Data to Plot and Reading Plots from the Communication Module.

Allen-Bradley and SoftPLC	Register Description
N18:64	Axis 0 plot type
N18:65	Axis 1 plot type
N18:66	Axis 2 plot type
N18:67	Axis 3 plot type
N18:68	Axis 4 plot type
N18:69	Axis 5 plot type
N18:70	Axis 6 plot type
N18:71	Axis 7 plot type

Digital (Discrete) I/O Registers:

These registers indicate the current state of the digital inputs and outputs. These registers may only be read; writes will be ignored, as this product does not support forcing inputs or outputs.

Because different PLCs label bit numbers differently, the following chart is provided to show the mapping between the devices:

	MSB								
RMC bit #	15	14	13	12	11	10	9	8	7
Allen-Bradley bit #	15	14	13	12	11	10	9	8	7

The bit numbers listed in the table below are in RMC format (0 is LSB, 15 is MSB):

Allen-Bradley and SoftPLC	**Register Description**
N18:72	CPU Digital Inputs 0 and 1 in LSBs of low byte, Outputs 0 and 1 in LSBs of high byte
N18:73	Unused
N18:74	Unused
N18:75	Sensor Digital I/O Inputs 0-15
N18:76	Sensor Digital I/O Inputs 16-17 (stored to two LSBs)
N18:77	Sensor Digital I/O Outputs 0-7 in high byte (low byte unused)
N18:78	Unused
N18:79	Unused

Plot Time Registers:

The Plot Time interval is configurable on the RMC. This interval indicates the number of control loops between each sample in a plot. Therefore, if the control loop is 0.976ms (e.g. RMC100-M1), this value indicates roughly the number of milliseconds between samples. If the control loop is 1.953ms (e.g. RMC100-M4), this indicates half of the number of milliseconds between samples.

These registers may be read or written. When read, they indicate the plot interval of the currently gathered plot. When written, they set the plot interval the RMC should use for the next plot it will gather. When the RMC starts the next plot, it copies the requested plot interval into the currently used plot interval.

Allen-Bradley	**Register Description**

and SoftPLC	
N18:80	Axis 0 plot time interval
N18:81	Axis 1 plot time interval
N18:82	Axis 2 plot time interval
N18:83	Axis 3 plot time interval
N18:84	Axis 4 plot time interval
N18:85	Axis 5 plot time interval
N18:86	Axis 6 plot time interval
N18:87	Axis 7 plot time interval

Last Parameter Error Registers:

Note: To use these registers through Ethernet, you must have RMC100 CPU control firmware dated 19990715 or later and Ethernet firmware dated 19990702 or later.

Each of these read-only registers holds the number of the last parameter error generated on an axis. This is useful for determining the cause of the Parameter Error bit in the status word. For a description of the values read from these registers, see Parameter Error Values.

Allen-Bradley and SoftPLC	Register Description
N18:88	Last parameter error on axis 0
N18:89	Last parameter error on axis 1
N18:90	Last parameter error on axis 2
N18:91	Last parameter error on axis 3
N18:92	Last parameter error on axis 4
N18:93	Last parameter error on axis 5
N18:94	Last parameter error on axis 6
N18:95	Last parameter error on axis 7

Firmware Date Registers:

Note: To use these registers through Ethernet, you must have RMC100 CPU control firmware dated 19990715 or later and Ethernet firmware dated 19990702 or later.

These read-only registers hold information about the firmware versions in the RMC100 CPU module. The Boot and Loader firmware versions have no effect on the actual performance of the RMC and therefore can usually be ignored.

Allen-Bradley and SoftPLC	**Register Description**
N18:96	Boot firmware month (MSB) and day (LSB)
N18:97	Boot firmware year
N18:98	Loader firmware month (MSB) and day (LSB)
N18:99	Loader firmware year
N18:100	Control firmware month (MSB) and day (LSB)
N18:101	Control firmware year
N18:102	Control firmware Beta Code. This will be 0 for standard release firmware, 'B' for Beta firmware, or 'SI' for Superimposed firmware.
N18:103	Feature code. This register is mainly reserved for internal use but does have two bits that may be useful to some users: • If bit 1 (value 0x0002) is set, the control loop is 2 ms, otherwise the control loop is 1 ms. • If bit 0 (value 0x0001) is set, a sensor DI/O is present, otherwise there is no sensor DI/O.

Transducer Counts (32-bit) Registers:

Note: To use these registers through Ethernet, you must have RMC Ethernet firmware dated 20020115 or later.

Each of these read-only registers holds the transducer counts for an axis. Notice that these registers are 32-bit registers. The low 16 bits will match the Counts register read from registers N7:3, N7:13, etc. However, when the transducer counts go below zero (for incremental transducers) or above 65,535 counts, some information is lost. By using these registers, all bits of the counts are available.

Allen-Bradley and SoftPLC	**Register Description**
L19:0	32-bit Transducer Counts for axis 0
L19:1	32-bit Transducer Counts for axis 1
L19:2	32-bit Transducer Counts for axis 2
L19:3	32-bit Transducer Counts for axis 3
L19:4	32-bit Transducer Counts for axis 4

L19:5	32-bit Transducer Counts for axis 5
L19:6	32-bit Transducer Counts for axis 6
L19:7	32-bit Transducer Counts for axis 7

Reserved Registers:

Reading these values will return zero, and writes are ignored.

Allen-Bradley and SoftPLC	Register Description
N20:0-N47:255	Unused

Spline Download Area:

These registers are write only. Reading them will return zero. This area is used to download intervals and points in a spline. This is a much more efficient alternative to using individual New Spline Point and Set Spline Interval/End Segment commands. For details on using this Spline Download Area, see Downloading Splines to the RMC.

Allen-Bradley and SoftPLC	Register Description
N48:0-N63:255	Spline Download Area

Plot Registers:

These registers can only be read; writes are ignored.

Note: Reading plots is not a trivial task; for further details, see Reading Plots from the Communication Module.

Allen-Bradley and SoftPLC	Register Description
N64:0-N87:255	Plot data for axis 0
N88:0-N111:255	Plot data for axis 1
N112:0-N135:255	Plot data for axis 2
N136:0-	Plot data for axis 3

N159:255	
N160:0-N183:255	Plot data for axis 4
N184:0-N207:255	Plot data for axis 5
N208:0-N231:255	Plot data for axis 6
N232:0-N255:255	Plot data for axis 7

5.2.6.7 Siemens Simatic TI505

5.2.6.7.1 Using the Siemens Simatic TI505 with the RMC Ethernet Module

The TI505 does not have built-in Ethernet TCP/IP support. However, Control Technology, Inc. sells a module called the CTI 2572 that fits into the TI505 backplane and adds Ethernet TCP/IP to the TI505.

When using this module, the CTI uses a combination of V-memory registers, and I/O registers (WX and WY) on the CTI 2572 itself to communicate from the TI505 ladder logic. It is beyond the scope of this manual to fully describe using the CTI 2572; please refer to the CTI 2572 Ethernet TCP/IP Adapter Installation and Operation Guide (Part #062-00146) that is shipped with the CTI 2572.

The CTI 2572 can read, write, or do a combined read and write from registers in compatible remote devices such as other TI505 PLCs with the CTI 2572 module or the RMC. The RMC has 65536 registers that are accessible from the CTI 2572. See RMC Register Map (Siemens TI505) for details on those registers and their addresses.

If you need help setting up your network, either consult your network administrator, or for simple stand-alone networks, see Setting up a Stand-alone TCP/IP Control Network.

5.2.6.7.2 RMC Register Map (Siemens TI505)

Tip: RMCWin's Address Tool provides an easy way to identify addresses in the RMC. Simply open the Address Tool and then move the cursor to any field in RMCWin that represents an RMC Register, and the Address Tool will display the address in the address format of your choice. See Address Tool for details.

The RMC module has 64K (65536) 16-bit registers that can be read from or written to over Ethernet, Modbus Plus, and PROFIBUS-DP. Each register is assigned an address. However, under the different communication methods, different addressing schemes are used. This topic describes the addressing from a TI505 using the CTI 2572 Ethernet TCP/IP module. For details on addressing from other modules refer to the following topics:

- RMC Register Map (Allen-Bradley)
- RMC Register Map (Automationdirect.com)

- RMC Register Map (Modbus/TCP and Modbus/RTU)
- RMC Register Map (Omron FINS)
- RMC Register Map (Siemens S7)
- RMC Register Map (Modbus Plus)
- RMC Register Map (PROFIBUS-DP Message Mode)

The Siemens TI505, when equipped with the CTI 2572 Ethernet TCP/IP module, can communicate with the RMC Ethernet module. From the TI505, the RMC registers are addressed as values 1-65536. They can be thought of as the RMC's V-memory addresses V1-V65536. For details on reading and writing these registers, see Using the Siemens Simatic TI505 with the RMC Ethernet Module.

Status Registers:
These registers can only be read; writes are ignored.

TI505 Address	Register Description
1	Axis 0 Command Position
2	Axis 0 Target Position
3	Axis 0 Actual Position
4	Axis 0 Transducer Counts
5	Axis 0 Status Word
6	Axis 0 Drive
7	Axis 0 Actual Speed
8	Axis 0 Null Drive
9	Axis 0 Event Step
10	Axis 0 Link Value
11-20	Same as above but for axis 1
21-30	Same as above but for axis 2
31-40	Same as above but for axis 3
41-50	Same as above but for axis 4
51-60	Same as above but for axis 5
61-70	Same as above but for axis 6

71-80	Same as above but for axis 7

Command Registers:
These registers can be read or written.

TI505 Address	Register Description
81	Axis 0 Mode Word
82	Axis 0 Acceleration
83	Axis 0 Deceleration
84	Axis 0 Speed
85	Axis 0 Command Value
86	Axis 0 Command
87-92	Same as above but for axis 1
93-98	Same as above but for axis 2
99-104	Same as above but for axis 3
105-110	Same as above but for axis 4
111-116	Same as above but for axis 5
117-122	Same as above but for axis 6
123-128	Same as above but for axis 7

Parameter Registers:
These registers can be read or written. Changes to these registers do not take effect until a Set Parameters (P) command is executed.

TI505 Address	Register Description
129	Axis 0 Configuration Word
130	Axis 0 Scale
131	Axis 0 Offset
132	Axis 0 Extend Limit
133	Axis 0 Retract Limit

134	Axis 0 Proportional Gain
135	Axis 0 Integral Gain
136	Axis 0 Differential Gain
137	Axis 0 Extend Feed Forward
138	Axis 0 Retract Feed Forward
139	Axis 0 Extend Acceleration Feed Forward
140	Axis 0 Retract Acceleration Feed Forward
141	Axis 0 Dead Band Eliminator
142	Axis 0 In Position Window
143	Axis 0 Following Error
144	Axis 0 Auto Stop
145-160	Same as above but for axis 1
161-176	Same as above but for axis 2
177-192	Same as above but for axis 3
193-208	Same as above but for axis 4
209-224	Same as above but for axis 5
225-240	Same as above but for axis 6
241-256	Same as above but for axis 7

Event Step Table Registers:
These registers can be read or written.

TI505 Address	Register Description
257	Step 0 Mode Word
258	Step 0 Acceleration
259	Step 0 Deceleration
260	Step 0 Speed
261	Step 0 Command Value
262	Step 0 Command/Commanded Axes

263	Step 0 Link Type/Link Next
264	Step 0 Link Value
257+n*8	Step n (0-255) Mode Word
258+n*8	Step n (0-255) Acceleration
259+n*8	Step n (0-255) Deceleration
260+n*8	Step n (0-255) Speed
261+n*8	Step n (0-255) Command Value
262+n*8	Step n (0-255) Command/Commanded Axes
263+n*8	Step n (0-255) Link Type/Link Next
264+n*8	Step n (0-255) Link Value

Input to Event Table Registers:
These registers can be read or written.

TI505 Address	Register Description
2305	Event Step for Axis 0 on Input 0 Rising Edge
2306	Event Step for Axis 1 on Input 0 Rising Edge
2307	Event Step for Axis 2 on Input 0 Rising Edge
2308	Event Step for Axis 3 on Input 0 Rising Edge
2309	Event Step for Axis 4 on Input 0 Rising Edge
2310	Event Step for Axis 5 on Input 0 Rising Edge
2311	Event Step for Axis 6 on Input 0 Rising Edge
2312	Event Step for Axis 7 on Input 0 Rising Edge
2313 + n	Event Step for Axes n (0-7) on Input 1 Rising Edge
:	:
2425 + n	Event Step for Axes n (0-7) on Input 15 Rising Edge
2433 + n	Event Step for Axes n (0-7) on Input 0 Falling Edge

:	:
2553 + n	Event Step for Axes n (0-7) on Input 15 Falling Edge

Status Map Registers:

This block of registers is only used by the Modbus Plus and PROFIBUS interfaces. Therefore, these registers are unused by this Ethernet protocol.

TI505 Address	**Register Description**
2561-2592	Status Map Entries

Plot Type Registers:

The plot type registers can be read or written. The values that are read indicate the extra plot information in the current graph. Values written to these registers tell the controller which extra plot information to obtain on the next plot. For these registers, the following values are used:

- 0: Extra position precision
- 1: Command and Command Value
- 2: Event Step and Link Value
- 3: Raw Transducer Counts
- 4: Internal Target and Actual Speeds
- 5: Integral Drive

For more information on these four types of plot information, see Selecting the Data to Plot and Reading Plots from the Communication Module.

TI505 Address	**Register Description**
2625	Axis 0 plot type
2626	Axis 1 plot type
2627	Axis 2 plot type
2628	Axis 3 plot type
2629	Axis 4 plot type
2630	Axis 5 plot type
2631	Axis 6 plot type
2632	Axis 7 plot type

Digital (Discrete) I/O Registers:

These registers indicate the current state of the digital inputs and outputs. These registers may only be read; writes will be ignored, as this product does not support forcing inputs or outputs.

Because different PLCs label bit numbers differently, the following chart is provided to show the mapping between the devices:

	MSB								
RMC bit #	15	14	13	12	11	10	9	8	7
TI505 bit #	1	2	3	4	5	6	7	8	9

The bit numbers listed in the table below are in RMC format (0 is LSB, 15 is MSB):

TI505 Address	Register Description
2633	CPU Digital Inputs 0 and 1 in LSBs of low byte, Outputs 0 and 1 in LSBs of high byte
2634	Unused
2635	Unused
2636	Sensor Digital I/O Inputs 0-15
2637	Sensor Digital I/O Inputs 16-17 (stored to two LSBs)
2638	Sensor Digital I/O Outputs 0-7 in high byte (low byte unused)
2639	Unused
2640	Unused

Plot Time Registers:

The Plot Time interval is configurable on the RMC. This interval indicates the number of control loops between each sample in a plot. Therefore, if the control loop is 0.976ms (e.g. RMC100-M1), this indicates roughly the number of milliseconds between samples. If the control loop is 1.953ms (e.g. RMC100-M4), this indicates half of the number of milliseconds between samples.

These registers may be read or written. When read, they indicate the plot interval of the currently gathered plot. When written, they set the plot interval the RMC should use for the next plot it will gather. When the RMC starts the next plot, it copies the requested plot interval into the currently used plot interval.

TI505 Address	Register Description
2641	Axis 0 plot time interval

2642	Axis 1 plot time interval
2643	Axis 2 plot time interval
2644	Axis 3 plot time interval
2645	Axis 4 plot time interval
2646	Axis 5 plot time interval
2647	Axis 6 plot time interval
2648	Axis 7 plot time interval

Last Parameter Error Registers:

Note: To use these registers through Ethernet, you must have RMC100 CPU control firmware dated 19990715 or later and Ethernet firmware dated 19990702 or later.

Each of these read-only registers holds the number of the last parameter error generated on an axis. This is useful for determining the cause of the Parameter Error bit in the status word. For a description of the values read from these registers, see Parameter Error Values.

TI505 Address	Register Description
2649	Last parameter error on axis 0
2650	Last parameter error on axis 1
2651	Last parameter error on axis 2
2652	Last parameter error on axis 3
2653	Last parameter error on axis 4
2654	Last parameter error on axis 5
2655	Last parameter error on axis 6
2656	Last parameter error on axis 7

Firmware Date Registers:

Note: To use these registers through Ethernet, you must have RMC100 CPU control firmware dated 19990715 or later and Ethernet firmware dated 19990702 or later.

These read-only registers hold information about the firmware versions in the RMC100 CPU module. The Boot and Loader firmware versions have no effect on the actual performance of the RMC and therefore can usually be ignored.

TI505 Address	Register Description

2657	Boot firmware month (MSB) and day (LSB)
2658	Boot firmware year
2659	Loader firmware month (MSB) and day (LSB)
2660	Loader firmware year
2661	Control firmware month (MSB) and day (LSB)
2662	Control firmware year
2663	Control firmware Beta Code. This will be 0 for standard release firmware, 'B' for Beta firmware, or 'SI' for Superimposed firmware.
2664	Feature code. This register is mainly reserved for internal use but does have two bits that may be useful to some users: • If bit 1 (value 0x0002) is set, the control loop is 2 ms, otherwise the control loop is 1 ms. • If bit 0 (value 0x0001) is set, a sensor DI/O is present, otherwise there is no sensor DI/O.

Reserved Registers:
Reading these values will return zero, and writes are ignored.

TI505 Address	Register Description
2665-12288	Unused

Spline Download Area:
These registers are write only. Reading them will return zero. This area is used to download intervals and points in a spline. This is a much more efficient alternative to using individual New Spline Point and Set Spline Interval/End Segment commands. For details on using this Spline Download Area, see Downloading Splines to the RMC.

TI505 Address	Register Description
12289-16384	Spline Download Area

Plot Registers:
These registers can only be read; writes are ignored.

Note: Reading plots is not a trivial task; for further details, see Reading Plots from the Communication Module.

TI505 Address	Register Description
16385-22528	Plot data for axis 0
22529-28672	Plot data for axis 1
28673-34816	Plot data for axis 2
34817-40960	Plot data for axis 3
40961-47104	Plot data for axis 4
47105-53248	Plot data for axis 5
53249-59392	Plot data for axis 6
59393-65536	Plot data for axis 7

5.2.6.8 Siemens S7

5.2.6.8.1 RMC Register Map (Siemens S7)

Tip: RMCWin's Address Tool provides an easy way to identify addresses in the RMC. Simply open the Address Tool and then move the cursor to any field in RMCWin that represents an RMC Register, and the Address Tool will display the address in the address format of your choice. See Address Tool for details.

The RMC module has 64K (65536) 16-bit registers that can be read from or written to over Ethernet, Modbus Plus, and PROFIBUS-DP. Each register is assigned an address. However, under the different communication methods, different addressing schemes are used. This topic describes the addressing from a Siemens S7. For details on addressing from other modules refer to the following topics:

- RMC Register Map (Allen-Bradley)
- RMC Register Map (Automationdirect.com)
- RMC Register Map (Modbus/TCP and Modbus/RTU)
- RMC Register Map (Omron FINS)
- RMC Register Map (Siemens TI505)

- RMC Register Map (Modbus Plus)
- RMC Register Map (PROFIBUS-DP Message Mode)

The Siemens S7-300 and S7-400 families of PLCs, when equipped with the CP 343-1 TCP or CP 443-1 TCP modules, can communicate with the RMC Ethernet module. From the S7, the RMC registers are viewed as data block (DB) registers.

Status Registers:
These registers can only be read; writes are ignored.

S7 Address	Register Description
DB1.DBW0	Axis 0 Command Position
DB1.DBW2	Axis 0 Target Position
DB1.DBW4	Axis 0 Actual Position
DB1.DBW6	Axis 0 Transducer Counts
DB1.DBW8	Axis 0 Status Word
DB1.DBW10	Axis 0 Drive
DB1.DBW12	Axis 0 Actual Speed
DB1.DBW14	Axis 0 Null Drive
DB1.DBW16	Axis 0 Event Step
DB1.DBW18	Axis 0 Link Value
DB1.DBW20-38	Same as above but for axis 1
DB1.DBW40-58	Same as above but for axis 2
DB1.DBW60-78	Same as above but for axis 3
DB1.DBW80-98	Same as above but for axis 4
DB1.DBW100-118	Same as above but for axis 5
DB1.DBW120-138	Same as above but for axis 6
DB1.DBW140-158	Same as above but for axis 7

Command Registers:
These registers can be read or written.

S7 Address	Register Description
DB1.DBW160	Axis 0 Mode Word
DB1.DBW162	Axis 0 Acceleration
DB1.DBW164	Axis 0 Deceleration
DB1.DBW166	Axis 0 Speed
DB1.DBW168	Axis 0 Command Value
DB1.DBW170	Axis 0 Command
DB1.DBW172-182	Same as above but for axis 1
DB1.DBW184-194	Same as above but for axis 2
DB1.DBW196-206	Same as above but for axis 3
DB1.DBW208-218	Same as above but for axis 4
DB1.DBW220-230	Same as above but for axis 5
DB1.DBW232-242	Same as above but for axis 6
DB1.DBW244-254	Same as above but for axis 7

Parameter Registers:
These registers can be read or written. Changes to these registers do not take effect until a Set Parameters (P) command is executed.

S7 Address	Register Description
DB2.DBW0	Axis 0 Configuration Word
DB2.DBW2	Axis 0 Scale
DB2.DBW4	Axis 0 Offset
DB2.DBW6	Axis 0 Extend Limit
DB2.DBW8	Axis 0 Retract Limit
DB2.DBW10	Axis 0 Proportional Gain

DB2.DBW12	Axis 0 Integral Gain
DB2.DBW14	Axis 0 Differential Gain
DB2.DBW16	Axis 0 Extend Feed Forward
DB2.DBW18	Axis 0 Retract Feed Forward
DB2.DBW20	Axis 0 Extend Acceleration Feed Forward
DB2.DBW22	Axis 0 Retract Acceleration Feed Forward
DB2.DBW24	Axis 0 Dead Band Eliminator
DB2.DBW26	Axis 0 In Position Window
DB2.DBW28	Axis 0 Following Error
DB2.DBW30	Axis 0 Auto Stop
DB2.DBW32-62	Same as above but for axis 1
DB2.DBW64-94	Same as above but for axis 2
DB2.DBW96-126	Same as above but for axis 3
DB2.DBW128-158	Same as above but for axis 4
DB2.DBW160-190	Same as above but for axis 5
DB2.DBW192-222	Same as above but for axis 6
DB2.DBW224-254	Same as above but for axis 7

Event Step Table Registers:
These registers can be read or written.

S7 Address	Register Description
DB3.DBW0	Step 0 Mode Word
DB3.DBW2	Step 0 Acceleration
DB3.DBW4	Step 0 Deceleration
DB3.DBW6	Step 0 Speed
DB3.DBW8	Step 0 Command Value

DB3.DBW10	Step 0 Command/Commanded Axes
DB3.DBW12	Step 0 Link Type/Link Next
DB3.DBW14	Step 0 Link Value
DB3.DBW0+n*16	Step n (0-255) Mode Word
DB3.DBW2+n*16	Step n (0-255) Acceleration
DB3.DBW4+n*16	Step n (0-255) Deceleration
DB3.DBW6+n*16	Step n (0-255) Speed
DB3.DBW8+n*16	Step n (0-255) Command Value
DB3.DBW10+n*16	Step n (0-255) Command/Commanded Axes
DB3.DBW12+n*16	Step n (0-255) Link Type/Link Next
DB3.DBW14+n*16	Step n (0-255) Link Value

Input to Event Table Registers:
These registers can be read or written.

S7 Address	Register Description
DB4.DBW0	Event Step for Axis 0 on Input 0 Rising Edge
DB4.DBW2	Event Step for Axis 1 on Input 0 Rising Edge
DB4.DBW4	Event Step for Axis 2 on Input 0 Rising Edge
DB4.DBW6	Event Step for Axis 3 on Input 0 Rising Edge
DB4.DBW8	Event Step for Axis 4 on Input 0 Rising Edge
DB4.DBW10	Event Step for Axis 5 on Input 0 Rising Edge
DB4.DBW12	Event Step for Axis 6 on Input 0 Rising Edge
DB4.DBW14	Event Step for Axis 7 on Input 0 Rising Edge
DB4.DBW16 + n*2	Event Step for Axes n (0-7) on Input 1 Rising Edge
:	:
DB4.DBW240 + n*2	Event Step for Axes n (0-7) on Input 15 Rising Edge
DB4.DBW256 +	Event Step for Axes n (0-7) on Input 0 Falling Edge

n*2	
:	:
DB4.DBW496 + n*2	Event Step for Axes n (0-7) on Input 15 Falling Edge

Status Map Registers:

This block of registers is only used by the Modbus Plus and PROFIBUS interfaces. Therefore, these registers are unused by this Ethernet protocol.

S7 Address	Register Description
DB5.DBW0-62	Status Map Entries

Plot Type Registers:

The plot type registers can be read or written. The values that are read indicate the extra plot information in the current graph. The values written to these registers tell the controller which extra plot information to obtain on the next plot. For these registers, the following values are used:

- 0: Extra position precision
- 1: Command and Command Value
- 2: Event Step and Link Value
- 3: Raw Transducer Counts
- 4: Internal Target and Actual Speeds
- 5: Integral Drive

For more information on these four types of plot information, see Selecting the Data to Plot and Reading Plots from the Communication Module.

S7 Address	Register Description
DB6.DBW0	Axis 0 plot type
DB6.DBW2	Axis 1 plot type
DB6.DBW4	Axis 2 plot type
DB6.DBW6	Axis 3 plot type
DB6.DBW8	Axis 4 plot type
DB6.DBW10	Axis 5 plot type
DB6.DBW12	Axis 6 plot type
DB6.DBW14	Axis 7 plot type

Digital (Discrete) I/O Registers:

These registers indicate the current state of the digital inputs and outputs. These registers may only be read; writes will be ignored, as this product does not support forcing inputs or outputs.

S7 Address	Register Description
DB6.DBW16	CPU Digital Inputs 0 and 1 in LSBs of low byte, Outputs 0 and 1 in LSBs of high byte
DB6.DBW18	Unused
DB6.DBW20	Unused
DB6.DBW22	Sensor Digital I/O Inputs 0-15
DB6.DBW24	Sensor Digital I/O Inputs 16-17 (stored to two LSBs)
DB6.DBW26	Sensor Digital I/O Outputs 0-7 in high byte (low byte unused)
DB6.DBW28	Unused
DB6.DBW30	Unused

Plot Time Registers:

The Plot Time interval is configurable on the RMC. This interval indicates the number of control loops between each sample in a plot. Therefore, if the control loop is 0.976ms (e.g. RMC100-M1), this indicates roughly the number of milliseconds between samples. If the control loop is 1.953ms (e.g. RMC100-M4), this indicates half of the number of milliseconds between samples.

These registers may be read or written. When read, they indicate the plot interval of the currently gathered plot. When written, they set the plot interval the RMC should use for the next plot it will gather. When the RMC starts the next plot, it copies the requested plot interval into the currently used plot interval.

S7 Address	Register Description
DB6.DBW32	Axis 0 plot time interval
DB6.DBW34	Axis 1 plot time interval
DB6.DBW36	Axis 2 plot time interval
DB6.DBW38	Axis 3 plot time interval
DB6.DBW40	Axis 4 plot time interval
DB6.DBW42	Axis 5 plot time interval
DB6.DBW44	Axis 6 plot time interval
DB6.DBW46	Axis 7 plot time interval

Last Parameter Error Registers:

Note: To use these registers through Ethernet, you must have RMC100 CPU control firmware dated 19990715 or later.

Each of these read-only registers holds the number of the last parameter error generated on an axis. This is useful for determining the cause of the Parameter Error bit in the status word. For a description of the values read from these registers, see Parameter Error Values.

S7 Address	Register Description
DB6.DBW48	Last parameter error on axis 0
DB6.DBW50	Last parameter error on axis 1
DB6.DBW52	Last parameter error on axis 2
DB6.DBW54	Last parameter error on axis 3
DB6.DBW56	Last parameter error on axis 4
DB6.DBW58	Last parameter error on axis 5
DB6.DBW60	Last parameter error on axis 6
DB6.DBW62	Last parameter error on axis 7

Firmware Date Registers:

Note: To use these registers through Ethernet, you must have RMC100 CPU control firmware dated 19990715 or later.

These read-only registers hold information about the firmware versions in the RMC100 CPU module. The Boot and Loader firmware versions have no effect on the actual performance of the RMC and therefore can usually be ignored.

S7 Address	Register Description
DB6.DBW64	Boot firmware month (MSB) and day (LSB)
DB6.DBW66	Boot firmware year
DB6.DBW68	Loader firmware month (MSB) and day (LSB)
DB6.DBW70	Loader firmware year
DB6.DBW72	Control firmware month (MSB) and day (LSB)
DB6.DBW74	Control firmware year
DB6.DBW76	Control firmware Beta Code. This will be 0 for standard release firmware, 'B' for Beta firmware, or 'SI' for Superimposed firmware.
DB6.DBW78	Feature code. This register is mainly reserved for internal use but does have two

bits that may be useful to some users:

- If bit 1 (value 0x0002) is set, the control loop is 2 ms, otherwise the control loop is 1 ms.
- If bit 0 (value 0x0001) is set, a sensor DI/O is present, otherwise there is no sensor DI/O.

Spline Download Area:

These registers are write only. Reading them will return zero. This area is used to download intervals and points in a spline. This is a much more efficient alternative to using individual New Spline Point and Set Spline Interval/End Segment commands. For details on using this Spline Download Area, see Downloading Splines to the RMC.

S7 Address	Register Description
DB192-207	Spline Download Area

Plot Registers:

These registers can only be read; writes are ignored.

Note: Reading plots is not a trivial task; for further details, see Reading Plots from the Communication Module.

Each plot data block has 1024 elements.

S7 Address	Register Description
DB208-213	Plot data for axis 0
DB214-219	Plot data for axis 1
DB220-225	Plot data for axis 2
DB226-231	Plot data for axis 3
DB232-237	Plot data for axis 4
DB238-243	Plot data for axis 5
DB244-249	Plot data for axis 6
DB250-255	Plot data for axis 7

5.2.6.9 SoftPLC's SoftPLC

5.2.6.9.1 Using the SoftPLC with the RMC Ethernet Module

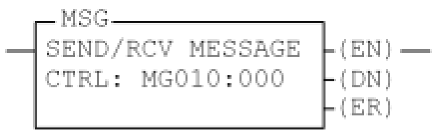
SoftPLC is a PC-based control solution that tightly emulates the PLC-5. Therefore, one who knows how to use the RMC from a PLC-5 should easily be able to get the SoftPLC to communicate with the RMC.

As with the Allen-Bradley PLC-5, the SoftPLC uses the MeSsaGe (MSG) block. This block takes a number of parameters, which are briefly described below. For a complete description of the parameters, refer to Allen-Bradley's Instruction Set Reference Manual.

SoftPLC can read or write from registers in compatible remote devices such as another SoftPLC, PLC-5, or the RMC. The RMC has 248 integer files (N7 and N8-N255) with 256 elements each (0-255) that are accessible over the Ethernet from the SoftPLC. See the RMC Register Map (Allen-Bradley) for details on those registers and their addresses.

If you need help setting up your network, either consult your network administrator, or for simple stand-alone networks, see Setting up a Stand-alone TCP/IP Control Network.

Using the MSG Block:
It is beyond the scope of this document to attempt to fully document the MSG block, which itself evolves over time. However, some general guidelines will be given here. TOPDOC—the SoftPLC programming software—displays the MSG block as follows:



The only parameter in this block is the CTRL block address, which should be an MG-file. To edit this parameter, select the MSG block with TOPDOC and press ALT+Z. This expands the block to show the following options:

- **PLC Command:** From this drop-down list, select **PLC-5 READ** to read values from the RMC, or **PLC-5 WRITE** to write values to the RMC.
- **My Table Address:** Enter the address of the first SoftPLC register to read RMC registers into, or to write to RMC registers from.
- **Size in Elements:** Enter the number of RMC registers to read or write in this field.
- **Channel:** Set this to the Ethernet channel number. On the SoftPLC, the channel numbers are configurable using ONEPRO. Before adding the MSG block you will need to exit SoftPLC, run ONECNFIG and add a TCP/IP channel with the IP address of the RMC. Then enter the channel number you added into this parameter.
- **Station:** This parameter is used only by DH+ and is not used for Ethernet communication.
- **Peer's Mem Address:** Enter the address of the first RMC register to read or write in this field. See the RMC Register Map (Allen-Bradley) for help on addresses.

5.2.6.9.2 RMC Register Map (SoftPLC)

Tip: RMCWin's Address Tool provides an easy way to identify addresses in the RMC. Simply open the Address Tool and then move the cursor to any field in RMCWin that represents an RMC Register, and the Address Tool will display the address in the address format of your choice. See Address Tool for details.

The SoftPLC Ethernet register map is identical to the register map for Allen-Bradley Ethernet. See RMC Register Map (Allen-Bradley).

5.2.6.10 Other PLCs and PC-based Control Packages

5.2.6.10.1 Using Other Ethernet Packages with the RMC ENET

The RMC can emulate a number of Ethernet PLC devices. A list of those supported is described in RMC Ethernet Module Overview. Any device or PC-based software package that can read and write registers in devices that the RMC ENET can emulate can most likely also read and write registers in the RMC. For example, a number of Human-Machine-Interfaces (HMI) can access registers in Allen-Bradley PLCs over Ethernet. Most of these will also be able to access registers in the RMC as well.

Similarly, any device that can read or write registers using one of the RMC's application protocols can also communicate with the RMC. For example, Wonderware's InControl PC-based control package supports a Modbus/TCP SuiteLink driver and can therefore control an RMC over Ethernet. Another example is Control Microsystems' SCADAPack PLC with the 5905 Ethernet Gateway, which supports Modbus/TCP and therefore can also control an RMC over Ethernet.

It is best to contact a Delta Computer Systems, Inc. sales engineer to discuss Ethernet devices that are not explicitly listed as compatible with the RMC. There may be subtle problems with using some devices with the RMC. For example, a device that proclaims Modbus/TCP support may only be a slave. The RMC is also a slave and therefore neither device will initiate transfers, preventing the devices from being able to work together. By talking with a sales engineer, we receive feedback on the devices our customers want to communicate with and may lead to enhancements in our documentation and device support.

5.2.6.11 Custom Ethernet Devices and Applications

5.2.6.11.1 Using the RMCLink ActiveX Control and .Net Assembly Component

There are two main ways to communicate to the RMC over Ethernet from a custom application:

- The **RMCLink ActiveX Control and .Net Assembly Component** can be used if the application is running on Windows. See the RMCLink ActiveX Control and .Net Assembly Component topic for details.
- A TCP/IP API such as BSD Sockets or Winsock can be used to directly write over TCP/IP. This method is not limited to Windows or even PCs. See Using Sockets to Access the RMC ENET for details.

5.2.6.11.2 Using Sockets to Access the RMC ENET

There are two main ways to communicate to the RMC over Ethernet from a custom application:

- The **RMCLink ActiveX Control and .Net Assembly Component** can be used if the application is running on Windows. See RMCLink topic for details.
- A TCP/IP API such as BSD Sockets or Winsock can be used to directly write over TCP/IP. This

method is not limited to Windows or even PCs. This method is discussed in this topic.

Choosing a TCP/IP Stack and API

The PC or device to be programmed must have a TCP/IP Stack and Application Programming Interface (API). All Windows platforms (since Windows 95) include support for a TCP/IP stack and include the Winsock API. Unix platforms typically also have a TCP/IP stack and the BSD Sockets API. It may be more challenging to find a TCP/IP stack and accompanying API for embedded devices, but we expect that such users will understand the depth of their undertaking.

Choosing an Application Protocol

Unless you already have one of the RMC's application protocols implemented, we recommend the Modicon Modbus/TCP protocol. This protocol is one of the simplest protocols and has an open standard. This means the protocol specification is publicly available. See the Modicon Modbus/TCP web site at http://www.modbus.org for complete details on the protocol including C examples for both Windows and Unix and a Java example.

Modbus/TCP Implementation Notes

While the Modicon Modbus/TCP web site has documentation and sample code for its Modbus/TCP protocol, it is worth pointing out a few specifics on communicating with the RMC over Modbus/TCP. This section neither replaces nor modifies the Modbus/TCP specification, but clarifies key points and defines the RMC's function support.

- To create a connection with the RMC over Modbus/TCP, open a client socket with the RMC on TCP port 502. The RMC will only respond to Modbus/TCP requests on this port. Other ports are reserved or used by other protocols.
- The Unit Identifier (Slave Address) field in the Modbus/TCP is only used by routing devices. Since the Slave Address is not used by the RMC, it should be set to zero (0) in requests to the RMC.
- The RMC supports the following function codes:
  - Read Multiple Registers (FC 3)
  - Write Multiple Registers (FC 16)
  - Read Input Registers (FC 4)
  - Write Single Register (FC 6)
  - Read/Write Registers (FC 23)
  - Get Diagnostics (FC 8)
- Typically, only functions 3 and 16 are used. Function 23 can be used to improve performance by doing a read and write in a single function. FC 4, 6, and 8 were included for compatibility with existing Modbus/TCP masters, but otherwise add no value to the RMC.
- Most of the above functions have a Reference Number field. This value is a register address, which must be converted to an RMC register address. Subtract one from the Reference Number to get the RMC's Modbus/TCP register address documented in RMC Register Map (Modbus/TCP

and Modbus/RTU). The addresses documented in RMC Register Map (PROFIBUS-DP Message Mode) happen to already have one subtracted, so you may prefer to use that register map even though it describes PROFIBUS-DP.

- The RMC handles incoming packets on a first-in first-out (FIFO) basis, making it possible to send multiple requests and then wait for the replies. However, the RMC cannot hold a large number of requests at a time, so it is advised that only one or two requests be sent before waiting for the replies.
- The RMC can handle up to four open TCP/IP connections at once. Typically each device uses one connection at a time, allowing the RMC to typically be connected to four devices at once.
- Multiple-register reads and writes are limited to 100 words for writes and 125 words for reads—per the Modbus/TCP specification—even though the length field in the Modbus/TCP header makes it look like larger packets would be supported.

5.3 Modbus Plus

5.3.1 Using the Modicon Modbus Plus Communication Module

Overview

Modbus Plus is a local area network system which supports up to 64 devices (nodes) and transfers data at 1 million bits per second (1Mbaud). Each node on the network must have a unique node address. For details on changing the node address of the RMC, see Changing the Modbus Plus Node Address.

The Modbus Plus network uses the Modbus Protocol, which is a master-slave protocol. A token is passed from node to node on the network. The node holding the token is the master, and therefore controls the network and may request data from or send data to any other node.

The RMC acts only as a slave, which means it will respond to master requests, but will not initiate requests. Some examples of Modbus Plus masters are the Quantum PLC, SA-85 PC-based card, 984-series Modicon PLCs, and Modicon Compact Series PLCs.

Communicating with the RMC

The master may request to read or write 16-bit registers in the RMC. The following information is stored in these registers: status, command and parameter fields of all axes, Event Step table, Input to Event table, and all graphs. For details on the registers in the RMC, see RMC Register Map (Modbus Plus). For details on reading and writing these registers, see Reading and Writing Modbus Plus Registers.

Speeding up Reads from the RMC

In addition to the standard method of reading and writing these Modbus Plus registers, the RMC allows the Global Data feature of Modbus Plus to greatly speed up accessing frequently used data. For details on using this feature, see Using Modbus Plus Global Data.

Programming from a Modicon PLC

When using a Modicon PLC as the master, the user uses a special function block called MSTR. For details on using this function block, see Using the MSTR Modicon Ladder Logic Block.

Understanding the Active LED

The Modbus Plus communication module has a single green LED labeled ”r;Active”. This LED blinks according to the Modbus Plus standard:

- Fast flash (once per 160msec) – This node is working correctly.
- Slow flash (once per second) – This node is monitoring the network prior to going online. The module will be in this state for five seconds during power up.
- 2 flashes, off 2 seconds – This node never receives the token, but hears other nodes passing the token. This node may have a faulty transmitter.
- 3 flashes, off 1.7 seconds – No other nodes are heard on the network. Either this node is not connected to any other nodes, or the receiver is faulty. Check the cabling and all connections.
- 4 flashes, off 1.4 seconds – This node detected another node using the same address. You will need to change the address of this or the like-addressed node.

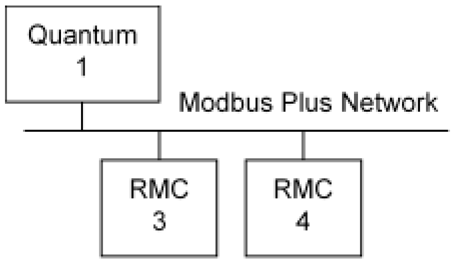
5.3.2 Changing the Modbus Plus Node Address

Each device on the Modbus Plus network must have a unique ”r;node address”. Each node address is a number between 1 and 64. The user changes the RMC’s node address using RMCWin with the following steps:

1. On the Tools menu, click Module Configuration.
2. In the Slot list, click the Modbus Plus item.
3. Click Slot options.
4. In the Node Address box, type the desired node address.
5. Click Update RMC.
6. The Update Module Configuration dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module manually.
7. Click Close from the RMC Configuration dialog.

5.3.3 Reading and Writing Modbus Plus Registers

To help understand the process of reading or writing to the RMC’s registers over Modbus Plus, imagine the following diagram represents a Modbus Plus network with a Quantum PLC as the master and two RMC modules as slaves (the numbers indicate node addresses):



When the master wishes to read from or write to a node it must take the following steps:

1. Wait for the token from the other nodes.
2. Send the request to the slave over the network to read or write data.
3. Give up the token.
4. Wait for the slave to receive the token and respond over the network.

Data Paths

This operation can take several milliseconds to complete, especially if many nodes are on the network, which increases the token loop time. For this reason, Modicon allows several read and write transactions to be in progress at a time. Each device has multiple data ”r;paths”. The RMC has eight slave paths and therefore can handle eight simultaneous transactions. Most Modicon PLCs have four master paths. The master (PLC) must specify the slave data path when reading or writing data. This is described in greater detail in Using the MSTR Modicon Ladder Logic Block.

Another method of speeding up the communication is to utilize Global Data. For details on this feature, see Using Modbus Plus Global Data.

For details on reading and writing using a Modicon PLC, see Using the MSTR Modicon Ladder Logic Block.

5.3.4 RMC Register Map (Modbus Plus)

Tip: RMCWin's Address Tool provides an easy way to identify addresses in the RMC. Simply open the Address Tool and then move the cursor to any field in RMCWin that represents an RMC Register, and the Address Tool will display the address in the address format of your choice. See Address Tool for details.

The RMC module has 64K (65536) 16-bit registers that can be read from or written to over Ethernet, Modbus Plus, and PROFIBUS-DP. Each register is assigned an address. However, under the different communication methods, different addressing schemes are used. This topic describes the addressing through Modbus Plus. For details on addressing from other modules refer to the following topics:

- RMC Register Map (Allen-Bradley)
- RMC Register Map (Automation Direct)
- RMC Register Map (Modbus/TCP)

- RMC Register Map (Omron FINS)
- RMC Register Map (Siemens TI505)
- RMC Register Map (Siemens S7)
- RMC Register Map (PROFIBUS-DP Message Mode)

To communicate with any Modbus Plus device, the RMC requires the Modbus Plus communication module. Under Modbus Plus, the RMC registers are addressed as values 1-65536. They can be thought of as equivalent to Modicon Holding Registers. For details on reading and writing these registers, see Reading and Writing Modbus Plus Registers.

Note: When entering register addresses into an MSTR block, do not add 40000 or 400000 to indicate holding registers. MSTR blocks expect the addresses to start at 1, rather than 40001 or 400001.

Status Registers:
These registers can only be read; writes are ignored.

Modbus Address	Register Description
1	Axis 0 Command Position
2	Axis 0 Target Position
3	Axis 0 Actual Position
4	Axis 0 Transducer Counts
5	Axis 0 Status Word
6	Axis 0 Drive
7	Axis 0 Actual Speed
8	Axis 0 Null Drive
9	Axis 0 Event Step
10	Axis 0 Link Value
11-20	Same as above but for axis 1
21-30	Same as above but for axis 2
31-40	Same as above but for axis 3
41-50	Same as above but for axis 4
51-60	Same as above but for axis 5
61-70	Same as above but for axis 6

71-80	Same as above but for axis 7

Command Registers:
These registers can be read or written.

Modbus Address	Register Description
81	Axis 0 Mode Word
82	Axis 0 Acceleration
83	Axis 0 Deceleration
84	Axis 0 Speed
85	Axis 0 Command Value
86	Axis 0 Command
87-92	Same as above but for axis 1
93-98	Same as above but for axis 2
99-104	Same as above but for axis 3
105-110	Same as above but for axis 4
111-116	Same as above but for axis 5
117-122	Same as above but for axis 6
123-128	Same as above but for axis 7

Parameter Registers:
These registers can be read or written. Changes to these registers do not take effect until a Set Parameters (P) command is executed.

Modbus Address	Register Description
129	Axis 0 Configuration Word
130	Axis 0 Scale
131	Axis 0 Offset
132	Axis 0 Extend Limit
133	Axis 0 Retract Limit

134	Axis 0 Proportional Gain
135	Axis 0 Integral Gain
136	Axis 0 Differential Gain
137	Axis 0 Extend Feed Forward
138	Axis 0 Retract Feed Forward
139	Axis 0 Extend Acceleration Feed Forward
140	Axis 0 Retract Acceleration Feed Forward
141	Axis 0 Dead Band Eliminator
142	Axis 0 In Position Window
143	Axis 0 Following Error
144	Axis 0 Auto Stop
145-160	Same as above but for axis 1
161-176	Same as above but for axis 2
177-192	Same as above but for axis 3
193-208	Same as above but for axis 4
209-224	Same as above but for axis 5
225-240	Same as above but for axis 6
241-256	Same as above but for axis 7

Event Step Table Registers:
These registers can be read or written.

Modbus Address	**Register Description**
257	Step 0 Mode Word
258	Step 0 Acceleration
259	Step 0 Deceleration
260	Step 0 Speed
261	Step 0 Command Value
262	Step 0 Command/Commanded Axes

263	Step 0 Link Type/Link Next
264	Step 0 Link Value
257+n*8	Step n (0-255) Mode Word
258+n*8	Step n (0-255) Acceleration
259+n*8	Step n (0-255) Deceleration
260+n*8	Step n (0-255) Speed
261+n*8	Step n (0-255) Command Value
262+n*8	Step n (0-255) Command/Commanded Axes
263+n*8	Step n (0-255) Link Type/Link Next
264+n*8	Step n (0-255) Link Value

Input to Event Table Registers:
These registers can be read or written.

Modbus Address	Register Description
2305	Event Step for Axis 0 on Input 0 Rising Edge
2306	Event Step for Axis 1 on Input 0 Rising Edge
2307	Event Step for Axis 2 on Input 0 Rising Edge
2308	Event Step for Axis 3 on Input 0 Rising Edge
2309	Event Step for Axis 4 on Input 0 Rising Edge
2310	Event Step for Axis 5 on Input 0 Rising Edge
2311	Event Step for Axis 6 on Input 0 Rising Edge
2312	Event Step for Axis 7 on Input 0 Rising Edge
2313 + n	Event Step for Axes n (0-7) on Input 1 Rising Edge
:	:
2425 + n	Event Step for Axes n (0-7) on Input 15 Rising Edge
2433 + n	Event Step for Axes n (0-7) on Input 0 Falling Edge

:	:
2553 + n	Event Step for Axes n (0-7) on Input 15 Falling Edge

Status Map Registers:

These registers can be read or written, although you should not manually change the values in this table. You should use the Status Map Editor to change this table and then download it to the RMC. You may then read this table into the PLC and send the table to the RMC each time the PLC is restarted.

Modbus Address	Register Description
2561-2592	Status Map Entries

Plot Type Registers:

The plot type registers can be read or written. The values that are read indicate the extra plot information in the current graph. The values written to these registers tell the controller which extra plot information to obtain on the next plot. For these registers, the following values are used:

- 0: Extra position precision
- 1: Command and Command Value
- 2: Event Step and Link Value
- 3: Raw Transducer Counts
- 4: Internal Target and Actual Speeds
- 5: Integral Drive

For more information on these four types of plot information, see Selecting the Data to Plot and Reading Plots from the Communication Module.

Modbus Address	Register Description
2625	Axis 0 plot type
2626	Axis 1 plot type
2627	Axis 2 plot type
2628	Axis 3 plot type
2629	Axis 4 plot type
2630	Axis 5 plot type

2631	Axis 6 plot type
2632	Axis 7 plot type

Digital (Discrete) I/O Registers:

These registers indicate the current state of the digital inputs and outputs. These registers may only be read; writes will be ignored, as this product does not support forcing inputs or outputs.

Because different PLCs label bit numbers differently, the following chart is provided to show the mapping between the devices:

	MSB								
RMC bit #	15	14	13	12	11	10	9	8	7
Modicon bit #	1	2	3	4	5	6	7	8	9

The bit numbers listed in the table below are in RMC format (0 is LSB, 15 is MSB):

Modbus Address	Register Description
2633	CPU Digital Inputs 0 and 1 in LSBs of low byte, Outputs 0 and 1 in LSBs of high byte
2634	Unused
2635	Unused
2636	Sensor Digital I/O Inputs 0-15
2637	Sensor Digital I/O Inputs 16-17 (stored to two LSBs)
2638	Sensor Digital I/O Outputs 0-7 in high byte (low byte unused)
2639	Unused
2640	Unused

Plot Time Registers:

The Plot Time interval is configurable on the RMC. This interval indicates the number of control loops between each sample in a plot. Therefore, if the control loop is 0.976ms (e.g. RMC100-M1), this indicates roughly the number of milliseconds between samples. If the control loop is 1.953ms (e.g. RMC100-M4), this indicates half of the number of milliseconds between samples.

These registers may be read or written. When read, they indicate the plot interval of the currently gathered plot. When written, they set the plot interval the RMC should use for the next plot it will gather. When the RMC starts the next plot, it copies the requested plot interval into the currently used plot interval.

Modbus Address	Register Description
2641	Axis 0 plot time interval
2642	Axis 1 plot time interval
2643	Axis 2 plot time interval
2644	Axis 3 plot time interval
2645	Axis 4 plot time interval
2646	Axis 5 plot time interval
2647	Axis 6 plot time interval
2648	Axis 7 plot time interval

Last Parameter Error Registers:

Note: To use these registers through Modbus Plus, you must have RMC100 CPU control firmware dated 19990819 or later.

Each of these read-only registers holds the number of the last parameter error generated on an axis. This is useful for determining the cause of the Parameter Error bit in the status word. For a description of the values read from these registers, see Parameter Error Values.

Modbus Address	Register Description
2649	Last parameter error on axis 0
2650	Last parameter error on axis 1
2651	Last parameter error on axis 2
2652	Last parameter error on axis 3
2653	Last parameter error on axis 4
2654	Last parameter error on axis 5
2655	Last parameter error on axis 6
2656	Last parameter error on axis 7

Firmware Date Registers:

Note: To use these registers through Modbus Plus, you must have RMC100 CPU control firmware dated 19990819 or later.

These read-only registers hold information about the firmware versions in the RMC100 CPU module. The Boot and Loader firmware versions have no effect on the actual performance of the RMC and therefore can usually be ignored.

Modbus Address	Register Description
2657	Boot firmware month (MSB) and day (LSB)
2658	Boot firmware year
2659	Loader firmware month (MSB) and day (LSB)
2660	Loader firmware year
2661	Control firmware month (MSB) and day (LSB)
2662	Control firmware year
2663	Control firmware Beta Code. This will be 0 for standard release firmware, 'r;B' for Beta firmware, or 'r;SI' for Superimposed firmware.
2664	Feature code. This register is mainly reserved for internal use but does have two bits that may be useful to some users: • If bit 1 (value 0x0002) is set, the control loop is 2 ms, otherwise the control loop is 1 ms. • If bit 0 (value 0x0001) is set, a sensor DI/O is present, otherwise there is no sensor DI/O.

Reserved Registers:

Reading these values will return zero, and writes are ignored.

Modbus Address	Register Description
2665-12288	Unused

Spline Download Area:

These registers are write only. Reading them will return zero. This area is used to download intervals and points in a spline. This is a much more efficient alternative to using individual New Spline Point and Set Spline Interval/End Segment commands. For details on using this Spline Download Area, see Downloading Splines to the RMC.

Modbus Address

	Register Description
12289-16384	Spline Download Area

Plot Registers:

These registers can only be read; writes are ignored.

Note: Reading plots is not a trivial task; for further details, see Reading Plots from the Communication Module.

Modbus Address	Register Description
16385-22528	Plot data for axis 0
22529-28672	Plot data for axis 1
28673-34816	Plot data for axis 2
34817-40960	Plot data for axis 3
40961-47104	Plot data for axis 4
47105-53248	Plot data for axis 5
53249-59392	Plot data for axis 6
59393-65536	Plot data for axis 7

5.3.5 Using the TSX Premium and Modbus Plus

The TSX Premium provides several functions for communicating over Modbus Plus. Unfortunately, only the SEND_REQ function allows a complete routing address, and is thus the only function that can be used to communicate with the RMC100 over Modbus Plus. Refer to the TSX Premium documentation for complete details on using the SEND_REQ function. The following information is intended to clarify how it is used with the RMC100:

SEND_REQ(addr, func, control, local_addr, status)

addr - the slot and port on the TSX Premium to communicate with, plus the first byte in the routing address. For example, to communicate with Modbus Plus node #30, this would be "ADR#0.1.30".

func - Which SEND_REQ operation to run (read, write, etc.). Refer to TSX Premium documentation for details.

control - A block of 5 registers, specifying additional information:

control:0: Second and third routing address bytes. Notice that the second address byte refers to the data path in the RMC100 and therefore must be a number between 1 and 8. Each simultaneous request from the same RMC should use a different data path value. Typically, the third address byte will be left 0.

control:1: Fourth and fifth address bytes. These will typically be left at zero.

control:2: Segment and type (depends on the type of the variable to be read or written; refer to TSX Premium documentation).

control:3: Address of the first internal word to be read from the remote (RMC100) device. This is an offset from 400001.

control:4: Size of the data to read (in bytes).

local_addr - This is where the data being read will be saved in the TSX Premium, or what data in the TSX Premium is to be written.

status - Refer to TSX Premium documentation for details

Example

The following example reads 10 registers from addresses 400001-400009 in the RMC100-MBP located at Modbus Plus node 30. It uses data path 1, and stores the information in %MW450-%MW459. The control area is located at %MW300-%MW304 and the status area is located at %MW600-%MW603 in the TSX Premium.

SEND_REQ(ADR#0.1.30, 16#36, %MW300:5, %MW450:10, %MW600:4)

%MW300 = 0x0001 Second and third routing addresses (01, 00)

%MW301 = 0x0000 Fourth and fifth routing addresses (00, 00)

%MW302 = 0x0768 Segment 104 and type 4 (dpeends on the type of the variable to be read)

%MW303 = 0 Address of the first word in the RMC100 (offset from 400001).

%MW304 = 20 Size of the data to be read in bytes (10 words at 2 bytes per word).

5.3.6 Modbus Plus Global Data

5.3.6.1 Using Modbus Plus Global Data

Note: Global data should be used in all applications; it is not only for advanced users.

You should first familiarize yourself with the standard method of reading and writing to a slave from a master, as described in Reading and Writing Modbus Plus Registers. Global data greatly speeds up the time required for a master to retrieve data from a slave such as the RMC.

Each node may send up to 32 registers of global data as it passes the token to the next node on the network. All nodes on the network receive this data and store it in an internal database. Therefore, whenever a node wants to know the global data of a particular node, all it has to do is look in its internal database, which gets updated automatically.

The RMC always broadcasts 32 registers of global data. The user can select which data will be included in the 32 registers of global data. For details on assigning the global data registers, see Using the Status Map Editor.

For a list of the default global data register mapping, see Default Status Map Data.

There are two methods of reading Global Data into the Modicon PLC:

- Use the MSTR Modicon Ladder Logic Block to explicitly read the global data. Using this method, you are limited to reading the global data into holding registers only.
- Use Modicon’s Peer Cop feature. This method works better in most applications, but takes more time to set up. See Using Modicon’s Peer Cop to Read Global Data topic.

5.3.6.2 Using Modicon’s Peer Cop to Read Global Data

Peer Cop Overview

Modicon’s Peer Cop feature is a method of having data read and written to and from remote nodes at the top of each PLC scan. Either of Modicon’s Modsoft® and Concept® software packages can be used to configure Peer Cop.

Peer Cop supports four types of automatic data transfers:

- Specific Inputs
- Specific Outputs
- Global Inputs
- Global Outputs

Of these four types, only **Global Inputs** are supported by the RMC. Global inputs refer to the global data sent out by other devices, such as the RMC. Modsoft and Concept allow the user to select which of the Global Data registers get copied to which Modicon PLC registers.

The following steps are required to set up Peer Cop. The exact details are not described as they differ between versions of Modsoft and Concept:

1. **Reserve Config Extension Memory in the PLC:**
   Peer Cop uses Config Extension memory. Refer to Modicon Modsoft Programmer User Manual for details on the exact procedure for allocating Config Extension memory and the method of calculating the memory requirements.

2. **Add the RMC Device as a Peer Cop Node:**
   Some versions of Modsoft require that the Peer Cop node be added.

3. **Add Global Data Sub-entries to the RMC Peer Cop Node:**

   For each device, the global data can be copied into eight sections of the PLC memory. Each section is called a sub-entry. Each sub-entry has the following pieces of information:

   - Address in PLC memory (0xxxxx, 1xxxxx, 3xxxxx, 4xxxxx).
   - Length (in 16-bit registers) to copy.
   - Type of the data. This should always be BIN for the RMC.
   - Index. The index refers to the number of the first register to be copied. Its range is from 1-31.

   At the top of each scan, for each sub-entry, the Global Data registers beginning at the Index—and continuing for the number of registers given by the Length—are copied into the PLC address.

   See the example below.

4. **Add a Network for Detecting the Peer Cop Health:**
   This step is not required, but highly recommended. Refer to MSTR Block Peer Cop Health Operation for an example of the Modicon logic for doing this.

Example

Suppose we have a four-axis RMC module at node address 3, and we have configured the Status Map to have the following assignments for the first nine registers:

Status Map Register:	RMC Register:	Holds:
0	0	Axis 0 Command Position
1	1	Axis 0 Target Position
2	2	Axis 0 Actual Position
3	4	Axis 0 Status Word
4	10	Axis 1 Command Position
5	11	Axis 1 Target Position
6	12	Axis 1 Actual Position

7	14	Axis 1 Status Word
8	2632	CPU Digital Inputs 0 and 1 in LSBs of low byte, Outputs 0 and 1 in LSBs of high byte.

Next suppose that we want to copy the first eight global registers into PLC holding registers from 400401 through 400408. In addition, we want the two status words also to be copied into coils at 000801 through 000816 for axis 0 and 000817 through 000832 for axis 1 so that each bit can be used easier. Finally, we want the CPU input and output bits to be mapped into discrete inputs 100001 through 100016. The following set of sub-entries would be added to node 3:

Length	**Reference**	**Type**	**Index**
8	400401-400408	BIN	1
1*	000801-000816	BIN	4
1*	000817-000832	BIN	8
1*	100001-100016	BIN	9

*The lengths are given in global registers, and therefore one global register corresponds to 16 discrete inputs.

The following table shows the placement of the input bits:

Address	**Content**
000801	MSB of axis 0 Status Word (No Transducer)
:	:
000816	LSB of axis 0 Status Word (In Position)
000817	MSB of axis 1 Status Word (No Transducer)
:	:
000832	LSB of axis 1 Status Word (In Position)
100007	CPU Digital Output 1
100008	CPU Digital Output 0
100015	CPU Digital Input 1
100016	CPU Digital Input 0

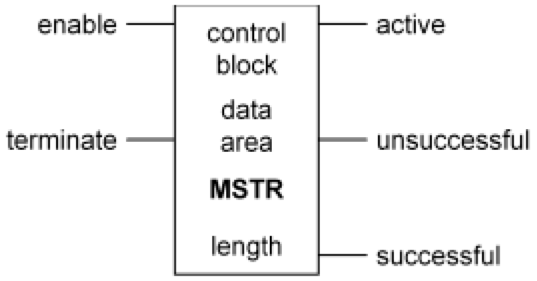
5.3.7 MSTR Modicon Ladder Logic Block

5.3.7.1 Using the MSTR Modicon Ladder Logic Block

When using a Modicon PLC with the Modbus Plus or Modbus/TCP network, the MSTR (for MaSTeR) ladder logic function block must be used. Before reading this topic for use with Modbus Plus, you should understand Reading and Writing Modbus Plus Registers and Using Modbus Plus Global Data. Before reading this topic for use with Modbus/TCP, you should understand Using the Modicon Quantum with the RMC Ethernet Module.

For complete documentation on this function block, refer to the Modicon Ladder Logic Block Library User Guide available from Schneider Electric. This topic uses information from the above text and will describe the aspects of this block that apply to using it with the RMC module.

This function block takes the following form:

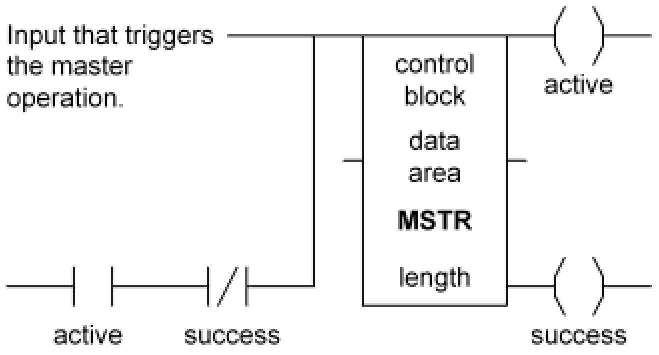


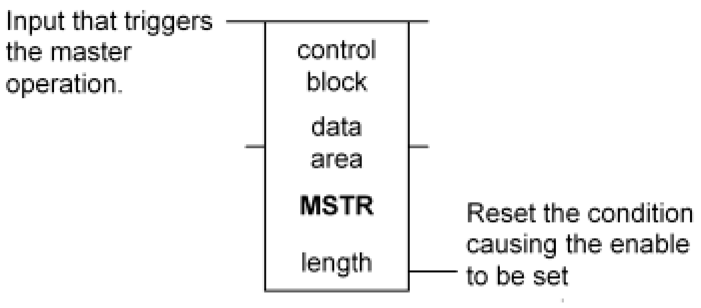
Inputs

There are two control inputs:

- Enable: This input must be ON for the duration of an operation
- Terminate: This input terminates the active operation when ON

You must keep the enable input on for the duration of the operation. The following two ladders accomplish this in different ways:




Input that triggers the master operation.
control block
data area
MSTR
length
Reset the condition causing the enable to be set

In the second method, you must not reset the condition that triggers the MSTR block until the block completes.

Outputs

There are three possible outputs:

- Active: This output will be ON while an operation is in progress
- Unsuccessful: The operation failed. Refer to the Error Status register described below under Control Block for details on the failure.
- Successful: The operation completed successfully

Control Block

A 4x (holding) register is given in this field (node). This register is the first of nine contiguous holding registers that comprise the control block:

Register	Content
1st	**Operation**: There are nine operation types, but only these four are used during normal operation and documented here: 1: Write data (click here for details and example) 2: Read data (click here for details and example) 6: Read Global Data (Modbus Plus only) (click here for details and example) 9: Peer Cop Health (Modbus Plus only) (click here for details and example)
2nd	**Error Status**:This register is filled by the operation. See MSTR Block Error Codes for a complete list.
3rd	**Length:** Indicates the number of registers to transfer.
4th	**Operation-dependent Value:** **Write:** Indicates the register address in the slave to write to **Read:** Indicates the register address in the slave to read from

Read Global Data: Gets filled by the PLC with the number of registers of global data available from the slave

Peer Cop Health: Gives the number of registers to read from the Peer Cop Health map.

5th-9th **Routing 1-5:**

Read/Write Data: The uses of these fields depend on whether Modbus Plus or Modbus/TCP is being used.

Modbus Plus: For local addresses, Routing 1 gives the node address, and Routing 2 gives the data path (1-8) to use in the node. Routing 3 through 5 are zeros. For more complicated routing methods, refer to the section on routing path structure in Modbus Plus Network Planning and Installation Guide.

Modbus/TCP: Routing 1 is broken into two bytes: the MSB holds the Quantum backplane Slot ID of the NOE module, and the LSB holds the Map Index, which should be zero for the RMC. Routing 2 through 5 holds the four dot-separated values of the RMC IP address (e.g. 192.168.0.5).

Global Data: Routing 1 gives the address of the node to read global data from. Routing 2 through 5 are zeros.

Peer Cop Health: Unused.

Data area

A 4x (holding) register is given in this field (node). This register is the first of a block of data that will either be written to the slave (in the case of a Write Data operation) or read into from the slave (in the case of Read Data and Read Global Data operations).

Length

This integer gives the number of registers available for reading and writing in the data area. Notice that this value serves as a maximum and does not indicate the actual number of registers written or read, which is given in the 3rd register in the Control Block.

5.3.7.2 MSTR Block Read Operation

For general information on the MSTR Block, see Using the MSTR Modicon Ladder Logic Block.

Reading from Modbus Plus is described in Reading and Writing Modbus Plus Registers. Reading from Modbus/TCP is described in Using the Modicon Quantum with the RMC Ethernet Module.

Control Block

In a Read operation, the nine control-block registers are assigned in as shown:

Register	Content
1st	**Operation:** 2
2nd	**Error Status**: This register is filled by the operation. See MSTR Block Error Codes for a complete list.
3rd	**Length**: Indicates the number of registers to read.
4th	**Operation-dependent Value**: Indicates the address in the slave to read from.
5th-9th	**Routing 1-5**: The uses of these fields depend on whether Modbus Plus or Modbus/TCP is being used:

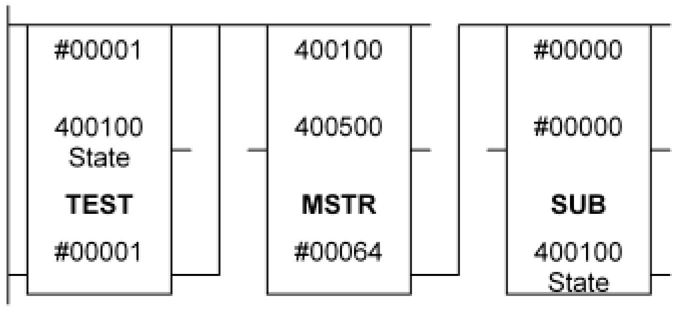
Note: When entering register addresses into an MSTR block, do not add 40000 or 400000 to indicate holding registers. MSTR blocks expect the addresses to start at 1, rather than 40001 or 400001.

Modbus Plus: For local addresses, Routing 1 gives the node address, and Routing 2 gives the data path (1-8) to use in the node. Routing 3 through 5 are zeros. For more complicated routing methods, refer to the section on routing path structure in Modbus Plus Network Planning and Installation Guide.

Modbus/TCP: Routing 1 is broken into two bytes: the MSB holds the Quantum backplane Slot ID of the NOE module, and the LSB holds the Map Index, which should be zero for the RMC. Routing 2 through 5 holds the four dot-separated values of the RMC IP address (e.g. 192.168.0.5).

Example (Modbus Plus)

Suppose that we wish to read the parameters for the first four axes of the RMC located at node address 3 whenever we reach state 1. After reading the parameters we are to return to state 0. The state is stored in holding register 400250, and we will read the parameters into the holding registers 400500 through 400563. The control block is located at 400100 to 400108. The following network would be used:

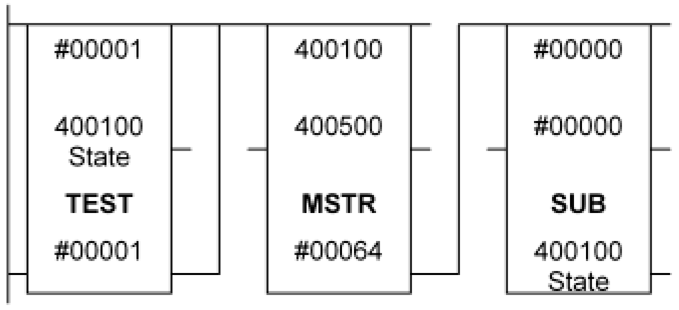


This network waits until the state is equal to 1 and then triggers the MSTR block using the control block below. When the MSTR read operation completes, the state is set to 0. Notice that the MSTR enable input is powered for the duration of the operation.

Register	**Content**
400100	2 (decimal): Operation type: Read data
400101	0000 (hex): Error status: will be filled in by function
400102	64 (decimal): Length: there are 16 parameters on each of four axes
400103	129 (decimal): Address in slave (RMC) memory: This is the address of the first parameter on the first axis
400104	3 (decimal): Routing 1 (Node address)
400105	1 (decimal): Routing 2 (Data path): This could have been 1-8
400106	0 (decimal): Routing 3
400107	0 (decimal): Routing 4
400108	0 (decimal): Routing 5

Example (Modbus/TCP)

Suppose that we wish to read the parameters for the first four axes of an RMC whenever we reach state 1. The RMC has an IP address of 192.168.0.5 and we will use the Modicon Ethernet/TCP module (140 NOE 211 00) located in the Quantum backplane slot #3. After reading the parameters we are to return to state 0. The state is stored in holding register 400250, and we will read the parameters into the holding registers 400500 through 400563. The control block is located at 400100 to 400108. The following network would be used:



This network waits until the state is equal to 1 and then triggers the MSTR block using the control block below. When the MSTR read operation completes, the state is set to 0. Notice that the MSTR enable input is powered for the duration of the operation.

Register **Content**

400100	2 (decimal): Operation type: Read data
400101	0000 (hex): Error status: will be filled in by function
400102	64 (decimal): Length: there are 16 parameters on each of four axes
400103	129 (decimal): Address in slave (RMC) memory: This is the address of the first parameter on the first axis
400104	0300 (hex): Routing 1: The high byte holds the Quantum backplane slot ID (3). The low byte holds the Map Index, which should be set to zero when communicating with the RMC.
400105	192 (decimal): Routing 2: First byte of the IP address: **192**.168.0.5
400106	168 (decimal): Routing 3: Second byte of the IP address: 192.**168**.0.5
400107	0 (decimal): Routing 4: Third byte of the IP address: 192.168.**0**.5
400108	5 (decimal): Routing 5: Fourth byte of the IP address: 192.168.0.**5**

5.3.7.3 MSTR Block Write Operation

For general information on the MSTR Block, see Using the MSTR Modicon Ladder Logic Block.

Writing with Modbus Plus is described in Reading and Writing Modbus Plus Registers. Writing with Modbus/TCP is described in Using the Modicon Quantum with the RMC Ethernet Module.

Control Block

In a write operation, the nine control-block registers are assigned as shown:

Register	Content
1st	**Operation**: 1
2nd	**Error Status**: This register is filled by the operation. See MSTR Block Error Codes for a complete list.
3rd	**Length**: Indicates the number of registers to write.
4th	**Operation-dependent Value**: Indicates the address in the slave to write to.

Note: When entering register addresses into an

MSTR block, do not add 40000 or 400000 to indicate holding registers. MSTR blocks expect the addresses to start at 1, rather than 40001 or 400001.

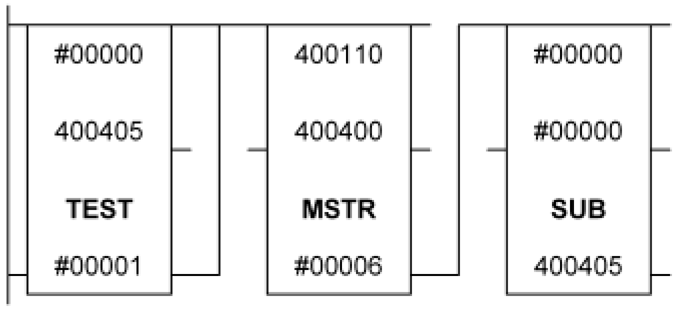
5th-9th **Routing 1-5**: The uses of these fields depend on whether Modbus Plus or Modbus/TCP is being used:

Modbus Plus: For local addresses, Routing 1 gives the node address, and Routing 2 gives the data path (1-8) to use in the node. Routing 3 through 5 are zeros. For more complicated routing methods, refer to the section on routing path structure in Modbus Plus Network Planning and Installation Guide.

Modbus/TCP: Routing 1 is broken into two bytes: the MSB holds the Quantum backplane Slot ID of the NOE module, and the LSB holds the Map Index, which should be zero for the RMC. Routing 2 through 5 holds the four dot-separated values of the RMC IP address (e.g. 192.168.0.5).

Example (Modbus Plus):

Suppose that you wish to write the six command registers to the first axis of the RMC located at node address 3 whenever the command is non-zero. The command registers to write are located in holding registers 400400 through 400005. You must first choose a location for the 9-register control block. In this example, 400110 to 400118 will be the location. The following network would be used:



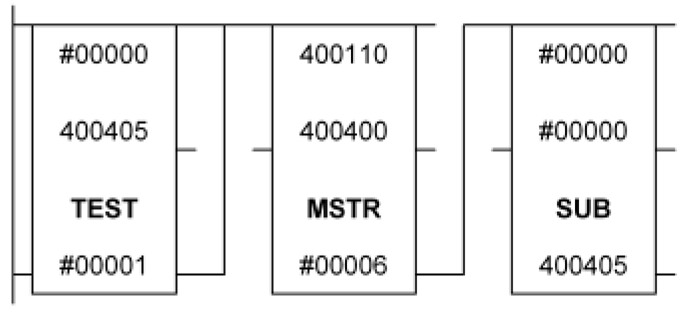
This network waits until the command word at 400405 is non-zero and then triggers the MSTR block using the control block below. When the MSTR write operation completes, the command word in 400405 is cleared. Notice that the MSTR enable input is powered for the duration of the operation.

Register	Content
400110	1 (decimal): Operation type: Write data
400111	0000 (hex): Error status: will be filled in by function
400112	6 (decimal): Length: there are 6 command

	registers on each axis
400113	81 (decimal): Address in slave (RMC) memory: This is the address of the first command register on the first axis
400114	3 (decimal): Routing 1 (Node address)
400115	2 (decimal): Routing 2 (Data path): This could have been 1-8
400116	0 (decimal): Routing 3
400117	0 (decimal): Routing 4
400118	0 (decimal): Routing 5

Example (Modbus/TCP):

Suppose that you wish to write the six command registers to the first axes of the RMC located at IP address 192.168.0.5 through a Modicon Ethernet TCP/IP module (140 NOE 211 00) in the Quantum backplane slot #3. You wish to write the command registers whenever the command is non-zero. The command registers to write are located in holding registers 400400 through 400005. You must first choose a location for the 9-register control block. In this example, 400110 to 400118 will be the location. The following network would be used:



This network waits until the command word at 400405 is non-zero and then triggers the MSTR block using the control block below. When the MSTR write operation completes, the command word in 400405 is cleared. Notice that the MSTR enable input is powered for the duration of the operation.

Register	Content
400110	1 (decimal): Operation type: Write data
400111	0000 (hex): Error status: will be filled in by function
400112	6 (decimal): Length: there are 6 command registers on each axis
400113	81 (decimal): Address in slave (RMC) memory: This is the address of the first command register on the first axis

400114	0300 (hex): The high byte holds the Quantum backplane slot ID (3). The low byte holds the Map Index, which should be set to zero when communicating with the RMC.
400115	192 (decimal): Routing 2: First byte of the IP address: 192.168.0.5
400116	168 (decimal): Routing 3: Second byte of the IP address: 192.168.0.5
400117	0 (decimal): Routing 4: Third byte of the IP address: 192.168.0.5
400118	5 (decimal): Routing 5: Fourth byte of the IP address: 192.168.0.5

5.3.7.4 MSTR Block Read Global Data Operation

For general information on the MSTR Block, see Using the MSTR Modicon Ladder Logic Block.

Note: This operation is only available for Modbus Plus and not Modbus/TCP.

This operation reads in Global Data registers from a particular slave device (such as the RMC). The read always begins with the first Global Data register on the device, but the number of registers read is set in the control block. For details on Global Data, see Using Modbus Plus Global Data.

Note: This operation will fail with an error code of 200B if Peer Cop is used. The two methods of accessing Global Data cannot be used at the same time.

Control Block

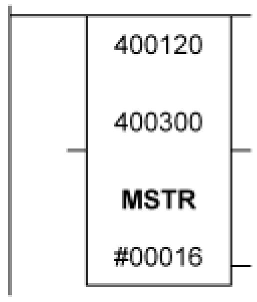
In a Read Global Data operation, the nine control-block registers are assigned as shown:

Register	Content
1st	**Operation**: 6
2nd	**Error Status**: This register is filled by the operation. See MSTR Block Error Codes for a complete list.
3rd	**Length**: Indicates the number of global data registers to read.
4th	**Operation-dependent Value**: Gets filled by the PLC with the number of registers of global data available from the slave.
5th	**Routing 1**: Address of the node to read global data from.

6th-9th	**Routing 2-5**: Unused. Set to zeros.

Example

Suppose that you wish to read four global data registers from each of the first four axes of the RMC located at node address 3. The commands to write are located in holding register 400400 through 400423. You must first choose a location for the 9-register control block. In this example, 400120 to 400128 is the location. The following MSTR block would be used:



As you can see, this function block indicates it will do an operation using the control block beginning at 400120, using up to 32 registers beginning at 400300. The next step is to set up the control block:

Register	Content
400120	6 (decimal): Operation type: Read global data
400121	0000 (hex): Error status: filled in by function
400122	16 (decimal): Length: we want the first 16 registers
400123	0 (decimal): Available Registers: filled in by function
400124	3 (decimal): Routing 1 (Node address)
400125-8	0 (decimal): Unused

5.3.7.5 MSTR Block Peer Cop Health Operation

For general information on the MSTR Block, see Using the MSTR Modicon Ladder Logic Block.

Note: This operation is only available for Modbus Plus and not Modbus/TCP.

This operation reads a portion of a 12-register Peer Cop Health table. For a description of Peer Cop, see Using Modicon's Peer Cop to Read Global Data. This table consists of a collection of one bit per node address for each of three types of Peer Cop transfer. For the RMC, only the Global Input portion is useful since Specific Inputs and Outputs are not used. Each bit that is set indicates that the device with that address is connected using Peer Cop:

Status Type	Word Index	MSB							Node
Global Input	0	16	15	14	13	12	11	10	9
	1	32	31	30	29	28	27	26	25
	2	48	47	46	45	44	43	42	41
	3	64	63	62	61	60	59	58	57
Specific Output	4	16	15	14	13	12	11	10	9
	5	32	31	30	29	28	27	26	25
	6	48	47	46	45	44	43	42	41
	7	64	63	62	61	60	59	58	57
Specific Input	8	16	15	14	13	12	11	10	9
	9	32	31	30	29	28	27	26	25
	10	48	47	46	45	44	43	42	41
	11	64	63	62	61	60	59	58	57

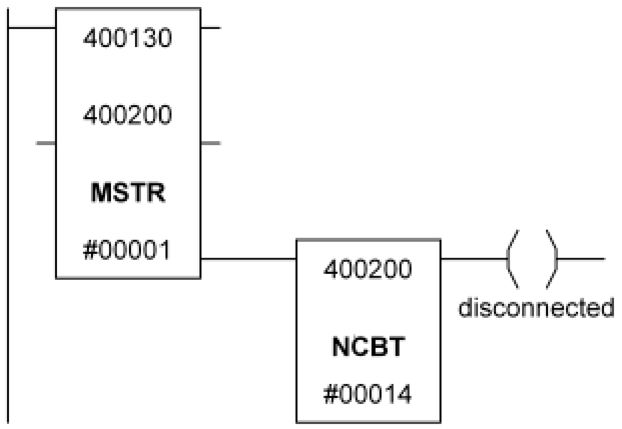
Control Block

The nine control-block registers are assigned in a Peer Cop Health operation as shown:

Register	Content
1st	**Operation**: 9
2nd	**Error Status**: This register is filled by the operation. See MSTR Block Error Codes for a complete list.
3rd	**Length**: Indicates the number of words to transfer.
4th	**Word Index**: Gives the index of the first Peer Health word to read.
5th-9th	**Routing 1-5**: Unused. Set zeros.

Example

Suppose that we have an RMC device on a Modbus Plus network with a node address of 3. We want to set a coil called ”r;disconnected” when the RMC is not found on the network. We will read one of the four Peer Cop Global Input Health registers into register 400200. We must first choose a location for the 9-register control block. In this example, 400130 to 400138 is the location. The following network would be used:



The MSTR block indicates it will do an operation using the control block beginning at 400130, using the one register at 400200. When this block is finished, we look at the third least significant bit, which represents node 3. If this bit is cleared, then the RMC is disconnected.

The next step is to set up the control block:

Register	Content
400130	9 (decimal): Operation type: Peer Cop Health
400131	0000 (hex): Error status: filled in by function
400132	1 (decimal): Length: we only want one register
400133	0 (decimal): Word Index: we want the first word in the table
400134-8	0 (decimal): Unused

5.3.7.6 MSTR Block Error Codes

The following list of error codes are returned by the MSTR function block:

Error Code (Hex)	Meaning
1001	User-initiated abort
2001	Invalid operation type
2002	User parameter changed
2003	Invalid length
2004	Invalid offset
2005	Invalid length + offset

2006	Invalid slave device data area
2007	Invalid slave device network area
2008	Invalid slave device network routing
2009	Route equal to your own address
200A	Attempting to obtain more global data registers than available
200B	Conflict with Peer Cop (Read Global Data cannot be used if Peer Cop is used)
30xx	Modbus slave exception response. See error values 3001-3007 below for specific errors:
3001	Slave device does not support the requested operation
3002	Nonexistent slave device registers requested
3003	Invalid data value requested
3005	Slave has accepted long-duration program command
3006	Function can’t be performed now—a long-duration command in effect
3007	Slave rejected long-duration program command
4001	Inconsistent Modbus slave response
5001	Inconsistent network response
6mxx	Routing failure. ’r;m’ indicates the routing entry at fault (0=Routing 1, 1=Routing 2, etc). See error values 6m01-6m80 below for specific errors:
6m01	No response received. For a local node, if m is 0, the node address was incorrect; check the node address. If m is 1, the data path was incorrect; ensure that a data path between 1 and 8 is used. See MSTR Block Read or MSTR Block Write for details.
6m02	Program access denied
6m03	Node off-line and unable to communicate
6m04	Exception response received
6m05	Router node data paths busy
6m06	Slave device down

6m07	Bad destination address
6m08	Invalid node type in routing path
6m10	Slave has rejected the command. Check the routing information to see if a valid data path (1-8) has been entered after the node address.
6m20	Initiated transaction forgotten by slave device
6m40	Unexpected master output path received
6m80	Unexpected response received
F001	Wrong destination node specified for the MSTR operation

5.4 PROFIBUS-DP

5.4.1 PROFIBUS-DP

PROFIBUS-DP is an RMC100 communication card that can be purchased and will be installed to the left of the RMC100 CPU. This card allows communication via the PROFIBUS-DP open industrial Fieldbus. This Fieldbus is vendor independent and therefore a large range of Programmable Controllers (P/C) or other PROFIBUS masters can control the RMC100.

The RMC100 performs as a PROFIBUS-DP slave. Therefore, a PROFIBUS-DP master must control it. To the RMC100, the type of the actual master—whether a Programmable Controller or Personal Computer—is unimportant. All will communicate with the RMC100 in the same manner.

RMC100 PROFIBUS-DP Specifications

- Operating baud-rates: 9.6Kbaud to 12Mbaud
- Manufacturer Identifier Number: 0x1630
- Supported: Freeze mode, sync mode, automatic baud-rate detect, set slave address
- Modularity: the RMC100 is a modular station that can have one module selected at a time.

PROFIBUS-DP Front Panel

The front panel of the PROFIBUS-DP communication module has both an LED and a 9-pin connector. The LED will be lit green when the module is communicating with the PROFIBUS-DP master. The 9-pin connector is used for connecting the module to the other PROFIBUS-DP devices via a standard PROFIBUS-DP cable.

RMC PROFIBUS-DP Modes

The latest RMC PROFIBUS-DP modules can operate in either of two modes: Compact Mode or Message Mode. Furthermore, Compact Mode can be divided into two sub-modes: Compact Mode with Sync and Compact Mode without Sync.

Compact Mode keeps the number of words sent over the PROFIBUS to a minimum. This is desirable to keep network traffic down and more importantly to keep the number of registers required in the PLC or PC to a minimum. Compact mode requires only two words in and out per axis, plus, optionally, one additional synchronization word in and out.

Note: Compact Mode without Sync requires RMC CPU firmware dated 19990916 or later (or RMC beta firmware dated 19990727B or later) and GSD file (DELT1630.GSD) version 1.3 or newer.

Message Mode is intended for systems that can support a large amount of data being sent over the PROFIBUS and in which the master computer can have 96 input words and 64 output words reserved for every RMC module. If the system can support these requirements, the Message Mode gives constant access to up to 32 words of status information, and can write up to 59 words or read up to 63 words with a single command.

For more information on these two commands, select one of the following topics:

Note: Message Mode requires RMC CPU firmware dated 19990625 or later (or RMC beta firmware dated 19990412B or later) and GSD file (DELT1630.GSD) version 1.2 or newer.

- Using the PROFIBUS-DP Compact Mode
- Using the PROFIBUS-DP Message Mode

PROFIBUS-DP Configuration

There is very little to be done to configure the RMC to communicate on the PROFIBUS network. In fact, the only thing to do is set the Station Address. However, you must also add the RMC as a device to the master's configuration. This is more difficult to do and the exact steps vary for each master. Refer to PROFIBUS Configuration for instructions on setting the Station Address and guidelines for adding the RMC as a device to the network configuration.

5.4.2 PROFIBUS Configuration

The following steps are required to connect an RMC unit to a PROFIBUS network. Each is described below:

- **Set the RMC's Station Address**
  The RMC requires a station address to be selected.
- **Determine the Appropriate GSD Configuration Module**
  The RMC's GSD file has a number of configuration modules; only one may be selected. However, it is vital that the correct configuration module be selected for a given RMC unit.
- **Configure the PROFIBUS Network**
  The master device requires knowledge of all slave devices on the network and the configuration of each slave device.

Setting the RMC's Station Address

The steps above set the station addresses as expected by the master; this step sets the station address that the RMC unit actually responds to. Both must be set to the same value for the same unit. The RMC100 can use any station address between 0 and 126; the address is saved in the

Flash memory. The default station address is 126. There are two methods of changing this address:

- RMCWin has the ability to change the RMC100 station address. Use the following steps:

  1. Start RMCWin.
  2. Establish a connection between RMCWin and the RMC. See Connecting RMCWin to an RMC for details.
  3. On the Tools menu, click Module Configuration.
  4. In the Slots list, click the PROFIBUS-DP item.
  5. Click Slot options.
  6. In the Station Address box, type in the new station address.
  7. Click Update RMC.
  8. The Update Module Configuration dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module manually.
  9. In the RMC Configuration dialog box, click Close.

- A PROFIBUS-DP class-2 master can request a change in the address of the RMC. The method in which this is done varies for each piece of master software. Refer to the documentation for your PROFIBUS-DP master for details on this method.
  When changing the address via a class-2 master, it is possible to lock further address changes so further master requests to change the address are rejected. The method of doing this varies depending on the master software used.

Note: It is still possible to change the address from RMCWin as described above; doing so will remove the lock on changing the address from a class-2 master.

Note: When the station address is changed with this method, the data will be automatically stored to Flash. It is also unnecessary to issue an Update Flash command. It is also important to note that all current parameters will be stored in the Flash, not just the station address.

Determine the Appropriate GSD Configuration Module

The nomenclature of the configuration modules in the GSD file has changed for each GSD-file version. This section uses version 1.3's nomenclature. The following chart shows the corresponding names in the other versions:

Version 1.3	**Version 1.2**	**Version 1.1**	**Version 1.0**
Message Mode	RMC in Message Mode	--	--
n Axes with Sync	RMC with n axes	RMC with n Axes	RMC1n0-PROFI

n Axes and DI/O with Sync	RMC with n Axes and DI/O	RMC with n Axes and Sensor DI/O	--
n Axes without Sync	--	--	--
n Axes and DI/O without Sync	--	--	--

The DELT1630.GSD file contains many configuration module entries. When adding an RMC to your PROFIBUS network in the step below, you will need to select exactly one of these configuration modules. Each of these modules is assigned a title, which is often referred to by PROFIBUS configuration software as an Order Number.

The first factor to take into consideration when determining the correct GSD order number is which operation mode you intend to use: Compact (with or without Sync) or Message Mode. If you do not know the difference between these modes or have not yet determined which is right for your application, please read these topics.

If you are using Message Mode, then the correct configuration module to select in the GSD file is entitled Message Mode.

If you are using Compact Mode, then the correct configuration module depends on (1) whether you wish to use a synchronization register or not, and (2) the configuration of the RMC itself. Example order numbers in the GSD file include 2 Axes with Sync, 4 Axes without Sync, and 4 Axes and DI/O with Sync. To help you select the correct Compact Mode order number, do the following:

1. Start RMCWin
2. Connect to the RMC module to be configured.
3. On the Tools menu, click Module Configuration.
4. In the Slots list, click the PROFIBUS-DP item.
5. Click Slot options.
6. Look in the Acceptable GSD Config Modules list. The second and third entries are the appropriate Compact Mode with and without Sync choices respectively.

Configuring your PROFIBUS Network

The PROFIBUS network configuration is stored in the master PROFIBUS device. Creating this configuration is the most difficult step in making the network work. This step requires a device description file (GSD file) from each device in the network; the RMC's GSD file is named DELT1630.GSD and is installed by the RMCWin software package.

The following steps outline configuring your PROFIBUS-DP network. The steps are general because several master configuration programs are available, and each handles these steps differently.

1. If you are using Analog cards (12-bit or 16-bit), you must first select the mode of operation for each analog channel. This must be done first because it may change the count of axes the module contains. See Analog Transducer Configuration for details.

2. Open your PROFIBUS-DP master configuration program.
3. If you are modifying an existing PROFIBUS-DP network, open your current configuration file.
4. If you are creating a new PROFIBUS-DP network, you must create a new network, add a master device to the network, and select the baud rate of the network.
5. Add the DELT1630.GSD file to your configuration program's GSD database if it is not already there.
6. Add a Delta RMC Family slave device to the network. Refer to the sub-topic Determining the Appropriate GSD Configuration Module above for details on selecting the correct configuration module.
7. Configure the RMC Slave Device. This involves selecting the correct configuration module for your RMC module and application. Refer to the sub-topic Determining the Appropriate GSD Configuration Module above for details on selecting the correct configuration module.
8. Select the addresses that the master will use for the registers read from the RMC.
9. Add any other RMC devices you want on the same network. To do this, repeat steps 6 through 8.
10. Save your configuration.
11. Send the configuration to the master device. This step varies greatly depending on the type of master you use.

    If you are using COM PROFIBUS, SyCon, or SST Profibus Configuration, you should select one of the following topics for more detailed instructions:

- Configuring a PROFIBUS-DP Network with COM PROFIBUS
- Configuring a PROFIBUS-DP Network with SST Profibus Configuration
- Configuring a PROFIBUS-DP Network with SyCon

See also:

PROFIBUS-DP Overview

Using the PROFIBUS-DP Compact Mode

Using the PROFIBUS-DP Message Mode

5.4.3 Configuring a PROFIBUS-DP Network with COM PROFIBUS

Before reading this topic, you should read and understand PROFIBUS Configuration. This topic only gives a specific example of doing one step of the configuration process. In addition, Siemens may, and probably will, change the steps taken here slightly with each version of COM PROFIBUS.

The following steps have been tested with COM PROFIBUS versions 3.0 and 3.3:

1. Start COM PROFIBUS.
2. If you are modifying an existing PROFIBUS-DP network, open your current configuration file.
3. If you are creating a new PROFIBUS-DP network, you must create a new network and add a master device to the network.
   - On the File menu, click New.
   - In the Master & Host Selection dialog box, enter the Address of the master device (in most cases, 1 is a good choice), the Master Station Type, and the appropriate Host Station Type if available. Click OK.
   - On the Configure menu, click Bus Parameters.
   - In the Bus Parameters dialog box, set the Bus Profile to PROFIBUS DP and the Baud Rate to the desired rate. The RMC is capable of speeds up to 12000kBaud (12MBaud); check the rates supported by your master and other slaves. Click OK.
4. If it is not already in the database, add the DELT1630.GSD file to your configuration program's GSD database.
   - Copy the DELT1630.GSD file to the GSD directory under the COM PROFIBUS directory.
   - Copy the DELTRMCN.BMP file to the BITMAPS directory under the COM PROFIBUS directory.
   - On the File menu, click Scan GSD Files.
5. Add and configure a Delta RMC Family slave device to the network.
   - On the Configure menu, click New Slave.
   - In the PROFIBUS Address dialog box, select the desired station address, and click OK.
   - In the Family list, click Other.
   - In the Station Type list, click Delta RMC Family.
   - Click OK. In most cases you will be prompted at this time to select a Preset Configuration. Refer to the Determining the Appropriate GSD Configuration Module section in the PROFIBUS Configuration topic for guidance on selecting the correct option.
   - If you are not prompted to select a Preset Configuration, then continue with the following steps:
     - o Right-click on the RMC slave device icon, and click Configure from the shortcut menu.
     - o In the Configure: Delta RMC Family dialog box, click Order No.
     - o In the Select by Order Number dialog box, click the order number determined in the previous steps. Click Accept. Click Close.
     - o Click OK.
6. You may wish to assign register addresses within the master at this time. This is optional for many masters, because it can be done in the master configuration software. In fact, at times it is not possible to edit the register addresses.

- Right-click on the RMC slave device icon, and select Configure from the shortcut menu.
- Move the cursor to the top row's I Addr cell, and either enter the offset that you wish to access the data at, or click the Auto Addr. button. Do the same for the O Addr cell.
- Click OK.

7. Add any other RMC devices you want on the same network. To do this, repeat steps 5 and 6.
8. Save your configuration.
9. Send the configuration to the master device. This step varies greatly depending on the type of master you use.

5.4.4 Configuring a PROFIBUS-DP Network with SST Profibus Configuration

Before reading this topic, you should read and understand PROFIBUS Configuration. This topic only gives a specific example of doing one step of the configuration process. In addition, SST may, and probably will, change the steps taken here slightly with each version of SST Profibus Configuration.

The following steps have been tested with SST Profibus Configuration versions 1.2 and 1.4:

1. Start SST Profibus Configuration.
2. If you are modifying an existing PROFIBUS-DP network, open your current configuration file.
3. If you are creating a new PROFIBUS-DP network, you must create a new network and add a master device to the network:

- On the File menu, click New.
- The tree-type control on the left is the GSD library. It contains all devices known about. In this tree, expand Masters and SST and double-click on the SST master you will be using (e.g. SST-PFB-SLC).
- In the Station box, enter the station address of your master. Click OK.
- On the Online menu, click Network Properties.
- In the Baud Rate field, select your desired baud rate. The RMC is capable of speeds up to 12000kBaud (12MBps); check the rates supported by your master and other slave devices. Click OK.

4. Add the DELT1630.GSD file to your configuration program's GSD database.

This step may not be necessary. SST ships a GSD file for the RMC. However, as of this writing it shipped revision level 1.1 of the GSD file. Revision 1.2 or newer is required for using Message Mode, and Rev 1.3 is required for Compact Mode without Sync. To check the revision level, do the following:

- In the GSD library tree, expand Slaves and Delta Computer Systems, Inc. and right-click on the

Delta RMC Family entry. On the shortcut menu, click Properties.

- In the Labels tab, look at the Revision field. If the revision displayed is new enough for the features you will use, you do not need to update the GSD file.

To update the GSD file do the following:

- In the GSD library tree, expand Slaves and Delta Computer Systems, Inc. and right-click on the Delta RMC Family entry. On the shortcut menu, click Delete. Click Yes when asked.
- Click New Device, which is the left-most button on the GSD library toolbar.
- Browse to the DELT1630.GSD that shipped with RMCWin, select that file, and click Open.
- When you check the revision of the GSD file now (per the instructions above), you should see Rev 1.2 or newer.

5. Add and configure a Delta RMC Family slave device to the network.

   - In the GSD library tree, expand Slaves and Delta Computer Systems, Inc. and double-click on Delta RMC Family.
   - On the General tab, select the station address of your master in the Station field of the dialog that is displayed.
   - In the Modules tab, click Add. In the dialog that is displayed, select the appropriate module from the Available Modules list and click OK. For help on determining the appropriate configuration module, refer to the Determining the Appropriate GSD Configuration Module section of the PROFIBUS Configuration topic.
   - In the Address (or SLC Address) tab, review the default addressing and change it if necessary.
   - Click OK.

6. Add any other RMC devices you want on the same network. To do this, repeat step 5.
7. Save your configuration.
8. Send the configuration to the master device. This step varies depending on the master you selected.

5.4.5 Configuring a PROFIBUS-DP Network with SyCon

Before reading this topic, you should read and understand PROFIBUS Configuration. This topic only gives a specific example of doing one step of the configuration process. In addition, Hilscher may, and probably will, change the steps taken here slightly with each version of SyCon System Configurator.

The steps that follow came from SyCon System Configurator 2.082:

1. Start SyCon System Configurator.
2. If you are modifying an existing PROFIBUS-DP network, open your current configuration file.

3. If you are creating a new PROFIBUS-DP network, you must create a new network and add a master device to the network.

- On the File menu, click New. If you have multiple networks installed you will need to then select the network type: select PROFIBUS. If you are not given the option of selecting PROFIBUS, you may not have installed the PROFIBUS driver for SyCon.
- On the Insert menu, click Master. Move the cursor to the top device slot in the window area (the cursor will change to an arrow with an ’r;M’ next to it), and click to place the master device.
- In the Insert Master dialog box, select the desired master and click the Add>> button. Enter the station address, and click OK.
- On the Settings menu, click Bus Parameter.
- In the Bus Parameter dialog box, set the Baud Rate to the desired rate. The RMC is capable of speeds up to 12000kBits/s (12MBaud); check the rates supported by your master and other slaves. Click OK.

4. Add the DELT1630.GSD file to your configuration program’s GSD database.

- You must first have an open PROFIBUS project file.
- On the File menu, click Copy GSD.
- In the Open dialog box, navigate to the directory where the DELT1630.GSD file is located (by default it is installed in the main RMCWin directory), select that file and click Open.
- NOTE: This will not add the bitmap file. If you wish to use the bitmap file; copy the DELTRMCN.BMP file to the Fieldbus\Profibus\BMP directory under the SyCon directory (by default, the total path will be C:\Program Files\Hilscher GmbH\SyCon\Fieldbus\Profibus\BMP).

5. Add a Delta RMC Family slave device to the network.

- On the Insert menu, click Slave. Move the cursor to the next available device slot in the window area (the cursor will change to an arrow with an S next to it), and click to place the device.
- In the Insert Slave dialog box, click Delta RMC Family, and then click the Add>> button. Enter the station address, and click OK.

6. Configure the RMC Slave Device. This involves selecting the correct configuration module for your RMC module and application. Refer to the sub-topic Determining the Appropriate GSD Configuration Module in the PROFIBUS Configuration topic for details on selecting the correct configuration module.

- Right-click on the RMC slave device icon, and click Slave Configuration from the shortcut menu.
- In the Slave Configuration dialog box, from the list of order numbers, select the configuration module you selected in the previous steps.
- Click Append Module.
- You may change the I Addr. and O Addr. fields for the added module to set the offsets of the data.
- Click OK.

7. Add any other RMC devices you want on the same network. To do this, repeat steps 5 and 6.
8. Save your configuration.
9. Send the configuration to the master device. This step varies depending on the master you selected.

5.4.6 Compact Mode

5.4.6.1 Using the PROFIBUS-DP Compact Mode

Compact Mode is one of two modes that can be used with the RMC PROFIBUS-DP module. The other mode is called Message Mode, which is described Using the PROFIBUS-DP Message Mode. Be sure to read both topics and consider each carefully before choosing the mode you will use.

Compact Mode uses two 16-bit input registers and two 16-bit output registers per axis. These registers are described in detail in Input registers and Output registers. The role of PROFIBUS-DP is to carry the commands given from the master through the Output registers to the RMC100, and to carry requested data from the RMC100 through the Input registers back to the master. PROFIBUS-DP achieves this by continually sending these registers back and forth.

Compact Mode gives two choices for synchronizing this data within the PROFIBUS master, called Compact Mode with Sync and Compact Mode without Sync.

Compact Mode *with* Sync
This sub-mode provides one additional input and one additional output register called the Synchronization Input and Synchronization Output registers respectively. In this mode, commands are not processed by the RMC until the Synchronization Output register changes. When this register changes, all commands that change on all axes are processed. As a result the PROFIBUS master’s data-update cycle can be asynchronous with its programming cycle. For example, some PLCs do not guarantee that the PROFIBUS I/O will be done at the top of the ladder logic scan. Therefore the input register could change at any time during the PROFIBUS scan.

The following general steps should be taken in a control program:

- Write all new commands and command values to the Output registers.
- Increment the Output Synchronization Register.
- Wait for the Input Synchronization Register to match the Output Synchronization Register.
- Process data in the Input registers.
- Repeat the process.

Note: When using Wonderware’s InControl with sequential function charts, it is necessary to increment the Output Synchronization Register in a step by itself, rather than in a step that also updates the command or command values. The reason for this is that InControl updates all the output registers at once, and apparently can have the data sent to the RMC100 module in the middle of updating the block. Therefore, some commands can be separated from their command values.

Compact Mode *without* Sync

CAUTION: In this mode, the synchronization is left to be the PROFIBUS master's responsibility. This is very important to realize, because attempting to control the RMC over PROFIBUS in this sub-mode from a PLC that asynchronously updates the PROFIBUS data will not work.

No additional synchronization registers are added to the two registers per axis. Instead, the RMC processes a new command any time either output register for an axis changes. Two ways of controlling the synchronization are to have the PLC either (1) enforce updating the PROFIBUS data at the top of the scan (the SIMATIC TI505 can operate in this mode, but does not by default), or (2) provide a function block for explicitly updating the PROFIBUS data (the SIMATIC S7-300 series can use this option using the DPRD_DAT and DPWR_DAT system function codes).

Once all commands have been processed and the data in the Input Registers has been updated, the RMC100 copies the value from the Output Synchronization Register to the Input Synchronization Register. This should be used as a cue for the Master control program to process the data returned and/or issue the next commands.

See also:

Compact Mode Input Register Overview

Compact Mode Output Register Overview

Command Words for PROFIBUS-DP Compact Mode

Using the PROFIBUS-DP Message Mode

5.4.6.2 Compact Mode Input Register Overview

In addition to the Output registers, each RMC module requires two 16-bit registers per axis to hold status information. The first input register for each axis always holds the current axis Status word. The second input register can hold any of several values depending on the value of the command register for the given axis.

When a Sensor DI/O module is installed, the state of inputs 0-17 and outputs 0-7 are reflected in an additional two registers. The contents of these registers is given as follows:

First Sensor DI/O Input Register:

Bits 0-15	Sensor DI/O Inputs 0-15

Second Sensor DI/O Input Register:

Bits 0-1	Sensor DI/O Inputs 16-17
Bits 8-15	Sensor DI/O Outputs 0-7

If Compact Mode with Sync is used (versus Compact Mode without Sync), one additional 16-bit register to hold the Synchronization Input is required. See Using the PROFIBUS-DP Compact

Mode for details on using this register and the differences between these two sub-modes.

The order of the input registers for an n-axis module with a Sensor DI/O running in Compact Mode with Sync is shown below. I represents the RMC's input base address. If your module does not have a Sensor DI/O, drop the last two registers:

Register Number	**Contents**
I + 0	Synchronization Input
I + 1	Status for axis 0
I + 2	Specified by command for axis 0
I + 3	Status for axis 1
I + 4	Specified by command for axis 1
:	:
I + 2*n + 1	Status for axis n
I + 2*n + 2	Specified by command for axis n
I + 2*n + 3	Sensor DI/O Inputs 0-15 (if present)
I + 2*n + 4	Sensor DI/O Inputs 16-17, Outputs 0-7 (if present)

The following chart shows the same for an RMC running in Compact Mode without Sync:

Register Number	**Contents**
I + 0	Status for axis 0
I + 1	Specified by command for axis 0
I + 2	Status for axis 1
I + 3	Specified by command for axis 1
:	:
I + 2*n + 0	Status for axis n
I + 2*n + 1	Specified by command for axis n
I + 2*n + 2	Sensor DI/O Inputs 0-15 (if present)
I + 2*n + 3	Sensor DI/O Inputs 16-17, Outputs 0-7 (if present)

For most commands, the value returned in the second register for each axis is selected using the Status Area Request bits in the command register. Refer to the individual commands for exceptions. The possible fields that can be returned in the second register when is it controlled by the Status Area Request bits are:

Status Area Request	**StatusRegister**
0 (0000)	Command Position
1 (0001)	Target Position
2 (0010)	Actual Position
3 (0011)	Transducer Counts
4 (0100)	Status Word
5 (0101)	Drive Output
6 (0110)	Actual Speed
7 (0111)	Null Drive
8 (1000)	Current Step
9 (1001)	Link Value
10-15 (1010-1111)	Reserved

Compact Mode *with* Sync Example

Suppose you have an RMC100-M1-PROFI and would like the Status and Actual Position for axis 0, and you would like the Status and Drive for axis 1. You would send commands with the following format:

```
          SAR CMND INDX (HEX)

O+1 0000|0010|XXXX|XXXX (02XX)        Requests Actual Position

+2  XXXX|XXXX|XXXX|XXXX (XXXX)        Data for command on axis 0

O+3 0000|0101|XXXX|XXXX (05XX)        Requests Drive Output

+4  XXXX|XXXX|XXXX|XXXX (XXXX)        Data for command on axis 1
```

After the Synchronization Output register is incremented, the RMC will process the commands and update the Synchronization Input register to match. At this time, the input registers would hold the following values:

```
I+1 XXXX|XXXX|XXXX|XXXX (XXXX)        Status of axis 0

+2  XXXX|XXXX|XXXX|XXXX (XXXX)        Actual Position of axis 0

I+3 XXXX|XXXX|XXXX|XXXX (XXXX)        Status of axis 1

+4  XXXX|XXXX|XXXX|XXXX (XXXX)        Drive of axis 1
```

Compact Mode *without* Sync Example

Suppose you have an RMC100-M1-PROFI and would like the Status and Actual Position for axis 0, and you would like the Status and Drive for axis 1. You would send commands with the following format:

```
          SAR CMND INDX (HEX)

O+0 0000|0010|XXXX|XXXX (02XX)        Requests Actual Position

+1  XXXX|XXXX|XXXX|XXXX (XXXX)        Data for command on axis 0

O+2 0000|0101|XXXX|XXXX (05XX)        Requests Drive Output

+3  XXXX|XXXX|XXXX|XXXX (XXXX)        Data for command on axis 1
```

When the RMC sees the changes to the SAR field, it processes the commands. As a result, the input registers would hold the following values:

```
I+0 XXXX|XXXX|XXXX|XXXX (XXXX)        Status of axis 0

+1  XXXX|XXXX|XXXX|XXXX (XXXX)        Actual Position of axis 0

I+2 XXXX|XXXX|XXXX|XXXX (XXXX)        Status of axis 1

+3  XXXX|XXXX|XXXX|XXXX (XXXX)        Drive of axis 1
```

See also:

Using the PROFIBUS-DP Compact Mode

Compact Mode Output Register Overview

Command Words for PROFIBUS-DP Compact Mode

5.4.6.3 Compact Mode Output Register Overview

In addition to the Input registers, each RMC module requires two 16-bit registers per axis to hold commands. The two command registers for each axis are processed each time the Synchronization Output register is changed.

When a Sensor DI/O module is installed, an additional two outputs words are reserved but are currently unused.

If Compact Mode with Sync is used (versus Compact Mode without Sync), one additional 16-bit register to hold the Synchronization Output register. See Using the PROFIBUS-DP Compact Mode for details on using this register and the differences between these two sub-modes.

The order of the output registers for an n-axis module with a Sensor DI/O running in Compact Mode with Sync is shown below. O represents the RMC's output base address. If your module does not have a Sensor DI/O, drop the last two registers:

Register Number	**Contents**
O + 0	Synchronization Input
O + 1	Command – axis 0
O + 2	Data out – axis 0
O + 3	Command – axis 1
O + 4	Data out – axis 1
:	:
O + 2*n + 1	Command – axis n
O + 2*n + 2	Data out – axis n
O + 2*n + 3	Reserved by Sensor DI/O
O + 2*n + 4	Reserved by Sensor DI/O

The following chart shows the same for an RMC running in Compact Mode without Sync:

Register Number	**Contents**
O + 0	Synchronization Input
O + 1	Command – axis 0
O + 2	Data out – axis 0
O + 3	Command – axis 1
:	:
O + 2*n + 0	Command – axis n
O + 2*n +	Data out – axis n

1	
O + 2*n + 2	Reserved by Sensor DI/O
O + 2*n + 3	Reserved by Sensor DI/O

PROFIBUS-DP has a much larger set of commands than the normal ASCII commands that can be used in the Command field, although ASCII commands are still available. For a complete description of using the Command and Data out registers, refer to Command Words for PROFIBUS-DP Compact Mode.

See also:

Using the PROFIBUS-DP Compact Mode

Compact Mode Input Register Overview

Command Words for PROFIBUS-DP Compact Mode

5.4.7 Message Mode

5.4.7.1 Using the PROFIBUS-DP Message Mode

Message Mode is one of two modes that can be used with the RMC PROFIBUS-DP module. The other mode is called Compact Mode, which is described in Using the PROFIBUS-DP Compact Mode. Be sure to read both topics and consider each carefully before choosing the mode you will use.

Message Mode is so called because it imposes a messaging type interface on PROFIBUS-DP. That is, PROFIBUS-DP only supports cyclic data transfer between a master and a slave. However, the RMC has nearly 64K registers, all of which cannot possibly be sent over PROFIBUS every PROFIBUS scan. Therefore, blocks of these registers can be written to and read from the RMC by using a messaging mechanism: the request is placed in one block of registers, and a response is received in the other block of registers.

In addition to the Command and Response blocks of data, there is also a third block of data—the Status Block—that includes status information that is constantly available. A summary of the Message Mode register blocks follows:

- **Status Block**
  This is a block of 32 contiguous input registers that are constantly updated by the RMC. The values will be updated every 2ms when no commands are in progress, and every 6ms when a command is in progress. This block is otherwise independent from the Command and Response blocks. The contents of these 32 registers are configurable using the Status Map editor. See Using the Status Map Editor for details.
- **Command Block**
  This is a block of 64 contiguous output registers. These registers are sent from the PLC or PC to the RMC constantly via PROFIBUS-DP, however, the last register in this block contains two flag

bits that are toggled to indicate a request. The command block is described in further detail below.

Note: Users of Compact Mode should be aware that commands issued over Message Mode are always handled. Specifically this means that if you send a command multiple times it will be processed every time it is received. Under Compact Mode duplicate commands are ignored. This means that Compact Mode users who want to re-issue a command (for example, an 'E' command to start an event sequence) need to toggle the case of the command (for example, toggle between 'E' and 'e').

- **Response Block**
  This is a block of 64 contiguous input registers. These registers are sent from the RMC to the PLC or PC constantly via PROFIBUS-DP, however, the values will not change unless a new request was triggered using the flag bits in the last register of the command block. This mechanism is described below.

Therefore, Message Mode requires 96 input words and 64 output words on the PROFIBUS master. Not all PROFIBUS masters support this; if your master does not support this many registers, you will need to use Compact Mode.

Messaging in Greater Detail

The Command Block has the following structure:

Register	Description
0-58	Write Data
59	Write address (0-65,535). See the RMC Register Map (PROFIBUS-DP Message Mode) topic for a description of all RMC registers and their addresses.
60	Write length (in words; 0-59)
61	Read address (0-65,535). See the RMC Register Map (PROFIBUS-DP Message Mode) topic for a description of all RMC registers and their addresses.
62	Read length (in words; 0-63)
63	Output Synchronization Register: Bit 15 - Read Request Bit 14 - Write Request

The Response Block has the following structure:

Register	Description
0-62	Read Data
63	Input Synchronization Register Bit 15 - Read Acknowledge Bit 14 - Write Acknowledge

To request a read from the RMC, use the following steps:

- Wait until the Read Request bit is equal to the Read Acknowledge bit. If they are not equal, the RMC is currently processing a read request.
- Set the Read Address and Read Length output registers. See RMC Register Map (PROFIBUS-DP Message Mode) topic for a description of all RMC registers and their addresses.
- Toggle the Read Request bit.

- Wait until the Read Request bit is equal to the Read Acknowledge bit. When they are equal, the RMC will have updated the Read Data area with the requested data.
- Use the data in the Read Data area of the Response Block input registers. Make sure that you do not change the Read Request bit until you are done with the data in the Read Data area.

To request a write to the RMC, use the following steps:

- Wait until the Write Request bit is equal to the Write Acknowledge bit. If they are not equal, the RMC is currently processing a write request.
- Copy the values you wish to write to the RMC into the Write Data area of the Command Block output registers.
- Set the Write Address and Write Length output registers. See RMC Register Map (PROFIBUS-DP Message Mode) topic for a description of all RMC registers and their addresses.
- Toggle the Write Request bit.
- Wait until the Write Request bit is equal to the Write Acknowledge bit. When they are equal, the RMC has received the data written to it.

The RMC processes reads and writes separately. That is, it is not necessary for a write to complete before starting a read, and in fact, you can simultaneously request both a read and a write; the write to the RMC will occur first, and the read from the RMC will occur after the write has completed. When both are done, both acknowledge bits will be toggled simultaneously.

To further clarify the ordering, keep these basic rules in mind in Message Mode:

- Do change all write data, the write address, and the write length before toggling the Write Request bit.
- Do change the read address and read length before toggling the Read Request bit.
- Do not change the Read Request bit after a read request until you have processed the data in the Read Data area.
- Do not change the write data, the write address, or the write length when the Write Request bit in the output synchronization register does not match the Write Acknowledge bit in the input synchronization register.
- Do not change the read address or the read length when the Read Request bit in the output synchronization register does not match the Read Acknowledge bit in the input synchronization register.

Using Splines with PROFIBUS

Wait bit 7 is used by the Profibus DP communications so the user can tell when spline downloads are finished. The Profibus DP spline processing routine sets wait bit 7 when a download to the spline area is detected. The user can then use the step table and the Check Wait Bits link to be sure the spline is ready before executing the spline.

5.4.7.2 RMC Register Map (PROFIBUS-DP Message Mode)

Tip: The RMCWin Address Tool provides an easy way to identify addresses in the RMC. Simply open the Address Tool and then move the cursor to any field in RMCWin that represents an RMC Register, and the Address Tool will display the address in the address format of your choice. See Address Tool for details.

The RMC module has 64K (65536) 16-bit registers that can be read from or written to over Ethernet, Modbus Plus, and PROFIBUS-DP. Each register is assigned an address. However, under the different communication methods, different addressing schemes are used. This topic describes the addressing through the PROFIBUS-DP Message Mode. For details on addressing from other modules refer to the following topics:

- RMC Register Map (Allen-Bradley)
- RMC Register Map (Automation Direct)
- RMC Register Map (Modbus/TCP)
- RMC Register Map (Omron FINS)
- RMC Register Map (Siemens TI505)
- RMC Register Map (Siemens S7)
- RMC Register Map (Modbus Plus)

To communicate with any PROFIBUS-DP master, the RMC requires the PROFIBUS-DP communication module. The PROFIBUS-DP module supports two communication modes: Compact and Message Mode. Only Message Mode uses this register map. Under PROFIBUS Message Mode, the RMC registers are addressed as values 0-65535. For details on reading and writing these registers, see Using the PROFIBUS-DP Message Mode.

Status Registers:
These registers can only be read; writes are ignored.

PROFIBUS Address	**Register Description**
0	Axis 0 Command Position
1	Axis 0 Target Position
2	Axis 0 Actual Position
3	Axis 0 Transducer Counts
4	Axis 0 Status Word
5	Axis 0 Drive
6	Axis 0 Actual Speed
7	Axis 0 Null Drive
8	Axis 0 Event Step
9	Axis 0 Link Value
10-19	Same as above but for axis 1
20-29	Same as above but for axis 2
30-39	Same as above but for axis 3

40-49	Same as above but for axis 4
50-59	Same as above but for axis 5
60-69	Same as above but for axis 6
70-79	Same as above but for axis 7

Command Registers:
These registers can be read or written.

PROFIBUS Address	Register Description
80	Axis 0 Mode Word
81	Axis 0 Acceleration
82	Axis 0 Deceleration
83	Axis 0 Speed
84	Axis 0 Command Value
85	Axis 0 Command
86-91	Same as above but for axis 1
92-97	Same as above but for axis 2
98-103	Same as above but for axis 3
104-109	Same as above but for axis 4
110-115	Same as above but for axis 5
116-121	Same as above but for axis 6
122-127	Same as above but for axis 7

Parameter Registers:
These registers can be read or written. Changes to these registers do not take effect until a Set Parameters (P) command is executed.

PROFIBUS Address	Register Description
128	Axis 0 Configuration Word
129	Axis 0 Scale

130	Axis 0 Offset
131	Axis 0 Extend Limit
132	Axis 0 Retract Limit
133	Axis 0 Proportional Gain
134	Axis 0 Integral Gain
135	Axis 0 Differential Gain
136	Axis 0 Extend Feed Forward
137	Axis 0 Retract Feed Forward
138	Axis 0 Extend Acceleration Feed Forward
139	Axis 0 Retract Acceleration Feed Forward
140	Axis 0 Dead Band Eliminator
141	Axis 0 In Position Window
142	Axis 0 Following Error
143	Axis 0 Auto Stop
144-159	Same as above but for axis 1
160-175	Same as above but for axis 2
176-191	Same as above but for axis 3
192-207	Same as above but for axis 4
208-223	Same as above but for axis 5
224-239	Same as above but for axis 6
240-255	Same as above but for axis 7

Event Step Table Registers:
These registers can be read or written.

PROFIBUS Address	Register Description
256	Step 0 Mode Word
257	Step 0 Acceleration
258	Step 0 Deceleration

259	Step 0 Speed
260	Step 0 Command Value
261	Step 0 Command/Commanded Axes
262	Step 0 Link Type/Link Next
263	Step 0 Link Value
256+n*8	Step n (0-255) Mode Word
257+n*8	Step n (0-255) Acceleration
258+n*8	Step n (0-255) Deceleration
259+n*8	Step n (0-255) Speed
260+n*8	Step n (0-255) Command Value
261+n*8	Step n (0-255) Command/Commanded Axes
262+n*8	Step n (0-255) Link Type/Link Next
263+n*8	Step n (0-255) Link Value

Input to Event Table Registers:
These registers can be read or written.

PROFIBUS Address	**Register Description**
2304	Event Step for Axis 0 on Input 0 Rising Edge
2305	Event Step for Axis 1 on Input 0 Rising Edge
2306	Event Step for Axis 2 on Input 0 Rising Edge
2307	Event Step for Axis 3 on Input 0 Rising Edge
2308	Event Step for Axis 4 on Input 0 Rising Edge
2309	Event Step for Axis 5 on Input 0 Rising Edge
2310	Event Step for Axis 6 on Input 0 Rising Edge

2311	Event Step for Axis 7 on Input 0 Rising Edge
2312 + n	Event Step for Axes n (0-7) on Input 1 Rising Edge
:	:
2424 + n	Event Step for Axes n (0-7) on Input 15 Rising Edge
2432 + n	Event Step for Axes n (0-7) on Input 0 Falling Edge
:	:
2552 + n	Event Step for Axes n (0-7) on Input 15 Falling Edge

Status Map Registers:

These registers can be read or written, although you should not manually change the values in this table. You should use the Status Map Editor to change this table and then download it to the RMC. You may then read this table into the PLC and send the table to the RMC each time the PLC is restarted.

PROFIBUS Address	**Register Description**
2560-2591	Status Map Entries

Plot Type Registers:

The plot type registers can be read or written. The values that are read indicate the extra plot information in the current graph. The values written to these registers tell the controller which extra plot information to obtain on the next plot. For these registers, the following values are used:

- 0: Extra position precision
- 1: Command and Command Value
- 2: Event Step and Link Value
- 3: Raw Transducer Counts
- 4: Internal Target and Actual Speeds
- 5: Integral Drive

For more information on these four types of plot information, see Selecting the Data to Plot and Reading Plots from the Communication Module.

PROFIBUS

Address	Register Description
2624	Axis 0 plot type
2625	Axis 1 plot type
2626	Axis 2 plot type
2627	Axis 3 plot type
2628	Axis 4 plot type
2629	Axis 5 plot type
2630	Axis 6 plot type
2631	Axis 7 plot type

Digital (Discrete) I/O Registers:
These registers indicate the current state of the digital inputs and outputs. These registers may only be read; writes will be ignored, as this product does not support forcing inputs or outputs.

Because the different PLCs label bit numbers differently, the following chart is provided to show the mapping between the devices:

	MSB								
RMC bit #	15	14	13	12	11	10	9	8	7
Allen-Bradley bit #	15	14	13	12	11	10	9	8	7
Modicon bit #	1	2	3	4	5	6	7	8	9
TI505 bit #	1	2	3	4	5	6	7	8	9

The bit numbers listed in the table below are in RMC format (0 is LSB, 15 is MSB):

PROFIBUS Address	Register Description
2632	CPU Digital Inputs 0 and 1 in LSBs of low byte, Outputs 0 and 1 in LSBs of high byte
2633	Unused
2634	Unused
2635	Sensor Digital I/O Inputs 0-15

2636	Sensor Digital I/O Inputs 16-17 (stored to two LSBs)
2637	Sensor Digital I/O Outputs 0-7 in high byte (low byte unused)
2638	Unused
2639	Unused

Plot Time Registers:

The Plot Time interval is configurable on the RMC. This interval indicates the number of control loops between each sample in a plot. Therefore, if the control loop is 0.976ms (e.g. RMC100-M1), this indicates roughly the number of milliseconds between samples. If the control loop is 1.953ms (e.g. RMC100-M4), this indicates half of the number of milliseconds between samples.

These registers may be read or written. When read, they indicate the plot interval of the currently gathered plot. When written, they set the plot interval the RMC should use for the next plot it will gather. When the RMC starts the next plot, it copies the requested plot interval into the currently used plot interval.

PROFIBUS Address	Register Description
2640	Axis 0 plot time interval
2641	Axis 1 plot time interval
2642	Axis 2 plot time interval
2643	Axis 3 plot time interval
2644	Axis 4 plot time interval
2645	Axis 5 plot time interval
2646	Axis 6 plot time interval
2647	Axis 7 plot time interval

Last Parameter Error Registers:

Note: To use these registers through PROFIBUS, you must have RMC CPU firmware dated 19990819 or later.

Each of these read-only registers holds the number of the last parameter error generated on an axis. This is useful for determining the cause of the Parameter Error bit in the status word. For a description of the values read from these registers, see Parameter Error Values.

PROFIBUS Address

	Register Description
2648	Last parameter error on axis 0
2649	Last parameter error on axis 1
2650	Last parameter error on axis 2
2651	Last parameter error on axis 3
2652	Last parameter error on axis 4
2653	Last parameter error on axis 5
2654	Last parameter error on axis 6
2655	Last parameter error on axis 7

Firmware Date Registers:

Note: To use these registers through PROFIBUS, you must have RMC CPU firmware dated 19990819 or later.

These read-only registers hold information about the firmware versions in the RMC100 CPU module. The Boot and Loader firmware versions have no affect on the actual performance of the RMC and therefore can usually be ignored.

PROFIBUS Address	Register Description
2656	Boot firmware month (MSB) and day (LSB)
2657	Boot firmware year
2658	Loader firmware month (MSB) and day (LSB)
2659	Loader firmware year
2660	Control firmware month (MSB) and day (LSB)
2661	Control firmware year
2662	Control firmware Beta Code. This will be 0 for standard release firmware, ’r;B’ for Beta firmware, or ’r;SI’ for Superimposed firmware.
2663	Feature code. This register is mainly reserved for internal use but does have two bits that may be useful to some users:

- If bit 1 (value 0x0002) is set, the control loop is 2 ms, otherwise the control loop is 1 ms.
- If bit 0 (value 0x0001) is set, a sensor DI/O is present, otherwise there is no sensor DI/O.

Reserved Registers:

Reading these values will return zero, and writes are ignored.

PROFIBUS Address	Register Description
2664-12287	Unused

Spline Download Area:

These registers are write only. Reading them will return zero. This area is used to download intervals and points in a spline. This is a much more efficient alternative to using individual New Spline Point and Set Spline Interval/End Segment commands. For details on using this Spline Download Area, see Downloading Splines to the RMC.

PROFIBUS Address	Register Description
12288-16383	Spline Download Area

Wait bit 7 is used by the Profibus DP communications so the user can tell when spline downloads are finished. The Profibus DP spline processing routine sets wait bit 7 when a download to the spline area is detected. The bit is cleared when the download is done. The user can then use the step table and the Check Wait Bits link to be sure the spline is ready before executing the spline. See the Set and Reset Wait Bits Command topic for more details on the wait bits.

Plot Registers:

These registers can only be read; writes are ignored.

Note: Reading plots is not a trivial task; for further details, see Reading Plots from the Communication Module.

PROFIBUS Address	Register Description
16384-22527	Plot data for axis 0
22528-28671	Plot data for axis 1

28672-34815	Plot data for axis 2
34816-40959	Plot data for axis 3
40960-47103	Plot data for axis 4
47104-53247	Plot data for axis 5
53248-59391	Plot data for axis 6
59392-65535	Plot data for axis 7

5.5 Serial (RS-232/422/485)

5.5.1 RMC SERIAL Overview

The RMC SERIAL module adds a single serial port to the RMC for communications with other devices such as HMIs and PLCs. This port cannot be used for communications with RMCWin or the RMCLink ActiveX Control and .NET Assembly Component. However, that one serial port has a great amount of flexibility, as summarized below:

- **Protocols**: Modbus/RTU, Allen-Bradley DF1 (Full- and Half-duplex), Mitsubishi Binary
- **Baud Rate**: 9,600 to 115,200.
- **Parity**: Odd, Even, or None
- **Line Drivers**: RS-232, RS-422, and RS-485
- **Flow Control (RS-232 only)**: None or Hardware (RTS/CTS)
- **Termination and Biasing (RS-422/485 only)**: Both software selectable on Rx and Tx wire pairs

Note: Hardware revision 1 of the RMC SERIAL module is more limited in its termination and biasing options. See RS-422/485 Termination and Biasing for details.

In addition, the RMC SERIAL has been designed to facilitate adding additional protocols and higher baud rates as driven by customer demand. Therefore, if your serial protocol or baud rate is not currently supported, contact Delta to request a new protocol or feature.

The RMC SERIAL module is passive in that it always waits for a request from a master serial device before responding. It does not initiate communications.

Because of the large number of options offered by the RMC SERIAL, setting up the RMC SERIAL can be intimidating to users new to serial communications. Therefore, we recommend reading each of the following topics carefully before designing your serial network:

- Configuring the RMC SERIAL
- Line Drivers: RS-232/422/485
- Serial Network Topologies
- RS-232 Wiring for the RMC SERIAL
- RS-422/485 Wiring for the RMC SERIAL
- RS-422/485 Termination and Biasing

Finally, the following topics describe how to use each protocol supported by the RMC SERIAL:

- Using Modbus/RTU with the RMC SERIAL
- Using DF1 (Full- and Half-Duplex) with the RMC SERIAL
- Using the Mitsubishi Bidirectional Protocol with the RMC SERIAL
- Using the Mitsubishi No Protocol with the RMC SERIAL

5.5.2 Configuration and Wiring

5.5.2.1 Configuring the RMC SERIAL

The RMC SERIAL module adds a single serial port to the RMC for communications with other devices such as HMIs and PLCs. However that one serial port has a great amount of flexibility. It supports most standard serial port options (e.g. baud rate and parity), several protocols, and several line drivers with their associated options (e.g. flow control, termination, and biasing).

These settings are changed through RMCWin and saved in the RMC CPU's Flash memory. To change these settings:

1. On the Tools menu, click Module Configuration.
2. In the Slots list, click the Serial line.
3. Click Slot Options.
4. In the Serial Module Options dialog box, make all changes.
5. Click Update RMC.

   This step will save the settings to Flash memory.

   The Update Module Configuration dialog box will be displayed to indicate the progress. If the RMC could not be reset automatically, you may be prompted to remove power and reset the RMC manually.
6. In the RMC Configuration dialog box, click Close.

The following options are available in the Serial Module Options dialog box:

- **Protocol**: Select the protocol supported by your master device. The Test Mode protocol is a special one used to test the communications. With this protocol, for each character the RMC receives, it will add one to it and respond with that character. For example, if you send the character A, it will respond with B. This can be used from HyperTerminal included in Windows 98/NT/2000/Me/XP.

- **Serial Port Settings**: The follow settings define how data is sent over the wire:

  Note: Some of these options may be disabled depending on which protocol was selected. For example, many protocols require eight data bits, and as such seven data bits is not available when these protocols are selected.

  - **Baud Rate**: Select the baud rate from 9,600 to 115,200. This must match the other device(s) on the network.

  - **Data Length**: Select seven (7) or eight (8) data bits. Most protocols require eight data bits. This must match the other device(s) on the network.

  - **Parity**: Select which type of parity error checking you want to include. This must match the other device(s) on the network.

  - **Line Driver**: Select the line driver you wish to use. For further details on the line drivers, see Line Drivers: RS-232/422/485.

  Each line driver also offers advanced settings. These are described below:

    - o **Advanced RS-232 Settings**: The type of Flow Control can be selected. If None is selected, then no flow control is used. If Hardware (RTS/CTS) is selected, then the RTS and CTS lines on the serial cable are used to control when a device can send data to avoid overflowing the receiver's buffer. This option is most often not used, as the protocols keep the packet sizes small to avoid overruns. This setting must match in both devices.

    - o **Advanced RS-422/485 Settings**: This option allows enabling and disabling the biasing and termination circuits for the receiver and transmitter circuits (4-wire) or transceiver circuit (2-wire). These options are described in detail in RS-422/485 Termination and Biasing.

- **Protocol-Specific Settings**: Depending on the protocol selected, additional options may be available under this heading. The most common setting is a node address. See the appropriate protocol-specific help topic for details:

  - Using Modbus/RTU with the RMC SERIAL

  - Using DF1 (Full- and Half-Duplex) with the RMC SERIAL

  - Using the Mitsubishi Bidirectional Protocol with the RMC SERIAL

  - Using the Mitsubishi No Protocol with the RMC SERIAL

5.5.2.2 RMC SERIAL Firmware Screen

The Firmware tab on the Serial Module Options dialog box holds the following information:

- **Communication Program Version**
  This field gives the version of the main serial program currently running in the RMC SERIAL module. This field will display "Debugger" if no firmware is currently loaded. To update your

firmware, see Downloading New Serial/Ethernet Firmware.

- **Boot Version**
  This field gives the version of the Boot firmware in the RMC SERIAL module.

- **Loader Version**
  This field gives the version of the Loader firmware used for updating the main communication program.

- **Hardware Revision**
  The hardware revision of the RMC SERIAL module is displayed. Unlike the above versions, the hardware revision cannot be field-upgraded.

  The Firmware tab of the Serial Module Options dialog box has the following commands:

- **Update**
  This command initiates the sequence to update the main communication program in the RMC SERIAL module. Use this button only when directed to do so by Delta technical support.

- **Update B/L**
  This command initiates the sequence to update the Boot and Loader firmware in the RMC SERIAL module. Use this button only when directed to do so by Delta technical support.

5.5.2.3 Line Drivers: RS-232/422/485

The RMC SERIAL module supports three different line drivers: RS-232, RS-422, and RS-485. The line driver is selected through RMCWin. The following chart compares these three line drivers:

	RS-232	**RS-422**	**RS-485 (2-wire)**	**RS-485 (4-wire)**
Duplex	Full	Full	Half	Half/Full*
Differential?	No	Yes	Yes	Yes
Topology	Point-to-point	Point-to-point	Point-to-point, Multi-drop	Point-to-point, Multi-drop
Wires	3 or 5**	4 + CMN	2 + CMN	4 + CMN
Max Length* **	50-100 ft	4000 ft	4000 ft	4000 ft
Flow Control	None, or Hardware	None	None	None

* Four-wire RS-485 can be full-duplex if the topology is point-to-point. It can also be somewhat full-duplex in a multi-drop scenario in that the master can transmit while a slave is transmitting. However, most multi-drop protocols do not take advantage of this capability.

** RS-232, as supported by the RMC SERIAL, uses only three wires (RxD, TxD, and GND) if flow control is disabled, but five wires (RTS, CTS) if flow control is enabled.

*** The maximum cable lengths vary depending on the baud rate, termination (for RS-422/485), and capacitance of the cable. See RS-232 Wiring for the RMC SERIAL and RS-422/485 Wiring for the RMC SERIAL for details.

Note: From the RMC SERIAL perspective, RS-422 and 4-wire RS-485 are identical. Therefore, the RMC SERIAL offers three drivers: RS-232, RS-422/RS-485 (4-wire), and RS-485 (2-wire).

Each of the above features is described below:

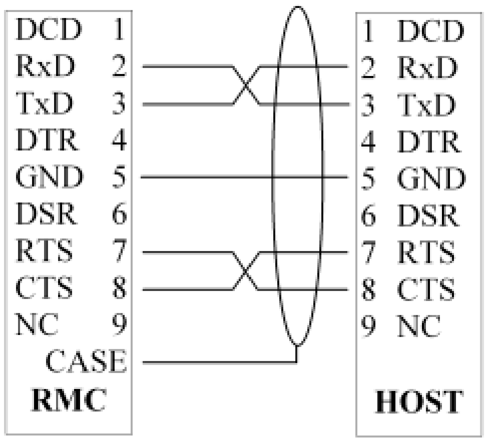
- **Duplex (Full or Half)**: Full-duplex means that each device on a serial network can send and receive at the same time, effectively doubling the bandwidth of the network. Half-duplex means that only one device on the network can send data at one time. For the above drivers, full-duplex requires separate send and receive wires.
- **Differential**: Differential wiring uses two wires per signal which allows common mode noise rejection. RS-232 does not use differential wiring, but instead has one wire per signal plus a ground. Differential wiring allows for longer cable distances and greater noise immunity.
- **Topology**: Topology describes the layout of the network. Point-to-point means that exactly two devices are wired together. Multi-drop means that two or more devices are chained together. Notice that "multi-drop" with only two devices becomes point-to-point. See Serial Network Topologies for details.
- **Wires**: This item refers to how many wires need to be connected between nodes. Notice that 2-wire and 4-wire RS-485 actually requires three and five wires respectively because of the Common in addition to the differential signal wires. See RS-232 Wiring for the RMC SERIAL and RS-422/485 Wiring for the RMC SERIAL for details.
- **Max Length**: The maximum cable lengths vary depending on the baud rate, termination (for RS-422/485), and capacitance of the cable. See RS-232 Wiring for the RMC SERIAL and RS-422/485 Wiring for the RMC SERIAL for details.
- **Flow Control**: Flow control can be used with RS-232 to ensure that one device does not overrun the other device. That is, if one device is sending data and the receiving device's buffers get full, then it can use flow control to pause the first device's sending until it has room in its buffers.

5.5.2.4 Serial Network Topologies

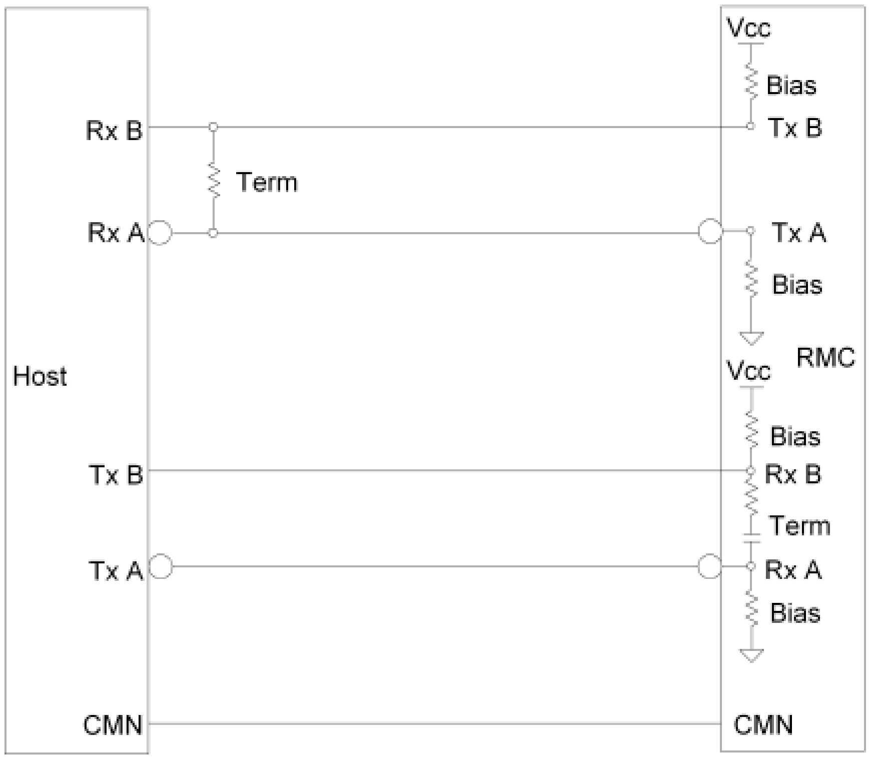
The RMC SERIAL supports two network topologies: point-to-point and multi-drop. Which topologies are available depend on the line driver (RS-232, RS-422, or RS-485) used. See Line Drivers: RS-232/422/485 for details on choosing the appropriate line driver.

Point-to-Point

Point-to-point means that exactly two devices are wired together. For the RMC, this means that there will be one RMC wired to one host. All three line drivers support point-to-point, as shown below:

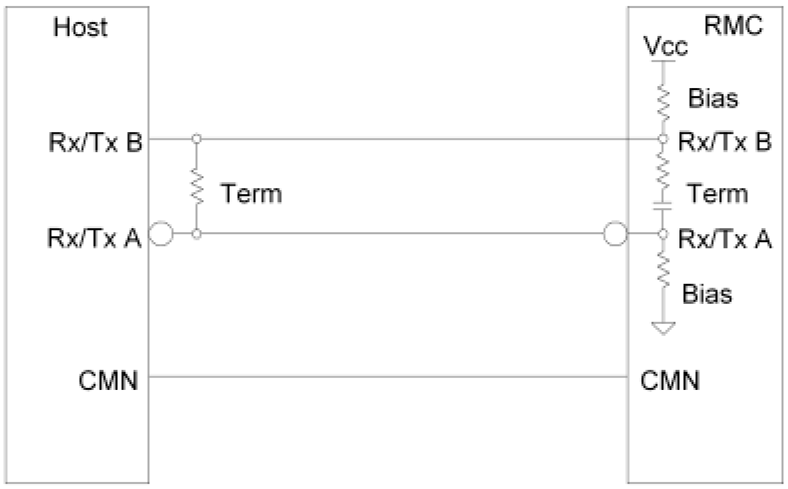


Point-to-Point RS-232 Network



Four-wire Point-to-Point RS-422/485 Network

Note: The above four-wire RS-422/485 diagram shows biasing internal to the RMC on the Tx wire pair. This is not available on RMC SERIAL hardware revision 1. Biasing will have to be provided externally or in the host.



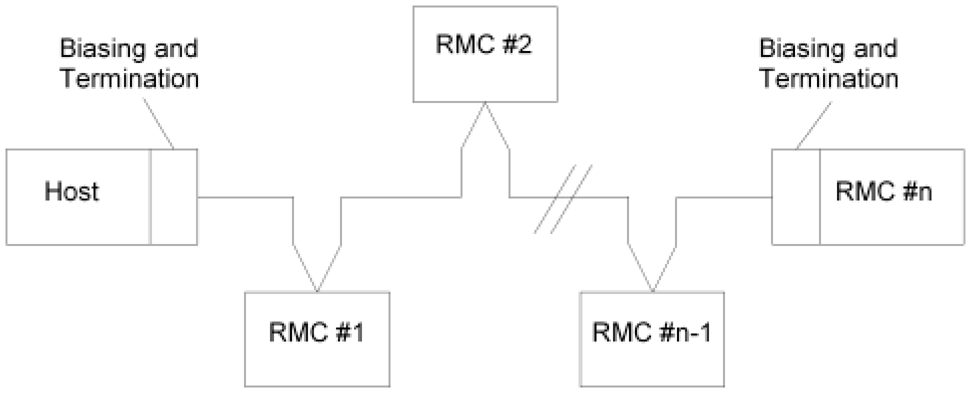
Two-wire Point-to-Point RS-485 Network

The RS-422 and RS-485 diagrams above show biasing and termination included. Termination and biasing can be left out of networks at the expense of maximum cable distance and noise immunity. See RS-422/485 Termination and Biasing for details.

Multi-Drop

Only RS-485 supports multi-drop. Multi-drop is the connecting of multiple slaves with a single master. Slaves should be chained together. Neither a star topology nor a chain with long stubs (wires from the main chain to the device) should be used. These topologies will cause excessive ringing on the network and unreliable data transmission.

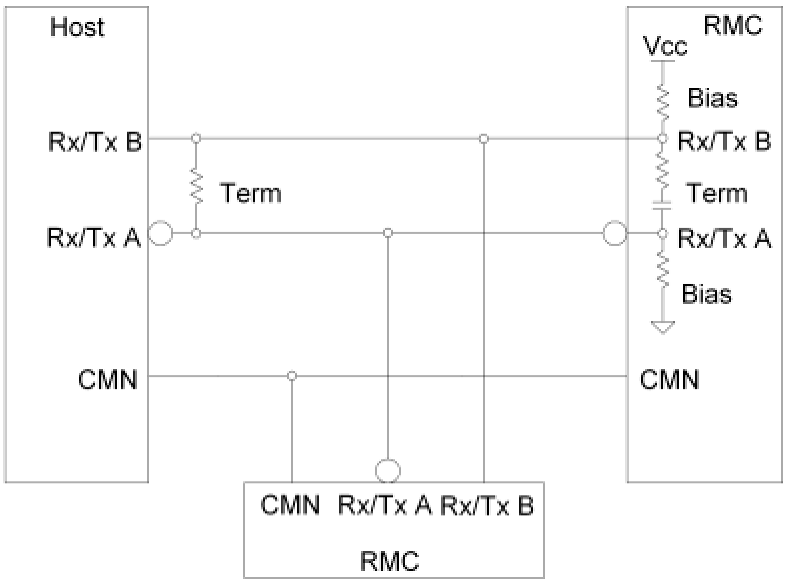
The following diagram shows a typical multi-drop chain. Notice that termination should only be located at the extreme ends of the network:



Daisy Chained RS-485 Network

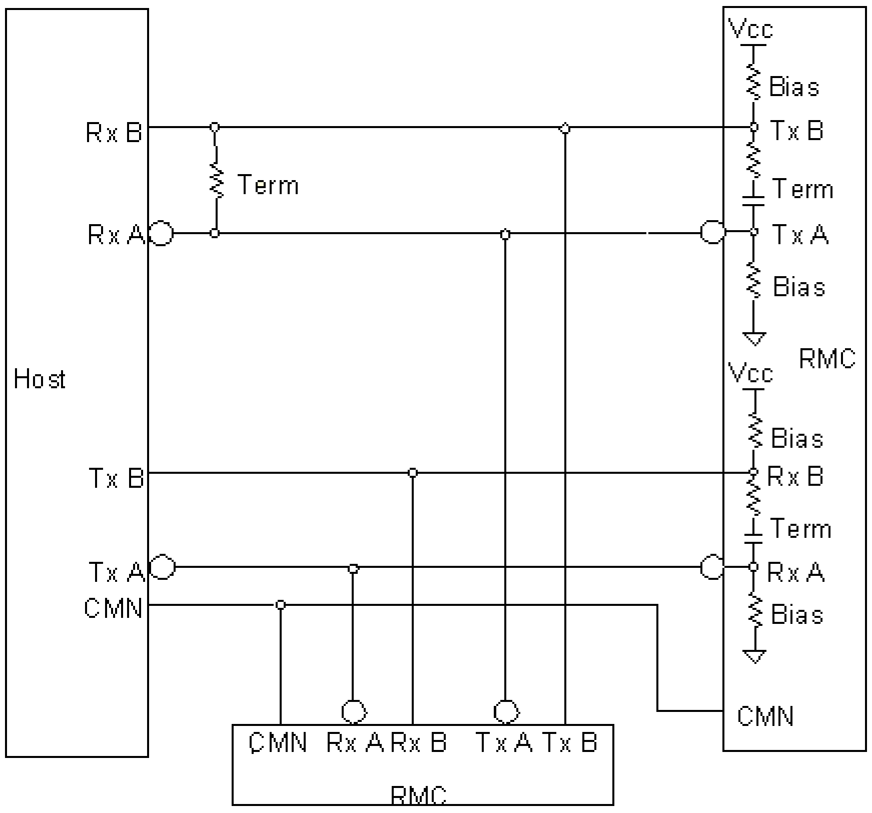
The number of devices that can be connected to the network is dictated by the number of unit loads that each represents. According to the TIA/EIA-485-A specification, there can be a maximum of 32 unit loads connected to a single network. Each RMC represents ¼ unit load for a total of 124 RMCs on the network, assuming the host is a unit load.

The following diagram shows one host with two RMC controllers in a daisy-chained two-wire RS-485 configuration:



Two-wire Multi-drop RS-485 Network

The following four-wire RS-485 network diagram is also supported by the RMC and allows full-duplex communications to the host from the RMC. Most multi-drop protocols do not support full-duplex communications between devices and so the actual utility must be carefully weighed against the extra cost of the cabling required for implementation.



Four-wire Multi-drop RS-485 Network

Note: The above 4-wire multi-drop RS-485 network diagram shows internal termination and biasing on the RMC for the Tx wire pair. This is not available on RMC SERIAL hardware revision 1. Termination will have to be provided externally. Biasing will have to be provided externally or in the host.

5.5.2.5 RS-232 Wiring for the RMC SERIAL

Connectors

The RMC SERIAL's 9-pin male DB connector is used for RS-232 communications. This requires a 9-pin female connector on the RMC SERIAL end of the connecting cable. The other end of the cable must match the master/host hardware, which may be a 9-pin, 25-pin, RJ-11, connector block, or other connector.

See also General Wiring Information.

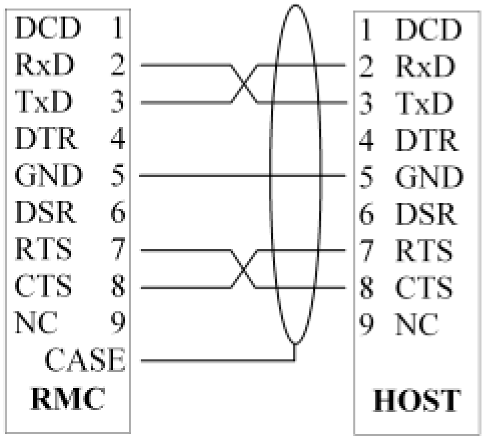
The following diagram shows the pin assignments on the RS-232 9-pin connector:

RMC Male DB9 Pin-out:

Pin #	**RS-232 Function**
1	DCD - Not used by RMC
2	RxD - Receive Data
3	TxD - Transmit Data
4	DTR - Not used by RMC
5	GND - Common
6	DSR - Not used by RMC
7	RTS - Ready To Send
8	CTS - Clear To Send
9	Not Connected

Cabling

A null-modem or crossover cable is typically used for RS-232 communications. See the following wiring diagram for details.



RS-232 Point-to-Point Wiring

Note: The RTS and CTS lines are only required if Hardware (RTS/CTS) flow control is selected in the RMC SERIAL configuration.

The RMC RS-232 communications require only three conductors in the cable: RxD, TxD, and GND. A five-conductor cable must be used if the CTS and RTS signals are used.

Delta recommends that a shielded cable be used to limit susceptibility to outside electrical interference.

Cable Length

One of the characteristics that limit the length of an RS-232 cable is capacitance. Most cables have a capacitance rating in pF/ft. The maximum distance that you can reliably transmit signals can be determined by answering a few questions and plugging the answers into the following formula:

MaxLength = 2400 pF / (C + (Shield * C))

Where: Shield = 2 for shielded cable and 0.5 for unshielded cable

C is the capacitance rating of the cable in pF/length

2400 pF is derived by taking the maximum capacitance specified by ANSI/TIA/EIA-232-F (2500 pF) and subtracting 100 pF for the input capacitance of the receiver

If you need to run your communications farther than allowed by this formula, then you must either use a cable with lower capacitance or use RS-422 or RS-485.

Example:

The cable that is being used is shielded and has a capacitance of 15 pF/ft.

MaxLength = 2400 pF / (15 pF/ft + (2 * 15 pF/ft))

= 2400 pF / (45 pF/ft)

= 53.3 ft

5.5.2.6 RS-422/485 Wiring for the RMC SERIAL

Connectors

Both RS-422 and RS-485 use the 6-pin connector block on the RMC SERIAL module. The pin-out is the same for RS-422 as 4-wire RS-485:

RS-422/485 (4-wire) Pin-out

Pin	RS-422/485 (4-wire) Function
1	Tx A (-)
2	Tx B (+)
3	Rx A (-)
4	Rx B (+)
5	Common
6	Case

Two-wire RS-485 uses the following pin-out:

RS-485 (2-wire) Pin-out

Pin	RS-485 (2-wire) Function
1	Unused
2	Unused
3	Rx/Tx A (-)
4	Rx/Tx B (+)
5	Common
6	Case

Note: Some manufacturers use A and B labeling, while others use + and - labeling. If you need to interface to equipment that uses an alternate labeling scheme, keep in mind that A corresponds to - and B corresponds to +.

See also General Wiring Information.

Cabling

All cabling for balanced or differential communications should consist of twisted pairs. Because the RMC's RS-422/485 interface is isolated, the signal common must be run alongside or in the cable. Therefore, for a two-wire network the cable must be either a one-pair cable with a separate ground line that is run externally or a two-pair cable in which one pair is used as the common. For a four-wire network this requirement changes to two-pair and three-pair. For a clean cabling solution, Delta recommends the option using an additional wire pair.

Another consideration when selecting communication cabling is the impedance of the cable. This impedance should match the termination resistance that is used. The RMC SERIAL module has a software selectable internal 120W termination. See RS-422/485 Termination and Biasing to determine whether or not your network will require termination.

No cable characteristics are specified in the TIA/EIA-422-B and TIA/EIA-485-A standards, but the RS-422-B standard does recommend 24AWG twisted pair cable with capacitance of 16 pF/ft and 100W characteristic impedance. These specifications will work well for RS-485 as well. One good choice would be to use Category 5 Ethernet cable. Category 5 Ethernet cable has a capacitance of 17 pF/ft max with 100W characteristic impedance. It is commonly available as shielded twisted pair (STP) or unshielded twisted pair (UTP). If this is not suitable then there are a number of manufacturers of communications cable such as Alpha and Beldon Wire and Cable.

Cable Length

The maximum cable length for RS-422 and RS-485 depends on the baud rate and termination. At higher baud rates, termination allows longer able lengths. For details on the effects of termination and how to apply it, see RS-422/485 Termination and Biasing.

The following chart shows the maximum cable length for RS-422 and RS-485 with and without termination:

Maximum RS-422/485 Cable Length:

Baud Rate	Max Unterminated Cable Length (ft)	Max Terminated Cable Length (ft)

115,200	475	3250
57,600	950	4000
38,400	1900	4000
19,200	3750	4000
9,600	4000	4000
4,800	4000	4000
2,400	4000	4000

5.5.2.7 RS-422/485 Termination and Biasing

Termination and Biasing are concepts that only apply to differential wiring. As such, they only apply to RS-422 and RS-485 and not RS-232. The Termination and Biasing concepts are described in detail below. First, however, we will describe the options provided by the RMC SERIAL module.

For RMC SERIAL modules with hardware revision 2 or later, biasing and termination can be independently selected for both wire pairs. For RMC SERIAL hardware revision 1, biasing and termination must be enabled and disabled together and are only available on the Tx/Rx wire pair and not on the Tx pair.

To change the termination and biasing options for the RMC SERIAL module:

1. On the Tools menu, click Module Configuration.
2. In the Slots list, click the Serial line.
3. Click Slot Options.
4. In the Serial Module Options dialog box, ensure that RS-422/RS-485 (4-wire) or RS-485 (2-wire) is selected.
5. Click Advanced.
6. Check or un-check the Enable Biasing Circuit and Enable Termination check boxes for the transmitter and receiver (for 4-wire) or transceiver (for 2-wire). The diagram in the dialog box will update accordingly.

Note: As described above, hardware revision 1 does not allow biasing and termination to be independently selected, and only allows enabling them on the Rx/Tx pair.

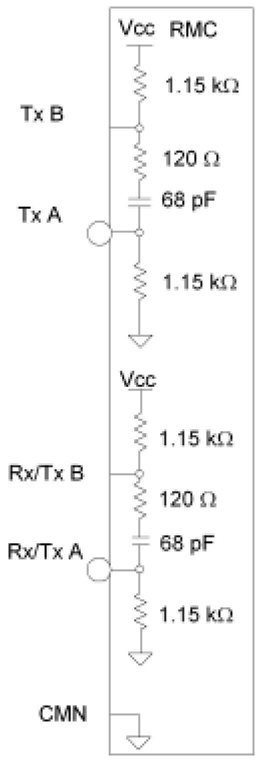
7. Click OK.
8. Click Update RMC.

   This step will save the settings to Flash memory.

   The Update Module Configuration dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module manually.

9. In the RMC Configuration dialog box, click Close.

The values of the resisters and capacitors in version 2 and later of the RMC SERIAL's termination/biasing circuit are shown in the diagram below:



RMC RS-422/485 Serial Termination Network

As described above, hardware revision 1 differs in its biasing and termination. It has the following differences:

- There is no biasing or termination on the Tx wire pair.
- There is no capacitor in the Rx/Tx termination circuit.
- The values of the resisters in the Rx/Tx biasing circuit are 2.2 kW instead of 1.15 kW.

Termination

Cable termination is a way of absorbing transmitted energy at the end of a network. This prevents signal reflections from bouncing back towards the transmitter and potentially upsetting signal quality and communications.

The termination resistor should match the characteristic impedance of the cable being terminated. The effective impedance of the RMC SERIAL's termination resistor and biasing resistors is 114W. Therefore, cabling with impedance of 100W to 120W is recommended.

Termination should be placed at the end of the network for each wire pair. The diagrams in Serial Network Topologies show the correct locations of the termination:

- For RS-422 (4-wire, point-to-point), terminate the receiver of each wire pair.
- For RS-485 (2-wire, point-to-point or multi-drop), terminate the wire pair at each end of the network.

- For RS-485 (4-wire, point-to-point or multi-drop), terminate the receivers of each end device in the chain, and the transmitter of the last slave (but not master).

Termination is not required on all differential networks, but it does typically extend the maximum cable length. The following chart shows the maximum cable lengths at various baud rates with and without termination:

Note: The maximum cable length is the length of the entire network and not just the distance between nodes on the network.

Termination vs. Cable Length:

Baud Rate	Max Unterminated Cable Length (ft)	Termination Requirements	Max Cable Length (ft)
115,200	475	Required beyond 475 ft	3250
57,600	950	Required beyond 950 ft	4000
38,400	1900	Required beyond 1900 ft	4000
19,200	3750	Required beyond 3750 ft	4000
9,600	4000	Not required	4000
4,800	4000	Not required	4000
2,400	4000	Not required	4000

Cable Length Derivation

The values presented in the chart above are based on 24AWG cable with capacitance of 16 pF/ft and the following reasoning. Signals travel through a cable at approximately 66% of c or 0.66 ft/ns. It is assumed that a signal transition will dampen out after three round trips in the cable. This damping must occur before the bit is sampled or within half a bit time. One bit time is equal to the reciprocal of the baud rate.

Example:

Compute the cable length for 115,200 baud RS422.

First, we compute a half bit time at this baud rate.

Half Bit Time	=	0.5 * 1 / 115200
	=	4,340 ns

Next, we convert this time to the distance the signal would travel in this time, assuming a speed of 0.66 ft/ns as described above:

Distance	=	4,340 ns * 0.66 ft / ns
	=	2890 ft

Since it requires three round trips for the signal transition to dampen and each round trip is twice the length of the cable, the total distance in feet is divided by six to get the final unterminated cable length:

Length	=	2890 ft / 6
	=	482 ft

This value is then rounded down to allow for inexact cable velocities and damping rates, giving us 475 ft.

Biasing

RS-422 and RS-485 indicate a binary 1 when the A line is at least 200 mV negative with respect to B, and a binary 0 when A is at least 200 mV positive with respect to B. It is important that the lines always be in a known state, not only when being driven. Biasing forces the network into a known state when the lines are idle and therefore otherwise not driven.

Hardware revisions 2 and later of the RMC SERIAL contain a 68 pF capacitor in series with the 120W termination resistor. This keeps the resister from loading the network when the network is idle, and maintains a known state without biasing. However, the software selectable biasing circuit can be used to bias the line when resistive termination is present on the other end of the network.

Hardware revision 1 of the RMC SERIAL module does not have the capacitor in its termination circuit and therefore will require biasing.

Biasing forces a valid state onto the network by allowing current to flow across the termination resistor.

Example:

This example assumes that there is a single master and two RMCs on the network. Compute the voltage across a 120W termination resistor when using 1150W biasing resistors.

First, we calculate how much DC resistance will be between the biasing resistors. Calculating the parallel resistance of all DC terminations and node input impedances does this. For a single master and two RMCs we have the following components:

- Master load: 1 unit load, which is defined as 12 kW.
- RMC loads: ¼ unit load each, which is 48 kW.
- Termination Resister in the RMC: 120W

Therefore, putting all of the resistances in parallel yields the following:

Termination	=	120W || 12kW ||

Resistance		48kW || 48kW
	=	118W

Then, we calculate how much DC resistance the network has between power rails:

Total Resistance	=	1150W + 118W + 1150W
	=	2418W

Next, we calculate how much current is flowing through this DC resistance:

Current	=	5VDC / 2418W
	=	2.068mA

Finally, we calculate the voltage drop across the termination resistor:

Voltage	=	2.068mA * 118W
	=	244mV

This value is greater than the ±200mV difference required by the TIA/EIA standards and constitutes a valid binary 0 state.

5.5.3 RMC SERIAL Protocols

5.5.3.1 Using Modbus/RTU with the RMC SERIAL

Modbus/RTU is a standard protocol managed by Schneider Automation (Modicon). The specification is available through Schneider Automation (http://www.modicon.com/openmbus) and Modbus.org (http://www.modbus.org). The Modicon Modbus Protocol Reference Guide (PI-MBUS-300) from Modicon is a good reference for this protocol. It is available through the Modicon web site. At the time of this writing it was necessary to select the **MODBUS/TCP Protocol** documents link to find an Acrobat Reader (.pdf) format version of this document.

Modbus, in its various forms such as Modbus/ASCII, Modbus/RTU, Modbus Plus, and Modbus/TCP, is a request/response protocol. That is, a Modbus master makes a request from a Modbus slave, and the slave responds. A number of functions are defined under Modbus. The following functions are supported by the RMC SERIAL module:

- 03: Read Holding Registers
- 06: Preset Single Register
- 16 (10 Hex): Preset Multiple Registers
- 23 (17 Hex): Read/Write 4X Registers

Each of the above functions acts on 4X or Holding registers. The RMC has these 4X registers mapped as described in the following topic:

- RMC Register Map (Modbus/TCP)

Protocol-Specific Settings

The RMC Modbus/RTU implementation has one protocol-specific setting: the node address. Enter this address in the **Serial Module Options** dialog box, as described in Configuring the RMC SERIAL.

5.5.3.2 Using DF1 (Full- and Half-Duplex) with the RMC SERIAL

Note: Full- and half-duplex DF1 was added to the RMC SERIAL module in 20010627 firmware. Modules with firmware dating prior to this do not support DF1.

The DF1 Protocol

Allen-Bradley’s DF1 Protocol and Command Set Reference Manual (pub. no. 1770-6.5.16) is the authority on the DF1 full- and half-duplex protocols. This manual is available on Allen-Bradley’s web site (http://www.ab.com). As of this writing, the following URL contains this document: http://www.ab.com/manuals/cn/17706516.pdf. When this link is out-of-date, try searching for the above publication number.

Full-duplex DF1 is used for peer-to-peer communication. Therefore, only two devices can communicate with one another. Only RS-232 or RS-422 support full-duplex.

Half-duplex DF1 is used for master-slave communication with one or more slaves. When more than two devices communicate with one another, 2-wire RS-485 is used. Otherwise, any line driver can be used.

RMC Support for DF1

Both full- and half-duplex DF1 use the same application protocol, which consists of commands and functions for the slave or peer to execute. The RMC supports the following DF1 functions:

- PLC5 Word Range Write (CMD=0x0F, FNC=0x00)
- PLC5 Word Range Read (CMD=0x0F, FNC=0x01)
- PLC5 Typed Write (CMD=0x0F, FNC=0x67)
- PLC5 Typed Read (CMD=0x0F, FNC=0x68)
- SLC Protected Typed Write with 2 Address Fields (CMD=0x0F, FNC=0xA9)
- SLC Protected Typed Read with 2 Address Fields (CMD=0x0F, FNC=0xA1)

- SLC Protected Typed Write with 3 Address Fields (CMD=0x0F, FNC=0xAA)
- SLC Protected Typed Read with 3 Address Fields (CMD=0x0F, FNC=0xA2)
- Echo (CMD=0x06, FNC=0x00)
- Diagnostic Status (CMD=0x06, FNC=0x03)

Most of the above functions address memory in the remote device. The RMC has memory registers defined (N7:0-255 and N9:0-N255:255) as described in the following topic:

- RMC Register Map (Allen-Bradley)

Protocol-Specific Settings

The RMC DF1 implementation has two protocol-specific settings:

- **Node Address**: Select a node address for the RMC that is unique to the DF1 network. This address will be entered in the master's request to identify the slave that should respond.
- **Error Checking**: Select one of the two error checking methods, BCC or CRC. Each provides error checking on the packet level. CRC is mathematically better at catching a lot of common types of errors, and as such is preferred. However, if the master does not support CRC then BCC must be used.

See Configuring the RMC SERIAL for details on using the **Serial Module Options** dialog box to change these and other RMC SERIAL options.

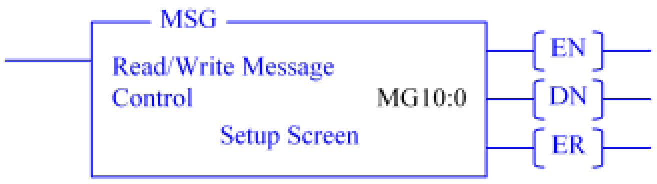
PLC Support for DF1

DF1 is a major industrial serial protocol supported by a large number of devices, both those built by Allen-Bradley and other companies. Any DF1 master implementation that uses the above blocks should also be able to read and write from the RMC.

Each Allen-Bradley PLC uses the Message (MSG) ladder logic block to initiate reads and writes over a serial port. For full details on this block, refer to Allen-Bradley’s Instruction Set Reference Manual for the appropriate PLC. While the same block is used by each PLC, the semantics differ slightly for each. Below are specific instructions for the PLC-5, SLC 5/0x, and MicroLogix. This documentation came using RSLogix 5 version 3.2.0.0 and RSLogix 500 version 4.10.00.

- **Allen-Bradley PLC-5**

The PLC-5 MSG block is displayed as follows:



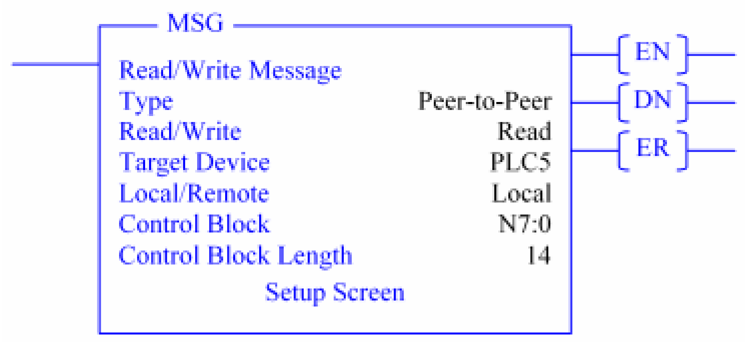
- **Control:** This parameter points a message (MG) file type element or a block of integer (N) file type elements. The number of N-file elements required varies from 11 to 15; look at the BLOCK SIZE field in the setup screen for the exact size. Set this to an unused block of registers, and then use the Setup Screen option in the MSG ladder logic block to modify those register values:
  - **This PLC-5:** This section holds parameters for the PLC-5.

- **Communication Command:** From this drop-down list, select **PLC-5 Typed Read** to read values from the RMC, or **PLC-5 Typed Write** to write values to the RMC.
- **Data Table Address:** Enter the address of the first Allen-Bradley PLC register to read RMC registers into, or to write to RMC registers from.
- **Size in Elements:** Enter the number of RMC registers to read or write in this field. Transfers are limited to 1000 bytes for PLC-5 Typed Reads and Writes. Therefore, this limit is 500 integers, 250 floats, etc. Notice that this limit is larger than the number of elements in the RMC's N-files. Reads or writes that extend beyond the end of a register file will continue into the next register file. For example, reading 300 elements from N9:0 will read N9:0 to N9:255, then N10:0 to N10:43.
- **Port Number:** Set this to the channel number of the serial port you want to use.

- **Target Device:** This section holds parameters for the target device.
  - **Data Table Address:** Enter the address of the first RMC register to read or write in this field. See the RMC Register Map (Allen-Bradley) for help on addresses.
  - **MultiHop:** This parameter should be set to No.
  - **Local Node Addr (dec):** Enter the node address of this RMC. The node address of the RMC is set up in the Serial Module Options dialog box, which is described in Configuring the RMC SERIAL. The node address entered on that screen is in decimal, so it is recommended that you enter the same number in the dec field.

- **Allen-Bradley SLC 5/0x**

The SLC 5/05 MSG block is displayed as the following:

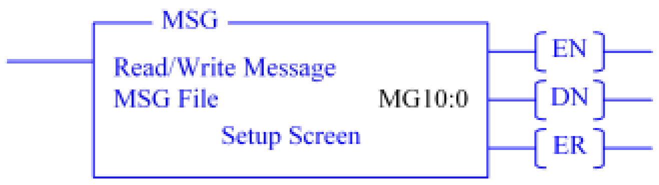

MSG
Read/Write Message
Type Peer-to-Peer
Read/Write Read
Target Device PLC5
Local/Remote Local
Control Block N7:0
Control Block Length 14
Setup Screen

- **Type:** This parameter is always set to Peer-to-Peer for serial communication channels.
- **Read/Write:** This parameter should be set to Read to read registers from the RMC, and to Write to write registers to the RMC.
- **Target Device:** This parameter has possible values of 500CPU, 485CIF, and PLC5. This should be set to PLC5 or 500CPU for communicating with the RMC.
- **Local/Remote:** This parameter has possible values of Local and Remote. This will be set to Local when the RMC is on the same serial network as the SLC.
- **Control Block:** This parameter points to a block of 12 integer-file registers. Set this to a block of registers, and then use the Setup Screen option in the MSG ladder logic block to modify those register values:

- **This Controller:** This section holds parameters for the SLC 5/05.
    - **Communication Command:** This parameter will be set to PLC5 Read, PLC5 Write, 500CPU Read, or 500CPU Write, depending on what was selected in the MSG block itself (as described above).
    - **Data Table Address:** Enter the address of the first Allen-Bradley PLC register to read RMC registers into, or to write to RMC registers from.
    - **Size in Elements:** Enter the number of RMC registers to read or write in this field. The range enforced by the SLC depends on the model used. The SLC 5/02 can transfer 1 to 41 integers, the SLC 5/03 and 5/04 can transfer 1 to 103 integers, and the SLC 5/05 can transfer 1 to 256 integers. Reads or writes that extend beyond the end of a register file will continue into the next register file. For example, reading 256 elements from N9:128 will read N9:128 to N9:255, then N10:0 to N10:127.
    - **Channel:** Set this to the channel number of the serial port you want to use. Keep in mind that on the SLC 5/05 channel #1 is the Ethernet channel.
- **Target Device:** This section holds parameters for the target device.
    - **Message Timeout:** Indicate the number of seconds to wait for the RMC to respond before determining that the attempt failed. This can be set as low as a few seconds.
    - **Data Table Address:** Enter the address of the first RMC register to read or write in this field. See the RMC Register Map (Allen-Bradley) for help on addresses.
    - **Local Node Addr (dec):** Enter the node address of this RMC. The node address of the RMC is set up in the Serial Module Options dialog box, which is described in Configuring the RMC SERIAL. The node address entered on that screen is in decimal, so it is recommended that you enter the same number in the dec field.
    - **Local/Remote and Bridge Parameters:** In most applications these will be set to Local and there will be no bridge parameters. If you are using a bridge then these parameters will need to be used. However, this is beyond the scope of RMC documentation. Refer to your Allen-Bradley documentation for instructions on using these fields.

- **Allen-Bradley MicroLogix**

The MicroLogix MSG block is displayed as follows:



To edit the parameters of the message block, select the MSG block, enter an unused MSG file in the **MSG File** parameter, and double-click **Setup Screen**. This brings up a dialog with the following options:

- **This Controller:** This section holds parameters for the MicroLogix.
    - **Communication Command:** This parameter can be set to PLC5 Read, PLC5 Write,

500CPU Read, or 500CPU Write. The type of PLC selected is not important, but the Read or Write determines whether you will read registers from the RMC or write registers into the RMC.

- **Data Table Address:** Enter the address of the first Allen-Bradley PLC register to read RMC registers into, or to write to RMC registers from.
- **Size in Elements:** Enter the number of RMC registers to read or write in this field. The MicroLogix can transfer 1 to 41 integers.
- **Channel:** Set this to the channel number of the serial port you want to use. The MicroLogix 1500 LRP Series B can use either channel 0 or 1, but all other MicroLogix PLCs will always use channel 0.

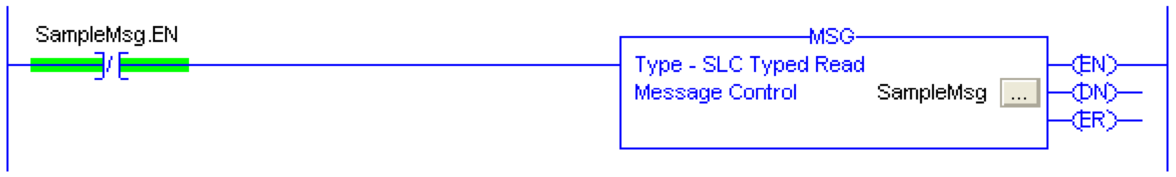
- **Target Device:** This section holds parameters for the target device.
  - **Message Timeout:** Indicate the number of seconds to wait for the RMC to respond before determining that the attempt failed. This can be set as low as a few seconds.
  - **Data Table Address:** Enter the address of the first RMC register to read or write in this field. See the RMC Register Map (Allen-Bradley) for help on addresses.
  - **Local Node Addr (dec):** Enter the node address of this RMC. The node address of the RMC is set up in the Serial Module Options dialog box, which is described in Configuring the RMC SERIAL. The node address entered on that screen is in decimal, so it is recommended that you enter the same number in the dec field.
  - **Local/Remote and Bridge Parameters:** In most applications these will be set to Local and there will be no bridge parameters. If you are using a bridge then these parameters will need to be used. However, this is beyond the scope of RMC documentation. Refer to your Allen-Bradley documentation for instructions on using these fields.

Using the MSG Block in Ladder Logic

The Allen-Bradley MSG block takes multiple ladder scans to complete. Therefore, it is important to enable the MSG block for the correct amount of time. Specifically, the MSG block must be energized until the message control's enable (EN) bit turns on. Delta has found some aspects of this to be difficult and therefore has provided the following ladder samples:

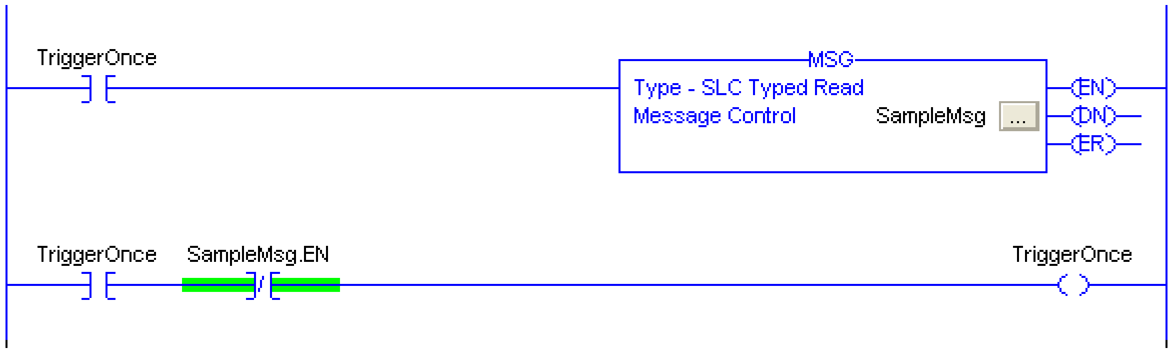
Read or Write Continuously

Using the Examine If Open instruction as shown below fulfills two requirements of continuous MSG transactions. First, it will keep the block energized until the EN turns on, and second, it de-energizes the MSG block once the transactions is started so that when the transaction is completed (EN goes low again), the MSG block sees a rising edge on its input, thus repeating the transaction:



Read or Write Once

This sample takes care to keep the MSG block energized until the MSG block starts, as indicated by the enable (EN) bit turning on. Once this happens, the application-controlled TriggerOnce coil is turned off. The message control's Done (DN) or Error (ER) bits can be used to process the results of the transaction.



5.5.3.3 Using the Mitsubishi No Protocol with the RMC SERIAL

Note: This Mitsubishi protocol was added to the RMC SERIAL module in 20010802 firmware. Modules with firmware dating prior to this do not support this protocol.

Introduction

This protocol is for serial communication with Mitsubishi FX controllers. For details on a serial protocol for Q series controllers, see the Using the Mitsubishi Bidirectional Protocol with the RMC SERIAL topic.

The Mitsubishi FX controllers support five types of serial communication. The one that matches the needs of the RMC is called "No Protocol." This name implies that the meaning of the data sent via this protocol is not predefined. The interpretation of this data for communication with an RMC will be called the Mitsubishi-RMC Protocol. Mitsubishi's No Protocol is implemented using the PLC's RS function. The No Protocol and the RS function are described in Mitsubishi’s FX Communication (RS-232C, RS-485) User’s Manual (manual number JY992D69901).

The Mitsubishi-RMC Protocol allows the Mitsubishi PLC to read and write binary data from an RMC over RS-232 or RS-485. Up to 96 registers can be transferred with just one RS function. However, it is recommended that large blocks of data be broken down into smaller packets when the RMC is active, because the RS function only allows one transfer to be in progress at a time, and thus the status information will not be transferred until other transfers are complete.

Configuring

Both the RMC and the Mitsubishi are very flexible and provide many options for configuring the serial ports. All of the settings below must be set as indicated, except the baud rate and parity, which can be changed as long as both the RMC and PLC use the same values. However, the values used give the fastest possible baud rate and highest level of error detection.

RMC SERIAL Settings:

Use the Serial Module Options dialog box as described in Configuring the RMC SERIAL to select the following options for the RMC SERIAL:

- Baud Rate: 19,200
- Data Bits: 8 bit
- Parity: Even

Mitsubishi PLC Settings:

The Mitsubishi PLCs use D8120 to control its communication format. Assuming the maximum baud rate of 19,200 and even parity are used, then the following value must be used to communicate with an RMC:

D8120 = H0097

In addition M8161 should be set as follows to select 16-bit data:

M8161 = OFF

D8120 can be broken down into the following options:

- Data Length: 8 bit (b0 = 1)
- Parity: Even (b1, b2 = 1, 1)
- Stop Bits: 1 (b3 = 0)
- Baud Rate (bps): 19,200 (b4-b7 = 1, 0, 0, 1)
- Header: None (b8 = 0)
- Terminator: None (b9 = 0)
- Control Line: None (b10-b12 = 0, 0, 0)
- Protocol: No protocol (b14 = 0)
- Bits 13 and 15 are reserved for No Protocol, and must be zero (0).

A demo program NOPROTO should be used as a starting point for any Mitsubishi FX program using a RMC. This program is posted on the web at http://www.deltacompsys.com/Files/rmcfx.zip.

Using the Mitsubishi-RMC Protocol

Communicating with the RMC from the Mitsubishi PLC requires two buffers in the PLC: one for sending data and one for receiving data. Each buffer should be 100 words long. This allows for data transfers of up to 96 words of data with a header consisting of a length and RMC address, and a trailing checksum using the CCD instruction that actually generates two words. The parity check result of the CCD function is the second word and is not used.

The RS function should be used to used as set the buffer addresses and the length of each transfer. The buffer address should never change from one transfer to the next but the length will change.

After the out going message is constructed it is sent using a RS function. This is enabled by setting bit M8122. At this point, the RS function will send data and wait to receive the requested number of bytes from the RMC. When both have occurred, bit M8123 will be set. At this time the

RMC should have returned its response as shown above. The checksum in the response can then be checked using the CCD function again.

The Mitsubishi-RMC Protocol defines the following three request/response packets. Each is composed of 16-bit fields, with each being sent low-byte first by the Mitsubishi. Send buffer locations will be referred to as Dxx00 to Dxx99, where xx are two digits, and receive buffer locations will be referred to as Dyy00 to Dyy99, where yy are two digits.

Read Data from the RMC

Request (to RMC):

Length (+)	Address	Checksum

Dxx00 Length. Gives the number of 16-bit registers to read from the RMC. This number will be positive, as differentiated from the Write Data request, which uses the negative of the length.

Dxx01 Address. Gives the address of the first register to read from the RMC. See RMC Register Map (PROFIBUS-DP) for the addresses of all RMC registers.

Dxx02 Checksum. The checksum is calculated using the PLC's CCD function. The length, address, and any data is included in the checksum. The CCD function calculates both a checksum and a parity check; only the checksum is used. Notice that the CCD function will also modify Dxx03.

Response (from RMC):

Length (+)	Address	Data0	Data1	…	DataN	Checksum

Dyy00 Length. Will match the request's length.

Dyy01 Address. Will match the request's address.

Dyy02 Data. First data word read.

Dyy03 Data. Second data word read.

…

Dyy01+N Data. Last data word read.

Dyy02+N Checksum. Validate with the CCD instruction.

Write Data to the RMC

Request (to RMC):

Length (-)	Address	Data0	Data1	…	DataN	Checksum

Dxx00 Length. Negative of the number of 16-bit registers to write to the RMC. This number is negative to differentiate it from the Read Data request, which uses the positive of the length.

Dxx01 Address. Gives the address of the first register to write in the RMC. See RMC Register Map (PROFIBUS-DP) for the addresses of all RMC registers.

Dxx02 Data. First data word to write.

Dxx03 Data. Second data word to write.

…

Dxx01+N Data. Last data word to write.

Dxx02+N Checksum. Set using the CCD instruction. Notice that Dxx03+N will also be modified by CCD with the parity, but this register should not be sent.

Response (from RMC):

Length (-)	Address	Checksum

Dyy00 Length. Will match the request's length.

Dyy01 Address. Will match the request's address.

Dyy02 Checksum. Validate with the CCD instruction.

Halt RMC

This function will send Disable Drive (K) commands to all axes in the RMC.

Request (to RMC):

0

Dxx00 Length. A length of zero indicates to the RMC to halt all axes with Disable Drive (K) commands.

Response (from RMC):

0	-6	Checksum

Dyy00 Length. This will be zero to indicate an error response.

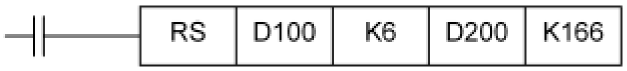
Dyy01 Error Number. This indicates that this error response was requested by the user. See Errors at the bottom of this topic for a list of other error responses.

Dyy02 Checksum. Validate with the CCD instruction.

Example RS Instructions:

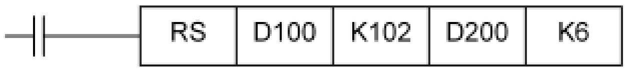
These examples assume the send buffer is at D100 and the receive buffer is at D200.

To read status for 8 axes or 80 words:

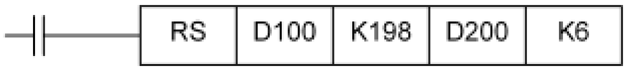


The lengths (K6 and K166) are in bytes. Remember to add the length, address and checksum to the length of the data. Therefore the send length of six bytes comes from the length, address, and checksum at two bytes each, and the receive length comes from 80 words (160 bytes) plus six bytes for the length, address, and checksum.

To write commands for 8 axes or 48 words:



To write 12 steps to the step table where each step is 8 words:



If more than 12 steps need to be transferred then more RS blocks are required to transfer the remaining steps.

Data Consistency

When writing to the RMC it is often important that all the data is written at once so the RMC gets the data in one scan. If the RMC uses partially updated data from the command area or the step table, unintended results may occur. It is also important that the status area be from the same RMC scan so that positions can be compared. The RMC guarantees data consistency over the following areas:

1. Data from Status reads from RMC's registers D0 to D79 will be from the same scan.
2. Data written to the RMC's command registers from D80 to D127 will be processed in the same scan.
3. Data read or written to the parameter area will be consistent if the transfer is 64 words or less. This means that initializing all 8 axes with 128 words of data will take two RMC scans. This should not be noticeable.
4. Data read or written to the step table is copied from the RMC's serial board to the step table in 100 word blocks. It is important that the RMC not execute a step that is partially updated. Therefore the PLC should either:

    a. Update the step table in blocks smaller than 100 words and in multiples of eight (8). Transfer sizes like 64, 80 or 96 work well.

    b. Stop the RMC from executing steps while downloading the step table.

    c. Use the CommTrig (C) link type so the RMC waits until the last step is updated.

Errors

When the RMC returns a length of zero, the value in the address field is an error number. The Error codes are as follows:

Value	Description
-1	**Time Out:** More than 100 milliseconds has gone by with out receiving a character.
-2	**Overrun Error:** The RMC was not able to read the character in the serial port before the next character over wrote it.
-3	**Framing Error:** The RMC's serial port received a incorrectly formed character. This is normally due to baud rate mismatch

	between the RMC and the PLC although it could be due to a corrupted character due to noise.
-4	**Parity Error:** This is caused by noise causing the RMC's serial port to incorrectly receive the character.
-5	**Break:** A break signal has been detected by the RMC's serial port. This is caused by the serial line being held low for an extended period of time. This will cause the RMC's state machine to start back at state 0.
-6	**Length of Zero:** The RMC received a length of zero. This is causes the RMC to stop all axis using a hard stop.
-7	**Invalid Checksum:** The RMC did not receive the correct check sum.
-8	**Invalid Length:** The RMC received a length greater than 96 or less than -96.

When the RMC returns only a 0 and error number the communications will stop because the PLC is expecting more characters to be returned. The PLC should have a communications time out timer that will restart the communications. It should also check the length for a return 0 and the error codes above. The error codes are useful for initial debugging because they give some indication as to why the communication is not working.

5.5.3.4 Using the Mitsubishi Bidirectional Protocol with the RMC SERIAL

Note: This Mitsubishi protocol was added to the RMC SERIAL module in 20051214 firmware. Modules with firmware dating prior to this do not support this protocol.

Introduction

Use the Bidirectional protocol to communicate with the RMC100 from a Mitsubishi Q series PLC with a QJ71C24 serial communication module. For details on communicating with an FX series PLC, see the sing the Using the Mitsubishi No Protocol with the RMC SERIAL topic.

Configuring

Both the RMC and the Mitsubishi are very flexible and provide many options for configuring the serial ports.

RMC SERIAL Settings:

Use the Serial Module Options dialog box as described in Configuring the RMC SERIAL to select the following options for the RMC SERIAL:

- Baud Rate
- Data Bits
- Parity

Mitsubishi PLC Settings:

Set the QJ71C24 intelligent function module switches for the desired serial settings. The serial settings on the Mitsubishi must match the settings on the RMC100. For details on the Mitsubishi serial settings, see section 4.5.2 of the Mitsubishi manual: Q Corresponding Serial Communication Module (User's Manual).

The sample GX Developer program RMCBIDIR should be used as a starting point for any Mitsubishi Q program using a RMC Serial module.

Sample Program

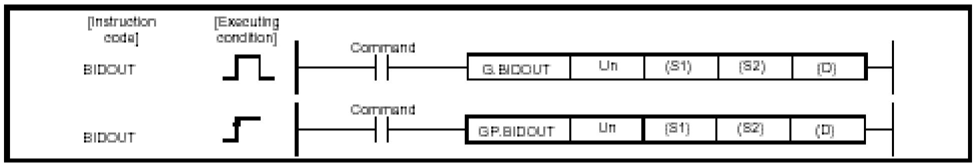
Delta has provided a sample GX Developer program (created with version 8.25B) called RMCBIDIR, available on the Downloads page of Delta's website at www.deltamotion.com. This program should be used as a starting point for any Mitsubishi Q program using a RMC Serial module.

Protocol Description

This protocol requires that the PLC program uses the BIDOUT and BIDIN instructions. A detailed explanation is found in the following Mitsubishi manual: Q Corresponding Serial Communication Module (User's Manual).

BIDOUT Instruction:

Set data	Applicable device								
	Internal device (System, user)		File register	MELSECNET/H, MELSECNET/10 Direct J□\□		Special module U□\G□	Index register Zn	Constant	Others
	Bit	Word		Bit	Word				
(S1)	—	○	○	—					
(S2)	—	○	○	—					
(D)	○	○	○	—					



Set data

Set data	Description	Set by	Data type
Un	Start I/O signal of the module (00 to FE: Top two-digit when I/O signals are expressed in 3-digit.)	User	BIN16 bit
(S1)	Head number of the devices in which control data is stored.	User, system	Device name
(S2)	Head number of the devices in which transmission data is stored.	User	Device name
(D)	Bit device number to be turned on at completion of execution.	System	Bit

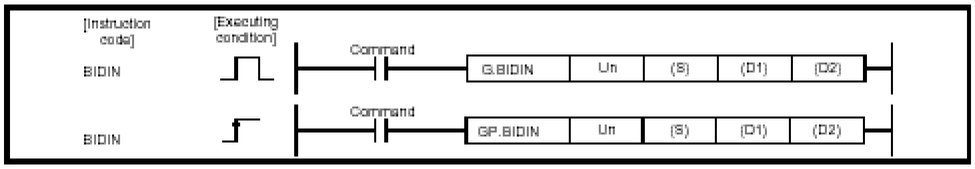
The file register per local device and program cannot be used as the set data.

Control data

Device	Item	Set data	Setting range	Set by *3
(S1) + 0	Transmission channel	• Set transmission channel 1: Channel 1 (CH1 side) 2: Channel 2 (CH2 side)	1, 2	User
(S1) + 1	Transmission result	• Transmission result by a BIDOUT instruction are stored. 0: Normal Other than 0: Error code *1	—	System
(S1) + 2	Transmission data count	• Set the transmission data count. *2	1 or more	User

BIDIN Instruction:

Set data	Applicable device								
	Internal device (System, user)		File register	MELSECNET/H, MELSECNET/10 Direct J□\□		Special module U□\G□	Index register Zn	Constant	Others
	Bit	Word		Bit	Word				
(S)	—	○		—					
(D1)	—	○		—					
(D2)	○	○		—					



Set data

Set data	Description	Set by	Data type
Un	Start I/O signal of the module (00 to FE: Top two-digit when I/O signals are expressed in 3-digit.)	User	BIN16 bit
(S)	Head number of the devices in which control data is stored.	User, system	Device name
(D1)	Head number of the devices in which receive data is stored.	System	Device name
(D2)	Bit device number to be turned on at completion of execution.	System	Bit

The file register per local device and program cannot be used as the set data.

Control data

Device	Item	Set data	Setting range	Set by [2]
(S) + 0	Receive channel	• Set receive channel 1: Channel 1 (CH1 side) 2: Channel 2 (CH2 side)	1, 2	User
(S) + 1	Reception result	• Reception result by a BIDIN instruction are stored. 0: Normal	—	System
(S) + 2	Receive data count	• The number of data received is stored. [1]	1 or more	System
(S) + 3	Allowable number of receive data	• Set the allowable number of words for received data that can be stored in (D1).	1 or more	User

Reading from the RMC100

To read data, first use the BIDOUT instruction to request the data, then use the BIDIN instruction to read the data.

The Head number of the data sent to the RMC100 is designated by (S2) in the BIDOUT instruction. The data must be according to the following format:

Each box is a 16-bit word.

Count	Address

Count: The number of registers to read.

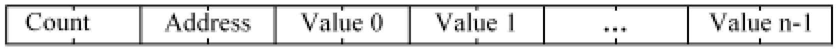
Address: The address of the first register to read. The address follows the PROFIBUS address format. See the PROFIBUS Register Map topic in the RMCWin help for details.

After receiving the request, the RMC100 will return the requested data. The BIDIN instruction is used to read the data. The X3 bit in the QJ71C24 will turn on when there is data available to be read from the QJ71C24 buffer. To read the returned data, (D1) in the BIDIN instruction indicates the head number of the device to store the data in.

Writing to the RMC100

The head number of the data sent to the RMC100 is designated by (S2) in the BIDOUT instruction. The data must be according to the following format:

Each box is a 16-bit word.



Count	Address	Value 0	Value 1	...	Value n-1

Count: The number of registers to write to. The count must be negated to indicate that it is a write. For example, to write to 6 registers, the count must be -6.

Address: The address of the first register to write to. The address follows the PROFIBUS address format. See the PROFIBUS Register Map topic in the RMCWin help for details.

Value x: The data values to be written to the RMC100.

5.6 RMC CPU RS232 Port

5.6.1 Using the CPU RS-232 Port with RMCWin

The RMC100 CPU serial port labeled "RS-232 Monitor" can be used to connect RMCWin to the RMC. There are several ways of connecting RMCWin running on your PC to the RMC's serial port. These methods are described in the following topics:

- Connecting RMCWin to an RMC
- Communication Driver: Serial Port Overview
- Communication Driver: TCP/IP-to-RS232 Bridge Overview

In addition to these topics, you may need to review the serial wiring for this port on the RMC:

- RS232 Wiring

5.6.2 RMCLink ActiveX Control and .NET Assembly

For communication from a PC to the RMC

Communicate with any RMC from a Custom Application

The RMCLink component enables direct communication with any of Delta Computer System's RMC family of motion controllers from numerous programming languages and applications. Supporting serial RS-232 and Ethernet communications, RMCLink provides full functionality to read and write registers, read bits, and issue commands to all RMC family controllers.

RMCLink comes with sample projects to help you get up and running quickly. The RMCLink help includes detailed walk-throughs and numerous code snippets.

RMCLink is available for free download on Delta's website: http://www.deltamotion.com/dloads/. The download includes a detailed help file and examples to get you started. After installing RMCLink, the help file will be accessible from the Windows Start button>>All Programs menu.

Supported Programming Languages and Applications

RMCLink can be used from numerous programming languages and applications. It has three interfaces to make it intuitive and easy to use from any language.

The table below lists supported programming languages and applications and the respective RMCLink interface that should be used for that language. All the interfaces are included in the RMCLink download.

Programming Languages and Applications	**RMCLink Interface**
Visual Basic 5.0/6.0 VBA (Microsoft Excel, Microsoft Word, etc.) VBScript JScript PHP	RMCLink COM Component
National Instruments LabVIEW	LabVIEW Virtual Instrument VIs (based on the RMCLink COM Component Interface)

Visual Basic .NET Visual C# Visual C++ with Managed Extensions (VS 2002/2003) Visual C++/CLI (VS 2005) Visual J#	RMCLink.Interop .NET Assembly
Visual C++ (Native Code)	RMCLink C++ Wrapper Class

Supported RMC Communication Ports

RMCLink can communicate via Ethernet or serial RS-232. The table below lists the ports on the RMCs that it can communicate with.

RMC Module	**RMCLink Supported Ports**
RMC100	ENET port on the RMC100-ENET module
	RS-232 Monitor port on the RMC100 CPU module
RMC150E	10/100 Enet port
RMC75E	10/100 Ethernet port
RMC75S	RS-232 Monitor port **Note:** If the serial settings on the second RMC75S RS-232 port are identical to the fixed RS-232 Monitor port settings, RMCLink can communicate with that port.
RMC75P	RS-232 Monitor port

Note:
The RMC100 is not supported by RMCTools and is not documented in this help file. It appears here only to fully explain RMCLink. Notice that RMCLink also is not a part of RMCTools, nor is fully documented in this help file. RMCLink contains its own very detailed help.

Using RMCLink

To use RMCLink, download it from Delta's website and install it. Open the RMCLink Documentation. You will find it on Windows Start menu > All Programs. Determine which Interface you need to use, based on your programming language. The **How to...** section is very helpful.

RMC Addresses in RMCLink

The RMCLink documentation includes the address maps you will need to use with RMCLink to access registers in the RMCs. Notice the addresses used in RMCLink may be different from any given in the RMC software.

5.6.3 RS232 Wiring

Using the CPU 9-pin Serial Port

The RMC100 CPU module has a DTE DB9 serial port labeled "RS-232 Monitor." Both RMCWin and the RMCLink ActiveX Control and .NET Assembly Component use this port. Communication over this port is always at 38400 baud with eight (8) data bits, no parity, and one (1) stop bit.

There are two versions of the serial port for the RMC product. The current style is a male DTE DB9. Early production RMCs have a female DTE DB9. However, because both connectors are DTE, the wiring is the same, so only a gender changer is needed to convert a cable between the two ports.

For details on using the serial port with RMCWin, see Using with RMCWin.

For details on using the serial port with the RMCCOM ActiveX control, see Using with the RMCCOM ActiveX Control.

See also General Wiring Information.

Note: The communication cable attached to the serial port is a potential source of electromagnetic radiation from the RMC. To minimize radiation, use a well-shielded cable that is as short as possible, and route it out the bottom of the module and against the back panel.

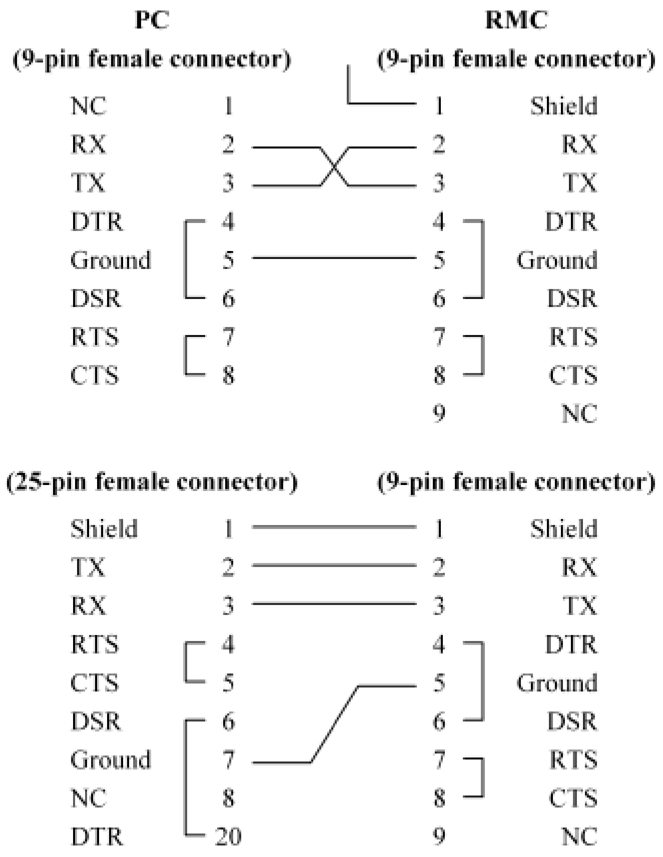
DB9 Pin Description

Pin	Function
2	Receive
3	Transmit
5	Common

Cable Recommendations for Male DB9 Connector

The following options are available for the cable between the PC and the RMC serial port:

- Generic null-modem (female on both ends) serial cable
- Allen-Bradley PLC serial cable
- Modicon Modbus cable with a 9-pin male-to-female gender changer
- Siemens SIMATIC 505 Programmable Controller serial cable
- You can use one of the following diagrams to build an RS232 cable:



- Any other cable you have will work if you can verify that pins 2, 3, and 5 on the RMC-end of the cable are connected as shown in the above diagram. An Ohmmeter or continuity checker will work for verifying the cable connections.

Cable Recommendations for Female DB9 Connector (older)

The recommendations for the older female DB9 connector are the same as for the male DB9 connector, but the gender on the RMC-end of the cable must be reversed:

- Generic null-modem (female on both ends) serial cable with a 9-pin female-to-male gender changer
- Allen-Bradley PLC Serial Cable with a 9-pin female-to-male gender changer
- Modicon Modbus cable
- Siemens SIMATIC 505 Programmable Controller Serial Cable with a 9-pin female-to-male gender changer
- You can build an RS232 cable using the diagram above, except the gender of the RMC-end connector should be reversed.
- Any other cable you have will work if you can verify that pins 2, 3, and 5 on the RMC-end of the cable are connected as shown in the above diagram. An Ohmmeter or continuity checker will work for verifying the cable connections.

5.7 LCD420 Terminal

5.7.1 LCD Display Terminal Overview

Using the LCD420 display as documented requires the following components:

- RMCWin 1.14.0 or newer
- RMC100 CPU with an RJ-11 port
- RMC CPU Firmware 20001204 or newer
- LCD420 purchased after December 4, 2000 (new units have an ESC key while the old units do not)

Prior to the December 4, 2000 release, the LCD420 display had a different keypad and the firmware behaved much differently. Users of the previous version may choose to keep their older displays, in which case it is important to not update the firmware to 20001204 or newer. Otherwise, contact Delta for details on upgrading to the new LCD420 display and firmware.

The LCD420 is an accessory available for the RMC100 series motion controllers. The LCD420 is a hand-held or panel-mounted terminal with a 4-row, 20-column LCD display and a 20-key keypad. This terminal can be programmed using RMCWin's LCD Screen Editor. The following list summarizes the LCD420 display capabilities:

- Build up to 16 custom screens.
- Include up to 4 fields on each screen.
- Write custom text on each screen.
- Include both read-only and editable fields.
- Include both integer and bit fields.
- Choose from status fields, parameters, step table entries, motion profile table entries, input-to-event table entries, plus several special fields.
- Choose where you want decimal points displayed for each value.
- Select custom limits for each editable field.
- Modify multiple RMC registers from a single editable field.
- Hot swap the LCD420.
- Power the LCD420 directly from the RMC.

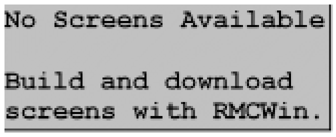
For details on using the LCD420 terminal, see Using the LCD420 Terminal.

For details on programming the LCD Terminal, see LCD Screen Editor: Overview.

5.7.2 Using the LCD420 Terminal

General

When the RMC powers up, it displays the first screen. If the RMC has no screens programmed, the following message will be displayed:



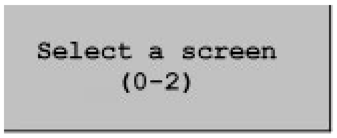
```
No Screens Available

Build and download
screens with RMCWin.
```

The screen is updated about 4 times per second. This allows updating fields continuously. This also allows hot-swapping LCD420 terminals, because when a display is plugged into the RMC it will be refreshed within 250 ms.

Selecting a Screen

If your set of screens has more than one screen, then you can use any of the following methods to switch to another screen:

Press	To
←(C)	Move to the previous screen. Pressing this key from the first screen will wrap around to the last screen.
→(D)	Move to the next screen. Pressing this key from the last screen will wrap around to the first screen.
MENU	Pressing the MENU button displays a message such as this:



```
 Select a screen
     (0-2)
```

From this prompt, press the number or letter of the screen you wish to display or press ESC to return to the screen you were last viewing.

Tip: Use the MENU feature when you want to document procedures for the LCD420 display. That is, it is easier to say "Press MENU and then 2" than to describe what screen the user should look for while scrolling through the screens with the arrow keys.

Selecting a Field

For screens that have editable fields, a blinking underline cursor will be located at the start of the selected field. Screens without editable fields will have no cursor. To select a different field, do the following:

Press	To
↑ (A)	Move to the previous field. Pressing this key when the first editable field is selected will wrap around to the last editable field.
↓ (B)	Move to the next selected. Pressing this key when the last editable field is

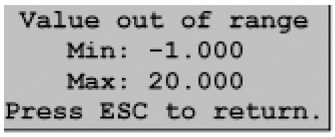
selected will wrap around to the first editable field.

Editing a Numerical Field

First, select the screen and field you wish to edit as described above. Then type in the value you wish to use for that field and press ENTER. While editing, the cursor moves to the last character in the field.

The following table summarizes the keys used for editing:

Press	To
0-9	Add another digit to the right end of the number.
+/-	Switch the sign of the number. This key can also be used to start an edit with a negative number. For example, to enter -4000, press +/-, 4, 0, 0, 0, ENTER.
BKSP	Retract the last number pressed. If all numbers are retracted, the edit will be cancelled.
ESC	Cancel the edit.
ENTER	Accept the edit. The value will first be compared with the limits. If the value is outside the limits, an error message like the following will be displayed:



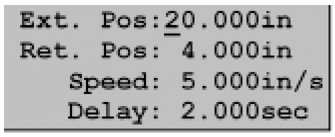
```
Value out of range
   Min: -1.000
   Max: 20.000
Press ESC to return.
```

If the value is within the limits, the value will be written to all write locations for the field. For details on having a value be sent to multiple locations, see Using Multiple Write Locations.

If the number contains a decimal point, enter the number without the decimal point, but make sure that you enter enough digits—possibly padding with zeros—so that your number is shifted correctly. The following example illustrates this.

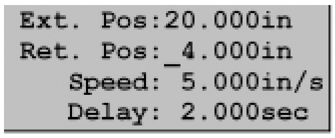
Example 1:

The user is looking at the following screen and wants to change the speed from 5.000 in/s to 4.800 in/s:



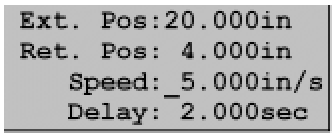
```
Ext. Pos:20.000in
Ret. Pos: 4.000in
   Speed: 5.000in/s
   Delay: 2.000sec
```

First, move the cursor the speed field. Press the down arrow (↓):



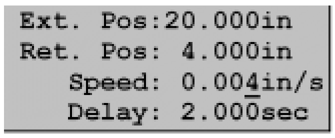
```
Ext. Pos:20.000in
Ret. Pos:_4.000in
   Speed: 5.000in/s
   Delay: 2.000sec
```

Press the down arrow (↓) again:



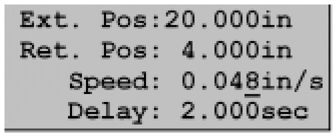
```
Ext. Pos:20.000in
Ret. Pos: 4.000in
   Speed:_5.000in/s
   Delay: 2.000sec
```

Next, enter the value of 4.8 in/s. Notice that the cursor moves to the end of the field during the edit. Press 4:



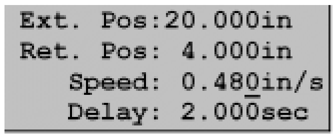
```
Ext. Pos:20.000in
Ret. Pos: 4.000in
   Speed: 0.004in/s
   Delay: 2.000sec
```

Press 8:



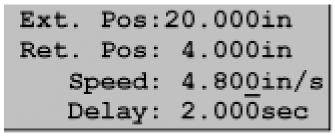
```
Ext. Pos:20.000in
Ret. Pos: 4.000in
   Speed: 0.048in/s
   Delay: 2.000sec
```

Notice that pressing ENTER at this point would enter a value of 0.048 in/s and not the desired 4.8 in/s, so you must pad the number of zeros. Press 0:



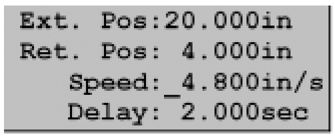
```
Ext. Pos:20.000in
Ret. Pos: 4.000in
   Speed: 0.480in/s
   Delay: 2.000sec
```

Press 0:



```
Ext. Pos:20.000in
Ret. Pos: 4.000in
   Speed: 4.800in/s
   Delay: 2.000sec
```

Now, press ENTER to accept the edit:



```
Ext. Pos:20.000in
Ret. Pos: 4.000in
   Speed:_4.800in/s
   Delay: 2.000sec
```

The cursor moves to the start of the field to indicate that the edit is complete.

Editing a Bit Field

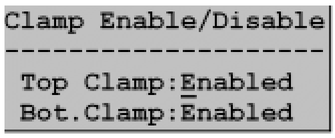
First, select the screen and field you wish to edit as described above. Then press 1 to turn the bit on or 0 to turn the bit off, and press ENTER. While editing, the cursor moves to the last character in the field. This indicates that the new value has not been accepted yet. ENTER must be pressed to have the new value take effect.

The following table summarizes the keys used for editing:

Press	To
0	Set the bit's value to OFF.
1	Set the bit's value to ON.
BKSP	Cancel the edit.
ESC	Cancel the edit.
ENTER	Accept the edit. The bit will be changed in the RMC.

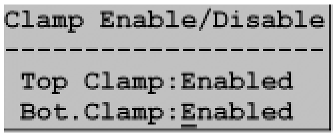
Example 2:

Suppose the user has commands in an event step sequence that turn on and off discrete outputs that control pneumatic clamps. By controlling the bits in the command value of the Set Outputs ([) command, the user can enable and disable using these clamps. Therefore, the user wants to use this screen to disable the bottom clamp:



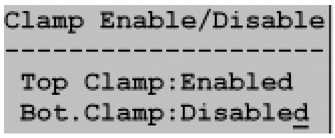
```
Clamp Enable/Disable
--------------------
 Top Clamp:Enabled
 Bot.Clamp:Enabled
```

The first step is to move the cursor to the bottom clamp field. Press the down arrow (↓):



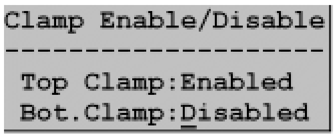
```
Clamp Enable/Disable
--------------------
 Top Clamp:Enabled
 Bot.Clamp:Enabled
```

Now that the correct field is selected, press 0 to turn this bit off:



```
Clamp Enable/Disable
--------------------
 Top Clamp:Enabled
 Bot.Clamp:Disabled
```

Notice that the cursor moved to the end of the field and the text changed to indicate the new setting. The change has not taken place yet. To make the change take place, press ENTER.



```
Clamp Enable/Disable
--------------------
 Top Clamp:Enabled
 Bot.Clamp:Disabled
```

The cursor moves to the start of the field to indicate that the edit is complete.

So, what about the FUNC key?

This key is reserved for future use. The current implementation does not allow initiating commands on the RMC. This key will be used for this purpose. If your application requires this functionality please contact Delta for further details on the availability of this feature.

5.7.3 Programming the LCD420 Terminal

The LCD420 terminal is programmed using RMCWin's LCD Screen Editor. This tool is fully documented in the Using RMCWin section of this help document. You can find that section through the following methods:

- In the table of contents, open the **Using RMCWin** book, then open the **LCD Screen Editor** book.
- Look up **LCD Screen Editor** in the index.
- Select any of the specific topics below:

LCD Screen Editor Overview

- Overview

Editor Window Elements

- Editor Window Elements
- Tree Pane Details
- Screen Pane Details
- Field Pane Details
- Data Tab Details
- Format Tab Details
- Toolbar Details
- Status Bar Details

Using the LCD Screen Editor

- Using LCD Screen Files
- Uploading and Downloading LCD Screens
- Using the Clipboard
- Changing the View Options
- Keyboard Shortcuts

Using Screens

- Adding and Removing Screens
- Changing the Screen Order
- Editing Screen Text
- Selecting Insert or Overtype Mode
- Renaming Screens

Using Fields

- Adding and Removing Fields
- Moving and Resizing Fields
- Editing Field Properties
- Using Editable Fields
- Using Fields with Multiple Write Locations
- Renaming Fields

5.8 Status Map

5.8.1 Using the Status Map Editor

Status Map Explained

Under the RMC's Modbus Plus and PROFIBUS-DP Message Mode interfaces, the RMC keeps 32 status registers readily accessible to network masters. For Modbus Plus, these registers are available through Modbus Plus Global Data. See Using Modbus Plus Global Data for a description of Modbus Plus Global Data. For PROFIBUS-DP in Message Mode, these registers come back as the first 32 input words of cyclic data, called the status block. See Using the PROFIBUS-DP Message Mode for details.

There are 32 entries in the Status Map, one for each global data or status block register. Each entry is the RMC register number to be mapped into global data. For a complete list of these registers, see RMC Register Map.

Changing the Status Map

Changes are made to the Status Map using the **Status Map Editor**. Refer to Table Editor Basics for topics common to all table editors. The default extension for saved Status Maps is **.map**.

Note: Prior to RMCWin version 1.8.4, this editor was called the Modbus Plus Global Data Map editor, and therefore the files were stored with the **.mbp** extension. You can open both **.mbp** and **.map** files with RMCWin version 1.8.5 or newer.

Editing a Cell

There are two ways to edit a cell:

- Use the Popup Editor. This is the preferred method of editing values in this table. To start the Popup Editor, either double-click on or press **Enter** in a cell. In the dialog that is displayed, select the piece of information you wish to have returned in the given status block register, and click **OK**.
- In the RMC Register cell for the Status Map Register you wish to reassign, type the address of the RMC register that you wish to have mapped into that global data or status block register. The only valid registers that can be used are the Status Registers (addresses 0-79), Step Table Registers (256-2303), and Digital I/O Registers (2632-2637). For a map of these registers, see RMC Register Map.

Expanding the RMC Register Column

While most users try to expand the RMC Register column by dragging to border of the window, this will not work. The reason for this is that changing the width of a spreadsheet window in RMCWin is used to change the number of columns displayed and not the width of columns. However, it is possible to change the width of the column and window by resizing the column itself.

1. Position the pointer over the right edge of the column header.

The cursor should change to a column resize cursor (✛), not to be confused with the window resize cursor.

2. Click and drag to resize the column.

When you release the mouse button, the column and window will be displayed at the new size.

For a list of the default global data register assignments, see Default Status Map Data.

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

5.8.2 Default Status Map Data

The following table lists the default mappings held in the Status Map table. For details on the Status Map, see Using the Status Map Editor.

Status Map Register:	**Register Value**	**Value Description:**
0	0	Address of axis 0 Command Position
1	1	Address of axis 0 Target Position
2	2	Address of axis 0 Actual Position
3	4	Address of axis 0 Status Word
4	10	Address of axis 1 Command Position
5	11	Address of axis 1 Target Position
6	12	Address of axis 1 Actual Position
7	14	Address of axis 1 Status Word
8	20	Address of axis 2 Command Position
9	21	Address of axis 2 Target Position
10	22	Address of axis 2 Actual Position
11	24	Address of axis 2 Status Word
12	30	Address of axis 3 Command Position
13	31	Address of axis 3 Target Position
14	32	Address of axis 3 Actual Position
15	34	Address of axis 3 Status Word

16	40	Address of axis 4 Command Position
17	41	Address of axis 4 Target Position
18	42	Address of axis 4 Actual Position
19	44	Address of axis 4 Status Word
20	50	Address of axis 5 Command Position
21	51	Address of axis 5 Target Position
22	52	Address of axis 5 Actual Position
23	54	Address of axis 5 Status Word
24	60	Address of axis 6 Command Position
25	61	Address of axis 6 Target Position
26	62	Address of axis 6 Actual Position
27	64	Address of axis 6 Status Word
28	70	Address of axis 7 Command Position
29	71	Address of axis 7 Target Position
30	72	Address of axis 7 Actual Position
31	74	Address of axis 7 Status Word

5.9 Communication Tasks

5.9.1 Reading Plots from the Communication Module

Overview

To load all plots on eight axes, the user would need to reserve 49,152 PLC registers. A single-axis full plot requires 6,144 PLC registers. A plot of a single piece of information on one axis requires 1,024 PLC registers.

The chart below can be used to determine the start of a given plot. n represents the axis number (0-7). Remember that there are 1024 registers per plot, so for non-Allen-Bradley addressing, the 1024 sequential registers hold each sample, but for Allen-Bradley addressing, only 256 registers fit in a register file. Therefore, the first 256 samples are in the register file calculated by the table below as elements 0-255. The next 256 samples are in the next register file, and so on.

Information **for Axis n**	**MB/TI505 Address**	**PROFIBUS Address**	**AB/SoftPLC Address**	**Siemens S7 Address**
Target Position†	16385 + 6144*n	16384 + 6144*n	N(64+24*n):0	DB(208+6*n).DBW0
Actual Position†	17409 + 6144*n	17408 + 6144*n	N(68+24*n):0	DB(209+6*n).DBW0
Status Word	18433 + 6144*n	18432 + 6144*n	N(72+24*n):0	DB(210+6*n).DBW0
Drive	19457 + 6144*n	19456 + 6144*n	N(76+24*n):0	DB(211+6*n).DBW0
Extra Plot Data #1†	20481 + 6144*n	20480 + 6144*n	N(80+24*n):0	DB(212+6*n).DBW0
Extra Plot Data #2†	21505 + 6144*n	21504 + 6144*n	N(84+24*n):0	DB(213+6*n).DBW0

†See the sections below for details on these arrays.

Position Units

The positions returned in these arrays are not in position units. The numbers returned are unsigned 16-bit numbers. To convert a ”r;position” returned in the array, perform the following math:

Position_real = (Position_plot x Sign) + Offset

In this equation, Sign will be +1 if the Scale is positive and -1 if the Scale is negative. For best results, this equation should be worked out using 32-bit integers or floating point math.

Extra Plot Data

Refer to the Selecting the Data to Plot topic for details on selecting the data to be entered into the last two plot arrays. The exact formats of the plot arrays are described below:

- **Extra Position Precision:**
  When reading extra precision graphs, Extra Plot Data #1 gives the fractional portion of the Target Position, and Extra Plot Data #2 gives the fractional portion of the Actual Position. To calculate the fraction, treat the fraction as an unsigned 16-bit number, and divide it by 65536. Then, add this number (which will be less than one) to the raw Target and Actual Positions returned in the first two plot arrays.

Note: Add the fractions before converting the raw Target and Actual to final Target and Actual Positions are described in the section above.

- **Command/Command Value:**
  Extra Plot Data #1 gives the Command, and Extra Plot Data #2 gives the Command Value.

- **Current Event/Link Value:**
  Extra Plot Data #1 gives the Event Step, and Extra Plot Data #2 gives the Link Value.

- **Raw Transducer Counts:**
  Extra Plot Data #1 gives the low 16-bits of the Raw Transducer Counts. Extra Plot Data #2 gives the high 16-bits of the same internal register.

- **Internal Speeds:**
  Extra Plot Data #1 holds the Target Speed, and Extra Plot Data #2 holds the Actual Speed. Both are in position units per second.

- **Integral Drive:**
  Extra Plot Data #1 holds the Integral Drive in drive count units. There are 8192 drive count units in 10000 mV. Therefore a value of 819 is equal to 1000 mV. Extra Plot Data #2 holds the fractional part of the Integral Drive, where this register is the numerator and the denominator is 65,536. The fractional part can usually be ignored except when determining the rate the integral is winding up or down.

Auxiliary Axes

For auxiliary (non-position analog) axes, the Drive, Extra Plot Data #1, and Extra Plot Data #2 tables are unused. In addition, the target and actual analog values are signed 16-bit values that match the values displayed in RMCWin. Therefore, no conversions need to be done on them as must be done with position units. The following chart shows the starting addresses of each auxiliary axis' 1024-element sample arrays:

MB/TI505 Address	PROFIBUS Address	AB/SoftPLC Address	Siemens S7 Address	Information
16385 + 6144*n	16384 + 6144*n	N(64+24*n):0	DB(208+6*n).DBW0	Axis n Target Analog
17409 + 6144*n	17408 + 6144*n	N(68+24*n):0	DB(209+6*n).DBW0	Axis n Actual Analog
18433 + 6144*n	18432 + 6144*n	N(72+24*n):0	DB(210+6*n).DBW0	Axis n Status Word

5.9.2 Downloading Splines to the RMC

Note: For a discussion on splines themselves and how the RMC can use them, see Spline Overview.

There are three ways to download splines to the RMC:

- **Use the RMCWin Curve Tool.**
  This method is useful in two cases. First, while experimenting with splines the curve tool is useful for drawing splines, sending them to the RMC, and executing them. Second, in applications where each axis has at most one spline, which never changes, the spline can be written using the curve tool and then saved to Flash memory. After this point, RMCWin is no longer needed to execute the spline because the spline is stored in the RMC.

- **Use the Communication Module and the Spline Download Area.**
  This method is ideal for most spline applications. An area of 4096 words of data can be written to in large blocks to efficiently send the RMC spline interval and point information. However, the Spline Download Area is only supported for the following communication modules: Modbus Plus, PROFIBUS-DP in Message Mode, and all Ethernet protocols. The Digital I/O and PROFIBUS-DP in Compact Mode do not support the Spline Download Area. This method is described in greater detail below.

- **Use the Communication Module and Repeated New Spline Point and Set Spline Interval/End Segment Commands.**
  This method is supported by all communication modules but requires issuing a new command (1) for every spline point, (2) for each time the interval between points changes, and (3) to indicate that the spline download is finished. This means that it does take a fair amount of time to download a spline using this method, but applications that only download the splines infrequently work well with this mode. However, if your communication card supports the Spline Download Area (see above), it should be used as it is easier to implement and faster to execute.

Spline Download Speeds

1. New Spline Point (X) and Set Spline Interval/End Segment (T) commands:

   When used to perform final calculations, this command may not finish immediately. This command will process up to four points immediately. If more points are in the segment, then an additional control loop will be required for every seven points in the segment. Therefore, a 50-point segment on a module with a one-millisecond control loop time will require eight (8) milliseconds to process. The Acknowledge bit of the Status word will toggle when this command is completed. For example, six 50-point splines (one spline per axis) would take 8 milliseconds total, assuming all start at the same time.

2. Spline Download Area (Ethernet or Serial):

   When writing to the Spline Download Area on RMC-ENET or RMC-SERIAL modules, the calculations are done by the communication module and therefore happen VERY quickly, significantly faster than the X/T commands. For short splines, it won't be much faster since the processors need to communicate with one another, but for longer splines, it'll be significantly faster.

3. Spline Download Area (PROFIBUS and Modbus Plus):

   The final calculations are done using the same algorithm as the X/T commands, and therefore they'll be done 4 points in the first loop and 7 per each subsequent loop. Notice that for the Spline Download Area, each axis's spline is not done at the same time.

Spline Download Area Details

Note: The Spline Download Area requires RMC100 CPU control firmware dated 19991124 or later (or beta firmware dated 19990910B or later). The Spline Download Area over Ethernet also requires Ethernet firmware dated 19990831 or later.

The Spline Download Area is a block of 4096 registers in the RMC Register Map. RMC Ethernet, RMC Modbus Plus, and RMC PROFIBUS-DP in Message Mode use the register map. Therefore, the Spline Download Area is not available with Digital I/O or the PROFIBUS-DP module in Compact Mode.

The assignment of these registers depends on the following:

- The number of axes that can have splines in the RMC. To determine the number of axes that can have splines in the RMC, start RMCWin and count all axis columns except auxiliary pressure and differential force channels. You can determine how an axis/channel is assigned in RMCWin by double-clicking on the axis name.
- The value of the first Spline Download Area register.
  - If the value is **0** or **1**, the Spline Download Area's 4096 registers are statically and evenly divided among the interval tables and point tables for each axis. This format is called **Static Spline Download Area format**.
  - If the value is **2**, the interval and point locations are dynamically sized and then positioned one after the other. The intervals for each spline are equal. This format makes the data in the Spline Download Area more compact and can significantly reduce the time it takes to download the splines to the RMC, at the expense of added complexity. This format is called **Dynamic Spline Download Area format**.

Note: The Static and Dynamic Spline Download Area formats apply to the entire table. Either the entire table is Static or it is dynamic.

Examples:

RMC100-M1-ENET

This module has two spline-capable axes, because MDT axes are never auxiliary pressure or force channels.

RMC100-Q2-A1-MB+

This module can have between 4 and 8 spline-capable axes. The quadrature axes are never auxiliary pressure or force channels, and therefore spline-capable. If the 12-bit analog module is configured as four auxiliary pressure channels, then none are spline capable, and therefore the RMC has a total of four spline-capable axes. If the 12-bit analog module is configured as four position- or velocity - reference channels, then all are spline capable, and therefore the RMC has a total of eight spline-capable axes.

Once you know the number of spline-capable axes, and whether you want to use the Static or Dynamic spline download area format, refer to the appropriate table below for the map of Spline Download Area register addresses:

Static Format with 2 Spline-Capable Axes

Register Description	Modbus and TI505	PROFIBUS	Allen-Bradley and SoftPLC	DL205/405	Siemens S7-300/400
Axis 0 Interval Table Format	12289	12288	N48:0	V10000	DB192.DBW0
Axis 0 Interval Table*	12290-13312	12289-13311	N48:1-N51:255	V10001-11777	DB192.DBW2-2046
Axis 1	13313	13312	N52:0	V12000	DB193.DBW0

Interval Table Format					
Axis 1 Interval Table*	13314-14336	13313-14335	N52:1-N55:255	V12001-13777	DB193.DBW2-2046
Axis 0 Point Count	14337	14336	N56:0	V14000	DB200.DBW0
Axis 0 Point Table**	14338-15360	14337-15359	N56:1-N59:255	V14001-15777	DB200.DBW2-2046
Axis 1 Point Count	15361	15360	N60:0	V16000	DB201.DBW0
Axis 1 Point Table**	15362-16384	15361-16383	N60:1-N63:255	V16001-17777	DB201.DBW2-2046

* The Interval Table contains 1022 intervals max.

** The Point Table contains 1023 points max.

Static Format with 3 or 4 Spline-Capable Axes

Register Description	Modbus and TI505	PROFIBUS	Allen-Bradley and SoftPLC	DL205/405	Siemens S7-300/400
Axis 0 Interval Table Format	12289	12288	N48:0	V10000	DB192.DBW0
Axis 0 Interval Table*	12290-12800	12289-12799	N48:1-N49:255	V10001-10777	DB192.DBW2-1022
Axis 1 Interval Table Format	12801	12800	N50:0	V11000	DB193.DBW0
Axis 1 Interval Table*	12802-13312	12801-13311	N50:1-N51:255	V11001-11777	DB193.DBW2-1022
Axis 2 Interval Table Format	13313	13312	N52:0	V12000	DB194.DBW0
Axis 2 Interval	13314-	13313-	N52:1-	V12001-	DB194.DBW2-

Table*	13824	13823	N53:255	12777	1022
Axis 3 Interval Table Format	13825	13824	N54:0	V13000	DB195.DBW0
Axis 3 Interval Table*	13826-14336	13825-14335	N54:1-N55:255	V13001-13777	DB195.DBW2-1022
Axis 0 Point Count	14337	14336	N56:0	V14000	DB200.DBW0
Axis 0 Point Table**	14338-14848	14337-14847	N56:1-N57:255	V14001-14777	DB200.DBW2-1022
Axis 1 Point Count	14849	14848	N58:0	V15000	DB201.DBW0
Axis 1 Point Table**	14850-15360	14849-15359	N58:1-N59:255	V15001-15777	DB201.DBW2-1022
Axis 2 Point Count	15361	15360	N60:0	V16000	DB202.DBW0
Axis 2 Point Table**	15362-15872	15361-15871	N60:1-N61:255	V16001-16777	DB202.DBW2-1022
Axis 3 Point Count	15873	15872	N62:0	V17000	DB203.DBW0
Axis 3 Point Table**	15874-16384	15873-16383	N62:1-N63:255	V17001-17777	DB203.DBW2-1022

* The Interval Table contains 510 intervals max.

** The Point Table contains 511 points max.

Static Format with 5 to 8 Spline-Capable Axes

Register Description	Modbus and TI505	PROFIBUS	Allen-Bradley and SoftPLC	DL205/405	Siemens S7-300/400
Axis 0 Interval Table Format	12289	12288	N48:0	V10000	DB192.DBW0
Axis 0 Interval Table*	12290-12544	12289-12543	N48:1-N48:255	V10001-10377	DB192.DBW2-510

Axis 1 Interval Table Format	12545	12544	N49:0	V10400	DB193.DBW0
Axis 1 Interval Table*	12546-12800	12545-12799	N49:1-N49:255	V10401-10777	DB193.DBW2-510
Axis 2 Interval Table Format	12801	12800	N50:0	V11000	DB194.DBW0
Axis 2 Interval Table*	12802-13056	12801-13055	N50:1-N50:255	V11001-11377	DB194.DBW2-510
Axis 3 Interval Table Format	13057	13056	N51:0	V11400	DB195.DBW0
Axis 3 Interval Table*	13058-13312	13057-13311	N51:1-N51:255	V11401-11777	DB195.DBW2-510
Axis 4 Interval Table Format	13313	13312	N52:0	V12000	DB196.DBW0
Axis 4 Interval Table*	13314-13568	13313-13567	N52:1-N52:255	V12001-12377	DB196.DBW2-510
Axis 5 Interval Table Format	13569	13568	N53:0	V12400	DB197.DBW0
Axis 5 Interval Table*	13570-13824	13569-13823	N53:1-N53:255	V12401-12777	DB197.DBW2-510
Axis 6 Interval Table Format	13825	13824	N54:0	V13000	DB198.DBW0
Axis 6 Interval Table*	13826-14080	13825-14079	N54:1-N54:255	V13001-13377	DB198.DBW2-510
Axis 7 Interval	14081	14080	N55:0	V13400	DB199.DBW0

Table Format					
Axis 7 Interval Table*	14082-14336	14081-14335	N55:1-N55:255	V13401-13777	DB199.DBW2-510
Axis 0 Point Count	14337	14336	N56:0	V14000	DB200.DBW0
Axis 0 Point Table**	14338-14592	14337-14591	N56:1-N56:255	V14001-14377	DB200.DBW2-510
Axis 1 Point Count	14593	14592	N57:0	V14400	DB201.DBW0
Axis 1 Point Table**	14594-14848	14593-14847	N57:1-N57:255	V14401-14777	DB201.DBW2-510
Axis 2 Point Count	14849	14848	N58:0	V15000	DB202.DBW0
Axis 2 Point Table**	14850-15104	14849-15103	N58:1-N58:255	V15001-15377	DB202.DBW2-510
Axis 3 Point Count	15105	15104	N59:0	V15400	DB203.DBW0
Axis 3 Point Table**	15106-15360	15105-15359	N59:1-N59:255	V15401-15777	DB203.DBW2-510
Axis 4 Point Count	15361	15360	N60:0	V16000	DB204.DBW0
Axis 4 Point Table**	15362-15616	15361-15615	N60:1-N60:255	V16001-16377	DB204.DBW2-510
Axis 5 Point Count	15617	15616	N61:0	V16400	DB205.DBW0
Axis 5 Point Table**	15618-15872	15617-15871	N61:1-N61:255	V16401-16777	DB205.DBW2-510
Axis 6 Point Count	15873	15872	N62:0	V17000	DB206.DBW0
Axis 6 Point Table**	15874-16128	15873-16127	N62:1-N62:255	V17001-17377	DB206.DBW2-510
Axis 7 Point Count	16129	16128	N63:0	V17400	DB207.DBW0
Axis 7 Point Table**	16130-16384	16129-16383	N63:1-N63:255	V17401-17777	DB207.DBW2-510

* The Interval Table contains 254 intervals max.

** The Point Table contains 255 points max.

Dynamic Format

The Interval Table contains 254 intervals max. The maximum number of points per axis is as follows:

1 - 2 spline capable axes: 1024

3 - 4 spline capable axes: 512

5 - 8 spline capable axes: 256

The Spline Download Area register map for Modbus, TI505, and PROFIBUS is given in the table below. For Allen Bradley, Soft PLC, DL205/405, and Siemens s7-300-400, add the Register # to the first Spline Download Area register:

	First Register
Allen Bradley and Soft PLC	N48:0
DL205/405	V10000
Siemens s7-300-400	DB192.DBW0

Register Description	**Register #**	**Modbus and TI505**	**PROFIBUS**
SDA Format - must be 2	0	12289	12288
Max Points per Axis* (MaxPts)	1	12290	12289
Axis 0 Interval	2	12291	12290
Axis 0 Actual Point Count	3	12292	12291
Axis 0 Point #0	4	12293	12292
Axis 0 Point #(MaxPts - 1)	2 + MaxPts +1	12291 + MaxPts +1	12290 + MaxPts +1
Axis 1 Interval	2 + MaxPts + 2	12291 + MaxPts + 2	12290 + MaxPts + 2
Axis 1 Actual Point Count	2 + MaxPts + 3	12291 + MaxPts + 3	12290 + MaxPts + 3

Axis 1 Point #0	2 + MaxPts + 4	12291 + MaxPts + 4	12290 + MaxPts + 4
Axis 1 Point #(MaxPts - 1)	2 + (MaxPts+2) +(MaxPts-1)	12291+ (MaxPts+2) +(MaxPts-1)	12290+ (MaxPts+2) +(MaxPts-1)
Axis 2 Interval	2 + (MaxPts+2)*2	12291+ (MaxPts+2)*2	12290+ (MaxPts+2)*2
Axis 3 Interval	2 + (MaxPts+2)*3	12291+ (MaxPts+2)*3	12290+ (MaxPts+2)*3
Axis 4 Interval	2 + (MaxPts+2)*4	12291+ (MaxPts+2)*4	12290+ (MaxPts+2)*4
Axis 5 Interval	2 + (MaxPts+2)*5	12291+ (MaxPts+2)*5	12290+ (MaxPts+2)*5
Axis 6 Interval	2 + (MaxPts+2)*6	12291+ (MaxPts+2)*6	12290+ (MaxPts+2)*6
Axis 7 Interval	2 + (MaxPts+2)*7	12291+ (MaxPts+2)*7	12290+ (MaxPts+2)*7

Interval Table Format

This single-register field defines the format of the Interval Table. The two formats defined are as follows:

Format	Interval Table
0	The interval between each spline point is equal. Only the first Interval Table entry is used. It is used for the intervals between all points. If the first Interval Table format register is 0, it specifies that Static Spline Download Area format is to be used. You must use the Static Spline Download Area Format Tables.
1	One Interval Table entry is used for each interval between points. If the first Interval Table format register is 0, it specifies that Static Spline Download Area format is to be used. You must use the Static Spline Download Area Format Tables.
2	The value 2 is only valid in the first register of the Spline Download Area. It specifies that Dynamic Spline Download Area format is to be used. You must use the Dynamic Spline Download Area Format table.

Interval Table

These registers are used to determine the distance between each point in the spline segment. The distance is in terms of milliseconds if the spline will be executed based on time, or counts the master moves if the spline will be geared to an axis or digital I/O counter. If the Interval Table Format is 0, then only the first register is used from this table, but it is used for the interval between every pair of points. If the Interval Table Format is 1, then as many intervals as the number of points that are in the spline minus one are used (since there are n-1 intervals between n points).

Note: The contents of this table are not used until the Point Table is filled in. Generally the Point Count and Point Table are downloaded after the Interval Table is downloaded. However, this table must be filled in prior to downloading values to the Point Table.

Point Count

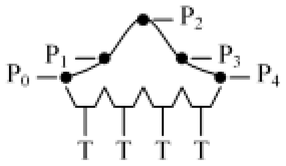
This register indicates the number of points in the spline.

Point Table

This table must be downloaded to last. When a number of points equal to the Point Count has been downloaded to this area, the spline segment is calculated and added to the given axis's spline table. If this operation is successful, the Acknowledge bit of the Status Word is toggled. If the operation fails, the Parameter Error bit of the Status Word is set and the Last Parameter Error register is set with the error number.

Example 1

In this spline segment, all points are equidistant along the X (time or geared) axis:

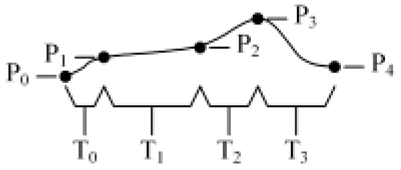


Assuming this segment is being downloaded to axis 0 on an RMC with just two spline-capable axes, here is the register assignment:

Address	Register	Value
12288	I.T. Format	0 (equal intervals)
12289	Interval Table – 0	T (interval between each point pair)
14336	Point Count	5
14337	Point Table – 0	P0
14338	Point Table – 1	P1
14339	Point Table – 2	P2
14340	Point Table – 3	P3
14341	Point Table – 4	P4

Example 2

In this spline segment, the distance along the X (time or geared) axis between points varies:



Assuming this segment is being downloaded to axis 0 on an RMC with just two spline-capable axes, here is the register assignment:

Address	Register	Value
12288	I.T. Format	1 (separate intervals)
12289	Interval Table – 0	T0 (distance between P0 and P1)
12290	Interval Table – 1	T1 (distance between P1 and P2)

12291	Interval Table – 2	T2 (distance between P2 and P3)
12292	Interval Table – 3	T3 (distance between P3 and P4)
14336	Point Count	5
14337	Point Table – 0	P0
14338	Point Table – 1	P1
14339	Point Table – 2	P2
14340	Point Table – 3	P3
14341	Point Table – 4	P4

Spline Download Procedure

1. Write to the Interval Table Format register to indicate the format of the Interval Table.
2. Write to the Interval Table. If the Interval Table Format register is zero (0), you will only need to write to the first register in the table. If the Interval Table Format is one (1), you will need to write to as many registers are there are points minus one (that is, for n points, there are n-1 intervals between them).
3. Write to the Point Count register. This will be the count of the points to be downloaded in the spline segment.
4. Write to the Point Table. When the last point in the spline (determined by the value of the Point Count register) is written, the spline segment will become part of the axis's spline table. When this operation is complete, the Acknowledge bit in the Status word for that axis will toggle.
5. If you wish to download another spline segment, only step 4 needs to be repeated. Any of steps 1, 2, and 3 can be repeated, but only need to be if their values are going to change. For example, you wish to change the number of points in the next segment, or you wish to change the intervals between points.

When writing the above registers, neither timing nor number of writes used are critical. For example, step 1 can precede step 2 by a few milliseconds or a few minutes. Or, every register can be written separately, or more commonly, steps 1 and 2 will be accomplished in a single write, and steps 3 and 4 will be accomplished in a second write. Of course, there may be cases where you are downloading more points than your PLC allows registers to be written at once. In these cases, it is perfectly acceptable to use multiple writes.

Important Notes

- The spline is actually added to the RMC axis's spline table when the last spline point has been downloaded.
- If there is an error while downloading the spline, a parameter error will be triggered. This means that the parameter error bit will be set in the Status word for the axis, and the Parameter Error value will be stored for retrieval by RMCWin or by reading the "r;Last Parameter Error" registers. See the RMC Register Map and Parameter Error Values topics for details on these registers.
- When the spline has been added to the RMC axis's spline table successfully, the Acknowledge bit in the Status word will be toggled.
- If there is already a spline segment in the axis's spline table, then the following rules will apply (these are the same rules that apply when downloading using X and T commands). If there is

more than one spline segment in the spline table, the new spline segment will be added to the end. If there are no spline segments in the spline table, the new spline segment will be added and made the next spline. If there is exactly one spline segment already in the spline table, then the new spline will be added after the existing spline segment if the existing spline segment is being currently followed or never has been followed; otherwise, the old spline segment will be deleted and the new spline segment will be added and made the next spline.

- You can simultaneously download splines to all spline-capable axes at one time.
- It is important that you avoid trying to download a spline from RMCWin's curve tool while downloading through the Spline Download Area.

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

5.9.3 Parameter Error Values

The Parameter Error bit in each axis's Status word is used to indicate a wide range of problems. This bit indicates that, either directly or indirectly, a command given to the RMC by the user cannot be executed as requested. However, it is not always obvious as to exactly what was wrong with the issued command. Therefore, the RMC has Last Parameter Error registers that hold an integer value for each axis. This integer value latches to the value of the last parameter error received. Therefore, it will not be cleared even if the Parameter Error bit itself clears.

RMCWin reads these registers when it detects the Parameter Error bit on an axis and it reports a text message corresponding to that error number. By using the Windows | Parameter Error List, the user can review the last parameter errors on each axis and receive error-specific help.

It is also possible to read the Last Parameter Error Registers from your PLC when using the RMC's Ethernet, Modbus Plus, or PROFIBUS (in Message Mode) communication modules. See the appropriate RMC Register Map topic for the addresses of these registers.

The purpose of this topic is to list the possible values you may receive from these registers, list a short description of each, and provide a link to a more detailed discussion on the problem. Click on the description text to jump to the topic with details on the error.

Value	Description
200	Target position moved outside limits
210	Flash contained no data on startup
220	Attempt to go beyond extend limit
221	Attempt to go beyond retract limit
222	Requested drive too large
223	Invalid command value
224	Invalid step number given in "Start Events" command
225	Invalid scale value
226	Extend limit must be greater than retract limit
227	Dead band eliminator out of range
228	Invalid command received
229	Non-existent pressure axis selected in "Config" word

230	Drive transfer percentage out of range
231	Feed forward terms must have the same sign
232	Resetting the position would cause a position overflow
233	Resetting the position is not allowed in this state
234	Axis must be initialized to use this command
235	Cannot overflow command pressure
236	Invalid MODE bits set for this command
237	Storage of parameters to Flash failed
238	Storage of splines to Flash failed
239	Steps per Rev and Position Units per Rev must not be zero
240	Reserved parameters must be zero
241	Maximum Steps per Millisecond parameter out of range
242	Invalid Address Used in Add (+) or Subtract (-) Command
243	Step Number in Teach (t) or Function (,) Command Out of Range
244	No Axes Selected for Use by the Function (,) Command
245	Function in the Function (,) Command Out of Range
246	The Accel Field Must Be Zero in the Command Issued
247	Invalid Screen Number in the Display LCD Screen ($) Command
250	Move would cause discontinuity
251	SSI transducer overflow
252	SSI transducer noise
253	Reserved command parameters must be 0
260	The acceleration or deceleration ramp is too slow
270	The command acceleration is invalid
271	The command deceleration is invalid
280	Both sync bits cannot be set in the "Mode" word
281	A synchronized axis is not initialized
282	Incompatible sync mode words
283	Synchronized axis was incorrectly dropped
284	Cannot issue a 'Z' or 'z' command to a synchronized axis
285	Cannot use synchronization with speed control
286	Cannot home an axis while synchronized
290	Gear ratio denominator is zero
291	Gearing and synchronization illegal in open loop
292	Invalid gear master selected
300	Pressure set A cannot be less than pressure set B
301	Command pressure cannot be less than pressure set A
302	Command pressure cannot be less than pressure set B

303	Axis reached command position while regulating pressure
304	No initialized pressure axis is assigned for monitoring pressure
305	Attempt to enter pressure immediately failed
306	Pressure Control went outside position limits
310	"Event Step Edit" indices are invalid
320	Too many spline points. Point not added.
321	Point cannot be added during calculation
322	The spline interval cannot be set below 5
323	There must be at least two points to begin calculations
324	Cannot clear a segment while interpolating
325	Fewer segments than were requested to be cleared existed
326	The axis must be stopped before following a spline
327	A valid segment has not been calculated
328	Target position must be equal to the first spline point
329	Velocity overflow while interpolating spline
330	Position overflow while interpolating spline
331	Acceleration overflow while calculating spline
332	Overflow while adding point. Point not added.
333	Unable to Download Curve over an Auto-Repeat Curve
334	Auto-Repeat Should Not be Used on Linear Axes with a Curve that Does Not Match Endpoints
340	Invalid command for this transducer type
341	Axis must be stopped for this command
350	Requested sine-move speed too low
351	Superimposed and geared mode bits required by the master-relative sine move command
360	Too many superimposed moves attempted
370	Numeric overflow while sending a spline to the Spline Download Area
371	Internal error while using the Spline Download Area
372	Attempt to send spline through Spline Download Area while download in progress
373	Attempt to write to the Spline Download Area of a non-existent or non-spline capable axis
374	Too many points attempted in the Spline Download Area
375	Spline Points downloaded out-of-order
376	Invalid Interval Table Format in the Spline Download Area
377	Invalid Point Count in the Spline Download Area
378	Unable to Download a Curve over an Auto-Repeat Curve Using Spline Download Area

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

6 Transducer Interface Modules

6.1 Analog

6.1.1 Analog Transducer Overview

There are four analog modules available for the RMC. Each is listed below:

Analog 16-bit with Pressure

- Four analog inputs with 16-bit Analog/Digital Converters.
- Two analog outputs with 12-bit Digital/Analog Converters assigned to analog channels 0 and 2.
- Able to utilize Pressure control firmware.
- Uses: position control, velocity control, speed reference (joystick), position reference (joystick), pressure control, differential force control, and pressure reference.

Analog 12-bit with Pressure

- Four analog inputs with 12-bit Analog/Digital Converters.
- No analog outputs.
- Able to utilize Pressure control firmware.
- Uses: speed reference (joystick), position reference (joystick), transitioned pressure control, transitioned differential force control, and pressure reference.

Analog 16-bit without Pressure

- Four analog inputs with 16-bit Analog/Digital Converters.
- Two analog outputs with 12-bit Digital/Analog Converters assigned to analog channels 0 and 2.
- Unable to utilize Pressure control firmware.
- Uses: position control, velocity control, speed reference (joystick), and position reference (joystick).

Analog 12-bit without Pressure

- Four analog inputs with 12-bit Analog/Digital Converters.
- No analog outputs.
- Unable to utilize Pressure control firmware.
- Uses: speed reference (joystick), and position reference (joystick).

Selecting the Roles of the Analog Channels

The first step for setting up analog module is to assign roles to each analog channel. This is described in Analog Transducer Configuration, it is possible to change the modules between some or all of the following roles:

- Using Analog Channels as Position Inputs
- Using Analog Channels as Velocity Inputs
- Using Analog Channels as Pressure Inputs
- Using Analog Channels as Differential Force Inputs

Selecting the Analog Transducer Type

Each of these analog modules can accept one of four voltage ranges or a current range from 4 to 20mA as input. Refer to Analog Transducer Configuration for details on selecting the appropriate type.

Analog Input Ranges

See the COUNTS topic for detailed information on the maximum and minimum limits of the various analog input ranges.

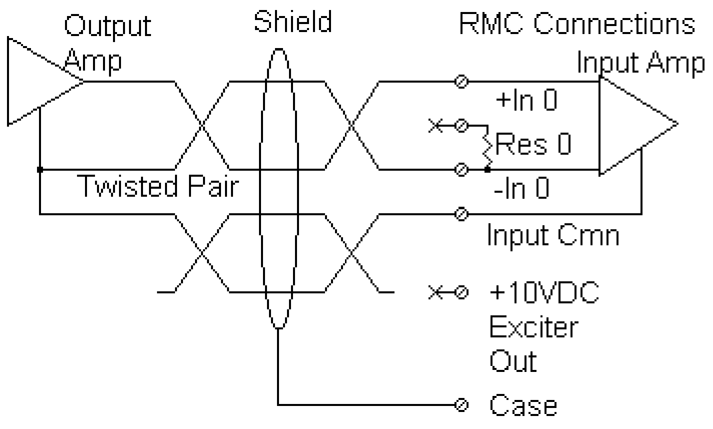
6.1.2 Analog Transducer Wiring

Overview

The analog modules for the RMC can read across voltage or current ranges. To switch between the four voltage ranges, only the Configuration word of the axis needs to be changed and saved to the Flash. To switch to the one current range, the same parameter needs to be changed and one external jumper needs to be attached. Read the appropriate section below for details on your particular transducer type.

Using Voltage Feedback Transducers

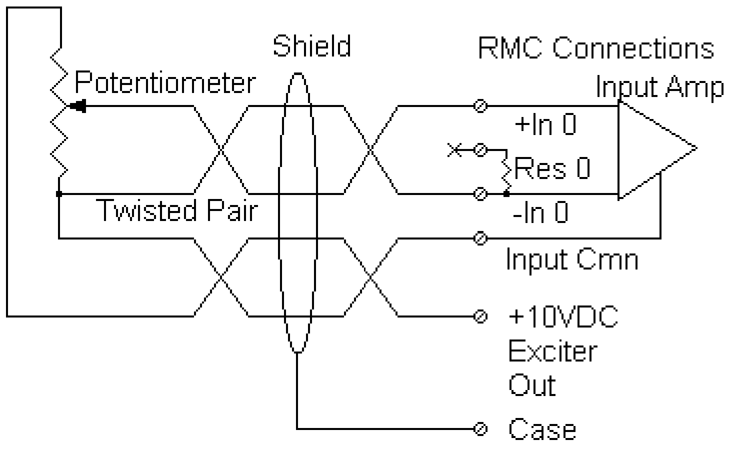
Voltage feedback transducers can be connected directly to the +In and -In connections for the desired channel. The Res connection is unused for voltage transducers. The following configuration is recommended:



TIP: If you are using a 10V transducer output, but are only using values in the low 5V, it is recommended that you select a 5V setting to increase the resolution of the analog to digital conversion.

Using the Exciter Output Pin with Potentiometer Feedback

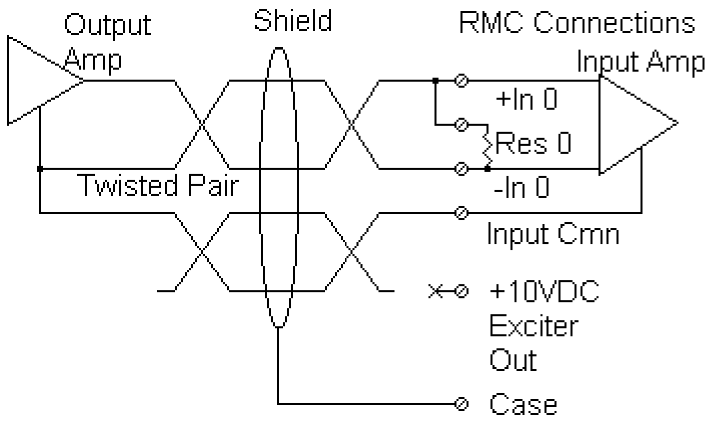
An exciter output pin is provided on the RMC analog modules as a convenience and also to increase the accuracy of the analog to digital conversion. This pin generates +10V with respect to the CMN pin. If a potentiometer type transducer is used, the following configuration is recommended:



Because the exciter voltage is the same voltage used as a comparison for the analog to digital conversion, the transducer is protected from being affected by increases in this positive voltage supply.

Using Current Feedback Transducers

Current feedback transducers are connected in the same way as voltage transducers except that a jumper must be inserted between the +In and Res connections. This places a resistor internal to the RMC across the two inputs, thus converting the current to a voltage input. The following wiring diagram shows a suggested configuration:



Analog Input Ranges

See the COUNTS topic for detailed information on the maximum and minimum limits of the various analog input ranges.

See also:

General Wiring Information

Analog Transducer Overview

Analog Transducer Configuration

Using Analog Channels as Position Inputs

Using Analog Channels as Velocity Inputs

Using Analog Channels as Pressure Inputs

Using Analog Channels as Differential Force Inputs

6.1.3 Analog Transducer Configuration

There are two parts to configuring an analog transducer:

1. Select the mode of operation of each analog channel
2. Select the analog transducer type (input Voltage or current range)

Each is described in this topic.

Select the Analog Channels Modes of Operation

The Analog Board Configuration dialog is used to select the mode of operation for each analog channel. To display this dialog:

1. On the Tools menu, click Module Configuration.
2. In the Slots list, click an analog module.
3. Click Slot options.

Many channel assignment options will be offered. Depending on the hardware and firmware present, some of the options may be disabled. See the descriptions below for a description of each operational mode and its possible uses.

To change the analog channel usage:

1. Click the Channels 0-1 tab.
2. Click the option button for your desired use for these channels.
3. Click the Channels 2-3 tab.
4. Click the option button for your desired use for these channels.

5. Click Update RMC.
6. The Update Module Configuration dialog box will be displayed to indicate the progress. If the RMC could not be reset automatically, you may be prompted to reset the RMC manually.
7. In the RMC Configuration dialog box, click Close.

Each channel-assignment option button gives the assignment of both channels. Some options assign the channel to a joint purpose, as in the case of differential force, some assign the channels to different purposes, and some assign the channels to the same purpose. Following are descriptions for each individual channel assignment:

- **Position Control**
  This mode uses a single analog channel and an analog drive output. Standard closed-loop position control may be performed on this axis.
- **Position Reference**
  This mode uses a single analog channel. This channel does not have an output and therefore cannot be used as in closed-loop control, but instead may provide a reference for another controlling axis. A common use for this type of axis is to serve as the master to another axis from a signal generated from a PLC or joystick.
- **Velocity Control**
  This mode uses a single analog channel and an analog drive output. The input channel value is assumed to indicate velocity, as given by a device such as a tachometer or velocity-control joystick. The drive output will work in conjunction with the input channel to control the speed of the axis.
- **Velocity Reference**
  This mode uses a single analog channel. The input channel value is assumed to indicate velocity, as given by a device such as a tachometer or velocity-control joystick. This channel does not have an output and therefore cannot be used as in closed-loop control, but instead may provide a reference for another controlling axis. A common use for this type of axis is to serve as the master to another axis from a speed signal generated from a PLC or velocity-control joystick.
- **Pressure Control**
  This mode uses a single analog channel and an analog drive output. The input channel is connected to a pressure transducer, the drive output is connected to the pressure-controlling valve or motor, and they work together to hold a pressure or ramp the pressure in closed-loop control.
- **Pressure Reference**
  This mode uses a single analog channel. The input channel is connected to a pressure transducer, but because it has no drive output, it cannot close the loop. Therefore, this input can only be used as a reference for another pressure control input. That is, a pressure control input/output pair can use a pressure reference input as its target pressure.
- **Auxiliary Pressure**
  This mode uses a single analog channel. This mode acts the same as Pressure Control mode described above, except that there is no drive output, so another axis' drive output must be used. The most common reason for doing so is to have a system with a position-to-pressure transition. In this case, a position control axis is assigned to the auxiliary pressure channel. The position transducer is connected to the position control axis' input, the pressure transducer is connected to the pressure channel, and each of the position and pressure inputs use the drive output when it is in control.

- **Differential Force Control**
  This mode uses both channels in the pair and an analog drive output. The input channels are connected to pressure transducers, the drive output is connected to the force-controlling valve or motor, and they work together to hold a force or ramp the force in closed-loop control.
  The first channel's input will be converted to a force using the Scale A and Offset A parameters, and will be displayed in the Actual Force A field. The second channel in the pair (channel 1 or 3 depending on the pair) will be converted to a force using Scale B and Offset B, and will be displayed in the Actual Force B field. Then, Force B will be subtracted from Force A to give the Actual Force, which is used for differential force control.

- **Auxiliary Differential Force**
  This mode uses both channels in the pair. This mode acts the same as Differential Force Control mode—described above—except that there is no drive output, so instead another axis' drive output must be used. The most common reason for doing so is to have a system with a position-to-force transition. In this case, a position control axis is assigned to the auxiliary differential force channel. The position transducer is connected to the position control axis' input, the pressure transducers are connected to the differential force channels, and each of the position and force inputs use the drive output when it is in control.

- **Auxiliary Analog**
  This mode uses a single analog channel. This mode is no longer supported.

- **Auxiliary Differential Analog**
  This mode uses both channels in the pair. This mode is no longer supported.

Note: At no time will the RMC100 allow more than eight axes. Any extra axes will be ignored. For example, suppose a system has one 2-axis MDT card and two 4-channel analog cards. If all eight channels are configured as single-ended analog axes, then this would total ten axes, which is not allowed. Therefore, the last two channels on the last analog card will not be usable.

Select the Analog Transducer Type

Use bits 12-15 of the Configuration word to select one of the following input ranges as described in that parameter's topic:

- Voltage: 0 to 10V
- Voltage: -10V to +10V
- Voltage: 0 to 5V
- Voltage: -5V to +5V
- Current: 4 to 20mA

See the COUNTS topic for detailed information on the maximum and minimum limits of the various analog input ranges.

See also:

Analog Transducer Overview

Analog Transducer Wiring

Using Analog Channels as Position Inputs

Using Analog Channels as Velocity Inputs

Using Analog Channels as Pressure Inputs

Using Analog Channels as Differential Force Inputs

6.1.4 Analog Transducer LED Indicators

Only the 16-bit analog modules have LED indicators. They correspond to the operational status of channels 0 and 2 when either is used in position mode.

LED Action	**Axis Status**
Off	Channel is not configured as position control (it is either a pressure or force axis)
Continuous Red	The following error has occurred: No Transducer
Alternating Red/Green	The above error has not occurred, and one or more of the below errors have occurred and are enabled in the Auto Stop: Transducer Noise Transducer Overflow Overdrive Error Parameter Error Position Overflow Integrator Windup Following Error
Continuous Green	Status good. None of the above are true.

Note: Prior to RMC100 CPU firmware dated 19991216, the Auto Stop parameter was not used in determining the LED states. Therefore, the only way to keep the LED from showing red was to clear the error.

Note: Prior to RMC100 CPU firmware dated 20020718, the Transducer Noise and Transducer Overflow errors were not maskable through the Auto Stop parameter, and therefore caused the LED to be solid red.

6.1.5 Analog Transducer Specifications

For general specifications on the RMC, see RMC100 Specifications.

16-bit Module (-H) Inputs

Inputs	Four 16-bit differential
Isolation	750VDC
Overvoltage Protection	40V
Input Ranges	+10V, ±10V, +5V, ±5V, and 4-20mA (each channel is independently configured using RMCWin)
Input impedance	1MW
Input filter slew rate	25V/ms
Conversion Rate	122ms on 1ms loop modules; 244ms on 2ms loop modules (8 times oversampling)
Offset drift with temperature	0.2 LSB/°C typical (+10V range)
Gain drift with temperature	20ppm/°C typical (+10V range)
Non-linearity	12 LSB (counts) typical (+10V range)
Exciter Output	10VDC ±2%, 8mA generated from A-D reference
Part # Designation	-Hx

12-bit Module (-A) Inputs

All specifications are the same as 16 bit except the following

Inputs	Four 12-bit differential
Offset Drift with Temperature	0.01 LSB/°C typical (+10V range)
Non-linearity	1 LSB (count) typical
Part # Designation	-Ax

Drive Outputs (on 16-bit Modules only)

Range	±10 V @ 5 mA (2 kW or greater load) (For current drive, use the VC2100 accessory: ±10 mA to ±200 mA in 10 mA steps)
Tolerance	At 10 V: +200 mV, -100 mV At 0 V: ±50 mV At -10 V: +100 mV, -200 mV
Resolution	12 bits
Output isolation	750 VDC, optically isolated
Overload protection	One-second short-circuit duration
Overvoltage protection	Outputs are protected by clamp diodes

6.1.6 Analog Transducer Scaling

Defining the Valid 16-bit Pressure/Force/Position/Velocity Range

For general scaling information, see the Scaling Overview topic.

Because the RMC uses 16-bit words for positions, pressures, forces, and speeds, these quantities must all fit within a range of 65,536 position units. Because the units used are user definable, this range does not limit most applications. See the section below on defining the units themselves.

The definition of the range of 65,536 position, pressure, force, or speed units depends on the use of the analog channel.

For pressure and force channels, the range is fixed to that of a signed 16-bit integer: 32768 to 32767.

For position or velocity axes, the Offset and Scale parameters define the range. If the Scale parameter is negative, then the position range extends from the Offset value minus 65535 up to the Offset value. If the Scale parameter is non-negative, then the position range extends from the Offset value up to the Offset value plus 65535. The following chart summarizes this concept:

Scale	Min. Position	Max. Position
< 0	Offset - 65535	Offset
³ 0	Offset	Offset + 65535

However, because the Offset is also used to convert transducer counts to position units, it cannot be set independently. For velocity control axes or velocity reference axes, the absolute position is typically not important, and the Offset can be set independently.

Translating to Pressure Units

For every scan the RMC makes, the Counts A status field is converted to a value for the Actual Pressure status field. This conversion uses the Scale A and Offset A parameters plus the Prescale Divisor bits of the Configuration word.

The following formula shows the conversion:

$$\text{Actual Pressure} = \frac{\text{Counts A} \times \text{Scale A}}{32768 \times \text{Prescale Divisor}} + \text{Offset A}$$

Translating to Force Units

For every scan the RMC makes, the Counts A and Counts B status fields are converted to the Actual Force A and Actual Force B, which are used to calculate Actual Force. These conversions use the Scale A, Scale B, Offset A, and Offset B parameters plus the Prescale Divisor bits of the Configuration word.

The following three formulas show these conversions:

$$\text{Actual Force A} = \frac{\text{Counts A} \times \text{Scale A}}{32768 \times \text{Prescale Divisor}} + \text{Offset A}$$

$$\text{Actual Force B} = \frac{\text{Counts B} \times \text{Scale B}}{32768 \times \text{Prescale Divisor}} + \text{Offset B}$$

$$\text{Actual Force} = \text{Actual Force A} - \text{Actual Force B}$$

Translating to Speed Units

The Scale, Offset, and the Prescale Divisor bits of the Configuration word parameters are used to define velocity units as a function of transducer counts.

The following formula summarizes the translation from transducer counts to Actual Speed units for velocity control or velocity reference axes:

$$\text{Actual Speed} = \frac{\text{Counts} \times \text{Scale}}{32768 \times \text{PreScale Divisor}}$$

The Actual Speed calculation does not use the Offset parameter. To add an offset for the Actual Speed, see the Set Count Offset Command topic.

The Offset parameter is used in calculating the Actual Position from the Actual Speed. However, the Actual Position for a velocity axis is typically not important, and for axes with an analog velocity input, it is typically not very accurate.

Translating to Position Units

The Scale, Offset, and the Prescale Divisor bits of the Configuration word parameters are used to define position units as a function of transducer counts.

The following formula summarizes the translation from transducer counts to Actual Position units:

$$\text{Actual Position} = \frac{\text{Counts} \times \text{Scale}}{32768 \times \text{PreScale Divisor}} + \text{Offset}$$

If the Actual Position overflows the valid 16-bit position range, as described above, the Position Overflow bit will turn on, forcing the axis to halt.

To calculate these parameters, you must physically measure the axis's position at two points and read how many counts the RMC reports at each position. If we call the two positions, in user position units, P0 and P1, and call the corresponding counts C0 and C1, the following two equations will give a Scale and Offset. Notice that the Prescale Divisor is left out at this point; it is assumed to be 1:

$$\text{Scale} = \frac{(P_0 - P_1)}{(C_0 - C_1)} \times 32768$$

$$\text{Offset} = P_0 - \frac{(P_0 - P_1) \times C_0}{(C_0 - C_1)}$$

Once the Scale is calculated, the Prescale Divisor can be calculated. The Prescale Divisor can have values of 1, 2, 4, or 8. Because the Scale is always divided by the Prescale Divisor, you essentially have a fractional scale. Pick the largest Prescale Divisor you can multiply the Scale by and still be between 32767 and 32767. For example, suppose your Scale comes to 6324.70. The following table shows the possible Scales and Prescale Divisors you could use and the effective scale:

Scale	Divisor	Effective Scale	Error from 6324.70
6325	1	6325/1 = 6325	0.005%
12649	2	12649/2 = 6324.5	0.003%
25299	4	25299/4 = 6324.75	0.0008%
50598	8	Invalid scale	Invalid

Therefore, in this example, a Scale of 25299 and a Prescale Divisor of 4 should be used.

These calculations can be done automatically using the Position Scale/Offset Calibration Utility feature in RMCWin.

Analog Input Ranges
See the COUNTS topic for detailed information on the maximum and minimum limits of the various analog input ranges.

6.1.7 Setup Details

6.1.7.1 Using Analog Channels as Position Inputs

Analog channels may be configured to be used as one of two position input types:

- Position Control - The input is used with the corresponding drive output (analog modules without drive outputs cannot be used for position control) for closed loop control. See Controlling Solely Position, Pressure, or Force for details.
- Position Reference - The input is used without the drive output. The incoming counts are scaled to position units, but otherwise the axis does no control. This is most often used for joysticks, in which case another axis will be slaved to the movements of the joystick on a position reference axis. See Using Position and Velocity Joysticks or Using an External Target Generator for details.

For both position inputs, all fields are used as described in the position status, command, and parameter field topics. One field worthy of note is the Transducer Counts field. See the section What to Expect in the Transducer Counts Field below for details.

When using Position Reference, the only parameters that are used are Configuration word, Scale, Offset, Extend Limit, and Retract Limit. All others are ignored.

Use the following steps to configure an analog position input:

Step 1: Assign Analog Channel(s) to be Position Input(s)

Because only channels 0 and 2 have drive outputs associated with them on the analog modules with drive outputs, it is only these two channels that can be configured as Position Control; any channel may be configured as Position Reference. Use the following steps:

1. On the Tools menu, click Module Configuration.
2. In the Slots list, click an analog module.
3. Click Slot options.
4. In the Analog Channel Assignment dialog box, click the tab of the channel pair (0-1 or 2-3) you wish to reassign.
5. Click the option button of a channel assignment that designates one or more channels as Position control and/or Position Reference. The assignment selected depends on how you intend to use the all channels.
6. Click Update RMC.
7. The Update Module Configuration dialog box will be displayed to indicate the progress. If the RMC could not be reset automatically, you may be prompted to reset it manually.
8. In the RMC Configuration dialog box, click Close.

Step 2: Configure the Analog Transducer Type for the Position Input

Refer to Configuring the Analog Transducer Type for details on selecting between a voltage or current input type.

Step 3: Set the Scale, Offset, Extend Limit, and Retract Limit Parameters for the Position Input

Refer to the individual field sections for details on setting each parameter.

What to Expect in the Transducer Counts Field

The Counts reading will vary depending on the transducer type selected. Refer to that topic for a description of the mapping of input voltage or current to counts.

Analog Input Ranges

See the COUNTS topic for detailed information on the maximum and minimum limits of the various analog input ranges.

See also:

Analog Transducer Overview

Analog Transducer Wiring

Analog Transducer Configuration

Using Analog Channels as Velocity Inputs

Using Analog Channels as Pressure Inputs

Using Analog Channels as Differential Force Inputs

6.1.7.2 Using Analog Channels as Velocity Inputs

Analog channels may be configured to be used as one of two velocity input types:

- **Velocity Control**
  The input is used with the corresponding drive output (analog modules without drive outputs cannot be used for velocity control) for closed loop control. As described below, the position for the axis changes at a speed determined by the transducer input. See Controlling Speed from a Tachometer Feedback for details.

- **Velocity Reference**
  The input is used without the drive output. As described below, the position for the axis changes at a speed determined by the transducer input. Because no drive output is used, the axis cannot do closed-loop control. This is most often used for joysticks, in which case another axis will be geared to the movements of the joystick on a velocity reference axis. See Using Position and Velocity Joysticks or Using an External Target Generator for details.

For the most part, all fields are used as described in the position status, command, and parameter field topics. However, the following fields are used differently:

- Transducer Counts. The counts are signed, as they are for pressure and force. See Counts for a mapping.

- Actual Position. The position is not derived directly from the Transducer Counts as it is with all other operation modes. Instead, the position is changed at the velocity given in the Actual Speed field.

- Actual Speed. The actual speed is derived directly from the Transducer Counts. To convert the Transducer Counts to the Actual Speed, do the following:

Actual Speed = (COUNTS - 325) x SCALE / (32768 x Prescale Divisor)

The 325 subtracted from the counts are a 1% dead-band. Therefore, if the input generates between -325 and +325 counts, the axis will have 0 velocity. Notice that this value is 1% of the full input value (e.g. 10V, 5V, 20mA), which will be represented by 32500 counts, as described in the Counts topic.

- Scale. This parameter is used to compute actual speed (as described above). A scale of 0 is treated like 32,768.

- Offset. This parameter sets the usable range of positions, as with standard position axes. It has no effect on the mapping of the input voltage/current and the speed generated.

When using Velocity Reference, the only parameters that are used are Configuration, Scale, Offset, Extend Limit, and Retract Limit. All others are ignored.

Use the following steps to configure an analog velocity input:

Step 1: Assign Analog Channel(s) to be Velocity Input(s)
Because only channels 0 and 2 have drive outputs associated with them on the analog modules

with drive outputs, it is only these two channels that can be configured as Velocity Control; any channel may be configured as Velocity Reference. Use the following steps:

1. On the Tools menu, click Module Configuration.
2. In the Slots list, click an analog module.
3. Click Slot options.
4. In the Analog Channel Assignment dialog box, click the tab of the channel pair (0-1 or 2-3) you wish to reassign.
5. Click the option button of a channel assignment that designates one or more channels as Velocity Control and/or Velocity Reference. The assignment selected depends on how you intend to use the channels.
6. Click Update RMC.
7. The Update Module Configuration dialog box will be displayed to indicate the progress. If the RMC could not be reset automatically, you may be prompted to reset it manually.
8. In the RMC Configuration dialog box, click Close.

Step 2: Configure the Analog Transducer Type for the Velocity Input

Refer to Configuring the Analog Transducer Type for details on selecting between a voltage or current input type.

Step 3: Set the Scale, Extend Limit, and Retract Limit Parameters for the Velocity Input

Refer to the individual field sections for details on setting each parameter.

Step 4: Set the Count Offset for the Velocity Input

Refer to the Set Count Offset (35) command.

Analog Input Ranges

See the COUNTS topic for detailed information on the maximum and minimum limits of the various analog input ranges.

See also:

Analog Transducer Overview

Analog Transducer Wiring

Analog Transducer Configuration

Using Analog Channels as Position Inputs

Using Analog Channels as Velocity Inputs

Using Analog Channels as Pressure Inputs

Using Analog Channels as Differential Force Inputs

6.1.7.3 Using Analog Channels as Pressure Inputs

Analog channels may be configured to be used as one of three pressure input types. All three modes require the Pressure Control firmware option:

- **Pressure Control**
  The input is used with the corresponding drive output (in this mode, analog modules without drive outputs cannot be used for pressure control) for closed-loop, pressure control. See Controlling Solely Position, Pressure, or Force for details.
- **Auxiliary Pressure**
  The input does not have its own drive output. However, pressure control is achieved by sharing a drive output of another input. This is used in applications where an axis switches between closed-loop position control and closed-loop pressure control. See Transitioning from Position to Auxiliary Pressure/Force Control for details.
- **Pressure Reference**
  The input is used without the drive output. Because no drive output is used, the axis cannot do closed-loop control. This is most often used in cases where the user wishes to have an outside source (such as a joystick or PLC) control the target pressure, in which case another axis will be slaved to the pressure reference input. See Using an External Target Generator for details.

All fields are used as described in the pressure/force status, command, and parameter field topics, except when pressure reference mode is used. In this mode on all parameters are ignored except for Configuration word, Scale A, and Offset A.

Step 1: Assign Analog Channel(s) to be Pressure Input(s)

To assign analog channels as pressure inputs, do the following:

1. On the Tools menu, click Module Configuration.
2. In the Slots list, click an analog module.
3. Click Slot options.
4. In the Analog Channel Assignment dialog box, click the tab of the channel pair (0-1 or 2-3) you wish to assign.
5. Click the option button of a channel assignment that designates one or more channels as Pressure Control, Auxiliary Pressure, and/or Pressure Reference. The assignment selected depends on how you intend to use the all channels.
6. Click Update RMC.
7. The Update Module Configuration dialog box will be displayed to indicate the progress. If the RMC could not be reset automatically, you may be prompted to reset it manually.
8. In the RMC Configuration dialog box, click Close.

The following general steps are necessary for setting up pressure inputs:

Step 2: Configure the Analog Transducer Type for the Pressure Inputs

Refer to Configuring the Analog Transducer Type for details on selecting a voltage or current input type.

Step 3: Set the Scale A and Offset A Parameters for the Pressure Inputs
Refer to the individual field sections for details on setting these parameters.

Analog Input Ranges
See the COUNTS topic for detailed information on the maximum and minimum limits of the various analog input ranges.

See also:

Analog Transducer Overview

Analog Transducer Wiring

Analog Transducer Configuration

Using Analog Channels as Position Inputs

Using Analog Channels as Velocity Inputs

Using Analog Channels as Differential Force Inputs

6.1.7.4 Using Analog Channels as Differential Force Inputs

Differential force control uses two separate analog channels as pressure inputs to generate one force reading. Both inputs are scaled to forces, and then one is subtracted from the other, giving a final differential force.

Notice that force and pressure are not interchangeable terms. This is because it is only acceptable to subtract pressures if the surface areas acted on by the pressure are equal. In a typical hydraulic cylinder with a single-ended rod, this is never the case because the rod takes up part of the surface area on one side of the cylinder.

Analog channels may be configured to be used as one of two differential force input types. Both require the Pressure Control firmware option:

- **Differential Force Control**
  Two pressure inputs are used with the corresponding drive output (in this mode, analog modules without drive outputs cannot be used for pressure control) for closed-loop, differential-force control. See Controlling Solely Position, Pressure, or Force for details.
- **Auxiliary Differential Force**
  The inputs do not have their own drive output. However, differential force control is achieved by sharing a drive output of another input. This is used in applications where an axis switches between closed-loop position control and closed-loop differential-force control. See Transitioning from Position to Auxiliary Pressure/Force Control for details.

The following general steps are necessary for setting up differential force inputs:

Step 1: Assign Analog Channels to be an Differential Force Input
To assign a pair of analog channels to be a differential-force input, do the following:

1. On the Tools menu, click Module Configuration.
2. In the Slots list, click an analog module.
3. Click Slot Options.
4. In the Analog Channel Assignment dialog box, click the tab of the channel pair (0-1 or 2-3) you wish to assign.
5. Click the option button of a channel assignment of either differential force control or auxiliary differential force.
6. Click Update RMC.
7. The Update Module Configuration dialog box will be displayed to indicate the progress. If the RMC could not be reset automatically, you may be prompted to reset it manually.
8. In the RMC Configuration dialog box, click Close.

Step 2: Configure the Analog Transducer Type for the Differential Force Input

Refer to Configuring the Analog Transducer Type for details on selecting a voltage or current input type. Notice that both channels will be affected by this configuration.

Step 3: Set the Scale A, Offset A, Scale B, and Offset B Parameters for the Differential Force Input

Refer to the individual field sections for details on setting these parameters. The first channel in the pair is channel A, and the second channel is channel B.

Analog Input Ranges

See the COUNTS topic for detailed information on the maximum and minimum limits of the various analog input ranges.

See also:

Analog Transducer Overview

Analog Transducer Wiring

Analog Transducer Configuration

Using Analog Channels as Position Inputs

Using Analog Channels as Velocity Inputs

Using Analog Channels as Pressure Inputs

6.1.8 Usage Details

6.1.8.1 Using Position and Velocity Joysticks

The analog module allows an external analog signal to act as the target of another axis. This is used for joysticks controlling either position or velocity.

To use a joystick to control speed together with a tachometer requires an analog module with drive outputs; therefore, only the 16-bit modules can be used. For details on speed control, see the Speed Control topic.

For details on configuring the analog module for your control and transducer type, refer to Using Analog Channels as Velocity Inputs, and Configuring the Analog Transducer Type.

The joystick feedback can be filtered to provide smooth motion. See the Reference Axis Filtering for details.

This topic has not been completed. Please contact Delta Computer Systems, Inc. for details on this topic.

See also:

Analog Transducer Overview

Analog Transducer Wiring

Analog Transducer Configuration

Using Analog Channels as Position Inputs

Using Analog Channels as Velocity Inputs

Controlling Speed from a Tachometer Feedback

Controlling Solely Position, Pressure, or Force

6.1.8.2 Using an External Target Generator

Some applications require that the target generator be external to the RMC. This target generator may be a PLC, a signal generator (such as an oscillator), or a joystick. Joysticks are discussed in more detail under Using Position and Velocity Joysticks.

Using an external target generator requires the following components:

- The target generator must provide an analog signal in one of the formats supported by the RMC’s analog modules (0 to 10V, ±10V, 0 to 5V, ±5V, and 4 to 20mA).
- A channel on either a 12-bit or 16-bit RMC analog module must be assigned to be the reference input (you must select position, velocity, or pressure reference).

Note: The reference input may also be any other axis type, even one that is controlling its own axis. The reference input is unaware that another axis is gearing to it.

- An axis (it may use any supported transducer type: quadrature, MDT, or analog) must be configured to control the same item as the reference input (you must select position, velocity, or pressure control; MDT and quadrature axes are always in position control).
- The control axis must be slaved to the reference input.

Step-by-Step External Target Generation

Follow these steps to set up a system using an external target generator:

1. **Assign the Analog Axes**

   Each of the channels on the analog module(s) must be assigned to a role. The reference input must be assigned to one of the three reference types (position, velocity, or pressure). Refer to Analog Transducer Configuration for details on assigning channels.

2. **Configure the Reference Input**

   Analog inputs must be configured to select the type of the input. These options include 0 to 10V, ±10V, 0 to 5V, ±5V, and 4 to 20mA. See Analog Transducer Configuration for details on setting this value.

3. **Configure the Control Axis**

   See the configuration topic of the transducer type you are using.

4. **Determine the Position Units and Limits of the Reference Input and Control Axis**

   Use the Scale and Offset parameters to select position units on the reference input and control axis. Remember that these position units will be used in the gearing ratio defined in step 6.

5. **Initialize the Control Axis and Reference Input**

   Send the above parameters followed by a Set Parameters (P) command to the control axis and reference input.

6. **Slave the Control Axis to the Reference Input**

   Use the Go (G) command to configure the control axis as a gearing slave. This is done by setting the Gearing bit in the Mode command field, and the numerator and denominator of the gear ratio in the Command Value and Speed command fields of the Go command. For details on Gearing, see Gearing Axes.

Note: This does not align the control axis with the reference axis. It just ensures that any further moves of the reference input will cause the control axis to move the same amount adjusted by the gear ratio. You must use a separate move command to align them.

Example:
Suppose you scaled the reference input so it will move between 1,000 and 10,000 position units. You want the control axis to be geared to the same range. However, after you do steps 1 through 5 the positions of the two inputs will most likely not be equal.

For example, assume the reference input's actual position is 1,500 and the control axis's actual position is 5,500. If the axes were geared together at this point with a 1:1 ratio, then the control axis's position will always remain 4,000 position units higher than the reference input's position.

Therefore, you will need to move the control axis's position back to match the reference input's position before gearing them. The simplest way is to use one of the Move Relative to An Axis (0xC0-0xCF) commands to move the command axis relative to the reference input.

6.1.8.3 Controlling Speed from a Tachometer Feedback

In addition to controlling speed with position feedback, the RMC100 is able to control speed with a tachometer feedback. To do this, an analog module with drive outputs is required. For details on configuring the module for your control and transducer type, refer to Using Analog Channels as Velocity Inputs, and Configuring the Analog Transducer Type.

The procedure for speed control with tachometer feedback is similar to speed control with position feedback. However, a tachometer with analog feedback will often have a small offset. When the tachometer is stopped, the feedback voltage may be slightly different than zero. This may cause the axis to rotate slowly when it has been commanded to stop in closed loop control. To adjust the tachometer feedback, use the Set Count Offset (#) command.

See also:

Analog Transducer Overview

Analog Transducer Wiring

Analog Transducer Configuration

Using Analog Channels as Velocity Inputs

Using Position and Velocity Joysticks

6.1.8.4 Controlling Analog Position, Pressure, or Force

Controlling analog position, pressure, or force requires an analog module with drive outputs; therefore, only the 16-bit modules can be used. Controlling pressure or force requires the pressure-control firmware option.

For controlling solely pressure or force, see the Controlling Solely Pressure or Force topic.

For position-pressure control, see the Position-Pressure Overview topic.

For details on configuring the module for your control and transducer type, refer to Using Analog Channels as Position Inputs, Using Analog Channels as Pressure Inputs, Using Analog Channels as Differential Force Inputs, and Configuring the Analog Transducer Type.

Analog Input Ranges
See the COUNTS topic for detailed information on the maximum and minimum limits of the various analog input ranges.

See also:

Analog Transducer Overview

Analog Transducer Wiring

Analog Transducer Configuration

Using Analog Channels as Position Inputs

Using Analog Channels as Pressure Inputs

Using Analog Channels as Differential Force Inputs

Using Position and Velocity Joysticks

Controlling Speed from a Tachometer Feedback

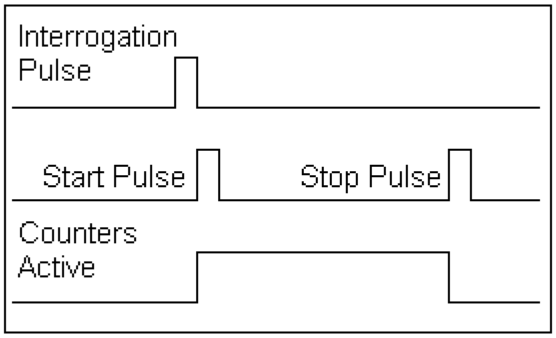
Transitioning from Position to Auxiliary Pressure/Force Control

6.2 MDT

6.2.1 MDT Overview

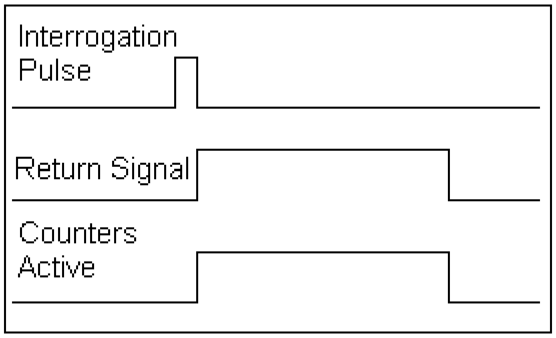
Magnetostrictive displacement transducers are designed for use in rugged industrial environments. They are non-contact, wear-free, highly reliable, and offer accurate and repeatable linear position measurement. In the motion control industry magnetostrictive displacement transducers are typically inserted into hydraulic cylinders for measurement of the cylinders' extension/retraction position.

Each RMC100 MDT interface module has circuitry for multiple magnetostrictive transducers. Each axis can be configured for a Start/Stop transducer or a Pulse Width Modulated transducer. To make a measurement with a Start/Stop transducer, the RMC sends an interrogation pulse to the transducer. The transducer responds by returning 2 pulses—a Start pulse and a Stop pulse. The RMC's internal counters begin to count when the first pulse, Start, is received and stop counting when the second pulse, Stop, is received. The time between the start pulse and the stop pulse is proportional to the transducer position.



Start/Stop Pulse Transducer

To make a measurement with a Pulse Width Modulated transducer, the RMC sends an interrogation pulse to the transducer. The transducer responds with a return signal. The return signal is high while the transducer is determining its' position. The counters on the RMC100 MDT interface module are counting during the time that the return signal is high. The time that the return signal is high is proportional to the transducer position.



Pulse Width Modulated Transducer

The RMC must then convert the counts accumulated during the transducer interrogation to an ACTUAL POSITION in user-defined Position Units (usually 0.001 inch) for use in the PID control loop.

See also:

MDT Wiring

MDT Configuration

MDT LED Indicators

MDT Specifications

6.2.2 MDT Wiring

Note: When positive voltage is sent to an axis's drive, the axis must extend. The extend direction is defined as the direction that causes the transducer to return increasing counts. The extend direction of a magnetostrictive transducer is away from the head.

Use shielded twisted pairs for all connections to inputs and outputs. Route the transducer wiring separate from other wiring. You must provide the power supplies needed by your transducers.

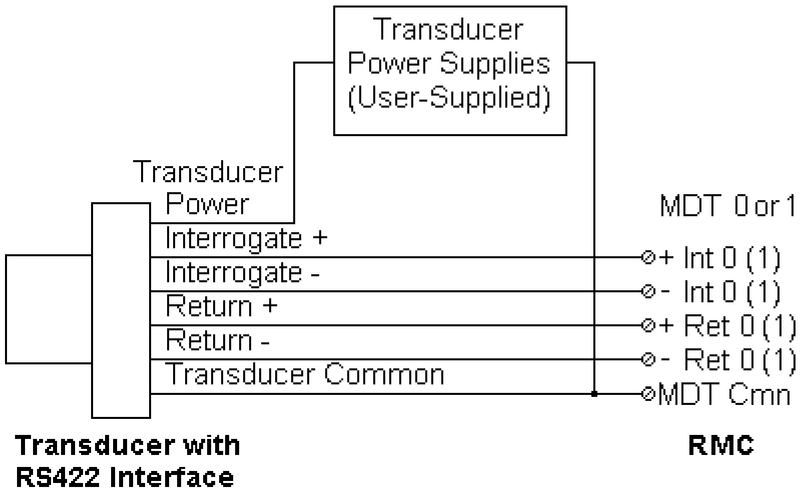
RMC Magnetostrictive Transducer Input

Six-Pin Plug-in Terminal Block

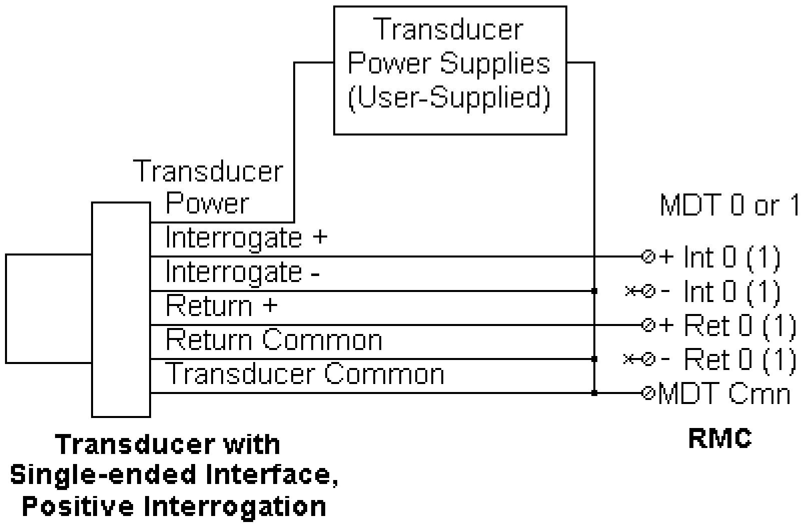
Pin	**Function**
1	MDT Axis + Interrogation
2	MDT Axis - Interrogation
3	MDT Axis Common
4	MDT Axis + Return
5	MDT Axis - Return
6	Case

Note: The following example schematics do not include transducer pin numbers, color codes, or power supply requirements, since these vary between different transducers. To determine your power supply needs and connector pin-outs or cable color codes, consult your transducer manufacturers' documentation.

The RMC can interface to transducers with either single-ended (TTL) or Differential Line Driver (RS422) interrogation signals. When wiring RS422 signals, connect both the '+Int' and '-Int' between the transducer and the RMC for the interrogation signal, and the '+Ret' and '-Ret' between the transducer and the RMC for the return signal. Connect the transducer DC ground to MDT Cmn.

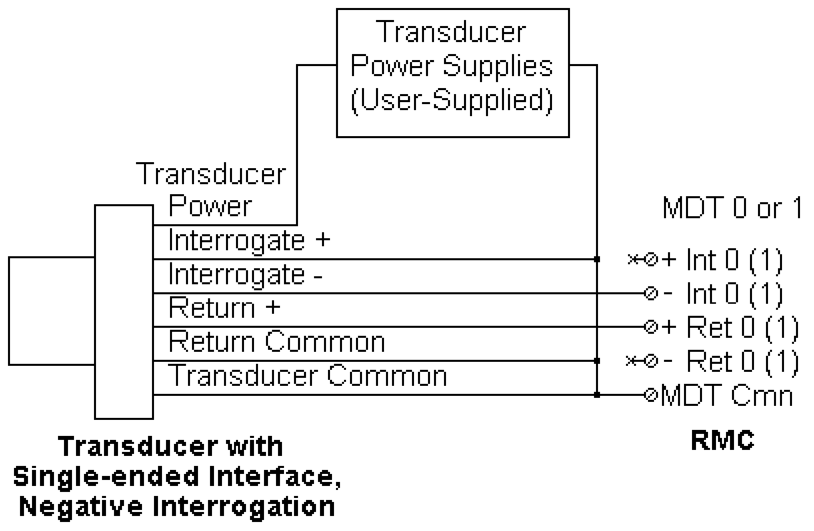


For a single-ended transducer with positive interrogation, connect the transducer '- interrogation in' wire to the ’r;MDT Cmn’ pin and the transducer '+ interrogation in' wire to the '+ Int' pin. CONNECT NOTHING TO THE '-Int' PIN OF THE RMC. Connect the transducer return plus wire to the '+Ret’ pin on the RMC and the transducer return common wire to ’r;MDT Cmn’ on the RMC. CONNECT NOTHING TO THE '-Ret' PIN OF THE RMC.



Temposonics I transducer users:

For the negative interrogation version of this transducer, connect the transducer '+ interrogation in' wire to the ’r;MDT Cmn’ pin and the transducer '- interrogation in' wire to the '-Int' pin. CONNECT NOTHING TO THE '+Int' PIN OF THE RMC. Connect the transducer return plus wire to the '+Ret’ pin on the RMC and the transducer return common wire to ’r;MDT Cmn’ on the RMC. CONNECT NOTHING TO THE '-Ret' PIN OF THE RMC.



Some Temposonics I transducers from MTS have 200 Ohm termination resistors installed between their interrogation pins and common. If yours do not, it may be necessary to install them as close to the transducers as possible to reduce electrical noise in the system.

RMC Drive Outputs

Four-Pin Plug-in Terminal Block

Pin	Function
1	Axis 0 Drive
2	Drive Common
3	Axis 1 Drive
4	Case

When positive voltage is sent to an axis's drive, the axis must extend. The extend direction is defined as the direction that causes the transducer to return increasing counts. The extend direction of a magnetostrictive transducer is away from the head.

CAUTION: If the outputs from the RMC are reversed, the axis will be uncontrollable when power is connected. Confirm that your wiring is correct!

See also:

General Wiring Information

MDT Overview

MDT Configuration

MDT LED Indicators

MDT Specifications

6.2.3 MDT Configuration

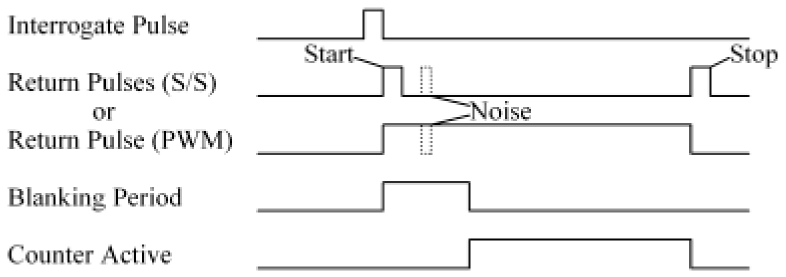
The RMC supports a wide range of Magnetostrictive Displacement Transducers. To select the type of the transducer and polarity of the drive output, the following settings can be changed:

- Support for Pulse-Width Modulated or Start-Stop MDTs.
- Support for between 1 and 15 recirculations on PWM MDTs.
- Support for using either edge of a Start-Stop pulse. This is required to support some Balluff transducers.
- Support for short and long blanking periods. The correct setting of this parameter depends on the transducer housing type. See the description below.
- Support for reversing the drive output in software.

These settings are set from software; no jumpers are required. Except for the blanking period, these settings are all in the Configuration word parameter; refer to that help topic for further details. See the descriptions below for details on the blanking period settings.

Blanking Period

The blanking period determines the amount of time to delay before starting the counter for an MDT. The time delay begins after the rising edge of the Start pulse on a Start/Stop MDT or the return pulse on a PWM MDT. Refer to the following diagram:



The reason for the blanking period is to ignore noise that occurs after edges on the Start or PWM pulse. This was especially a problem with older transducers. As shown in the diagram above, noise during the blanking period is ignored.

However, the longer the blanking period is, the longer the minimum distance measurable by the motion controller becomes. That is, if the Stop pulse comes back during the blanking period, it is not registered, and no position can be read. This becomes a problem only with the newer clevis-mount transducers (for example, the IP-67 housing type from Temposonics).

Therefore, to allow for this trade-off, the blanking period can be set through RMCWin to be either 5ms (recommended for clevis-mount transducers) or 21ms (recommended for all other transducers). This setting cannot be changed from a PLC, but must instead be set through RMCWin and saved in the RMC's Flash memory.

Therefore, to allow for this trade-off, the blanking period may be configured:

1. On the Tools menu, click Module Configuration.
2. In the Slots list, click the MDT module you want to edit.
3. Click Slot options. The MDT Options dialog box will be displayed with a tab for each axis on that slot.
4. Click the Axis 0 tab.
5. Click Standard (21ms), unless the transducer on that axis is a clevis-mounted transducer, in which case you would click Short (5ms).
6. Click the Axis 1 tab.
7. Click Standard (21ms), unless the transducer on that axis is a clevis-mounted transducer, in which case you would click Short (5ms).
8. Click Update RMC.
9. The Update Module Configuration dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module yourself.

See also:

MDT Overview

MDT Wiring

MDT LED Indicators

MDT Specifications

6.2.4 MDT LED Indicators

The two LEDs above the Drive connector on the RMC's MDT interface card are axis status LEDs. These LEDs will reflect the operational status of their corresponding axis according to the following table.

LED Action	**Axis Status**
Continuous Red	One or more of these errors have occurred: No Transducer Transducer Noise Transducer Overflow
Alternating Red/Green	None of the above errors occurred, and one or more of the below errors have occurred and are enabled in the Auto Stop:

	Overdrive Error
	Parameter Error
	Position Overflow
	Integrator Windup
	Following Error
Continuous Green	Status good. None of the above are true.

Note: Prior to RMC CPU firmware dated 19991216, the Auto Stop parameter was not used in determining the LED states. Therefore, the only way to keep the LED from showing red was to clear the error.

See also:

MDT Overview

MDT Wiring

MDT Configuration

MDT Specifications

6.2.5 MDT Specifications

For general specifications on the RMC, see RMC100 Specifications. See SSI Overview for details on using MDTs with SSI output.

MDT Interface

Axes	Two per module
Inputs	Two RS422 differential (can also be used single-ended)
Outputs	Two RS422 differential (can also be used single-ended)
ESD protection	15kV
Resolution	0.001" Start/Stop; 0.0001" PWM
Circulations	1 Start/Stop; up to 16 PWM
Count rate	120MHz

Drive Outputs

Range	±10 V @ 5 mA (2 kW or greater load) (For current drive, use the VC2100 accessory: ±10 mA to ±200 mA in 10 mA steps)
Tolerance	At 10 V: +200 mV, -100 mV At 0 V: ±50 mV At -10 V: +100 mV, -200 mV
Resolution	12 bits
Output isolation	750 VDC, optically isolated
Overload protection	One-second short-circuit duration
Overvoltage protection	Outputs are protected by clamp diodes

See also:

MDT Overview

MDT Wiring

MDT Configuration

MDT LED Indicators

6.2.6 MDT Scaling

Defining the Valid 16-bit Position Range

For general scaling information, see the Scaling Overview topic.

Because the RMC uses 16-bit positions, positions must all fit within a range of 65,536 position units. Because position units are user definable, this range does not limit most applications. See the section below on defining position units.

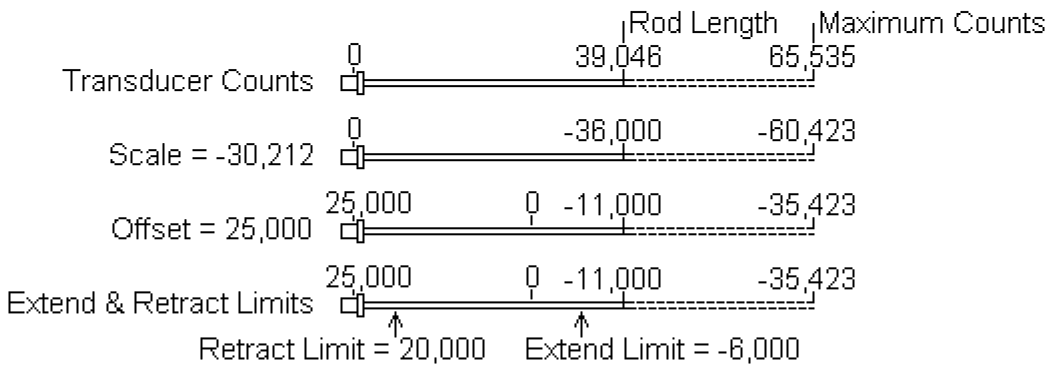
For MDT axes, the Offset parameter is used with the Scale parameter to define the position range. If the Scale parameter is negative, then the position range extends from the Offset value minus 65535 up to the Offset value. If the Scale parameter is non-negative, then the position range extends from the Offset value up to the Offset value plus 65535. The following chart summarizes this concept:

Scale	Min. Position	Max. Position
< 0	Offset - 65535	Offset
³ 0	Offset	Offset + 65535

However, because the Offset is also used to convert transducer counts to position units, it cannot be set independently.

Translating to and from Position Units

The Scale, Offset, and the Prescale Divisor bits of the Configuration word parameters are used to define position units as a function of transducer counts. Below is shown an example MDT and the effect of the Scale, Offset, Extend and Retract Limits:



The following formula summarizes the translation from transducer counts to Actual Position units:

$$\text{Actual Position} = \frac{\text{Counts} \times \text{Scale}}{32768 \times \text{Prescale Divisor}} + \text{Offset}$$

If the Actual Position overflows the valid 16-bit position range, as described above, the Position Overflow bit will turn on, forcing the axis to halt.

There are two main ways to calculate the Scale, Offset, and Prescale Divisor bits. Each method is described below. Notice that both have automatic Calibration Utilities built into RMCWin. It is recommended that these be used, but the underlying math is described below.

Method 1: P0/P1 Calculation

The simplest way is the physically measure the axis's position at two points and read how many counts the RMC reports at each position. If we call the two positions, in user position units, P0 and P1, and call the corresponding counts C0 and C1, the following two equations will give a Scale and Offset. Notice that the Prescale Divisor is left out at this point; it is assumed to be 1:

$$\text{Scale} = \frac{(P_0 - P_1)}{(C_0 - C_1)} \times 32768$$

$$\text{Offset} = P_0 - \frac{(P_0 - P_1)}{(C_0 - C_1)} \times C_0$$

Once the Scale is calculated, the Prescale Divisor can be calculated. The Prescale Divisor can have values of 1, 2, 4, or 8. Because the Scale is always divided by the Prescale Divisor, you essentially have a fractional scale. Pick the largest Prescale Divisor you can multiply the Scale by and still be between 32768 and 32768. For example, suppose your Scale comes to 6324.70. The following table shows the possible Scales and Prescale Divisors you could use and the effective scale:

Scale	**Divisor**	**Effective Scale**	**Error from 6324.70**
6325	1	6325/1 = 6325	0.005%
12649	2	12649/2 = 6324.5	0.003%
25299	4	25299/4 = 6324.75	0.0008%
50598	8	Invalid scale	Invalid

Therefore, in this example, a Scale of 25299 and a Prescale Divisor of 4 should be used. If the Scale you calculate comes to 32768, enter 0 instead.

These calculations can be done automatically using the Position Scale/Offset Calibration Utility feature in RMCWin.

Method 2: Using the MDT Calibration Number

The Scale can also be calculated using the MDT calibration number (nominally 9.012 microseconds per inch) with the following formula. Notice that the Prescale Divisor is left out at this point; it is calculated after the initial Scale value has been calculated, as described in Method 1:

$$\text{Scale} = \frac{\text{Position Units per Inch} \times 32768}{\text{Cal. Number } (\mu s/\text{in}) \times 120\text{MHz} \times \#\text{ of Recirculations}} \times \textit{Sign}$$

In the above formula, the 120MHz comes from the RMC’s internal counter, and Sign equals +1 when an extend move yields increasing Actual Position counts, and -1 when an extend move yields decreasing Actual Position counts. The calibration number must be specified in the same units as the Position Units (inches, mm, etc.). Position Units are generally 0.001 inches or 0.1 mm. Usually there are 1000 Position Units per inch when the calibration number is in microseconds per inch.

Once the Scale and Prescale Divisor have been calculated, the Offset can be calculated:

$$\text{Offset} = \text{P}_0 - \frac{\text{Scale} \times \text{C}_0}{32768 \times \text{Prescale Divisor}}$$

These calculations are done automatically using the MDT Scale/Offset Calibration Utility feature in RMCWin.

Example 1

A system has an MDT with a calibration number of 9.1392 ms per inch and 1 recirculation. The desired position units are thousandths of an inch. Therefore, there are 1000 position units per inch. At the desired 0 position, the MDT produces 425 counts.

We first calculate the exact Scale:

$$\text{Scale} = \frac{1000(\text{Position Units per Inch}) \times 32768}{9.1392(\mu\text{s/in}) \times 120\text{MHz} \times 1\,(\text{Recirculations})} = 29878.62$$

With the exact Scale value, we must choose the Prescale Divisor and the rounded Scale value. The highest divisor value we can use is 1; multiplying the scale by 2, 4, or 8 would all overflow the Scale limits of ±32767. Therefore we have the following parameters:

```
Scale = 29879
Prescale Divisor = 1
```

Next, we must calculate the offset:

$$\text{Offset} = 0 - \frac{29879 \times 425}{32768 \times 1} = -387.5 = -388$$

Example 2

A system has an MDT with a calibration number of 9.0108 ms per inch and 1 recirculation. The desired position units are tenths of a millimeter. At the desired 0 position, the MDT produces 1122 counts.

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:

$$\frac{10\text{ Position Units}}{1\text{ millimeter}} = \frac{25.4\text{ millimeters}}{1\text{ inch}} = \frac{254\text{ Position Units}}{1\text{ inch}}$$

So, we now calculate the exact Scale:

$$\text{Scale} = \frac{254(\text{Position Units per Inch}) \times 32768}{9.0108(\mu\text{s/in}) \times 120\text{MHz} \times 1\,(\text{Recirculations})} = 7697.31$$

With the exact Scale value, we must choose the Prescale Divisor and the rounded Scale value. The highest divisor value we can use is 4; multiplying the scale by 8 would overflow the Scale limits of ±32767. Multiplying the exact scale by this divisor value and rounding gives us the following parameters:

```
Scale = 30789
Prescale Divisor = 4
```

Next, we must calculate the offset:

$$\text{Offset} = 0 - \frac{30789 \times 1122}{32768 \times 4} = -263.6 = -264$$

Example 3

A system has an MDT with a calibration number of 8.9772 ms per inch and 8 recirculations. The desired position units are thousandths of an inch. Therefore, there are a thousand position units per inch. At the desired 0 position, the MDT produces 28226 counts.

We begin by calculating the exact Scale:

$$\text{Scale} = \frac{1000(\text{Position Units per Inch}) \times 32768}{8.9772(\mu s/\text{in}) \times 120\text{MHz} \times 8\,(\text{Recirculations})} = 3802.22$$

With the exact Scale value, we must choose the Prescale Divisor and the rounded Scale value. The highest divisor value we can use is 8. Multiplying the exact scale by this divisor value and rounding gives us the following parameters:

```
Scale = 30418
```

```
Prescale Divisor = 8
```

Next, we must calculate the offset:

$$\text{Offset} = 0 - \frac{30418 \times 28226}{32768 \times 8} = -3275.2 = -3275$$

6.3 Quadrature with Analog Output

6.3.1 Quadrature with Analog Output Overview

The QUAD module is one of two RMC modules with an interface for quadrature encoder feedback. This module provides analog drive output, while the STEP interface module provides stepper drive output. Together, these modules allow the RMC100 series motion controllers to control a wide range of motors and linear actuators with quadrature encoder feedback. For details on the STEP module, see Quadrature with Stepper Output Overview.

Features

- Two Complete Axes per Module. Each includes the following:
  - Quadrature Encoder Interface:
    - 4,000,000 counts/second
    - A and B Inputs
    - Index Input with High-speed 125hs Position Latch
  - Drive Amplifier Interface:
    - Isolated ±10V, 12-bit Analog Drive Output
    - Amplifier Enable Output
    - Amplifier Fault Input
  - Additional Inputs:
    - Extend (CW) Travel Limit Input
    - Retract (CCW) Travel Limit Input
    - Home Input with High-speed 50µs Position Latch

- Status LED
- Digital Noise Filters on All Inputs
- All Discrete Inputs are Isolated
- Use with Servo Drives in Velocity or Torque/Force Modes

Quadrature Encoder Inputs

The A and B signals from the encoder are decoded to generate a positive or negative count; the count is then used to monitor the axis position. Each signal from the encoder is received into a differential line receiver for compatibility with differential line driver encoders. The negative input is biased so some encoders with open collector or TTL outputs can also be used, however such encoders should only be used in lab environments with very little electrical noise and short wire runs.

Index (Z) Input

The index input is normally active for one count of each quadrature encoder revolution. It is used to qualify or narrow the effective width of the home input to make the homing more repeatable; see Homing a Quadrature Axis for details. The user may change the active state of this input by changing the Home Active State bit in the axis's Config word.

Home (H) Input

This input may be used in homing the system; see Homing a Quadrature Axis for details. This input will be connected to a normally-off proximity switch. The user may change the active state of this input by changing the Home Active State bit in the axis's Config word.

Amplifier Enable Output

This output to the drive amplifier allows the RMC to turn off the amplifier using the Amp Enable/Disable command. The enable output (labeled ENABLE + and -) is a solid state relay that is closed when enabled or active.

Amplifier Fault Input

FAULT + and - is an input from the drive amplifier or some other source that tells the RMC it no longer has control and should output the null drive, usually zero volts, and go into open loop mode. The user can select the active state of this input by the Fault Active State bit in the axis's Config word; by default the RMC is set-up to fault when current stops flowing through the input. This way the axis will fault on loss of control power. Bit 15 of the Status word is set if a Fault or Encoder error has occurred. The Auto Stop parameter determines how the axis will stop when a fault occurs.

Note: The axis will soft stop when this error occurs if no auto stop bits are selected.

Extend and Retract Travel Limit Inputs

The Extend Limit (CW) and Retract Limit (CCW) switch inputs tell the RMC that a travel limit has been reached and the axis should halt. By default the limit switches are considered inactive when enough current is flowing through the RMC inputs. This way the limit switches will look active to the RMC if control power is lost. The user may change the active state by setting the Limit Switch Active State bit in the axis's Config word.

Note: There is only one bit in the Config word for both limit switches so both limit switches must have the same active state.

If a limit switch is not used it should be wired in the inactive state or the Limit Switch Active State bit must be configured to disable the inputs. The Auto Stop bits must be set to automatically stop the axis.

See also:
Quadrature Wiring
Quadrature Configuration
Quadrature LED Indicators
Quadrature Specifications
Quadrature Scaling
Quadrature Homing

6.3.2 Quadrature Wiring

Use shielded twisted pairs for all connections to inputs and outputs. Route the quadrature encoder wiring separate from other wiring. You must provide the power supplies needed by your quadrature encoders.

CAUTION: If the drive outputs from the RMC are reversed, the axis will be uncontrollable when power is connected. Confirm that your wiring is correct!

DB25S Pin-out

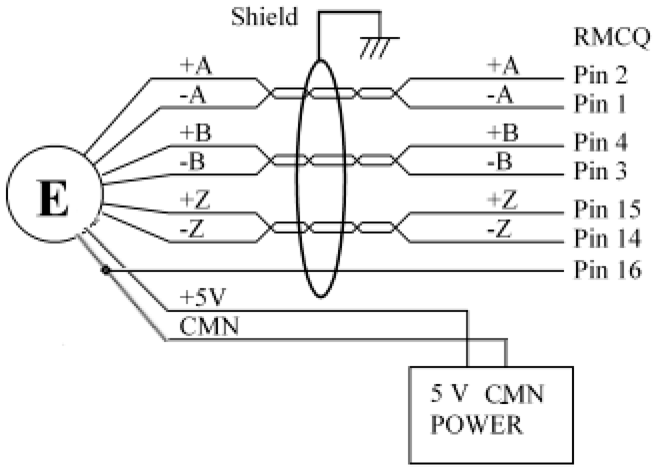
Note: Color codes are for cable part number RMC-CB-QUAD-01. This cable has three wire groups: encoder, limits, and drive. The wire color column lists the wire group and the individual wire color within that group.
The encoder common should be connected to both wires listed.

Pin	Function	Wire Color	Pin	Function	Wire Color
1	A-	Enc: white/blue	14	Index (Z)-	Enc: white/green
2	A+	Enc: blue/white	15	Index (Z)+	Enc: green/white
3	B-	Enc: white/orange	16	Encoder Common	Enc: brown/white
4	B+	Enc: orange/white			Enc: white/brown
5	N/C		17	N/C	
6	Retract Lim-	Lim: white/orange	18	Home-	Lim: white/green
7	Retract Lim+	Lim: orange/white	19	Home+	Lim: green/white
8	Extend Lim-	Lim: white/blue	20	Fault-	Drv: white/green
9	Extend Lim+	Lim: blue/white	21	Fault+	Drv: green/white
10	N/C		22	N/C	
11	N/C		23	N/C	
12	Drive*	Drv: blue/white	24	Enable*-	Drv: white/orange
13	Drive Common*	Drv: white/blue	25	Enable*+	Drv: orange/white

*Outputs, all others are inputs

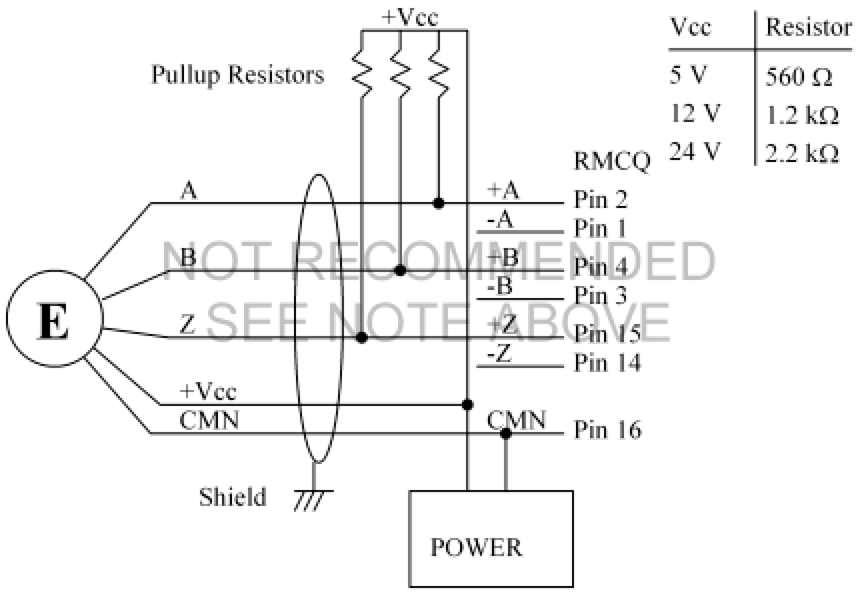
Encoder Wiring

5 Volt differential driver:



NPN Open Collector (NOT RECOMMENDED):

Note: Open collector encoders should only be used in lab environments with very little electrical noise and short wire runs.



Vcc	Resistor
5 V	560 Ω
12 V	1.2 kΩ
24 V	2.2 kΩ

Noise immunity can be improved in the diagram above by adding a capacitor across each RMC encoder input: from +A to -A, from +B to -B, and from +Z to -Z. Notice the capacitors are connected to the minus inputs even though no signal wires are connected there. The capacitor will limit the frequency response of the input. Some suggested capacitor values are listed along with the resultant frequency response limit:

Capacitor (µF)	Frequency

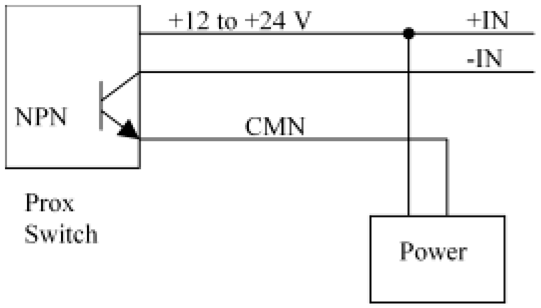
0.0047	1 MHz
0.010	500 kHz
0.022	200 kHz
0.047	100 kHz
0.10	50 kHz
0.22	20 kHz
0.47	10 kHz
1.0	5 kHz

Input Wiring (Home and Limits)

NPN Proximity sensors

Normally Open recommended for Home

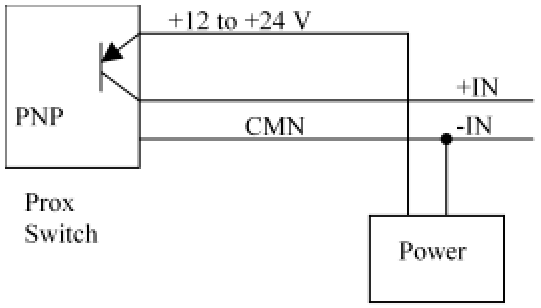
Normally Closed recommended for Limits



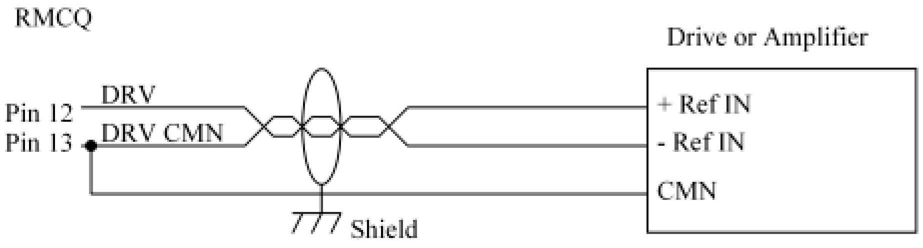
PNP Proximity sensors

Normally Open recommended for Home

Normally Closed recommended for Limits

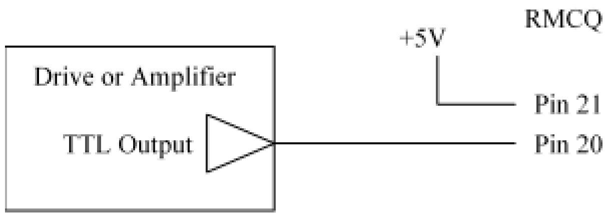


Drive Output Wiring

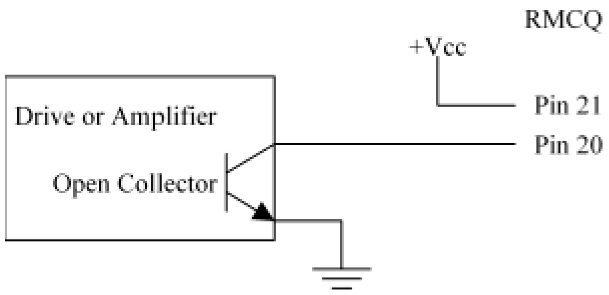


Fault Input Wiring

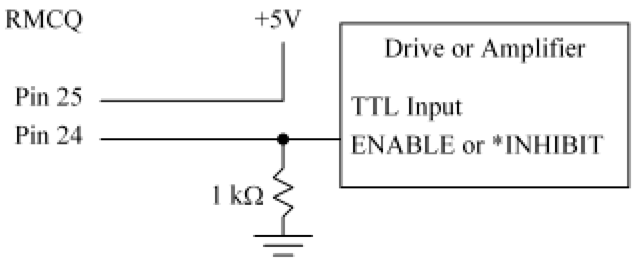
From TTL output:



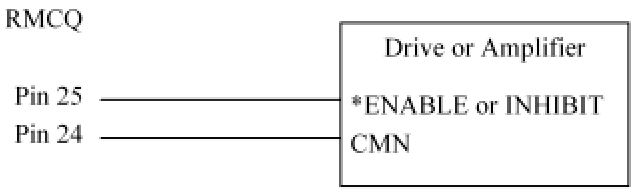
From Open Collector Output:



Enable Output Wiring
To TTL input (high = enable):



To active low Enable input:



See also:
General Wiring Information
Quadrature Overview
Quadrature/Analog Cable
Quadrature Configuration
Quadrature LED Indicators
Quadrature Specifications
Quadrature Scaling
Quadrature Homing

6.3.3 Quadrature/Analog Cable

A cable can be purchased that connects directly to an axis's DB-25 connector on the RMC QUAD module. The cable can be purchased in one of three lengths. It separates the wires into three groups: drive, encoder, and limits. Each group has its own braided shield and insulation.

Quadrature Input/Analog Output Cable

Part Number	RMC-CB-QUAD-01
Length	6 feet (2 m), 10 ft (3 m), or 15 ft (4.5 m)
Connector	DB-25 male with shielded housing (mates to connector on RMC QUAD)
User Connections	Flying leads (pigtails)
Cable	Each DB-25 connectors has three cables coming from it. Each has 24-gauge twisted pairs with an overall braided shield: • Drive (3 pair): Amplifier connections • Encoder (4 pair): Quadrature encoder connections • Limits (3 pair): Home and limit switches

See Quadrature Wiring for wire color codes.

6.3.4 Quadrature Configuration

The RMC supports a wide range of quadrature encoders and homing configurations. To select the type of the quadrature encoder and the polarity of the drive output, the following settings can be changed:

- Active state of the Index (Z), Home (H), Fault, and Limit inputs. This is set in the Configuration word. See the Quadrature Specific Configuration help topic for details.
- How the Home status bit is used. This is set in the Configuration word. See the Quadrature Specific Configuration help topic for details.
- How the Encoder Error/Fault Input status bit is used. Setting this bit is described below.
- Support for reversing the drive output in software. This is set in the Configuration word. See that topic for details.

No jumpers are required for changing any of these settings. You need to set the Configuration word parameter for each axis used.

Defining the Encoder Error/Fault Input Status Bit

This status bit can represent either an encoder error or a fault error. The user configures which of these two conditions is reported in this status bit. There are four options:

- Encoder Error only. This bit will go high if the encoder circuitry detects an error, which is defined as an invalid transition of the A and B lines. This usually occurs due to over-speed or noise conditions.
- Fault Input only. This bit will go high if the Fault input goes active. The fault input is intended to be a means for the drive amplifier to let the RMC know that it no longer has control. If the fault input is not used, it should be made inactive by wiring or by reversing the polarity in the Configuration Word parameter.
- Encoder Error or Fault Input (default). This bit will go high if either of these conditions occurs. This is the default behavior for QUAD and STEP axes, but it is ambiguous whether the error was triggered due to an encoder error or the Fault input.
- Unused. This bit will always be low.

The use of this bit can be set only from RMCWin. Use the following steps:

1. On the Tools menu, click Module Configuration.
2. In the Slots list, click the Quadrature module you want to edit.
3. Click Slot options. The Quadrature with Analog Output Options dialog box will be displayed with a tab for each axis on that slot.
4. Click the Axis 0 tab.
5. Check the conditions that you want to have set this status bit. You can check one, both, or neither condition.
6. Click the Axis 1 tab.
7. Check the conditions that you want to have set this status bit. You can check one, both, or neither condition.
8. Click Update RMC.
9. The Update Module Configuration dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module yourself.

See also:
Quadrature Overview
Quadrature Wiring
Quadrature LED Indicators
Quadrature Specifications
Quadrature Scaling
Quadrature Homing

6.3.5 Quadrature LED Indicators

The two LEDs above the Drive connector on the RMC's Quadrature interface card are axis status LEDs. These LEDs will reflect the operational status, as determined by the Status Word, of their corresponding axis according to the following table.

LED Action	Axis Status

Alternating Red/Green	One or more of the following status bits are on and are enabled in the Auto Stop: Encoder Fault Limit Switches Home Input Overdrive Error Parameter Error Integrator Windup Following Error
Continuous Green	Status good. The above is not true.

Note: Prior to RMC CPU firmware dated 19991216, the Auto Stop parameter was not used in determining the LED states. Therefore, the only way to keep the LED from showing red was to clear the error.

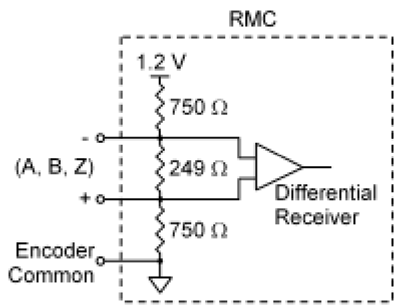
See also:
Quadrature Overview
Quadrature Wiring
Quadrature Configuration
Quadrature Specifications
Quadrature Scaling
Quadrature Homing

6.3.6 Quadrature Specifications

For general specifications on the RMC, see RMC100 Specifications.

Encoder

Axes	Two per module
Encoder Inputs	RS422 differential receiver (can also be used as TTL) Quadrature A, B, and Index Z
Input Impedance	215 W Differential Input 430 W Single-Ended Input



ESD Protection	15 kV
Max. Encoder Frequency	4,000,000 quadrature counts/second
Index (Z) Response Time	125 nanoseconds

Inputs and Outputs

Fault Inputs, Home Inputs, and Limit Inputs (Ext. & Ret.)	2.7 V @ 2.8 mA typical (3.2 V @ 3.5 mA max) threshold, 26.4 V maximum input voltage, 500 VDC isolation, compatible with most limit switches, TTL, and CMOS outputs
Home and Registration Input Response Time	50 microseconds
Home Options	Any combination of Index (Z) input and Home (H) input
Amplifier Enable Output (1 per axis)	Solid State relay, 50 W, 30 V, 100 mA 500 VDC isolation

Drive Outputs

Range	±10 V @ 5 mA (2 kW or greater load) (For current drive, use the VC2100 accessory: ±10 mA to ±200 mA in 10 mA steps)
Tolerance	At 10 V: +200 mV, -100 mV At 0 V: ±50 mV At -10 V: +100 mV, -200 mV
Resolution	12 bits for revisions 1, 2, and 4 14 bits for revisions 3 and 5 or newer
Output Isolation	500 VDC, optically isolated
Overload Protection	One-second short-circuit duration
Overvoltage Protection	Outputs are protected by clamp diodes

See also:
Quadrature Overview
Quadrature Wiring
Quadrature Configuration
Quadrature LED Indicators
Quadrature Scaling
Quadrature Homing

6.3.7 Quadrature Scaling

Defining the Valid 16-bit Position Range

For general scaling information, see the Scaling Overview topic.

Because the RMC uses 16-bit positions, positions must all fit within a range of 65,536 position units. Because position units are user definable, this range does not limit most applications. See the section below on defining position units.

For quadrature axes, the Coordinate Limit parameter is used with the Scale parameter to define the position range. If the Scale parameter is negative, then the position range extends from the Coordinate Limit value minus 65535 up to the Coordinate Limit value. If the Scale parameter is non-negative, then the position range extends from the Coordinate Limit value up to the Coordinate Limit value plus 65535. The following chart summarizes this concept:

Scale	Min. Position	Max. Position
< 0	Coord. Limit - 65535	Coord. Limit
³ 0	Coord. Limit	Coord. Limit + 65535

Other than using the sign of the scale, the Coordinate Limit parameter is independent of the scale parameter. Therefore, it is recommended that you choose one of the following Coordinate Limit values to allow you to use standard signed or unsigned position units. Other position ranges are non-standard and take special treatment in PLCs:

Scale	Coordinate Limit	Range	Standard
< 0	65535	0 to 65535	Unsigned
< 0	32767	-32768 to 32767	Signed
³ 0	0	0 to 65535	Unsigned
³ 0	-32768	-32768 to 32767	Signed

Translating to Position Units

The translation of quadrature counts to position units is done incrementally. That is, the change in quadrature counts is converted to a change in position units. These quantities are then added to the current Counts and Actual Position status words. Therefore, only the Scale and the Prescale Divisor bits are used in the conversion from Counts to Actual Position.

The following formula summarizes the translation from change in quadrature counts to the change in position units:

$$\Delta \text{Actual Position} = \frac{\Delta \text{Counts} \times \text{Scale}}{32768 \times \text{Prescale Divisor}}$$

The RMC ensures that no fractional position units are lost in this conversion.

Determining the correct Scale and Prescale Divisor is a three-step process. First, the exact Scale value is calculated assuming a Prescale Divisor of 1. Next, the optimum Prescale Divisor is selected. Finally, the exact Scale is multiplied by the Prescale Divisor and rounded to the nearest integer. The best way to do this is to use RMCWin’s Quadrature Calibration Utility. This utility does all of these calculations for you, plus helps you select the appropriate position unit range. If you do not choose to use this utility, follow the instructions in the steps below:

1. Calculating the Exact Scale Value

The precise scale value is calculated according to the following formula:

$$\text{Scale} = \frac{\Delta \text{Position Units} \times 32768 \times \text{Sign}}{\Delta \text{Counts}}$$

Therefore, the information required to calculate your exact scale is the number of position units you want for a given number of counts. The Sign factor determines the direction your position units move in relation to your counts; enter a 1 if increasing counts should increase the actual position, or enter a 1 if increasing counts should decrease the actual position. For example, if you have a 1000-line encoder and therefore 4000 quadrature counts per revolution, and you want to have 3600 position units per revolution, then you would do the following calculation because you want 3600 position units change when you have a 4000 quadrature count change:

$$\text{Scale} = \frac{3600 \times 32768}{4000} = 29491.2$$

One restriction you have at this point is that you cannot have more position units than counts over a given range. That is, DPosition Units must be less than or equal to DCounts. If this is not the case, you must either reduce the number of position units—usually dividing it by a factor of 10 accomplishes this—or choose an encoder that has more counts. For example, a user who wanted 36000 position units per 4000 quadrature counts would need to upgrade to a 9000-or-more-line encoder or drop to 3600 position units per revolution.

2. Selecting the Optimum Prescale Divisor

The Prescale Divisor can have values of 1, 2, 4, or 8. Because the Scale is divided by the Prescale Divisor, this feature allows a fractional scale. Pick the largest Prescale Divisor you can multiply the Scale by and still be between 32768 and 32768. For example, suppose your Scale comes to 6324.70. The following table shows the possible Scales and Prescale Divisors you could use and the effective scale:

Scale	**Divisor**	**Effective Scale**	**Error from 6324.70**
6325	1	6325/1 = 6325	0.005%
12649	2	12649/2 = 6324.5	0.003%
25299	4	25299/4 =	0.0008%

		6324.75	
50598	8	Invalid scale	Invalid

Therefore, in this example, a Scale of 25299 and a Prescale Divisor of 4 should be used. Notice that this reduced the error in the scale factor.

3. Entering the Correct Scale Value

Now that the Prescale Divisor has been calculated, you must multiply the exact Scale value you found in step one by this Prescale Divisor and round to the nearest integer. For example, if you had a scale that came to 2949.12, then you would have selected a Prescale Divisor of 8. Therefore your correct Scale parameter would be 2949.12 x 8 or 23592.96 rounded to 23593. Therefore, your effective scale is 23593 / 8 = 2949.125, which is within 0.0002% of the exact scale.

Note: If the Scale parameter value you calculate to enter is 32768, you must enter 0 instead. This is required because the range of signed 16-bit number is 32768 to 32767. Therefore, 32768 is not a valid number. We chose 0 to represent this value instead.

Example 1:

A user with a 1024-line encoder wishes to have position units in tenths of a degree of rotation on the encoder’s shaft. Therefore, the user has 4096 counts per revolution and wants to have 3600 position units over that distance. This gives us the following exact scale calculation:

$$\text{Scale} = \frac{3600 \times 32768}{4096} = 28800.0$$

Notice that the scale comes out to a whole number. This is an ideal situation, as there will be no scaling error. This will be the case when encoders are used that have line counts that are powers of two, such as 64, 256, 512, 1024, 2048, and 4096.

We choose a Prescale Divisor of 1 for the following two reasons:

- We have no choice. Any larger divisor, when multiplied by the Scale value, would overflow the acceptable range for the Scale value.
- Whenever the Scale calculated in the first step comes out to an exact number, the larger Prescale Divisors do not add any accuracy to the scale. For example, suppose this user wanted position units of degrees. Therefore, the scale would be 360 x 32768 divided by 4096, which comes exactly to 2880. Even if we used a divisor of 8, the scale value entered would be 2880 x 8, which is 23040. The effective scale is 23040/8, which comes out to 2880.

Therefore, we use the following two values:

```
Scale = 28800

Prescale Divisor = 1
```

Example 2:

A user has a 200-line encoder mounted on shaft A. Shaft A is geared 1:15 with shaft B, which does the work. Therefore, it is on shaft B that the user wishes to have position units measured in 1000ths of a revolution.

We know that the user has 1000 position units per revolution on shaft B. We need to find out how many encoder counts there will be for one revolution of shaft B. There are 800 quadrature counts per shaft A revolution. Because it will take fifteen turns on shaft A to turn shaft B once, there will be 800 x 15 or 12000 quadrature counts for each revolution of the shaft B.

Notice that we could increase the position units to be 10,000ths of a revolution and still have more counts than position units, but we will assume that the user really doesn't want to use this extra resolution. Increasing the position unit resolution would lower the maximum speed, as speeds are limited to 65535 position units per second. Therefore, we can calculate our scale:

$$\text{Scale} = \frac{1000 \times 32768}{12000} = 2730.67$$

Next, we need to choose a Prescale Divisor. The largest of the valid divisors (1, 2, 4, and 8) that can be multiplied by 2730.667 and still be within ±32768 is 8. When we multiply this divisor by the scale and round it to the nearest integer, we get the following parameters:

```
Scale = 21845
```

```
Prescale Divisor = 8
```

In this case our effective scale is 2730.625 instead of 2730.667, which means our scale has an error of 0.0015%.

See also:
Quadrature Overview
Quadrature Wiring
Quadrature Configuration
Quadrature LED Indicators
Quadrature Specifications
Quadrature Homing

6.3.8 Homing a Quadrature Axis

Quadrature encoders do not provide absolute position. To use a quadrature encoder in a position application, a known reference position must be established. This position is called the **Home** position. The process of moving the axis to find the home position is called Homing. As the encoder is rotated it increments or decrements the position from the initial home position.

Each RMC quadrature axis has two inputs that can be used for homing:

- The index (Z) input comes from the encoder to indicate that the encoder has rotated a full revolution.
- The home (H) input comes from a limit switch or proximity sensor.

Both the Z and H inputs must be active for the RMC to register an active Home. Therefore, if either the Z or H input is not used, the input must be made active either by wiring or inverting the Z or H input in the Config parameter word.

The RMC is commanded to wait for the Home condition by the Arm Home command. That RMC axis is then said to be **Armed**. Unless the quadrature axis is armed, the Home condition is ignored. If the Auto Home Re-arm configuration bit is set, the axis will automatically be re-armed after homing. Otherwise, the Arm home command must be issued to arm the axis again after homing.

Note: If the Auto Home Re-arm configuration bit is cleared after the Arm Home command has been issued, the axis will still be armed. However, once the axis is homed, it will not be automatically re-armed.

When the axis is armed and there is a falling edge in the Home active, the exact quadrature counts at the time of the falling edge are used as the physical home position. The Actual Position will be changed to reflect the current position with respect to the home position as specified in the Arm Home command. The Command Position and Target Position will be offset by the difference between the new and old Actual Positions.

The Home Status bit in the Status word can be used to help home the system. It can be configured with the Level/Edge bit in the Config word either to be latched on the falling edge of the Home active condition or to match the level of the Home Active condition.

Linear Positioner Homing Hints

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 H input to the axis is placed at one end of the travel; this will be the home position. The Z input is not used (not connected) and should be configured to be always active.

Set the Level/Edge bit in the Config word to **Level of H input** so that the status bit will be active when the H input is active. Next, arm the axis's homing by issuing an Arm Home command.

When starting from an unknown position, retract until the Home Status bit is ON. Then, extend until the Home Status bit is OFF and stop. When the Level/Edge bit is set, the falling edge on the Home Status bit indicates the axis has been homed.

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

6.4 Quadrature with Stepper Output

6.4.1 Quadrature with Stepper Output Overview

The STEP interface module is one of two RMC interface modules with an interface for quadrature encoder feedback. This module provides stepper drive output, while the QUAD interface module provides analog drive output. Together, these modules allow the RMC100 series motion controllers to control a wide range of motors and linear actuators with quadrature encoder feedback. For details on the QUAD module, see Quadrature with Analog Output Overview.

Each stepper module can control two stepper axes, each with both stepper-motor and quadrature-encoder interfaces.

The quadrature-encoder interface does not have to be used; the motor can be controlled in open loop and still take advantage of the RMC's host of target generating methods including trapezoidal and s-curve point-to-point moves, gearing, synchronization, and splines.

When used with quadrature feedback, faults can be triggered on following errors, and motor compensation can be enabled. See Stepper Compensation for details on this feature.

Features

- Two Complete Axes per Module. Each includes the following:
- Stepper Motor Interface:
    - 1 MHz Maximum Output Frequency
    - Step Output
    - Direction Output
    - Drive Enable Output
    - Drive Fault Input
- Quadrature Encoder Interface:
    - 4,000,000 counts/second Maximum Encoder Rate
    - A and B Inputs
    - Index Input with High-speed 63hs Position Latch
- Additional Inputs:
    - Extend (CW) Travel Limit Input
    - Retract (CCW) Travel Limit Input
    - Home Input with High-speed 50µs Position Latch
- Status LED
- Digital Noise Filters on All Inputs
- All Discrete Inputs are Isolated
- Control in Open or Closed Loop (with or without quadrature interface)

Step and Direction Outputs
Each axis has two outputs for controlling the stepper motor movement. These are the Step and Direction signals that connect to the stepper drive. The Step signal carries a series of pulses that tell the drive how many steps (or microsteps) the motor should move. The Direction signal controls the direction the motor moves. Both of these signals are generated by differential line drivers.

Drive Enable Output
This output to the drive allows the RMC to turn off the drive using the Amp Enable/Disable command. The enable output (labeled ENABLE + and -) is a solid state relay that is closed when enabled or active.

Vcc Output
This is a nominal 5 V output that can be used in conjunction with the Step, Direction, and Enable outputs to interface drives that require a positive common input. See Stepper Wiring for details.

Note: DO NOT use this output to power encoders.

Drive Fault Input

FAULT + and - is an input from the drive or some other source that can be set up to trigger the RMC to stop its target generator and stop generating step pulses. The user can select the active state of this input by the Fault Active State bit in the axis's Config word; by default the RMC is set-up to fault when current stops flowing through the input. This way the axis will fault on loss of control power. Bit 15 of the Status word is set if a Fault or Encoder error has occurred. The Auto Stop parameter determines how the axis will stop when a fault occurs.

Quadrature Encoder Inputs

The RMC stepper module accepts signals from quadrature encoders. The A and B signals from the encoder are decoded to generate a positive or negative count; the count is then used to monitor the axis position. Each signal from the encoder is received into a differential line receiver for compatibility with differential line driver encoders. The negative input is biased so some encoders with open collector or TTL outputs can also be used, though this is not recommended due to the possibility of electrical noise.

Index (Z) Input

The index input is active for one count of each quadrature encoder revolution. It is used to qualify or narrow the effective width of the home input to make the homing more repeatable; see Homing a Quadrature Axis for details. The user may change the active state of this input by changing the Home Active State bit in the axis's Config word.

Home (H) Input

This input may be used in homing the system; see Homing a Quadrature Axis for details. This input will be connected to a normally-off proximity switch. The user may change the active state of this input by changing the Home Active State bit in the axis's Config word.

Extend and Retract Travel Limit Inputs

The Extend Limit (CW) and Retract Limit (CCW) switch inputs tell the RMC that a travel limit has been reached and can be configured to automatically halt the axis. By default the limit switches are considered inactive when enough current is flowing through the RMC inputs. This way the limit switches will look active to the RMC if control power is lost. The user may change the active state by setting the Limit Switch Active State bit in the axis's Config word.

The Auto Stop bits must be set to automatically stop the axis based on these limit inputs.

Note: There is only one bit in the Config word for both limit switches so both limit switches must have the same active state.

If a limit switch is not used it should be wired in the inactive state or the Limit Switch Active State bit must be configured to disable the inputs.

Motor Position Compensation

When quadrature encoder feedback is used, the RMC may be configured to compensate for differences between the target and actual position of the motor. See Stepper Compensation for details.

See also:
Stepper Wiring
Stepper Configuration
Stepper LED Indicators
Stepper Specifications
Stepper Scaling
Stepper Compensation
Homing

6.4.2 Stepper Wiring

Use shielded twisted pairs for all connections to inputs and outputs. Route the quadrature encoder wiring separate from other wiring. You must provide the power supplies needed by your quadrature encoders and drives, although the RMC can provide +5 VDC for the optoisolators on the drive module as shown in the Stepper Output Wiring section below.

DB25S Pin-out

Pin	Function	Pin	Function
1	A-	14	Index (Z)-
2	A+	15	Index (Z)+
3	B-	16	Encoder Common
4	B+	17	N/C
5	N/C	18	Home-
6	Retract Lim-	19	Home+
7	Retract Lim+	20	Fault-
8	Extend Lim-	21	Fault+
9	Extend Lim+	22*	Step-
10*	Vcc Out	23*	Step+
11*	Direction-	24*	Drive Enable-
12*	Direction+	25*	Drive Enable+
13*	Drive Common		

*Outputs, all others are inputs

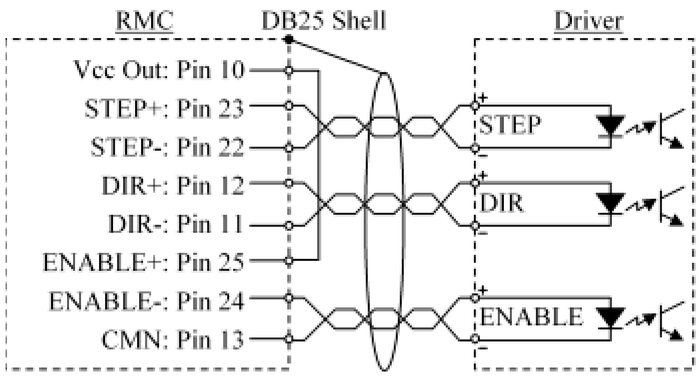
See also General Wiring Information.

Stepper Output Wiring

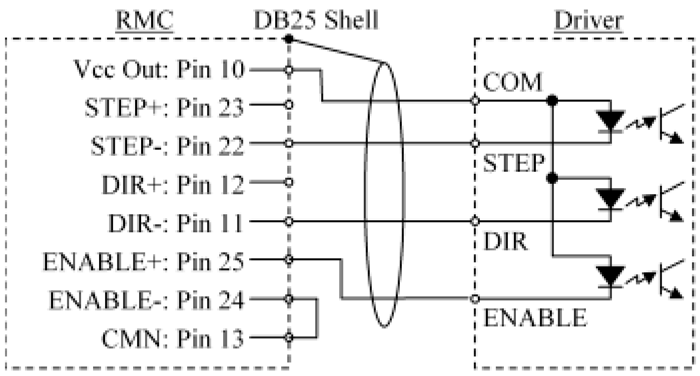
Note: Do not use Vcc Out to power your encoder because:
- This output’s voltage is not regulated.
- 50 mA is not adequate current to power the encoder. Each of the encoder outputs (A, B, and Z) requires 20 to 25 mA in addition to the current used by the encoder’s internal circuit.
- Using this output to power the encoder would violate the isolation that exists between the inputs and outputs.

Note: The STEP and DIR outputs are generated by a differential driver, but the ENABLE output is a relay. Therefore, power must be provided through the ENABLE output, as shown in the diagrams below.

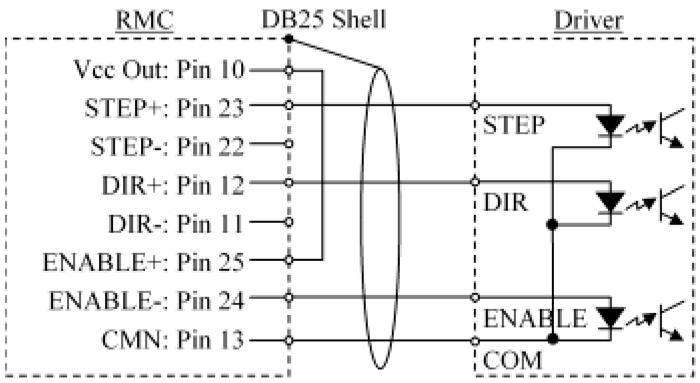
Drive with bipolar inputs:



Drive with common anode inputs:



Drive with common cathode inputs:



Input Wiring

The wiring for all inputs is identical to the wiring for the quadrature interface module with analog outputs. See Quadrature Wiring for diagrams on these inputs.

See also:
Stepper Overview
Stepper Configuration
Stepper LED Indicators
Stepper Specifications
Stepper Scaling

Stepper Compensation
Homing

6.4.3 Stepper Configuration

The RMC supports a wide range of quadrature encoders and homing configurations. The following settings should be set to match your system:

- Active state of the Index (Z), Home (H), Fault, and Limit inputs. This is set in the Configuration word. See the Quadrature/Stepper Specific Configuration help topic for details.
- How the Home status bit is used. This is set in the Configuration word. See the Quadrature/Stepper Specific Configuration help topic for details.
- How the **Encoder Error/Fault Input** status bit is used. Setting this bit is described below.
- Support for reversing the direction output in software. This is set in the Configuration word. See that topic for details.
- Maximum steps per millisecond. The Maximum Steps Per Millisecond parameter is used to limit the stepper motor from being overdriven.

No jumpers are required for changing any of these settings. Instead, set the Configuration word and Maximum Steps Per Millisecond parameter for each axis used.

Defining the Encoder Error/Fault Input Status Bit

This status bit can represent either an encoder error or a fault error. Which of these two conditions gets reported in this status bit is configurable by the user. There are four options:

- **Encoder Error only.** This bit will go high if the encoder circuitry detects an error, which is defined as an invalid transition of the A and B lines. This usually occurs due to over-speed or noise conditions.
- **Fault Input only.** This bit will go high if the Fault input goes active. The fault input is intended to be a means of the drive amplifier letting the RMC know that it no longer has control. If the fault input is not used, it should be made inactive by wiring or by reversing the polarity in the Configuration Word parameter.
- **Encoder Error or Fault Input (default).** This bit will go high if either of these conditions occurs. This is the default behavior for QUAD and STEP axes, but it is ambiguous whether the error was triggered due to an encoder error or the Fault input.
- Unused. This bit will always be low.

The use of this bit can be set only from RMCWin. Use the following steps:

1. On the Tools menu, click Module Configuration.
2. In the Slots list, click the Stepper module you want to edit.
3. Click Slot options. The Quadrature with Stepper Output Options dialog box will be displayed with a tab for each axis on that slot.

4. Click the Axis 0 tab.
5. Check the conditions that you want to have set this status bit. You can check one, both, or neither condition.
6. Click the Axis 1 tab.
7. Check the conditions that you want to have set this status bit. You can check one, both, or neither condition.
8. Click Update RMC.
9. The Update Module Configuration dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module yourself.

See also:
Stepper Overview
Stepper Wiring
Stepper LED Indicators
Stepper Specifications
Stepper Scaling
Stepper Compensation
Homing

6.4.4 Stepper LED Indicators

The two LEDs above the Drive connector on the RMC's Stepper interface card are axis status LEDs. These LEDs will reflect the operational status, as determined by the Status Word, of their corresponding axis according to the following table.

LED Action	**Axis Status**
Alternating Red/Green	One or more of the following status bits are on and are enabled in the Auto Stop: Encoder Fault Limit Switches Home Input Overdrive Error Parameter Error Compensation Timeout Following Error
Continuous Green	Status good. The above is not true.

Note: Prior to RMC CPU firmware dated 19991216, the Auto Stop parameter was not used in determining the LED states. Therefore, the only way to keep the LED from showing red was to clear the error.

See also:
Stepper Overview
Stepper Wiring
Stepper Configuration
Stepper Specifications
Stepper Scaling
Stepper Compensation
Homing

6.4.5 Stepper Specifications

For general specifications on the RMC, see RMC100 Specifications.

Encoder

Axes	Two per module
Encoder Inputs	RS422 differential receiver Quadrature A, B, and Index Z
ESD Protection	15 kV
Max. Encoder Frequency	4,000,000 counts/second
Index (Z) Response Time	63 nanoseconds

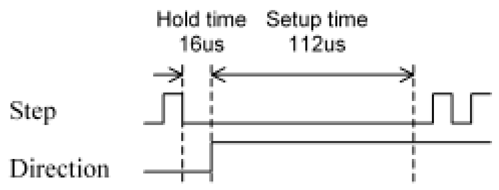
Inputs and Outputs

Fault Inputs, Home Inputs, and Limit Inputs (Ext. & Ret.)	2.7 V @ 2.8 mA typical (3.2 V @ 3.5 mA max) threshold, 26.4 V maximum input voltage, 500 VDC isolation, compatible with most limit switches, TTL, and CMOS outputs
Drive Enable Output (1 per axis)	Solid State relay, 50 W, 30 V, 100 mA 500 VDC isolation
Vcc Out	5 V ±20% @ 50 mA for use with Step, Direction, and Enable outputs. DO NOT power encoders with this output. See Stepper Wiring for details.

Motor Interface

Step and Direction Outputs	RS422 (5 V differential)
Pulse Width	50% duty cycle ± 80 nanoseconds
Max. Output Frequency	1 MHz

Direction Change Delays	Hold time = 16 us Setup time = 112 us plus half step period



See also:
Stepper Overview
Stepper Wiring
Stepper Configuration
Stepper LED Indicators
Stepper Scaling
Stepper Compensation
Homing

6.4.6 Stepper Scaling

Defining the Valid 16-bit Position Range

For general scaling information, see the Scaling Overview topic.

Because the RMC uses 16-bit positions, positions must all fit within a range of 65,536 position units. Because position units are user definable, and speed control and indexing applications can get around this limitation, most applications are not limited by this range. See the section below on defining position units.

Stepper axes use the Coordinate Limit parameter to define the valid range of position units. The valid Coordinate Limit values extend from 65535 to 0, and the valid position range then extends from the Coordinate Limit value to the Coordinate Limit plus 65535. Delta recommends using one of the following two values and ranges:

Coordinate Limit	**Range**
0	0 to 65535
-32768	-32768 to 32767

The first uses unsigned position units and the second uses signed position units. Other position ranges require extra work to use with PLCs.

Translating to and from Position Units

The method of setting up the scale for a stepper axis with quadrature feedback differs significantly from the scaling of other axis types on the RMC. This was done to better fit the needs of stepper applications.

The stepper module has two scales. One scale defines the relationship between incoming quadrature counts and the resultant Actual Position. The second scale defines the relationship between the Target Position and the outgoing steps.

Each scale is defined as a ratio of two numbers. The user can enter both numbers in each of the ratios, although one number is shared by both ratios. Here is the definition of the two scales:

$$\frac{\Delta \text{Steps}}{\Delta \text{Target Position}} = \frac{\text{Steps/Rev}}{\text{Pos Units/Rev}}$$

$$\frac{\Delta \text{Actual Position}}{\Delta \text{Quad Counts}} = \frac{\text{Pos Units/Rev}}{\text{Quad Counts/Rev}}$$

The three values the user can enter are Steps/Rev, Pos Units/Rev, and Quad Cnts/Rev. These are parameters for each stepper axis. You will notice that both scales use Pos Uints/Rev.

The RMC allows the user to select the direction of both the stepper and quadrature counts with respect to position units. That is, phase A leading phase B in the quadrature can mean increasing or decreasing actual positions. Similarly, a low direction line can be the result of increasing or decreasing target positions.

The sign of the Quad Counts/Rev parameter controls the quadrature direction per the following chart:

Quad Counts/Rev	**The actual position increases when…**
+	…phase A leads phase B
-	…phase B leads phase A

The Reverse Drive bit of the Configuration Word controls the use of the direction output to the stepper motor per the following chart:

	When the target positions…	
Reverse Drive bit	**…increase…**	**…decrease…**
0	…Direction will be low.	…Direction will be high.
1	…Direction will be high.	…Direction will be low.

These scales are used to determine the change in steps from the change in target position and to determine the change in actual position from the change in quadrature counts according to the following formulas:

$$\Delta \text{Steps} = \Delta \text{Target Position} \times \frac{\text{Steps/Rev}}{\text{Pos Units/Rev}}$$

$$\Delta \text{Actual Position} = \Delta \text{Quad Counts} \times \frac{\text{Pos Units/Rev}}{\text{Quad Counts/Rev}}$$

Note: The RMC ensures that no fractional position units or steps are lost in these conversions. That is, even though the result of this equation will likely not be a whole number, the remainder is carried from scan to scan.

Note: If quadrature feedback is not used, enter a 0 in QUAD CNTS/REV. This eliminates the Following Error from being generated.

There are several of ways that these scales can be set up. The following examples illustrate the potential uses of these scale parameters.

Example 1:
The user has a 100-line encoder, a 1.8°-per-step (200 steps per revolution) motor, and a stepper drive configured to use half steps. The encoder is mounted on the same shaft as the stepper motor, so each will turn one revolution in the same amount of time.

The user chose these devices so that the number of quadrature counts per revolution is equal to the number of half steps per revolution, and wishes to use the raw counts as position units. That is, because the stepper drive uses half steps there are 400 steps per revolution to match the 400 quadrature counts per revolution on the 100-line encoder.

Therefore, the ratio of steps to counts to position units is 1:1:1. The following parameters achieve this:

Parameter	**Value**
Steps/Rev	400
Position Units/Rev	400
Quad Counts/Rev	400

Example 2:
The user has a 256-line encoder, a 7.5°-per-step (48 steps per revolution) motor, and a full-step stepper drive. Therefore, the encoder generates 1024 quadrature counts per revolution. The encoder is mounted on the same shaft as the stepper motor, so each will turn one revolution in the same amount of time. The user wants positions to be given in degrees, so we will have 360 position units per revolution.

In this example, we simply enter the following values direct from our calculations above:

Parameter	**Value**
Steps/Rev	48
Position Units/Rev	360
Quad Counts/Rev	1024

Therefore, each quadrature count will affect the actual position by 360/1024 of a position unit. Each target position unit causes 48/360 of a step. Notice that this means that it takes 7.5 position units to get one step output. This would need to be taken into consideration when the Following Error Window parameter is set.

Example 3:
Suppose we have the same motor, drive, and encoder as in the previous example: the stepper has 48 steps per revolution, and the encoder has 256 lines. However, this time the stepper motor is on shaft A and the encoder is on shaft B. Shaft A must turn 15 revolutions in order to turn shaft B one revolution; the gear ratio is 1:15. The user wishes to measure positions in degrees on shaft B.

Therefore, we define a revolution as one turn of shaft B. This means we have 1024 quadrature counts per revolution of our 256-line encoder, and 360 position units per revolution. Calculating the number of steps per revolution of shaft B requires multiplying by the gear ratio between shaft A to shaft B:

$$\frac{\text{Steps}}{\text{Shaft B Revolution}} = \frac{\text{Steps}}{\text{Shaft A Revolution}} \times \frac{\text{Shaft A Revolutions}}{\text{Shaft B Revolutions}}$$

$$\frac{\text{Steps}}{\text{Shaft B Revolution}} = 48 \times 15 = 720$$

Therefore, our parameters should be as follows:

Parameter	**Value**
Steps/Rev	720
Position Units/Rev	360
Quad Counts/Rev	1024

Therefore, each quadrature count will affect the actual position by 360/1024 of a position unit. Each target position unit causes 720/360 of a step or 2 steps. Therefore, half of a position unit causes one step.

Suppose that when these parameters are entered, it is found that the stepper moves in the correct direction, but the actual positions go backwards. This means that the count to position unit scale needs to be reversed. As described above, reversing the sign of the Quad Counts/Rev parameter changes the direction of this scale. Therefore, we would use the following parameters:

Parameter	**Value**
Steps/Rev	720
Position Units/Rev	360
Quad Counts/Rev	-1024

Example 4:

Suppose we have a stepper motor with 1.8° per step or 200 steps per revolution, and its drive is configured to have 25 microsteps per step. Therefore, the RMC will need to give 5000 steps to turn the motor one revolution. A 1000-line encoder is on the same shaft, which generates 4000 quadrature counts per revolution.

The motor moves a belt, and the user wants positions in inches that the belt moves. Therefore, we need to be able to determine the distance the belt moves per motor revolution. This depends on the circumference of the drive wheel. If we assume the radius of this wheel is 2.5 inches, then the circumference is 2 x p x 2.5 inches or 15.70796 inches.

Given this, a first attempt might be to enter the nearest integer in the POS UNITS/REV field, so our parameters would be:

Parameter	**Value**
Steps/Rev	5000
Position Units/Rev	16
Quad Counts/Rev	4000

There are two problems with this solution. First, the user sees the position only in inches; no fractions of an inch will be displayed. Second, the distance reported will be off by 1.825% or 0.292” per revolution, which is unacceptable in most applications, especially those involving multiple motor turns.

We can improve both of these problems by using hundredths of an inch as our position units. Since we have about 1571 hundredths of an inch per revolution, we could use the following parameters.

Parameter	**Value**
Steps/Rev	5000
Position Units/Rev	1571
Quad Counts/Rev	4000

This eliminates the first problem if a hundredth of an inch is an adequate resolution. It also reduces the position error to 0.013% or 0.002” per revolution. This is a major improvement and may be adequate for many applications.

We can further improve our scale by maximizing the dynamic range of these three parameters. That is, since these parameters are used as ratios, we can multiply each by a constant without changing the effective scales. Any multiplier—integer or non-integer—is acceptable as long is at lowers the scaling error and doesn’t exceed the parameter limits: Steps/Rev and Position Units/Rev are limited to 65535, and Quad Counts/Rev is limited to ±32767. A good value to multiply by is 5. The Position Units/Rev parameter becomes 15.70796 x 100 x 5, which is 7853.98 and gets rounded to 7854. Our new parameters are:

Parameter	**Value**
Steps/Rev	25000
Position Units/Rev	7854
Quad Counts/Rev	20000

Because we multiplied these parameters by 5, the Position Units/Rev parameter is equivalent to 7854/(5 x 100) or 15.70800 inches per revolution. This gives a scaling error of 0.0002% or 0.000037” per revolution. This is effectively the best we can do with our scaling precision for this application.

Two questions remain: What is the effect of this remaining scaling error? What are the limitations of the 16-bit positions? Each is discussed in turn below.

What is the effect of the remaining scaling error?

The remaining scaling error must be carefully evaluated for your system. In many cases it may be adequate, or perhaps the mechanical error overshadows the remaining scaling error. For example, the tolerance of the radius of the driving wheel may result in a greater error than the scaling error.

However, in many indexing applications the error for a single cycle is acceptable, but over many cycles, the error accumulates until it becomes significant. If this is the case, then you may need to use the home input or index pulse to reset the positions between—or even during—each cycle or set of cycles. See Quadrature Homing for details on this topic.

If the scaling error still appears too large, contact a Delta Computer Systems Application Specialist to discuss your system. See Technical Support for details on contacting Delta.

What are the limitations of the 16-bit positions?

Because positions are stored in 16-bits, the RMC can neither display positions greater than 65535 position units nor receive commands to move more than 65535 position units. In our application, we have position units equal to a hundredth of an inch, so this limitation is 655.35 inches (over 54 feet). This won’t limit most linear applications, but applications requiring longer distances will need to use a lower position unit resolution. For example, using tenths of an inch gives a range of 6553.5 inches, and using inches gives a range of 65535 inches. However, this usually increases the scaling error.

In indexing applications, the range limitation of 655.35 inches may become a problem in another way. Suppose that one cycle moves only 6 inches and this cycle is repeated 10,000 times per day. Therefore, in one day the total movement will be 60,000 inches, which is beyond our maximum of 655.35 inches. The proper way to deal with this problem is to reset the position at the beginning or end of each cycle, using either quadrature homing or the Zero Position/Set Target or Offset Positions commands. Therefore, the positions would only go as high as 600 (6.00 inches) before being reset back to 0 (0.00 inches).

See also:
Stepper Overview
Stepper Wiring
Stepper Configuration
Stepper LED Indicators
Stepper Specifications
Stepper Compensation
Homing

6.4.7 Stepper Compensation

Most stepper motors are used without feedback. They are homed occasionally, but are otherwise expected to go to the position to which they are commanded. However, some applications require verifying that the load is in the correct position. There are three main reasons why the load does not reach or maintain the commanded position:

1. There is backlash in the gearing between the motor and load.
2. The load gets pushed or bumped with a force large enough to overcome the motor torque.
3. The motor is commanded to move faster than it is capable of moving.

The method of compensation for reasons 1 and 2 is very different from the method required for 3. The RMC currently will only compensate for 1 and 2. It is left to the user to make sure that the requested speeds and accelerations are not beyond the motor's capabilities.

How the Compensation Works

Three parameters control the compensation:

- Compensation Rate
  This parameter determines the speed at which the axis will try to return to the target position. If the axis is in motion, this rate will be added to the target speed. The compensation rate is only added when the actual position is outside the Compensation or In Position Window as described below. To disable compensation, enter a 0 for this parameter.

- Compensation Window
  When the axis is not moving, the compensation is applied only when the axis is outside the In Position Window parameter. When the axis is moving, the compensation rate is applied when the actual position differs from the target position by more than the Compensation Window parameter. This window should normally be set to a value greater than the In Position Window, but less than the Following Error Window.

- Compensation Timeout

When the axis is stopped outside the In Position Window, compensation will be applied to try to move the it back inside the window. If the axis is not able to move back into the In Position Window in the time specified by the Compensation Timeout parameter, the Timeout bit will be set in the Status word. This bit can be used to trigger a hard stop on the axis, as controlled by the Auto Stop Mask parameter. Do not use the soft stop with this bit, since the axis is already stopped, so the closed loop stop initiated by the soft stop does nothing.

The timeout is only active when the axis is stopped or geared to an axis that is stopped. It does not take effect while moving since the Following Error Window will catch problems at this stage.

A Compensation Timeout of 0 will disable the timeout feature so the Timeout bit will always be off.

Several modes of compensation can be configured using these three parameters:

- **Disabled**
  To entirely disable compensation, enter a zero (0) Compensation Rate. The Compensation Window parameter will then have no effect. However, the Compensation Timeout parameter can still be used to turn on the Timeout bit if the axis is not In Position within a specified number of seconds of the Target Position reaching the Command Position.

- **Disabled When Moving, Active When Stopped**
  To disable compensation while moving, enter a non-zero Compensation Rate and a large Compensation Window (for example, 65535). You can set the Compensation Timeout parameter as desired. This effectively disables compensation while moving because the error will always be within the Compensation Window, and therefore the Compensation Rate will never be added to the target speed. While stopped, the In Position Window is used instead of the Compensation Window, and therefore compensation will work normally.

- **Active when Moving and Stopped**
  Enter a non-zero Compensation Rate. Enter a Compensation Window between the In Position Window and Following Error Window parameters. Enter a Compensation Timeout to fit your needs.

See also:
Stepper Overview
Stepper Wiring
Stepper Configuration
Stepper LED Indicators
Stepper Specifications
Stepper Scaling
Homing

6.5 Resolver

6.5.1 Resolver Overview

Resolvers are absolute rotary position transducers. They are very simple and robust, can be very accurate, and, in some cases, inexpensive. A wide variety of resolvers are available in many configurations and for various applications.

How Resolvers Work

Resolvers are rotary transformers with one primary winding and two secondary windings. The primary winding is generally on the rotor and the two secondary windings are on the stator. The secondary windings are arranged 90 degrees from each other such that when one is lined up with the rotor winding (full coupling) the other is at a right angle (no coupling).

The primary winding is driven with an alternating current signal at a specified voltage and frequency. The position measurement is determined by the ratio of the amplitudes of the signals on the secondary windings and their phase with respect to the primary signal.

The RMC100 Resolver Interface Module

Resolvers are commercially available in many varieties with different specifications. The primary specifications of interest as applied to the RMC are:

a. Frequency
The frequency of the signal driving the primary winding.

b. Voltage
The specified voltage of the signal driving the primary winding.

c. Output Voltage
The output voltage from the secondary windings or the transformation ratio between primary and secondary windings.

The RMC resolver interface card is designed to work with only a subset of all resolver types available. The Resolver Specifications topic lists the range of values compatible with the standard RMC Resolver module. In addition, a wider range of resolvers can be used by modifying components on the interface card module or by using external reference generators to drive the primary winding. Contact Delta for more details.

Determining Absolute Position

There are several conditions when the resolver axis establishes its absolute position within one turn of the resolver. These conditions are:

- On power-up
- After a transducer fault
- After a Reset Position command
- After a Set Parameters command if the Scale or Coordinate Limit parameters have changed

The axis determines its absolute position as follows.

1. The 16-bit value read from the resolver is added to the Count Offset
2. If the result is outside the range of -32768 to +32767 then 65536 is added or subtracted to force it into the range. (This is the range that can be represented by a 16 bit number.)
3. This value is converted to Position Units:

Position Units = Scale x Counts / 32768

This gives a value between -Scale and +Scale

4. The result is compared with the Coordinate Limit to determine if it is within the allowable range of Position Units. (See the Resolver Scaling topic for the definition of valid range). If it is outside the valid range then Scale x 2 is added or subtracted to bring it back within range.

Useful Commands

In addition to the typical motion commands, the following commands are useful for resolver axes, especially for resetting the position when used in rotational applications:

Zero Position/Set Target Command

Reset Position Command

Offset Positions Command

See also:

Resolver Configuration

Resolver LED Indicators

Resolver Wiring

Resolver Scaling

Resolver Specifications

Rotational Mode

6.5.2 Resolver Wiring

Note: When positive voltage is sent to an axis's drive, the axis must extend. The extend direction is defined as the direction that causes the transducer to return increasing counts.

Use shielded twisted pairs for all connections to inputs and outputs. Route the resolver wiring separate from other wiring.

RMC Resolver Input

Eight-Pin Plug-in Terminal Block

Pin	Function
R1	Reference Output +
Ref In	Reference In (normally not used)
R3	Reference Ouput -

S1	Sine Input +
S3	Sine Input -
S2	Cosine Input +
S4	Cosine Input -
Case	Controller chassis ground (shield)

Note: The RMC case must be mounted to a grounded panel or otherwise grounded to avoid Transducer Faults when operating at the 16-bit resolution setting.

RMC Drive Outputs

Four-Pin Plug-in Terminal Block

Pin	Function
1	Axis 0 Drive
2	Drive Common
3	Axis 1 Drive
4	Case

When positive voltage is sent to an axis's drive, the axis must extend. The extend direction is defined as the direction that causes the transducer to return increasing counts.

CAUTION: If the outputs from the RMC are reversed, the axis will be uncontrollable when power is connected. Confirm that your wiring is correct!

See also:

General Wiring Information

Resolver Overview

Resolver Configuration

Resolver LED Indicators

Resolver Specifications

Resolver Scaling

6.5.3 Resolver Configuration

The resolver interface must be configured properly to work with your resolver. To configure it, use the Module Configuration dialog:

1. On the Tools menu, click Module Configuration.
2. In the Slots list, click the Resolver module you want to edit.
3. Click Slot Options.
   The dialog displayed will offer three tabs. See the descriptions of each below. After you have chosen your settings, click Update RMC.
4. The Update Module Configuration dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module yourself.
5. In the RMC Configuration dialog box, click Close.

Reference Output tab:

Use this tab to configure the Reference Output that will be generated for the resolvers. This reference is applied by the Reference Out + and Reference Out – pins on the connector.

- Frequency:
  Your resolver specification sheet should indicate the design frequency for your resolver. The RMC resolver interface supports frequencies between 800 Hz and 5000 Hz.
  If your resolver specifies a frequency outside the range available you can try using the closet frequency available. In general it is OK to operate resolvers at frequencies higher than the specified frequency. Operating at frequencies below the specified frequency can cause problems due to saturation or incorrect impedance.
- Amplitude:
  Your resolver specification sheet should indicate either a Transformation Ratio or the Input and Output Voltages. Use the radio button to select the type of information you will use to configure the interface and then enter the appropriate information in the appropriate edit boxes.
  Notice that the RMC expects a 2 Vrms signal from the resolver, and this may not match up with your resolver’s specification. This is generally OK. RMCWin will use the information you supplied to calculate the correct reference output voltage to obtain 2 Vrms back from the resolver. Some resolvers are specified for operation at 11 or 26 Volts. These resolvers will generally still work well at 2 Volts, but there is some potential loss of accuracy and the signal-to-noise ratio will be reduced. Contact Delta Computer Systems, Inc. to discuss options for your application.

Axis0 and Axis1 tabs:

Use these tabs to configure the Count Offset for each axis. The Count Offset is used to align the zero point on the resolver with the machine zero without having to mechanically adjust the machine. You can also set this value using the Zero Position/Set Target Command. If you use the Z command you will need to save the value to non-volatile memory using the Update Flash Command after you have completed the setup, or it will be lost when power is removed from the RMC.

See also:

Resolver Overview

Resolver LED Indicators

Resolver Wiring

Resolver Scaling

Resolver Specifications

Rotational Mode

6.5.4 Resolver LED Indicators

The two LEDs above the Drive connector on the RMC's Resolver interface card are axis status LEDs. These LEDs will reflect the operational status of their corresponding axis according to the following table.

LED Action	**Axis Status**
Continuous Red	No Transducer
Alternating Red/Green	The above error has not occurred, and one or more of the below errors have occurred and are enabled in the Auto Stop: Overdrive Error Parameter Error Position Overflow Integrator Windup Following Error
Continuous Green	Status good. None of the above are true.

See also:

Resolver Overview

Resolver Configuration

Resolver Wiring

Resolver Specifications

Resolver Scaling

6.5.5 Resolver Specifications

For general specifications on the RMC, see RMC100 Specifications.

Resolver Interface

Axes	Two per module
Reference Frequency	800 Hz to 5 kHz
Reference Output Voltage	1.41 to 4.8 V RMS
Reference Output Current	28 mA max
Resolver Transformation Ratio	0.42 to 1.41
Resolution	14 or 16 bits
Maximum Speed	3000 RPM at 14 bits and 600 RPM at 16 bits
Maximum Acceleration	1200 RPS per second at 14 bits and 60 RPS per second at 16 bits
Accuracy	4 Minutes +1 LSB

Drive Outputs

Range	±10 V, 5 mA maximum (For current drive, use the VC2100 and VC2124 accessory: ±10 mA to ±200 mA in 10 mA steps)
Resolution	14 bits
Output isolation	500 VDC, optically isolated

Environment

Operating temperature	+32 to +140 °F (0 to +60 °C)
Storage temperature	-40 to +185 °F (-40 to +85 °C)
Agency compliance	

See also:

Resolver Overview

Resolver Configuration

Resolver LED Indicators

Resolver Wiring

Resolver Specifications

Resolver Scaling

6.5.6 Resolver Scaling

Defining the Valid 16-bit Position Range

For general scaling information, see the Scaling Overview topic.

Because the RMC uses 16-bit positions, positions must all fit within a range of 65,536 position units. Because position units are user definable, this range does not limit most applications. See the section below on defining position units.

For resolver axes, the Coordinate Limit parameter is used with the Scale parameter to define the position range. If the Scale parameter is positive, then the position range extends from the Coordinate Limit value (which would normally be negative) up to the Coordinate Limit + 65535. If the Scale parameter is negative, then the position range extends from the Coordinate Limit value (which would normally be positive) down to the Coordinate Limit – 65535. The following chart summarizes this concept:

Scale	Min. Position	Max. Position
^{3} 0	Coordinate Limit	Coordinate Limit + 65535
< 0	Coordinate Limit – 65535	Offset

Translating to Position Units

Before the translation from counts to position units begins, the Count Offset is applied to the counts. This means that the value of the Count Offset is subtracted from the Counts before they are displayed or processed in any way. For a full discussion on the use of the Count Offset setting, see Resolver Configuration.

The following formula summarizes the translation from transducer counts to Actual Position units:

$$\text{Actual Position} = \frac{(\text{Resolver Counts - Count Offset}) \times \text{Scale}}{32768}$$

As described above, the Count Offset configuration setting has already been subtracted from the Counts field before using this formula. If the Actual Position overflows the valid 16-bit position range due to excessive travel of the axis, the position will wrap around to the opposite extreme shown in the table of the minimum and maximum positions above.

Notice that the Scale can hold values from -32768 to 32768, but 16-bit signed numbers are limited to -32768 to 32767. Therefore, enter a scale of 32768 as 0; the RMC will interpret it as 32768.

Calculating the Scale and Count Offset

Follow these steps to calculate the Scale and Count Offset parameters:

1. Determine the Scale

Each revolution of the resolver generates 65536 counts. To calculate the Scale parameter, determine how many position units there are in one turn of the resolver.

$$\text{Scale} = \frac{\text{Position Units per Turn}}{2}$$

The sign of the Scale is positive if the counts and the position units increase and decrease together. The sign is negative if the counts and position units move in opposite directions.

You can use the Resolver Calibration Tool to assist with this step. To access it, on the Tools menu, click Resolver Calibration Tool.

2. Select the Coordinate Limit
   In most cases, the Coordinate Limit can be set to -32768 (for a positive scale) or 32767 (for a negative scale). This will allow position units to go from -32768 to 32767, which is the same range as most PLCs use for 16-bit numbers.

You can pick other values for the coordinate limit if you wish to specify a different range of valid position units.

You can use the Resolver Calibration Tool to assist with this step. To access it, on the Tools menu, click Resolver Calibration Tool.

3. Set the Count Offset
   Move the axis to a known position and issue a Z command with the current desired position for the Command Value. This will change the Count Offset parameter such that the Actual Position will indicate the current position.

The Count Offset value must be stored in non-volatile memory in order to read the correct position the next time the RMC powers up. Use the Update Flash command to store the parameters to non-volatile memory.

See also:

Resolver Overview

Resolver Configuration

Resolver LED Indicators

Resolver Wiring

Resolver Specifications

6.6 SSI

6.6.1 SSI Overview

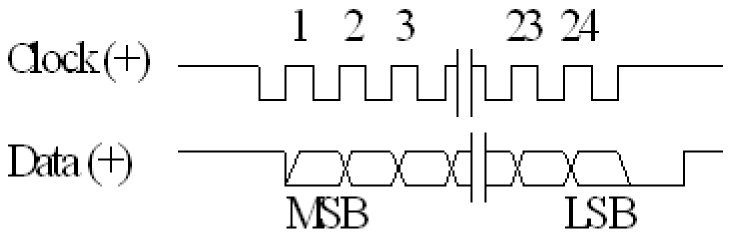
Synchronous Serial Interface (SSI) is a widely accepted controller interface. Position data from the sensor is encoded in a binary or Gray Code format and transmitted over a high-speed synchronous interface.

SSI transducers and absolute encoders offer the following advantages:

- High resolution. Up to 2 µm (approx. 0.00008")
- Noise immunity
- Cost effective data transfer (only one 6-wire cable is needed)

- Transmission rate independent of data length and resolution
- Transmission over long distances
- Direct connection to the RMC's SSI interface module

Each RMC100 SSI interface module has circuitry for two SSI transducers. Each axis can be configured independently for different types of SSI transducers. To read an SSI position, the RMC sends clock pulses to the transducer, and on each rising edge of the +Clock signal, the SSI transducer places one bit of the digital position on the +Data and Data signals. The most significant bit is sent first. Below is shown a 24-bit SSI reading:



SSI Transducer

The RMC must then convert the counts returned from the transducer to an Actual Position in user-defined Position Units (usually 0.001 inch or 0.1 mm) for use in the PID control loop.

Synchronized SSI for MDTs

For motion control, MDT (Magnetostrictive Displacement Transducer) type transducers with SSI output **must** be of the Synchronized type. This ensures that the time between position samples matches the control loop time of the RMC controller. If the transducer is not synchronized, the sample time may not match and will adversely affect control. Make sure to specify that the transducer be of the synchronized type.

Synchronized SSI is not an issue for encoders.

See also:

SSI Wiring

SSI Configuration

SSI LED Indicators

SSI Specifications

6.6.2 SSI Wiring

Note: When positive voltage is sent to an axis's drive, the axis must extend. The extend direction is defined as the direction that causes the transducer to return increasing counts.

Use shielded twisted pairs for all connections to inputs and outputs. Route the transducer wiring separate from other wiring. You must provide the power supplies needed by your transducers.

RMC SSI Input

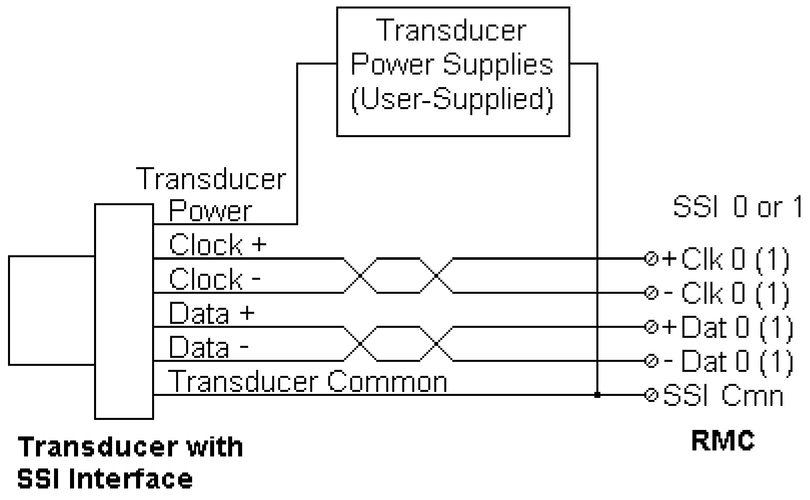
Six-Pin Plug-in Terminal Block

Pin	Function
1	SSI Axis + Clock
2	SSI Axis - Clock
3	SSI Axis Common
4	SSI Axis + Data
5	SSI Axis - Data
6	Case

SSI uses differential line driver (RS422) clock and data signals. Connect both the +Clock and Clock between the RMC and transducer for the clock signal, and both the +Data and Data between the RMC and transducer for the data signal.

The power (+24V) and ground is not provided by the RMC, and therefore must be provided externally to the transducer.

Connect the transducer DC ground to SSI Cmn.



RMC Drive Outputs

Four-Pin Plug-in Terminal Block

Pin	Function
1	Axis 0 Drive
2	Drive Common
3	Axis 1 Drive

4	Case

When positive voltage is sent to an axis's drive, the axis must extend. The extend direction is defined as the direction that causes the transducer to return increasing counts.

CAUTION: If the outputs from the RMC are reversed, the axis will be uncontrollable when power is connected. Confirm that your wiring is correct!

See also:

General Wiring Information

SSI Overview

SSI Configuration

SSI LED Indicators

SSI Specifications

6.6.3 SSI Configuration

The RMC supports a wide range of SSI transducers. The RMC offers the following configurable settings to accommodate your SSI feedback device:

- Support for Binary or Gray Code output formats.
- Support for data lengths from 8 to 25 bits.
- Support for offsetting the SSI counts. This allows the user to have a set number of counts subtracted from the SSI counts before the counts are used for control. This allows the 16-bit RMC controller to control virtually any 65,536-position unit range on an SSI device.
- Support for Balluff's additional error checking bit.
- Support for reversing the drive output in software.

All of these settings, except for the drive-reverse option must be configured up-front from RMCWin. The drive-reverse option is set in the Configuration word axis parameter. Refer to that topic for details.

Selecting SSI Configuration Options

The SSI Configuration dialog is used to configure the RMC to match your SSI feedback device. To display this dialog:

1. On the Tools menu, click Module Configuration.
2. In the Slots list, click the SSI module you want to edit.
3. Click Slot Options.

The dialog displayed will offer two tabs; one for each axis on the SSI module. See the option descriptions below. To change the SSI configuration:

1. Click the Axis 0 tab.

2. Set all options to your desired settings.
3. Click the Axis 1 tab.
4. Set all options to your desired settings.
5. Click Update RMC.
6. The Update Module Configuration dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module yourself.
7. In the RMC Configuration dialog box, click Close.

Data Length

SSI devices may be purchased which provide a specific number of bits of data. Some common values include 13, 24, and 25 bits of data. Set the data length field to match your device.

Certain SSI devices have a number of bits but pad it with a zero, which means the total Data Length must be one greater than the manufacturer's stated data length.

If the most significant bit from the SSI feedback is set, the RMC100 will see the value as a negative number and will report a transducer overflow. The counts will go to zero.

Output Format

SSI devices may be purchased which use either Gray code or binary data output. Although the RMC supports both, binary is preferred. Gray code is generally used internally by most absolute rotary encoders, but it offers no advantage when the data is transmitted to the RMC over the SSI interface. More importantly, there are some cases where using Gray code can interfere with transducer error detection. Under Output Format, click the appropriate option to match your device.

Error Method

Choose the method of indicating an overflow error on the transducer.

- **Bit 20**
  If bit 20 in the SSI data turns on, or if the SSI data bits are all zeros, the Overflow bit in the Status word will turn on. While the SSI standard does not specify the use of an overflow bit, Balluff does provide an overflow bit (bit 20) on some of its BTL-5 series of transducers. When no magnet is detected, the Balluff transducer sets this bit to indicate an error.

- **All Zeros**
  If the SSI data bits are all zeros, the Overflow bit in the Status word will turn on. Most linear SSI transducer manufacturers set the SSI data bits to all zeros if a magnet is not detected.

Count Offset

The RMC imposes a restriction that all positions must be specified as a 16-bit value in position units. A position unit is a user scaled value and therefore is quite flexible. However, there are limitations to this situation. Consider the following example:

A user is using a 2m-long, 5µm-resolution transducer, and wishes a position unit to equal 10 µm. Therefore, the scale is set up so that 1 (one) position unit equals 2 (two) counts.

The user wishes to control the positions beginning at 1500 mm and ending at 2000 mm on the transducer. This would result in position units of 150,000 and 200,000. Because the maximum value for a 16-bit number is 65,535, this would not work. This is where the Count Offset is used.

The number of counts at the starting and ending positions are 300,000 and 400,000. However, because there will be no controlling taking place before 300,000 counts, we can immediately subtract 300,000 counts from the position and act as though the counts received are 0 through 100,000. Therefore, the position units are between 0 and 50,000, which fits within the 16-bit, 65,535 limit.

The above example should describe the need for this field. This field is used in steps of 256 counts. That is, you may subtract off counts of 0, 256, 512, etc., but not values in between. This is not a significant limitation because precise scaling and offsetting can be done through the Scale and Offset axis parameters.

To calculate your desired Count Offset, do the following:

1. Determine the absolute-minimum count value you will control.
2. Divide this count by 256.
3. Discard the fractional portion.
4. Enter this value into the Count Offset field.

In the example above, the following values would be used.

1. The absolute-minimum count value to be controlled is 300,000.
2. Dividing 300,000 by 256 yields 1171.875
3. Discarding the fractional portion of 1171.875 yields 1171.
4. Entering 1171 in Count Offset results in a value of 299,776 being used as the actual Count Offset.

Configuring Balluff SSI Transducers

As of this printing, Balluff has two generations of transducers that support SSI: BTL-3 and BTL-5. The part numbers both begin with the format of BTLa-S1bc… (e.g. BTL3-S102), where the lower-case letters are described below.

Note: It is highly recommended that a Synchronized transducer be selected for motion control. BTL5s can be purchased as synchronous by specifying a B in the part number. For example, BTL5-SxxxB-Mxxx-x-xxxx.

a: Generation

3 = Generation 3

5 = Generation 5

This setting does not affect the configuration of the RMC.

b: Code Format

0 = Binary increasing (24 bits)

1 = Gray Code increasing (24 bits)

2 = Binary decreasing (24 bits)

3 = Gray Code decreasing (24 bits)

6 = Binary increasing (25 bits)

7 = Gray Code increasing (25 bits)

8 = Binary decreasing (25 bits)

9 = Gray Code decreasing (25 bits)

Use the Binary/Gray Code and Data Length fields in the SSI Configuration dialog to match the code format of the BTL transducer. The increasing/decreasing option indicates whether the counts increase or decrease away from the head of the transducer. This does not affect the SSI Configuration dialog, but will need to be taken into account when setting the Scale.

c: System Resolution

1 = 1 µm

2 = 5 µm

3 = 10 µm

4 = 20 µm

5 = 40 µm

7 = 2 µm

This setting does not affect the SSI Configuration dialog, but must be taken into account when setting the Scale.

Configuring MTS SSI Transducers

MTS's Temposonics® III family of transducers uses SSI. These transducers may be purchased with a number of options. These options are indicated by the last seven digits of the part number, in the format of Sabcdef (e.g. RH-T-0360U-RG0-1-S2B1102), where the lower-case letters each indicate an option, as described below:

a: Data Length

1 = 25 bits - Set the Data Length field to 25.

2 = 24 bits - Set the Data Length field to 24.

b: Output Format

B = Binary - Select the Binary output-format option.

G = Gray Code - Select the Gray Code output-format option.

c: Resolution

1 = 0.005 mm (5 µm)

2 = 0.01 mm (10 µm)

3 = 0.05mm (50 µm)

4 = 0.1 mm (100 µm)

5 = 0.02 mm (20 µm)

6 = 0.002 mm (2 µm)

This setting does not affect the SSI Configuration dialog, but must be taken into account when setting the Scale.

d: Performance

1 = Standard

This setting does not affect the Configuration word.

e, f: Scale Orientation

00 = Forward-acting

01 = Reverse-acting

02 = Forward-acting, Synchronized

This setting does not affect the Configuration word.

Note: It is highly recommended that a Synchronized transducer (Scale Orientation of 02) be selected. This ensures that the time between position samples matches the control loop time of the RMC controller. If the transducer is not synchronized, the sample time may not match and make precise speed control difficult.

See also:

SSI Overview

SSI Wiring

SSI LED Indicators

SSI Specifications

6.6.4 SSI LED Indicators

The two LEDs above the Drive connector on the RMC's SSI interface card are axis status LEDs. These LEDs will reflect the operational status of their corresponding axis according to the following table.

LED Action	Axis Status
Continuous Red	One or more of these errors have

	occurred: No Transducer Transducer Noise Transducer Overflow
Alternating Red/Green	None of the above errors occurred, and one or more of the below errors have occurred and are enabled in the Auto Stop: Overdrive Error Parameter Error Position Overflow Integrator Windup Following Error
Continuous Green	Status good. None of the above are true.

Note: Prior to RMC CPU firmware dated 19991216, the Auto Stop parameter was not used in determining the LED states. Therefore, the only way to keep the LED from showing red was to clear the error.

See also:

SSI Overview

SSI Wiring

SSI Configuration

SSI Specifications

6.6.5 SSI Specifications

For general specifications on the RMC, see RMC100 Specifications.

Transducer Inputs

Axes	Two per module
Inputs	Two RS422 differential
Clock outputs	Two RS422 differential

Clock frequency	220 kHz
Cable type	Twisted pair, shielded
Cable length maximum	Transducer dependent (approx. 300-600 ft)
ESD protection	15 kV
Resolution	Transducer dependent (up to 2 mm or approximately 0.00008" for MDTs)
Count encoding	Binary or Gray Code
Count data length	4 to 32 bits

Drive Outputs

Range	±10 V @ 5 mA (2 kW or greater load) (For current drive, use the VC2100 accessory: ±10 mA to ±200 mA in 10 mA steps)
Tolerance	At 10 V: +200 mV, -100 mV At 0 V: ±50 mV At -10 V: +100 mV, -200 mV
Resolution	12 bits
Output isolation	750 VDC, optically isolated
Overload protection	One-second short-circuit duration
Overvoltage protection	Outputs are protected by clamp diodes

See also:

SSI Overview

SSI Wiring

SSI Configuration

SSI LED Indicators

6.6.6 SSI Scaling

Defining the Valid 16-bit Position Range

For general scaling information, see the Scaling Overview topic.

Because the RMC uses 16-bit positions, positions must all fit within a range of 65,536 position units. Because position units are user definable, this range does not limit most applications. See the section below on defining position units.

For SSI axes, the Offset parameter is used with the Scale parameter to define the position range. If the Scale parameter is negative, then the position range extends from the Offset value minus 65535 up to the Offset value. If the Scale parameter is non-negative, then the position range extends from the Offset value up to the Offset value plus 65535. The following chart summarizes this concept:

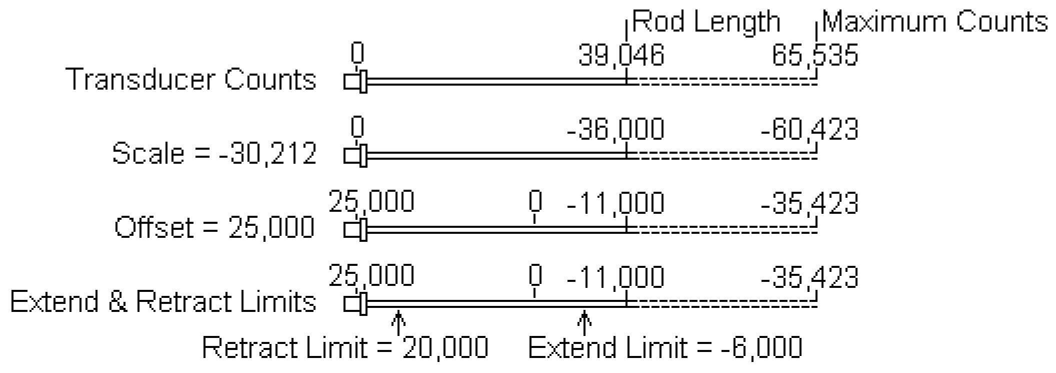
Scale	**Min. Position**	**Max. Position**
< 0	Offset - 65535	Offset
^{3} 0	Offset	Offset + 65535

However, because the Offset is also used to convert transducer counts to position units, it cannot be set independently.

Translating to Position Units

Before the translation from counts to position units begins, the Count Offset SSI configuration setting is applied to the counts. This means that the value of the Count Offset is subtracted from the Counts before they are displayed or processed in any way. For a full discussion on the use of the Count Offset setting, see SSI Configuration.

The Scale, Offset, and the Prescale Divisor bits of the Configuration word parameters are used to define position units as a function of transducer counts. Below is shown an example SSI and the effect of the Scale, Offset, Extend and Retract Limits:



The following formula summarizes the translation from transducer counts to Actual Position units:

$$\text{Actual Position} = \frac{\text{Counts} \times \text{Scale}}{32768 \times \text{Prescale Divisor}} + \text{Offset}$$

As described above, the Counts field has already had the Count Offset configuration setting subtracted from it before using this formula. If the Actual Position overflows the valid 16-bit position range, as described above, the Position Overflow bit will turn on, forcing the axis to halt.

Notice that the Scale can hold values from -32768 to 32768, but 16-bit signed numbers are limited to -32768 to 32767. Therefore, enter a scale of 32768 as 0; the RMC will interpret it as 32768.

There are two main ways to calculate the Scale, Offset, and Prescale Divisor bits. Each method is described below. Notice that both have automatic Calibration Utilities built into RMCWin. It is recommended that one of these utilities is used, but the underlying math is described below.

Method 1: P0/P1 Calculation

The simplest way is to physically measure the axis's position at two points and read how many counts the RMC reports at each position. If we call the two positions, in user position units, P0 and P1, and call the corresponding counts C0 and C1, the following two equations will give a Scale and Offset. Notice that the Prescale Divisor is left out at this point; it is assumed to be 1:

$$\text{Scale} = \frac{(P_0 - P_1)}{(C_0 - C_1)} \times 32768$$

$$\text{Offset} = P_0 - \frac{(P_0 - P_1) \times C_0}{(C_0 - C_1)}$$

Once the Scale is calculated, the Prescale Divisor can be calculated. The Prescale Divisor can have values of 1, 2, 4, or 8. Because the Scale is always divided by the Prescale Divisor, you essentially have a fractional scale. Pick the largest Prescale Divisor you can multiply the Scale by and still be between 32767 and 32767. For example, suppose your Scale comes to 6324.70. The following table shows the possible Scales and Prescale Divisors you could use and the effective scale:

Scale	**Divisor**	**Effective Scale**	**Error from 6324.70**
6325	1	6325/1 = 6325	0.005%
12649	2	12649/2 = 6324.5	0.003%
25299	4	25299/4 = 6324.75	0.0008%
50598	8	Invalid scale	Invalid

Therefore, in this example, a Scale of 25299 and a Prescale Divisor of 4 should be used.

These calculations can be done automatically using the Position Scale/Offset Calibration Utility feature in RMCWin.

Method 2: Using the Transducer Resolution

The Scale can also be calculated mathematically using the SSI transducer's reported resolution. The following formula does this. Notice that the Prescale Divisor is left out at this point; it is calculated after the initial Scale value has been calculated, as described in Method 1:

Scale = Resolution x Position Units/Resolution Unit x 32768 x Sign

In the above formula, Sign equals +1 when an extend move yields increasing Actual Position counts, and -1 when an extend move yields decreasing Actual Position counts. Position Units are generally 0.001 inches or 0.1 mm. The transducer resolution is often given in mm or µm.

Once the Scale and Prescale Divisor have been calculated, the Offset can be calculated:

$$\text{Offset} = P_0 - \frac{\text{Scale} \times C_0}{32768 \times \text{Prescale Divisor}}$$

These calculations are done automatically using the SSI Scale/Offset Calibration Utility feature in RMCWin.

Example 1

A system has an MDT with an SSI interface that has a resolution of 5 microns. The user wishes to have positions in thousandths of an inch. At the desired 0 position, the MDT produces 3221 counts.

We first calculate the exact Scale:

$$\text{Scale} = \text{Resolution} \times \text{Position Units/Resolution Unit} \times 32768 \times \text{Sign}$$

$$\text{Scale} = 5\mu\text{m} \times \frac{1000\,\text{position units}}{1\,\text{inch}} \times \frac{1\,\text{inch}}{25400\,\mu\text{m}} \times 32768 \times 1 = 6450.39$$

With the exact Scale value, we must choose the Prescale Divisor and the rounded Scale value. The highest divisor value we can use is 4; multiplying the scale by 8 would overflow the Scale limits of ±32767. Therefore we have the following parameters:

```
Scale = 25802
```

```
Prescale Divisor = 4
```

Next, we must calculate the offset:

$$\text{Offset} = 0 - \frac{25802 \times 3221}{32768 \times 4} = -634.1 = -634$$

Example 2

A system has an MDT with an SSI interface that has a resolution of 2 microns. The user wishes to have positions in hundredths of a millimeter. At the desired 0 position, the MDT produces 8273 counts.

We first calculate the exact Scale:

$$\text{Scale} = \text{Resolution} \times \text{Position Units/Resolution Unit} \times 32768 \times \text{Sign}$$

$$\text{Scale} = 2\mu\text{m} \times \frac{100\,\text{position units}}{1\,\text{mm}} \times \frac{1\,\text{mm}}{1000\,\mu\text{m}} \times 32768 \times 1 = 6553.60$$

With the exact Scale value, we must choose the Prescale Divisor and the rounded Scale value. The highest divisor value we can use is 4; multiplying the scale by 8 would overflow the Scale limits of ±32767. Therefore we have the following parameters:

```
Scale = 26214
```

```
Prescale Divisor = 4
```

Next, we must calculate the offset:

$$\text{Offset} = 0 - \frac{26214 \times 8273}{32768 \times 4} = -1654.6 = -1655$$

7 Support and Troubleshooting

7.1 Warranty

The RMC100 shall be free from defects in materials and workmanship under normal and proper use and service for a period of fifteen (15) months from the date of shipment by Delta Computer Systems, Inc. (Delta) or Delta's authorized distributor so long as the module was under warranty when shipped to the customer by the distributor.

The obligation of Delta under this warranty shall be limited to repairing or replacing this motion controller or any part thereof which, in the opinion of Delta, shall be proved defective in materials or workmanship under normal use and service during the warranty period.

Repairs required because of obvious installation failures (burned resistors, traces, etc.) are not covered and will be billed at standard repair rates.

Disclaimer of other Warranties: There are no other representations or warranties made by Delta, express or implied. **Delta expressly disclaims any and all implied warranties, including any implied warranty of merchantability and any implied warranty of fitness for a particular purpose.** Further, Delta disclaims any liability for special, consequential or incidental damages resulting from any breach of warranty by Delta under this Agreement.

RMC Return for Repair
Should an RMCrequire repair, refer to Technical Support for return details.

7.2 Troubleshooting

7.2.1 Programming Hints

- The Programmable Controller is responsible for storing the initialization parameters used by the RMC and initializing the RMC with those parameters.
- The RMC provides a Status word for each axis. If an error bit is enabled in the Status word, the Programmable Controller is responsible for shutting down the axis drive power. It must have a watchdog timer that will shut down the drives if a time-out occurs.
- The PC should write to the RMC only once per scan, otherwise the newest data will overwrite the previous data.

7.2.2 Error Handling

The RMC reports errors to the Programmable Controller within one control loop of detection. Errors are reported by setting bits in the affected axis's Status word and turning on the appropriate LEDS. The Programmable Controller is responsible for checking errors by reading the Status words. It is up to the Programmable Controller to determine what should be done if an error is detected.

The system must be able to shut down the axis drive power using a normally open output that is held closed when the system is running. This contact should be in series with an operator emergency off button. If power to the rack is lost, the contact will open and the axes will stop. If an error occurs in the RMC, the contact can be deactivated, which stops the axes. Usually the Programmable Controller will not take so drastic a step until it has determined that all control is lost. A Halt command to the axis with an error can take care of most error conditions.

When two axes are making a coordinated move and one axis starts moving slower than it should, it is best to issue a Halt command to both axes to stop all movement until the problem with the faulty axis has been resolved.

You can also use Auto Stop detection by setting the appropriate bits in the SOFT STOP the HARD STOP bytes corresponding to the error bits in the Status word.

7.2.3 RMC Module Problems

Control program cannot access parameters or operate module

1. Module not configured properly. Refer to the topics on your communication type. Use the online help's table of contents to find these topics.

2. Make sure the Programmable Controller is accessing the correct I/O registers.

Axis LEDS are red

This indicates the transducer is not responding to the module. Every millisecond the module interrogates the transducer's position. If the transducer does not respond correctly, the red axis LEDs are turned on. Check the transducer power supply and the wires to the transducer.

During a move, the Actual Position is erratic

Electrical noise or a defective transducer is usually the cause of this problem. Monitor bits 13, 14, and 15 of the axis's Status word to determine if the module is detecting a transducer error. To reduce electrical noise check the following:

1. Make sure the transducer wiring is separated from all other wiring.

2. Add a termination resistor (220 ohm for Temposonics I) as close to the transducer as possible.

3. Connect the shield at the module end, the transducer end, or both.

During a move, the drive comes to a halt for no apparent reason

When the RMC detects a 'transducer not responding' error it makes a Hard Stop. See 'Actual Position is erratic' above for more information. If any of the following conditions are enabled, the axis will also halt:

Following Error

Overdrive Error

Position Overflow

Parameter Error

Integrator Windup

Transducer counts field not indicating transducer location
See ”Axis LEDS are red” above.

Transducer counts field changes but output drive does not work
See ”During a move, the drive comes to a halt for no apparent reason” above.

The System is unresponsive and hard to tune
This problem could have several causes. The first items to check are:

1. Is hose, rather than rigid pipe, installed between the hydraulic valve and the cylinder? The hose expands and acts like an accumulator and the fluid goes to fill the hose rather than move the cylinder.
2. Does the valve have overlap? Overlap in hydraulic valves causes a significant dead band and slows the system response. Some proportional valve amplifier cards have dead band eliminator circuits that make tuning easier.
3. If you have a servo motor and a ball screw, is there any backlash? Backlash produces a dead band when the axis changes direction, so the controller will tend to oscillate around the dead band.

The axis oscillates
This problem could have several causes. The first items to check are:

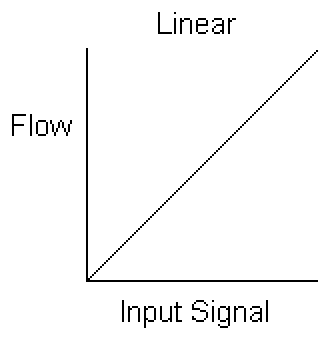
1. Make sure that the Dead Band Eliminator value is not too high.
2. Try reducing the Proportional, Integral, or Differential Gains.

The axis does not finish moves or moves differently than expected
This can be caused by the P/C issuing unintended commands to the RMC. Use the Command Log to monitor commands sent by the P/C to the RMC. Confirm that only the expected commands are being sent.

7.2.4 Hydraulic System Problems

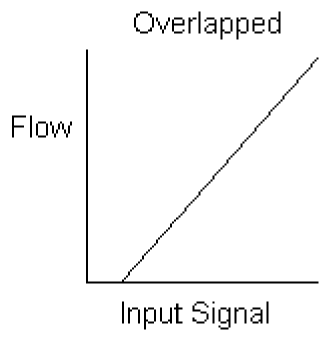
These hydraulic system problems can make system tuning difficult or impossible.

Nonlinear Valves
A valve is linear when the flow through it is directly proportional to the input signal over the entire range of the input signal:

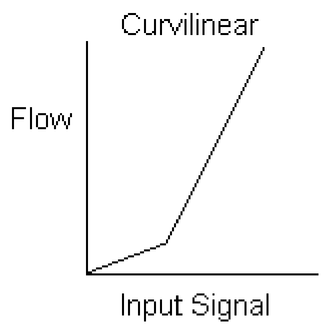


It is nonlinear when the output is not directly proportional to the input. You may find two types of valve non-linearity:

Overlapped valves - Oil does not start to flow through these valves until the spool has moved some distance. This causes a dead band in the system, where small amounts of drive do not produce motion. Overlapped valves are designed for manual and on/off type control and are not suited for servo control. These valves should be replaced with non-overlapped valves.



Curvilinear valves - The flow through these valves increases slowly as the input signal increases for the first 20% of range. Beyond 20% the flow increases rapidly as the input increases. This is equivalent to having two different gains for different signal levels. The low gain at low flow causes poor response at slow speeds, and the high gain at high speed can cause instability. These problems are more pronounced when heavy loads are moved by relatively small cylinders.



Slow-Response Valves

Valves with slow response cause the RMC to overcompensate for disturbances in the motion of the system. Since the system does not respond immediately to the control signal, the RMC continues to increase the drive signal. By the time the system begins to respond to the error, the control signal has become too large and the system overshoots. The RMC then attempts to control in the opposite direction, but again it overshoots. These valves can cause the system to oscillate around the set point as the RMC overshoots first in one direction, then the other.

Hoses

Long hoses between the valves and cylinder act as accumulators and make the system respond as if it has a spring in it (imagine trying to control the position of one end of a Slinky™ by moving the other end!). The lines between the valves and cylinders must be as short and rigid as possible.

Pumps and Accumulators

Insufficient pump and/or accumulator capacity will cause the system response to degrade during a move because the effective pressure drops.

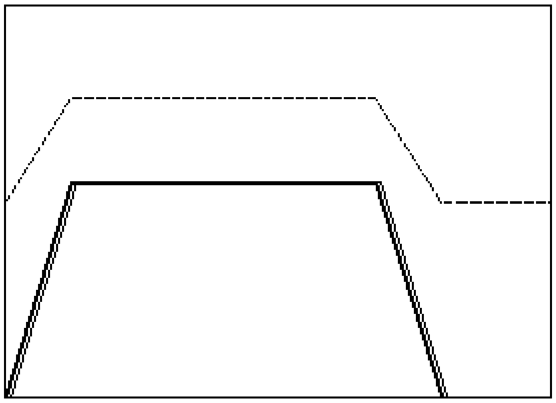
Pressure transients due to insufficient accumulator volume cause jerky motion, particularly during starts and stops.

Note: Even systems with 'r;fast' pumps usually require at least a small accumulator near the cylinder to maintain the constant pressure needed to get smooth motion.

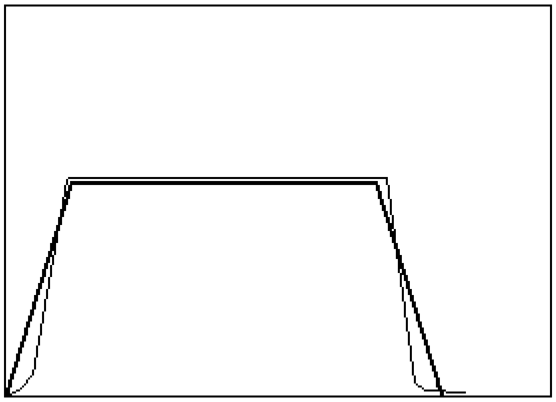
Insufficient pump capacity can result in inadequate control when moving many axes simultaneously or when making long moves. In these cases pressure can drop so much a fully open valve cannot maintain the requested speed.

Identification and Correction

To identify and correct these problems, make a move with very low (or zero) gains except the Feed Forward terms. Graph the move. The graph (ignoring the Position terms) should show:

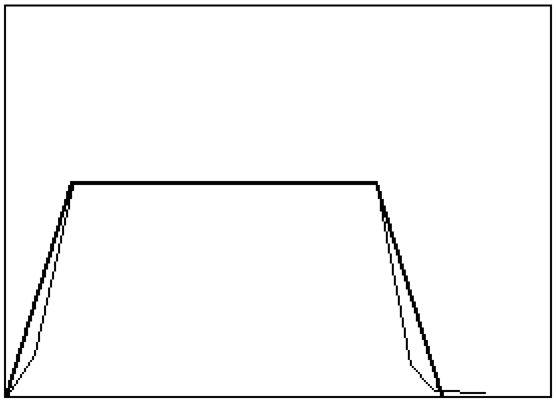


If the actual speed and target speed show:



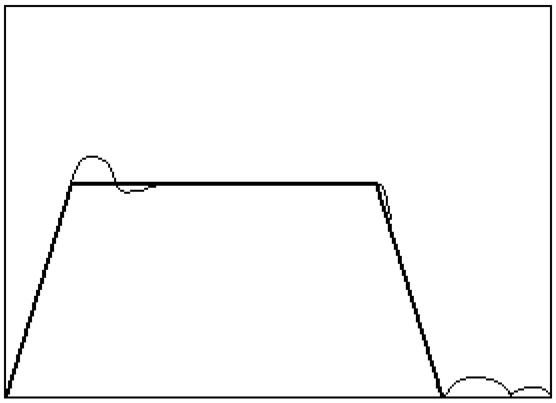
Your valve probably has overlap. Replace the valve with a linear one or try increasing the Dead Band Eliminator value.

If the speeds show:



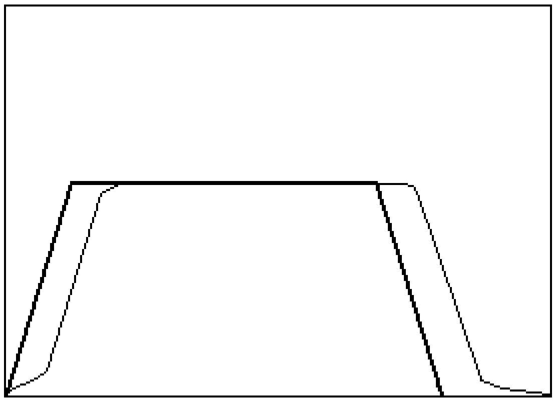
Your valve is probably curvilinear. Replace the valve with a linear one or increase the proportional gain and tune the system for high-speed stability; expect poor control at low speed and when stopped.

If the speeds show:



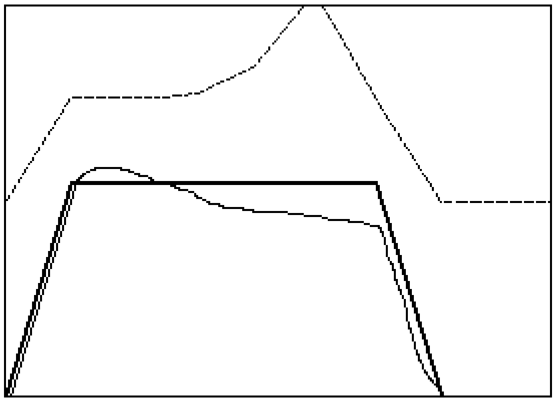
You may have too much hose between the valve and the cylinder. Reduce the amount of hose or add differential gain (usually less than 5).

If the speeds show:



Your valve may have slow response. Change to a faster valve or add Acceleration Feed Forward.

With normal gain values, if the graph shows:



Your pump and/or accumulator may be inadequate (you are running out of oil). Reduce speed, increase pump pressure, add accumulator volume, or get a bigger pump.

7.3 Technical Support

7.3.1 Technical Support

Delta Technical Support Numbers
Phone: 360-254-8688 (24-hour emergency support available)

Fax: 360-254-5435

www.deltamotion.com

support@deltamotion.com

RMC Return for Repair
If you need to return the RMC for repair, please contact Delta prior to shipment for an RMA number. Returned RMCs must be packaged in static protection material and have the RMA number clearly marked on the outside of the package. Please include a short note explaining the

problem. Send the module to:

Delta Computer Systems, Inc.

1818 SE 17th St

Battle Ground, WA 98604

Returns
Contact Delta for details on returning items. An RMA number must be issued before an item is returned.

7.4 Parameter Errors

7.4.1 A valid segment has not been calculated

This indicates that a Follow Spline Segment command was issued but there are no completed segments to follow. To fix this problem, you must add a segment by using New Spline Point to add two or more points, and then issue an End Segment to calculate the curve of the spline.

7.4.2 Acceleration overflow while calculating spline

This parameter error occurs during an End Segment command when the spline points added calculate out to have accelerations that are too large. If this occurs, you must increase the time between each point by using the Set Spline Interval or add points closer together.

The segment will still remain in the RMC's buffer, and must be deleted using the Clear Spline Segments command if you want to start another segment.

7.4.3 Attempt to enter pressure immediately failed

This parameter error occurs when a Set Position/Pressure command is issued with the Monitor Pressure mode bit set, and the axis cannot enter pressure control. There are several reasons why this may occur:

- The axis this command is issued to may not have a pressure axis assigned to it. Ensure that you have set the Pressure Axis bits in the CONFIG word. If you have done so, ensure that you have issued a Set Parameters command since the Pressure Axis bits have been set.
- The axis assigned to the pressure/force axis may not be initialized. Issue a Set Parameters command to the pressure/force axis.

The current pressure on the analog pressure axis cannot be below the Pressure Set B command value for the axis. This prevents the axis from immediately falling out of pressure.

However, it is acceptable that the current pressure be below Pressure Set A (which otherwise controls the pressure required to enter pressure control).

For more details on controlling pressure, refer to Controlling Pressure or Force.

7.4.4 Attempt to go beyond extend limit

This parameter error will occur whenever either a Go or Relative Move command requests that the axis moves to a position beyond the EXTEND LIMIT. The COMMAND POSITION will be set to the EXTEND LIMIT. The move will continue if the bits for the Parameter Error are cleared in the AUTO STOP word. To remedy this problem:

- Fix the control program to avoid issuing illegal commands.
- If the EXTEND LIMIT is set incorrectly, fix its value.

7.4.5 Attempt to go beyond retract limit

This parameter error will occur whenever either a Go or Relative Move command requests that the axis moves to a position beyond the RETRACT LIMIT. The COMMAND POSITION will be set to the RETRACT LIMIT. The move will continue if the bits for the Parameter Error are cleared in the AUTO STOP word. To remedy this problem:

- Fix the control program to avoid issuing illegal commands.
- If the RETRACT LIMIT is set incorrectly, fix its value.

7.4.6 Attempt to send spline through Spline Download Area while download in progress

A spline may not be downloaded through the Spline Download Area when a download is in progress to the same axis either through another Ethernet connection or using X and T commands. You must re-issue the spline through the Spline Download Area after the other download has completed.

7.4.7 Attempt to write to the Spline Download Area of a non-existent or non-spline capable axis

Splines may only be downloaded to axes that are spline-capable. You attempted to download a spline segment to either a non-existent axis, or to an axis that exists but is not spline capable. The only non-spline-capable axes are auxiliary pressure or differential force channels.

7.4.8 Auto-Repeat Should Not be Used on Linear Axes with a Curve that Does Not Match Endpoints

All of the following was true:

- An Auto-Repeat curve was downloaded to an axis with an absolute transducer type such as MDT, SSI, or Analog, which cannot be offset.
- The curve's first and last points were at different positions.
- The curve was followed beyond its last point, thus triggering an Auto-Repeat.

If any of the above was not true, then an error would not have been generated. That is, Auto-Repeat curves are allowed on non-offsettable axes, provided their endpoints match. However, because this situation did occur, the curve still Auto-Repeated, but did not offset the position back to the first position in the curve.

7.4.9 Axis must be initialized to use this command

Many commands require that the axis to which they are issued be initialized. An axis is initialized if the Axis Initialized bit is set in its Status Word, which happens when the axis receives a Set Parameters (P) command. Ensure that you have issued this command the axis that got this error. The only way the Axis Initialized bit will clear is if the module is reset, as will happen when power to the module is cycled.

7.4.10 Axis must be stopped for this command

The 0x60 (`) command "Set Absolute Rotary Encoders to Same Turn" requires that the axis be stopped or in open loop. This command was ignored because the axis was not in either of these states.

This command is available in the 'B' code only.

7.4.11 Both sync bits cannot be set in the "Mode" word

There are two Sync bits in the MODE word. Only one or neither can be set at a time. This parameter error is generated if both are set.

7.4.12 Cannot clear a segment while interpolating

This indicates that a Clear Spline Segments command was issued while a Follow Spline Segment command was in progress. Because splines are cleared from the oldest to the newest, and the currently followed spline is the newest, it is not possible to clear any segments while the current spline is being followed.

7.4.13 Cannot home an axis while synchronized

Homing an axis changes the current position and would confuse a synchronized axis. For this reason, homing is not allowed while an axis is synchronized.

7.4.14 Cannot issue a ’r;Z’ or ’r;z’ command to a synchronized axis

These commands change the current position of the axis, which would confuse the synchronized axes; therefore, these commands cannot be issued while an axis is synchronized.

7.4.15 Cannot overflow command pressure

This error is generated when the Move Relative (J) command is issued to a pressure axis, but the current Command Pressure plus the Command Value would overflow the range for the Command Pressure: -32768 to 32767. You either did not have the Command Pressure you expected when the command was issued, or you need to change the Command Value issued with the Move Relative (J) command.

If this error occurs, the command will ignored.

7.4.16 Cannot use synchronization with speed control

Speed Control and Synchronization are mutually exclusive features; that is, an axis may use one or the other but not both at the same time. However, Gearing can be used with Speed Control. Refer to the Mode word for details on enabling these features.

7.4.17 Command pressure cannot be less than pressure set A

Note: This condition is no longer considered a parameter error in RMC CPU firmware versions newer than 19980414. In those versions of firmware, when Pressure Set A is set above Command Pressure and the pressure reaches Pressure Set A, then the pressure is ramped to the Command Pressure using a full ramp.

When using Pressure Set Mode, the Command Pressure—which is issued as the Command Value in a Set Pressure command—must always be greater than or equal to Pressure Set A.

7.4.18 Command pressure cannot be less than pressure set B

When using Pressure Set Mode, the Command Pressure—which is issued as the Command Value in a Set Pressure command—must always be greater than or equal to Pressure Set B.

7.4.19 Dead band eliminator out of range

This parameter error indicates that the DEAD BAND ELIMINATOR parameter was set to a value greater than ±2000. This maximum prevents a dead band of greater than 2 volts. If a valve still does not work to satisfaction with this large of a dead band, then the valve is most like not appropriate for the job. Refer to Hydraulic System Problems for details.

7.4.20 Drive transfer percentage out of range

This parameter error indicates that the DRIVE TRANSFER PERCENT was set to a value outside the maximum range of ±500. The value will have been truncated to fit in this range. Change this parameter to a value inside the range and issue a Set Parameters command.

7.4.21 "Event Step Edit" indices are invalid

The Programmable Controller sets the beginning and ending step numbers to edit with the E0 and E1 Event Step Edit commands. When the E3 Event Step Edit command is issued—which actually performs the step table modifications—the beginning and ending step numbers are checked. Both must hold valid step numbers (between 0 and 255, inclusive), and the ending step number must be greater than or equal to the starting step number.

If the beginning and ending step numbers are out of range or reversed, this parameter error is generated.

7.4.22 Extend limit must be greater than retract limit

This parameter error indicates that the EXTEND LIMIT was set to a position in which the TRANSDUCER COUNTS would be less than or equal to those at the RETRACT LIMIT position. Therefore, if the SCALE is positive, this error indicates that the EXTEND LIMIT is less than the RETRACT LIMIT. Otherwise, this error indicates that the EXTEND LIMIT is greater than the RETRACT LIMIT.

7.4.23 Feed forward terms must have the same sign

On pressure axes it is possible to use both positive and negative values for gains and feed forwards. However, all should have the same sign. If the EXTEND FEED FORWARD and

RETRACT FEED FORWARD parameters have opposite signs, this parameter error will be generated.

7.4.24 Fewer segments than were requested to be cleared existed

This indicates that a Clear Spline Segments cleared all segments available, but there were fewer segments than the user requested to be cleared available to clear. This indicates that the user somehow lost track of the count of segments, which can indicate problems with the controlling program.

7.4.25 Flash contained no data on startup

This parameter error will appear on the first axis on startup whenever no data is found stored in the Flash memory. Although the parameter error is only displayed on the first axis, it indicates that parameters for all axes, event steps, profile tables, input to event tables, and configuration data was not loaded from Flash. To store this in the Flash, load the motion controller with all desired data, and issue an Update Flash command. You must wait for the CPU LED to stop flashing green before powering off the module. If this parameter error appears, one of the following has occurred:

- The module is newly shipped. New modules have no data stored in the Flash.
- The module was powered down while updating the Flash. This results in an invalid checksum being found for the data, so the Flash is not usable on next power-up.
- The Flash itself has a problem. If you find that the Flash consistently does not store data, although the module is not being powered down while the CPU LED is flashing (the Flash is being written to), contact the Delta for details on having the Flash replaced.

7.4.26 Function in the Function (,) Command Out of Range

The Function (,) command was issued with an invalid function value in the Deceleration command field.

The only valid function values are:

1. Minimum of Selected Axes' Actual Positions
2. Maximum of Selected Axes' Actual Positions
3. Mid of Selected Axes' Actual Positions
4. Average of Selected Axes' Actual Positions

7.4.27 Gear ratio denominator is zero

The error indicates that this axis was given a geared command with a zero denominator. The gear ratio is given with the numerator in the Command Value field and the denominator in the Speed field.

To fix this error, simply enter the correct ratio in these two fields before issuing the geared command or, if you unintentionally issued the geared command, check that you have set up the Mode word correctly.

7.4.28 Gearing and Synchronization Illegal in Open Loop

When using the Open Loop command, the axis must be neither synchronized nor geared. Clear the Gear Mode and Sync Group mode bits.

7.4.29 Incompatible sync mode words

This error indicates that the low eight bits of the mode words do not match between a set of synchronized axes. These bits are required to match to ensure that the same type of move is executed between all of the synchronized axes. Decide which MODE bits you would like to use, and use the same mode bits for all synchronized axes. Because only the low eight bits are required to match, the Graph Disable bit can be different between synchronized axes.

7.4.30 Internal error while using the Spline Download Area

This parameter error indicates that the RMC had an internal failure while download a spline. Contact Delta Computer Systems, Inc. technical support with details on your system if this error occurs.

7.4.31 Invalid Address Used in Add (+) or Subtract (-) Command

The Add (+) and Subtract (-) commands use two address fields. Each uses the value in the Deceleration command field as the source address, and the value in the Command Value as the destination address. Only addresses in the range of 0 to 2303 are valid. For a map of these addresses, see RMC Register Map (PROFIBUS-DP Message Mode), but keep in mind that, once again, only addresses 0-2303 are accepted by these commands.

7.4.32 Invalid command received

This parameter error indicates that the COMMAND is invalid. Refer to that topic for a list of valid commands. Ensure that the command you issued is available for your axis type. That is, some commands are specific to only quadrature or only pressure/force control.

7.4.33 Invalid command value

This parameter error is set when the command value exceeds the range for that command, as described in the table below. The command will be ignored, except Set Null Drive, which will be truncated to +/-2000.

Command	**Valid Command Value Range**
Set Integral Drive	±10000
Set Null Drive	±2000
Limit Drive	0 to 10000
Amp Enable/Disable Command	0 or 1
Clear Spline Segments Command	Greater than -16
Map Output to Axis Position	See the Map Output to Axis Position topic for valid Command Values.
Set Mode Command	Only the following bits can be changed: Gear Type bit, Gear Master Select bits, Monitor Pressure bit, Rotational bit. The Gear Type and Gear Master Select bits can only be changed when the Gear Mode bit is set and when in Spline or Gear states, except Gear mode 2. The Rotational bit can only be changed when doing a Speed control move or point-to-point move, unless stopped.
Update Flash Segment	0 or 1

7.4.34 Invalid Gear Master Selected

This error indicates the Mode word was set wrong. When the Gear Mode bit is selected, bits 4-6 of the mode word select the gearing master. The following are reasons this error may occur:

- An axis cannot be geared to itself.
- An axis cannot be geared in a loop. For example, axis 0 cannot be geared to axis 1 if axis 1 is geared to axis 0.
- An axis cannot be geared to an auxiliary pressure or force axis.

In RMCWin, use the Popup Editor on the Mode word to select a valid gear master axis.

7.4.35 Invalid Interval Table Format in the Spline Download Area

The Interval Table Format register in the Spline Download Area may either have a value of 0 or 1, as described in Downloading Splines to the RMC. You will also get this parameter error if you download the Spline Points without having set this register at all.

7.4.36 Invalid MODE bits set for this command

This error is generated when reserved bits are set in the MODE field when a command that uses the MODE field is issued. Double-check that you have not set any bits other than those documented in this manual.

7.4.37 Invalid Point Count in the Spline Download Area

The Point Count register in the Spline Download Area was given an invalid value or was not set before downloading Spline Points. For details on this register see Downloading Splines to the RMC. This register cannot be below 2 or above the point limit per axis.

The point limits depend on the number of spline-capable axes in the RMC. All axes and channels listed by RMCWin as a separate column are spline-capable axes except auxiliary pressure or differential force channels.

The following chart lists the number of points allowed on a given axis depending on the number of spline-capable axes:

Axes	**Point Limit**
2	1023
3-4	511
5-8	255

7.4.38 Invalid scale value

This parameter error indicates that a SCALE was set to an invalid value: either 0 or 32768. If you are having a difficult time setting the SCALE parameter, you may want to try using the Scale/Offset Calibration Utilities.

Note: This error does not exist in RMC CPU firmware dated 19991216 or later, as scales of 0 and 32768 are both allowed on all types of axes.

7.4.39 Invalid Screen Number in the Display LCD Screen ($) Command

The screen number indicated in the Command Value field of the Display LCD Screen ($) command was out of range. The valid screen numbers range from 0 to the highest screen number available. Therefore, if the RMC has four screens loaded into it, then the valid range of screens would be 0 to 3.

7.4.40 Invalid step number given in "Start Events" command

Because there are only 256 event steps—numbered 0 to 255—any COMMAND VALUE outside of that range with the Start Events command will generate this error.

7.4.41 Maximum Steps per Millisecond parameter out of range

This error is generated when a Set Parameters (P) command is issued when the MAX STEPS/MSEC parameter is out of range. The valid range is from 1 to 1024 steps per millisecond.

Enter a valid value for this parameter and re-issue the Set Parameters (P) command.

7.4.42 Move would cause discontinuity

This parameter error is generated when the requested move fields would have generated a discontinuity in the speed. This occurs if the axis is stopped, and the user specifies an ACCELERATION distance of 0 on a move to another location. In order to carry out this request, the motion controller would have to instantaneously increase the speed to the maximum speed. This would generate a discontinuity, and therefore instead the move is rejected and this parameter error is generated. To fix this problem:

- Verify that your control program is sending the command correctly.
- If you really want to accelerate instantaneously to the maximum speed, you can use a very high ACCELERATION in Acceleration Modes 0 and 1, or a very low ACCELERATION in Acceleration Modes 2 and 3.

7.4.43 No Axes Selected for Use by the Function (,) Command

The Speed command field is used by the Function (,) command to select which axes the function acts on. This field cannot be zero, because all of these functions must act on one or more axes.

7.4.44 No initialized pressure axis is assigned

This parameter error occurs when a move is started with the Monitor Pressure bit set in the MODE word, but either no pressure axis assigned to the position axis, or the pressure axis assigned has not been initialized. Consider the following three possibilities:

- Did you intend to set the Monitor Pressure bit in the MODE word? If not, then you should open the MODE word popup editor in RMCWin and clear the Monitor Pressure check box.
- Have you assigned a pressure axis to the pressure axis? To verify this, check the CONFIG word to see if a pressure axis has been assigned. Make sure that you've issued a Set Parameters command to send the current parameters to the motion controller.
- Have you initialized the assigned pressure axis? You must send a Set Parameters command to the pressure axis as well as the position axis.

For more details on controlling pressure, refer to Controlling Pressure or Force.

7.4.45 Non-existent pressure axis selected in "Config" word

This parameter error indicates that the CONFIGURATION word has been set incorrectly. Specifically, bits 8, 9 and 10 have been set incorrectly. These bits are used to reference the pressure axis that should be assigned to the position axis. This could occur because of any of the following reasons:

- There are no analog input modules installed in the RMC that have pressure control enabled. These are included in the RMC101-series products from Delta Computer Systems, Inc.
- There are multiple axes trying to use the same analog axis. Each pressure axis can only be assigned to a single position axis.
- The pressure axis number selected is larger than the number of pressure axes available. For example, in an RMC101-M1-A1 with the four pressure channels configured as two differential force axes, only pressure axes 0 and 1 can be selected, and attempts to access pressure axis 2 and 3 (using CONFIGURATION values of 0x0A00 and 0x0B00) will generate this error.

7.4.46 Numeric overflow while sending a spline to the Spline Download Area

The last sequence of points sent to the Spline Download Area resulted in a numeric overflow. This will only happen if your points are positioned such that speeds and accelerations for the spline curve would have been too high. Try increasing the time (or master counts) between points, or reducing the distance (in position units) between points.

The spline you were attempting to download will not succeed. You should restart sending the spline at the first point.

7.4.47 One or more synced axes are uninitialized

This error indicates that at least one of the axes selected to participate in the synchronized move has not been initialized. The axis that is not initialized will be marked with the Parameter Error bit and will not have the Parameters Initialized bit set in the STATUS word. To initialize the parameters on this axis, issue a Set Parameters command.

7.4.48 Overflow while adding point. Point not added

This parameter error occurs during a New Spline Point command when the spline point added would generate an acceleration overflow. If this occurs, you must increase the time between each point by using the Set Spline Interval or add the point closer to the last point.

The rest of the segment currently being added will still be in the RMC's buffer, and can be added to or ended.

7.4.49 Point cannot be added during calculations

While the current spline segment is being calculated, no new points can be added. Refer to the End Segment for details on waiting properly for the calculations to complete.

7.4.50 Position overflow while interpolating spline

This parameter error indicates that the TARGET POSITION calculated while executing a Follow Spline Segment became too large in either the positive or negative directions. The position will be truncated. This can occur because of the overshoot that occurs with splines. To fix this problem you will either need to move the points at the point where the position overflow occurs, or shift the entire set of spline points away from the direction of the overflow.

7.4.51 Pressure Control went outside position limits

This error occurs on a position/pressure axis if the Actual Position moves outside the extend and retract limits while controlling pressure. That is, this error is used to ensure that while pressure is being controlled, the position remains within reasonable limits, as defined by the Extend Limit and Retract Limit parameters. This error will cause the axis to halt, even if the Parameter Error Autostop is set to Status Only.

7.4.52 Pressure set A cannot be less than pressure set B

When using Pressure Set Mode, PRESSURE SET A must always be greater than or equal to PRESSURE SET B. If this were not the case, then it would be possible for the axis to oscillate between entering and exiting pressure regulation mode whenever it is between these two pressures.

7.4.53 Reached command position while regulating pressure

This parameter error indicates that the axis was regulating pressure, but then reached the COMMAND POSITION specified on the move that was monitoring pressure. At this point, the axis does a hard stop. If you do not want the axis to reach the COMMAND POSITION, then you should artificially place the COMMAND POSITION where it cannot be reached.

7.4.54 Requested drive too large

This parameter error indicates that requested drive used by either the Open Loop or Set Bias Drive commands or a move command using Quick Mode is too large. The maximum allowed drive is ±12000 millivolts, although values this large can easily cause an Overdrive Error.

7.4.55 Reserved command parameter must be 0

Certain commands require one or more of the command fields be 0. One of these command fields contained a non-zero value. This field is reserved for future use.

7.4.56 Reserved parameters must be zero

This error is generated when a Set Parameters (P) command is issued when one or more reserved parameters have non-zero values in them. All reserved parameters must be zero. This rule is enforced to allow using these parameters in future upgrades without affecting existing customers.

Enter zero (0) for all reserved parameters and re-issue the Set Parameters (P) command.

7.4.57 Resetting the position is not allowed in this state

The Set Position (Z) and Offset Position (z) commands cannot be issued while in any of the following states:

- Synchronized
- Controlling pressure
- Generating drive outputs based on position from the Map Output to Axis Position (0x7F) command.

7.4.58 Resetting the position would cause a position overflow

This error occurs when either a Set Position (Z) or Offset Position (z) command is issued and the Target or Command position is shifted below the minimum possible position or above the highest possible position. For the Offset Position (z) command, the positions are offset by the amount requested by the user. For the Set Position (Z) command, the positions are offset by the difference between the current Actual Position and the requested position.

For example, suppose that the Target Position is at 3599 position units, the Actual Position is at 3600 position units, and an Offset Position (z) command is issued with a command value of 3600. This will attempt to move the Actual Position back to 0 and the Target Position back to 1. An error may or may not be generated, depending on where the position-unit range is defined. If the Offset is 0 and the Scale is positive, then the position-unit range is from 0 to 65535. A Target Position of 1 is illegal with this range, so this error is generated and the Offset Position (z) command is ignored. However, if the Offset is 32768, and the Scale is positive, then the position-unit range is from 32768 to 32767 and the Target Position can be adjusted back to 1. For this reason, we recommend setting the position-unit range so that the limits are not reached during normal operation, as is the case with the second position-unit range.

The exact same behavior would happen if a Set Position (Z) command is issued with a 0 command value.

7.4.59 Spline Points downloaded out-of-order

While the Interval Table Format, Interval Table, and Point Count registers from the Spline Download Area can be downloaded in any order, the Spline Point table must be downloaded in order, from the first point to the last.

If you feel that you did do this then you may have received this error because one of your previous downloads was in error. For example, if you downloaded the first 20 points of a 40-point spline and it had a parameter error then the RMC will expect you to start over your download. If you continue the download instead, perhaps not checking for the parameter error after the first download, then you will get this parameter error.

7.4.60 SSI transducer overflow

This parameter error is generated for an SSI/Stepper module when the RMC detects that the position on the transducer is outside acceptable limits.

7.4.61 SSI transducer noise

This parameter error is generated when the position of an SSI/Stepper module changes by more than 1/4 revolution in a single control loop.

7.4.62 Step Number in Teach (t) or Function (,) Command Out of Range

The Command Value of the Teach (t) and Function (,) commands refer to the step number. Therefore, these values must be between 0 and 255, since there are only 256 steps in the RMC. A value above 255 was issued in either one of these commands.

7.4.63 Steps per Rev and Position Units per Rev must not be zero

This error is generated when a Set Parameters (P) command is issued when either the STEPS/REV or POS UNITS/REV parameter is zero. These values are invalid. For a complete discussion on how to set these values correctly see Stepper Scaling.

Enter a valid value for this parameter and re-issue the Set Parameters (P) command.

7.4.64 Storage of parameters to Flash failed

This error is generated when an Update Flash (U) command is issued but there is a failure to get the data written to the Flash. You can first try to re-issue the Update Flash (U) command in order to get your machine working, but you should also contact Delta Computer Systems, Inc. to discuss the failure. It may or may not be covered in the warranty.

7.4.65 Storage of splines to Flash failed

This error is generated when splines were being saved to the Flash as requested through RMCWin curve tool, but there is a failure in writing to the Flash. You can first try re-saving the splines to the Flash in order to get your machine working, but you should also contact Delta Computer Systems, Inc. to discuss the failure. It may or may not be covered in the warranty.

7.4.66 Superimposed and gear mode bits required by Master Relative Sine Move command.

The Master Relative Sine Move command requires that both the Superimposed and Gear Mode bits are set. This is required because, otherwise this command is useless.

7.4.67 Synchronized axis was incorrectly dropped

This error indicates that an axis that was currently synchronized did not have its synchronization bits set when a new synchronization command was given. This results in all axes being halted because this situation is ambiguous as to whether the user intentionally unsynchronized the axis, or if the axis was accidentally left out of a new synchronized command. Either one of the following two steps should be taken:

- If the axis no longer should be synchronized, then first explicitly unsynchronize the axis by giving it a Go or Relative Move without any sync bits set in the Mode word before giving a new synchronized command to the remaining synchronized axes.

  NOTE: If the axis that is being unsynchronized was the master axis, then all slave axes will halt.

- Ensure that all synchronized axes have valid Mode, Acceleration, Deceleration, Speed, and Command Value before giving the new synchronized command to the last synchronized axis.

7.4.68 Target position moved outside limits

This parameter error indicates that the axis was about to move its Target Position outside the extend or retract limit. The target position is truncated to the limit when this error occurs. Commands that have a requested position outside the limits are caught immediately by the Attempt to go beyond extend limit and Attempt to go beyond the retract limit parameter errors.

Therefore, this error occurs only in these cases:

- An axis has its deceleration rate lowered in the middle of a move enough so that it overshoots the commanded position and limit. We could catch this error when the command is issued but do not because some applications have an axis decelerate slowly for a period of time, but then issue another command to stop it within the limits. These applications would be impossible if we didn’t allow this deceleration even though, if uninterrupted, it would bring the axis outside the limit.

- A spline is being followed and the spline goes outside the limits. Unless the Auto Stop is set up to halt on parameter errors, the spline will continue even though the positions are truncated at the limits.

Note: This error will not occur when the Actual Position goes outside the limits. That is, you will be able to control position at either limit even though the Actual Position may go outside the limits temporarily.

Note: In firmware dated earlier than 19991021, this error is also used for when a position/pressure axis is controlling pressure and the Actual Position moves beyond the limits. In firmware on or after this date, this condition uses the new Pressure control went outside position limits error instead.

7.4.69 Target position must be equal to the first spline point

The axis receiving a Follow Spline Segment command must have its current TARGET POSITION equal to the position of the first spline point in the segment. This is required to avoid a discontinuity in TARGET POSITION at the start of the spline. An easy way to change the TARGET POSITION to a particular position is to issue a Go command to the position and wait for the axis to reach that position (by monitoring the In Position bit).

7.4.70 The Accel Field Must Be Zero in the Command Issued

The Add (+), Subtract (-), and Function (,) commands all require that the Acceleration command field be 0. One of these commands was issued with a non-zero Acceleration value. This field is reserved for future use.

7.4.71 The acceleration or deceleration ramp is too slow

When a new move command is sent to the motion controller, the motion controller must calculate the distance that each of the ramps (acceleration and deceleration) will take. If this calculation overflows, this parameter error is generated. The overflow occurs when the calculated ramp would take longer than 65535 position units. This overflow can occur even if the axis is moving a much shorter distance. There are two ways to fix this problem:

- Increase the ACCELERATION and/or DECELERATION values.
- Decrease the maximum SPEED value.

If this error occurs and the Parameter Error autostop is set to Status Only, the accel/decel ramp will be set to 65535.

7.4.72 The axis must be stopped before following a spline

Because each spline segment begins with zero velocity, it is necessary that the axis be stopped at the first spline point when the Follow Spline Segment command is issued.

7.4.73 The command acceleration is invalid

This parameter error indicates that the ACCELERATION command parameter was invalid:

Set Bias Drive command:
The ACCELERATION must be less than 20000 millivolts per millisecond in Acceleration Mode 1, and greater than zero in Acceleration Mode 3. If this error occurs, the value will be adjusted to the closest valid value.

P command on a reference axis:

The ACCELERATION parameter is the Reference Acceleration Limit for the reference axis, and must be a valid value. If this error occurs, the P command will be accepted, but it will use the previous Reference Acceleration Limit.

Reference command:

The ACCELERATION parameter is the Acceleration Limit, and must be a valid value. If this error occurs, the value will be truncated.

7.4.74 The command deceleration is invalid

This parameter error indicates that an invalid DECELERATION was given for a Set Bias Drive command. The DECELERATION must be less than 20000 millivolts per millisecond in Deceleration Mode 1, and greater than zero in Deceleration Mode 3. If this error occurs, the command deceleration will be truncated to the closest valid value.

7.4.75 The spline interval cannot be set below 5

This parameter error indicates that a COMMAND VALUE between 1 and 4 inclusive, was used with the Set Spline Interval command. These low values are not allowed because of the increased risk of overflows occurring.

7.4.76 Invalid command for this transducer type

Several commands are valid only for particular transducer types. The following list shows the commands that are specific to a transducer type or types:

Command	Specific to…
Z	Quadrature, Stepper, Rotary Absolute Encoder. All transducer types supported with 20000928 and later firmware.
z	Quadrature, Stepper, Rotary Absolute Encoder. All transducer types supported with 20000928 and later firmware.
@	Quadrature, Stepper, Rotary Absolute Encoder
a	Quadrature, Stepper
060 (`)	Rotary Absolute Encoder
G (Quick Mode)	All except Stepper. Stepper can do all Go commands except Quick mode.

7.4.77 There must be at least two points to begin calculations

An End Segment command cannot be issued until at least two points have been added using the New Spline Point command.

7.4.78 Requested sine-move speed too low

Note: This parameter error has been eliminated in RMC100 CPU firmware dating 19991130 or later. In this newer firmware all speeds are allowed.

This parameter error occurs when, for a Sine Move command, the time it would take to move at the requested maximum speed exceeds 65.535 seconds, resulting in an internal overflow. There are several alternatives if you receive this error:

- Reduce the distance you wish to move with the Sine Move command.
- Increase the maximum speed in your Sine Move command.
- If you cannot reduce the distance traveled or increase the speed, then perhaps you should use a Go command with a Sine profile. The advantage of the Go command is that you can select to have the ramp up and ramp down stages of the move to be almost any length and travel at the maximum velocity in between. This is more efficient: you will be able to move farther over a shorter amount of time than with a Sine Move. The Sine Move command never moves at constant velocity; it ramps up for half the distance and then ramps down over the second half.
- If you really wish to make a Sine Move that takes longer than 65.535 seconds, then consider upgrading to RMC100 CPU firmware dating 19991130 or newer.

7.4.79 Too many points attempted in the Spline Download Area

There is a limit to the number of points that can be downloaded to any given axis. This limit is on the total number of points for all segments on the axis. It does not include segments that have been deleted after they have been executed. If you receive this error, the problem is most likely that you have a spline segment already downloaded to this axis that fills up the rest of the spline buffer.

The point limits depend on the number of spline-capable axes in the RMC. All axes and channels listed by RMCWin as a separate column are spline-capable axes except auxiliary pressure or differential force channels.

The following chart lists the number of points allowed on a given axis depending on the number of spline-capable axes:

Axes	Point Limit
2	1023
3-4	511
5-8	255

7.4.80 Too many spline points. Point not added

This parameter error indicates that the maximum total number of spline points allowed has been reached. This limit is described in Spline Overview. The following is a list of ways around this problem:

- Use the Clear Spline Segments to clear out unused spline segments when possible.
- Break the spline down into smaller segments and follow each segment before issuing the next.

7.4.81 Too many superimposed moves attempted

The superimposed move feature of the RMC allows for, at most, two moves to be executed on a single axis at one time. This consists of a base move (geared, speed control, spline, or sine move) with a superimposed move (spline or sine move). If the superimposed move is still in progress when another superimposed move command is attempted, this parameter error is given. You must first wait until the superimposed-move command is completed. If you wish to take over control of the axis with the command, leave the superimposed mode bit cleared when issuing the command.

7.4.82 Unable to Download a Curve over an Auto-Repeat Curve Using Spline Download Area

The Spline Download Area was used to attempt to download a curve over an Auto-Repeat curve on an axis. While this is technically a valid thing to do, this is not allowed because in many cases this will have been accidental. If you actually intended to overwrite the Auto-Repeat curve, issue a Clear Spline Segments (C) Command to clear the Auto-Repeat curve before downloading the new curve.

7.4.83 Unable to Download Curve over an Auto-Repeat Curve

Either a New Spline Point (X/x) Command or End Segment (T) Command was issued to an axis whose current curve is an Auto-Repeat curve. While this is technically a valid thing to do, this is not allowed because in many cases this will have been accidental. If you actually intended to overwrite the Auto-Repeat curve, issue a Clear Spline Segments (C) Command to clear the Auto-Repeat curve before downloading the new curve.

7.4.84 Velocity overflow while interpolating spline

This parameter error indicates that the calculated velocity reached while executing a Follow Spline Segment command exceeded 65535 position units per second. To eliminate this error, you must give more time between adjacent spline points added with New Spline Point. The time between points is set using the Set Spline Interval command. Alternatively, you can change the points themselves so that the axis does not have to move as far.

When this error occurs, the velocity will be truncated at 65536 position units per second.

7.4.85 Unknown Parameter Error

Automatic parameter identification is a feature that has been added to the RMC in RMC100 CPU firmware dated 19971016 (year-month-day) and newer. If you have firmware older than this date and would like to use this feature, contact Delta Computer Systems, Inc. technical support and ask for the latest firmware. You can check your controller firmware date by selecting Module Configuration… from the Tools menu.

If your firmware is newer than the above date, and a parameter error is still not known, contact Delta Computer Systems, Inc. technical support with the parameter error number. It will most likely be necessary to obtain a newer version of RMCWin that recognizes the new parameter error number.

Otherwise, you can attempt to trace down the error by looking at the command that generated the parameter error and compare your usage of the command with the help topic for the particular Command. To clear a parameter error, issue a Set Parameters command to the axis. If this does not clear the error, then one of the sixteen axis parameters has an invalid value.

Appendix A: Command Reference

A.1 General ASCII Commands

A.1.1 I-PD Position Move Command

Character: !
Decimal: 33
Hexadecimal: 0x21
Command Value: Requested Position, in position units

The I-PD Position Move command initiates a position move using the I-PD control algorithm. All other motion commands in the RMC100 use the PID control algorithm.

The Acceleration and Deceleration fields are not used. The Speed field is the maximum velocity allowed during the move. The Command Value is the requested position.

I-PD Explained

The I-PD control algorithm does not generate a target motion profile like the PID. Instead, the Target Position is always the same as the Command Position. This causes a step jump in the Target Position, which would normally cause a large step jump in the drive with the PID. The I-PD does not cause a large jump in the drive.

I-PD moves are simpler than PID moves, but are not as precise in terms of specifying the profile or the time it will take to reach final position.

Using I-PD to Follow Another Axis

The I-PD can be used to follow another axis. The I-PD will lag slightly behind the master during motion, but will get to the same position as the master when stopped. The I-PD can only follow the Actual or Target Position of the master with an offset. One suitable use for the I-PD is following a moving axis when the initial difference between the master and slave positions is large.

To follow an axis:

- Select the Gearing bit in the Mode word and select whether to gear to the master's Actual or Target Position.
- The Accel, Decel, and Speed fields are not used.
- Specify the offset in the Command Value field.
- Issue the I-PD command.

Tuning I-PD Motion

The I-PD uses the Proportional, Integral and Differential gains, but not the Feed Forwards or Accel Feed Forwards. The gains determine how quickly the I-PD gets into final position, as described below.

Use the following tips when tuning I-PD motion:

- The Proportional Gain works the same as in the PID.
- The ratio of the Integral Gain to Proportional Gain determines the response. For a faster response (to get into position quicker), increase the Integral Gain/Proportional Gain ratio.
- The Differential Gain works the same as in the PID.
- Feed Forwards and Accel Feed Forwards are ignored.

Why Bother?

- The I-PD is simpler than the PID for basic moves where the goal is simply to get to position.
- Some gearing applications may benefit from the I-PD, as explained above.

A.1.2 Set Count Offset Command

Character: #
Decimal: 35
Hexadecimal: 0x23
Command Value: Count Offset

Note: This command is supported in RMC100 CPU firmware dated 20030403 or later.

This command is used to offset the Transducer Counts on SSI and Analog Velocity axes, as explained below. This Count Offset value can be saved to Flash with the Update Flash (U) command and will then be valid on the axis until changed by another Set Count Offset (35) command.

SSI Axes:

This command offsets the Transducer Counts (before it is converted to position units) by the amount specified in the Command Value (32767 to +32768). This is useful on SSI rotary absolute axes for setting the zero point of the axis.

Analog Velocity Axes (control and reference):

This command offsets the Transducer Counts (before it is converted to velocity units) by the amount specified in the Command Value (32767 to +32768). This is useful for setting the zero velocity point of an axis with tachometer feedback.

Example:

A tachometer is wired to an RMC100 analog input. The Transducer Counts field (the Counts field in the Status area of RMCWin) shows -150 when the tachometer is not rotating. This incorrect feedback will cause the axis to rotate after issuing a command for 0 velocity.

Issuing a Set Count Offset (#) command with a command value of 150 results in the Counts field correctly showing 0. The axis will now control properly.

A.1.3 Display LCD Screen Command

Character: $
Decimal: 36
Hexadecimal: 0x24
Command Value: LCD Screen Number (0-15)

Note: This command is supported in RMC100 CPU firmware dated 20010522 or newer.

This command will change which screen is currently displayed on the LCD Screen. As described in the LCD Screen Editor topics, there can be up to sixteen screens stored in a single RMC. This command allows the user to programmatically display a specific screen.

Only the Command Value is used. This field indicates the screen number (from 0 to 15) to be displayed. This command will interrupt any operations in progress on the LCD display such as the user being in the menu or entering a value in a field.

Uses for this command include reporting error conditions to the user and guiding the user through a process, step-by-step.

A.1.4 MulDiv Command

Character: '
Decimal: 39
Hexadecimal: 0x27
Command Value: Destination Address (80-2303)

Note: This command is supported in RMC100 CPU firmware dated 20011113 or newer.

Note: This command should only be used in the Event Step table. It is usually not useful to issue this command directly from the PLC.

This command is used to multiply, divide, or scale a value in the RMC. It reads a value from an RMC register, multiplies it by one constant, divides it by another constant, rounds the result, and stores it in another RMC register.

Tip: To use this command to multiply only, set the Speed field to 1. To use this command to divide only, set the Deceleration field to 1. Refer to the definitions of the command fields below.

The memory locations available for the source of this operation include any Status, Command, Parameter, or Event Step register. The memory locations available for the destination of this operation include any Command, Parameter, or Event Step register.

The addresses of the source and destination registers match the addresses shown in the RMC Register Map (PROFIBUS-DP Message Mode) topic, except that only addresses 0-2303 are usable by this command.

Tip: Use the Address Tool to greatly simplify entering addresses for this and other commands that use addresses. See Address Tool for details.

This command uses the command fields as follows:

Mode: The Mode field controls how the 16-bit source, destination, and constant values are sign-extended. Its bits are defined as follows:

Bits 7-15 Reserved. Must be 0.

Bits 4-6 Position Range Axis. These bits are used only if bit 1 (Register Sign Extension) is set. They then determine which axis's position range is used for the source and destination registers.

Bits 2-3 Reserved. Must be 0.

Bit 1 Register Sign Extension. If this bit is 1, then the source and destination registers are sign extended to fit the position range for the axis selected by Mode bits 4-6.

If this bit is 0, then the source and destination registers are sign extended the same way as the constants.

Bit 0 Constant Sign Extension. If this bit is 0, then the constants in the Deceleration and Speed fields are signed. Otherwise, these constants are unsigned.

Tip: To simplify computing and entering the Mode value, use the pop-up editor. First enter the command so that RMCWin knows which Mode dialog box to display. Then either double-click the mode value or press ENTER when the cursor is in the Mode field. You will then be able to select the options you want, and the Mode will be computed for you.

Acceleration: Source Address. This is the address (see above) of the register to start with. The register's value can be signed, unsigned, or a position value depending on Mode bits 0-1.

Deceleration: This is a constant value that is multiplied by the register specified by the Acceleration field before being divided by the constant in the Speed field. This constant can be signed or unsigned as determined by Mode bit 0.

Speed: This is a constant value that is divided into the product of the register specified by the Acceleration field and the constant in the Deceleration field. This constant can be signed or unsigned as determined by Mode bit 0.

Command Value: Destination Address. This is an address (see above) of the register to store the result in. Before the result is stored, it is rounded to the nearest integer and checked if it falls in the range selected by Mode bits 0-1 (signed, unsigned, or a position value). If the result does not fall within this range, the result is limited to this range and an internal error bit is set, which can be tested with the MathOK (0x12) and MathERR (0x13) link types.

Command: ' (39 decimal, 0x27 hexadecimal)

Although this command is limited to multiplying and dividing by constants, it is possible to multiply and divide by computed values by stringing other math commands together whose destinations are the Deceleration and/or Speed fields of this command. For example, the first event step might be an Add command with a 0 in the Speed command field to copy a field from the original RMC register into the Speed command field of the next event step. The next event step would then perform a "constant" MulDiv command, but because this constant had just been set to another register value, the MulDiv effectively divides one register by another.

Example:

Suppose that Axis1 needs to go to a position that is 90% of Axis0's Actual Position. This can be done using two Event Steps. The first will calculate 90% of the Axis0 Actual Position and store it in the second step's Command Value. The second step will issue a Go (G) command to the newly-calculated position. A step sequence that accomplishes this is shown below:

	Step 18	**Step 19**
Mode	0x0012	0x0081
Accel	(2)	100
Decel	900	100
Speed	1000	10000
Command Value	(412)	0
Command	'	G
Commanded Axes	Default	1
Link Type	MathERR	BitsON
Link Value	0	0x0001
Link Next	40	20

Step 18:

We use a MulDiv command to compute 90% of the Axis0 Actual Position and store it in Step 19 Command Value. We first need to find the source and destination register addresses. We recommend using the Address Tool to bookmark the Axis0 Actual Position and Step 19 Command Value on screen, although the RMC Register Map (PROFIBUS-DP Message Mode) help topic can be used to manually look up the addresses. Both methods show that address 2 (Axis0 Actual Position) should be entered in the Accel field, and address 412 (Step 19 Command Value) should be entered in the Command Value field.

Note: If these steps were moved around in the Event Step table (say to steps 20 and 21), the address that pointed to Step 19 Command Value (#412) would be automatically adjusted to point to Step 21 Command Value (#428).

The constants in the Decel and Speed fields are used to give 90%. That is 900/1000 is 0.9 or 90%. We could also have used 90 and 100 or 9 and 10. Because Mode bit 0 is zero, these values are signed, although for positive values under 32,767, this makes no difference.

Finally, notice that the range of the register values has been selected by Mode bits 0-1 and 4-6 to be the position range for Axis1. Therefore, if the result of the multiply and divide does not fall in the position range for Axis1, then the Math Error bit will be set. Notice that the value read from the Axis0 Actual Position is also treated as being in the Axis1 position range. Therefore, these axes must have similar position ranges.

As you can see, the MathERR link type is used to detect an overflow and jump to step 40 if there is such an error. If there is no error, then the MathERR link type links to the next step (step 19).

Step 19:

This step issues a Go (G) command to Axis1. The Command Value will have been overwritten by step 18 with a value that is 90% of the Axis0 Actual Position. This step then waits for the In Position bit to set before linking to step 20.

See also: Add Command, Subtract Command

A.1.5 Add Command

Character: +
Decimal: 43
Hexadecimal: 0x2B
Command Value: Destination Address (80-2303)

Note: This command is supported in RMC100 CPU firmware dated 20010402 or newer.

Note: This command should only be used in the Event Step table. It is usually not useful to issue this command directly from the PLC.

This command allows the user to add a constant value to a memory location in the RMC and store the result in another memory location in the RMC. The memory locations available for the source of this operation include any Status, Command, Parameter, or Event Step register. The memory locations available for the destination of this operation include any Command, Parameter, or Event Step register.

The addresses of the source and destination registers match the addresses shown in the RMC Register Map (PROFIBUS-DP Message Mode) topic, except that only addresses 0-2303 are usable by this command.

Tip: Use the Address Tool to greatly simplify entering addresses for this and other commands that use addresses. See Address Tool for details.

This command uses the command fields as follows:

Mode:
The Mode field controls how the 16-bit source, destination, and constant values are sign-extended. Its bits are defined as follows:

Bits 7-15
Reserved. Must be 0.

Bits 4-6
Position Range Axis. These bits are used only if bit 1 (Register Sign Extension) is set. They then determine which axis's position range is used for the source and destination registers.

Bits 2-3
Reserved. Must be 0.

Bit 1
Register Sign Extension. If this bit is 1, then the source and destination registers are sign extended to fit the position range for the axis selected by Mode bits 4-6.

If this bit is 0, then the source and destination registers are sign extended the same way as the constant.

Bit 0
Constant Sign Extension. If this bit is 0, then the constant in the Speed field is signed. Otherwise, this constant is unsigned.

Tip: To simplify computing and entering the Mode value, use the pop-up editor. First enter the command so that RMCWin knows which Mode dialog box to display. Then either double-click the mode value or press ENTER when the cursor is in the Mode field. You will then be able to select the options you want, and the Mode will be computed for you.

Acceleration:
Reserved (must be 0)

Deceleration:
Source Address. This is an address (see above) of the register to start with. The register's value can be signed, unsigned, or a position value depending on Mode bits 0-1.

Speed:
This is a constant value that is added to the register specified by the Deceleration field. This constant can be signed or unsigned as determined by Mode bit 0.

Command Value:
Destination Address. This is an address (see above) of the register to store the result in. Before the result is stored, it is checked if it falls in the range selected by Mode bits 0-1 (signed, unsigned, or a position value). If the result does not fall within this range, then an internal error bit is set, which can be tested with the MathOK (0x12) and MathERR (0x13) link types.

Command:
+ (43 decimal, 0x2B hexadecimal)

Although this command is limited to adding a constant to an RMC register, it is possible to add one register to another using two of these commands in a row. The first event step would use either an Add or Subtract command with a 0 in the Speed command field to essentially move a field from the original RMC register into the Speed command field of the next event step. The next event step would then perform a "constant" Add command, but because this constant had just been set to another register value, the Add command effectively adds one register to another.

See also: Subtract Command, MulDiv Command

A.1.6 Function Command

Character: ,
Decimal: 44
Hexadecimal: 0x2C
Command Value: Destination Step (0-255)

Note: This command is supported in RMC100 CPU firmware dated 20010402 or newer.

This command performs any of a number of functions on the Actual Positions of any group of axes. The result of the function is stored in the Command Value of the Event Step specified by the Command Value. The available functions include the following:

Function	Description
Min (0)	Uses the minimum Actual Position among the selected axes.
Max (1)	Uses the maximum Actual Position among the selected axes.
Mid (2)	Uses the middle Actual Position among the selected axes.
Average (3)	Uses the average of the selected axes' Actual Positions.

This command uses the command fields as follows:

Mode: Reserved (must be 0)

Acceleration: Reserved (must be 0)

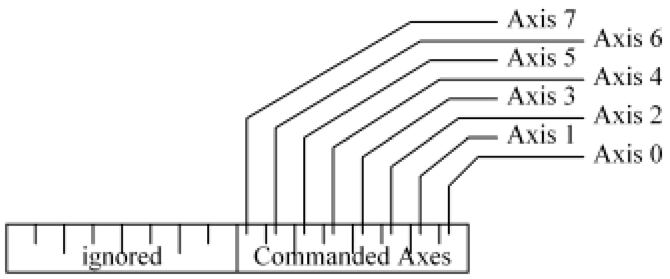
Deceleration: Function Number (see above)

Speed: Axes to Perform Function On (see below)

Command Value: Destination Step

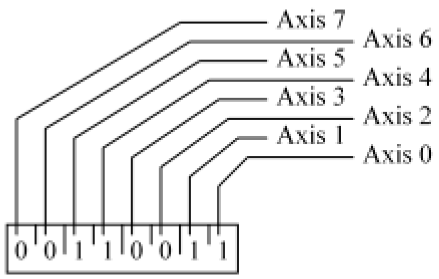
Command: , (44 decimal, 0x2C hexadecimal)

The axes to include in the function are selected using the Speed command field. Bits 0 to 7 correspond to axes 0 to 7, as shown below. Set the bit corresponding to each axis you want included in the function. Bits 8-15 of the Speed command field are ignored.



Example

The user wishes to select axes 0, 1, 4 and 5. Setting each of the corresponding bits to 1, as shown below, gives the binary value 00110011.



The binary value 00110011must be converted to hexadecimal or decimal so it can be put into the speed parameter. This binary value converts to the hexadecimal value 0x0033 or the decimal value 51.

Tip:
The Windows calculator is an easy way to convert from binary to decimal or hexadecimal. To open, click the Start button, point to Programs, then Accessories, and click Caculator. On the View menu click Scientific. Click Bin, enter the binary value, then click Hex or Dec to show the converted value.

A.1.7 Subtract Command

Character: -
Decimal: 45
Hexadecimal: 0x2D
Command Value: Destination Address (80-2303)

Note: This command is supported in RMC100 CPU firmware dated 20010402 or newer.

Note: This command should only be used in the Event Step table. It is usually not useful to issue this command directly from the PLC.

This command allows the user to subtract a constant value from a memory location in the RMC and store the result in another memory location in the RMC. The memory locations available for the source of this operation include any Status, Command, Parameter, or Event Step register. The memory locations available for the destination of this operation include any Command, Parameter, or Event Step register.

The addresses of the source and destination registers match the addresses shown in the RMC Register Map (PROFIBUS-DP Message Mode) topic, except that only addresses 0-2303 are usable by this command.

Tip: Use the Address Tool to greatly simplify entering addresses for this and other commands that use addresses. See Address Tool for details.

This command uses the command fields as follows:

Mode: The Mode field controls how the 16-bit source, destination, and constant values are sign-extended. Its bits are defined as follows:

Bits 7-15
Reserved. Must be 0.

Bits 4-6
Position Range Axis. These bits are used only if bit 1 (Register Sign Extension) is set. They then determine which axis's position range is used for the source and destination registers.

Bits 2-3
Reserved. Must be 0.

Bit 1
Register Sign Extension. If this bit is 1, then the source and destination registers are sign extended to fit the position range for the axis selected by Mode bits 4-6.

If this bit is 0, then the source and destination registers are sign extended the same way as the constant.

Bit 0
Constant Sign Extension. If this bit is 0, then the constant in the Speed field is signed. Otherwise, this constant is unsigned.

Tip: To simplify computing and entering the Mode value, use the pop-up editor. First enter the command so that RMCWin knows which Mode dialog box to display. Then either double-click the mode value or press ENTER when the cursor is in the Mode field. You will then be able to select the options you want, and the Mode will be computed for you.

Acceleration:
Reserved (must be 0)

Deceleration:
Source Address. This is an address (see above) of the register to start with. The register's value can be signed, unsigned, or a position value depending on Mode bits 0-1.

Speed:
This is a constant value that is subtracted from the register specified by the Deceleration field. This constant can be signed or unsigned as determined by Mode bit 0.

Command Value:
Destination Address. This is an address (see above) of the register to store the result in. Before the result is stored, it is checked if it falls in the range selected by Mode bits 0-1 (signed, unsigned, or a position value). If the result does not fall within this range, then an internal error bit is set, which can be tested with the MathOK (0x12) and MathERR (0x13) link types.

Command:
- (45 decimal, 0x2D hexadecimal)

Although this command is limited to subtracting a constant from an RMC register, it is possible to subtract one register from another using two of these commands in a row. The first event step would use either an Add or Subtract command with a 0 in the Speed command field to essentially move a field from the original RMC register into the Speed command field of the next event step. The next event step would then perform a "constant" Subtract command, but because this constant had just been set to another register value, the Subtract effectively subtracts one register from another.

See also: Add Command, MulDiv Command

A.1.8 Poll Command

Character: ?
Decimal: 63
Hexadecimal: 0x3F
Command Value: Extended Link Value

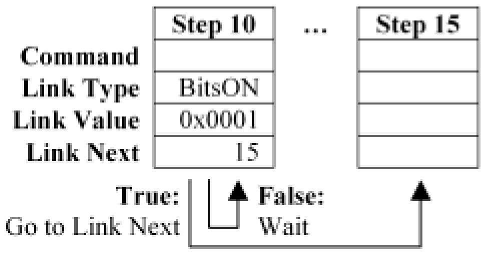
Note: This command is supported in RMC100 CPU firmware dated 20010420 or newer.

Note: This command should only be used in the Event Step table. This command does nothing when issued directly from the PLC.

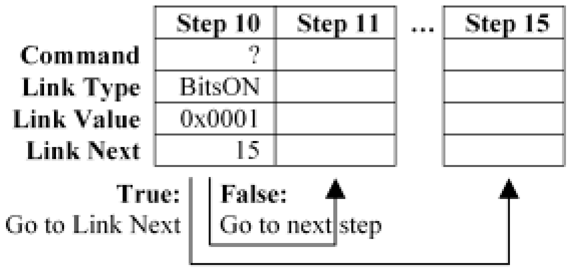
The Poll (?) command modifies the operation of most link types (exceptions are listed below). This command can only be issued in the Event Step table and affects the link type on the same step as the Poll command.

The Poll command is useful in applications where it is necessary to check for two or more conditions at the same time. For instance it may be required to start a move and then wait until the axis is in position or an error bit is set.

When a link type is encountered without this command, the sequence will wait until the condition is true before proceeding. No other steps are run on that axis's step sequence until the condition is met, as shown below:



When a link type is encountered with this command, then one of two steps will be run next. If the condition is true, then the step number indicated in the Link Next field will be run. If the condition is false, then the next step (current step plus one) will be run. This is shown below:



When the Poll command is executed, the Command Value is stored in the Extended Link Value, in the same way as in the Set Extended Link Value (l) command. This allows polling of link types that require the Extended Link Value to be set. Because most link types do not use the Extended Link Value, the Command Value can usually be left at zero (0).

Note: Only one polled link type is evaluated per control loop (1 or 2 ms).

The following link types should not be used with the Poll command:

- DelayMS (D) and DelayTicks (d) link types

These link types do not work with the Poll command because the timer or counter is restarted each time the link type is encountered in the sequence. An alternative for controlling time in a polled loop is to use the Timer (T and t) link types.

- InputRise (I) and InputFall (i) link types

These link types detect rising or falling edges in the current control loop. However, because a polled loop will take more than one control loop, rising or falling edges will likely be missed. Use the level-sensitive InputHigh (O) and InputLow (o) link types instead.

- LoopStart (#) and LoopEnd (-) link types

These links types already define both true and false actions, and therefore should not be used with the Poll command.

- Math Compares/Errors link types

These links types already define both true and false actions, and therefore should not be used with the Poll command.

- Jump (J) link type

This link type will already take a link immediately in all cases and therefore should not be used with the Poll command.

Example 1:

The user wants to make a move to position 4000, and wants to know when either the axis is in position or an overdrive error occurs. The following Event Step sequence accomplishes this using the Poll (?) command:

	Step 10	**Step 11**	**Step 12**	**Step 13**	**…**	**Step 15**
Mode	0x0081	0x0081	0x0081	0x0081		0x0000
Accel	100	100	100	100		0
Decel	100	100	100	100		0
Speed	10000	10000	10000	10000		0
Command Value	4000	0	0	0		0x0001
Command	G	?	?			[
Commanded Axes	Default	Default	Default	Default		Default
Link Type	DelayMS	BitsON	BitsON	DelayMS		DelayMS
Link Value	0	0x1000	0x0001	0		0
Link Next	11	15	14	11		0

Step 10 starts the move to 4000, and links immediately using the DelayMS (D) link type to step 11.

Step 11 checks for an overdrive error. It does this by doing a polled BitsON (B) link type for the Overdrive Error bit (0x1000) of the Status word on the current axis. If this error bit is set, then control goes to the step indicated by the Link Next: step 15. Otherwise, control proceeds to the next step: step 12. Notice that without the Poll (?) command, this link type would wait indefinitely until an overdrive error occurs and would otherwise never proceed to step 12.

Step 12 checks the In Position bit (0x0001) of the Status word, also using the BitsON (B) link type. So, if the axis is in position, control passes to step 14, otherwise control passes to the next step: step 13.

Step 13 links immediately back to step 11 to repeat the polling loop. Notice that this extra step does take one control loop to process and therefore increases the time taken to process the entire polling loop by one control loop (1 or 2 ms).

Step 15 will be jumped to when an Overdrive Error occurs. It simply turns on discrete output 0, perhaps to indicate the overflow to the operator in the form of a red light.

To summarize, the above sequence starts a move to position 4000, jumps to step 15 if an overdrive error occurs, jumps to step 14 if the axis gets in position, and loops through steps 11-13.

Now, compare this method of implementing a polling loop with the one used in the next example, which is a little more difficult to follow but removes the need for the extra do-nothing step in the polling loop (step 13 above).

Example 2:

This example offers an alternative approach to the same problem as in the previous example: move to position 4000, detect either when the axis is in position or when an overdrive error occurs. It is a little more difficult to follow, but removes one step from the polling loop, therefore reducing the amount of time taken to catch any condition. Keep in mind that for many applications, this slight gain in performance will not justify the reduction in clarity:

	Step 10	**Step 11**	**Step 12**	**…**	**Step 15**
Mode	0x0081	0x0081	0x0081		0x0000
Accel	100	100	100		0
Decel	100	100	100		0
Speed	10000	10000	10000		0
Command Value	4000	0	0		0x0001
Command	G	?	?		[
Commanded Axes	Default	Default	Default		Default
Link Type	DelayMS	BitsON	BitsOFF		DelayMS
Link Value	0	0x1000	0x0001		0
Link Next	11	15	11		0

Step 10 starts the move to 4000, and links immediately to step 11.

Step 11 checks for the Overdrive Error bit (0x1000) of the Status word on the current axis. If this error bit is set, control goes to step 15, otherwise control goes to step 12.

Step 12 checks the In Position bit (0x0001) of the Status word. Notice that, unlike the previous example, the BitsOFF (b) link type is used instead of the BitsON (B) link type. Therefore, if the In Position bit is off—that is, the axis is not yet in position—then the Link Next is taken back to step 11 and the polling continues. Otherwise, if the In Position bit is not off—that is, it is on and the axis is in position—then control proceeds to the next step: step 13.

Step 15 will be jumped to when an Overdrive Error occurs. It simply turns on discrete output 0, perhaps to indicate the overflow to the operator in the form of a red light.

To summarize, the above sequence starts a move to position 4000, jumps to step 15 if an overdrive error occurs, and otherwise jumps to step 13 when the axis gets in position.

Here is a summary of the advantages and disadvantages of doing a polled loop using this method versus the method shown in Example 1:

- The advantage of this method is that both conditions will be checked every two control loops since there are only two steps in the polling loop (steps 11 and 12).
- The disadvantage of this method is that many users will find it confusing to have to reverse the sense of the last condition in the polling loop. That is, the user has to use the BitsOFF (b) link type instead of the more intuitive BitsON (B) link type.

A.1.9 Arm Home Command

Character: @
Decimal: 64
Hexadecimal: 0x40
Command Value: Home Position

Note: This command is available on quadrature controlled axes only. This includes quadrature axes with servo or stepper output.

This command is issued to arm the home input before homing the axis. The axis must have the home input armed to detect and report a home condition. When both the Home (H) and Index (Z) inputs are active and the home is enabled, the axis is in the Home Active state. When either of these signals go inactive, the axis position is set to the home position specified by the Command Value of the @ command. This gives software the final say over when the position gets reset. It is recommended that homing is always done in the same direction and at the same speed for repeatability.

The Home Status bit will go on only if the home input is armed. For details on homing, see Homing a Quadrature Axis.

A.1.10 Change Acceleration Command

Character: A
Decimal: 65
Hexadecimal: 0x41
Command Value: New Acceleration value (units depend on Accel/Decel Mode in the Mode word)

This command sets the Acceleration parameter to the value in the Command Value field. The change in acceleration takes place immediately if a move is in progress.

A.1.11 Amp Enable/Disable Command

Character: a
Decimal: 97
Hexadecimal: 0x61

Command Value: 0 = Disable AMP, 1 = Enable AMP

Note: This command is only available in RMC100 CPU firmware dated 19991124 or later.

This command is for Quadrature and Stepper axes only. It controls the Amp Enable output for the axis that receives this command. If the Command Value is 0, then this output goes low. If the Command Value is 1, then this output goes high.

A.1.12 Clear Spline Segments Command

Character: C
Decimal: 67
Hexadecimal: 0x43
Command Value: Described below

Because the RMC can store multiple spline segments at a time, it may be necessary to clear one or more of these segments from the RMC's memory. Use the command value to control how many spline segments you want to delete:

Command Value	Effect
0	Deletes all segments.
n (1 to 16)	Deletes the oldest n segments.
-n (-1 to -16)	Delete all but the most recently downloaded n segments. The actual number of segments deleted depends on how many segments were available. It is common to use a value of -1 to keep only the most-recently-downloaded spline segment.

Note: Negative command values are supported only in RMC CPU firmware dated 20000421 or later.

This command may not finish immediately. That is, it will clear one spline segment per control loop. Therefore, if you are clearing three segments on a module with a one-millisecond control loop time, this command will take three milliseconds to finish. The Acknowledge bit of the Status word will toggle when this command is completed.

Note: The Acknowledge bit toggles only if the command was issued over the communication module (PROFIBUS-DP, Modbus Plus, etc.) and not if it was issued from the Event Step table or RMCWin.

This command will fail for the following reasons, as indicated by the Parameter Error bit in the Status word:

- A spline is being followed at the time this command is issued. No segments will be deleted, and the Parameter Error bit will be set in the Status word.
- An invalid command value was specified; it must be between -16 and +16. No segments will be deleted, and the Parameter Error bit will be set in the Status word.

- You specified a positive command value, but not that many segments are available on that axis. All segments will be deleted, but the Parameter Error bit will be set to indicate that it did not delete as many segments as you requested. If you get this error, keep in mind that spline segments are automatically discarded after they are executed.
- You specified a negative command value, but not that many segments are available to keep. No segments will be deleted, and the Parameter Error bit will be set in the Status word.
- If multiple segments are being deleted, but another command is sent to this axis before the Acknowledge bit in the Status word has been toggled. This will not be reflected in the Parameter Error bit.

A.1.13 Set Position/Pressure Command

Character: c (lower case C)
Decimal: 99
Hexadecimal: 0x63
Command Value: Unused

Note: This command is supported in RMC100 CPU firmware dated 19980414 or later.

This command can be used in two modes, controlled by the Monitor Pressure bit in the Mode field:

Set Position (Monitor Pressure bit cleared)

This command sets the Command Position to the current Actual Position, and places the axis into closed loop control (thereby exiting open loop or pressure control). This results in the axis exiting open loop or pressure control and holding the current position. One practical use for this command is to follow it with a Relative Move command to back up a specified amount from the point where the axis exited pressure or open loop.

Set Pressure (Monitor Pressure bit set)

This command begins controlling pressure by holding the current pressure. Therefore the Command Pressure is set to the Actual Pressure and the axis enters pressure control mode. The Pressure Set A parameter is ignored as the threshold to enter pressure; however, the Actual Pressure must be above Pressure Set B when this command is issued or a Parameter Error will result (this is to avoid having the axis immediately exit pressure control). This command will generate a parameter error if the axis is already controlling pressure, as the actual Set Pressure command should be used to change the pressure.

A.1.14 Change Deceleration Command

Character: D
Decimal: 68
Hexadecimal: 0x44
Command Value: New Deceleration value (units depend on Accel/Decel Mode in the Mode word)

This command sets the Deceleration parameter to the value in the Command Value field. The change in deceleration takes place immediately if a move is in progress.

A.1.15 Start Events Command

Character: E
Decimal: 69
Hexadecimal: 0x45
Command Value: Event Step to Start Execution

This command starts an event step sequence at the step specified in the Command Value field. If the axis already was in the middle of an event sequence, that sequence will end and the new event sequence will begin.

Note: RMC100 CPU Firmware dating prior to 19980414 supports event sequences only on position axes. Issuing this command to a pressure or force axis will result in a parameter error.

A.1.16 Set Feed Forward Command

Character: F
Decimal: 70
Hexadecimal: 0x46
Command Value: Unused

The 'F' command is used to automatically set the feed forward values. After a move is made where the axis is allowed to reach constant velocity and the overdrive bit is not set, an 'F' command will set the Feed Forward for the direction last moved. This command is quick and easy, and it allows the system to adjust for changing system dynamics. This also makes setup easier.

Note: The 'F' command should be used only after the axis is moving smoothly. If the axis oscillates or doesn’t reach steady state during the move, this command will give erroneous results.
If the Null Drive is not adjusted correctly, the value for the Feed Forward will be incorrect.

A.1.17 Follow Spline Segment Command

Character: f
Decimal: 102
Hexadecimal: 0x66
Command Value: Unused

This command will begin following the next available spline segment. If there is only one spline segment loaded, then this segment will be repeated even if it has already been followed. See also the Follow Spline Relative command.

This command does not use Accel, Decel, Speed, and Command Value. These unused fields should be set to 0 for forward compatibility. However, the following bits of the Mode word are used: Graph Disable, Monitor Pressure, Rotational, and Geared bits. Each should be adequately described in the Mode word topic, but see the description below for more information on Gearing a spline.

The In Position bit of the Status will be cleared when this command begins and will be set when the axis reaches the end of the spline. The State A and State B status bits will be cleared throughout the spline.

The axis must already be at the position of the first spline point (the Follow Spline Relative command does not require this). This can be ensured by the programmable controller by giving a Go (G) command to the first location and waiting for the In Position status bit to be on or for State A and State B status bits to both be off. This is left as the user's responsibility to allow the user to control the manner that the axis moves to the first location.

This command can fail for the following reasons, as indicated by the Parameter Error bit in the Status word:

- The axis is not stopped when the command was issued. The axis is stopped when State A and State B status bits are off.
- The axis is not at the location of the first point in the spline. The Target Position must be equal to the Command Position which must be equal to the first position of the spline.
- There is no spline segment available to follow.

The rate at which the axis moves through the spline is determined in the following manner:

- If the Mode word has the Geared Mode bit set, the spline is geared to another axis. Gearing a spline is most useful when geared to an axis in speed control, or an independent quadrature input.

Note: In RMC100 CPU firmware dated prior to 19990625 (or beta firmware dated prior to 19980827B) the Geared Mode bit is ignored and therefore assumed to be cleared.

The gearing relationship works as follows:

  - The master axis is given by the Gear Master Select bits in the Mode word.
  - The slave axis may be geared based on the master axis's actual or target position. Select this with the Gear Type bit in the Mode word.
  - Once the gearing has been set up, each position unit that the master axis increases results in the slave axis moving forward one time unit in the spline. Similarly, when the master axis position decreases by a position unit, the slave axis moves backward one time unit in the spline.

- If the Mode word is cleared and the module has a Sensor DI/O with an Edge or Quadrature counter enabled, the spline is geared to this counter. Each time interval represents one counter tick. For a quadrature counter, reversing the direction of the quadrature counts reverses the direction the spline is interpolated (forward vs. backward).
- If the Mode word is cleared and the module does not have an enabled Sensor DI/O counter, the spline is interpolated at a fixed rate based on time. Each time interval represents one millisecond.

Addendum for Special Firmware 'r;SI' (e.g. 19980827SI)

This firmware is available only on request, as it useful only for special applications. This firmware includes superimposed mode. The above description of this command is appended in the following ways:

- The Geared Mode bit is used, despite the note about firmware versions.
- The Superimposed Mode bit is used as follows:
  - If set, the move may be superimposed on top of a Geared move, a Speed Control move, a Sine Move, or another Spline. If cleared, the spline will be the only move. Notice, that if the axis is stopped, the Superimposed bit may still be used without harm.
  - If set, the Status word's State A and B bits will not be affected. However, the In Position bit will be affected. If cleared, the State A and B bits will be cleared during the spline.
  - If set, the entire spline is performed as relative to the current position. Therefore, the entire spline is adjusted to start at zero position, and then the curve is added to any other motion being performed on the axis. If cleared, the spline is assumed to be absolute and the axis must be at the first spline position before beginning the spline. This can be ensured by first issuing a Go (G) command to the first spline location.

A.1.18 Go Command

Character: G or g
Decimal: 71 or 103
Hexadecimal: 0x47 or 0x67
Command Value: Requested Position, in position units

The Go command will initiate a position move, a speed control move, or a new gear ratio. The user must make sure that all parameter words are valid when the Go command is issued. Normally, once the Mode, Acceleration, Deceleration, and Speed are set, only the Command Value needs to be changed.

Note: The 'r;G' command can be given while the axis is in motion. If you do this, the RMC will ramp to the new speed at the rate specified by the Acceleration and Deceleration parameters.

Bits in the Mode word select the type of move:

Position Control (Rotational bit cleared):

This command moves the axis to the position specified in the Command Value field.

Speed Control with Position Loop (Rotational bit set):

This command requests the axis to ramp up or down to the new requested speed in the Speed field. The Command Value field indicates the direction the axis should move: 0 represents the direction of increasing counts (extend or clockwise), -1 represents the direction of decreasing counts (retract or counter-clockwise). See Speed Control for details on this mode.

The direction can be changed while the axis is moving; the axis will ramp down to a stop and then immediately ramp up to the new speed in the new direction.

Speed Control with Velocity Loop (Rotational bit set):

This command is identical to Speed Control with Position Loop except that closed loop control is performed on the speed, not the position. See Speed Control for details on this mode.

Geared Control (Gear bit set):

This command requests the axis to ramp its gear ratio up or down. The requested gear ratio is held in the Command Value and Speed fields. Please see Gearing Axes for details on using the Go command for gearing.

A.1.19 Halt Command

Character: H or h
Decimal: 72 or 104
Hexadecimal: 0x48 or 0x68
Command Value: Unused

Note: A Go command with zero speed is typically a way to stop an axis than using the Halt command. See below for details.

The Halt command starts an immediate deceleration to a stop in closed loop. It uses the previous deceleration rate specified for the axis. This command also disables the integrator and the null update so the integrator does not wind up if the hydraulic pump is turned off or the motor drive disabled.

This command should only be used with PROFIBUS Compact mode communications. Other communication types should use a Go command with a Speed parameter of zero. This will immediately decelerate the axis to a stop. It gives you control over the deceleration rate (Decel parameter) and the integrator mode (Mode paramter). When using the Go command with zero speed, the specified position is not important as long as it is within the extend and retract limits.

If an event sequence is in progress when the Halt command is issued, the event sequence will be stopped. Compare this command with the Disable Drive Output and Quit Events commands.

A.1.20 Set Integral Drive Command

Character: I (Upper case i)
Decimal: 73
Hexadecimal: 0x49
Command Value: New Integral Drive, in millivolts

This command sets the Integral Drive to the value in the Command Value field. This command can be used to unwind the integrator. There are several alternative ways to unwind the integrator. See also the Set Integral Drive to Null Drive, Save Integral Drive and Restore Integral Drive commands.

A.1.21 Set Integral Drive to Null Drive Command

Character: i
Decimal: 105
Hexadecimal: 0x69
Command Value: Unused

This command sets the Integral Drive to the Null Drive value. This can be used to unwind the integrator. In most cases this is more desirable than using the Set Integral Drive command because this accounts for the valve being non-nulled. You can also use the Save Integral Drive and Restore Integral Drive commands to unwind the integrator.

A.1.22 Relative Move Command

Character: J or j
Decimal: 74 or 106
Hexadecimal: 0x4A or 0x6A
Command Value: Change in Position, in position units.

This command changes the Command Position by the amount specified in the Command Value field. For example, suppose that one position unit is equal to 0.001" and the current position is 5000 (5.0"). If a J command is given with a Command Value of -1000, then the axis will move to 4000 (4.0").

A.1.23 Disable Drive Output Command

Character: K
Decimal: 75
Hexadecimal: 0x4B
Command Value: Unused

This command immediately sets the drive output on the module to the current null value. This is equivalent to a Hard Stop. The output will remain at null in open loop control until a new command is issued. This command will clear any error bits that can be cleared.

When this command is issued from RMCWin, it is issued to all axes simultaneously. When issued from the controller, only the axis receiving this command is affected.

If an event sequence is in progress when this command is issued, the event sequence will be stopped. Compare this command with the Halt and Quit Events commands.

A.1.24 Limit Drive Command

Character: L
Decimal: 76
Hexadecimal: 0x4C

Command Value: Drive Limit, in millivolts

Note: This command is available only in RMC CPU firmware dated 20000331 or later.

This command sets the drive limits for the axis receiving the command. The limits are set to plus or minus the Command Value in millivolts. For example, issuing this command with a Command Value of 5000 will limit the drive output for the axis to ±5000 mV. This command can only be issued to axes with analog outputs. That is, it is ignored on axes that have no outputs or have stepper outputs.

By default, all analog drive outputs are limited to ± 10,000 mV.

A.1.25 Set Extended Link Value Command

Character: l (lower case L)
Decimal: 108
Hexadecimal: 0x6C
Command Value: Extended Link Value

Note: This command is available only in RMC CPU firmware dated 20000504 or later.

This command is used to set the extended link value, which is currently used only by the Skew Detection (<) and CommTrig (C) link types. Refer to those link type topics for details on how this value is used.

A.1.26 Set Mode Command

Character: M
Decimal: 77
Hexadecimal: 0x4D
Command Value: New Mode value

This command is used to change the Mode for the current command in one of several ways. The changes take place immediately. The allowed bits (described below) are copied from the Command Value to the Mode field for this axis. The following Mode bits can be changed with this command:

- **Gearing Type Bit (bit 14) and Gear Master Select Bits (bits 4-6):**
If the axis is already gearing to another axis, then these bits can be changed to switch immediately to gearing to another axis or position quantity. Changing these bits when the axis is not already geared will have no effect. The gearing bits cannot be changed if the axis is gearing in mode 2.

- **Monitor Pressure Bit (bit 8):**
Clearing this bit while regulating pressure will drop the axis out of pressure regulation. Changing this bit at any other time will simply make the axis stop or resume monitoring the pressure for entering pressure control.

- **Rotational Bit (bit 9):**

Changing this bit puts the axis into or pulls it out of Rotational mode, as described in Rotational Mode. Changing this bit while the axis is doing a Point-to-Point, Synchronized, Quick, or Speed Control move will have no effect, since Point-to-Point, Quick, and Synchronized moves are only defined in non-Rotational mode and Speed Control is only defined in Rotational mode. It is allowed in Open Loop, Stopped, Spline, Gear, and Reference control modes.

- **Pressure Mode Bits (bits 0-1)—PRESSURE AXES ONLY:** If these bits are changed while regulating pressure, the axis will come out of pressure regulating mode, and return to pressure monitoring mode. Depending on the conditions, the axis may immediately re-enter pressure regulating mode, but there will be a discontinuity. For this reason, it is not recommended that the mode be changed while regulating pressure.

A.1.27 Set Null Drive Command

Character: N
Decimal: 78
Hexadecimal: 0x4E
Command Value: Millivolts of Drive

The 'N' command sets the Null Drive to the value in the Command Value field.

A.1.28 Set Null Drive to Integral Drive Command

Character: n
Decimal: 110
Hexadecimal: 0x6E
Command Value: Unused

This command sets the Null Drive to the current Integral Drive value.

Why Bother?
It is important that the Null Drive be set because it is used in open loop mode and hard stops. The ’n’ command should only be issued when the axis is in position and its speed is zero.

A.1.29 Open Loop Command

Character: O
Decimal: 79
Hexadecimal: 0x4F
Command Value: Millivolts of Drive

CAUTION: Use this command with care! Open Loop operation disables all safety features on the RMC!

Note: The open loop command will not affect drive output while the Simulate bit is set in the Config word. Drive output is always 0 volts in simulate mode.

Note: For pressure/force axes, this command will take effect on the analog module's drive output, if one is available. Only 16-bit analog cards have drive outputs, therefore this command will have no affect on 12-bit analog cards. Also, because drive outputs are only assigned to input channels 0 and 2 on 16-bit cards, only analog or pressure axes using those input channels will be able to issue this command. Otherwise, this command behaves normally on pressure/force axes, except that no status registers other than the Drive field are modified by this command.

The Open Loop command allows the Controller to directly specify values for the analog output. The output range is -10000 to 10000 and is given in millivolts. However, the drive outputs are not precision outputs. You cannot rely on the output being exactly the value you specify. Look up specifications in the online help index for the tolerance on your particular interface module's drive outputs.

The O command uses the following parameters from the last open loop profile specified:

The Command Value field specifies the desired drive in millivolts to output. The current null drive is added to this value.

The Speed field is not used.

The Acceleration and Deceleration fields control the rate at which the drive output ramps to the requested value. Acceleration is used when the drive output is moving away from 0 and deceleration is used when drive output is moving toward 0. The actual meaning of the values depends on the Acceleration/Deceleration mode bits in the Mode word. Notice that only modes 1, 2 and 3 are used:

Mode 1
Acceleration and deceleration are given in millivolts per millisecond. For example, if the current drive output is 0 millivolts, and the Command value is 1000 millivolts, and the Acceleration is 50, then it will take 20 milliseconds for the drive output to reach 1000 millivolts.

Mode 2
Acceleration and deceleration are given as distances. That is, the drive will be ramped over the distance specified. This mode is most useful when stopping on poor performing hydraulics. This mode cannot be used for starting from a stop because no drive will be applied since the position does not move, and conversely, the position does not move because there is no drive. For example, if the current drive output is 1000 millivolts, and the Command value is 0 millivolts, and the Deceleration is 4000, then the axis will ramp the drive down to 0 in the time it takes the axis to move 4000 position units.

Mode 3
Acceleration and deceleration are given in milliseconds. For example, if the current drive output is 0 millivolts, the Command value is 1000 millivolts, and the Acceleration is 50, then it will take 50 milliseconds for the drive output to reach 1000 millivolts, as the drive will increase 20 millivolts each millisecond.

Note: If a move crosses 0 drive, then Deceleration will be used until the drive reaches 0, and then Acceleration will be used until the final drive is reached.

Tip: If you are going to be changing from closed loop to open loop, you must be aware of the following. While in closed loop and holding a position, the drive may be switching back and forth between small positive and negative drives. If an open loop command is given to go to a non-zero drive, the starting drive may have a sign opposite of the requested drive. In this case, the drive will decelerate down to zero millivolts and then accelerate up to the new drive. For example, if the current drive is -10 millivolts and an open loop command requests a drive of 2000 millivolts (2V), the drive will ramp from -10 to 0 and then accelerate from 0 to 2000mV. To ensure that this deceleration happens quickly, the deceleration must be set properly (by setting it to 0 in Mode 3, and to a high number (1000 or greater) in Mode 1).

Stepper Axes (QST Modules)

Stepper axes do not have an analog output so the Open Loop command generates pulses on the Step+ and Step- Outputs. It also sets the polarity on the Direction+ and Direction- signals.

The pulse frequency is specified in the Command Value register as the number of pulses per millisecond.

The Acceleration and Deceleration parameters specify the rate at which the Open Loop output value is changed. See the discussion above for a description of the modes available.

A.1.30 Set Parameters Command

Character: P or p
Decimal: 80 or 112
Hexadecimal: 0x50 or 0x70
Command Value: Unused

When a 'P' command is given all initialization parameters are updated.

Tip: If you wish to change parameters without stopping the axis, see the Set Parameter On-the-Fly commands.

Non-pressure/force Axes

The minimum requirement of this command is to set the Extend and Retract Limits to their proper values (see Start-Up and Tuning). When a 'P' command is given, the RMC will copy the Actual Position of the axis into the Target and Command Positions. Therefore the axis will be in closed loop control around its current position.

Pressure/Force Axes

When this command is received, the RMC will copy the Actual Pressure of the axis into the Target Pressure status field.

A.1.31 Quit Events Command

Character: Q
Decimal: 81
Hexadecimal: 0x51
Command Value: Unused

This command stops the event control sequence on the axis the command is issued to. Any motion commands in progress on the axis are not stopped. Therefore, if an axis is currently moving to 4.000”, it will continue moving to that position; if an axis is in open loop with 1.000V drive, the drive will continue holding that drive, etc. If you desire to stop the axis at the same time, look at the Halt and Disable Drive Output commands.

Note: RMC100 CPU firmware dating prior to 19980414 supports event sequences only on position axes, issuing this command to a pressure or force axis will result in a parameter error.

A.1.32 Reset Position Command

Character: q
Decimal: 113
Hexadecimal: 0x71
Command Value: New Position
Command Value (Resolver): Not used

This command is used to put a quadrature encoder axis into a known state. The COMMAND POSITION, TARGET POSITION and ACTUAL POSITION are set to the value specified in the Command Value register. The axis is placed in the closed loop position control state and error bits in the status word are cleared.

Note: If the axis is in motion or in Open Loop before this command is given, it will come to an immediate stop and hold the specified position. If the axis is controlling pressure, the positions will be reset but the pressure control will not be interrupted.

Resolver Axes

This command will determine the axis absolute position within one turn of the resolver and set the Actual Position to that value. If the position had previously incremented through several turns, all the turns will be unwound (without causing motion). If the axis is moving when this command is issued, the axis will come to a stop and will hold position in closed loop position control.

See the Resolver Overview topic for a description of how the absolute position is determined.

A.1.33 Restore Null Drive Command

Character: R
Decimal: 82
Hexadecimal: 0x52
Command Value: Unused

This command restores the last saved value of null. This value will be 0 if no previous Save Null Drive was issued.

A.1.34 Restore Integral Drive Command

Character: r
Decimal: 114
Hexadecimal: 0x72
Command Value: Unused

This command restores a previously saved Integral Drive value. The Integral Drive value that is restored is taken from a buffer internal to the RMC that is set using the Save Integral Drive command. The Integral Drive will be set to 0 if no previous Save Integral Drive command was issued. The Save/Restore Integral Drive pair can be used to unwind the integrator while in the middle of a move.

For example, suppose the integral drive is saved while the move is taking place. If during the move, the axis gets stuck and the integrator winds up, then the integral drive can be restored after the cause of the stall is fixed to avoid an overshoot.

For alternative ways to unwind the integrator, see the Set Integral Drive and Set Integral Drive to Null Drive commands.

A.1.35 Save Null Drive Command

Character: S
Decimal: 83
Hexadecimal: 0x53
Command Value: Unused

The Save Null Drive command saves the current value of the null so it can be recalled later with a Restore Null Drive command ('R').

A.1.36 Save Integral Drive Command

Character: s (lower case S)
Decimal: 115
Hexadecimal: 0x73
Command Value: Unused

This command saves the current value of the Integral Drive so it can be recalled later with a Restore Integral Drive command ('r'). For details on the uses of these two commands, see Restore Integral Drive.

A.1.37 Set Spline Interval/End Segment Command

Character: T
Decimal: 84
Hexadecimal: 0x54
Command Value: 0 to end segment, otherwise point interval

Set Spline Interval (non-zero command value)

If the command value is non-zero, then this command stores that command value as the current interval between spline points. The units for this interval depend on the presence of the Sensor Digital I/O. If the Sensor Digital I/O is present, then the interval is in terms of counter ticks (either quadrature or edge counter). If it is not present, then the interval is in terms of milliseconds. As spline points are added, they use the current interval value as the time between the added point and the next one to be added. The minimum value for an interval is 5 units and the maximum interval is 65535 units.

This command will fail only when a value between 1 and 4 is used. These small intervals tend to create unrealistic accelerations and are therefore not allowed. This error is indicated by the Parameter Error bit in the Status word:

End Spline Segment (zero command value)

When the command value is zero, this command will perform final calculations on the spline points in a segment. After each segment is added, a 'r;T' command with a command value of 0 must be sent. Any points sent after this 'r;T' command will belong to the next spline segment.

When used to perform final calculations, this command may not finish immediately. This command will process up to four points immediately. If more points are in the segment, then an additional control loop will be required for every seven points in the segment. Therefore, a 50-point segment on a module with a one-millisecond control loop time will require eight (8) milliseconds to process. The Acknowledge bit of the Status word will toggle when this command is completed.

This command will fail when using a zero command value for the following reasons, as indicated by the Parameter Error bit in the Status word:

- No spline points have been added since the last spline segment.
- Only 1 spline point has been added since the last spline segment.
- The spline generated an unrealistic acceleration or speed that would have caused an overflow. If this happens the spline segment is invalid. The limit for speed is from 0 to 65535 position units per second. The limit for acceleration is -512 to 511 position units per second per second. If position units are thousandths of inches then this would translate to 511 inches per second per second, or about 1.25 G's.
- Another command is issued before the Acknowledge Bit of the Status word is toggled. Notice that this is not indicated by the Parameter Error bit.

A.1.38 Set Spline Interval/End Segment Command

Character: T
Decimal: 84
Hexadecimal: 0x54
Command Value: 0 to end segment, otherwise point interval

Set Spline Interval (non-zero command value)

If the command value is non-zero, then this command stores that command value as the current interval between spline points. The units for this interval depend on the presence of the Sensor Digital I/O. If the Sensor Digital I/O is present, then the interval is in terms of counter ticks (either quadrature or edge counter). If it is not present, then the interval is in terms of milliseconds. As spline points are added, they use the current interval value as the time between the added point and the next one to be added. The minimum value for an interval is 5 units and the maximum interval is 65535 units.

This command will fail only when a value between 1 and 4 is used. These small intervals tend to create unrealistic accelerations and are therefore not allowed. This error is indicated by the Parameter Error bit in the Status word:

End Spline Segment (zero command value)

When the command value is zero, this command will perform final calculations on the spline points in a segment. After each segment is added, a ’r;T’ command with a command value of 0 must be sent. Any points sent after this ’r;T’ command will belong to the next spline segment.

When used to perform final calculations, this command may not finish immediately. This command will process up to four points immediately. If more points are in the segment, then an additional control loop will be required for every seven points in the segment. Therefore, a 50-point segment on a module with a one-millisecond control loop time will require eight (8) milliseconds to process. The Acknowledge bit of the Status word will toggle when this command is completed.

This command will fail when using a zero command value for the following reasons, as indicated by the Parameter Error bit in the Status word:

- No spline points have been added since the last spline segment.
- Only 1 spline point has been added since the last spline segment.
- The spline generated an unrealistic acceleration or speed that would have caused an overflow. If this happens the spline segment is invalid. The limit for speed is from 0 to 65535 position units per second. The limit for acceleration is -512 to 511 position units per second per second. If position units are thousandths of inches then this would translate to 511 inches per second per second, or about 1.25 G’s.
- Another command is issued before the Acknowledge Bit of the Status word is toggled. Notice that this is not indicated by the Parameter Error bit.

A.1.39 Teach Step Command

Character: t
Decimal: 116
Hexadecimal: 0x74
Command Value: Event Step to Teach

The Teach Step command takes the current Target Position for the axis and places that value in the Command Value field of the step given by the Command Value field. This is used for building a system that can teach its own step table positions. See the Teach Mode Overview topic for details.

A.1.40 Update Flash Command

Character: U
Decimal: 85
Hexadecimal: 0x55
Command Value: Unused

This command instructs the RMC to write all parameters, profiles, configuration data, Input-to-Event table entries, Events Steps, and LCD screens to Flash memory for storage in case of power loss or reset. While a Flash update is in progress, the green CPU LED will flash. Removing power while the LED is still flashing will result in the parameters being lost.

Note: The Update Flash command does not write the splines to Flash. For details on writing splines to Flash, see the Update Flash Segment command.

A.1.41 Update Flash Segment Command

Character: u
Decimal: 117
Hexadecimal: 0x75
Command Value: Segment to Write

This command instructs the RMC to write a certain segment or segments to Flash memory for storage in case of power loss or reset. While a Flash update is in progress, the green CPU LED will flash. Removing power while the LED is still flashing will result in the parameters being lost.

The Command Value determines which segments will be written to Flash:

Command Value	Segment to Write
0	All segments except splines.
1	Splines only.

For details on writing all segments except splines to Flash, see the Update Flash Command.

A.1.42 Set Speed (Unsigned) Command

Character: V
Decimal: 86
Hexadecimal: 0x56
Command Value: New Speed Value

This command sets Speed to the value in the Command Value field. This will cause the current move to ramp up or down to the new speed. This command does not change the Command Position of the axis.

If you wish to simply jog in one direction or another under closed loop control, see the Set Speed (Signed) command.

A.1.43 Set Speed (Signed) Command

Character: v
Decimal: 118

Hexadecimal: 0x76
Command Value: New Speed Value (Signed)

This command sets the Speed of the axis to the Command Value, which is a value between -32,768 and 32,767. In addition, the axis is given a Command Position of either the extend or retract limit depending on the sign of the Command Value.

If the Command Value is positive, the axis will move in the direction of increasing position units. If the Command Value is negative, the axis will move in the direction of decreasing position units.

If you wish to change only the speed of a move in progress, you may wish to use the Set Speed (Unsigned) command instead.

A.1.44 Reference Command

Character: W
Decimal: 87
Hexadecimal: 0x57

Note: This command is available only in RMC CPU firmware dated 20020620 or later.

The Reference (W) command places the axis in the reference state, and configures the position filter used while in this state. The reference state is defined as follows:

- The Actual Position status field reflects the actual reading from the transducer, and the Target Position reflects the filtered position, after applying the Filter Time Constant, Reference Deadband, Velocity Limit, and Acceleration Limit parameters described below. Axes gearing to this reference axis should select to gear to the reference axis's Target Position to use the filtered position.
- The drive output is in open loop. It can be changed normally using Open Loop (O) commands. When the Reference (W) command is issued, the axis will switch from closed to open loop, if necessary.
- While in the reference state, all motion commands for this axis are disabled. For example, Go (G) and Relative Move (J) commands will generate an Invalid Command parameter error but otherwise be ignored.
- To exit the reference state, you must issue a Set Parameters (P) command, which will place the axis back into closed loop, disable the position filter, and re-enable motion commands.

Analog inputs configured as Position Reference or Velocity Reference axes are always in the reference state. Therefore, the Set Parameters (P) command will not exit the reference state. These axes have axis parameters dedicated to the position filter parameters: Filter Time Constant, Velocity Limit, Acceleration Limit, and Reference Deadband. These parameters can be edited like any other axis parameter or set through the Reference (W) command.

The Reference (W) command parameters are defined as follows:

Command Parameter	Description
Mode	**Rotational/Linear Mode Select:** All bits of the Mode must be zero except for the Rotational bit. The Rotational bit can only be set on axes with Quadrature, SSI, or analog velocity feedback. When this bit is set, the positions are wrapped

	between the Retract Limit and the Extend Limit.
Acceleration	**Acceleration Limit:** This parameter limits the rate that the Target Velocity can change. It is specified in position units per second per millisecond. Notice that this limit is ignored if the Filter Time Constant (described below) is zero.
Deceleration	**Reference Deadband:** This parameter configures a deadband that is used to eliminate jitter in the Target Position when the input is at rest. The Target Position does not change until the Actual Position moves at least the number of position units specified by this parameter. This parameter is specified in position units. A value of 0 disables the deadband feature.
Speed	**Velocity Limit:** This parameter limits the rate that the Target Position can change. It is specified in position units per second. Notice that this limit is ignored if the Filter Time Constant (described below) is zero.
Command Value	**Filter Time Constant:** This parameter defines the time constant for the filter to apply to the input. This parameter is specified in milliseconds. The cut-off frequency for a filter in Hertz, given the time constant (t) in seconds is computed by 1/2pt. If this parameter is zero, then no filter is applied, and the Velocity Limit, Acceleration Limit, and Reference Deadband are not applied.

For a complete discussion on the reference mode and configuring the position filter including examples and diagrams, see Reference Axis Filtering.

A.1.45 Spline Relative Sine Move

Character: w
Decimal: 119
Hexadecimal: 0x77
Command Value: Offset from the Spline that the Slave Must Move to

Note: This command is only available in the special ’r;SI’ RMC100 CPU firmware (e.g. 20020624SI). ’r;SI’ firmware is available only on request, as it useful only for special applications. It may not be compatible with standard firmware.

This command moves an axis from its current position to a position relative to a spline. The spline must previously have been loaded to the RMC.

This command carries out two actions:

- It activates the spline currently in the RMC.
- It then superimposes a sine move from the current position to an offset from the spline. Note that the offset can be zero, moving the axis onto the spline.

The Spline Relative Sine Move (w) command parameters are defined as follows:

Command Parameter	**Description**
Mode	The superimposed and gear bits must be set.
Acceleration	Time/Distance Select: If the Speed parameter is zero, this parameter specifies the time for this move to complete.
Deceleration	Spline Position: Specifies the position of the leftmost point of the spline. This effectively shifts the spline right or left as viewed on the curve tool, providing a starting point other than the default of zero.
Speed	Master Move Distance: The sine move occurs as the master moves this distance. If this parameter is zero, the sine move will be based on time. Note that this value can be negative.
Command Value	Offset: This is the distance the slave must move from the spline.

Why Bother?

This command can be used to transition from one spline to the next if the start and end points are not the same. This command can also be used when changing splines on the fly. If this command is issued while already following a spline, it will move to the specified offset to the new spline.

To carry out his command, the RMC performs the following steps:

1. First the RMC calculates where to index into the spline table. This is done by using the master axes current position and subtracting the DECEL field to get the X index in the curve tool. For example, if the master is at 500 position units and the minimum position (DECEL field) is 400 position units, the X index would be 100 position units. NOTE: Never let the master position go below the minimum or maximum positions!
2. The RMC uses the X index (or master position) to interpolate a position from the spline to find the Y position (spline position).
3. The Command Value (spline offset) is added to the Y position to get the destination to where the slave must move.
4. The relative distance between the slave’s destination and current position is calculated.
5. The speed field determines how the sine move is executed as a function of the change in the master's position.
6. The slave now initializes two target generators. One calculates the current spline position as a function of the master's X index. The other starts the sine move that will move the relative distance calculated in step 4. The resulting speeds and differences in position are summed to make a smooth move that converges at the offset from the spline.
7. Once the axis position has reached the offset spline position, the axis will continue to follow the spline at the offset given the Command Value.

A.1.46 New Spline Point Command

Character: X or x
Decimal: 88 or 120
Hexadecimal: 0x58 or 0x78
Command Value: Requested Spline Position, Position units

This command adds a point to the current spline segment. This segment cannot be followed (using the Follow Spline Segment command) until the End Spline Segment command has been issued to calculate the final curve.

This command will fail for the following reasons, as indicated by the Parameter Error bit in the Status word:

- The maximum number of points for the axis has been reached. This limit is 1024 points per axis on a 2-axis RMC, 512 points per axis on a 3- to 4-axis RMC, and 256 points per axis on RMCs with five or more axes.
- The final calculations for the last segment have not been completed. This can be avoided by waiting for the Acknowledge bit in the Status word to toggle after sending the End Spline Segment command to end the previous segment. If this occurs, you will have to re-send the End Spline Segment command.
- Adding the point generates an overflow in calculating the spline curve. If this happens on this command, then the point will not be added, but the rest of the spline points will be retained. This will occur if there is too great a change in position over too short of an interval. You will need to either increase the interval between the points or move the point closer to the previous point. The general limitation is that the speed must be between 0 and 65535 position units per second, and acceleration must be between -512 and 511 position units per second per second. If position units are thousandths of inches then this would translate to 511 inches per second per second, or about 1.25 G’s.

A.1.47 Start a Graph Command

Character: y
Decimal: 121
Hexadecimal: 0x79
Command Value: Unused

This command begins a new plot immediately. The only case where a new plot will not be started is when RMCWin is currently reading up the currently captured plot.

A.1.48 Zero Position/Set Target Command

Character: Z
Decimal: 90
Hexadecimal: 0x5A
Command Value: New Target Position

This command is used to set the Target Position to any value. This command also changes the Command Position and Actual Position by the difference between the new target position and the old target position.

When this command is issued to absolute positioned axes such as MDT, SSI (linear), and analog, the Offset parameter is adjusted in order to make the position match the requested position. On resolver axes, the Count Offset parameter is adjusted to make the position match the requested position. If you wish to store the Count Offset or Offset to non-volatile memory, you must issue the Update Flash Command.

If the axis is in motion when the ’r;Z’ command is issued, it will continue moving to the same physical location it was originally going to, but the indicated position will be different.

This command requires firmware 20000918 or newer, but for best results, use only in versions 20011015 or newer.

A.1.49 Offset Positions Command

Character: z
Decimal: 122
Hexadecimal: 0x7A
Command Value: Target Position Offset
Command Value (Resolver): Offset in Turns

For non-resolver axes, this command is used to offset the Command, Target and Actual Positions by the amount specified in the Command Value (-32767 to +32768). Normally this command is used when indexing to reset the target position back to 0 before repeating a sequence of steps in the step table.

When this command is issued to absolute positioned axes such as MDT, SSI (linear), and analog, the Offset parameter is adjusted in order to make the position match the requested position.

Resolver Axes

For Resolver axes, this command is used to offset the Command, Target, and Actual Positions by the number of turns specified in the Command Value. This command will not change the Count Offset.

This command is intended for use on axes that rotate continuously in one direction as the machine goes through its cycle. After each machine cycle, issuing this command would set the positions back by the number of revolutions the resolver made in each cycle.

If the axis is in Rotational Mode when this command is issued, it is possible to set the positions outside the limits established by the Extend and Retract limits. The positions will be moved back within limits the next loop time. Axis control will not be affected.

A.1.50 Reset Outputs Command

Character:]
Decimal: 93
Hexadecimal: 0x5D
Command Value: Mask of Digital Output Bits to Clear

This command uses the Command Value to determine which outputs to reset. It uses the Command Value in the same manner as the Set Outputs command, except that it resets each output that corresponds to a bit set in the Command Value.

See also: Set Outputs

A.1.51 Set Outputs Command

Character: [
Decimal: 91
Hexadecimal: 0x5B
Command Value: Mask of Digital Output Bits to Set

This command uses the Command Value to determine which outputs to set. The value is treated as a bit mask. The bits set in the Command Value will set the corresponding outputs.

With a Communication DI/O, bits 0 (hexadecimal value 0x0001) to 7 (hexadecimal value 0x0080) are used for the corresponding outputs on that module. However, some of these outputs are reserved, so refer to the Communication Digital I/O topic for more details.

Without this module, the CPU outputs 0 and 1 are represented by hexadecimal values 0x0001 and 0x0002 respectively.

When a Sensor Digital I/O is used, bits 8 (hexadecimal value 0x0100) to 15 (hexadecimal value 0x8000) represent outputs 0 to 7 on that module.

Use the following chart to help find the correct hexadecimal Command Value for turning on the desired bits. The Command Value is comprised of four hexadecimal digits. The first two digits are used for setting outputs on the Sensor Digital I/O, and the last two digits are used for setting outputs on the Communication Digital I/O. Each row of the chart begins with a hexadecimal digit. To the right of each digit, the corresponding binary pattern is repeated four times, once for each digit. For this command, binary 1's represent outputs to be turned on, while binary 0's represent unchanged outputs. Notice, that you will be treating each digit of the command value separately, so that you will most likely use more than one hexadecimal digit row. Remember that you must enter a leading 'r;0' to enter a hexadecimal value. Following are some examples:

- To turn on bit 0 of the Communication Digital I/O, the command value should be 0x0001. The first three digits are zero to indicate that all Sensor Digital I/O outputs and outputs 4-7 of the Communication Digital I/O are untouched. The 'r;1' in the fourth digit indicates that output 0 is turned on, but outputs 1-3 are untouched. Recall that the leading 0 in the command value is only used to indicate that the number is hexadecimal.
- To turn on bits 2, 3, 6 and 7 of the Communication Digital I/O, the command value should be 0x00CC.
- To turn on all eight Sensor Digital I/O outputs, the command value should be 0xFF00.
- To turn on bit 2 of the Communication Digital I/O, the command value should be 0x0004.

Command	**First**	**Second**	**Th**

Value	**Digit**				**Digit**				**Di**
Hex F =	1	1	1	1	1	1	1	1	1
Hex E =	1	1	1	0	1	1	1	0	1
Hex D =	1	1	0	1	1	1	0	1	1
C =	1	1	0	0	1	1	0	0	1
B =	1	0	1	1	1	0	1	1	1
A =	1	0	1	0	1	0	1	0	1
9 =	1	0	0	1	1	0	0	1	1
8 =	1	0	0	0	1	0	0	0	1
7 =	0	1	1	1	0	1	1	1	0
6 =	0	1	1	0	0	1	1	0	0
5 =	0	1	0	1	0	1	0	1	0
4 =	0	1	0	0	0	1	0	0	0
3 =	0	0	1	1	0	0	1	1	0
2 =	0	0	1	0	0	0	1	0	0
1 =	0	0	0	1	0	0	0	1	0
0 =	0	0	0	0	0	0	0	0	0
Output	Sensor DI/O								Co
Numbers	7	6	5	4	3	2	1	0	7

Note: The above discussion on determining the correct hexadecimal number to turn on one or more outputs applies also to the Reset Outputs command. The only difference is that the outputs represented by a binary ’r;1’ are turned off instead of on.

See also: Reset Outputs

A.1.52 Simulate Rising Edge Command

Character: {
Decimal: 123
Hexadecimal: 0x7B
Command Value: Input to Simulate in Input-to-Event Table

Note: This command is supported in RMC100 CPU firmware dated 19980331 or later.

This command is used to start event sequences on one or more axes simultaneously. This is done by simulating a rising edge on an input row of the Input to Event table, even if there is no physical discrete input on the controller. Each row of the Input to Event table holds the event number to start on each axis. For further details on the Input to Event table, see Editing the Input to Event Table.

Note: In no way does this command affect the physical discrete input; that is, it does not force the input high.

A.1.53 Simulate Falling Edge Command

Character: }
Decimal: 125
Hexadecimal: 0x7D
Command Value: Input to Simulate in Input-to-Event Table

Note: This command is supported in RMC100 CPU firmware dated 19980331 or later.

This command is used to start event sequences on one or more axes simultaneously. This is done by simulating a falling edge on an input row of the Input to Event table, even if there is no physical discrete input on the controller. Each row of the Input to Event table holds the event number to start on each axis. For further details on the Input to Event table, see Editing the Input to Event Table.

Note: In no way does this command affect the physical discrete input; that is, it does not force the input low.

A.1.54 Sine Move Command

Character: ~
Decimal: 126
Hexadecimal: 0x7E
Command Value: Change in Position, in position units.

Note: This command is supported in RMC100 CPU firmware dated 19990625 or later (and beta firmware dated 19980827B or later).

This command generates a simple move that follows a sine curve for a motion profile. It is intended to be the simple type of move, whether geared or not. In the non-geared case, give the distance to move, the time you would like it to take, and as a safeguard, the maximum speed you allow. In the geared case, give the distance you want the slave axis to move for the distance you want the master to move.

The primary disadvantage of this command is that the move profile has no constant velocity section, and therefore may not be suited for longer moves.

If the axis is not geared, the command fields are used as follows:

- In the Mode word…
  - …the Gear Mode bit must be cleared.

  - o …the Graph Disable, Rotational, and Monitor Pressure bits may be set if desired.

- o The Acceleration field is reserved. It should be set to zero.
- o The Deceleration field is used as the requested time in milliseconds that the move may take to complete. This time may not actually be used if the maximum speed (described below) would be exceeded. In this case a longer time will be used. If you do not care about the time taken by the move, but instead want the move to use the maximum speed, enter zero in this field. This field is unsigned. The time is an unsigned number between 0 and 65,535 milliseconds.
- o The Speed field is used as the maximum speed in position units per second allowed by this move. If the speed required to complete the move in the time given in the Deceleration field is above this maximum speed, the time will be extended to use the maximum speed. The speed is an unsigned number between 0 and 65,535 position units per second.
- o The Command Value field is used as the requested change in position, given in position units. It is a signed number between -32,768 and 32,767.

Gearing a sine move is most useful when geared to an axis in speed control, or an independent quadrature input. If the axis is geared, the command fields are used as follows:

- In the Mode word…
  - o …the Gear Mode bit must be set.
  - o …the Gear Master Select bits are used to select the axis that will be the master for this move.
  - o …the Gear Type bit is used to determine if the axis is slaved off its master’s target or actual position.
  - o …the Graph Disable, Rotational, and Monitor Pressure bits may be set if desired.
- o The Acceleration and Deceleration fields are reserved. They should be set to zero.
- o The Speed field is used as the distance that the gear master axis will travel during the slave’s sine move. This is given in position units.
- o The Command Value field is used to determine the distance the slave will travel during the sine move.

Addendum for Special SI Firmware (e.g. 19980827SI)

The above description of this command is correct except for the following additions:

- The Superimposed mode bit is used. When set, this move can be made while the axis is already performing another Geared, Speed Control, or Spline move. When cleared, this move will be the only move performed by the axis. This command is especially useful when superimposed, for example, to catch up one speed control axis with another.
- A second nearly identical command is added: Master Relative Sine Move (|, decimal 124, hexadecimal 0x7C). The existing ’r;~’ command is renamed more precisely to be the ”r;Slave Relative Sine Move.” This command is identical in every way except the following items:
  - o Both the Superimposed and Geared Mode bits must be set, otherwise a parameter error will occur.

- o The distance the user wishes the slave to travel (given by the Command Value) is added to the distance that the slave’s target position lags behind the master’s target position. For example, if the master is at 1000 position units, the slave is at 850 position units, and a Master Relative Sine Move is given with a Speed (master distance) of 500, and a Command Value (slave distance) of 500, then the master will move 500, and the slave will move 650 (500 requested distance plus 1000 - 850 = 150 position units to catch up) so that both arrive at 1500 position units.

Example of the Master Relative Sine Move:

Axis 0 and 1 are controlling two rollers via quadrature encoder feedback from motors. Both axes use the same position units. The drums are rotating at the same speed, but because the drums started at different times, their positions are not guaranteed to be exactly lined up. The user wishes to catch up one drum to the other.

To catch up the two drums, the user would use a sine move superimposed on the speed control of the drum that the user wishes to catch up to the other drum. The following command would be issued to axis 1 (the slave):

Mode	0x2600 (Geared to axis 0, speed control, superimposed)
Accel	0 (Unused)
Decel	0 (Unused)
Speed	1000 (Master distance)
Cmd Value	0 (Slave distance; difference in positions will be added to this value)
Command	| (0x7C)

This single command will calculate the distance between the two axes, and it will generate a move which will move this distance (plus the command value, which is 0) in the time that the master axis moves 1000 position units. This command works whether the master is ahead of the slave (the slave will speed up temporarily) or the slave is ahead of the slave (the slave will slow down temporarily).

A.1.55 Set and Reset Wait Bits Command

Character: none
Decimal: 23
Hexadecimal: 0x17
Command Value: Bits to Reset : Bits to Set

Note: This command is supported in RMC100 CPU firmware dated 20060216 or newer.

The RMC100 has 8 internal bits that can be used as semaphores. These bits can be set and reset using the Set and Reset Wait Bits command. These same internal bits can be tested with the Check Wait Bits link type. This allows an axis' task to signal other tasks or allows a task to wait until some event has happened.

The 16-bit Command Value is split into two bytes. The upper 8 bits specify which bits are to be cleared. For example, setting bits 8 and 10 of the Command Value would clear wait bits 0 and 2. Likewise the lower 8 bits specify the bits to be set. It is possible to set and clear multiple bits with just one command.

Example:

Axis 4 may need to know when axes 0 to 3 have cleared a position. The step sequence running on Axis 4 would set wait bits 0 to 3 and then wait for axis 0 to 3 to clear their bit. When all four bits are cleared then axis 4 would know it is safe to proceed. Axis 4 would use the check wait bits link type to insure that wait bits 0 to 3 are cleared before proceeding. It is much easier for axis 4 to do just one check for the 4 wait bits to be cleared than doing four individual enhanced links that check axis 0 to 3's position one by one.

Note: Wait bit 7 is used by the Profibus DP communications so the user can tell when spline downloads are finished. The Profibus DP spline processing routine sets wait bit 7 when a download to the spline area is detected. The bit is cleared when the download is done. The user can then use the step table and the Check Wait Bits link to be sure the spline is ready before executing the spline.

A.1.56 Sine Move Continuous Command

Character: none
Decimal: 48
Hexadecimal: 0x30
Command Value: Amplitude

This command moves the position, pressure or force axis in a sinusoid. The user specifies the amplitude, frequency, and number of cycles (count). The count may be changed while the move is in progress, but the amplitude and frequency should not be changed until the Sine Move Continuous is complete. The Sine Move Continuous will stop at the starting position after completing the specified number of cycles.

The cycling can be stopped by issuing a Sine Move Continuous command with the count set at 1. When the actuator gets to the starting position the cycling will stop.

Note: On pressure control axes, the axis must be in closed-loop pressure control when this command is issued. If the pressure control axis is in open loop control when this command is issued, a parameter error will result.

Command Value: Amplitude in position units. The current target position is used as a starting point and the sinusoidal motion can go positive or negative from there depending on the sign of the amplitude. For example: If the actuator is a 1000 and a sinusoidal command has an amplitude of 100 then the actuator will cycle between 1000 and 1200. If the amplitude is –50, the actuator will cycle between 1000 and 900.

Speed: Frequency in HZ x 100. This allows frequencies up to 655Hz but in reality it is hard to attain. For example, 500 is 5 Hz, and 20 is 0.2 Hz.

Deceleration: Count. Enter 1-65535 for a set number of cycles. Enter 0 for continuous cycling.

Acceleration: Not Used.

Stopping a Sine Move Continuous in Progress:

The cycling can be stopped by issuing a Sine Move Continuous command with the count set to 1. When the actuator gets to the starting position the cycling will stop.

Note: Do not change the frequency or amplitude on-the-fly unless the actuator is back at the start position.

Gearing:
The position or pressure/force PID can be geared to an external reference that represents a rate in engineering units per second. If the external reference has a rate of 1000 position (or pressure) units per second, the gear ratio is one to one. If the external reference has a rate of 2000 engineering units per second then the gear ratio is two to one.

A.1.57 Follow Spline Relative Command

Character: none
Decimal: 6
Hexadecimal: 0x06
Command Value: Reserved (must be 0)

This command will begin following the next available spline relative to the current axis position.

This command is identical to the Follow Spline Segment(f) command, with the following exceptions:

- The axis is not required to be at the position of the first spline point.
- The Accel, Decel, Speed, and Command Value parameters are reserved and must be set to 0.
- The following Mode word bits must be set to 0:
    - Acceleration and Deceleration Mode Select Bits (bits 0-1)
    - S-Curve Bit (bit 7)

This command allows a spline to be started at any position, effectively shifting the entire spline along the axis. This may cause parts of the spline to exceed the extend or retract limits, or the maximum or minimum axis positions. When the position exceeds the limits, the following occurs:

- Exceeding Extend or Retract limits:
  This causes a Target Position Moved Outside Limits parameter error and a Hard Stop.
- Exceeding maximum or minimum axis positions (see one of the Scaling topics):
  This causes a Target Position Moved Outside Limits parameter error. If the Parameter Error Auto Stop bit is set to Status Only, the axis will continue to follow the spline. However, the spline

position will be truncated to the maximum or minimum position until the spline re-enters the limits.

A.1.58 Map Output to Axis Position

Character: None
Decimal: 127
Hexadecimal: 0x7F
Command Value: Axis and Position Select (see below)

This command maps the drive output for this axis to an axis's position. Note that this command is not valid on pressure axes. This command may be used for applications where a chart recording analog inputs must be used to capture a position graph. The Command Value is used to select the axis, the position type (Command, Target, or Actual), and the direction of the drive output.

There are two separate mappings for the direction of the drive output:

In the first (positive) mapping, a position at or below the retract limit gives an output of 0V, a position at or above the extend limit gives an output of 10V, and positions between the two are linearly interpolated using the following formula:

$$\text{Output} = \frac{(\text{Position} - \text{Retract Limit})}{(\text{Extend Limit} - \text{Retract Limit})} \times 10\text{V}$$

In the second (negative) mapping, a position at or below the retract limit gives an output of 10V, a position at or above the extend limit gives an output of 0V, and positions between the two are linearly interpolated using the following formula:

$$\text{Output} = \frac{(\text{Position} - \text{Extend Limit})}{(\text{Extend Limit} - \text{Retract Limit})} \times 10\text{V}$$

Note: The drive outputs are not precision outputs. The output may not be exactly the value you specify. Look up specifications in the online help index for the tolerance on your particular interface module's drive outputs.

See the following chart for the exact Command Values:

Command Value	Axis	Position Type	Mapping
0	0	Command	Positive

1	0	Target	Positive
2	0	Actual	Positive
10	1	Command	Positive
11	1	Target	Positive
12	1	Actual	Positive
n*10+0	n	Command	Positive
n*10+1	n	Target	Positive
n*10+2	n	Actual	Positive
-1	0	Command	Negative
-2	0	Target	Negative
-3	0	Actual	Negative
-11	1	Command	Negative
-12	1	Target	Negative
-13	1	Actual	Negative
-(n*10+1)	n	Command	Negative
-(n*10+2)	n	Target	Negative
-(n*10+3)	n	Actual	Negative

A.1.59 Move Relative to An Axis

Character: None
Decimal: 192-207
Hexadecimal: 0xC0-0xCF
Command Value: Change in Position, in position units.

Note: This command is supported in RMC100 CPU firmware dated 19980425 or later.

This set of commands is used to move one axis based on the position of another axis. For example, suppose two axes are working together in a press application. Both press the product in pressure-control mode. Then, to release the gases they ramp back the pressure. Next, it is desired that both axes back off ¼ inch. To ensure that no skew is allowed, each axis could be given the same Move Relative to Another Axis command to back up the same distance or to the same position.

The Acceleration, Deceleration, Mode bits and Speed are all used as in a normal Go command.

This group of sixteen commands can use either the Command or Target Position of any of up to eight axes as the base for the move. The table below shows which command to use to select the desired base:

Command		
Hex	**Decimal**	**Move Relative to…**
0xC0	192	…Axis 0 Command Position
0xC1	193	…Axis 1 Command Position
0xC2	194	…Axis 2 Command Position
0xC3	195	…Axis 3 Command Position
0xC4	196	…Axis 4 Command Position
0xC5	197	…Axis 5 Command Position
0xC6	198	…Axis 6 Command Position
0xC7	199	…Axis 7 Command Position
0xC8	200	…Axis 0 Target Position
0xC9	201	…Axis 1 Target Position
0xCA	202	…Axis 2 Target Position
0xCB	203	…Axis 3 Target Position
0xCC	204	…Axis 4 Target Position
0xCD	205	…Axis 5 Target Position
0xCE	206	…Axis 6 Target Position
0xCF	207	…Axis 7 Target Position

Why Bother?

These commands are handy when exiting open loop or pressure mode and re-synchronizing axes. Remember the Target and Actual Positions are the same in these modes. That is why there is not a relative move from an actual position.

A.1.60 Set Parameter On-the-Fly

Character: None
Decimal: 208-223
Hexadecimal: 0xD0-0xDF
Command Value: Value of the Parameter

Note: This command is supported in RMC100 CPU firmware dated 19980414 or later.

Each of these commands sets one of the parameters. The following table can be used to determine which command sets which parameter. Reference axes parameters cannot be set on-the-fly. Notice that some parameters can never be set on the fly:

Command		**MDT, SSI, Quad, Analog (Position)**	**Stepper**	**Pressure/force**
Hex	**Decimal**	**Parameter**	**Parameter**	**Parameter**
0xD0	208	Config. Word	Invalid command	Invalid command
0xD1	209	Scale (2) (4)	Invalid command	Invalid command
0xD2	210	Offset (1)	Coordinate Limit (2)	Invalid command
0xD3	211	Extend Limit (1)	Extend Limit (2)	Invalid command
0xD4	212	Retract Limit (1)	Retract Limit (2)	Invalid command
0xD5	213	Proportional Gain	Compensation Rate (2)	Proportional Gain
0xD6	214	Integral Gain	Compensation Timeout (2)	Integral Gain
0xD7	215	Differential Gain	Invalid command	Differential Gain
0xD8	216	Extend Feed Fwd.	Steps/Rev (2)	Extend Feed Fwd
0xD9	217	Retract Feed Fwd.	Pos. Units/Rev (2)	Retract Feed Fwd
0xDA	218	Extend Acc. Feed Fwd.	Quad Counts/Rev (2)	Integrator Preload (
0xDB	219	Retract Acc. Feed Fwd.	Max. Steps/ms (2)	Filter Time Constan (3)

0xDC	220	Dead Band Eliminator	Compensation Window (2)	Invalid command
0xDD	221	In Position Window	In Position Window (2)	Invalid command
0xDE	222	Following Error Window	Following Error Window (2)	Invalid command
0xDF	223	Auto Stop Error Mask	Auto Stop Error Mask (2)	Invalid command

1. Changing these parameters on-the-fly requires firmware 20000913 or later.
2. Changing these parameters on-the-fly requires firmware 20001204 or later.
3. Changing these parameters on-the-fly requires firmware 20020426 or later.
4. Setting the scale on-the-fly requires that the axis be in open loop.

You will notice that some parameters cannot be changed on the fly. They can be changed by stopping the axis, writing the parameters, and issuing a Set Parameters command.

Warning: Changing any of the gains or feed forwards generates a drive discontinuity in most cases.

A.2 Pressure/Force Control ASCII Commands

A.2.1 Set Bias Drive Command

Character: B
Decimal: 66
Hexadecimal: 0x42
Command Value: Requested Bias Drive in Millivolts

At times it may be necessary to apply an extra amount of drive if you know the system tends to lag at a particular time in a pressure ramp. This command can be used to add an additional amount of drive while regulating pressure.

Note: This command should only be used as a last resort. In most cases the Feed Forward or Drive Transfer Percent fields will give the desired results with greater consistency.

This command uses the Mode, Pressure Set A, Pressure Set B and Command Value fields different from their usual uses:

Pressure Set A and Pressure Set B:
These fields are used to give the rate that the Bias Drive is increased and decreased. Pressure Set A is used when the Bias Drive is being moved away from 0 volts. Pressure Set B is used to control the rate that the Bias Drive moves back toward 0 volts. See the description of the Mode field below for details on the units used by these fields.

Mode:
The Mode field is used to specify the units of the Pressure Set A and Pressure Set B fields during

the Bias Drive command. The possible values are:

Mode 1
The Pressure Set A and Pressure Set B fields use units of millivolts per millisecond. For example, if the current Bias Drive is 0 volts, the Command Value is 1000 millivolts, and the Pressure Set A field is set to 50, it will take 20 milliseconds for the Bias Drive to reach 1000 millivolts.

Mode 3
The Pressure Set A and Pressure Set B fields are given in milliseconds. For example, if the current drive output is 0 millivolts, the Command Value is 1000 millivolts, and the Pressure Set A field is 50, then it will take 50 milliseconds for the Bias Drive to reach 1000 millivolts.

Command Value (Requested Drive):

This value gives the number of millivolts to ramp the Bias Drive up or down to. The ramping of the Bias Drive will begin as soon as the axis is in pressure monitoring mode, however this Bias Drive will not actually be added to the output drive until the axis begins regulating pressure.

Note: Using this command requires overwriting the Mode, Pressure Set A and Pressure Set B fields. If this is done using a full command profile from the Event Step table, then the current pressure fields will not be messed up. However, these fields must be restored to their intended values when the next Set Pressure command is issued.

A.2.2 Start Events Command

Character: E
Decimal: 69
Hexadecimal: 0x45
Command Value: Event Step to Start Execution

This command starts an event step sequence at the step specified in the Command Value field. If the axis already was in the middle of an event sequence, that sequence will end and the new event sequence will begin.

Note: RMC100 CPU Firmware dating prior to 19980414 supports event sequences only on position axes. Issuing this command to a pressure or force axis will result in a parameter error.

A.2.3 Set Mode Command

Character: M
Decimal: 77
Hexadecimal: 0x4D
Command Value: New Mode value

This command is used to change the Mode for the current command in one of several ways. The changes take place immediately. The allowed bits (described below) are copied from the Command Value to the Mode field for this axis. The following Mode bits can be changed with this command:

- **Gearing Type Bit (bit 14) and Gear Master Select Bits (bits 4-6):**
  If the axis is already gearing to another axis, then these bits can be changed to switch immediately to gearing to another axis or position quantity. Changing these bits when the axis is not already geared will have no effect. The gearing bits cannot be changed if the axis is gearing

in mode 2.

- **Monitor Pressure Bit (bit 8):**
  Clearing this bit while regulating pressure will drop the axis out of pressure regulation. Changing this bit at any other time will simply make the axis stop or resume monitoring the pressure for entering pressure control.

- **Rotational Bit (bit 9):**
  Changing this bit puts the axis into or pulls it out of Rotational mode, as described in Rotational Mode. Changing this bit while the axis is doing a Point-to-Point, Synchronized, Quick, or Speed Control move will have no effect, since Point-to-Point, Quick, and Synchronized moves are only defined in non-Rotational mode and Speed Control is only defined in Rotational mode. It is allowed in Open Loop, Stopped, Spline, Gear, and Reference control modes.

- **Pressure Mode Bits (bits 0-1)—PRESSURE AXES ONLY:** If these bits are changed while regulating pressure, the axis will come out of pressure regulating mode, and return to pressure monitoring mode. Depending on the conditions, the axis may immediately re-enter pressure regulating mode, but there will be a discontinuity. For this reason, it is not recommended that the mode be changed while regulating pressure.

A.2.4 Open Loop Command

Character: O
Decimal: 79
Hexadecimal: 0x4F
Command Value: Millivolts of Drive

CAUTION: Use this command with care! Open Loop operation disables all safety features on the RMC!

Note: The open loop command will not affect drive output while the Simulate bit is set in the Config word. Drive output is always 0 volts in simulate mode.

Note: For pressure/force axes, this command will take effect on the analog module’s drive output, if one is available. Only 16-bit analog cards have drive outputs, therefore this command will have no affect on 12-bit analog cards. Also, because drive outputs are only assigned to input channels 0 and 2 on 16-bit cards, only analog or pressure axes using those input channels will be able to issue this command. Otherwise, this command behaves normally on pressure/force axes, except that no status registers other than the Drive field are modified by this command.

The Open Loop command allows the Controller to directly specify values for the analog output. The output range is -10000 to 10000 and is given in millivolts. However, the drive outputs are not precision outputs. You cannot rely on the output being exactly the value you specify. Look up specifications in the online help index for the tolerance on your particular interface module’s drive outputs.

The O command uses the following parameters from the last open loop profile specified:

The Command Value field specifies the desired drive in millivolts to output. The current null drive is added to this value.

The Speed field is not used.

The Acceleration and Deceleration fields control the rate at which the drive output ramps to the requested value. Acceleration is used when the drive output is moving away from 0 and deceleration is used when drive output is moving toward 0. The actual meaning of the values depends on the Acceleration/Deceleration mode bits in the Mode word. Notice that only modes 1, 2 and 3 are used:

Mode 1
Acceleration and deceleration are given in millivolts per millisecond. For example, if the current drive output is 0 millivolts, and the Command value is 1000 millivolts, and the Acceleration is 50, then it will take 20 milliseconds for the drive output to reach 1000 millivolts.

Mode 2
Acceleration and deceleration are given as distances. That is, the drive will be ramped over the distance specified. This mode is most useful when stopping on poor performing hydraulics. This mode cannot be used for starting from a stop because no drive will be applied since the position does not move, and conversely, the position does not move because there is no drive. For example, if the current drive output is 1000 millivolts, and the Command value is 0 millivolts, and the Deceleration is 4000, then the axis will ramp the drive down to 0 in the time it takes the axis to move 4000 position units.

Mode 3
Acceleration and deceleration are given in milliseconds. For example, if the current drive output is 0 millivolts, the Command value is 1000 millivolts, and the Acceleration is 50, then it will take 50 milliseconds for the drive output to reach 1000 millivolts, as the drive will increase 20 millivolts each millisecond.

Note: If a move crosses 0 drive, then Deceleration will be used until the drive reaches 0, and then Acceleration will be used until the final drive is reached.

Tip: If you are going to be changing from closed loop to open loop, you must be aware of the following. While in closed loop and holding a position, the drive may be switching back and forth between small positive and negative drives. If an open loop command is given to go to a non-zero drive, the starting drive may have a sign opposite of the requested drive. In this case, the drive will decelerate down to zero millivolts and then accelerate up to the new drive. For example, if the current drive is -10 millivolts and an open loop command requests a drive of 2000 millivolts (2V), the drive will ramp from -10 to 0 and then accelerate from 0 to 2000mV. To ensure that this deceleration happens quickly, the deceleration must be set properly (by setting it to 0 in Mode 3, and to a high number (1000 or greater) in Mode 1).

Stepper Axes (QST Modules)

Stepper axes do not have an analog output so the Open Loop command generates pulses on the Step+ and Step- Outputs. It also sets the polarity on the Direction+ and Direction- signals.

The pulse frequency is specified in the Command Value register as the number of pulses per millisecond.

The Acceleration and Deceleration parameters specify the rate at which the Open Loop output value is changed. See the discussion above for a description of the modes available.

A.2.5 Set Parameters Command

Character: P or p
Decimal: 80 or 112
Hexadecimal: 0x50 or 0x70

Command Value: Unused

When a 'P' command is given all initialization parameters are updated.

Tip: If you wish to change parameters without stopping the axis, see the Set Parameter On-the-Fly commands.

Non-pressure/force Axes

The minimum requirement of this command is to set the Extend and Retract Limits to their proper values (see Start-Up and Tuning). When a 'P' command is given, the RMC will copy the Actual Position of the axis into the Target and Command Positions. Therefore the axis will be in closed loop control around its current position.

Pressure/Force Axes

When this command is received, the RMC will copy the Actual Pressure of the axis into the Target Pressure status field.

A.2.6 Quit Events Command

Character: Q
Decimal: 81
Hexadecimal: 0x51
Command Value: Unused

This command stops the event control sequence on the axis the command is issued to. Any motion commands in progress on the axis are not stopped. Therefore, if an axis is currently moving to 4.000”, it will continue moving to that position; if an axis is in open loop with 1.000V drive, the drive will continue holding that drive, etc. If you desire to stop the axis at the same time, look at the Halt and Disable Drive Output commands.

Note: RMC100 CPU firmware dating prior to 19980414 supports event sequences only on position axes, issuing this command to a pressure or force axis will result in a parameter error.

A.2.7 Set Pressure Ramp Time Command

**Character: **
Decimal: 92
Hexadecimal: 0x5C
Command Value: Milliseconds

This command sets the Ramp Time value to the Command Value. Refer to Ramp Time for details on this field.

The change in the Ramp Time value will take affect immediately if the axis is regulating pressure at the time. The remaining ramp time is taken as a fraction of the entire Ramp Time and multiplied by the new Ramp Time.

Example:

Suppose that an axis was to ramp the pressure from 2000psi to 2200psi over two seconds. Therefore, the Ramp Time would have been 2000 milliseconds. Suppose that one and a half seconds into the ramp, this command is issued with a Command Value of 4000. Therefore, because one fourth of the original Ramp Time was remaining, one fourth of the new Ramp Time (one fourth of 4000 milliseconds or 1000 milliseconds) will remain. So, the rest of the ramp will take a full second.

A.2.8 Set Pressure Command

Character: ^
Decimal: 94
Hexadecimal: 0x5E
Command Value: Pressure Value

This command is used as the main command for regulating pressure. When this command is issued, the current Mode, Pressure Set A, Pressure Set B, and Ramp Time are processed. Additionally, the Command Value will be copied into the Command Pressure. There are two ways that this command is used:

1. Use this command to set up the entry into pressure regulating mode. This command is used to set up the target pressure curve before a command is issued to the position axis, which would put the axis in pressure-monitoring mode. The axis will then only monitor the pressure and will continue moving based on position until the pressure reaches Pressure Set A. At this point, the pressure will be ramped to the Command Pressure that was specified in the Command Value field of this command.

Note: If this command was not issued before a command put the axis in pressure monitoring mode, the Pressure Set A point may not be defined and the axis may begin ramping to an undesired pressure immediately.

2. Use this command while in pressure regulating mode to ramp between two pressures. When the axis is in pressure regulating mode, this command takes effect immediately and the axis begins to ramp. To avoid a discontinuity in the pressure curve, this command should not be issued until the previous pressure ramp is complete.

If this command is issued on a position-pressure axis, the behavior when the Actual Position reaches the Command Position or the Extend Limit is as follows:

- The Target Pressure will begin decreasing. If the Actual Pressure goes below Pressure Set B, the axis will exit pressure control.

A.2.9 Set Pressure Set A Command

Character: |
Decimal: 124
Hexadecimal: 0x7C
Command Value: Pressure Value

This command sets the Pressure Set A value to the Command Value. The new value for Pressure Set A is used as soon as the command is processed. Refer to Pressure Set A for details.

A.2.10 Set Pressure Set B Command

Character: _
Decimal: 95
Hexadecimal: 0x5F
Command Value: Pressure Value

This command sets the Pressure Set B value to the Command Value. The new value for Pressure Set B is used as soon as the command is processed. Refer to Pressure Set B for details.

A.2.11 Set Parameter On-the-Fly

Character: None
Decimal: 208-223
Hexadecimal: 0xD0-0xDF
Command Value: Value of the Parameter

Note: This command is supported in RMC100 CPU firmware dated 19980414 or later.

Each of these commands sets one of the parameters. The following table can be used to determine which command sets which parameter. Reference axes parameters cannot be set on-the-fly. Notice that some parameters can never be set on the fly:

Command		**MDT, SSI, Quad, Analog (Position)**	**Stepper**	**Pressure/force**
Hex	**Decimal**	**Parameter**	**Parameter**	**Parameter**
0xD0	208	Config. Word	Invalid command	Invalid command
0xD1	209	Scale (2) (4)	Invalid command	Invalid command
0xD2	210	Offset (1)	Coordinate Limit (2)	Invalid command
0xD3	211	Extend Limit (1)	Extend Limit (2)	Invalid command
0xD4	212	Retract Limit (1)	Retract Limit (2)	Invalid command
0xD5	213	Proportional Gain	Compensation Rate (2)	Proportional Gain
0xD6	214	Integral Gain	Compensation Timeout (2)	Integral Gain
0xD7	215	Differential Gain	Invalid command	Differential Gain

0xD8	216	Extend Feed Fwd.	Steps/Rev (2)	Extend Feed Fwd
0xD9	217	Retract Feed Fwd.	Pos. Units/Rev (2)	Retract Feed Fwd
0xDA	218	Extend Acc. Feed Fwd.	Quad Counts/Rev (2)	Integrator Preload (
0xDB	219	Retract Acc. Feed Fwd.	Max. Steps/ms (2)	Filter Time Constan (3)
0xDC	220	Dead Band Eliminator	Compensation Window (2)	Invalid command
0xDD	221	In Position Window	In Position Window (2)	Invalid command
0xDE	222	Following Error Window	Following Error Window (2)	Invalid command
0xDF	223	Auto Stop Error Mask	Auto Stop Error Mask (2)	Invalid command

1. Changing these parameters on-the-fly requires firmware 20000913 or later.
2. Changing these parameters on-the-fly requires firmware 20001204 or later.
3. Changing these parameters on-the-fly requires firmware 20020426 or later.
4. Setting the scale on-the-fly requires that the axis be in open loop.

You will notice that some parameters cannot be changed on the fly. They can be changed by stopping the axis, writing the parameters, and issuing a Set Parameters command.

Warning: Changing any of the gains or feed forwards generates a drive discontinuity in most cases.

A.3 Programmable Controller Commands

A.3.1 Command Words for Digital I/O's Command Mode

There are two command words. First, the command itself is placed on digital inputs 0-15 when the Command Strobe (CPU input 1) is low. Its value is read by the RMC when the Command Strobe transitions high. The second word—the command value or data—is placed on digital inputs 0-15 when the Command Strobe is high and is read by the RMC when the Command Strobe transitions low. Refer to Command Mode for further details on sending the command and command value.

The command given in the command word is not executed until the command value is strobed in (that is, the Command Strobe transitions back low). Commands that request data are an exception; the low 8 bits of the requested data are available as soon as the RMC toggles the Acknowledge bit (CPU output 1). This is done to allow a full 16-bit piece of data to be received in one command cycle.

The following table lists the available commands that can be placed on inputs 0-15:

```
    1111|11  |    |
Bit#5432|1098|7654|3210
    -------------------
    0AAA|RRRR|0000|0000 No command
    0AAA|RRRR|0000|CCCC Open Loop Using Profile
    0AAA|RRRR|0001|CCCC Set Parameter
    0AAA|RRRR|0010|CCCC Set Profile
    0AAA|RRRR|0011|XXXX Reserved
    0AAA|RRRR|01CC|CCCC ASCII Commands
    0AAA|RRRR|1000|CCCC Go/Set Pressure Using Profile
    0AAA|0000|1001|CCCC Get Parameter
    0AAA|0000|1010|CCCC Get Profile
    0AAA|RRRR|1011|XXXX Reserved
    0AAA|RRRR|1100|XXXX Reserved
    0AAA|RRRR|1101|CCCC Set Parameter On-the-fly
    0AAA|RRRR|1110|00CC Event Step Edit
    0000|0000|1111|0000 Diagnostics
    0001|XXXX|XXXX|XXXX Reserved
    :       :                  :
    1101|XXXX|XXXX|XXXX Reserved
    1110|CCCC|CCCC|CCCC Event Step Transfer
    1111|XXXX|XXXX|XXXX Reserved
```

Refer to the sections below for descriptions of using each of these types of bits (A, C, R, and X):

A (Axis Select) Bits:

These bits are used to select the axis to which the command is sent and from which the requested data is gathered. Use the following bit patterns to select the desired axis:

Bit #

14	**13**	**12**	**Axis**

0	0	0	0
0	0	1	1
0	1	0	2*
0	1	1	3*
1	0	0	4*
1	0	1	5*
1	1	0	6*
1	1	1	7*

* Commands to invalid axes are ignored.

C (Command Index) Bits:

These bits are used by the selected command. Refer to the command you wish to use for information on bits marked with C in the chart above.

R (Status Area Request) Bits:

These bits are used to select the data returned in digital inputs 0-7. Refer to the following chart for selecting the data of your choice.

Bit #

11	**10**	**9**	**8**	**Requested Data**
0	0	0	0	Command Position*
0	0	0	1	Target Position*
0	0	1	0	Actual Position*
0	0	1	1	Transducer Counts*
0	1	0	0	Status*
0	1	0	1	Drive*
0	1	1	0	Actual Speed*
0	1	1	1	Null Drive*
1	0	0	0	Step*
1	0	0	1	Link Value*

1	0	1	0	In Position/Auto Stop Errors†
1	0	1	1	Reserved
1	1	X	X	Reserved

* These are 16-bit values. At the time this data is requested, the full 16-bit value is stored in the RMC, and is thus latched. The low byte (bits 0-7) is returned on digital outputs 0-7 after the command word is acknowledged, and the high byte (bits 8-15) is returned on the same outputs after the command value is acknowledged.

† This value returns the In Position and Auto Stop Error bits. An axis In Position bit is set when the bit of the same name in the Status word is set. An axis Auto Stop Error bit is set when an error bit in the Status word is set that has been set to trigger a soft or hard stop using the Auto Stop parameter.

Note: Because commands are not executed until both the command and command data have been received, the In Position bit will reflect the In Position status prior to the command being issued.

After the command word is acknowledged, the In Position bits are put on the eight digital outputs using the following bit positions:

Output	**Represents…**
0	Axis 0 In Position
1	Axis 1 In Position
2	Axis 2* In Position
3	Axis 3* In Position
4	Axis 4* In Position
5	Axis 5* In Position
6	Axis 6* In Position
7	Axis 7* In Position

* This bit is always zero if the axis is not available.

After the command value is acknowledged, the Auto Stop Error bits are returned on the following digital outputs:

Output	**Represents…**
0	Axis 0 Auto Stop Error
1	Axis 1 Auto Stop Error
2	Axis 2* Auto Stop Error
3	Axis 3* Auto Stop Error
4	Axis 4* Auto Stop Error
5	Axis 5* Auto Stop Error
6	Axis 6* Auto Stop Error
7	Axis 7* Auto Stop Error

* This bit is always zero if the axis is not available.

X (Don’t Care) Bits:

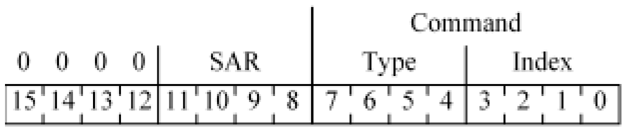
These bits are ignored.

A.3.2 Command Words for PROFIBUS-DP Compact Mode

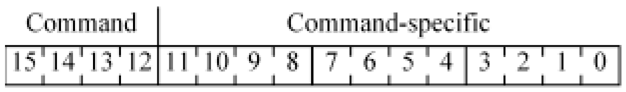
This topic applies to the RMC PROFIBUS module when Compact Mode is used; see Using the PROFIBUS-DP Compact Mode for details. For details on using Message Mode, see Using the PROFIBUS-DP Message Mode.

As described in Output Register Overview, each axis has one command register and one data out register. The use of the data out register depends on the command being selected.

The command register is divided into four nibbles (one nibble is four bits or a half byte). There are two formats used: Standard and Expanded. In the Standard format, the most significant nibble (MSN) is binary 0000 (hex 0), the next MSN is most often the Status Area Request (SAR) (described in Input Register Overview), the next MSN is the command type, and the use of the LSN is usually defined by the command itself. The following chart demonstrates the breakdown of the command word:



In the Expanded format, the MSN is never 0000 (hex 0) and describes the command type. The rest of the bits are defined by the command itself, as shown in the chart below:



The following table lists all commands that can be issued over the RMC's PROFIBUS-DP Compact Mode:

```
    1111|11  |    |
Bit#5432|1098|7654|3210
    -------------------
    0000|RRRR|0000|0000 No command
    0000|RRRR|0000|CCCC Open Loop Using Profile
    0000|RRRR|0001|CCCC Set Parameter
    0000|RRRR|0010|CCCC Set Profile
    0000|RRRR|0011|XXXX Reserved
    0000|RRRR|01CC|CCCC ASCII Commands
    0000|RRRR|1000|CCCC Go/Set Pressure Using Profile
    0000|0000|1001|CCCC Get Parameter
    0000|0000|1010|CCCC Get Profile
    0000|RRRR|1011|XXXX Reserved
    0000|RRRR|1100|XXXX Reserved
    0000|RRRR|1101|CCCC Set Parameter On-the-fly
    0000|RRRR|1110|00CC Event Step Edit
    0000|0000|1111|0000 Diagnostics

    0001|XXXX|XXXX|XXXX Reserved
    0010|CCCC|CCCC|CCCC Download Graph Data
    :    :         :
    1101|CCCC|CCCC|CCCC Download Graph Data
    1110|CCCC|CCCC|CCCC Event Step Transfer
    1111|XXXX|XXXX|XXXX Reserved
```

Refer to the sections below for descriptions of using each of these types of bits (C, R, and X):

C (Command) Bits:

These bits are used by the selected command. Refer to the command you wish to use for information on bits marked with C in the chart above.

R (Status Area Request) Bits:

These four bits define the Status Area Request (SAR) field. They request the data to be returned in the second input register for the axis. For details on input registers, see Input Register Overview. The table below lists the choices for this field:

Bit #				
11	**10**	**9**	**8**	**Requested Data**
0	0	0	0	Command Position*
0	0	0	1	Target Position*
0	0	1	0	Actual Position*
0	0	1	1	Transducer Counts*
0	1	0	0	Status*
0	1	0	1	Drive*
0	1	1	0	Actual Speed*
0	1	1	1	Null Drive*
1	0	0	0	Step*
1	0	0	1	Link Value*
1	0	1	X	Reserved
1	1	X	X	Reserved

X (Don’t Care) Bits:

These bits are ignored.

See also:

Input Register Overview

Output Register Overview

Communicating with the RMC100 using PROFIBUS-DP

A.3.3 Receiving Data from the Motion Controller

The method of receiving data from the RMC in the PLC depends on the type of communication module you are using.

- If you are using PROFIBUS-DP, refer to Input Register Overview.
- If you are using a Communication Digital I/O in Command Mode, refer to Command Words for Command Mode.

A.3.4 Sending Data from the PLC

The method of sending data to the RMC from the PLC depends on the type of communication module you are using.

- If you are using PROFIBUS-DP, refer to Output Register Overview.
- If you are using a Communication Digital I/O in Command Mode, refer to Command Words for Command Mode.

A.3.5 Open Loop Using Profile Commands

Format: 0AAA RRRR 0000 NNNN

R Used for Status Area Request

N Used for Command Index described below

A Used only by Communication Digital I/O; 0 for PROFIBUS-DP

These commands allow the controller to tell the RMC100 to change the output drive to a specified value with respect to null. The drive output will change at a rate specified by the select pre-stored profile.

Note: This command shares the profile table used by the Go Using Profile commands. Profiles used for one type of command should NOT be used for the other.

These commands behave identical to the Open Loop ASCII command, except that the specified stored profile information is copied into the active Mode, Accel, Decel and Speed first. Recall that the Speed field of the profile is not used by the Open Loop. Refer to that command for further details.

The format of the Command Register for the Open Loop Using Profile commands is given below:

```
1111|11 | |

BIT # 5432|1098|7654|3210
```

```
-------------------
HEX |SAR |CMND|INDX
VALUE -------------------
0X00 0AAA|XXXX|0000|0000 NO COMMAND
0X01 0AAA|XXXX|0000|0001 OPEN LOOP USING PROFILE 1
0X02 0AAA|XXXX|0000|0010 OPEN LOOP USING PROFILE 2
0X03 0AAA|XXXX|0000|0011 OPEN LOOP USING PROFILE 3
0X04 0AAA|XXXX|0000|0100 OPEN LOOP USING PROFILE 4
0X05 0AAA|XXXX|0000|0101 OPEN LOOP USING PROFILE 5
0X06 0AAA|XXXX|0000|0110 OPEN LOOP USING PROFILE 6
0X07 0AAA|XXXX|0000|0111 OPEN LOOP USING PROFILE 7
0X08 0AAA|XXXX|0000|1000 OPEN LOOP USING PROFILE 8
0X09 0AAA|XXXX|0000|1001 OPEN LOOP USING PROFILE 9
0X0A 0AAA|XXXX|0000|1010 OPEN LOOP USING PROFILE 10
0X0B 0AAA|XXXX|0000|1011 OPEN LOOP USING PROFILE 11
0X0C 0AAA|XXXX|0000|1100 OPEN LOOP USING PROFILE 12
0X0D 0AAA|XXXX|0000|1101 OPEN LOOP USING PROFILE 13
0X0E 0AAA|XXXX|0000|1110 OPEN LOOP USING PROFILE 14
0X0F 0AAA|XXXX|0000|1111 OPEN LOOP USING PROFILE 15
```

Note: When this command's index is zero (0), it is treated as though no new command is given.

The command data represents the requested drive in millivolts.

Example for PROFIBUS-DP in Compact Mode with Sync:

Suppose you would like axis 0 to go into open loop at 2000mV of drive using profile 10 and axis 1 to go into open loop at 4000mV of drive using profile 11. You would send the commands in the following format:

	SAR CMND INDX (HEX)	
O+1	0000|XXXX|0000|1010 (0X0A)	Issue an Open Loop using Profile 10 command
+2	2000 (07D0)	Requested Drive

```
O+3 0000|XXXX|0000|1011 (0X0B)
```
Issue an Open Loop using Profile 11 command

```
 +4                4000 (0FA0)
```
Requested Drive

Example for Digital I/O:

Suppose you would like axis 0 to go into open loop at 2000mV of drive using profile 10. You would send the command in the following format:

Send on Command Strobe going high:

`0000|XXXX|0000|1010 (0X0A)` Issue an Open Loop using Profile 10 command

Receive after Acknowledge:

`XXXX|XXXX     (XX)` Low byte of requested data

Send on Command Strobe going high:

`0000|0111|1101|0000 (07D0)` Requested Drive

Receive after Acknowledge:

`XXXX|XXXX     (XX)` High byte of requested data

A.3.6 Set Parameter Commands

Format: 0AAA RRRR 0001 NNNN

R Used for Status Area Request

N Used for Command Index described below

A Used only by Communication Digital I/O; 0 for PROFIBUS-DP

These commands allow the Programmable Controller to download new initialization parameters to the RMC. New parameters can be downloaded at any time, but they do not take effect until a 'P' ASCII command is issued. An axis must be stopped when issuing the 'P' ASCII command.

The format of the Command Register for the Set Parameter commands is given below:

```
     1111|11  |    |

BIT# 5432|1098|7654|3210

     -------------------

HEX      |SAR |CMND|INDX

VALUE-------------------

0X10 0AAA|XXXX|0001|0000 Set CONFIGURATION
```

```
0X11 0AAA|XXXX|0001|0001 Set SCALE

0X12 0AAA|XXXX|0001|0010 Set OFFSET

0X13 0AAA|XXXX|0001|0011 Set EXTEND LIMIT

0X14 0AAA|XXXX|0001|0100 Set RETRACT LIMIT

0X15 0AAA|XXXX|0001|0101 Set PROPORTIONAL GAIN

0X16 0AAA|XXXX|0001|0110 Set INTEGRAL GAIN

0X17 0AAA|XXXX|0001|0111 Set DIFFERENTIAL GAIN

0X18 0AAA|XXXX|0001|1000 Set EXTEND FEED FORWARD

0X19 0AAA|XXXX|0001|1001 Set RETRACT FEED FORWARD

0X1A 0AAA|XXXX|0001|1010 Set EXTEND ACCEL FEED FORWARD

0X1B 0AAA|XXXX|0001|1011 Set RETRACT ACCEL FEED FORWARD

0X1C 0AAA|XXXX|0001|1100 Set DEAD BAND ELIMINATOR

0X1D 0AAA|XXXX|0001|1101 Set IN POSITION

0X1E 0AAA|XXXX|0001|1110 Set FOLLOWING ERROR

0X1F 0AAA|XXXX|0001|1111 Set AUTO STOP
```

The command data represents the new parameter value.

Example for PROFIBUS-DP in Compact Mode with Sync:
Suppose you have an RMC100-M1-PROFI, and you would like to set the Extend Limit for both axes. You would send commands with the following format:

```
          SAR  CMND INDX (HEX)

O+1 0000|XXXX|0001|0011 (0X13) Sets axis 0 Extend Limit

 +2               24000 (5DC0) New Extend Limit value

O+3 0000|XXXX|0001|0011 (0X13) Sets axis 1 Extend Limit

 +4               12000 (2EE0) New Extend Limit value
```

Example for Digital I/O:
Suppose you would like to set the Extend Limit for axis 0. You would send commands with the following format:

Send on Command Strobe going high:

`0000|XXXX|0001|0011 (0X13)` Set axis 0 Extend Limit

Receive after Acknowledge:

`XXXX|XXXX     (XX)` Low byte of requested data

Send on Command Strobe going high:

`0101|1101|1100|0000 (5DC0)` New Extend Limit value

Receive after Acknowledge:

`XXXX|XXXX     (XX)` High byte of requested data

A.3.7 Set Profile Commands

Format: 0AAA RRRR 0010 NNNN

R Used for Status Area Request

N Used for Command Index described below

A Used only by Communication Digital I/O; 0 for PROFIBUS-DP

These commands allow the programmer to change motion profiles stored in the RMC. Only one value in one profile can be changed per axis each time the synchronization register is changed, but the profiles can be changed while the axis is moving. The new profile will be used by the next Go Using Profile command specifying that profile. Recall that new Go and Go Using Profile commands can be given while the axis is moving.

The format of the Command Register for the Set Profile commands is given below:

```
     1111|11  |    |
BIT# 5432|1098|7654|3210
     -------------------
HEX      |SAR |CMND|INDX
VALUE-------------------
0X20 0AAA|XXXX|0010|0000 SET PROFILE 0, 4, 8 or 12 MODE
0X21 0AAA|XXXX|0010|0001 SET PROFILE 0, 4, 8 or 12 ACCEL
0X22 0AAA|XXXX|0010|0010 SET PROFILE 0, 4, 8 or 12 DECEL
0X23 0AAA|XXXX|0010|0011 SET PROFILE 0, 4, 8 or 12 SPEED
0X24 0AAA|XXXX|0010|0100 SET PROFILE 1, 5, 9 or 13 MODE
0X25 0AAA|XXXX|0010|0101 SET PROFILE 1, 5, 9 or 13 ACCEL
```

`0X26 0AAA|XXXX|0010|0110` SET PROFILE 1, 5, 9 or 13 DECEL

`0X27 0AAA|XXXX|0010|0111` SET PROFILE 1, 5, 9 or 13 SPEED

`0X28 0AAA|XXXX|0010|1000` SET PROFILE 2, 6, 10 or 14 MODE

`0X29 0AAA|XXXX|0010|1001` SET PROFILE 2, 6, 10 or 14 ACCEL

`0X2A 0AAA|XXXX|0010|1010` SET PROFILE 2, 6, 10 or 14 DECEL

`0X2B 0AAA|XXXX|0010|1011` SET PROFILE 2, 6, 10 or 14 SPEED

`0X2C 0AAA|XXXX|0010|1100` SET PROFILE 3, 7, 11 or 15 MODE

`0X2D 0AAA|XXXX|0010|1101` SET PROFILE 3, 7, 11 or 15 ACCEL

`0X2E 0AAA|XXXX|0010|1110` SET PROFILE 3, 7, 11 or 15 DECEL

`0X2F 0AAA|XXXX|0010|1111` SET PROFILE 3, 7, 11 or 15 SPEED

Notice that for each command, four different profile numbers are listed. The actual profile affected is determined by the axis issuing the command. The following chart shows which commands affect which profiles when issued on a particular axis. Notice that an RMC100-M1-PROFI can only set the first eight (8) profiles:

COMMAND	**AXIS 0**	**AXIS 1**	**AXIS 2**	**AXIS 3**
0x20-0x23	0	4	8	12
0x24-0x27	1	5	9	13
0x28-0x2B	2	6	10	14
0x2C-0x2F	3	7	11	15

The command data represents the value of the field to be set in the profile.

Example for PROFIBUS-DP in Compact Mode with Sync:
We wish to set profiles 2 and 7 with the following values:

	PROFILE 2	**PROFILE 7**
MODE	0x0001	0x0001
ACCEL	100	150
DECEL	70	70
SPEED	12000	20000

Looking at the chart above, we can see that we issue Set Profile commands to axis 0 to set profile 2 and we use axis 1 Set Profile commands to set profile 7. Therefore, we can do both at the same time. This would take four scans because we can send one word of each profile on each axis per scan:

First scan:

O+1	0X28h	(Set profile 2 MODE)
O+2	0001h	(Value of profile 2 MODE)
O+3	0X2Ch	(Set profile 7 MODE)
O+4	0001h	(Value of profile 7 MODE)

Second scan:

O+1	0X29h	(Set profile 2 ACCEL)
O+2	100	(Value of profile 2 ACCEL)
O+3	0X2Dh	(Set profile 7 ACCEL)
O+4	150	(Value of profile 2 ACCEL)

Third scan:

O+1	0X2Ah	(Set profile 2 DECEL)
O+2	70	(Value of profile 2 DECEL)
O+3	0X2Eh	(Set profile 7 DECEL)
O+4	70	(Value of profile 2 DECEL)

Fourth scan:

O+1	0X2Bh	(Set profile 2 SPEED)
O+2	12000	(Value of profile 2 SPEED)
O+3	0X2Fh	(Set profile 7 SPEED)
O+4	20000	(Value of profile 2 SPEED)

Example for Digital I/O:
We wish to set profile 7 to the following values:

	PROFILE 7
MODE	0x0001
ACCEL	100
DECEL	70

SPEED 12000

Looking at the chart above, we can see that Set Profile commands must be to axis 1 to set profile 7. Therefore, we send the following commands:

First Command:

Send on Command Strobe going high:

`0001|XXXX|0010|1000 (1X28)` Set profile 7 Mode

Receive after Acknowledge:

`XXXX|XXXX (XX)` Low byte of requested data

Send on Command Strobe going high:

`0000|0000|0000|0001 (0000)` Profile 7 Mode

Receive after Acknowledge:

`XXXX|XXXX (XX)` High byte of requested data

Second Command:

Send on Command Strobe going high:

`0001|XXXX|0010|1001 (1X29)` Set profile 7 Speed

Receive after Acknowledge:

`XXXX|XXXX (XX)` Low byte of requested data

Send on Command Strobe going high:

`0000|0000|0110|0100 (0064)` Profile 7 Speed

Receive after Acknowledge:

`XXXX|XXXX (XX)` High byte of requested data

Third Command:

Send on Command Strobe going high:

`0001|XXXX|0010|1010 (1X2A)` Set profile 7 Decel

Receive after Acknowledge:

`XXXX|XXXX (XX)` Low byte of requested data

Send on Command Strobe going high:

`0000|0000|0100|0110 (0046)` Profile 7 Decel

Receive after Acknowledge:

`XXXX|XXXX (XX)` High byte of requested data

Fourth Command:

Send on Command Strobe going high:

`0001|XXXX|0010|1011 (1X2B)` Set profile 7 Speed

Receive after Acknowledge:

`XXXX|XXXX (XX)` Low byte of requested data

Send on Command Strobe going high:

`0010|1110|1110|0000 (2EE0)` Profile 7 Speed

Receive after Acknowledge:

`XXXX|XXXX (XX)` High byte of requested data

A.3.8 ASCII Commands

Format: 0AAA RRRR 01NN NNNN

R Used for Status Area Request

N Used for Command Index described below

A Used only by Communication Digital I/O; 0 for PROFIBUS-DP

The ASCII commands can be used to send any command that can be sent from RMCWin. These commands are listed in the Command Field topic. To send an ASCII command, use the hex value for the commands listed in the table in the Command Field topic and use the desired Status Area Request value. For example, to send a Change Deceleration (D) command (hexadecimal 0x44) using a Status Area Request value of 2, you would send a command register of 0x0244.

The command data sent represents the Command Value of the ASCII command being sent, if one is required.

A.3.9 Go/Set Pressure Using Profile Commands

Format: 0AAA RRRR 1000 NNNN

R Used for Status Area Request

N Used for Command Index described below

A Used only by Communication Digital I/O; 0 for PROFIBUS-DP

This command is treated differently when issued to a pressure or position axis. When issued to a position axis, it issues a Go (G) command to the axis after copying the selected profile to the Mode, Accel, DECEL and Speed fields.

When issued to a pressure axis, it issues a Set Pressure (^) command to the axis after copying the selected profile to the Mode, Pressure Set A, Pressure Set B, and Ramp Time fields. NOTE: In the profile editor, these fields are labeled Mode, Accel, Decel, and Speed.

The format of the Command Register for the Go/Set Pressure Using Profile commands is given below:

```
     1111|11  |    |
BIT# 5432|1098|7654|3210
     -------------------
HEX      |SAR |CMND|INDX
VALUE-------------------
0X80 0AAA|XXXX|1000|0000 GO/PRESSURE USING PROFILE 0
0X81 0AAA|XXXX|1000|0001 GO/PRESSURE USING PROFILE 1
0X82 0AAA|XXXX|1000|0010 GO/PRESSURE USING PROFILE 2
0X83 0AAA|XXXX|1000|0011 GO/PRESSURE USING PROFILE 3
0X84 0AAA|XXXX|1000|0100 GO/PRESSURE USING PROFILE 4
0X85 0AAA|XXXX|1000|0101 GO/PRESSURE USING PROFILE 5
0X86 0AAA|XXXX|1000|0110 GO/PRESSURE USING PROFILE 6
0X87 0AAA|XXXX|1000|0111 GO/PRESSURE USING PROFILE 7
0X88 0AAA|XXXX|1000|1000 GO/PRESSURE USING PROFILE 8
0X89 0AAA|XXXX|1000|1001 GO/PRESSURE USING PROFILE 9
0X8A 0AAA|XXXX|1000|1010 GO/PRESSURE USING PROFILE 10
0X8B 0AAA|XXXX|1000|1011 GO/PRESSURE USING PROFILE 11
0X8C 0AAA|XXXX|1000|1100 GO/PRESSURE USING PROFILE 12
0X8D 0AAA|XXXX|1000|1101 GO/PRESSURE USING PROFILE 13
0X8E 0AAA|XXXX|1000|1110 GO/PRESSURE USING PROFILE 14
0X8F 0AAA|XXXX|1000|1111 GO/PRESSURE USING PROFILE 15
```

The command data represents the requested position in position units when issued to a position axis, and a requested pressure in pressure units when issued to a pressure axis.

Example for PROFIBUS-DP in Compact Mode with Sync:

Suppose you have an RMC100-M1-PROFI, and you would like to move axis 0 to 5000 position units using profile 2 and axis 1 to 10000 position units using profile 5. You would send commands with the following format:

```
          SAR CMND INDX (HEX)

O+1 0000|XXXX|1000|0010 (0X82) Axis 0 Go using profile 2

 +2                5000 (1388) Axis 0 Requested Position

O+3 0000|XXXX|1000|0101 (0X85) Axis 1 Go using profile 5

 +4               10000 (2710) Axis 1 Requested Position
```

Example for Digital I/O:

Suppose you would like to move axis 0 to 5000 position units using profile 2. You would send commands with the following format:

Send on Command Strobe going high:

```
  0000|XXXX|1000|0010 (0X82) Axis 0 Go using profile 2
```

Receive after Acknowledge:

```
            XXXX|XXXX   (XX) Low byte of requested data
```

Send on Command Strobe going high:

```
  0001|0011|1000|1000 (1388) Axis 0 Requested Position
```

Receive after Acknowledge:

```
            XXXX|XXXX   (XX) High byte of requested data
```

A.3.10 Get Parameter Commands

Format: 0AAA 0000 1001 NNNN

N Used for Command Index described below

A Used only by Communication Digital I/O; 0 for PROFIBUS-DP

These commands allow the Programmable Controller to read the current state of the RMC initialization parameters. The requested parameter will be returned in the same way that Status Area Request data is returned. Therefore, bits 8-11 are not used.

The format of the Command Register for the Get Parameter commands is given below:

Note: The Status Area Request area (bits 8-11) of the command and the entire command data are ignored.

```
     1111|11  |    |
BIT# 5432|1098|7654|3210
-------------------
HEX      |    |CMND|INDX
VALUE -------------------
0X90 0AAA|XXXX|1001|0000 Get CONFIGURATION
0X91 0AAA|XXXX|1001|0001 Get SCALE
0X92 0AAA|XXXX|1001|0010 Get OFFSET
0X93 0AAA|XXXX|1001|0011 Get EXTEND LIMIT
0X94 0AAA|XXXX|1001|0100 Get RETRACT LIMIT
0X95 0AAA|XXXX|1001|0101 Get PROPORTIONAL GAIN
0X96 0AAA|XXXX|1001|0110 Get INTEGRAL GAIN
0X97 0AAA|XXXX|1001|0111 Get DIFFERENTIAL GAIN
0X98 0AAA|XXXX|1001|1000 Get EXTEND FEED FORWARD
0X99 0AAA|XXXX|1001|1001 Get RETRACT FEED FORWARD
0X9A 0AAA|XXXX|1001|1010 Get EXTEND ACCEL FEED FORWARD
0X9B 0AAA|XXXX|1001|1011 Get RETRACT ACCEL FEED FORWARD
0X9C 0AAA|XXXX|1001|1100 Get DEAD BAND ELIMINATOR
0X9D 0AAA|XXXX|1001|1101 Get IN POSITION
0X9E 0AAA|XXXX|1001|1110 Get FOLLOWING ERROR
0X9F 0AAA|XXXX|1001|1111 Get AUTO STOP
```

The data returned where the Status Area Request data would be returned is the requested parameter.

Example for PROFIBUS-DP in Compact Mode with Sync:
Suppose you have an RMC100-M1-PROFI, and you would like to get the Scale from both axes. You would send commands with the following format:

```
              CMND INDX (HEX)
O+1 0000|0000|1001|0001 (0091) Requests Scale for axis 0
 +2 XXXX|XXXX|XXXX|XXXX (XXXX) Ignored
```

```
O+3 0000|0000|1001|0001 (0091) Requests Scale for axis 1

 +4 XXXX|XXXX|XXXX|XXXX (XXXX) Ignored
```

After the Synchronization Output register is incremented, the RMC will process the commands and update the Synchronization Input register to match. At this time, the input registers would hold the following values:

```
I+1 XXXX|XXXX|XXXX|XXXX (XXXX) Status of axis 0

 +2 XXXX|XXXX|XXXX|XXXX (XXXX) Scale of axis 0

I+3 XXXX|XXXX|XXXX|XXXX (XXXX) Status of axis 1

 +4 XXXX|XXXX|XXXX|XXXX (XXXX) Scale of axis 1
```

Example for Digital I/O:
Suppose you would like to get the Scale from axis 0. You would send the following command:

Send on Command Strobe going high:

```
0000|0000|1001|0001 (0091) Get Scale for axis 0
```

Receive after Acknowledge:

```
          1111|1111   (FF) Low byte of Scale
```

Send on Command Strobe going high:

```
XXXX|XXXX|XXXX|XXXX (XXXX) Unused
```

Receive after Acknowledge:

```
          0111|1111   (7F) High byte of Scale
```

A.3.11 Get Profile Commands

Format: 0AAA 0000 1010 NNNN
N Used for Command Index described below

A Used only by Communication Digital I/O; 0 for PROFIBUS-DP

These commands allow the programmer to retrieve the RMC's stored motion profiles. Only one value in one profile can be read per axis each time the synchronization register is changed

The format of the Command Register for the Get Profile commands is given below:

Note: The Status Area Request area (bits 8-11) of the command and the command data are ignored.

```
     1111|11  |    |

BIT# 5432|1098|7654|3210
```

```
     -------------------
HEX      |    |CMND|INDX
VALUE-------------------
0XA0 0AAA|XXXX|1010|0000 GET PROFILE 0, 4, 8 or 12 MODE
0XA1 0AAA|XXXX|1010|0001 GET PROFILE 0, 4, 8 or 12 ACCEL
0XA2 0AAA|XXXX|1010|0010 GET PROFILE 0, 4, 8 or 12 DECEL
0XA3 0AAA|XXXX|1010|0011 GET PROFILE 0, 4, 8 or 12 SPEED
0XA4 0AAA|XXXX|1010|0100 GET PROFILE 1, 5, 9 or 13 MODE
0XA5 0AAA|XXXX|1010|0101 GET PROFILE 1, 5, 9 or 13 ACCEL
0XA6 0AAA|XXXX|1010|0110 GET PROFILE 1, 5, 9 or 13 DECEL
0XA7 0AAA|XXXX|1010|0111 GET PROFILE 1, 5, 9 or 13 SPEED
0XA8 0AAA|XXXX|1010|1000 GET PROFILE 2, 6, 10 or 14 MODE
0XA9 0AAA|XXXX|1010|1001 GET PROFILE 2, 6, 10 or 14 ACCEL
0XAA 0AAA|XXXX|1010|1010 GET PROFILE 2, 6, 10 or 14 DECEL
0XAB 0AAA|XXXX|1010|1011 GET PROFILE 2, 6, 10 or 14 SPEED
0XAC 0AAA|XXXX|1010|1100 GET PROFILE 3, 7, 11 or 15 MODE
0XAD 0AAA|XXXX|1010|1101 GET PROFILE 3, 7, 11 or 15 ACCEL
0XAE 0AAA|XXXX|1010|1110 GET PROFILE 3, 7, 11 or 15 DECEL
0XAF 0AAA|XXXX|1010|1111 GET PROFILE 3, 7, 11 or 15 SPEED
```

Notice that for each command, four different profile numbers are listed. The actual profile read is determined by the axis issuing the command. The following chart shows which profile is affected by a given command on a given axis. Notice that an RMC100-M1-PROFI can only get the first eight (8) profiles:

COMMAND	**AXIS 0**	**AXIS 1**	**AXIS 2**	**AXIS 3**
0XA0-0XA3	0	4	8	12
0XA4-0XA7	1	5	9	13
0XA8-0XAB	2	6	10	14
0XAC-0XAF	3	7	11	15

The data returned where the Status Area Request data would be returned is the requested profile field.

Example for PROFIBUS-DP in Compact Mode with Sync:

We wish to get profiles 2 and 7 into the RMC. Looking at the chart above, we can see that we issue Get Profile commands to axis 0 to get profile 2 and we use axis 1 Get Profile commands to get profile 7. Therefore, we can do both at the same time. Suppose that the RMC had the following values for profiles 2 and 7:

	PROFILE 2	**PROFILE 7**
MODE	0x0001	0x0001
ACCEL	100	150
DECEL	70	70
SPEED	12000	20000

This would take at least five scans because we can get one word of each profile on each axis per scan:

First scan:

O+1	00A8h	(Get profile 2 MODE)
O+2	XXXXh	(Unused)
O+3	00ACh	(Get profile 7 MODE)
O+4	XXXXh	(Value of profile 7 MODE)

Second scan:

After the Synchronization Output register is incremented, the RMC will process the commands and update the Synchronization Input register to match. At this time, the input registers would hold the following values:

I+1	XXXXh	(Axis 0 STATUS)
I+2	0001h	(Profile 2 MODE)

I+3	XXXXh	(Axis 1 STATUS)
I+4	0001h	(Profile 7 MODE)

We can now send the second set of requests:

O+1	00A9h	(Get profile 2 ACCEL)
O+2	XXXXh	(Unused)
O+3	00ADh	(Get profile 7 ACCEL)
O+4	XXXXh	(Unused)

Third scan:

After the Synchronization Output register is incremented, the RMC will process the commands and update the Synchronization Input register to match. At this time, the input registers would hold the following values:

I+1	XXXXh	(Axis 0 STATUS)
I+2	100h	(Profile 2 ACCEL)
I+3	XXXXh	(Axis 1 STATUS)
I+4	150h	(Profile 7 ACCEL)

We can now send the third set of requests:

O+1	00AAh	(Get profile 2 DECEL)
O+2	XXXXh	(Unused)
O+3	00AEh	(Get profile 7 DECEL)
O+4	XXXXh	(Unused)

Fourth scan:

After the Synchronization Output register is incremented, the RMC will process the commands and update the Synchronization Input register to match. At this time, the input registers would hold the following values:

I+1	XXXXh	(Axis 0 STATUS)
I+2	70h	(Profile 7 DECEL)
I+3	XXXXh	(Axis 1 STATUS)
I+4	70h	(Profile 7 DECEL)

We can now send the fourth set of requests:

O+1	00ABh	(Get profile 2 SPEED)
O+2	XXXXh	(Unused)
O+3	00AFh	(Get profile 7 SPEED)
O+4	XXXXh	(Unused)

Fifth scan:

After the Synchronization Output register is incremented, the RMC will process the commands and update the Synchronization Input register to match. At this time, the input registers would hold the following values:

I+1	XXXXh	(Axis 0 STATUS)
I+2	12000h	(Profile 7 SPEED)
I+3	XXXXh	(Axis 1 STATUS)
I+4	12000h	(Profile 7 SPEED)

Example for Digital I/O:

We wish to get profile 7 into the RMC. Looking at the chart above, we can see that we issue Get

Profile commands to axis 1 to get profile 7. Suppose that the RMC had the following values for profile 7:

	PROFILE 7
MODE	0x0001
ACCEL	100
DECEL	70
SPEED	12000

The following steps would be taken to read profile 7:

First Command:

Send on Command Strobe going high:

`0001|0000|1010|1000 (10A8)` Get Profile 7 Mode

Receive after Acknowledge:

`0000|0001    (01)` Low byte of Mode

Send on Command Strobe going high:

`XXXX|XXXX|XXXX|XXXX (XXXX)` Unused

Receive after Acknowledge:

`0000|0000    (00)` High byte of Mode

Second Command:

Send on Command Strobe going high:

`0001|0000|1010|1001 (10A9)` Get Profile 7 Accel

Receive after Acknowledge:

`0110|0100    (64)` Low byte of Accel

Send on Command Strobe going high:

`XXXX|XXXX|XXXX|XXXX (XXXX)` Unused

Receive after Acknowledge:

`0000|0000    (00)` High byte of Accel

Third Command:

Send on Command Strobe going high:

```
  0001|0000|1010|1010 (10AA) Get Profile 7 Decel
```

Receive after Acknowledge:

```
            0100|0110   (46) Low byte of Decel
```

Send on Command Strobe going high:

```
  XXXX|XXXX|XXXX|XXXX (XXXX) Unused
```

Receive after Acknowledge:

```
            0000|0000   (00) High byte of Decel
```

Fourth Command:

Send on Command Strobe going high:

```
  0001|0000|1010|1011 (10AB) Get Profile 7 Speed
```

Receive after Acknowledge:

```
            1110|0000   (E0) Low byte of Speed
```

Send on Command Strobe going high:

```
  XXXX|XXXX|XXXX|XXXX (XXXX) Unused
```

Receive after Acknowledge:

```
            0010|1110   (2E) High byte of Speed
```

A.3.12 Set Parameter On-the-fly PLC Commands

Format: 0AAA RRRR 1101 NNNN

R Used for Status Area Request

N Used for Command Index described below

A Used only by Communication Digital I/O; 0 for PROFIBUS-DP

Note: This topic describes issuing the Set Parameter On-the-fly commands through PROFIBUS-DP's Compact Mode or Communication DI/O's Command Mode. For a discussion on these commands issued any other way (e.g. from RMCWin, the Event Step table, Ethernet, Modbus Plus), see Set Parameter On-the-Fly.

The format of the Command Register for the Set Parameters On-the-fly commands is given below:

```
     1111|11  |    |
BIT# 5432|1098|7654|3210
```

```
     -------------------
HEX      |    |CMND|INDX
VALUE-------------------
0xD0 0000|XXXX|1101|0000 SET CONFIG WORD ON-THE-FLY
0xD1 0000|XXXX|1101|0001 RESERVED
0xD2 0000|XXXX|1101|0010 RESERVED
0xD3 0000|XXXX|1101|0011 RESERVED
0xD4 0000|XXXX|1101|0100 RESERVED
0xD5 0000|XXXX|1101|0101 SET PROPORTIONAL GAIN ON-THE-FLY
0xD6 0000|XXXX|1101|0110 SET INTEGRAL GAIN ON-THE-FLY
0xD7 0000|XXXX|1101|0111 SET DIFFERENTIAL GAIN ON-THE-FLY
0xD8 0000|XXXX|1101|1000 SET EXTEND FEED FORWARD ON-THE-FLY
0xD9 0000|XXXX|1101|1001 SET RETRACT FEED FORWARD ON-THE-FLY
0xDA 0000|XXXX|1101|1010 SET EXTEND ACC FEED FORWARD ON-THE-FLY
0xDB 0000|XXXX|1101|1011 SET RETRACT ACC FEED FORWARD ON-THE-FLY
0xDC 0000|XXXX|1101|1100 SET DEAD BAND ELIMINATOR ON-THE-FLY
0xDD 0000|XXXX|1101|1101 SET IN POSITION WINDOW ON-THE-FLY
0xDE 0000|XXXX|1101|1110 SET FOLLOWING ERROR WINDOW ON-THE-FLY
0xDF 0000|XXXX|1101|1111 SET AUTO STOP ERROR MASK ON-THE-FLY
```

The command data represents the new parameter value.

If these commands are issued to a pressure control axis, then the parameters above are replaced by the pressure-axis equivalents.

You will notice that the Scale, Offset, Extend Limit, and Retract Limit cannot be changed on the fly. They can be changed by stopping the axis, changing the value, and issuing a Set Parameters (P) command.

Warning: Changing any of the gains or feed forwards while the axis is moving will generate a drive discontinuity in most cases.

Example for PROFIBUS-DP in Compact Mode with Sync:
Suppose you have an RMC100-M1-PROFI, and you would like to set the Proportional Gain for both axes. You would send commands with the following format:

```
          SAR  CMND INDX (HEX)
O+1 0000|XXXX|1101|0101 (0XD5) Sets axis 0 Proportional Gain
```

```
+2                  150 (0096) New Proportional Gain value

O+3 0000|XXXX|1101|0101 (0XD5) Sets axis 1 Proportional Gain

 +4                 175 (00AF) New Proportional Gain value
```

Example for Digital I/O:

Suppose you would like to set the Extend Limit for axis 1. You would send commands with the following format:

Send on Command Strobe going high:

```
  0001|XXXX|1101|0101 (1XD5) Set axis 0 Proportional Gain
```

Receive after Acknowledge:

```
            XXXX|XXXX    (XX) Low byte of requested data
```

Send on Command Strobe going high:

```
  0000|0000|1001|0110 (0096) New Proportional Gain value
```

Receive after Acknowledge:

```
            XXXX|XXXX    (XX) High byte of requested data
```

A.3.13 Event Step Edit Commands

Format: 0AAA RRRR 1110 NNNN

R Used for Status Area Request

N Used for Command Index described below

A Used only by Communication Digital I/O; 0 for PROFIBUS-DP

These commands let you change an event parameter's value across a range of steps in a single scan. The format of the Command Register for the Event Step Edit commands is given below:

```
     1111|11  |    |

BIT #5432|1098|7654|3210

-------------------

HEX      |SAR |CMND|INDX

VALUE -------------------

0XE0 0AAA|XXXX|1110|0000 Start Step Number (0 to 255)

0XE1 0AAA|XXXX|1110|0001 End Step Number (0 to 255; > E0)
```

`0XE2 0AAA|XXXX|1110|0010` Parameter to Modify (0 to 7)

`0XE3 0AAA|XXXX|1110|0011` Value to be Used

For command types 0XE0 and 0XE1, the command data represents a step number. For command type 0XE2, the command data represents the field that will be modified by 0XE3. Use the following table to select a field:

0XE2 DATA VALUE	**STEP FIELD**
0	MODE
1	ACCEL
2	DECEL
3	SPEED
4	COMMAND VALUE
5	COMMAND/COMMANDED AXES
6	LINK TYPE/NEXT
7	LINK VALUE

To edit the Event Step table, each of these commands must be issued once to initialize the parameters; this will take four scans. Once the beginning step number, the ending step number, and the parameter to be changed have been defined, all the steps can be changed in a single scan by sending 0XE3 command with the new parameter value. All four of these commands must be sent on the same axis.

Example for PROFIBUS-DP in Compact Mode with Sync:

If you have a step table that is 100 steps long and you want to change the time delay link value in each step, it would take 100 scans by using the Event Step Transfer commands. If the scan time is 20ms, this would take 2 seconds. By using the Event Step Edit commands, it would take 80ms for the first change (four scans times 20ms/scan) and only 20ms for another change.

Assuming we want to modify steps 0 to 99 and the new link value will be 500, the following commands would be sent:

First scan:	O+1	0XE0h	(Set Start Step Number)
	O+2	0	(Start Step Number of

			0)
Second scan:	O+1	0XE1h	(Set End Step Number)
	O+2	99	(End Step Number of 99)
Third scan:	O+1	0XE2h	(Set Parameter to Modify)
	O+2	7	(Select Link Value)
Fourth scan:	O+1	0XE3h	(Set Actual Value)
	O+2	500	(Link Value itself)

Example for Digital I/O:

If you have a step table that is 100 steps long and you want to change the time delay link value in each step, it would take 100 scans by using the Event Step Transfer commands. If the scan time is 20ms, this would take 2 seconds. By using the Event Step Edit commands, it would take 80ms for the first change (four scans times 20ms/scan) and only 20ms for another change.

Assuming we want to modify steps 0 to 99 and the new link value will be 500, the following commands would be sent:

First scan:

Send on Command Strobe going high:

`0000|XXXX|1110|0000 (0XE0)` Set start step number

Receive after Acknowledge:

`XXXX|XXXX (XX)` Low byte of requested data

Send on Command Strobe going high:

`0000|0000|0000|0000 (0000)` Start at step number 0

Receive after Acknowledge:

`XXXX|XXXX (XX)` High byte of requested data

Second scan:

Send on Command Strobe going high:

`0000|XXXX|1110|0001 (0XE1)` Set End Step Number

Receive after Acknowledge:

`XXXX|XXXX (XX)` Low byte of requested data

Send on Command Strobe going high:

`0000|0000|0110|0011 (0063)` End Step Number of 99

Receive after Acknowledge:

`XXXX|XXXX (XX)` High byte of requested data

Third scan:

Send on Command Strobe going high:

`0000|XXXX|1110|0010 (0XE2)` Set Parameter to Modify

Receive after Acknowledge:

`XXXX|XXXX (XX)` Low byte of requested data

Send on Command Strobe going high:

`0000|0000|0000|0111 (0007)` Select Link Value

Receive after Acknowledge:

`XXXX|XXXX (XX)` High byte of requested data

Fourth scan:

Send on Command Strobe going high:

`0000|XXXX|1110|0011 (0XE3)` Set actual value

Receive after Acknowledge:

`XXXX|XXXX (XX)` Low byte of requested data

Send on Command Strobe going high:

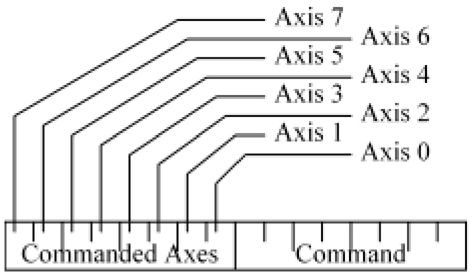
`0000|0001|1101|1000 (01D8)` Link Value (500)

Receive after Acknowledge:

`XXXX|XXXX (XX)` High byte of requested data

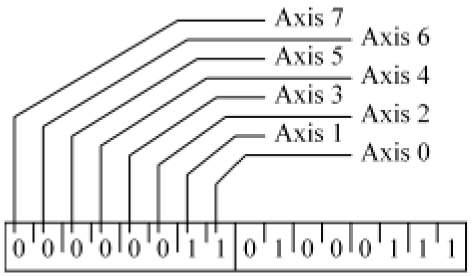
A.3.14 Command/Commanded Axes

In order to fit a single Event Step into eight words, the Command and Commanded Axes fields share a single word. The Commanded Axes field is stored in the most significant byte. If all eight bits are zero, the axis running the event sequence executes the command. Otherwise, the command is issued to each axis that has its corresponding bit set (bits 0-7). The Command is stored in the least significant byte:



Example

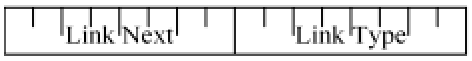
A Go (G) command needs to be issued to axes 0 and 1. Since the hexadecimal value for the Go command is 0x47 (01000111 binary), then the bit map for the Command/Commanded Axes field would be as follows:



Next convert this binary field to hexadecimal (0x0347) or decimal (839), and use it as the Command Value in either a Event Step Edit or Event Step Transfer command.

A.3.15 LINK TYPE/NEXT

In order to fit a single Event Step into eight words, the Link Type and Link Next fields share a single word. The Link Next field is stored in the most significant byte; the Link Type is stored in the least significant byte:

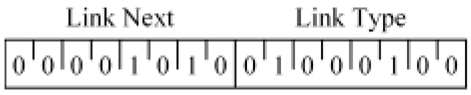


Available Link Types

For an overview of all available link types and a list of all available link types organized by use, see Link Types and Link Values.

Example

A DelayMS (D) Link Type needs to be used, which will link next to step 10. Since the hexadecimal value for the DelayMS (D) link type is 0x44 (01000100 binary) and the hexadecimal value for the link next value of 10 is 0x0A (00001010 binary), then the bit map for the Link Type/Next field would be as follows:



Next convert this binary field to hexadecimal (0x0A44) or decimal (2628), and use it as the Command Value in either a Event Step Edit or Event Step Transfer command.

A.3.16 Diagnostics Command

Format: 0000 XXXX 1111 XXXX

X Don’t care.

These commands are for testing the RMC under safe conditions. The behavior depends on whether PROFIBUS-DP or Communication Digital I/O is used.

For PROFIBUS-DP, both command and command data are echoed in the status and data in registers for the axis.

For Communication Digital I/O, the low byte of the command is returned on the acknowledge of the command, and the low byte of the command data is returned on the acknowledge of the command data.

Example for PROFIBUS-DP in Compact Mode with Sync:

If you sent out the following data in the Output registers:

CMND (HEX)

```
O+1 0000|0011|1111|0101 (03F5) Diagnostic command

 +2 0001|0010|0100|1000 (1248) Diagnostic data

O+3 0000|1100|1111|1010 (0CFA) Diagnostic command

 +4 1000|0100|0010|0001 (8421) Diagnostic data
```

After the Synchronization Output register is incremented, the RMC will process the commands and update the Synchronization Input register to match. At this time, the input registers would hold the following values:

```
I+1 0000|0011|1111|0101 (03F5) Same as Output

 +2 0001|0010|0100|1000 (1248) Same as Output

I+3 0000|1100|1111|1010 (0CFA) Same as Output
```

`+4 1000|0100|0010|0001 (8421)` Same as Output

Example for Digital I/O:

Suppose you want to test using the Digital I/O for sending data. You must send a 0x00F0 to trigger the diagnostic command, and you can then send any data the second scan. The following sequence is one possible way this would take place:

Send on Command Strobe going high:

`0000|0000|1111|0000 (00F0)` Diagnostic command

Receive after Acknowledge:

`1111|0000    (F0)` Low byte of diagnostic command

Send on Command Strobe going high:

`0000|0000|0000|1111 (000F)` Diagnostic data

Receive after Acknowledge:

`0000|1111    (0F)` Low byte of diagnostic data

A.3.17 Download Graph Data Commands

Format: GGGG GGNN NNNN NNNN

G Graph selection index (includes axis number and data type). 8 to 55.

N Time index. 0 to 1023.

Note: These commands are available only through RMC PROFIBUS modules using Compact Mode and require RMC100 CPU firmware dated 19980811 or newer. RMC DI/O modules cannot download graphs. Modbus Plus, Ethernet, and PROFIBUS in Message Mode can read the plots registers directly. See the appropriate RMC Register Map and Reading Plot Data from the Motion Controller for further details.

Download Graph Data commands allow the PROFIBUS-DP master to get the same graph data that is available to RMCWin for graphing and debugging. Each of these commands read a single word of graph data. The RMC starts its graph data acquisition when a Go, Move Relative, Open Loop, Sine Move, Follow Spline, or Start Graph command is issued. At that time the RMC starts to fill six 1024-word arrays at a rate determined by the Plot Time parameter. The master can retrieve the graph data one word per scan per axis until the graph data gets overwritten by the next graph. This will only happen when another graph is initiated using one of the commands above. If you wish to issue one of the commands above but don't want to overwrite the current graph for that axis, set the Graph Disable bit in the Mode word.

Note: You need not read the entire 1024-element graph. Therefore, if you only require 256 points from the graph, just read those 256 points.

Note: Because only a single word of graph data can be read by each axis each command cycle, reading a full graph can be quite slow. For higher performance, consider using the RMC PROFIBUS in Message Mode instead of Compact Mode. See Using the PROFIBUS-DP Message Mode for details.

Following is a list of the Graph Selection Index values. These values are stored in bits 10-15 of the command word. Each Graph Selection Index holds a block of 1024 registers of the same data type and for the same axis.

GSI	Command Range	Description
8	0x2000-0x23FF	Axis 0 Target Position/Pressure
9	0x2400-0x27FF	Axis 0 Actual Position/Pressure
10	0x2800-0x2BFF	Axis 0 Status Bits
11	0x2C00-0x2FFF	Axis 0 Drive Output
12	0x3000-0x33FF	Axis 0 Option Data A
13	0x3400-0x37FF	Axis 0 Option Data B
14	0x3800-0x3BFF	Axis 1 Target Position/Pressure
15	0x3C00-0x3FFF	Axis 1 Actual Position/Pressure
16	0x4000-0x43FF	Axis 1 Status Bits
17	0x4400-0x47FF	Axis 1 Drive Output
18	0x4800-0x4BFF	Axis 1 Option Data A
19	0x4C00-0x4FFF	Axis 1 Option Data B
20	0x5000-0x53FF	Axis 2 Target Position/Pressure
21	0x5400-0x57FF	Axis 2 Actual Position/Pressure
22	0x5800-0x5BFF	Axis 2 Status Bits
23	0x5C00-0x5FFF	Axis 2 Drive Output

24	0x6000-0x63FF	Axis 2 Option Data A
25	0x6400-0x67FF	Axis 2 Option Data B
26	0x6800-0x6BFF	Axis 3 Target Position/Pressure
27	0x6C00-0x6FFF	Axis 3 Actual Position/Pressure
28	0x7000-0x73FF	Axis 3 Status Bits
29	0x7400-0x77FF	Axis 3 Drive Output
30	0x7800-0x7BFF	Axis 3 Option Data A
31	0x7C00-0x7FFF	Axis 3 Option Data B
32	0x8000-0x83FF	Axis 4 Target Position/Pressure
33	0x8400-0x87FF	Axis 4 Actual Position/Pressure
34	0x8800-0x8BFF	Axis 4 Status Bits
35	0x8C00-0x8FFF	Axis 4 Drive Output
36	0x9000-0x93FF	Axis 4 Option Data A
37	0x9400-0x97FF	Axis 4 Option Data B
38	0x9800-0x9BFF	Axis 5 Target Position/Pressure
39	0x9C00-0x9FFF	Axis 5 Actual Position/Pressure
40	0xA000-0xA3FF	Axis 5 Status Bits
41	0xA400-0xA7FF	Axis 5 Drive Output
42	0xA800-0xABFF	Axis 5 Option Data A
43	0xAC00-0xAFFF	Axis 5 Option Data B
44	0xB000-	Axis 6 Target

	0xB3FF	Position/Pressure
45	0xB400-0xB7FF	Axis 6 Actual Position/Pressure
46	0xB800-0xBBFF	Axis 6 Status Bits
47	0xBC00-0xBFFF	Axis 6 Drive Output
48	0xC000-0xC3FF	Axis 6 Option Data A
49	0xC400-0xC7FF	Axis 6 Option Data B
50	0xC800-0xCBFF	Axis 7 Target Position/Pressure
51	0xCC00-0xCFFF	Axis 7 Actual Position/Pressure
52	0xD000-0xD3FF	Axis 7 Status Bits
53	0xD400-0xD7FF	Axis 7 Drive Output
54	0xD800-0xDBFF	Axis 7 Option Data A
55	0xDC00-0xDFFF	Axis 7 Option Data B

The data returned in the option data arrays (Option Data A and Option Data B) is selected by the plot options; see Selecting the Data to Plot for details. The data retrieved from each of these arrays is listed below:

- Extra Position Precision
  - o Option Data A: Fractional part of the target position.
  - o Option Data B: Fractional part of the actual position.
- o Command/Command Value
  - o Option Data A: Last command issued by the PROFIBUS master to this axis.
  - o Option Data B: Command value of the last command.
- o Current Event/Link Value
  - o Option Data A: Current event step number being executed.
  - o Option Data B: Link value of current wait condition.

- Raw Transducer Counter.
    - Option Data A: Low 16 bits of the transducer counter.
    - Option Data B: Transducer-type dependent; either upper bits and/or status bits.
- Internal Speeds.
    - Option Data A: The Target Speed computed internally in position units per second.
    - Option Data B: The Actual Speed computed internally in position units per second.
- Integral Drive.
    - Option Data A: The integral drive in drive count units. There are 8192 drive count units in 10000 mV. Therefore a value of 819 is equal to 1000 mV.
    - Option Data B: The fractional part of the integral drive. The fractional part is in parts in 65,536 of one drive count. This can usually be ignored except when determining the rate the integral is winding up or down.

Below are examples of two methods of downloading graph data. The first uses only one axis's PROFIBUS registers, which allows the other axes to be receiving different commands but is slow. The second uses multiple axes' PROFIBUS registers, which speeds up the graph download by receiving multiple graph data words per command cycle. In the list of steps below, PROFIBUS Input Registers are referred to as I+n, where n is the offset from the first input register. Similarly, PROFIBUS Output Register are referred to as O+n. See Input Register Overview and Output Register Overview for further details:

To download Axis 0's Target Position graph data using only axis 0:

1. Write 0x2000 hex to the Axis 0 Command Register (O+1).
2. If using sync registers, increment the Sync Out Register (O+0) and wait for Sync In (I+0) to match Sync Out (O+0).
3. The Axis 0 Data In Register (I+2) will hold a Axis 0's Target Position value from the graph. Store this value in your graph array.
4. Increment the Axis 0 Command Register (O+1) to the next time period.
5. Repeat steps 2 through 4 until all the desired data has been read.

To download Axis 0's Target Position graph data using axes 0-3:

1. Write 0x2000 hex to the Axis 0 Command Register (O+1).
2. Write 0x2001 hex to the Axis 1 Command Register (O+3).
3. Write 0x2002 hex to the Axis 2 Command Register (O+5).
4. Write 0x2003 hex to the Axis 3 Command Register (O+7).
5. If using sync registers, increment the Sync Out Register (O+0) and wait for Sync In (I+0) to match Sync Out (O+0).

6. The Axis 0 Data In Registers (I+2, I+4, I+6, I+8) will hold Axis 0's Target Position values from the graph. Store these values in your graph array.
7. Add four to each of the Axis 0 Command Registers (O+1, O+3, O+5, O+7) to request the next four time periods.
8. Repeat steps 5 through 7 until all the desired data has been read.

Tips

- Do not download more data than you need. For example, if only 256 points are required for a graph, don't download all 1024 points.

Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

A.3.18 Event Step Transfer Command

Format: 1110 DSSS SSSS SNNN

D Direction. Write (0) or read (1) event step.

S Step number. 0 to 255.

N Field in the event step. 0 to 7.

These commands are used to move event step information to and from the RMC. The D bit shown in the binary expansion of the Event Step Transfer Command register indicates whether the command will write an Event Step value (indicated by a 0) or read an Event Step value (indicated by a 1). Therefore, commands E000 to E7FF set (write) data to the RMC, while commands E800 to EFFF get (read) data from the RMC. The eight S bits specify the step number, and the three N bits represent the field in the Event Step to access. Use the following chart to select the desired field:

N BITS VALUE	STEP FIELD
000	MODE
001	ACCEL
010	DECEL
011	SPEED
100	COMMAND VALUE
101	COMMAND/COMMANDED AXES
110	LINK TYPE/NEXT
111	LINK VALUE

To write event step information to the RMC, use the following commands to set the corresponding fields. The command data value will be copied into the corresponding step field:

Command Register	**Data Register**
E000	Step 0 MODE
E001	Step 0 ACCELERATION
E002	Step 0 DECELERATION
E003	Step 0 SPEED
E004	Step 0 COMMAND VALUE
E005	Step 0 COMMAND/COMMANDED AXES
E006	Step 0 LINK TYPE/NEXT
E007	Step 0 LINK VALUE
E008	Step 1 MODE
E009	Step 1 ACCELERATION
...	
E7FD	Step 255 COMMAND/COMMANDED AXES
E7FE	Step 255 LINK TYPE/NEXT
E7FF	Step 255 LINK VALUE

When using PROFIBUS-DP, you can update multiple words every Programmable Controller scan by using the Command and Data registers for multiple axes. For example, by using one axis to transfer the first half of the table and the second axis to transfer the second half of the table, transfer time can be cut in half.

To read event step information from the RMC, use the following commands to retrieve the corresponding pieces of data from the step table. The data will be available where the Status Area Request is normally returned:

Command Register	**Data In Register**
E800	Step 0 MODE
E801	Step 0 ACCELERATION
E802	Step 0 DECELERATION
E803	Step 0 SPEED

E804	Step 0 COMMAND VALUE
E805	Step 0 COMMAND/COMMANDED AXES
E806	Step 0 LINK TYPE/NEXT
E807	Step 0 LINK VALUE
E808	Step 1 MODE
E809	Step 1 ACCELERATION
...	
EFFD	Step 255 COMMAND/COMMANDED AXES
EFFE	Step 255 LINK TYPE/NEXT
EFFF	Step 255 LINK VALUE

Appendix B: Command Field Reference

B.1 Position Command Fields

B.1.1 MODE (Non-Pressure/Force)

Default: 0x0000

The individual bits of the Mode word determine the way the RMC responds to control commands and parameters. Bit 0 is the LSB and bit 15 is the MSB.

Note: It is highly recommended that the Pop-up Editors in RMCWin be used for editing this and other hexadecimal fields.

Click here for the Mode Bit Map

Bit 15 - Graph Disable
When this bit is set for an axis, the controller will not start logging data for a new graph but it will continue logging data for a previous command if the plot time hasn't expired. This is useful for troubleshooting long sequences of moves. The Graph Disable bit can also be set when an error occurs so that the following commands will not restart the data logging and write over the previous graph data that shows the error.

This bit is used by the following commands:

- Go (G)
- Move Relative (J)
- Sine Move (~)
- Follow Spline (f)
- Follow Spline Relative
- Open Loop (O)
- Set Bias Drive (B)
- Set Pressure (^) (Pressure/Force Control axes only)

Bit 14 - Gear Type Bit
This bit is only used if the Gear Mode bit (13) is set. It determines whether this axis will be on the master axis's Target or Actual Position. If this bit is clear, the slave is geared based on the Target

Position. If this bit is set, the slave is geared based on the Actual Position. For details on gearing, see Gearing Axes.

This mode bit can apply to the following commands:

- Go (G)
- Move Relative (J)
- Sine Move (~)
- Follow Spline (f)

Bit 13 - Gear Mode Bit

Note: This feature is only available in RMC100 CPU firmware dated 19990625 or newer (and beta firmware dating 19980827B or newer).

This bit makes the move act in Gearing mode. When this bit is set, the Gear Type (bit 14) and Gear Master Select (bits 4-6) bits are also used. For details on gearing, see Gearing Axes.

This mode bit can apply to the following commands:

- Go (G)
- Move Relative (J)
- Sine Move (~)
- Follow Spline (f)
- Follow Spline Relative

Bit 11 - Velocity Loop Bit

Note: This feature is only available in RMC100 CPU firmware dated 20030515 or newer.

This bit is used only when the Rotational bit is set in a Go (G) command (a speed control move). If this bit is cleared, the speed control will be closed loop based on position. If this bit is set, the speed control will be closed loop based on velocity. See Speed Control for more details.

Bit 10 - Superimposed Bit

Note: This feature is only available in the special ’r;SI’ RMC100 CPU firmware (e.g. 19980827SI).

This bit may be applied to the Follow Spline (f), Follow Spline Relative and Sine Move (~) commands. When this bit is set for one of these commands, the move will be superimposed on top of one of the following types of moves: Spline (started by the Follow Spline (f) command), Sine Move (started by the Sine Move (~) command), Geared move (started by a Go (G) command with the Gear Mode bit set), and Speed Control move (started by a Go (G) command with the Rotational bit set). There may never be more than two moves in progress on an axis at one time.

If this bit is cleared for the Follow Spline (f) and Sine Move (~) commands, the newly requested move is executed without being superimposed.

Bit 9 - Rotational Bit

This bit needs to be set for most applications where the axis will be rotating multiple times. Setting this bit causes the following:

1. The Extend and Retract Limits do not limit the motion. Instead they are used to control the wrap of the position: when the position exceeds the extend limit, the difference between the two limits is subtracted from the position, and likewise when the position exceeds the retract limit, the difference is added to the position. Generally, these limits will be set to indicate one rotation (such as 0 and 36,000 for hundredths of a degree).

2. The Go, Set Speed (Unsigned), and Set Speed (Signed) act in Speed Control mode, as described in the Speed Control topic.

Bit 8 - Monitor Pressure

Only axes which have pressure axes assigned to them use this bit. If this bit is set on such an axis, the pressure will be monitored to see if it reaches the minimum pressure threshold at which point the axis will begin regulating pressure. For details on using this bit to regulate pressure, refer to Controlling Pressure or Force.

Bit 7 - S-Curve

When this bit is set the RMC calculates a s-shaped target, resulting in smoother motion with more gradual starts and stops and higher peak speeds. In a s-curve move, the maximum acceleration is 1.53 times the straight-line acceleration (used by a trapezoidal motion profile). If another move command is given before the s-curve is finished, there may be a discontinuity (jerk) at the transition.

This mode bit can apply to the following commands:

- Go (G)
- Move Relative (J)

Bit 4-6 - Gear Master Select (When Gear Mode bit is set)

When the Gear Mode bit (13) is set, these bits are used together to select the gear master for this axis. Refer to Gearing Axes for details on gearing. We highly recommend that you use the Pop-up Editor for the Mode word to edit this field rather than enter the value directly. However, if this is necessary, the following table shows how to select a master axis using these bits:

Master Axis	**Bit #**	**6**	**5**	**4**
Axis 0		0	0	0
Axis 1		0	0	1
Axis 2		0	1	0
Axis 3		0	1	1
Axis 4		1	0	0
Axis 5		1	0	1
Axis 6		1	1	0
Axis 7		1	1	1

These mode bits can apply to the following commands:

- Go (G)
- Move Relative (J)
- Sine Move (~)
- Follow Spline (f)
- Follow Spline Relative

Bit 6 - Quick Mode (When Gear Mode bit is not set)

When this bit is set, a move will ramp the drive up in Open Loop Mode to the value specified (in millivolts) in the Speed field and maintain it there until it reaches the deceleration point. It will then ramp down to the requested position (specified by the Command Value) in Closed Loop Mode . The Acceleration and Deceleration fields specify the ramp based on Mode bits 0 and 1.

For this mode to work properly the Extend Feed Forward and Retract Feed Forward parameters should be adjusted correctly. They are use to calculate the drive given to the axis at the moment ramp down begins.

If the speed achieved by the axis during the open loop portion of the move is greater than 65,535, when the ramp down starts, the target speed will be limited to 65,535.

This mode bit can apply to the following commands:

- Go (G)
- Move Relative (J)

Why Bother?

If the application requires quick positioning without the need to synchronize or gear with other axes, use the Quick Mode. With Quick Mode, the Drive output can be saturated for maximum speed without worrying about the PID's integrator winding up. When the axis transitions from Open Loop Mode to Closed Loop Mode the Target Speed is set to the Actual Speed, the Target Position is set to the Actual Position and the integrator is set to zero.

Bits 4-5 - Sync Group A or B (When Gear Mode bit is not set)

These two bits are used for axis synchronization. If the Gear Mode bit is cleared, these bits select which synchronization group this axis belongs to; see Synchronizing Axes for details.

Therefore, two independent groups of axes can be synchronized using these two bits. At no time should both of these bits be set at the same time.

A side benefit is that if an error occurs that stops an axis with a sync bit on, other axes with the same sync bit set will also stop.

These mode bits can apply to the following commands:

- Go (G)
- Move Relative (J)

- Halt

Bits 2-3 - Integrator Mode Select

These two bits define four integrator modes:

Bit #	3	2
Mode 0 - Integrator always active - Default	0	0
Mode 1 - Integrator active only during deceleration and in position	0	1
Mode 2 - Integrator active only during in position	1	0
Mode 3 - Integrator never active	1	1

Why Bother?

The integrator can be disabled so that it doesn't wind up. This can be handy in applications where the Actual Position can't reach the Target Position. Even-ending or clamping applications should use these bits.

Bits 0-1 - Acceleration and Deceleration Mode Select

These two bits define four acceleration/deceleration modes:

Bit #	1	0
Mode 0 - Ramp Rate (Default)	0	0
Mode 1 - Ramp Rate * 1000	0	1
Mode 2 - Distance to specified speed	1	0
Mode 3 - Time to specified speed	1	1

In Mode 0, the Acceleration and Deceleration parameters define the ramp rate in position units per second per second.

In Mode 1, the Acceleration and Deceleration parameters define the ramp rate in 1000 position units per second per second. For example, with position units of 0.001 inches, an Acceleration of 50 will cause the axis to accelerate at 50 inches per second per second.

In Mode 2, the parameters define the distance in position units to accelerate or decelerate to the specified Speed.

In Mode 3, the parameters define the time in milliseconds to accelerate or decelerate to the specified Speed.

Refer to the appropriate section in Gearing or Open Loop for details on the use of these bits in these modes.

B.1.2 Mode (Non-Pressure/Force) Bit Map

The axis Mode word contains 16 bits of information. The hexadecimal table below provides an easy way to convert hexadecimal numbers to bit patterns.

```
                        F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                        E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
  Hexadecimal To        D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
  Binary Conversion     C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
  Table                 B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                        A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                        9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                        8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                        7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                        6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                        5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                        4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                        3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                        2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                        1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                        0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0

                           _______   _______   _______   _______
                          | | | | | | | | | | | | | | | | | | | |
                          |1|1|1|1| |1|1| | | | | | | | | | | | |
                          |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
  Bit Definition          |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
 ----------------          / / / /   / / / /   / / / /   / / / /
 Graph Disable ----- 15 -/ / / /   / / / /   / / / /   / / / /
 Gear Type --------- 14 --/ / /   / / / /   / / / /   / / / /
 Gear Mode --------- 13 ---/ /   / / / /   / / / /   / / / /
 Reserved ---------- 12 ----/   / / / /   / / / /   / / / /
                               / / / /   / / / /   / / / /
 Velocity Loop ----- 11 ------/ / / /   / / / /   / / / /
 Superimposed ------ 10 -------/ / /   / / / /   / / / /
 Rotational -------- 09 --------/ /   / / / /   / / / /
 Monitor Pressure -- 08 ---------/   / / / /   / / / /
                                    / / / /   / / / /
 S-Curve ----------- 07 -----------/ / / /   / / / /
 Quick Mode/Gear Ms- 06 ------------/ / /   / / / /
 Sync Grp B/Gear Ms- 05 -------------/ /   / / / /
 Sync Grp A/Gear Ms- 04 --------------/   / / / /
                                         / / / /
 Integrator Mode --- 03 ----------------/ / / /
 Integrator Mode --- 02 -----------------/ / /
 Accel/Decel Mode -- 01 ------------------/ /
 Accel/Decel Mode -- 00 -------------------/
```

B.1.3 Acceleration

Default: 1000

Range: 0 to 65535

This parameter defines the acceleration ramp rate used by the axis for a move. It has four meanings depending on the Acceleration and Deceleration Mode bits in the Mode word.

In Mode 0, this parameter is interpreted as an acceleration rate and is expressed in Position Units/sec/sec. If Scale is set so one Position Unit equals 0.001", then a value of 200 would represent an acceleration rate of 0.200 inches/sec/sec. In Mode 0, the relationship between the ramp distance, Acceleration and Speed is:

$$\text{Ramp Distance} = \frac{\text{Speed x Speed}}{\text{Acceleration x 2}}$$

$$\text{Acceleration} = \frac{\text{Speed x Speed}}{\text{Ramp Distance x 2}}$$

In Mode 1, this field is an acceleration rate and is expressed in 1000 position units/sec/sec. If the Scale is set so one position unit equal 0.001”, then a value of 200 would represent an acceleration rate of 200 inches/second/second. In Mode 1, the relationship between the ramp distance, acceleration and speed is:

$$\text{Ramp Distance} = \frac{\text{Speed x Speed}}{\text{Acceleration x 2000}}$$

$$\text{Acceleration} = \frac{\text{Speed x Speed}}{\text{Ramp Distance x 2000}}$$

In Mode 2, this parameter defines the distance (in position units) it will take to ramp to the specified Speed.

In Mode 3, it defines the time (in milliseconds) it will take to ramp to the specified Speed.

B.1.4 Deceleration

Default: 1000
Range: 0 to 65535

This parameter is the same as Acceleration except it specifies the deceleration ramp length or deceleration rate.

B.1.5 Speed

Default: 1000
Range: 0 to 65535

The Speed parameter sets the constant speed to be achieved after acceleration. Speed is expressed in position units/second. If the Scale is set so one position unit equals 0.001", a speed of 25 inches per second is expressed as 25000.

Note: When using Acceleration/Deceleration Mode 2, changing the Speed without changing the Acceleration and Deceleration distances will change the acceleration and deceleration rates.

Note: If the Speed is set to zero, the axis will not move.

B.1.6 Command Value

Range: Depends on Command

The Command Value field is multipurpose. It is often used with the 'G' command to specify the position where the axis is to move. When used this way, the value is checked to be within the Extend Limit and Retract Limit and is then used as the Command Position value.

Here is the complete list of commands that use this field:

For this command:	**Command Value is:**	**Command Value Range:**
'$' (Display LCD Screen)	Screen Number	0 to 15
'+' (Add)	Destination Address	80 to 2303
'-' (Subtract)	Destination Address	80 to 2303
' ' ' (MulDiv)	Destination Address	80 to 2303

',' (Function)	Destination Step	0 to 255
'?' (Poll)	Extended Link Value	Depends on usage
'@' (Arm Home Input)	Home position	Valid 16-bit Position
'A' (Change Accel)	Acceleration value	0 to 65,535
’r;a’ (Amp Enable/Disable)	Enable/Disable	0=disable, 1=enable
’r;B’ (Set Bias Drive)	Millivolts of Drive	-10000 to 10000
'D' (Change DECEL)	Deceleration value	0 to 65,535
'E' (Start Events)	Event number	0 to 255
'G' (Position Mode)	Requested position	Valid 16-bit Position
'G' (Rotational Mode)	Requested direction	0 or -1
'G' (Geared Mode)	Requested numerator	-32,768 to 32,767
'I' (Set Integral Drive)	Integral drive value	-10,000 to 10,000
'J' (Go Relative)	Relative position	-32,768 to 32,767
'L' (Limit Drive)	Drive Limit in millivolts	-10,000 to 10,000
'l' (Set Extended Link Value)	Extended Link Value	Depends on usage
'M' (Set Mode)	Mode	0 to 0xFFFF
'N' (Set Null Drive)	New null value	-2000 to 2000
'O' (Open Loop Output)	Requested millivolts of drive	-12,000 to 12,000
’r;q’ (Quadrature Reset)	New position	Valid 16-bit Position
’r;t’ (Teach Step)	Step number to teach	0 to 255
'V' (Set Speed-Unsigned)	Speed	0 to 65,535
'v' (Set Speed-Signed)	Speed	-32,768 to 32,767
'W' (Reference)	Filter Time Constant in ms	0 to 65,535
'Z' (Set Target Position)	New target position	Valid 16-bit Position
'z' (Offset Positions)	Target position offset	-32,768 to 32,767
’r;[’r; (Set output bits)	Bit mask of outputs to set	0 to 0xFFFF
’r;]’ (Reset output bits)	Bit mask of outputs to reset	0 to 0xFFFF

'r;{ 'r; (Simulate rising edge)	Input to simulate	0 to 15
'r;}' (Simulate falling edge)	Input to simulate	0 to 15
'r;~' (Sine move)	Relative position	-32,768 to 32,767
'r;|' (Set Pressure Set A)	Pressure	Any Valid Pressure
'r;\' (Set Ramp Time)	Milliseconds	0 to 65535
'r;^' (Set Pressure)	Pressure	Any Valid Pressure
'r;_' (Set Pressure Set B)	Pressure	Any Valid Pressure
0x7F (Map Output to Axis Position)	Position to Map Output	0-2, 10-12, …, 70-72
0xD0,0xD5-0xDF (Set Parameter On-the-Fly)	New value of parameter	Parameter-dependent

Make sure that the data in the Command Value field is correct before you issue any of these commands.

B.1.7 COMMAND

This field represents the command to be sent to the RMC. You may enter a command by either typing the letter or symbol of the command in a command field, or by entering the ASCII number in either decimal or hexadecimal in the field. The charts below lists all available commands with their ASCII values. The first chart shows commands that can be issued to position, speed, or velocity control axes:

Description	**ASCII**	**Decimal**	**Hex**
Set and Reset Wait Bits	None	23	0x17
I-PD Position Move	!	33	0x21
Set Count Offset	#	35	0x23
Display LCD Screen	$	36	0x24
MulDiv	'	39	0x27
Add	+	43	0x2B
Function	,	44	0x2C
Subtract	-	45	0x2D
Sine Move Continuous	None	48	0x30
Poll	?	63	0x3F

Arm Home Input	@	64	0x40
Change Acceleration	A	65	0x41
Amp Enable/Disable	a	97	0x61
Clear Spline Segments	C	67	0x43
Set Position/Pressure	c	99	0x63
Change Deceleration	D	68	0x44
Start Events	E	69	0x45
Set Feed Forward	F	70	0x46
Follow Spline Segment	f	102	0x66
Go	G or g	71 or 103	0x47 or 0x67
Halt	H or h	72 or 104	0x48 or 0x68
Set Integral Drive	I	73	0x49
Set Integral Drive to Null Drive	i	105	0x69
Relative Move	J or j	74 or 106	0x4A or 0x6A
Disable Drive Output	K	75	0x4B
Limit Drive	L	76	0x4C
Set Extended Link Value	l	108	0x6C
Set Mode	M	77	0x4D
Set Null Drive	N	78	0x4E
Set Null Drive to Integral Drive	n	110	0x6E
Open Loop	O	79	0x4F
Set Parameters	P or p	80 or 112	0x50 or 0x70
Quit Events	Q	81	0x51

Reset Position	q	113	0x71
Restore Null Drive	R	82	0x52
Restore Integral Drive	r	114	0x72
Save Null Drive	S	83	0x53
Save Integral Drive	s	115	0x73
Set Spline Interval	T	84	0x54
Teach Step	t	116	0x74
Update Flash	U	85	0x55
Update Flash Segment Command	u	117	0x75
Set Speed (Unsigned)	V	86	0x56
Set Speed (Signed)	v	118	0x76
Reference	W	87	0x57
New Spline Point	X or x	88 or 120	0x58 or 0x78
Start a Graph	y	121	0x79
Zero/Set Target Position	Z	90	0x5A
Offset Positions	z	122	0x7A
Set Outputs	[	91	0x5B
Reset Outputs	]	93	0x5D
Simulate Rising Edge	{	123	0x7B
Simulate Falling Edge	}	125	0x7D
Sine Move	~	126	0x7E
Follow Spline Relative Command	None	6	0x06
Map Output to Axis Position	None	127	0x7F
Move Relative to An Axis	None	192-207	0xC0-0xCF
Set Parameter On-the-Fly	None	208-223	0xD0-

0xDF

This chart shows commands that can be issued to pressure axes:

Description	ASCII	Decimal	Hex
Set Pressure Set A	|	124	0x7C
Set Bias Drive	B	66	0x42
Start Events	E	69	0x45
Set Mode	M	77	0x4D
Open Loop	O	79	0x4F
Set Parameters	P	80	0x50
Quit Events	Q	81	0x51
Set Pressure Ramp Time	\	92	0x5C
Set Pressure	^	94	0x5E
Set Pressure Set B	_	95	0x5F
Set Parameter on-the-fly	None	208-223	0xD0-0xDF

B.2 Pressure/Force Command Fields

B.2.1 Mode (Pressure/Force)

Default: 0x0000

Bits in the Mode word determine the way the RMC regulates pressure. Bit 15 is the MSB and bit 0 is the LSB.

Click here for the Mode Bit Map

Bit 5 - Ramp Mode Type
If this bit is set, the pressure ramps linearly. If this bit is cleared, the pressure ramp is an S-curve. It is recommended that this bit be left cleared, as it is much smoother and easier for the hydraulic system to track curved ramps. Notice that the Ramp Time field is used differently for the two modes when entering pressure. In S-curve mode the entering pressure ramp is only a half ramp (the deceleration), so only half the Ramp Time is used for the half ramp. Linear mode ramps have no acceleration/deceleration phases, so the Ramp Time field always refers to the entire ramp and is therefore not divided by two.

Bit 4 - Ramp Time Type

This bit affects the value used for the Ramp Time when the axis begins regulating pressure. If this bit is not set, then the Ramp Time field is used. This is always the case when ramping between pressures rather than entering pressure. If this bit is set, then the Ramp Time field is ignored upon entering pressure, and instead the RMC calculates the ramp time required to have a continuous pressure curve.

Bit 3 - Integrator Mode Select

This bit defines two integrator modes:

	Bit 3
Mode 0 - Integrator always active	0
Mode 1 - Integrator active only while at pressure	1

Bit 0-1 - Pressure Mode Select

These two bits define four pressure modes; only one is currently available:

	Bit 1	**Bit 0**
Mode 0 - Pressure Set Mode	0	0
Mode 1 - Reserved	0	1
Mode 2 - Reserved	1	0
Mode 3 - Reserved	1	1

B.2.2 Mode (Pressure/Force) Bit Map

The axis Mode word contains 16 bits of information. The hexadecimal table below provides an easy way to convert hexadecimal numbers to bit patterns.

```
                      F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                      E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
 Hexadecimal To       D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
 Binary Conversion    C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
 Table                B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                      A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                      9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                      8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                      7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                      6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                      5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                      4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                      3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                      2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                      1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                      0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0

                         _______   _______   _______   _______
                        | | | | | | | | | | | | | | | | | | | |
                        |1|1|1|1| |1|1| | | | | | | | | | | | |
                        |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
 Bit Definition         |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
----------------         / / / /   / / / /   / / / /   / / / /
Reserved ---------- 15 -/ / / /   / / / /   / / / /   / / / /
Reserved ---------- 14 --/ / /   / / / /   / / / /   / / / /
Reserved ---------- 13 ---/ /   / / / /   / / / /   / / / /
Reserved ---------- 12 ----/   / / / /   / / / /   / / / /
                              / / / /   / / / /   / / / /
Reserved ---------- 11 ------/ / / /   / / / /   / / / /
Reserved ---------- 10 -------/ / /   / / / /   / / / /
Reserved ---------- 09 --------/ /   / / / /   / / / /
Reserved ---------- 08 ---------/   / / / /   / / / /
                                   / / / /   / / / /
Reserved ---------- 07 -----------/ / / /   / / / /
Reserved ---------- 06 ------------/ / /   / / / /
Ramp Mode Type ---- 05 -------------/ /   / / / /
Ramp Time Type ---- 04 --------------/   / / / /
                                        / / / /
Integrator Mode --- 03 ----------------/ / / /
Reserved ---------- 02 -----------------/ / /
Pressure Mode ----- 01 ------------------/ /
Pressure Mode ----- 00 -------------------/
```

B.2.3 Pressure Set A

Default: 0
Range: Valid Pressure Units

Pressure Set A is used by Pressure Set Mode as the pressure above which the axis will begin regulating pressure. When a command is issued to the position axis with the Pressure Mode bit set, then the pressure is monitored. When the Actual Pressure becomes equal to or greater than

Pressure Set A, the axis begins regulating pressure. There are two ways to set the this field:

- Issue a Set Pressure (^) command, and the second command parameter will be the new value for Pressure Set A.
- Issue a Set Pressure Set A (|) command, and the Command Value will be the new value for Pressure Set A.

Note: Although Pressure Set A is used as the threshold above which an axis can enter pressure regulating mode, it is not used as the threshold below which the axis leaves pressure regulating mode; the Pressure Set B parameter is used for this.

Note: Pressure Set A must be set greater than or equal to Pressure Set B.

B.2.4 Pressure Set B

Default: 0
Range: Valid Pressure Units

Pressure Set B is used by Pressure Set Mode as the pressure below which the axis will stop regulating pressure. When the Actual Pressure becomes equal to or less than Pressure Set B, the axis stops regulating pressure and returns to position mode. There are two ways to set this field:

- Issue a Set Pressure (^) command, and the third command parameter will be the new value for Pressure Set B.
- Issue a Set Pressure Set B (_) command, and the Command Value will be the new value for Pressure Set B.

Note: Although Pressure Set B is used as the threshold below which an axis will leave pressure regulating mode, it is not used as the threshold above which the axis enters pressure regulating mode; the Pressure Set A parameter is used for this.

Note: Pressure Set A must be set greater than or equal to Pressure Set B.

B.2.5 Ramp Time (Pressure/Force)

Default: 0
Range: 0 to 65535

The Ramp Time is used to designate the amount of time it will take to ramp to a pressure. There are two types of pressure ramps:

- One type is a ramp from one pressure to another while the axis is already regulating pressure. In this case, the pressure will begin at rest at one pressure and move to another. This will involve accelerating the pressure and then decelerating down to the final pressure. The time to get from one pressure to the other will take the number of milliseconds given in Ramp Time.

- The second type of ramp occurs when the axis enters pressure mode. In this case the pressure is generally not at rest at the beginning. Therefore, for s-curve ramps, only the deceleration half of the ramp is used and to ramp to a halt at the Command Pressure will take one half the number of milliseconds given in the Ramp Time field. However, the whole Ramp Time will be used in linear mode. For details on selecting s-curve versus linear ramps see the Ramp Mode Type bit description.

There are two ways to set this field:

- Issue a Set Pressure (^) command, and the fourth command parameter will be the new value for Ramp Time.
- Issue a Set Pressure Ramp Time (\) command, and the Command Value will be the new value for Ramp Time.

Note: In the case where the Pressure Set A field is above the Command Pressure, the pressure needs to be ramped in the reverse direction. In this case, the pressure is first accelerated and then decelerated. The Ramp Time gives the entire length of this ramp. Do not divide this time by two.

Note: Setting this field to zero will cause the Target Pressure to instantaneously jump to the Command Pressure.

B.2.6 Command Value

Range: Depends on Command

The Command Value field is multipurpose. It is often used with the 'G' command to specify the position where the axis is to move. When used this way, the value is checked to be within the Extend Limit and Retract Limit and is then used as the Command Position value.

Here is the complete list of commands that use this field:

For this command:	**Command Value is:**	**Command Value Range:**
'$' (Display LCD Screen)	Screen Number	0 to 15
'+' (Add)	Destination Address	80 to 2303
'-' (Subtract)	Destination Address	80 to 2303
' ' ' (MulDiv)	Destination Address	80 to 2303
',' (Function)	Destination Step	0 to 255
'?' (Poll)	Extended Link Value	Depends on usage
'@' (Arm Home Input)	Home position	Valid 16-bit Position

'A' (Change Accel)	Acceleration value	0 to 65,535
'r;a' (Amp Enable/Disable)	Enable/Disable	0=disable, 1=enable
'r;B' (Set Bias Drive)	Millivolts of Drive	-10000 to 10000
'D' (Change DECEL)	Deceleration value	0 to 65,535
'E' (Start Events)	Event number	0 to 255
'G' (Position Mode)	Requested position	Valid 16-bit Position
'G' (Rotational Mode)	Requested direction	0 or -1
'G' (Geared Mode)	Requested numerator	-32,768 to 32,767
'I' (Set Integral Drive)	Integral drive value	-10,000 to 10,000
'J' (Go Relative)	Relative position	-32,768 to 32,767
'L' (Limit Drive)	Drive Limit in millivolts	-10,000 to 10,000
'l' (Set Extended Link Value)	Extended Link Value	Depends on usage
'M' (Set Mode)	Mode	0 to 0xFFFF
'N' (Set Null Drive)	New null value	-2000 to 2000
'O' (Open Loop Output)	Requested millivolts of drive	-12,000 to 12,000
'r;q' (Quadrature Reset)	New position	Valid 16-bit Position
'r;t' (Teach Step)	Step number to teach	0 to 255
'V' (Set Speed-Unsigned)	Speed	0 to 65,535
'v' (Set Speed-Signed)	Speed	-32,768 to 32,767
'W' (Reference)	Filter Time Constant in ms	0 to 65,535
'Z' (Set Target Position)	New target position	Valid 16-bit Position
'z' (Offset Positions)	Target position offset	-32,768 to 32,767
'r;['r; (Set output bits)	Bit mask of outputs to set	0 to 0xFFFF
'r;]' (Reset output bits)	Bit mask of outputs to reset	0 to 0xFFFF
'r;{ 'r; (Simulate rising edge)	Input to simulate	0 to 15
'r;}' (Simulate falling edge)	Input to simulate	0 to 15
'r;~' (Sine move)	Relative position	-32,768 to 32,767

’r;|’ (Set Pressure Set A)	Pressure	Any Valid Pressure
’r;\’ (Set Ramp Time)	Milliseconds	0 to 65535
’r;^’ (Set Pressure)	Pressure	Any Valid Pressure
’r;_’ (Set Pressure Set B)	Pressure	Any Valid Pressure
0x7F (Map Output to Axis Position)	Position to Map Output	0-2, 10-12, …, 70-72
0xD0,0xD5-0xDF (Set Parameter On-the-Fly)	New value of parameter	Parameter-dependent

Make sure that the data in the Command Value field is correct before you issue any of these commands.

B.2.7 COMMAND

This field represents the command to be sent to the RMC. You may enter a command by either typing the letter or symbol of the command in a command field, or by entering the ASCII number in either decimal or hexadecimal in the field. The charts below lists all available commands with their ASCII values. The first chart shows commands that can be issued to position, speed, or velocity control axes:

Description	**ASCII**	**Decimal**	**Hex**
Set and Reset Wait Bits	None	23	0x17
I-PD Position Move	!	33	0x21
Set Count Offset	#	35	0x23
Display LCD Screen	$	36	0x24
MulDiv	'	39	0x27
Add	+	43	0x2B
Function	,	44	0x2C
Subtract	-	45	0x2D
Sine Move Continuous	None	48	0x30
Poll	?	63	0x3F
Arm Home Input	@	64	0x40
Change Acceleration	A	65	0x41
Amp Enable/Disable	a	97	0x61

Clear Spline Segments	C	67	0x43
Set Position/Pressure	c	99	0x63
Change Deceleration	D	68	0x44
Start Events	E	69	0x45
Set Feed Forward	F	70	0x46
Follow Spline Segment	f	102	0x66
Go	G or g	71 or 103	0x47 or 0x67
Halt	H or h	72 or 104	0x48 or 0x68
Set Integral Drive	I	73	0x49
Set Integral Drive to Null Drive	i	105	0x69
Relative Move	J or j	74 or 106	0x4A or 0x6A
Disable Drive Output	K	75	0x4B
Limit Drive	L	76	0x4C
Set Extended Link Value	l	108	0x6C
Set Mode	M	77	0x4D
Set Null Drive	N	78	0x4E
Set Null Drive to Integral Drive	n	110	0x6E
Open Loop	O	79	0x4F
Set Parameters	P or p	80 or 112	0x50 or 0x70
Quit Events	Q	81	0x51
Reset Position	q	113	0x71
Restore Null Drive	R	82	0x52
Restore Integral Drive	r	114	0x72

Save Null Drive	S	83	0x53
Save Integral Drive	s	115	0x73
Set Spline Interval	T	84	0x54
Teach Step	t	116	0x74
Update Flash	U	85	0x55
Update Flash Segment Command	u	117	0x75
Set Speed (Unsigned)	V	86	0x56
Set Speed (Signed)	v	118	0x76
Reference	W	87	0x57
New Spline Point	X or x	88 or 120	0x58 or 0x78
Start a Graph	y	121	0x79
Zero/Set Target Position	Z	90	0x5A
Offset Positions	z	122	0x7A
Set Outputs	[	91	0x5B
Reset Outputs	]	93	0x5D
Simulate Rising Edge	{	123	0x7B
Simulate Falling Edge	}	125	0x7D
Sine Move	~	126	0x7E
Follow Spline Relative Command	None	6	0x06
Map Output to Axis Position	None	127	0x7F
Move Relative to An Axis	None	192-207	0xC0-0xCF
Set Parameter On-the-Fly	None	208-223	0xD0-0xDF

This chart shows commands that can be issued to pressure axes:

Description	**ASCII**	**Decimal**	**Hex**
Set Pressure Set A	|	124	0x7C
Set Bias Drive	B	66	0x42
Start Events	E	69	0x45
Set Mode	M	77	0x4D
Open Loop	O	79	0x4F
Set Parameters	P	80	0x50
Quit Events	Q	81	0x51
Set Pressure Ramp Time	\	92	0x5C
Set Pressure	^	94	0x5E
Set Pressure Set B	_	95	0x5F
Set Parameter on-the-fly	None	208-223	0xD0-0xDF

Appendix C: Parameter Field Reference

C.1 MDT, SSI, Analog, Resolver Position Parameters

C.1.1 Configuration Word

Default: 0x0000

This 16-bit word controls the configuration of the module. Bit 0 is the LSB; bit 15 is the MSB.

Click here for the Config Word Bit Map

Bits 7, 12-15 - Transducer Type bits
These bits are used differently depending on the transducer type used. Refer to one of the following topics for details on these four bits:

- Analog Transducer bits
- Magnetostrictive Displacement Transducer (MDT) bits
- Quadrature/Stepper Transducer bits
- Synchronous Serial Interface (SSI) bits
- Resolver Transducer Bits

Bit 11 - Pressure Assign bit
This bit is used only if the RMC has one or more auxiliary pressure or force channels. If such channels are installed, then this bit can be set to indicate that this position axis has an assigned pressure axis. Bits 8 and 9 are used to define the pressure axis assigned.

Bit 10 - Auto Home Re-arm
This bit is for Quadrature axes only. When it is set, the axis will automatically be re-armed after homing. This requires the axis to first be armed using the Arm Home Command. If this bit is cleared after homing, the axis will still be armed. See the Homing a Quadrature Axis topic for details.

Bits 8 - 9 - Pressure Axis Select bits
These bits are used only if the RMC has one or more auxiliary pressure or force channels and bit 11 is set to indicate that this axis has one of the pressure or force channels assigned to it. These bits then select between up to four auxiliary pressure/force channels:

Bit 9	**Bit 8**	
0	0	The first auxiliary pressure/force axis is

		assigned.
0	1	The second auxiliary pressure/force axis is assigned.
1	0	The third auxiliary pressure/force axis is assigned.
1	1	The fourth auxiliary pressure/force axis is assigned.

It is necessary to assign a pressure axis to a position axis in order to switch between position and pressure control. Refer to Using an Analog Channel as a Pressure Axis for details.

Note: Two position axes cannot be assigned to the same pressure axis. If this does occur, only the first axis to make the assignment followed by the Set Parameters (P) command will succeed in being assigned. Any later attempt to assign the pressure axis will fail with a parameter error.

Bit 6 - Reverse Drive Mode bit

Warning: This bit must be set properly when the Set Parameters command is issued.

This bit determines the relationship between the RMC's position direction and the drive output direction.

For analog-output axes, setting this bit reverses the polarity of the axis output voltage. Although in many cases the differential drive wires can be swapped on the drive unit, this is not always possible or it may simply be easier to use this bit.

Bit 6	
0	The RMC will generate positive drive to increase transducer counts.
1	The RMC will generate negative drive to increase transducer counts.

For stepper-output axes, this bit is used to change the meaning of the Direction output with relation to the axis transducer counts.

Bit 6	
0	The RMC will use a low Direction output to increase transducer counts.
1	The RMC will use a high Direction output to increase transducer counts.

Bit 5 - Absolute Mode bit

This mode is intended for use with a two-valve system, one controlling the flow rate and the other the direction. This axis controls the flow-rate valve. The directional valve must be controlled by other means. In this mode, the axis generates a positive drive output regardless of the direction of the move. The drive will not go negative if the motion controller overshoots the target. This is useful for some injection and blow-molding applications.

When the axis is stopped, you must go into open loop mode.

Warning: This bit must be set properly when the Set Parameters command is issued.

Bit 4 - Continue Mode bit

This bit affects what happens when the module loses contact with the Programmable Controller. When this bit is set, the module will finish any move it has started, otherwise it will halt immediately upon detecting loss of contact with the PLC. This is useful for finishing a move that must complete to prevent machine downtime (for example, a partial shot in an injection-molding machine).

Bit 3 - Simulate bit

When this bit is set, the drive output is set to zero and the magnetostrictive transducer inputs are ignored. Internally the Target Position is used as the Actual Position. This mode is used for debugging. (The transducer error bits and LEDS will be cleared.)

Note: Drive output is always 0 volts while in simulate mode; no commands, including open loop, will change this.

Bits 1 - 2 - Prescale Divisor Bits

Prescaling may be desired on systems with long strokes. The prescale bits are used along with the Scale parameter to convert Transducer Counts to Actual Position. The effect is to divide the Scale by 1, 2, 4, or 8. The divide-by factor is specified as follows:

Counts ÷	**Bit 2**	**Bit 1**
1	0	0
2	0	1
4	1	0
8	1	1

The advantage of these bits is that they let you maintain a higher resolution in the Scale parameter. For example, a Scale of 31027 and a divide-by of 4 gives an effective scale of 7756.75. Without using the divide-by bits, the Scale would have to be rounded up to 7757, which has less resolution—with a 64" rod, the rounded value would make a 0.002" error at 64". Note: The stroke is always limited to 220 inches due to the 18-bit transducer counter resolution.

These bits must be zero for Resolver modules.

Bit 0 - Integrator Limit

When this bit is cleared, the integrator limit is 20% of full drive. When the bit is set, the limit is 80%. If the integrator tries to go above the limit, the Integrator Windup bit in the status word is set and the integrator value is held at the limit. The integrator limit is designed to prevent the drive output from saturating.

Note: The actual drive output may be reduced based on the values of the Extend Feed Forward and Retract Feed Forward .

C.1.2 Configuration Word Bit Map

The axis Configuration word contains 16 bits of information. The hexadecimal table below provides an easy way to convert hexadecimal numbers to bit patterns.

```
                         F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                         E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
  Hexadecimal To         D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
  Binary Conversion      C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
  Table                  B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                         A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                         9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                         8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                         7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                         6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                         5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                         4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                         3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                         2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                         1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                         0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0
                            _______   _______   _______   _______
                           | | | | | | | | | | | | | | | | | | | |
                           |1|1|1|1| |1|1| | | | | | | | | | | | |
                           |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
  Bit Definition           |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
 ----------------           / / / /   / / / /   / / / /   / / / /
 Transducer Type --- 15 -/ / / /   / / / /   / / / /   / / / /
 Transducer Type --- 14 --/ / /   / / / /   / / / /   / / / /
 Transducer Type --- 13 ---/ /   / / / /   / / / /   / / / /
 Transducer Type --- 12 ----/   / / / /   / / / /   / / / /
                               / / / /   / / / /   / / / /
 Pressure Assign --- 11 ------/ / / /   / / / /   / / / /
 Reserved ---------- 10 -------/ / /   / / / /   / / / /
 Pressure Axis Sel - 09 --------/ /   / / / /   / / / /
 Pressure Axis Sel - 08 ---------/   / / / /   / / / /
                                    / / / /   / / / /
 Transducer Type --- 07 -----------/ / / /   / / / /
 Reverse Drive ----- 06 ------------/ / /   / / / /
 Absolute Mode ----- 05 -------------/ /   / / / /
 Continue Mode ----- 04 --------------/   / / / /
                                         / / / /
 Simulate Mode ----- 03 ----------------/ / / /
 Prescale Bits ----- 02 -----------------/ / /
 Prescale Bits ----- 01 ------------------/ /
 Integrator Limit -- 00 -------------------/
```

C.1.3 Configuration Bits - MDT Specific

Transducer Type - Bits 12-15

15	14	13	12	Transducer Type
0	0	0	0	Start/Stop (S-S)
x	x	x	x	Pulse-width Modulated (PWM)

Setting any of these four bits to non-zero (as described below in the Recirculation Count section) will result in the transducer being treated as pulse-width modulated.

Recirculation Count - Bits 12-15

If the transducer is pulse-width modulated (PWM or gated), these bits select number of recirculations. The following values are available:

15	14	13	12	Recirculations
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

When a transducer is set for multiple recirculations, there is a small time delay between those recirculations. During this time the counter on the motion controller continues to count, resulting in "extra" counts which are added to the Transducer Counts. These extra counts use some of the 65535 maximum counts that are available, reducing the maximum stroke the module can control to 65535 minus the "extra" counts. To correct for this effect the module calculates the "extra" counts based on the number of transducer recirculations and subtracts them from the transducer count. If the value of these bits and the transducer recirculations agree, the transducer count will be slightly positive when the transducer is at its minimum position.

In some cases it may be desirable to set bits 12-15 to a value greater than the number of recirculations to subtract additional counts, but if the value selected is too large it is possible to get a position underflow condition. If this happens, the Position Overflow status bit will be set and the Actual Position will be equal to the Offset parameter.

Rising-/Falling-Edge Selection - Bit 7

Use this bit to select the edge of a start/stop transducer's waveform to be used. Notice that this bit is ignored if a PWM transducer is selected as described above. Most transducers use rising edges, but some transducers manufactured by Balluff are known to use falling edges.

Bit 7	Transducer Edge
0	Rising (default)
1	Falling

C.1.4 Configuration Bits - SSI Specific

Unused - Bits 7, 12-15

These bits are reserved for future use and should always be set to 0 on SSI axes. Refer to SSI Configuration for details on configuring the RMC to use your SSI device.

C.1.5 Configuration Bits - Analog Specific

Analog Input Type - Bits 12-14

Use the following table to select the appropriate input range:

14	13	12	Analog Input Type
0	0	0	Voltage: 0V to +10V
0	0	1	Voltage: -10V to +10V
0	1	0	Voltage: 0V to +5V
0	1	1	Voltage: -5V to +5V
1	0	0	Current: 4mA to 20mA

For a description of the corresponding transducer count ranges, refer to the Counts topic.

Unused - Bits 7 and 15

These bits are reserved for future use and should be cleared on analog axes.

C.1.6 Configuration Bits - Resolver Specific

Bits 12 - Resolver Resolution

This bit selects the resolution of the resolver feedback.

0 = 14-bit resolution

1 = 16-bit resolution

The counts from the resolver interface will always be a 16-bit number regardless of the resolution selected. With 14-bit resolution, bits 0 and 1 in the counts (the least significant bits) will always be zero.

Using 14 bits allows operation at higher speeds and accelerations than with 16 bits. See the Resolver Specifications for details.

C.1.7 Scale

Default: Transducer specific
Range: -32768 to 32767

This scale is used on MDT, SSI, Analog position axes with the Offset parameter and the Prescale Divisor bits of the Config word to convert the Transducer Counts to an Actual Position. Quadrature axes use this parameter with the Prescale Divisor bits to convert from the change in Transducer Counts to a change in the Actual Position. Resolver axes use this parameter together with the Count Offset to calculate the Actual Position.

For details on converting counts to position units see the appropriate topics for your transducer types:

- Analog Scaling
- MDT Scaling
- Quadrature Scaling
- SSI Scaling
- Stepper Scaling
- Resolver Scaling

Note: When the Scale is set to 0, it is treated as though the Scale is set to 32768. This allows a one-to-one relationship between counts and position units. Similarly, a Scale value of -32768 represents a negative one-to-one relationship between counts and position units (increasing counts mean decreasing position units).

Note: You should never set the Scale below 16383 (except for resolver axes). If it does go below that value, then use the Prescale Divisor bits in the Configuration Word. For details, refer to the discussion on scaling for your transducer type.

Note: In RMC100 CPU firmware prior to 19991216, Scale could not be 0 or -32768 for MDT, SSI, and analog transducers.

C.1.8 Offset

Default: 0
Range: -65536 to 65535

This parameter is available on all axis types except those with quadrature or stepper feedback. Quadrature and stepper axes use the Coordinate Limit parameter in place of the Offset parameter.

The Offset parameter is used for the following two purposes:

- Translating transducer counts to actual position in user-defined position units.
- Defining the 16-bit position range.

For a complete discussion on the use of this parameter, select the topics that apply to the transducer types you use:

- Analog Scaling
- MDT Scaling
- SSI Scaling

What if it is displayed under RMCWin incorrectly?
The Offset parameter may be displayed incorrectly on the RMCWin main screen in some circumstances. This is because the offset specifies the zero location of the axis. The numbers are not necessarily incorrect (the module still functions correctly), but they may not look right.

To correct the problem:

1. Right-click the Offset parameter and choose **Toggle Offset Sign**.
2. You can now save the parameters to a file (on the File menu, click Save). Whenever you start RMCWin, if you read the parameters from the RMC, the problem should be corrected, as long as the Offset did not change in the RMC.

C.1.9 Extend Limit

Default: Current position on power-up
Range: Valid 16-bit Position

The Extend Limit specifies the maximum requested position value that the RMC will allow as a Command Value. (When the Scale is negative, this is the minimum value.) A Command Value that exceeds this value will be set to the Extend Limit, and the Parameter Error bit in the Status word will be set. The Extend Limit is given in Position Units.

Note: The Extend Limit must be changed when the Scale or Offset parameters are changed. Extending is the direction that gives increasing Transducer Counts.

Note: On startup, the Extend Limit defaults to the current position of the axis, so new Extend and Retract Limits must be issued followed by a 'r;P' command before the axis will move, unless different limits have been saved in the Flash.

C.1.10 Retract Limit

Default: Current position on power-up
Range: Valid 16-bit Position

The Retract Limit specifies the minimum value the motion controller will allow as a position Command Value. (When the Scale is negative, this is the maximum value.) A Command Value below this value will be set to the Retract Limit and will cause the parameter error bit in the Status word to be set. The Retract Limit is given in Position Units.

Note: The Retract Limit must be changed when the Scale or Offset parameters are changed.

Note: On startup, the Retract Limit defaults to the current position of the axis, so new Extend and Retract Limits must be issued followed by a 'r;P' command before the axis will move, unless different limits have been saved in the Flash.

C.1.11 Proportional Gain

Default: 1
Range: 0 to 65535

The Proportional Gain controls how much drive is generated proportional to the Position Error. The Position Error is defined as the Target Position minus the Actual Position. The units on the Proportional Gain is millivolts per 10 units of Position Error.

The Proportional Drive is defined as follows:

$$\text{ProDrive (mV)} = \text{ProGain (mV/10pu)} \times \text{PosError (pu)}$$

Or more simply:

$$\text{ProDrive (mV)} = \frac{\text{ProGain (mV/pu)} \times \text{PosError (pu)}}{10}$$

where pu is position units.

Note: The actual drive output may be reduced based on the values of the Extend Feed Forward and Retract Feed Forward .

CAUTION: Increase the Proportional Gain gradually. Excessive gain can cause oscillation that could cause damage or injury.

Think about this:

Internally, the motion controller must compare the error between the Target and Actual Positions with error limits to keep values from overflowing. The error limit is the error at which full drive (10 volts) will occur. This internal error limit is calculated as follows:

Error Limit = 100,000 / Proportional Gain

Therefore, if the Proportional Gain is set to 100, the Error Limit will be 1,000, which means any error greater than 1,000 will be treated as an error of 1,000 and the Overdrive error bit will be set in the Status Word.

C.1.12 Integral Gain

Default: 1
Range: 0 to 65535

The Integral Gain is used to control the amount of drive provided by the integrator. The integrator adds the position error to an accumulator every millisecond. The Integral Gain should be adjusted after the feed forwards have been set to optimal values. Using the integrator before the feed forwards have been set properly will cause the system to overshoot the target. We recommend that you set the Integral Gain to a value of at least 50.

Integral Gain is defined as:

Integral Gain = millivolts per 10240 counts of accumulated Position Error

Integral Drive is defined as:

Integral Drive = Integral Gain x Accumulated Counts / 10240

Note: The actual drive output may be reduced based on the values of the Extend Feed Forward and Retract Feed Forward .

Why Bother?
Integral Gain should be used to compensate for the fact that loads may vary, valves are non-linear and the axis may have trouble getting to the Command Position without Integral Gain.

C.1.13 Differential Gain

Default: 0
Range: 0 to 65535

The Differential Gain field is used to apply a gain based on the rate of change between the target and actual positions. There are two ways to view this.

First, this is a gain multiplied by the current rate of change in the position error. The differential drive, in millivolts is computed as follows:

$$\text{Differential Drive (mV)} = \frac{\text{ErrorChangeRate} \times K_d}{10}$$

$$\text{ErrorChangeRate(pos-units/ms)} = \frac{(E_0 - E_1)}{\text{LoopTime}}$$

where:

K_d=	Differential Gain in mV/[pos-units/ms]
E_0=	position error this control loop in position units
E_1=	position error last control loop in position units
LoopTime =	RMC Control Loop Time in ms (1 or 2)

A second equivalent way of viewing this parameter is as the gain multiplied by the velocity error. When looked at from this angle, the above equations become the following:

$$\text{Differential Drive (mV)} = \frac{(\text{TarVel} - \text{ActVel}) \times K_d}{10240}$$

where:

K_d=	Differential Gain in mV/[pos-units/s]
TarVel =	target velocity in pos-units/s
ActVel =	actual velocity in pos-units/s

Note: The actual drive output may be reduced based on the values of the Extend Feed Forward and Retract Feed Forward.

CAUTION: To avoid oscillation during initial tuning start with values below 10.

Why Bother?
The Differential Gain field will usually be set to zero. Noisy or low-resolution feedback will limit the amount the Differential Gain can be increased.

C.1.14 Extend Feed Forward

Default: 100
Range: 0 to 65535

TIP: After the axis has made a complete move without oscillations or overdrive errors, use the 'F' command to automatically set the Feed Forward value.

Feed Forward is an open loop compensation that is proportional to the Target Speed of the axis. This value is expressed in terms of millivolts per 1,000 Position Units per second. Extend Feed Forward drive is added to the output only when the axis is extending. The drive output provided by the Extend Feed Forward is determined as follows:

$$\text{Feed Forward Drive} = \frac{\text{Extend Feed Forward} \times \text{Target Speed}}{1000}$$

You can find the appropriate value for Extend Feed forward by making a move with the axis using a Speed of 1,000. The amount of output drive required to maintain this speed should be used as the Extend Feed Forward parameter. If the axis lags after the parameter has been set, the feed forward is too small or the system response is too slow. If the axis leads, the feed forward is too large or the system response is too slow. The 'F' command used after an extend move will automatically adjust the Extend Feed Forward parameter. The Extend and Retract Feed Forwards are also known as velocity feed forwards.

Note:
Feed Forward Drive is limited to a maximum value of 10,000. Inserting that value in the equation above and solving for the Feed Forward term gives the follow relationship:

Feed Forward <= (10,000 * 1,000) / Target speed

That is, the larger the Speed the smaller the maximum Feed Forward value.

For example, at 30 in/sec Speed, the maximum Feed Forward is:

Max Feed Forward = (10,000 * 1,000)/30,000 = 10,000/30 = 333

If you set Speed to 30,000 and enter a Feed Forward value larger than 333, the value will be reduced to 333.

Think about this:
If the feed forward value is less than 300, an axis going at a speed of 65535 will not get a full valve drive from the feed forward term alone. If the feed forward value is less than 200, the axis drive system is too big and the system may have trouble controlling position. If this happens, then reduce the gain on the amplifier or use a smaller valve or reduce pressure until the feed forward terms are over 300.

On hydraulic systems, the extend and retract feed forward terms will be different by the ratio of the extend and retract surface areas of the position.

The Proportional Gain, Integral Gain and Differential Gain as well as the Integral Drive Limit are adjusted internally based on the ratio of the Extend and Retract Feed Forwards. The direction with the lower Feed Forward value will have effectively lower Gains and Integral Limit. This is done to compensate for the different system dynamics in the two directions.

For example, for a system with an Extend Feed forward of 100 and a Retract Feed Forward of 200 the effective gains in the extend direction will be one half (100 / 200) as big as the specified gains.

C.1.15 Retract Feed Forward

Default: 100
Range: 0 to 65535

Same as Extend Feed Forward, except that it is used when retracting.

Note: Retracting is the direction that returns decreasing Transducer Counts.

C.1.16 Extend Acceleration Feed Forward

Default: 0
Range: 0 to 65535

The Extend Acceleration Feed Forward causes the controller to give extra drive while accelerating and a braking drive while decelerating (negative acceleration) in the extend direction. The drive due to Acceleration Feed Forward is calculated as follows:

$$\text{Accel FF Drive} = \frac{\text{Extend Accel FF} \times 1\text{mV}}{100{,}000 \text{ position units/sec/sec}}$$

The Acceleration Feed Forward provides a second order approximation (prediction) of how much drive is required to move.

If the position unit is 0.001 inches, this equals millivolts of drive per 100 inches per second per second.

Why Bother?
The acceleration feed forward helps the axis track better while accelerating or decelerating. The acceleration feed forwards should not be adjusted until the velocity feed forwards are adjusted correctly.

C.1.17 Retract Acceleration Feed Forward

Default: 0
Range: 0 to 65535

Same as Extend Acceleration Feed Forward, except it is used when retracting.

C.1.18 Dead Band Eliminator

Default: 0

Range: 0 to 2000

Some valves and drives do not react to small changes in output around the null drive value; this effect is termed "dead band". The Dead Band Eliminator helps alleviate this problem by applying extra drive.

The RMC100 has two different Deadband eliminator algorithms:

- **Proportional (recommended)**
  The number of millivolts specified in the Dead Band Eliminator value is added to or subtracted from the drive output (depending on the direction of travel) when the Actual Position is outside of the In Position window. This causes the drive to be outside of the deadband. When the Actual Position is within the In Position window, the drive is applied proportional to the distance from the actual position. The Integral gain is always held at its current value. This algorithm is the recommended algorithm.
- **Immediate (RMC100 default)**
  The number of millivolts specified in the Dead Band Eliminator value is always added to or subtracted from the drive output (depending on the direction of travel) so the drive output is outside the dead band.

Note: In RMC100 CPU firmware prior to 20030916, the Dead Band Eliminator always used the immediate algorithm.

If a value outside the valid range is entered, the parameter error bit will be set, and the value will be set to zero.

CAUTION: Do not make this value too large or the drive will oscillate.

Selecting a Deadband Algorithm

To select a Deadband algorithm follow these steps:

1. In RMCWin, on the Tools menu, click Module Configuration.
2. In the Slots list, click the CPU.
3. Click Slot Options. The RMC100/101 CPU Options dialog box will be displayed.
4. Click the Deadband tab.
5. Select an algorithm.
6. Click Update RMC.
7. The Update Module Configuration dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module yourself.

C.1.19 In Position

Default: 50
Range: 0 to 65535

This parameter specifies the size of a window around the Command Position. When the Actual Position gets within this window, the In Position bit is set in the Status word. Notice that the In Position bit is not latched and therefore could go off again if the axis moves back outside the In Position window.

Example:

If an axis Command Position is 10,000 and the In Position parameter is 30, the In Position bit will be set when the axis is stopped and its Actual Position is between 9,971 and 10,029. The bit will be cleared whenever the Actual Position is outside the range.

C.1.20 Following Error

Default: 250
Range: 0 to 65535

The Following Error determines how large the difference between the Target Position and Actual Position can get before the Following Error bit is set in the Status word.

C.1.21 Auto Stop

Default: 0x1FE0 (Soft Stops enabled, Hard Stops enabled for Transducer Errors)
Click here for the Auto Stop Bit Map

The Auto Stop parameter controls the action taken on the rising edge of any of the axis' error bits (the eight (8) most-significant bits in the Status word listed below). The user has control over which error bits cause which levels of stop, or whether an error will cause a stop at all. The default setting of all the AutoStops is Hard Stop, as described below.

During startup and tuning, you will typically need to set some AutoStops to Status Only to keep halts from interfering with the tuning. After completeing the startup procedure, make sure to set the AutoStops to setting that are safe for the machine operation.

Auto Stops are always active unless the axis is in Open Loop, in which case no Auto Stops will be triggered even though error bits may be set.

Note: AutoStops will not occur if the axis is in Open Loop!

The eight (8) most-significant bits in the Status word, as listed below, are the error bits, with the exception of the Home Input status bit for Quadrature axes with Analog (QUAD) or Stepper (STEP) output:

Fault	QUAD	STEP	All Others
7	Encoder Error/Fault	Encoder Error/Fault	No Transducer
6	Extend Limit	Extend Limit	Transducer Noise
5	Retract Limit	Retract Limit	Transducer Overflow
4	Overdrive	Overdrive	Overdrive
3	Parameter Error	Parameter Error	Parameter Error
2	Home Input	Home Input	Pos./Press. Overflow
1	Integrator Windup	Compensation Timeout	Integrator Windup
0	Following Error	Following Error	Following Error

The available actions for each fault are listed below:

- **Status Only**

The fault will be reflected in its corresponding status bit in the Status word, but no further action will be taken. For some transducer types, this option may not be available for the following faults: No Transducer, Transducer Overflow, and Transducer Noise.

- **Soft Stop**
  If the axis is in closed loop, the fault will trigger ramping the axis to a stop. The speed will ramp down to zero using the current Deceleration value. If the axis is in Open Loop, the drive will not be affected. If there is an event sequence on the axis, it will stop executing. The Halt status bit will also be set in the Status word. Some faults will use the open loop ramp down although the axis was in closed loop: Position Overflow, No Transducer, Transducer Overflow, Transducer Noise, and Encoder Error/Fault Input. This is done because the position feedback is not dependable and closed loop control cannot be maintained.
- **Hard Stop**
  If the axis is in closed loop, the fault will trigger the drive output to go immediately to 0 mV on analog outputs and no steps for stepper outputs, and the axis will be placed in open loop mode. If the axis is in Open Loop, the drive will not be affected. If there is an event sequence on the axis, it will stop executing. The Halt and Open Loop status bit will also be set in the Status word.
- **Disable Drive**
  The fault will be handled as in a Hard Stop, but additionally the Amp Enable output on QUAD and STEP will be opened. Use the Amp Enable (a) command to close the output again. This option is only available for faults on QUAD and STEP axes.

To view or change the Auto Stop parameter value, open the Auto Stop popup editor in RMCWin using one of the methods described in Using Popup Editors. Select the desired action for each fault type and click OK. Changes to this parameter do not take effect until you issue a Set Parameters (P) command.

Manual Parameter Entry

This parameter can also be edited manually, but this is discouraged since it is much easier to use the popup editor. You can enter hexadecimal numbers by typing a leading 0x, as in 0x1FE0. Each of the eight faults listed above has two bits assigned to it, labeled S and H in the table below:

Bit	Description	Bit	Description
15	Fault 7 - Bit S	7	Fault 7 - Bit H
14	Fault 6 - Bit S	6	Fault 6 - Bit H
13	Fault 5 - Bit S	5	Fault 5 - Bit H
12	Fault 4 - Bit S	4	Fault 4 - Bit H
11	Fault 3 - Bit S	3	Fault 3 - Bit H
10	Fault 2 - Bit S	2	Fault 2 - Bit H
9	Fault 1 - Bit S	1	Fault 1 - Bit H
8	Fault 0 - Bit S	0	Fault 0 - Bit H

For each fault, the two bits in the Auto Stop parameter define the action as follows:

Bit H	Bit S	Description

0	0	Status Only
0	1	Soft Stop
1	0	Hard Stop
1	1	Disable Drive

If you select Status Only for a fault that cannot use that action, then the axis will use a Soft Stop action for that fault. Similarly, if you select Disable Drive for a fault on an axis that does not have an Amp Enable output, the axis will use the Hard Stop action for the fault.

C.1.22 Auto Stop Bit Map

The table below provides an easy method to convert bit patterns to hexadecimal numbers.

```
                         F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                         E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
    Hexadecimal To       D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
    Binary Conversion    C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
    Table                B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                         A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                         9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                         8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                         7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                         6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                         5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                         4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                         3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                         2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                         1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                         0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0
                             _______   _______   _______   _______
                           | | | | | | | | | | | | | | | | | | | |
                           |1|1|1|1| |1|1| | | | | | | | | | | | |
                           |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
 Bit Definition            |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
----------------            / / / /   / / / /   / / / /   / / / /
S  No Xdcr/Encdr Flt - 15 -/ / / /   / / / /   / / / /   / / / /
O  Xdcr Nse/Ext Lmt -- 14 --/ / /   / / / /   / / / /   / / / /
F  Xdcr Ovr/Ret Lmt -- 13 ---/ /   / / / /   / / / /   / / / /
T  Overdrive  -------- 12 ----/   / / / /   / / / /   / / / /
                                 / / / /   / / / /   / / / /
S  Parameter Error --- 11 ------/ / / /   / / / /   / / / /
T  Pos Ovr/Home Inpt - 10 -------/ / /   / / / /   / / / /
O  Integrator Windup - 09 --------/ /   / / / /   / / / /
P  Following Error --- 08 ---------/   / / / /   / / / /
=========================             / / / /   / / / /
H  No Xdcr/Encdr Flt - 07 -----------/ / / /   / / / /
A  Xdcr Nse/Ext Lmt -- 06 ------------/ / /   / / / /
R  Xdcr Ovr/Ret Lmt -- 05 -------------/ /   / / / /
D  Overdrive --------- 04 --------------/   / / / /
                                           / / / /
S  Parameter Error --- 03 ----------------/ / / /
T  Pos Ovr/Home Inpt - 02 -----------------/ / /
O  Integrator Windup - 01 ------------------/ /
P  Following Error --- 00 -------------------/
```

If both Soft Stop and Hard Stop bits are set for a particular error condition, a Hard Stop will be executed and the Amp Enable output will be opened on QUAD and STEP axes.

C.2 Quadrature with Analog Output Parameters

C.2.1 Configuration Word

Default: 0x0000

This 16-bit word controls the configuration of the module. Bit 0 is the LSB; bit 15 is the MSB.

Click here for the Config Word Bit Map

Bits 7, 12-15 - Transducer Type bits
These bits are used differently depending on the transducer type used. Refer to one of the following topics for details on these four bits:

- Analog Transducer bits
- Magnetostrictive Displacement Transducer (MDT) bits
- Quadrature/Stepper Transducer bits
- Synchronous Serial Interface (SSI) bits
- Resolver Transducer Bits

Bit 11 - Pressure Assign bit
This bit is used only if the RMC has one or more auxiliary pressure or force channels. If such channels are installed, then this bit can be set to indicate that this position axis has an assigned pressure axis. Bits 8 and 9 are used to define the pressure axis assigned.

Bit 10 - Auto Home Re-arm
This bit is for Quadrature axes only. When it is set, the axis will automatically be re-armed after homing. This requires the axis to first be armed using the Arm Home Command. If this bit is cleared after homing, the axis will still be armed. See the Homing a Quadrature Axis topic for details.

Bits 8 - 9 - Pressure Axis Select bits
These bits are used only if the RMC has one or more auxiliary pressure or force channels and bit 11 is set to indicate that this axis has one of the pressure or force channels assigned to it. These bits then select between up to four auxiliary pressure/force channels:

Bit 9	**Bit 8**	
0	0	The first auxiliary pressure/force axis is assigned.
0	1	The second auxiliary pressure/force axis is assigned.
1	0	The third auxiliary pressure/force axis is assigned.

1	1	The fourth auxiliary pressure/force axis is assigned.

It is necessary to assign a pressure axis to a position axis in order to switch between position and pressure control. Refer to Using an Analog Channel as a Pressure Axis for details.

Note: Two position axes cannot be assigned to the same pressure axis. If this does occur, only the first axis to make the assignment followed by the Set Parameters (P) command will succeed in being assigned. Any later attempt to assign the pressure axis will fail with a parameter error.

Bit 6 - Reverse Drive Mode bit

Warning: This bit must be set properly when the Set Parameters command is issued.

This bit determines the relationship between the RMC's position direction and the drive output direction.

For analog-output axes, setting this bit reverses the polarity of the axis output voltage. Although in many cases the differential drive wires can be swapped on the drive unit, this is not always possible or it may simply be easier to use this bit.

Bit 6	
0	The RMC will generate positive drive to increase transducer counts.
1	The RMC will generate negative drive to increase transducer counts.

For stepper-output axes, this bit is used to change the meaning of the Direction output with relation to the axis transducer counts.

Bit 6	
0	The RMC will use a low Direction output to increase transducer counts.
1	The RMC will use a high Direction output to increase transducer counts.

Bit 5 - Absolute Mode bit

This mode is intended for use with a two-valve system, one controlling the flow rate and the other the direction. This axis controls the flow-rate valve. The directional valve must be controlled by other means. In this mode, the axis generates a positive drive output regardless of the direction of the move. The drive will not go negative if the motion controller overshoots the target. This is useful for some injection and blow-molding applications.

When the axis is stopped, you must go into open loop mode.

Warning: This bit must be set properly when the Set Parameters command is issued.

Bit 4 - Continue Mode bit

This bit affects what happens when the module loses contact with the Programmable Controller. When this bit is set, the module will finish any move it has started, otherwise it will halt immediately upon detecting loss of contact with the PLC. This is useful for finishing a move that must complete to prevent machine downtime (for example, a partial shot in an injection-molding machine).

Bit 3 - Simulate bit

When this bit is set, the drive output is set to zero and the magnetostrictive transducer inputs are ignored. Internally the Target Position is used as the Actual Position. This mode is used for debugging. (The transducer error bits and LEDS will be cleared.)

Note: Drive output is always 0 volts while in simulate mode; no commands, including open loop, will change this.

Bits 1 - 2 - Prescale Divisor Bits

Prescaling may be desired on systems with long strokes. The prescale bits are used along with the Scale parameter to convert Transducer Counts to Actual Position. The effect is to divide the Scale by 1, 2, 4, or 8. The divide-by factor is specified as follows:

Counts ÷	Bit 2	Bit 1
1	0	0
2	0	1
4	1	0
8	1	1

The advantage of these bits is that they let you maintain a higher resolution in the Scale parameter. For example, a Scale of 31027 and a divide-by of 4 gives an effective scale of 7756.75. Without using the divide-by bits, the Scale would have to be rounded up to 7757, which has less resolution—with a 64" rod, the rounded value would make a 0.002" error at 64". Note: The stroke is always limited to 220 inches due to the 18-bit transducer counter resolution.

These bits must be zero for Resolver modules.

Bit 0 - Integrator Limit

When this bit is cleared, the integrator limit is 20% of full drive. When the bit is set, the limit is 80%. If the integrator tries to go above the limit, the Integrator Windup bit in the status word is set and the integrator value is held at the limit. The integrator limit is designed to prevent the drive output from saturating.

Note: The actual drive output may be reduced based on the values of the Extend Feed Forward and Retract Feed Forward .

C.2.2 Configuration Word Bit Map

The axis Configuration word contains 16 bits of information. The hexadecimal table below provides an easy way to convert hexadecimal numbers to bit patterns.

```
                      F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                      E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
 Hexadecimal To       D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
 Binary Conversion    C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
 Table                B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                      A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                      9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                      8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                      7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                      6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                      5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                      4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                      3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                      2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                      1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                      0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0

                         _______   _______   _______   _______
                        | | | | | | | | | | | | | | | | | | | |
                        |1|1|1|1| |1|1| | | | | | | | | | | | |
                        |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
 Bit Definition         |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
----------------         / / / /   / / / /   / / / /   / / / /
Transducer Type --- 15 -/ / / /   / / / /   / / / /   / / / /
Transducer Type --- 14 --/ / /   / / / /   / / / /   / / / /
Transducer Type --- 13 ---/ /   / / / /   / / / /   / / / /
Transducer Type --- 12 ----/   / / / /   / / / /   / / / /
                              / / / /   / / / /   / / / /
Pressure Assign --- 11 ------/ / / /   / / / /   / / / /
Reserved ---------- 10 -------/ / /   / / / /   / / / /
Pressure Axis Sel - 09 --------/ /   / / / /   / / / /
Pressure Axis Sel - 08 ---------/   / / / /   / / / /
                                   / / / /   / / / /
Transducer Type --- 07 -----------/ / / /   / / / /
Reverse Drive ----- 06 ------------/ / /   / / / /
Absolute Mode ----- 05 -------------/ /   / / / /
Continue Mode ----- 04 --------------/   / / / /
                                        / / / /
Simulate Mode ----- 03 ----------------/ / / /
Prescale Bits ----- 02 -----------------/ / /
Prescale Bits ----- 01 ------------------/ /
Integrator Limit -- 00 -------------------/
```

C.2.3 Configuration Bits - Quadrature/Stepper Specific

For quadrature axes, these bits are used to select the active states of four inputs and to define the use of the Home Input status bit in the Status word.

Fault Input Active State - Bit 15

This bit determines the active state of the Fault Input. The active state of this input indicates that a fault has occurred. The Encoder Error status bit is set when the Fault input is active.

Bit 15	Active Input State
0	No current applied
1	Current Applied

Limit Inputs Active State - Bit 14

Determines the active state of the Extend (CW) and Retract (CCW) Limit Inputs. The active state of these inputs indicates that a limit has been reached. The Extend Limit status bit will be set when the Extend (CW) Limit Input is active, and the Retract Limit status bit will be set when the Retract (CCW) Limit Input is active:

Bit 14	Active Input State
0	No current applied
1	Current applied

Index (Z) Input Active State - Bit 13

Determines the active state of the Index (Z) Input:

Bit 13	Active Input State
0	Positive signal applied
1	Negative signal applied

Home (H) Input Active State - Bit 12

Determines the active state of the Home (H) Input:

Bit 12	Active Input State
0	Current applied
1	No current applied

Home Input Status Bit Level/Edge Select - Bit 7

This bit determines the use of the Home Input status bit, as shown below:

Bit 7	Home Input Status Bit Behavior
0	Latched Home Edge The Home Input status bit is set and latched on the transition from both the index (Z) and home (H) inputs being active to one or both being inactive.

1	Level of H Input The Home Input status bit is set when the Home (H) input is active. It is not latched.

For details on using this bit, see Homing a Quadrature Axis.

C.2.4 Scale

Default: Transducer specific
Range: -32768 to 32767

This scale is used on MDT, SSI, Analog position axes with the Offset parameter and the Prescale Divisor bits of the Config word to convert the Transducer Counts to an Actual Position. Quadrature axes use this parameter with the Prescale Divisor bits to convert from the change in Transducer Counts to a change in the Actual Position. Resolver axes use this parameter together with the Count Offset to calculate the Actual Position.

For details on converting counts to position units see the appropriate topics for your transducer types:

- Analog Scaling
- MDT Scaling
- Quadrature Scaling
- SSI Scaling
- Stepper Scaling
- Resolver Scaling

Note: When the Scale is set to 0, it is treated as though the Scale is set to 32768. This allows a one-to-one relationship between counts and position units. Similarly, a Scale value of -32768 represents a negative one-to-one relationship between counts and position units (increasing counts mean decreasing position units).

Note: You should never set the Scale below 16383 (except for resolver axes). If it does go below that value, then use the Prescale Divisor bits in the Configuration Word. For details, refer to the discussion on scaling for your transducer type.

Note: In RMC100 CPU firmware prior to 19991216, Scale could not be 0 or -32768 for MDT, SSI, and analog transducers.

C.2.5 Coord. Limit

Default: 0
Range: -65536 to 65535 (Quadrature), -65536 to 0 (Stepper)

This parameter is available only on stepper, resolver, and quadrature axes. It defines the 16-bit position unit range. For a complete discussion on the use of this parameter, read the topic that applies to the transducers you are using:

- Scaling Servo Axes

- Scaling Stepper Axes
- Scaling Resolver Axes

What if it is displayed under RMCWin incorrectly?

The Coordinate Limit parameter may be displayed incorrectly on the RMCWin main screen in some circumstances. The numbers are not necessarily incorrect (the module still functions correctly), but they may not look right. This is because RMCWin assumes the Coordinate Limit read from the RMC has a sign opposite of the scale for Quadrature (servo) axes until it is told otherwise. To correct the problem, enter the correct value for the Coordinate Limit and save the parameters to a file (on the File menu, click Save). Whenever you start RMCWin, read the parameters from the file and the problem should be corrected.

C.2.6 Extend Limit

Default: Current position on power-up
Range: Valid 16-bit Position

The Extend Limit specifies the maximum requested position value that the RMC will allow as a Command Value. (When the Scale is negative, this is the minimum value.) A Command Value that exceeds this value will be set to the Extend Limit, and the Parameter Error bit in the Status word will be set. The Extend Limit is given in Position Units.

Note: The Extend Limit must be changed when the Scale or Offset parameters are changed. Extending is the direction that gives increasing Transducer Counts.

Note: On startup, the Extend Limit defaults to the current position of the axis, so new Extend and Retract Limits must be issued followed by a 'r;P' command before the axis will move, unless different limits have been saved in the Flash.

C.2.7 Retract Limit

Default: Current position on power-up
Range: Valid 16-bit Position

The Retract Limit specifies the minimum value the motion controller will allow as a position Command Value. (When the Scale is negative, this is the maximum value.) A Command Value below this value will be set to the Retract Limit and will cause the parameter error bit in the Status word to be set. The Retract Limit is given in Position Units.

Note: The Retract Limit must be changed when the Scale or Offset parameters are changed.

Note: On startup, the Retract Limit defaults to the current position of the axis, so new Extend and Retract Limits must be issued followed by a 'r;P' command before the axis will move, unless different limits have been saved in the Flash.

C.2.8 Proportional Gain

Default: 1
Range: 0 to 65535

The Proportional Gain controls how much drive is generated proportional to the Position Error. The Position Error is defined as the Target Position minus the Actual Position. The units on the Proportional Gain is millivolts per 10 units of Position Error.

The Proportional Drive is defined as follows:

$$\text{ProDrive (mV)} = \text{ProGain (mV/10pu)} \times \text{PosError (pu)}$$

Or more simply:

$$\text{ProDrive (mV)} = \frac{\text{ProGain (mV/pu)} \times \text{PosError (pu)}}{10}$$

where pu is position units.

Note: The actual drive output may be reduced based on the values of the Extend Feed Forward and Retract Feed Forward .

CAUTION: Increase the Proportional Gain gradually. Excessive gain can cause oscillation that could cause damage or injury.

Think about this:

Internally, the motion controller must compare the error between the Target and Actual Positions with error limits to keep values from overflowing. The error limit is the error at which full drive (10 volts) will occur. This internal error limit is calculated as follows:

Error Limit = 100,000 / Proportional Gain

Therefore, if the Proportional Gain is set to 100, the Error Limit will be 1,000, which means any error greater than 1,000 will be treated as an error of 1,000 and the Overdrive error bit will be set in the Status Word.

C.2.9 Integral Gain

Default: 1
Range: 0 to 65535

The Integral Gain is used to control the amount of drive provided by the integrator. The integrator adds the position error to an accumulator every millisecond. The Integral Gain should be adjusted after the feed forwards have been set to optimal values. Using the integrator before the feed forwards have been set properly will cause the system to overshoot the target. We recommend that you set the Integral Gain to a value of at least 50.

Integral Gain is defined as:

Integral Gain = millivolts per 10240 counts of accumulated Position Error

Integral Drive is defined as:

Integral Drive = Integral Gain x Accumulated Counts / 10240

Note: The actual drive output may be reduced based on the values of the Extend Feed Forward and Retract Feed Forward .

Why Bother?

Integral Gain should be used to compensate for the fact that loads may vary, valves are non-linear and the axis may have trouble getting to the Command Position without Integral Gain.

C.2.10 Differential Gain

Default: 0
Range: 0 to 65535

The Differential Gain field is used to apply a gain based on the rate of change between the target and actual positions. There are two ways to view this.

First, this is a gain multiplied by the current rate of change in the position error. The differential drive, in millivolts is computed as follows:

$$\text{Differential Drive (mV)} = \frac{\text{ErrorChangeRate} \times K_d}{10}$$

$$\text{ErrorChangeRate(pos - units/ms)} = \frac{(E_0 - E_1)}{\text{LoopTime}}$$

`where:`

K_d=	Differential Gain in mV/[pos-units/ms]
E_0=	position error this control loop in position units
E_1=	position error last control loop in position units
LoopTime =	RMC Control Loop Time in ms (1 or 2)

A second equivalent way of viewing this parameter is as the gain multiplied by the velocity error. When looked at from this angle, the above equations become the following:

$$\text{Differential Drive (mV)} = \frac{(\text{TarVel} - \text{ActVel}) \times K_d}{10240}$$

where:

K_d=	Differential Gain in mV/[pos-units/s]
TarVel =	target velocity in pos-units/s
ActVel =	actual velocity in pos-units/s

Note: The actual drive output may be reduced based on the values of the Extend Feed Forward and Retract Feed Forward.

CAUTION: To avoid oscillation during initial tuning start with values below 10.

Why Bother?
The Differential Gain field will usually be set to zero. Noisy or low-resolution feedback will limit the amount the Differential Gain can be increased.

C.2.11 Extend Feed Forward

Default: 100
Range: 0 to 65535

TIP: After the axis has made a complete move without oscillations or overdrive errors, use the 'F' command to automatically set the Feed Forward value.

Feed Forward is an open loop compensation that is proportional to the Target Speed of the axis. This value is expressed in terms of millivolts per 1,000 Position Units per second. Extend Feed Forward drive is added to the output only when the axis is extending. The drive output provided by the Extend Feed Forward is determined as follows:

$$\text{Feed Forward Drive} = \frac{\text{Extend Feed Forward} \times \text{Target Speed}}{1000}$$

You can find the appropriate value for Extend Feed forward by making a move with the axis using a Speed of 1,000. The amount of output drive required to maintain this speed should be used as the Extend Feed Forward parameter. If the axis lags after the parameter has been set, the feed forward is too small or the system response is too slow. If the axis leads, the feed forward is too large or the system response is too slow. The 'F' command used after an extend move will automatically adjust the Extend Feed Forward parameter. The Extend and Retract Feed Forwards are also known as velocity feed forwards.

Note:
Feed Forward Drive is limited to a maximum value of 10,000. Inserting that value in the equation

above and solving for the Feed Forward term gives the follow relationship:

Feed Forward <= (10,000 * 1,000) / Target speed

That is, the larger the Speed the smaller the maximum Feed Forward value.

For example, at 30 in/sec Speed, the maximum Feed Forward is:

Max Feed Forward = (10,000 * 1,000)/30,000 = 10,000/30 = 333

If you set Speed to 30,000 and enter a Feed Forward value larger than 333, the value will be reduced to 333.

Think about this:

If the feed forward value is less than 300, an axis going at a speed of 65535 will not get a full valve drive from the feed forward term alone. If the feed forward value is less than 200, the axis drive system is too big and the system may have trouble controlling position. If this happens, then reduce the gain on the amplifier or use a smaller valve or reduce pressure until the feed forward terms are over 300.

On hydraulic systems, the extend and retract feed forward terms will be different by the ratio of the extend and retract surface areas of the position.

The Proportional Gain, Integral Gain and Differential Gain as well as the Integral Drive Limit are adjusted internally based on the ratio of the Extend and Retract Feed Forwards. The direction with the lower Feed Forward value will have effectively lower Gains and Integral Limit. This is done to compensate for the different system dynamics in the two directions.

For example, for a system with an Extend Feed forward of 100 and a Retract Feed Forward of 200 the effective gains in the extend direction will be one half (100 / 200) as big as the specified gains.

C.2.12 Retract Feed Forward

Default: 100
Range: 0 to 65535

Same as Extend Feed Forward, except that it is used when retracting.

Note: Retracting is the direction that returns decreasing Transducer Counts.

C.2.13 Extend Acceleration Feed Forward

Default: 0
Range: 0 to 65535

The Extend Acceleration Feed Forward causes the controller to give extra drive while accelerating and a braking drive while decelerating (negative acceleration) in the extend direction. The drive due to Acceleration Feed Forward is calculated as follows:

$$\text{Accel FF Drive} = \frac{\text{Extend Accel FF} \times 1\text{mV}}{100{,}000 \text{ position units/sec/sec}}$$

The Acceleration Feed Forward provides a second order approximation (prediction) of how much drive is required to move.

If the position unit is 0.001 inches, this equals millivolts of drive per 100 inches per second per second.

Why Bother?

The acceleration feed forward helps the axis track better while accelerating or decelerating. The acceleration feed forwards should not be adjusted until the velocity feed forwards are adjusted correctly.

C.2.14 Retract Acceleration Feed Forward

Default: 0
Range: 0 to 65535

Same as Extend Acceleration Feed Forward, except it is used when retracting.

C.2.15 Dead Band Eliminator

Default: 0
Range: 0 to 2000

Some valves and drives do not react to small changes in output around the null drive value; this effect is termed "dead band". The Dead Band Eliminator helps alleviate this problem by applying extra drive.

The RMC100 has two different Deadband eliminator algorithms:

- **Proportional (recommended)**
  The number of millivolts specified in the Dead Band Eliminator value is added to or subtracted from the drive output (depending on the direction of travel) when the Actual Position is outside of the In Position window. This causes the drive to be outside of the deadband. When the Actual Position is within the In Position window, the drive is applied proportional to the distance from the actual position. The Integral gain is always held at its current value. This algorithm is the recommended algorithm.
- **Immediate (RMC100 default)**
  The number of millivolts specified in the Dead Band Eliminator value is always added to or subtracted from the drive output (depending on the direction of travel) so the drive output is outside the dead band.

Note: In RMC100 CPU firmware prior to 20030916, the Dead Band Eliminator always used the immediate algorithm.

If a value outside the valid range is entered, the parameter error bit will be set, and the value will be set to zero.

CAUTION: Do not make this value too large or the drive will oscillate.

Selecting a Deadband Algorithm

To select a Deadband algorithm follow these steps:

1. In RMCWin, on the Tools menu, click Module Configuration.
2. In the Slots list, click the CPU.
3. Click Slot Options. The RMC100/101 CPU Options dialog box will be displayed.
4. Click the Deadband tab.
5. Select an algorithm.
6. Click Update RMC.
7. The Update Module Configuration dialog box will be displayed to indicate the progress. If the module could not be reset automatically, you may be prompted to reset the module yourself.

C.2.16 In Position

Default: 50
Range: 0 to 65535

This parameter specifies the size of a window around the Command Position. When the Actual Position gets within this window, the In Position bit is set in the Status word. Notice that the In Position bit is not latched and therefore could go off again if the axis moves back outside the In Position window.

Example:

If an axis Command Position is 10,000 and the In Position parameter is 30, the In Position bit will be set when the axis is stopped and its Actual Position is between 9,971 and 10,029. The bit will be cleared whenever the Actual Position is outside the range.

C.2.17 Following Error

Default: 250
Range: 0 to 65535

The Following Error determines how large the difference between the Target Position and Actual Position can get before the Following Error bit is set in the Status word.

C.2.18 Auto Stop

Default: 0x1FE0 (Soft Stops enabled, Hard Stops enabled for Transducer Errors)

Click here for the Auto Stop Bit Map

The Auto Stop parameter controls the action taken on the rising edge of any of the axis' error bits (the eight (8) most-significant bits in the Status word listed below). The user has control over which error bits cause which levels of stop, or whether an error will cause a stop at all. The default setting of all the AutoStops is Hard Stop, as described below.

During startup and tuning, you will typically need to set some AutoStops to Status Only to keep halts from interfering with the tuning. After completeing the startup procedure, make sure to set the AutoStops to setting that are safe for the machine operation.

Auto Stops are always active unless the axis is in Open Loop, in which case no Auto Stops will be triggered even though error bits may be set.

Note: AutoStops will not occur if the axis is in Open Loop!

The eight (8) most-significant bits in the Status word, as listed below, are the error bits, with the exception of the Home Input status bit for Quadrature axes with Analog (QUAD) or Stepper (STEP) output:

Fault	QUAD	STEP	All Others
7	Encoder Error/Fault	Encoder Error/Fault	No Transducer
6	Extend Limit	Extend Limit	Transducer Noise
5	Retract Limit	Retract Limit	Transducer Overflow
4	Overdrive	Overdrive	Overdrive
3	Parameter Error	Parameter Error	Parameter Error
2	Home Input	Home Input	Pos./Press. Overflow
1	Integrator Windup	Compensation Timeout	Integrator Windup
0	Following Error	Following Error	Following Error

The available actions for each fault are listed below:

- **Status Only**
  The fault will be reflected in its corresponding status bit in the Status word, but no further action will be taken. For some transducer types, this option may not be available for the following faults: No Transducer, Transducer Overflow, and Transducer Noise.
- **Soft Stop**
  If the axis is in closed loop, the fault will trigger ramping the axis to a stop. The speed will ramp down to zero using the current Deceleration value. If the axis is in Open Loop, the drive will not be affected. If there is an event sequence on the axis, it will stop executing. The Halt status bit will also be set in the Status word. Some faults will use the open loop ramp down although the axis was in closed loop: Position Overflow, No Transducer, Transducer Overflow, Transducer Noise, and Encoder Error/Fault Input. This is done because the position feedback is not dependable and closed loop control cannot be maintained.
- **Hard Stop**
  If the axis is in closed loop, the fault will trigger the drive output to go immediately to 0 mV on analog outputs and no steps for stepper outputs, and the axis will be placed in open loop mode. If the axis is in Open Loop, the drive will not be affected. If there is an event sequence on the axis, it will stop executing. The Halt and Open Loop status bit will also be set in the Status word.
- **Disable Drive**
  The fault will be handled as in a Hard Stop, but additionally the Amp Enable output on QUAD and STEP will be opened. Use the Amp Enable (a) command to close the output again. This option is only available for faults on QUAD and STEP axes.

To view or change the Auto Stop parameter value, open the Auto Stop popup editor in RMCWin using one of the methods described in Using Popup Editors. Select the desired action for each fault type and click OK. Changes to this parameter do not take effect until you issue a Set Parameters (P) command.

Manual Parameter Entry

This parameter can also be edited manually, but this is discouraged since it is much easier to use the popup editor. You can enter hexadecimal numbers by typing a leading 0x, as in 0x1FE0. Each of the eight faults listed above has two bits assigned to it, labeled S and H in the table below:

Bit	Description	Bit	Description
15	Fault 7 - Bit S	7	Fault 7 - Bit H
14	Fault 6 - Bit S	6	Fault 6 - Bit H
13	Fault 5 - Bit S	5	Fault 5 - Bit H
12	Fault 4 - Bit S	4	Fault 4 - Bit H
11	Fault 3 - Bit S	3	Fault 3 - Bit H
10	Fault 2 - Bit S	2	Fault 2 - Bit H
9	Fault 1 - Bit S	1	Fault 1 - Bit H
8	Fault 0 - Bit S	0	Fault 0 - Bit H

For each fault, the two bits in the Auto Stop parameter define the action as follows:

Bit H	Bit S	Description
0	0	Status Only
0	1	Soft Stop
1	0	Hard Stop
1	1	Disable Drive

If you select Status Only for a fault that cannot use that action, then the axis will use a Soft Stop action for that fault. Similarly, if you select Disable Drive for a fault on an axis that does not have an Amp Enable output, the axis will use the Hard Stop action for the fault.

C.2.19 Auto Stop Bit Map

The table below provides an easy method to convert bit patterns to hexadecimal numbers.

```
                         F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                         E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
    Hexadecimal To       D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
    Binary Conversion    C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
    Table                B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                         A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                         9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                         8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                         7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                         6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                         5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                         4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                         3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                         2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                         1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                         0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0
                            _______   _______   _______   _______
                           | | | | | | | | | | | | | | | | | | | |
                           |1|1|1|1| |1|1| | | | | | | | | | | | |
                           |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
 Bit Definition            |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
----------------            / / / /   / / / /   / / / /   / / / /
S  No Xdcr/Encdr Flt - 15 -/ / / /   / / / /   / / / /   / / / /
O  Xdcr Nse/Ext Lmt -- 14 --/ / /   / / / /   / / / /   / / / /
F  Xdcr Ovr/Ret Lmt -- 13 ---/ /   / / / /   / / / /   / / / /
T  Overdrive  -------- 12 ----/   / / / /   / / / /   / / / /
                                 / / / /   / / / /   / / / /
S  Parameter Error --- 11 ------/ / / /   / / / /   / / / /
T  Pos Ovr/Home Inpt - 10 -------/ / /   / / / /   / / / /
O  Integrator Windup - 09 --------/ /   / / / /   / / / /
P  Following Error --- 08 ---------/   / / / /   / / / /
=========================             / / / /   / / / /
H  No Xdcr/Encdr Flt - 07 -----------/ / / /   / / / /
A  Xdcr Nse/Ext Lmt -- 06 ------------/ / /   / / / /
R  Xdcr Ovr/Ret Lmt -- 05 -------------/ /   / / / /
D  Overdrive --------- 04 --------------/   / / / /
                                           / / / /
S  Parameter Error --- 03 ----------------/ / / /
T  Pos Ovr/Home Inpt - 02 -----------------/ / /
O  Integrator Windup - 01 ------------------/ /
P  Following Error --- 00 -------------------/
```

If both Soft Stop and Hard Stop bits are set for a particular error condition, a Hard Stop will be executed and the Amp Enable output will be opened on QUAD and STEP axes.

C.3 Quadrature with Stepper Output Parameters

C.3.1 Configuration Word

Default: 0x0000

This 16-bit word controls the configuration of the module. Bit 0 is the LSB; bit 15 is the MSB.

Click here for the Config Word Bit Map

Bits 7, 12-15 - Transducer Type bits
These bits are used differently depending on the transducer type used. Refer to one of the following topics for details on these four bits:

- Analog Transducer bits
- Magnetostrictive Displacement Transducer (MDT) bits
- Quadrature/Stepper Transducer bits
- Synchronous Serial Interface (SSI) bits
- Resolver Transducer Bits

Bit 11 - Pressure Assign bit
This bit is used only if the RMC has one or more auxiliary pressure or force channels. If such channels are installed, then this bit can be set to indicate that this position axis has an assigned pressure axis. Bits 8 and 9 are used to define the pressure axis assigned.

Bit 10 - Auto Home Re-arm
This bit is for Quadrature axes only. When it is set, the axis will automatically be re-armed after homing. This requires the axis to first be armed using the Arm Home Command. If this bit is cleared after homing, the axis will still be armed. See the Homing a Quadrature Axis topic for details.

Bits 8 - 9 - Pressure Axis Select bits
These bits are used only if the RMC has one or more auxiliary pressure or force channels and bit 11 is set to indicate that this axis has one of the pressure or force channels assigned to it. These bits then select between up to four auxiliary pressure/force channels:

Bit 9	**Bit 8**	
0	0	The first auxiliary pressure/force axis is assigned.
0	1	The second auxiliary pressure/force axis is assigned.
1	0	The third auxiliary pressure/force axis is assigned.

1	1	The fourth auxiliary pressure/force axis is assigned.

It is necessary to assign a pressure axis to a position axis in order to switch between position and pressure control. Refer to Using an Analog Channel as a Pressure Axis for details.

Note: Two position axes cannot be assigned to the same pressure axis. If this does occur, only the first axis to make the assignment followed by the Set Parameters (P) command will succeed in being assigned. Any later attempt to assign the pressure axis will fail with a parameter error.

Bit 6 - Reverse Drive Mode bit

Warning: This bit must be set properly when the Set Parameters command is issued.

This bit determines the relationship between the RMC's position direction and the drive output direction.

For analog-output axes, setting this bit reverses the polarity of the axis output voltage. Although in many cases the differential drive wires can be swapped on the drive unit, this is not always possible or it may simply be easier to use this bit.

Bit 6	
0	The RMC will generate positive drive to increase transducer counts.
1	The RMC will generate negative drive to increase transducer counts.

For stepper-output axes, this bit is used to change the meaning of the Direction output with relation to the axis transducer counts.

Bit 6	
0	The RMC will use a low Direction output to increase transducer counts.
1	The RMC will use a high Direction output to increase transducer counts.

Bit 5 - Absolute Mode bit

This mode is intended for use with a two-valve system, one controlling the flow rate and the other the direction. This axis controls the flow-rate valve. The directional valve must be controlled by other means. In this mode, the axis generates a positive drive output regardless of the direction of the move. The drive will not go negative if the motion controller overshoots the target. This is useful for some injection and blow-molding applications.

When the axis is stopped, you must go into open loop mode.

Warning: This bit must be set properly when the Set Parameters command is issued.

Bit 4 - Continue Mode bit

This bit affects what happens when the module loses contact with the Programmable Controller. When this bit is set, the module will finish any move it has started, otherwise it will halt immediately upon detecting loss of contact with the PLC. This is useful for finishing a move that must complete to prevent machine downtime (for example, a partial shot in an injection-molding machine).

Bit 3 - Simulate bit

When this bit is set, the drive output is set to zero and the magnetostrictive transducer inputs are ignored. Internally the Target Position is used as the Actual Position. This mode is used for debugging. (The transducer error bits and LEDS will be cleared.)

Note: Drive output is always 0 volts while in simulate mode; no commands, including open loop, will change this.

Bits 1 - 2 - Prescale Divisor Bits

Prescaling may be desired on systems with long strokes. The prescale bits are used along with the Scale parameter to convert Transducer Counts to Actual Position. The effect is to divide the Scale by 1, 2, 4, or 8. The divide-by factor is specified as follows:

Counts ÷	Bit 2	Bit 1
1	0	0
2	0	1
4	1	0
8	1	1

The advantage of these bits is that they let you maintain a higher resolution in the Scale parameter. For example, a Scale of 31027 and a divide-by of 4 gives an effective scale of 7756.75. Without using the divide-by bits, the Scale would have to be rounded up to 7757, which has less resolution—with a 64" rod, the rounded value would make a 0.002" error at 64". Note: The stroke is always limited to 220 inches due to the 18-bit transducer counter resolution.

These bits must be zero for Resolver modules.

Bit 0 - Integrator Limit

When this bit is cleared, the integrator limit is 20% of full drive. When the bit is set, the limit is 80%. If the integrator tries to go above the limit, the Integrator Windup bit in the status word is set and the integrator value is held at the limit. The integrator limit is designed to prevent the drive output from saturating.

Note: The actual drive output may be reduced based on the values of the Extend Feed Forward and Retract Feed Forward .

C.3.2 Configuration Word Bit Map

The axis Configuration word contains 16 bits of information. The hexadecimal table below provides an easy way to convert hexadecimal numbers to bit patterns.

```
                      F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                      E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
 Hexadecimal To       D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
 Binary Conversion    C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
 Table                B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                      A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                      9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                      8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                      7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                      6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                      5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                      4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                      3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                      2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                      1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                      0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0

                         _______   _______   _______   _______
                        | | | | | | | | | | | | | | | | | | | |
                        |1|1|1|1| |1|1| | | | | | | | | | | | |
                        |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
 Bit Definition         |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
----------------         / / / /   / / / /   / / / /   / / / /
Transducer Type --- 15 -/ / / /   / / / /   / / / /   / / / /
Transducer Type --- 14 --/ / /   / / / /   / / / /   / / / /
Transducer Type --- 13 ---/ /   / / / /   / / / /   / / / /
Transducer Type --- 12 ----/   / / / /   / / / /   / / / /
                              / / / /   / / / /   / / / /
Pressure Assign --- 11 ------/ / / /   / / / /   / / / /
Reserved ---------- 10 -------/ / /   / / / /   / / / /
Pressure Axis Sel - 09 --------/ /   / / / /   / / / /
Pressure Axis Sel - 08 ---------/   / / / /   / / / /
                                   / / / /   / / / /
Transducer Type --- 07 -----------/ / / /   / / / /
Reverse Drive ----- 06 ------------/ / /   / / / /
Absolute Mode ----- 05 -------------/ /   / / / /
Continue Mode ----- 04 --------------/   / / / /
                                        / / / /
Simulate Mode ----- 03 ----------------/ / / /
Prescale Bits ----- 02 -----------------/ / /
Prescale Bits ----- 01 ------------------/ /
Integrator Limit -- 00 -------------------/
```

C.3.3 Configuration Bits - Quadrature/Stepper Specific

For quadrature axes, these bits are used to select the active states of four inputs and to define the use of the Home Input status bit in the Status word.

Fault Input Active State - Bit 15

This bit determines the active state of the Fault Input. The active state of this input indicates that a fault has occurred. The Encoder Error status bit is set when the Fault input is active.

Bit 15	Active Input State
0	No current applied
1	Current Applied

Limit Inputs Active State - Bit 14

Determines the active state of the Extend (CW) and Retract (CCW) Limit Inputs. The active state of these inputs indicates that a limit has been reached. The Extend Limit status bit will be set when the Extend (CW) Limit Input is active, and the Retract Limit status bit will be set when the Retract (CCW) Limit Input is active:

Bit 14	Active Input State
0	No current applied
1	Current applied

Index (Z) Input Active State - Bit 13

Determines the active state of the Index (Z) Input:

Bit 13	Active Input State
0	Positive signal applied
1	Negative signal applied

Home (H) Input Active State - Bit 12

Determines the active state of the Home (H) Input:

Bit 12	Active Input State
0	Current applied
1	No current applied

Home Input Status Bit Level/Edge Select - Bit 7

This bit determines the use of the Home Input status bit, as shown below:

Bit 7	Home Input Status Bit Behavior
0	Latched Home Edge The Home Input status bit is set and latched on the transition from both the index (Z) and home (H) inputs being active to one or both being inactive.

1	Level of H Input The Home Input status bit is set when the Home (H) input is active. It is not latched.

For details on using this bit, see Homing a Quadrature Axis.

C.3.4 Coord. Limit

Default: 0
Range: -65536 to 65535 (Quadrature), -65536 to 0 (Stepper)

This parameter is available only on stepper, resolver, and quadrature axes. It defines the 16-bit position unit range. For a complete discussion on the use of this parameter, read the topic that applies to the transducers you are using:

- Scaling Servo Axes
- Scaling Stepper Axes
- Scaling Resolver Axes

What if it is displayed under RMCWin incorrectly?

The Coordinate Limit parameter may be displayed incorrectly on the RMCWin main screen in some circumstances. The numbers are not necessarily incorrect (the module still functions correctly), but they may not look right. This is because RMCWin assumes the Coordinate Limit read from the RMC has a sign opposite of the scale for Quadrature (servo) axes until it is told otherwise. To correct the problem, enter the correct value for the Coordinate Limit and save the parameters to a file (on the File menu, click Save). Whenever you start RMCWin, read the parameters from the file and the problem should be corrected.

C.3.5 Extend Limit

Default: Current position on power-up
Range: Valid 16-bit Position

The Extend Limit specifies the maximum requested position value that the RMC will allow as a Command Value. (When the Scale is negative, this is the minimum value.) A Command Value that exceeds this value will be set to the Extend Limit, and the Parameter Error bit in the Status word will be set. The Extend Limit is given in Position Units.

Note: The Extend Limit must be changed when the Scale or Offset parameters are changed. Extending is the direction that gives increasing Transducer Counts.

Note: On startup, the Extend Limit defaults to the current position of the axis, so new Extend and Retract Limits must be issued followed by a ’r;P’ command before the axis will move, unless different limits have been saved in the Flash.

C.3.6 Retract Limit

Default: Current position on power-up
Range: Valid 16-bit Position

The Retract Limit specifies the minimum value the motion controller will allow as a position Command Value. (When the Scale is negative, this is the maximum value.) A Command Value below this value will be set to the Retract Limit and will cause the parameter error bit in the Status word to be set. The Retract Limit is given in Position Units.

Note: The Retract Limit must be changed when the Scale or Offset parameters are changed.

Note: On startup, the Retract Limit defaults to the current position of the axis, so new Extend and Retract Limits must be issued followed by a 'r;P' command before the axis will move, unless different limits have been saved in the Flash.

C.3.7 Comp. Rate

Default: 0
Range: 0 (disabled) or 1 to 65535 position units per second

This parameter is available only on stepper axes. It is one of several parameters that control the compensation feature. For a complete description of the compensation feature, see the Stepper Compensation topic.

This parameter determines the speed at which the axis will try to return to the target position. If the axis is in motion, this rate will be added to the target speed. The compensation rate is only added when the actual position is outside the Compensation or In Position Window as described in the Stepper Compensation topic. To disable compensation, enter a 0 for this parameter.

C.3.8 Comp. Timeout

Default: 0
Range: 0 (no timeout) or 1 to 65535 seconds

This parameter is available only on stepper axes. It is one of several parameters that control the compensation feature. For a complete description of the compensation feature, see the Stepper Compensation topic.

When the axis is stopped outside the In Position Window, compensation will be applied to try to move the axis back to within the window. If the axis is not able to move back into the In Position Window in the time specified by the Compensation Timeout parameter, the Timeout bit will be set in the Status word. This bit can be used to trigger a hard stop on the axis, as controlled by the Auto Stop Mask parameter. Do not use the soft stop with this bit, since the axis is already stopped, so the closed loop stop initiated by the soft stop does nothing.

A Compensation Timeout of 0 will disable the timeout feature so compensation will continue until the position moves within the window.

C.3.9 Steps/Rev

Default: 1
Range: 1 to 65535

This parameter is available on stepper axes only. It is used with the Pos Units/Rev parameter to determine the scaling between position units and steps output by the RMC. These steps can be interpreted by the stepper drive as either a full step or microstep.

The use of this parameter is summarized by the following equation, which is applied every control loop to convert the change in the Target Position to steps that are output.

$$\Delta\text{Steps} = \Delta\text{Target Position} \times \frac{\text{Steps/Rev}}{\text{Pos Units/Rev}}$$

As the name implies, the recommended value to enter in this parameter is the number of steps required to move the motor one revolution. Therefore, for a step motor with 48 steps per revolution and drive with 100 microsteps per step, you would enter a 4800 in this field. However, there may be times when you will not want to enter this value, but instead adjust this field in conjunction with the QUAD CNTS/REV and POS UNITS/REV fields. For a full discussion, including examples, on scaling with stepper modules, see Stepper Scaling.

C.3.10 Pos Units/Rev

Default: 1
Range: 1 to 65535

This parameter is available on stepper axes only. It is used with the Steps/Rev parameter to determine the scaling between position units and steps. It is also used with Quad Cnts/Rev to determine the scaling between position units and quadrature counts. Therefore it is used in the following two conversions:

$$\Delta\text{Steps} = \Delta\text{Target Position} \times \frac{\text{Steps/Rev}}{\text{Pos Units/Rev}}$$

$$\Delta\text{Actual Position} = \Delta\text{Quad Counts} \times \frac{\text{Pos Units/Rev}}{\text{Quad Counts/Rev}}$$

Because this parameter is used in both scales, changing its value will change the effective scale between position units and both quadrature counts and steps. For more information on the quadrature scale, see Quad Cnts/Rev. For more information on the step scale, see Steps/Rev.

As this parameter's name implies, the intended use of this parameter is to enter the number of position units you wish to have in one revolution. This works well in applications where the stepper, quadrature encoder, and position units are all taken from the same axis and there are a whole number of position units per revolution. The following are some possible values:

Position Units	**Pos Units/Rev**

Degrees	360
Tenths of a degree	3600
Hundredths of a degree	36000
Thousandths of a rev.	1000
Ten-thousandths of a rev.	10000

For a full discussion, including examples, on scaling with stepper modules, see Stepper Scaling.

C.3.11 Quad Cnts/Rev

Default: 1
Range: -32768 to 32767

This parameter is available on stepper axes only. It is used with the Pos Units/Rev parameter to determine the scaling between position units and quadrature counts. If the quadrature feedback is not being used, enter a zero for this parameter. This will make the Actual Position always match the Target Position, therefore eliminating the Following Error.

The sign of this parameter determines whether increasing quadrature counts (phase A leading phase B) translates to increasing or decreasing the current position:

Sign	Increasing quadrature counts means…
+	…increasing positions
-	…decreasing positions

The use of this parameter is summarized by the following equation, which is applied every control loop to convert the incoming quadrature counts to an Actual Position in user-defined position units.

$$\Delta \text{Actual Position} = \Delta \text{Quad Counts} \times \frac{\text{Pos Units/Rev}}{\text{Quad Counts/Rev}}$$

As the name implies, the recommended value to enter in this parameter is the number of quadrature counts generated in a single revolution of the motor. Therefore, for a 1000-line encoder mounted on the same shaft as the motor, you would enter 4000 in this parameter since there are 4000 quadrature counts per revolution on a 1000-line encoder. However, this becomes more complicated if the quadrature feedback is on another shaft that is mechanically geared to the motor shaft with a ratio other than 1:1. For a full discussion, including examples, on scaling with stepper modules, see Stepper Scaling.

C.3.12 Max Steps/MSec

Default: 1024

Range: 1 to 1024 steps per millisecond

This parameter is available only on stepper axes. It is used to ensure that the stepper is never driven beyond a given rate. If the axis is requested to give more steps per millisecond than allowed by this parameter, the Overdrive error bit will be set in the Status word. The axis can be set up to automatically stop based on this bit using the Auto Stop Mask parameter.

Following are three cases in which this parameter would be used:

- Limit the step rate to the capability of the motor.
- Limit the step rate to the capability of the system.
- Limit the step rate to a safe speed, especially during startup.

Because this parameter is given in steps per millisecond, this parameter should be set to 1/1000th of the maximum step frequency desired. For example, to limit the output to 100 kHz, enter 100 for this parameter.

C.3.13 Comp. Window

Default: 100
Range: 0 to 65535 position units

This parameter is available only on stepper axes. It is one of several parameters that control the compensation feature. For a complete description of the compensation feature, see the Stepper Compensation topic.

When the axis is not moving, the compensation is applied only when the axis is outside the In Position Window parameter. When the axis is moving, the compensation rate is applied when the actual position differs from the target position by more than the Compensation Window parameter. This window should normally be set to a value greater than the In Position Window, but less than the Following Error Window.

C.3.14 In Position

Default: 50
Range: 0 to 65535

This parameter specifies the size of a window around the Command Position. When the Actual Position gets within this window, the In Position bit is set in the Status word. Notice that the In Position bit is not latched and therefore could go off again if the axis moves back outside the In Position window.

Example:

If an axis Command Position is 10,000 and the In Position parameter is 30, the In Position bit will be set when the axis is stopped and its Actual Position is between 9,971 and 10,029. The bit will be cleared whenever the Actual Position is outside the range.

C.3.15 Following Error

Default: 250

Range: 0 to 65535

The Following Error determines how large the difference between the Target Position and Actual Position can get before the Following Error bit is set in the Status word.

C.3.16 Auto Stop

Default: 0x1FE0 (Soft Stops enabled, Hard Stops enabled for Transducer Errors)
Click here for the Auto Stop Bit Map

The Auto Stop parameter controls the action taken on the rising edge of any of the axis' error bits (the eight (8) most-significant bits in the Status word listed below). The user has control over which error bits cause which levels of stop, or whether an error will cause a stop at all. The default setting of all the AutoStops is Hard Stop, as described below.

During startup and tuning, you will typically need to set some AutoStops to Status Only to keep halts from interfering with the tuning. After completeing the startup procedure, make sure to set the AutoStops to setting that are safe for the machine operation.

Auto Stops are always active unless the axis is in Open Loop, in which case no Auto Stops will be triggered even though error bits may be set.

Note: AutoStops will not occur if the axis is in Open Loop!

The eight (8) most-significant bits in the Status word, as listed below, are the error bits, with the exception of the Home Input status bit for Quadrature axes with Analog (QUAD) or Stepper (STEP) output:

Fault	QUAD	STEP	All Others
7	Encoder Error/Fault	Encoder Error/Fault	No Transducer
6	Extend Limit	Extend Limit	Transducer Noise
5	Retract Limit	Retract Limit	Transducer Overflow
4	Overdrive	Overdrive	Overdrive
3	Parameter Error	Parameter Error	Parameter Error
2	Home Input	Home Input	Pos./Press. Overflow
1	Integrator Windup	Compensation Timeout	Integrator Windup
0	Following Error	Following Error	Following Error

The available actions for each fault are listed below:

- **Status Only**
  The fault will be reflected in its corresponding status bit in the Status word, but no further action will be taken. For some transducer types, this option may not be available for the following faults: No Transducer, Transducer Overflow, and Transducer Noise.
- **Soft Stop**
  If the axis is in closed loop, the fault will trigger ramping the axis to a stop. The speed will ramp down to zero using the current Deceleration value. If the axis is in Open Loop, the drive will not be affected. If there is an event sequence on the axis, it will stop executing. The Halt status bit will also be set in the Status word. Some faults will use the open loop ramp down although the axis was in closed loop: Position Overflow, No Transducer, Transducer Overflow, Transducer Noise,

and Encoder Error/Fault Input. This is done because the position feedback is not dependable and closed loop control cannot be maintained.

- **Hard Stop**
  If the axis is in closed loop, the fault will trigger the drive output to go immediately to 0 mV on analog outputs and no steps for stepper outputs, and the axis will be placed in open loop mode. If the axis is in Open Loop, the drive will not be affected. If there is an event sequence on the axis, it will stop executing. The Halt and Open Loop status bit will also be set in the Status word.
- **Disable Drive**
  The fault will be handled as in a Hard Stop, but additionally the Amp Enable output on QUAD and STEP will be opened. Use the Amp Enable (a) command to close the output again. This option is only available for faults on QUAD and STEP axes.

To view or change the Auto Stop parameter value, open the Auto Stop popup editor in RMCWin using one of the methods described in Using Popup Editors. Select the desired action for each fault type and click OK. Changes to this parameter do not take effect until you issue a Set Parameters (P) command.

Manual Parameter Entry

This parameter can also be edited manually, but this is discouraged since it is much easier to use the popup editor. You can enter hexadecimal numbers by typing a leading 0x, as in 0x1FE0. Each of the eight faults listed above has two bits assigned to it, labeled S and H in the table below:

Bit	Description	Bit	Description
15	Fault 7 - Bit S	7	Fault 7 - Bit H
14	Fault 6 - Bit S	6	Fault 6 - Bit H
13	Fault 5 - Bit S	5	Fault 5 - Bit H
12	Fault 4 - Bit S	4	Fault 4 - Bit H
11	Fault 3 - Bit S	3	Fault 3 - Bit H
10	Fault 2 - Bit S	2	Fault 2 - Bit H
9	Fault 1 - Bit S	1	Fault 1 - Bit H
8	Fault 0 - Bit S	0	Fault 0 - Bit H

For each fault, the two bits in the Auto Stop parameter define the action as follows:

Bit H	Bit S	Description
0	0	Status Only
0	1	Soft Stop
1	0	Hard Stop
1	1	Disable Drive

If you select Status Only for a fault that cannot use that action, then the axis will use a Soft Stop action for that fault. Similarly, if you select Disable Drive for a fault on an axis that does not have an Amp Enable output, the axis will use the Hard Stop action for the fault.

C.3.17 Auto Stop Bit Map

The table below provides an easy method to convert bit patterns to hexadecimal numbers.

```
                         F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                         E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
    Hexadecimal To       D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
    Binary Conversion    C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
    Table                B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                         A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                         9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                         8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                         7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                         6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                         5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                         4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                         3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                         2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                         1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                         0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0

                            _______   _______   _______   _______
                           | | | | | | | | | | | | | | | | | | | |
                           |1|1|1|1| |1|1| | | | | | | | | | | | |
                           |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
 Bit Definition            |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
----------------            / / / /   / / / /   / / / /   / / / /
S  No Xdcr/Encdr Flt - 15 -/ / / /   / / / /   / / / /   / / / /
O  Xdcr Nse/Ext Lmt -- 14 --/ / /   / / / /   / / / /   / / / /
F  Xdcr Ovr/Ret Lmt -- 13 ---/ /   / / / /   / / / /   / / / /
T  Overdrive  -------- 12 ----/   / / / /   / / / /   / / / /
                                 / / / /   / / / /   / / / /
S  Parameter Error --- 11 ------/ / / /   / / / /   / / / /
T  Pos Ovr/Home Inpt - 10 -------/ / /   / / / /   / / / /
O  Integrator Windup - 09 --------/ /   / / / /   / / / /
P  Following Error --- 08 ---------/   / / / /   / / / /
=========================             / / / /   / / / /
H  No Xdcr/Encdr Flt - 07 -----------/ / / /   / / / /
A  Xdcr Nse/Ext Lmt -- 06 ------------/ / /   / / / /
R  Xdcr Ovr/Ret Lmt -- 05 -------------/ /   / / / /
D  Overdrive --------- 04 --------------/   / / / /
                                           / / / /
S  Parameter Error --- 03 ----------------/ / / /
T  Pos Ovr/Home Inpt - 02 -----------------/ / /
O  Integrator Windup - 01 ------------------/ /
P  Following Error --- 00 -------------------/
```

If both Soft Stop and Hard Stop bits are set for a particular error condition, a Hard Stop will be executed and the Amp Enable output will be opened on QUAD and STEP axes.

C.4 SSI with Stepper Output Parameters

C.4.1 Configuration Word

Default: 0x0000

This 16-bit word controls the configuration of the module. Bit 0 is the LSB; bit 15 is the MSB.

Click here for the Config Word Bit Map

Bits 7, 12-15 - Transducer Type bits
These bits are used differently depending on the transducer type used. Refer to one of the following topics for details on these four bits:

- Analog Transducer bits
- Magnetostrictive Displacement Transducer (MDT) bits
- Quadrature/Stepper Transducer bits
- Synchronous Serial Interface (SSI) bits
- Resolver Transducer Bits

Bit 11 - Pressure Assign bit
This bit is used only if the RMC has one or more auxiliary pressure or force channels. If such channels are installed, then this bit can be set to indicate that this position axis has an assigned pressure axis. Bits 8 and 9 are used to define the pressure axis assigned.

Bit 10 - Auto Home Re-arm
This bit is for Quadrature axes only. When it is set, the axis will automatically be re-armed after homing. This requires the axis to first be armed using the Arm Home Command. If this bit is cleared after homing, the axis will still be armed. See the Homing a Quadrature Axis topic for details.

Bits 8 - 9 - Pressure Axis Select bits
These bits are used only if the RMC has one or more auxiliary pressure or force channels and bit 11 is set to indicate that this axis has one of the pressure or force channels assigned to it. These bits then select between up to four auxiliary pressure/force channels:

Bit 9	Bit 8	
0	0	The first auxiliary pressure/force axis is assigned.
0	1	The second auxiliary pressure/force axis is assigned.
1	0	The third auxiliary pressure/force axis is assigned.

1	1	The fourth auxiliary pressure/force axis is assigned.

It is necessary to assign a pressure axis to a position axis in order to switch between position and pressure control. Refer to Using an Analog Channel as a Pressure Axis for details.

Note: Two position axes cannot be assigned to the same pressure axis. If this does occur, only the first axis to make the assignment followed by the Set Parameters (P) command will succeed in being assigned. Any later attempt to assign the pressure axis will fail with a parameter error.

Bit 6 - Reverse Drive Mode bit

Warning: This bit must be set properly when the Set Parameters command is issued.

This bit determines the relationship between the RMC's position direction and the drive output direction.

For analog-output axes, setting this bit reverses the polarity of the axis output voltage. Although in many cases the differential drive wires can be swapped on the drive unit, this is not always possible or it may simply be easier to use this bit.

Bit 6	
0	The RMC will generate positive drive to increase transducer counts.
1	The RMC will generate negative drive to increase transducer counts.

For stepper-output axes, this bit is used to change the meaning of the Direction output with relation to the axis transducer counts.

Bit 6	
0	The RMC will use a low Direction output to increase transducer counts.
1	The RMC will use a high Direction output to increase transducer counts.

Bit 5 - Absolute Mode bit

This mode is intended for use with a two-valve system, one controlling the flow rate and the other the direction. This axis controls the flow-rate valve. The directional valve must be controlled by other means. In this mode, the axis generates a positive drive output regardless of the direction of the move. The drive will not go negative if the motion controller overshoots the target. This is useful for some injection and blow-molding applications.

When the axis is stopped, you must go into open loop mode.

Warning: This bit must be set properly when the Set Parameters command is issued.

Bit 4 - Continue Mode bit

This bit affects what happens when the module loses contact with the Programmable Controller. When this bit is set, the module will finish any move it has started, otherwise it will halt immediately upon detecting loss of contact with the PLC. This is useful for finishing a move that must complete to prevent machine downtime (for example, a partial shot in an injection-molding machine).

Bit 3 - Simulate bit

When this bit is set, the drive output is set to zero and the magnetostrictive transducer inputs are ignored. Internally the Target Position is used as the Actual Position. This mode is used for debugging. (The transducer error bits and LEDS will be cleared.)

Note: Drive output is always 0 volts while in simulate mode; no commands, including open loop, will change this.

Bits 1 - 2 - Prescale Divisor Bits

Prescaling may be desired on systems with long strokes. The prescale bits are used along with the Scale parameter to convert Transducer Counts to Actual Position. The effect is to divide the Scale by 1, 2, 4, or 8. The divide-by factor is specified as follows:

Counts ÷	**Bit 2**	**Bit 1**
1	0	0
2	0	1
4	1	0
8	1	1

The advantage of these bits is that they let you maintain a higher resolution in the Scale parameter. For example, a Scale of 31027 and a divide-by of 4 gives an effective scale of 7756.75. Without using the divide-by bits, the Scale would have to be rounded up to 7757, which has less resolution—with a 64" rod, the rounded value would make a 0.002" error at 64". Note: The stroke is always limited to 220 inches due to the 18-bit transducer counter resolution.

These bits must be zero for Resolver modules.

Bit 0 - Integrator Limit

When this bit is cleared, the integrator limit is 20% of full drive. When the bit is set, the limit is 80%. If the integrator tries to go above the limit, the Integrator Windup bit in the status word is set and the integrator value is held at the limit. The integrator limit is designed to prevent the drive output from saturating.

Note: The actual drive output may be reduced based on the values of the Extend Feed Forward and Retract Feed Forward .

C.4.2 Configuration Word Bit Map

The axis Configuration word contains 16 bits of information. The hexadecimal table below provides an easy way to convert hexadecimal numbers to bit patterns.

```
                      F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                      E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
 Hexadecimal To       D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
 Binary Conversion    C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
 Table                B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                      A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                      9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                      8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                      7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                      6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                      5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                      4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                      3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                      2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                      1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                      0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0

                         _______   _______   _______   _______
                        | | | | | | | | | | | | | | | | | | | |
                        |1|1|1|1| |1|1| | | | | | | | | | | | |
                        |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
 Bit Definition         |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
----------------         / / / /   / / / /   / / / /   / / / /
Transducer Type --- 15 -/ / / /   / / / /   / / / /   / / / /
Transducer Type --- 14 --/ / /   / / / /   / / / /   / / / /
Transducer Type --- 13 ---/ /   / / / /   / / / /   / / / /
Transducer Type --- 12 ----/   / / / /   / / / /   / / / /
                              / / / /   / / / /   / / / /
Pressure Assign --- 11 ------/ / / /   / / / /   / / / /
Reserved ---------- 10 -------/ / /   / / / /   / / / /
Pressure Axis Sel - 09 --------/ /   / / / /   / / / /
Pressure Axis Sel - 08 ---------/   / / / /   / / / /
                                   / / / /   / / / /
Transducer Type --- 07 -----------/ / / /   / / / /
Reverse Drive ----- 06 ------------/ / /   / / / /
Absolute Mode ----- 05 -------------/ /   / / / /
Continue Mode ----- 04 --------------/   / / / /
                                        / / / /
Simulate Mode ----- 03 ----------------/ / / /
Prescale Bits ----- 02 -----------------/ / /
Prescale Bits ----- 01 ------------------/ /
Integrator Limit -- 00 -------------------/
```

C.4.3 Configuration Bits - Quadrature/Stepper Specific

For quadrature axes, these bits are used to select the active states of four inputs and to define the use of the Home Input status bit in the Status word.

Fault Input Active State - Bit 15

This bit determines the active state of the Fault Input. The active state of this input indicates that a fault has occurred. The Encoder Error status bit is set when the Fault input is active.

Bit 15	Active Input State
0	No current applied
1	Current Applied

Limit Inputs Active State - Bit 14

Determines the active state of the Extend (CW) and Retract (CCW) Limit Inputs. The active state of these inputs indicates that a limit has been reached. The Extend Limit status bit will be set when the Extend (CW) Limit Input is active, and the Retract Limit status bit will be set when the Retract (CCW) Limit Input is active:

Bit 14	Active Input State
0	No current applied
1	Current applied

Index (Z) Input Active State - Bit 13

Determines the active state of the Index (Z) Input:

Bit 13	Active Input State
0	Positive signal applied
1	Negative signal applied

Home (H) Input Active State - Bit 12

Determines the active state of the Home (H) Input:

Bit 12	Active Input State
0	Current applied
1	No current applied

Home Input Status Bit Level/Edge Select - Bit 7

This bit determines the use of the Home Input status bit, as shown below:

Bit 7	Home Input Status Bit Behavior
0	Latched Home Edge The Home Input status bit is set and latched on the transition from both the index (Z) and home (H) inputs being active to one or both being inactive.

1	Level of H Input The Home Input status bit is set when the Home (H) input is active. It is not latched.

For details on using this bit, see Homing a Quadrature Axis.

C.4.4 Coord. Limit

Default: 0
Range: -65536 to 65535 (Quadrature), -65536 to 0 (Stepper)

This parameter is available only on stepper, resolver, and quadrature axes. It defines the 16-bit position unit range. For a complete discussion on the use of this parameter, read the topic that applies to the transducers you are using:

- Scaling Servo Axes
- Scaling Stepper Axes
- Scaling Resolver Axes

What if it is displayed under RMCWin incorrectly?

The Coordinate Limit parameter may be displayed incorrectly on the RMCWin main screen in some circumstances. The numbers are not necessarily incorrect (the module still functions correctly), but they may not look right. This is because RMCWin assumes the Coordinate Limit read from the RMC has a sign opposite of the scale for Quadrature (servo) axes until it is told otherwise. To correct the problem, enter the correct value for the Coordinate Limit and save the parameters to a file (on the File menu, click Save). Whenever you start RMCWin, read the parameters from the file and the problem should be corrected.

C.4.5 Extend Limit

Default: Current position on power-up
Range: Valid 16-bit Position

The Extend Limit specifies the maximum requested position value that the RMC will allow as a Command Value. (When the Scale is negative, this is the minimum value.) A Command Value that exceeds this value will be set to the Extend Limit, and the Parameter Error bit in the Status word will be set. The Extend Limit is given in Position Units.

Note: The Extend Limit must be changed when the Scale or Offset parameters are changed. Extending is the direction that gives increasing Transducer Counts.

Note: On startup, the Extend Limit defaults to the current position of the axis, so new Extend and Retract Limits must be issued followed by a ’r;P’ command before the axis will move, unless different limits have been saved in the Flash.

C.4.6 Retract Limit

Default: Current position on power-up
Range: Valid 16-bit Position

The Retract Limit specifies the minimum value the motion controller will allow as a position Command Value. (When the Scale is negative, this is the maximum value.) A Command Value below this value will be set to the Retract Limit and will cause the parameter error bit in the Status word to be set. The Retract Limit is given in Position Units.

Note: The Retract Limit must be changed when the Scale or Offset parameters are changed.

Note: On startup, the Retract Limit defaults to the current position of the axis, so new Extend and Retract Limits must be issued followed by a ’r;P’ command before the axis will move, unless different limits have been saved in the Flash.

C.4.7 Comp. Rate

Default: 0
Range: 0 (disabled) or 1 to 65535 position units per second

This parameter is available only on stepper axes. It is one of several parameters that control the compensation feature. For a complete description of the compensation feature, see the Stepper Compensation topic.

This parameter determines the speed at which the axis will try to return to the target position. If the axis is in motion, this rate will be added to the target speed. The compensation rate is only added when the actual position is outside the Compensation or In Position Window as described in the Stepper Compensation topic. To disable compensation, enter a 0 for this parameter.

C.4.8 Comp. Timeout

Default: 0
Range: 0 (no timeout) or 1 to 65535 seconds

This parameter is available only on stepper axes. It is one of several parameters that control the compensation feature. For a complete description of the compensation feature, see the Stepper Compensation topic.

When the axis is stopped outside the In Position Window, compensation will be applied to try to move the axis back to within the window. If the axis is not able to move back into the In Position Window in the time specified by the Compensation Timeout parameter, the Timeout bit will be set in the Status word. This bit can be used to trigger a hard stop on the axis, as controlled by the Auto Stop Mask parameter. Do not use the soft stop with this bit, since the axis is already stopped, so the closed loop stop initiated by the soft stop does nothing.

A Compensation Timeout of 0 will disable the timeout feature so compensation will continue until the position moves within the window.

C.4.9 Steps/Rev

Default: 1
Range: 1 to 65535

This parameter is available on stepper axes only. It is used with the Pos Units/Rev parameter to determine the scaling between position units and steps output by the RMC. These steps can be interpreted by the stepper drive as either a full step or microstep.

The use of this parameter is summarized by the following equation, which is applied every control loop to convert the change in the Target Position to steps that are output.

$$\Delta\text{Steps} = \Delta\text{Target Position} \times \frac{\text{Steps/Rev}}{\text{Pos Units/Rev}}$$

As the name implies, the recommended value to enter in this parameter is the number of steps required to move the motor one revolution. Therefore, for a step motor with 48 steps per revolution and drive with 100 microsteps per step, you would enter a 4800 in this field. However, there may be times when you will not want to enter this value, but instead adjust this field in conjunction with the QUAD CNTS/REV and POS UNITS/REV fields. For a full discussion, including examples, on scaling with stepper modules, see Stepper Scaling.

C.4.10 Pos Units/Rev

Default: 1
Range: 1 to 65535

This parameter is available on stepper axes only. It is used with the Steps/Rev parameter to determine the scaling between position units and steps. It is also used with Quad Cnts/Rev to determine the scaling between position units and quadrature counts. Therefore it is used in the following two conversions:

$$\Delta\text{Steps} = \Delta\text{Target Position} \times \frac{\text{Steps/Rev}}{\text{Pos Units/Rev}}$$

$$\Delta\text{Actual Position} = \Delta\text{Quad Counts} \times \frac{\text{Pos Units/Rev}}{\text{Quad Counts/Rev}}$$

Because this parameter is used in both scales, changing its value will change the effective scale between position units and both quadrature counts and steps. For more information on the quadrature scale, see Quad Cnts/Rev. For more information on the step scale, see Steps/Rev.

As this parameter’s name implies, the intended use of this parameter is to enter the number of position units you wish to have in one revolution. This works well in applications where the stepper, quadrature encoder, and position units are all taken from the same axis and there are a whole number of position units per revolution. The following are some possible values:

Position Units	Pos Units/Rev

Degrees	360
Tenths of a degree	3600
Hundredths of a degree	36000
Thousandths of a rev.	1000
Ten-thousandths of a rev.	10000

For a full discussion, including examples, on scaling with stepper modules, see Stepper Scaling.

C.4.11 SSI Counts/Rev

Default: 1
Range: -32768 to 32767

This parameter is available only on SSI axes with stepper output. It is used with the Pos Units/Rev parameter to determine the scaling between position units and SSI counts. If the SSI feedback is not being used, enter a zero for this parameter. This will make the Actual Position always match the Target Position, therefore eliminating the Following Error.

The sign of this parameter determines whether increasing SSI counts translates to increasing or decreasing the current position:

Sign	Increasing SSI counts means…
+	…increasing positions
-	…decreasing positions

The use of this parameter is summarized by the following equation, which is applied every control loop to convert the incoming quadrature counts to an Actual Position in user-defined position units.

$$\Delta \text{Actual Position} = \Delta \text{SSI Counts} \times \frac{\text{Pos Units/Rev}}{\text{SSICounts/Rev}}$$

As the name implies, the recommended value to enter in this parameter is the number of SSI counts generated in a single revolution of the motor. Therefore, for a 13-bit single-turn encoder mounted on the same shaft as the motor, you would enter 8192 in this parameter since there are 8192 SSI counts per revolution on a 13-line single-turn encoder. However, this becomes more complicated if the SSI feedback is on another shaft that is mechanically geared to the motor shaft with a ratio other than 1:1. For a full discussion on scaling with stepper modules, including examples, see Stepper Scaling.

C.4.12 Max Steps/MSec

Default: 1024
Range: 1 to 1024 steps per millisecond

This parameter is available only on stepper axes. It is used to ensure that the stepper is never driven beyond a given rate. If the axis is requested to give more steps per millisecond than allowed by this parameter, the Overdrive error bit will be set in the Status word. The axis can be set up to automatically stop based on this bit using the Auto Stop Mask parameter.

Following are three cases in which this parameter would be used:

- Limit the step rate to the capability of the motor.
- Limit the step rate to the capability of the system.
- Limit the step rate to a safe speed, especially during startup.

Because this parameter is given in steps per millisecond, this parameter should be set to 1/1000th of the maximum step frequency desired. For example, to limit the output to 100 kHz, enter 100 for this parameter.

C.4.13 Comp. Window

Default: 100
Range: 0 to 65535 position units

This parameter is available only on stepper axes. It is one of several parameters that control the compensation feature. For a complete description of the compensation feature, see the Stepper Compensation topic.

When the axis is not moving, the compensation is applied only when the axis is outside the In Position Window parameter. When the axis is moving, the compensation rate is applied when the actual position differs from the target position by more than the Compensation Window parameter. This window should normally be set to a value greater than the In Position Window, but less than the Following Error Window.

C.4.14 In Position

Default: 50
Range: 0 to 65535

This parameter specifies the size of a window around the Command Position. When the Actual Position gets within this window, the In Position bit is set in the Status word. Notice that the In Position bit is not latched and therefore could go off again if the axis moves back outside the In Position window.

Example:

If an axis Command Position is 10,000 and the In Position parameter is 30, the In Position bit will be set when the axis is stopped and its Actual Position is between 9,971 and 10,029. The bit will be cleared whenever the Actual Position is outside the range.

C.4.15 Following Error

Default: 250
Range: 0 to 65535

The Following Error determines how large the difference between the Target Position and Actual Position can get before the Following Error bit is set in the Status word.

C.4.16 Auto Stop

Default: 0x1FE0 (Soft Stops enabled, Hard Stops enabled for Transducer Errors)
Click here for the Auto Stop Bit Map

The Auto Stop parameter controls the action taken on the rising edge of any of the axis' error bits (the eight (8) most-significant bits in the Status word listed below). The user has control over which error bits cause which levels of stop, or whether an error will cause a stop at all. The default setting of all the AutoStops is Hard Stop, as described below.

During startup and tuning, you will typically need to set some AutoStops to Status Only to keep halts from interfering with the tuning. After completeing the startup procedure, make sure to set the AutoStops to setting that are safe for the machine operation.

Auto Stops are always active unless the axis is in Open Loop, in which case no Auto Stops will be triggered even though error bits may be set.

Note: AutoStops will not occur if the axis is in Open Loop!

The eight (8) most-significant bits in the Status word, as listed below, are the error bits, with the exception of the Home Input status bit for Quadrature axes with Analog (QUAD) or Stepper (STEP) output:

Fault	QUAD	STEP	All Others
7	Encoder Error/Fault	Encoder Error/Fault	No Transducer
6	Extend Limit	Extend Limit	Transducer Noise
5	Retract Limit	Retract Limit	Transducer Overflow
4	Overdrive	Overdrive	Overdrive
3	Parameter Error	Parameter Error	Parameter Error
2	Home Input	Home Input	Pos./Press. Overflow
1	Integrator Windup	Compensation Timeout	Integrator Windup
0	Following Error	Following Error	Following Error

The available actions for each fault are listed below:

- **Status Only**
  The fault will be reflected in its corresponding status bit in the Status word, but no further action will be taken. For some transducer types, this option may not be available for the following faults: No Transducer, Transducer Overflow, and Transducer Noise.
- **Soft Stop**
  If the axis is in closed loop, the fault will trigger ramping the axis to a stop. The speed will ramp

down to zero using the current Deceleration value. If the axis is in Open Loop, the drive will not be affected. If there is an event sequence on the axis, it will stop executing. The Halt status bit will also be set in the Status word. Some faults will use the open loop ramp down although the axis was in closed loop: Position Overflow, No Transducer, Transducer Overflow, Transducer Noise, and Encoder Error/Fault Input. This is done because the position feedback is not dependable and closed loop control cannot be maintained.

- **Hard Stop**
  If the axis is in closed loop, the fault will trigger the drive output to go immediately to 0 mV on analog outputs and no steps for stepper outputs, and the axis will be placed in open loop mode. If the axis is in Open Loop, the drive will not be affected. If there is an event sequence on the axis, it will stop executing. The Halt and Open Loop status bit will also be set in the Status word.
- **Disable Drive**
  The fault will be handled as in a Hard Stop, but additionally the Amp Enable output on QUAD and STEP will be opened. Use the Amp Enable (a) command to close the output again. This option is only available for faults on QUAD and STEP axes.

To view or change the Auto Stop parameter value, open the Auto Stop popup editor in RMCWin using one of the methods described in Using Popup Editors. Select the desired action for each fault type and click OK. Changes to this parameter do not take effect until you issue a Set Parameters (P) command.

Manual Parameter Entry

This parameter can also be edited manually, but this is discouraged since it is much easier to use the popup editor. You can enter hexadecimal numbers by typing a leading 0x, as in 0x1FE0. Each of the eight faults listed above has two bits assigned to it, labeled S and H in the table below:

Bit	Description	Bit	Description
15	Fault 7 - Bit S	7	Fault 7 - Bit H
14	Fault 6 - Bit S	6	Fault 6 - Bit H
13	Fault 5 - Bit S	5	Fault 5 - Bit H
12	Fault 4 - Bit S	4	Fault 4 - Bit H
11	Fault 3 - Bit S	3	Fault 3 - Bit H
10	Fault 2 - Bit S	2	Fault 2 - Bit H
9	Fault 1 - Bit S	1	Fault 1 - Bit H
8	Fault 0 - Bit S	0	Fault 0 - Bit H

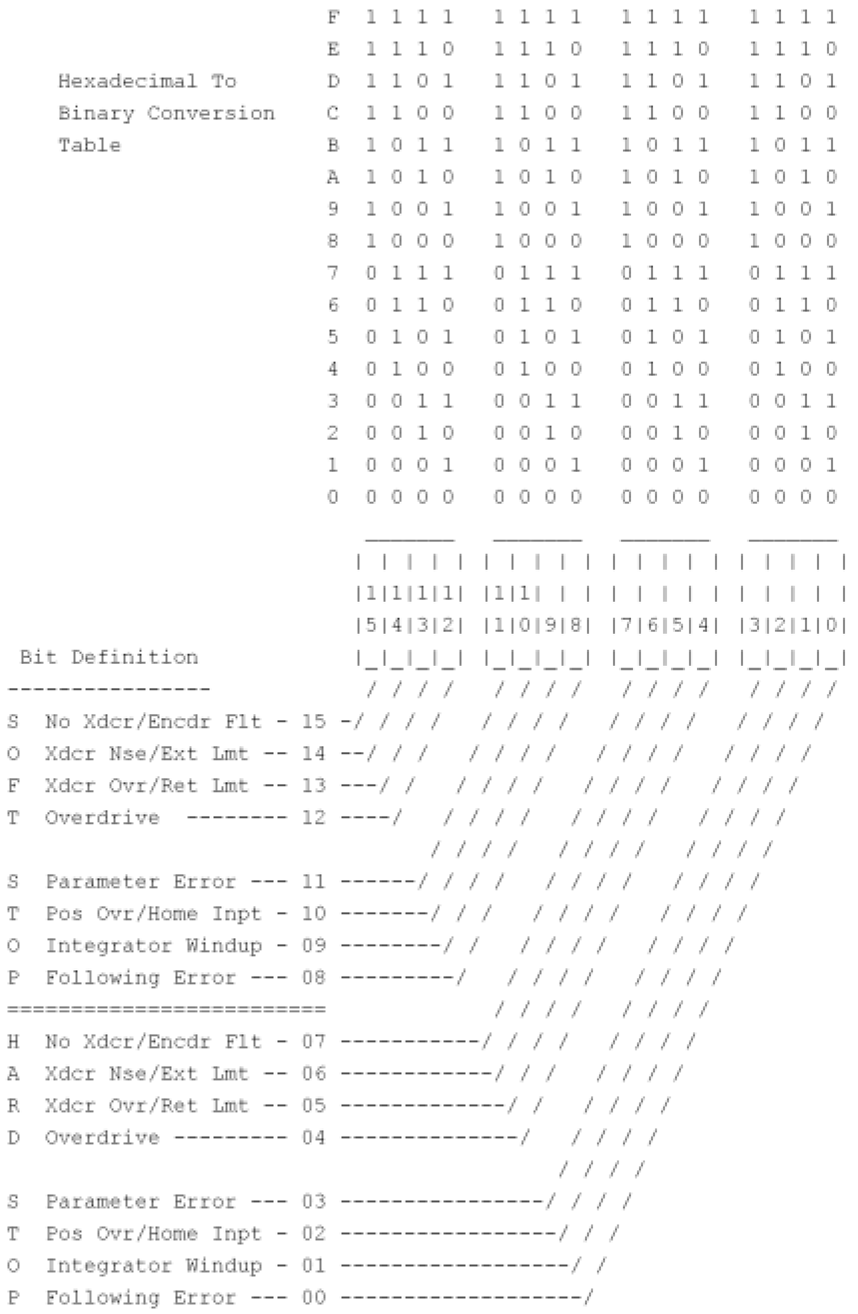
For each fault, the two bits in the Auto Stop parameter define the action as follows:

Bit H	Bit S	Description
0	0	Status Only
0	1	Soft Stop
1	0	Hard Stop
1	1	Disable Drive

If you select Status Only for a fault that cannot use that action, then the axis will use a Soft Stop action for that fault. Similarly, if you select Disable Drive for a fault on an axis that does not have an Amp Enable output, the axis will use the Hard Stop action for the fault.

C.4.17 Auto Stop Bit Map

The table below provides an easy method to convert bit patterns to hexadecimal numbers.



```
                         F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                         E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
    Hexadecimal To       D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
    Binary Conversion    C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
    Table                B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                         A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                         9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                         8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                         7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                         6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                         5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                         4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                         3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                         2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                         1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                         0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0

                            _______   _______   _______   _______
                           | | | | | | | | | | | | | | | | | | | |
                           |1|1|1|1| |1|1| | | | | | | | | | | | |
                           |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
 Bit Definition            |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
----------------            / / / /   / / / /   / / / /   / / / /
S  No Xdcr/Encdr Flt - 15 -/ / / /   / / / /   / / / /   / / / /
O  Xdcr Nse/Ext Lmt -- 14 --/ / /   / / / /   / / / /   / / / /
F  Xdcr Ovr/Ret Lmt -- 13 ---/ /   / / / /   / / / /   / / / /
T  Overdrive  -------- 12 ----/   / / / /   / / / /   / / / /
                                 / / / /   / / / /   / / / /
S  Parameter Error --- 11 ------/ / / /   / / / /   / / / /
T  Pos Ovr/Home Inpt - 10 -------/ / /   / / / /   / / / /
O  Integrator Windup - 09 --------/ /   / / / /   / / / /
P  Following Error --- 08 ---------/   / / / /   / / / /
=========================             / / / /   / / / /
H  No Xdcr/Encdr Flt - 07 -----------/ / / /   / / / /
A  Xdcr Nse/Ext Lmt -- 06 ------------/ / /   / / / /
R  Xdcr Ovr/Ret Lmt -- 05 -------------/ /   / / / /
D  Overdrive --------- 04 --------------/   / / / /
                                           / / / /
S  Parameter Error --- 03 ----------------/ / / /
T  Pos Ovr/Home Inpt - 02 -----------------/ / /
O  Integrator Windup - 01 ------------------/ /
P  Following Error --- 00 -------------------/
```

If both Soft Stop and Hard Stop bits are set for a particular error condition, a Hard Stop will be executed and the Amp Enable output will be opened on QUAD and STEP axes.

C.5 Pressure/Force Parameters

C.5.1 Configuration Word (Pressure)

Default: 0x0000

Six bits of this 16-bit word control the configuration of the module. Bit 0 is the LSB; bit 15 is the MSB.

Click here for the Config Word Bit Map

Bits 12 - 14 - Analog Input Type bits
These bits are used to select the type of analog input being used. Use the following table to select the appropriate input range:

Bit 14	**Bit 13**	**Bit 12**	**Analog Input Type**
0	0	0	Voltage: 0V to +10V
0	0	1	Voltage: -10V to +10V
0	1	0	Voltage: 0V to +5V
0	1	1	Voltage: -5V to +5V
1	0	0	Current: 4mA to 20mA

For a description of the corresponding transducer count ranges, refer to the Counts topic.

Bit 6 - Reverse Drive Mode bit
When this bit is set, the polarity of the axis's output voltage is reversed. Notice that this only affects the analog axis's output. Therefore, for auxiliary pressure or force axis only Open Loop commands are affected by this mode, and position/pressure control is not (recall that position/pressure control uses the drive output of the position axis). However, for pressure-only and force-only control axes, this will reverse the drive output when in Open Loop or pressure control.

This bit is only useful when you are connected to two drives which do not have differential or isolated inputs, and you need one to extend with positive drive and the other to extend with negative drive. (If the drives have differential or isolated inputs, you can just reverse the connections to the drive that must extend with negative drive.) When this bit is set, transducer counts will DECREASE with positive drive.

Warning: This bit must be set properly when the Set Parameters command is issued.

Bit 5 - Absolute Mode bit

Note: This bit is only available on pressure-only and force-only control axes and not on auxiliary pressure and force axes. You can still use absolute mode with auxiliary pressure and force axes, but the Absolute Mode bit of the position axis is used.

When this bit is set on a pressure-only or force-only axis, the drive output is limited between 0 and 10 volts. Therefore, if the drive would have gone negative it will be set to zero and the integrator will be cleared to avoid integrator windup. This is used in situations where either the pressure input is controlled by the RMC but the output is constantly bleeding out, or in situations where the RMC controls the output but the input is constant.

Bits 1 - 2 - Prescale Divisor bits

These bits are used to prescale (reduce the resolution of) the Counts. Use bits 1 and 2 as follows:

Counts ÷	**Bit 2**	**Bit 1**
1	0	0
2	0	1
4	1	0
8	1	1

For example, if bit 1 is set, but bit 2 is cleared, then the transducer counts will be divided by two before being processed. It is more desirable to use these bits to apply the same divide-by on the Scale factor; round off errors are minimized with the divide-by bits.

For example, suppose we calculate that our scale should be 2015.87, but because all parameters must be integers, we enter 2016. We have an error of 0.01%. However we could use a Prescale Divisor of 8 and a Scale of 16127 to give us an effective scale of 16127/8 or 2015.875. The error on our scale is reduced to 0.0002%.

Notice that for differential analog axes, this pre-scale divisor affects both Counts A and B.

Bit 0 - Integrator Limit

When this bit is cleared, the integrator limit is 20% of the maximum drive output. When the bit is set, the limit is 80%. If the integrator tries to go above the limit, the Integrator Windup bit in the Status word is set and the integrator value is held at the limit. The integrator limit is designed to prevent the drive output from saturating. This bit does not take affect until the axis is regulating the pressure because the integrator is not active while in position mode.

C.5.2 Configuration Word (Pressure) Bit Map

The axis Configuration word contains 16 bits of information. The hexadecimal table below provides an easy way to convert hexadecimal numbers to bit patterns.

```
                      F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                      E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
 Hexadecimal To       D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
 Binary Conversion    C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
 Table                B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                      A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                      9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                      8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                      7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                      6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                      5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                      4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                      3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                      2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                      1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                      0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0

                         _______   _______   _______   _______
                        | | | | | | | | | | | | | | | | | | | |
                        |1|1|1|1| |1|1| | | | | | | | | | | | |
                        |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
 Bit Definition         |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
----------------         / / / /   / / / /   / / / /   / / / /
Reserved ---------- 15 -/ / / /   / / / /   / / / /   / / / /
Analog Input Type - 14 --/ / /   / / / /   / / / /   / / / /
Analog Input Type - 13 ---/ /   / / / /   / / / /   / / / /
Analog Input Type - 12 ----/   / / / /   / / / /   / / / /
                              / / / /   / / / /   / / / /
Reserved ---------- 11 ------/ / / /   / / / /   / / / /
Reserved ---------- 10 -------/ / /   / / / /   / / / /
Reserved ---------- 09 --------/ /   / / / /   / / / /
Reserved ---------- 08 ---------/   / / / /   / / / /
                                   / / / /   / / / /
Reserved ---------- 07 -----------/ / / /   / / / /
Reverse Drive ----- 06 ------------/ / /   / / / /
Absolute Mode ----- 05 -------------/ /   / / / /
Reserved ---------- 04 --------------/   / / / /
                                        / / / /
Reserved ---------- 03 ----------------/ / / /
Prescale Bits ----- 02 -----------------/ / /
Prescale Bits ----- 01 ------------------/ /
Integrator Limit -- 00 -------------------/
```

C.5.3 Configuration Bits - Analog Specific

Analog Input Type - Bits 12-14

Use the following table to select the appropriate input range:

14	**13**	**12**	**Analog Input Type**

0	0	0	Voltage: 0V to +10V
0	0	1	Voltage: -10V to +10V
0	1	0	Voltage: 0V to +5V
0	1	1	Voltage: -5V to +5V
1	0	0	Current: 4mA to 20mA

For a description of the corresponding transducer count ranges, refer to the Counts topic.

Unused - Bits 7 and 15
These bits are reserved for future use and should be cleared on analog axes.

C.5.4 Scale A, Scale B (Pressure)

Default: 32767 for 16-bit, 4096 for 12-bit
Range: -32768 to 32767

TIP: On the Tools menu, click Scale/Offset Calibration to help calculate Scale and Offset.

These two parameters have slightly different uses when using the axis as a differential force versus when the axis is used for single-ended pressure.

For single-ended pressure, Scale A is used to calculate the Actual Pressure in pressure units from Counts A. Scale B is unused.

For differential force, Scale A is used to calculate Actual Force A in force units from Counts A, and Scale B is used to calculate Actual Force B in force units from Counts B. Actual Force B is then subtracted from Actual Force A to give the Actual Force, which is displayed in the Actual Pressure status field.

Note: When a scale is set to 0, it is treated as though it were 32768. This allows a one-to-one relationship between counts and pressure or force units. Similarly, a Scale value of -32768 represents a negative one-to-one relationship between counts and position units (increasing counts mean decreasing pressure or force units).

Note: In RMC100 CPU firmware prior to 19991216, these scales could not be set to 0 or -32768.

Scale is defined as 32768 times the Prescale Divisor times the number of pressure units for pressure axes or force units for differential force axes per count:

$$\text{Scale} = \frac{(P_0 - P_1)}{(C_0 - C_1)} \times 32768 \times \text{Prescale Divisor}$$

Where P0 and P1 are two pressures for forces and C0 and C1 are the corresponding counts at those pressures/forces. The Prescale Divisor is specified in the Configuration Word.

In pressure and force applications, Scale can be used for two purposes:

1. Transducer Compensation: The Scale parameter compensates for differences in analog transducers. Each transducer will indicate a different current or voltage level for the same pressure. Usually, only a small change from the default value is necessary to compensate the transducer.

2. Pressure Translation: A second and more useful feature of Scale is its ability to translate a fixed physical reading to usable pressure units (pressure units from thousandths to hundredths or vice versa).

   Click here for Pressure/Force Scale and Offset examples

C.5.5 Offset A, Offset B (Pressure)

Default: 0
Range: -32768 to 32767

TIP: On the Tools menu, click Scale/Offset Calibration to help calculate Scales and Offsets.

The Offset fields have different uses depending on whether the axis will be used for differential force or single-ended pressure.

For single-ended pressure, Offset A is used with Scale A to convert Counts A to Actual Pressure; it shifts the Actual Pressure with respect to the analog input zero. Offset B is unused.

For differential force, Offset A is used with Scale A to convert Counts A to Actual Force A while Offset B is used with Scale B to convert Counts B to Actual Force B. Both offset the force units by their amount.

These offsets can be defined in either of the following ways:

$$\text{Offset} = P_0 - \frac{(P_0 - P_1) \times C_0}{(C_0 - C_1)}$$

$$\text{Offset} = P_0 - \frac{\text{Scale} \times C_0}{32768 \times \text{Prescale}}$$

In the above equation, P0 and P1 are two pressure or force values in the desired pressure or force units, and C0 and C1 are the corresponding counts given in the Counts A or B field.

Click here for Pressure/Force Scale and Offset examples

Why Bother?
These offsets can be used to set your 0 point at any pressure reading. They are often left at 0 for voltage feedback gauges (e.g. 0 to 10V or 0 to 5V), but will likely need to be non-zero for current (4 to 20mA) gauges. This is because zero pressure or force units will have zero counts on a voltage gauge, but will have 6500 counts (4mA) on a current gauge.

C.5.6 Pressure/Force Scale and Offset Calculation Examples

Example 1

Suppose a pressure transducer gives an output from 0 to 10 volts. This range is represented by Counts A from 0 to 32500. Also suppose that the range of pressures which will be used are from 0 psi to 2000 psi, and 0 psi gave a reading of 12 counts and 2000 psi gave a reading of 32522 counts.

$$\text{Scale A} = \frac{(2000-0)\times 32768}{(32522-12)} = 2015.87$$

Since the scale is less than 16383.5, use the Prescale Divisor bits in the Configuration Word to divide the transducer counts by 8. Then multiply the scale by 8 to obtain:

$$\text{Scale A} = 2015.87\times 8 = 16127$$

This gives us an effective scale of 16127/8 or 2015.875, which is closer than we would have been if we did not use the Prescale Divisor and instead used a scale of 2016.

An alternative way of doing this without actually measuring the pressure independently is to trust the gauge. That is, assume that 0 psi will give 0 counts, and 2000 psi will give 32,500 counts. Using these numbers, we get the following equation:

$$\text{Scale A} = \frac{(2000-0)\times 32768}{(32500-0)} = 2016.49$$

Again, we will apply a Prescale Divisor of 8 to obtain:

$$\text{Scale A} = 2016.49\times 8 = 16132$$

Next, we must calculate the offset. We use the following equation:

$$\text{Offset A} = F_0 - \frac{C_0\times \text{Scale}}{32768\times \text{PreScale}}$$

Therefore, by using the first point (0 psi at 0 counts or 0mV) and the second method of calculating the scale, we get the following:

$$\text{Offset A} = 0 - \frac{0\times 16132}{32768\times 8}$$
$$\text{Offset A} = 0$$

As you can see, the offset for gauges that give 0 volts at 0 pressure units will always have 0 for an offset.

Example 2

Suppose a pressure transducer gives an output from 4 to 20mA. This range is represented by Counts from 6500 to 32500. Also suppose that the range of pressures which will be used are from 0 to 10 bars, and 0.0 bars read 6487 counts and 10.0 bars read 32662 counts. We must choose our position units first. It is not a good idea to just use bars, as the user will only be able to see 0 to 10 bars and no fractions. Therefore, it makes more sense to use millibars as position units and go from 0 to 10,000 millibars. The scale calculation goes as follows:

$$\text{Scale A} = \frac{(10000-0)\times 32768}{(32662-6487)} = 12518.82$$

Again, since the scale is less than 16383.5, use the Prescale Divisor bits in the Configuration Word to divide the transducer counts by 2. Then multiply the scale by 2 to obtain:

$$\text{Scale A} = 12518.82 \times 2 = 25038$$

Assuming we cannot or do not want to verify the gauge's accuracy, we can use the specified gauge requirements. Therefore, we assume that 0.0 bars reads 6500 counts, and 10.0 bars reads 32500 counts. Using these numbers, we get the following equation:

$$\text{Scale A} = \frac{(10000-0)\times 32768}{(32500-6500)} = 12603.08$$

Again, we will apply a Prescale Divisor of 2 to obtain:

$$\text{Scale A} = 12603.08 \times 2 = 25206$$

Next, we must calculate the offset. We use the following equation:

$$\text{Offset A} = F_0 - \frac{C_0 \times \text{Scale}}{32768 \times \text{PreScale}}$$

Therefore, by using the first point (0 psi at 6500 counts or 4mA) and the second method of calculating the scale, we get the following:

$$\text{Offset A} = 0 - \frac{6500 \times 25206}{32768 \times 2}$$

$$\text{Offset A} = -2500$$

Example 3

Suppose we want to control differential force. We use the same gauges on both the A and B sides of the cylinder. These gauges are rated to have 4mA at 0 psi and 20mA at 7500 psi. Therefore, each channel will have 6500 counts at 0 psi and 32500 counts at 7500 psi. The cylinder has an internal diameter of 6 inches. The rod has an outside diameter of 2 inches. Therefore we find the maximum force applied on each end:

Force = Pressure x Area

Force on A Side = Pressure x Cross Section of Cylinder
Force on A Side at 20mA = 7500 psi x (p x 3 inches x 3 inches)
Force on A Side at 20mA = 212,057.5 pounds

Force on B Side = Pressure x (Cross Section of Cylinder - Cross Section of Rod)
Force on B Side at 20mA = 7500 psi x (p x 3 inches x 3 inches - p x 1 inch x 1 inch)
Force on B Side at 20mA = 188,495.6 pounds

We assume that we will use forces up to 150,000 pounds. Because the RMC is limited in force units from ±32,767, we cannot use pounds as our force units, but instead must use 10-pound units. Therefore, 150,000 pounds will be represented as 15,000 force units.

Using this assumption, can calculate the scale.

$$\text{Scale} = \frac{(F_1 - F_0) \times 32768}{(C_1 - C_0)}$$

$$\text{Scale A} = \frac{(21{,}205.75 - 0) \times 32768}{(32500 - 6500)} = 26725.76$$

$$\text{Scale B} = \frac{(18{,}849.56 - 0) \times 32768}{(32500 - 6500)} = 23756.24$$

Both are above 16383.5, so our Prescale Divisor must be 1 (the Prescale Divisor bits will both be 0). Therefore, our scales are as follows:

$$\text{Scale A} = 26726$$

$$\text{Scale B} = 23756$$

Next, we must calculate the offsets for both channels. We use the following equation:

$$\text{Offset} = F_0 - \frac{C_0 \times \text{Scale}}{32768 \times \text{PreScale}}$$

Therefore, by using the first point (0 ten-pound units at 6500 counts or 4mA) for both channels, we get the following:

$$\text{Offset A} = 0 - \frac{6500 \times 26726}{32768 \times 1}$$

$$\text{Offset A} = -5301$$

$$\text{Offset B} = 0 - \frac{6500 \times 23756}{32768 \times 1}$$

$$\text{Offset B} = -4712$$

C.5.7 Proportional Gain (Pressure)

Default: 1
Range: -32768 to 32767

Note: Use positive Feed Forward and Gain values if the pressure increases in the extend direction, and negative values if the pressure increases in the retract direction.

The Proportional Gain controls how much drive is generated proportional to the Pressure Error. The Pressure Error is defined as the Target Pressure minus the Actual Pressure. The units on the Proportional Gain is millivolts per 10 units of Pressure Error.

The Proportional Drive is defined as follows:

$$\text{ProDrive (mV)} = \text{ProGain (mV/10pu)} \times \text{PressError (pu)}$$

Or more simply:

$$\text{ProDrive (mV)} = \frac{\text{ProGain (mV/pu)} \times \text{PressError (pu)}}{10}$$

where pu is pressure units.

CAUTION: Increase the Proportional Gain gradually. Excessive gain can cause oscillation that could cause damage or injury.

Think about this:

Internally, the motion controller must compare the error between the Target and Actual Pressures with error limits to keep values from overflowing. The error limit is the error at which full drive (10 volts) will occur. This internal error limit is calculated as follows:

Error Limit = 1,000,000 / Proportional Gain

Therefore, if the Gain is set to 100, the Error Limit will be 10,000, which means any error greater than 10,000 will be treated as an error of 10,000 and the Overdrive error bit will be set in the Status word.

C.5.8 Integral Gain (Pressure)

Default: 1
Range: -32768 to 32767

Note: Use positive Feed Forward and Gain values if the pressure increases in the extend direction, and negative values if the pressure increases in the retract direction.

The Integral Gain is used to control the amount of drive provided by the integrator. The integrator adds the pressure error to an accumulator every millisecond. The Integral Gain should be adjusted after the Feed Forwards have been set to optimal values. Using the integrator before the feed forwards have been set properly will cause the system to overshoot the target pressure. We recommend that you set the Integral Gain to a value of at least 50.

Integral Gain is defined as:

Integral Gain = 0.1 mV per 1024 counts of accumulated Pressure Error

Integral Drive is defined as:

Integral Drive = Integral Gain x Accumulated Counts

Why Bother?
Integral Gain should be used to compensate for the fact that loads may vary, valves are non-linear and the axis may have trouble getting to the Command Pressure without Integral Gain.

C.5.9 Differential Gain (Pressure)

Default: 0
Range: -32768 to 32767

Note: Use positive Feed Forward and Gain values if the pressure increases in the extend direction, and negative values if the pressure increases in the retract direction.

The Differential Gain field is used to minimize the error between the Target Pressure and the Actual Pressure. The change in error is multiplied by the Differential Gain value to get the differentiator drive term. Differential Gain is defined as:

Differential Gain = 0.1 millivolts per change in Pressure Error

Differential Drive is defined as:

Differential Drive = (E0 - E1) x Differential Gain

where E0 is the pressure error (Target Pressure - Actual Pressure) in the first time period and E1 is the pressure error in the second time period. (E0 - E1) is the change in pressure error between the two time periods.

CAUTION: To avoid oscillation during initial tuning, start with values below 10.

Why Bother?
The differentiator field should usually be set to a value of 0. This is because analog input signal noise causes the speed calculations to be noisy.

C.5.10 Extend Feed Forward (Pressure)

Default: 100
Range: -32768 to 32767

Note: Use positive Feed Forward and Gain values if the pressure increases in the extend direction, and negative values if the pressure increases in the retract direction.

Feed Forward is an open loop compensation that is proportional to the rate the Target Pressure is changing. This value is expressed in terms of millivolts per 1,000 Pressure Units per second. Extend Feed Forward drive is added to the output only when the axis is extending. The drive output provided by the Extend Feed Forward is determined as follows:

$$\text{Feed Forward Drive} = \frac{\text{Extend Feed Forward} \times \Delta\text{Target Pressure}}{2000}$$

To set the Feed Forward parameters, try ramping the pressure with very small gains. If the axis lags after this parameter has been set, the feed forward is too small or the system response is too slow. If the axis leads, the feed forward is too large or the system response is too slow.

Note: Because pressure does not necessarily react linearly to the drive output (that is, the same amount of drive does not increase the pressure by the same amount at all positions), it is important to tune the Feed Forwards at the particular pressure ramp that will be used. If you will be ramping the pressure multiple times, you may want to use the 0xD8 and 0xD9 commands to set these parameters between pressure ramps.

Think about this:

On hydraulic systems, the extend and retract feed forward terms will be different by the ratio of the extend and retract surface areas of the position.

C.5.11 Retract Feed Forward (Pressure)

Default: 100
Range: -32768 to 32767

Note: Use positive Feed Forward and Gain values if the pressure increases in the extend direction, and negative values if the pressure increases in the retract direction.

Same as Extend Feed Forward, except it is used when retracting.

Note: Retracting is the direction that returns decreasing Transducer Counts.

C.5.12 Integrator Preload (Pressure)

Default: 0
Range: -32768 to 32767 (millivolts of drive)

The value of this parameter represents millivolts of drive that will be placed in the integral drive term at the time when the pressure axis begins controlling pressure. Some applications require this additional drive (either positive or negative) immediately on the transition.

Note: If this term is non-zero, then it is important that the Integral Gain also be non-zero. Otherwise, the initial integral drive will never return to zero and the pressure may never reach the requested pressure.

C.5.13 Filter Time Constant (Pressure/Force)

Default: 0 (disabled)
Range: 1 to 65,535 milliseconds, or 0 to disable

Note: This parameter was introduced in RMC CPU firmware dated 20020429 or later. This parameter is reserved in earlier firmware.

This parameter allows filtering the pressure/force feedback value. By default the filter is disabled. If this parameter is set to a non-zero value, then the pressure/force feedback is filtered using an Infinite Impulse Response (IIR) filter with a time constant set to this parameter's value in milliseconds.

This feature should be left disabled in systems that need to react to quick changes in pressure, as increasing this filter value will increase the phase delay in the system feedback, making the system more likely to oscillate. Systems with slower acting pressures can benefit from applying this filter.

To compute an initial value for your time constant value, recall that for a step jump in the feedback value, an IIR filter will reach 63% in 1 time constant, and 99% in 5 time constants.

C.5.14 Drive Transfer Percent

Default: 0
Range: -500 to 500

This parameter affects the integral drive at the point when an axis transitions from position control to pressure control. The drive at that point is multiplied by this parameter and divided by 100, and this value is placed into the integral drive. The effect is that some controlled portion of the drive is carried forward from position control into pressure control.

Note: If this term is non-zero, then it is important that the Integral Gain also be non-zero. Otherwise, the initial integral drive will never return to zero and the pressure may never reach the requested pressure.

Why Bother?

There are two reasons for using this parameter to control pressure. First, a negative value can be entered to apply extra braking power at the time pressure mode is entered. This is desired in applications where momentum tends to cause the pressure to pass its command value even though no additional drive is applied. By applying negative drive, the momentum can be taken away.

Second, some users desire a bumpless transfer. A bumpless transfer is one in which the drive is kept the same on the transition from position to pressure control and therefore the mass being moved does not jerk on the transition. To achieve this affect, one would use a value in Drive Transfer Percent near 100.

TIP: In many applications the same effect of a bumpless transfer can be obtained with better results by using appropriate Extend and Retract Feed Forward values.

C.5.15 At Pressure

Default: 50
Range: 0 to 65535

At Pressure specifies the size of a window around the Command Pressure. When the Actual Pressure gets within this window, the At Pressure bit is set (but not latched) in the Status word.

The window around the Command Pressure and the Actual Pressure must be less than the At Pressure value. Therefore, an AT PRESSURE value of 0 will prohibit the At Pressure bit in the Status word from being set.

Example:

If an axis Command Pressure is 2,000 and the AT PRESSURE parameter is 30, the At Pressure bit will be set when the Actual Pressure is between 1,971 and 2,029. The bit will be cleared whenever the Actual Pressure is outside the range.

C.5.16 Pressure Window

Default: 250
Range: 0 to 65535

The PRESSURE WINDOW determines how large the difference between the Target Pressure and Actual Pressure can get before the Following Error bit is set in the Status word.

C.5.17 Auto Stop

Default: 0x1FE0 (Soft Stops enabled, Hard Stops enabled for Transducer Errors)
Click here for the Auto Stop Bit Map

The Auto Stop parameter controls the action taken on the rising edge of any of the axis' error bits (the eight (8) most-significant bits in the Status word listed below). The user has control over which error bits cause which levels of stop, or whether an error will cause a stop at all. The default setting of all the AutoStops is Hard Stop, as described below.

During startup and tuning, you will typically need to set some AutoStops to Status Only to keep halts from interfering with the tuning. After completeing the startup procedure, make sure to set the AutoStops to setting that are safe for the machine operation.

Auto Stops are always active unless the axis is in Open Loop, in which case no Auto Stops will be triggered even though error bits may be set.

Note: AutoStops will not occur if the axis is in Open Loop!

The eight (8) most-significant bits in the Status word, as listed below, are the error bits, with the exception of the Home Input status bit for Quadrature axes with Analog (QUAD) or Stepper (STEP) output:

Fault	QUAD	STEP	All Others

7	Encoder Error/Fault	Encoder Error/Fault	No Transducer
6	Extend Limit	Extend Limit	Transducer Noise
5	Retract Limit	Retract Limit	Transducer Overflow
4	Overdrive	Overdrive	Overdrive
3	Parameter Error	Parameter Error	Parameter Error
2	Home Input	Home Input	Pos./Press. Overflow
1	Integrator Windup	Compensation Timeout	Integrator Windup
0	Following Error	Following Error	Following Error

The available actions for each fault are listed below:

- **Status Only**
  The fault will be reflected in its corresponding status bit in the Status word, but no further action will be taken. For some transducer types, this option may not be available for the following faults: No Transducer, Transducer Overflow, and Transducer Noise.
- **Soft Stop**
  If the axis is in closed loop, the fault will trigger ramping the axis to a stop. The speed will ramp down to zero using the current Deceleration value. If the axis is in Open Loop, the drive will not be affected. If there is an event sequence on the axis, it will stop executing. The Halt status bit will also be set in the Status word. Some faults will use the open loop ramp down although the axis was in closed loop: Position Overflow, No Transducer, Transducer Overflow, Transducer Noise, and Encoder Error/Fault Input. This is done because the position feedback is not dependable and closed loop control cannot be maintained.
- **Hard Stop**
  If the axis is in closed loop, the fault will trigger the drive output to go immediately to 0 mV on analog outputs and no steps for stepper outputs, and the axis will be placed in open loop mode. If the axis is in Open Loop, the drive will not be affected. If there is an event sequence on the axis, it will stop executing. The Halt and Open Loop status bit will also be set in the Status word.
- **Disable Drive**
  The fault will be handled as in a Hard Stop, but additionally the Amp Enable output on QUAD and STEP will be opened. Use the Amp Enable (a) command to close the output again. This option is only available for faults on QUAD and STEP axes.

To view or change the Auto Stop parameter value, open the Auto Stop popup editor in RMCWin using one of the methods described in Using Popup Editors. Select the desired action for each fault type and click OK. Changes to this parameter do not take effect until you issue a Set Parameters (P) command.

Manual Parameter Entry

This parameter can also be edited manually, but this is discouraged since it is much easier to use the popup editor. You can enter hexadecimal numbers by typing a leading 0x, as in 0x1FE0. Each of the eight faults listed above has two bits assigned to it, labeled S and H in the table below:

Bit	Description	Bit	Description
15	Fault 7 - Bit S	7	Fault 7 - Bit H
14	Fault 6 - Bit S	6	Fault 6 - Bit H
13	Fault 5 - Bit	5	Fault 5 - Bit H

	S		
12	Fault 4 - Bit S	4	Fault 4 - Bit H
11	Fault 3 - Bit S	3	Fault 3 - Bit H
10	Fault 2 - Bit S	2	Fault 2 - Bit H
9	Fault 1 - Bit S	1	Fault 1 - Bit H
8	Fault 0 - Bit S	0	Fault 0 - Bit H

For each fault, the two bits in the Auto Stop parameter define the action as follows:

Bit H	**Bit S**	**Description**
0	0	Status Only
0	1	Soft Stop
1	0	Hard Stop
1	1	Disable Drive

If you select Status Only for a fault that cannot use that action, then the axis will use a Soft Stop action for that fault. Similarly, if you select Disable Drive for a fault on an axis that does not have an Amp Enable output, the axis will use the Hard Stop action for the fault.

C.5.18 Auto Stop Bit Map

The table below provides an easy method to convert bit patterns to hexadecimal numbers.

```
                         F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                         E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
    Hexadecimal To       D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
    Binary Conversion    C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
    Table                B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                         A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                         9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                         8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                         7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                         6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                         5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                         4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                         3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                         2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                         1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                         0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0

                            _______   _______   _______   _______
                           | | | | | | | | | | | | | | | | | | | |
                           |1|1|1|1| |1|1| | | | | | | | | | | | |
                           |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
 Bit Definition            |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
----------------            / / / /   / / / /   / / / /   / / / /
S  No Xdcr/Encdr Flt - 15 -/ / / /   / / / /   / / / /   / / / /
O  Xdcr Nse/Ext Lmt -- 14 --/ / /   / / / /   / / / /   / / / /
F  Xdcr Ovr/Ret Lmt -- 13 ---/ /   / / / /   / / / /   / / / /
T  Overdrive  -------- 12 ----/   / / / /   / / / /   / / / /
                                 / / / /   / / / /   / / / /
S  Parameter Error --- 11 ------/ / / /   / / / /   / / / /
T  Pos Ovr/Home Inpt - 10 -------/ / /   / / / /   / / / /
O  Integrator Windup - 09 --------/ /   / / / /   / / / /
P  Following Error --- 08 ---------/   / / / /   / / / /
=========================             / / / /   / / / /
H  No Xdcr/Encdr Flt - 07 -----------/ / / /   / / / /
A  Xdcr Nse/Ext Lmt -- 06 ------------/ / /   / / / /
R  Xdcr Ovr/Ret Lmt -- 05 -------------/ /   / / / /
D  Overdrive --------- 04 --------------/   / / / /
                                           / / / /
S  Parameter Error --- 03 ----------------/ / / /
T  Pos Ovr/Home Inpt - 02 -----------------/ / /
O  Integrator Windup - 01 ------------------/ /
P  Following Error --- 00 -------------------/
```

If both Soft Stop and Hard Stop bits are set for a particular error condition, a Hard Stop will be executed and the Amp Enable output will be opened on QUAD and STEP axes.

C.6 Analog Reference Parameters

C.6.1 Configuration Word

Default: 0x0000

This 16-bit word controls the configuration of the module. Bit 0 is the LSB; bit 15 is the MSB.

Click here for the Config Word Bit Map

Bits 7, 12-15 - Transducer Type bits
These bits are used differently depending on the transducer type used. Refer to one of the following topics for details on these four bits:

- Analog Transducer bits
- Magnetostrictive Displacement Transducer (MDT) bits
- Quadrature/Stepper Transducer bits
- Synchronous Serial Interface (SSI) bits
- Resolver Transducer Bits

Bit 11 - Pressure Assign bit
This bit is used only if the RMC has one or more auxiliary pressure or force channels. If such channels are installed, then this bit can be set to indicate that this position axis has an assigned pressure axis. Bits 8 and 9 are used to define the pressure axis assigned.

Bit 10 - Auto Home Re-arm
This bit is for Quadrature axes only. When it is set, the axis will automatically be re-armed after homing. This requires the axis to first be armed using the Arm Home Command. If this bit is cleared after homing, the axis will still be armed. See the Homing a Quadrature Axis topic for details.

Bits 8 - 9 - Pressure Axis Select bits
These bits are used only if the RMC has one or more auxiliary pressure or force channels and bit 11 is set to indicate that this axis has one of the pressure or force channels assigned to it. These bits then select between up to four auxiliary pressure/force channels:

Bit 9	**Bit 8**	
0	0	The first auxiliary pressure/force axis is assigned.
0	1	The second auxiliary pressure/force axis is assigned.
1	0	The third auxiliary pressure/force axis is assigned.

1	1	The fourth auxiliary pressure/force axis is assigned.

It is necessary to assign a pressure axis to a position axis in order to switch between position and pressure control. Refer to Using an Analog Channel as a Pressure Axis for details.

Note: Two position axes cannot be assigned to the same pressure axis. If this does occur, only the first axis to make the assignment followed by the Set Parameters (P) command will succeed in being assigned. Any later attempt to assign the pressure axis will fail with a parameter error.

Bit 6 - Reverse Drive Mode bit

Warning: This bit must be set properly when the Set Parameters command is issued.

This bit determines the relationship between the RMC's position direction and the drive output direction.

For analog-output axes, setting this bit reverses the polarity of the axis output voltage. Although in many cases the differential drive wires can be swapped on the drive unit, this is not always possible or it may simply be easier to use this bit.

Bit 6	
0	The RMC will generate positive drive to increase transducer counts.
1	The RMC will generate negative drive to increase transducer counts.

For stepper-output axes, this bit is used to change the meaning of the Direction output with relation to the axis transducer counts.

Bit 6	
0	The RMC will use a low Direction output to increase transducer counts.
1	The RMC will use a high Direction output to increase transducer counts.

Bit 5 - Absolute Mode bit

This mode is intended for use with a two-valve system, one controlling the flow rate and the other the direction. This axis controls the flow-rate valve. The directional valve must be controlled by other means. In this mode, the axis generates a positive drive output regardless of the direction of the move. The drive will not go negative if the motion controller overshoots the target. This is useful for some injection and blow-molding applications.

When the axis is stopped, you must go into open loop mode.

Warning: This bit must be set properly when the Set Parameters command is issued.

Bit 4 - Continue Mode bit

This bit affects what happens when the module loses contact with the Programmable Controller. When this bit is set, the module will finish any move it has started, otherwise it will halt immediately upon detecting loss of contact with the PLC. This is useful for finishing a move that must complete to prevent machine downtime (for example, a partial shot in an injection-molding machine).

Bit 3 - Simulate bit

When this bit is set, the drive output is set to zero and the magnetostrictive transducer inputs are ignored. Internally the Target Position is used as the Actual Position. This mode is used for debugging. (The transducer error bits and LEDS will be cleared.)

Note: Drive output is always 0 volts while in simulate mode; no commands, including open loop, will change this.

Bits 1 - 2 - Prescale Divisor Bits

Prescaling may be desired on systems with long strokes. The prescale bits are used along with the Scale parameter to convert Transducer Counts to Actual Position. The effect is to divide the Scale by 1, 2, 4, or 8. The divide-by factor is specified as follows:

Counts ÷	Bit 2	Bit 1
1	0	0
2	0	1
4	1	0
8	1	1

The advantage of these bits is that they let you maintain a higher resolution in the Scale parameter. For example, a Scale of 31027 and a divide-by of 4 gives an effective scale of 7756.75. Without using the divide-by bits, the Scale would have to be rounded up to 7757, which has less resolution—with a 64" rod, the rounded value would make a 0.002" error at 64". Note: The stroke is always limited to 220 inches due to the 18-bit transducer counter resolution.

These bits must be zero for Resolver modules.

Bit 0 - Integrator Limit

When this bit is cleared, the integrator limit is 20% of full drive. When the bit is set, the limit is 80%. If the integrator tries to go above the limit, the Integrator Windup bit in the status word is set and the integrator value is held at the limit. The integrator limit is designed to prevent the drive output from saturating.

Note: The actual drive output may be reduced based on the values of the Extend Feed Forward and Retract Feed Forward .

C.6.2 Configuration Word Bit Map

The axis Configuration word contains 16 bits of information. The hexadecimal table below provides an easy way to convert hexadecimal numbers to bit patterns.

```
                      F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                      E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
 Hexadecimal To       D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
 Binary Conversion    C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
 Table                B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                      A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                      9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                      8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                      7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                      6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                      5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                      4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                      3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                      2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                      1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                      0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0

                         _______   _______   _______   _______
                        | | | | | | | | | | | | | | | | | | | |
                        |1|1|1|1| |1|1| | | | | | | | | | | | |
                        |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
 Bit Definition         |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
----------------         / / / /   / / / /   / / / /   / / / /
Transducer Type --- 15 -/ / / /   / / / /   / / / /   / / / /
Transducer Type --- 14 --/ / /   / / / /   / / / /   / / / /
Transducer Type --- 13 ---/ /   / / / /   / / / /   / / / /
Transducer Type --- 12 ----/   / / / /   / / / /   / / / /
                              / / / /   / / / /   / / / /
Pressure Assign --- 11 ------/ / / /   / / / /   / / / /
Reserved ---------- 10 -------/ / /   / / / /   / / / /
Pressure Axis Sel - 09 --------/ /   / / / /   / / / /
Pressure Axis Sel - 08 ---------/   / / / /   / / / /
                                   / / / /   / / / /
Transducer Type --- 07 -----------/ / / /   / / / /
Reverse Drive ----- 06 ------------/ / /   / / / /
Absolute Mode ----- 05 -------------/ /   / / / /
Continue Mode ----- 04 --------------/   / / / /
                                        / / / /
Simulate Mode ----- 03 ----------------/ / / /
Prescale Bits ----- 02 -----------------/ / /
Prescale Bits ----- 01 ------------------/ /
Integrator Limit -- 00 -------------------/
```

C.6.3 Configuration Bits - Analog Specific

Analog Input Type - Bits 12-14

Use the following table to select the appropriate input range:

14	13	12	Analog Input Type

0	0	0	Voltage: 0V to +10V
0	0	1	Voltage: -10V to +10V
0	1	0	Voltage: 0V to +5V
0	1	1	Voltage: -5V to +5V
1	0	0	Current: 4mA to 20mA

For a description of the corresponding transducer count ranges, refer to the Counts topic.

Unused - Bits 7 and 15
These bits are reserved for future use and should be cleared on analog axes.

C.6.4 Scale

Default: Transducer specific
Range: -32768 to 32767

This scale is used on MDT, SSI, Analog position axes with the Offset parameter and the Prescale Divisor bits of the Config word to convert the Transducer Counts to an Actual Position. Quadrature axes use this parameter with the Prescale Divisor bits to convert from the change in Transducer Counts to a change in the Actual Position. Resolver axes use this parameter together with the Count Offset to calculate the Actual Position.

For details on converting counts to position units see the appropriate topics for your transducer types:

- Analog Scaling
- MDT Scaling
- Quadrature Scaling
- SSI Scaling
- Stepper Scaling
- Resolver Scaling

Note: When the Scale is set to 0, it is treated as though the Scale is set to 32768. This allows a one-to-one relationship between counts and position units. Similarly, a Scale value of -32768 represents a negative one-to-one relationship between counts and position units (increasing counts mean decreasing position units).

Note: You should never set the Scale below 16383 (except for resolver axes). If it does go below that value, then use the Prescale Divisor bits in the Configuration Word. For details, refer to the discussion on scaling for your transducer type.

Note: In RMC100 CPU firmware prior to 19991216, Scale could not be 0 or -32768 for MDT, SSI, and analog transducers.

C.6.5 Offset

Default: 0
Range: -65536 to 65535

This parameter is available on all axis types except those with quadrature or stepper feedback. Quadrature and stepper axes use the Coordinate Limit parameter in place of the Offset parameter.

The Offset parameter is used for the following two purposes:

- Translating transducer counts to actual position in user-defined position units.
- Defining the 16-bit position range.

For a complete discussion on the use of this parameter, select the topics that apply to the transducer types you use:

- Analog Scaling
- MDT Scaling
- SSI Scaling

What if it is displayed under RMCWin incorrectly?
The Offset parameter may be displayed incorrectly on the RMCWin main screen in some circumstances. This is because the offset specifies the zero location of the axis. The numbers are not necessarily incorrect (the module still functions correctly), but they may not look right.

To correct the problem:

1. Right-click the Offset parameter and choose **Toggle Offset Sign**.
2. You can now save the parameters to a file (on the File menu, click Save). Whenever you start RMCWin, if you read the parameters from the RMC, the problem should be corrected, as long as the Offset did not change in the RMC.

C.6.6 Extend Limit

Default: Current position on power-up
Range: Valid 16-bit Position

The Extend Limit specifies the maximum requested position value that the RMC will allow as a Command Value. (When the Scale is negative, this is the minimum value.) A Command Value that exceeds this value will be set to the Extend Limit, and the Parameter Error bit in the Status word will be set. The Extend Limit is given in Position Units.

Note: The Extend Limit must be changed when the Scale or Offset parameters are changed. Extending is the direction that gives increasing Transducer Counts.

Note: On startup, the Extend Limit defaults to the current position of the axis, so new Extend and Retract Limits must be issued followed by a 'r;P' command before the axis will move, unless different limits have been saved in the Flash.

C.6.7 Retract Limit

Default: Current position on power-up
Range: Valid 16-bit Position

The Retract Limit specifies the minimum value the motion controller will allow as a position Command Value. (When the Scale is negative, this is the maximum value.) A Command Value below this value will be set to the Retract Limit and will cause the parameter error bit in the Status word to be set. The Retract Limit is given in Position Units.

Note: The Retract Limit must be changed when the Scale or Offset parameters are changed.

Note: On startup, the Retract Limit defaults to the current position of the axis, so new Extend and Retract Limits must be issued followed by a 'r;P' command before the axis will move, unless different limits have been saved in the Flash.

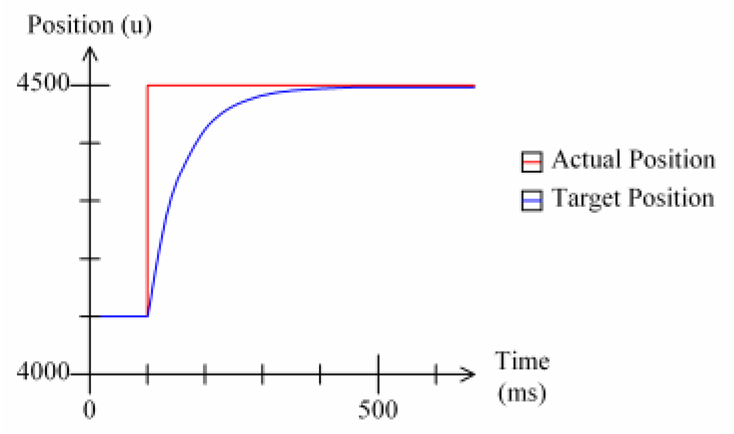
C.6.8 Velocity Limit (Reference)

Default: 65,535
Range: 0 to 65,535 position units per second

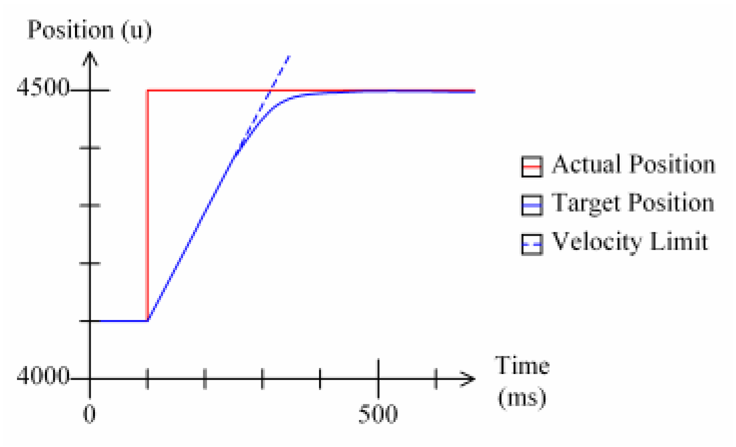
Note: This parameter is available in RMC100 CPU firmware dated 20020222 or later.

This parameter is available on analog Position and Velocity Reference axes and configures the position filter. For reference axes, the Actual Position status field reflects the actual reading from the transducer, and the Target Position reflects the filtered position, after applying the Filter Time Constant, Deadband, Velocity Limit, and Acceleration Limit parameters. This parameter limits the Target Velocity to the specified number of position units per second.

For example, suppose that the Actual Position jumps from 4100 to 4500 position units. With a high Velocity Limit (65,535) and a non-zero Filter Time Constant, the Target Position would be computed as follows:



By lowering the Velocity Limit parameter, the Target Position changes to the following:



The Velocity Limit is ignored if the position filter is disabled through the Filter Time Constant parameter.

This and the other position filter parameters can also be changed through the Reference (W) command.

For details on reference axis filtering, see Reference Axis Filtering.

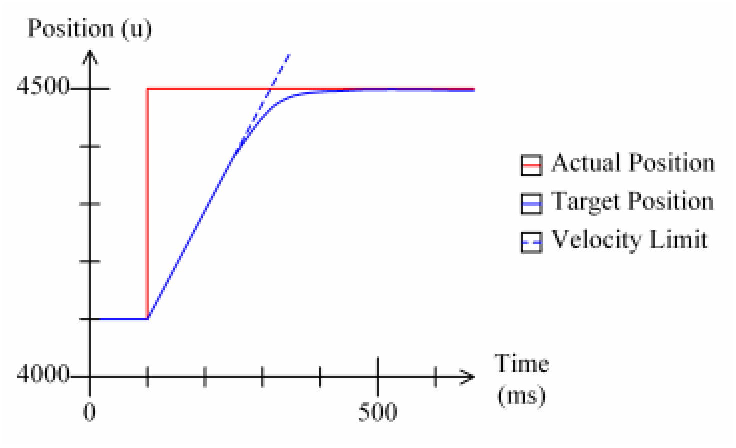
C.6.9 Acceleration Limit (Reference)

Default: 65,535
Range: 0 to 65,535 (position units per second per millisecond)

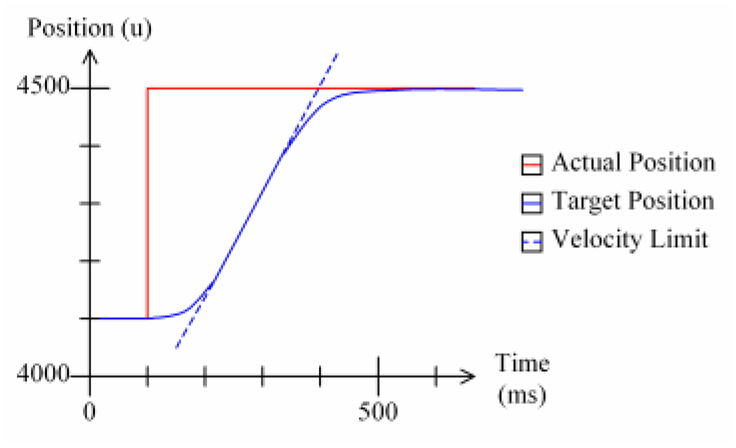
Note: This parameter is available in RMC100 CPU firmware dated 20020222 or later.

This parameter is available on analog Position and Velocity Reference axes and configures the position filter. For reference axes, the Actual Position status field reflects the actual reading from the transducer, and the Target Position reflects the filtered position, after applying the Filter Time Constant, Deadband, Velocity Limit, and Acceleration Limit parameters. This parameter limits the rate of change of the Target Velocity to the this parameters value in position units per second per millisecond.

For example, suppose that the Actual Position jumps from 4100 to 4500 position units. Using a non-zero Filter Time Constant and the Velocity Limit, but with the Acceleration Limit set to a high value (such as 65,535), the Target Position would be computed as follows:



By lowering the Acceleration Limit parameter, the Target Position changes to the following:



Notice that the Target Position was smoothed at the start of the step jump by limiting the acceleration of the Target Position.

The Acceleration Limit is ignored if the position filter is disabled through the Filter Time Constant parameter.

The Acceleration Limit parameter must be at least equal to the Velocity Limit (in pos-units/sec) divided by twice the Filter Time Constant (in ms). For example, with a Filter Time Constant of 2 ms and a Velocity Limit of 1000 position units per second, the minimum Acceleration Limit will be 250. This limit is introduced because lower values for the Acceleration Limit would result in the Target Position overshooting the Actual Position significantly.

This and the other position filter parameters can also be changed through the Reference (W) command.

For details on reference axis filtering, see Reference Axis Filtering.

C.6.10 Filter Time Constant (Reference)

Default: 0 (disabled)
Range: 1 to 65,535 milliseconds, or 0 to disable

Note: This parameter is available in RMC100 CPU firmware dated 20020222 or later.

This parameter allows filtering the position on an analog position or velocity reference axis. By default the filter is disabled. If this parameter is set to a non-zero value, then the position feedback is filtered using an Infinite Impulse Response (IIR) filter with a time constant set to this parameter's value in milliseconds. The equivalent cut-off frequency in Hertz is found by 1/2pt, where t is the time constant in seconds. For example, a time constant of 5 milliseconds is equivalent to a cut-off frequency of 32 Hz.

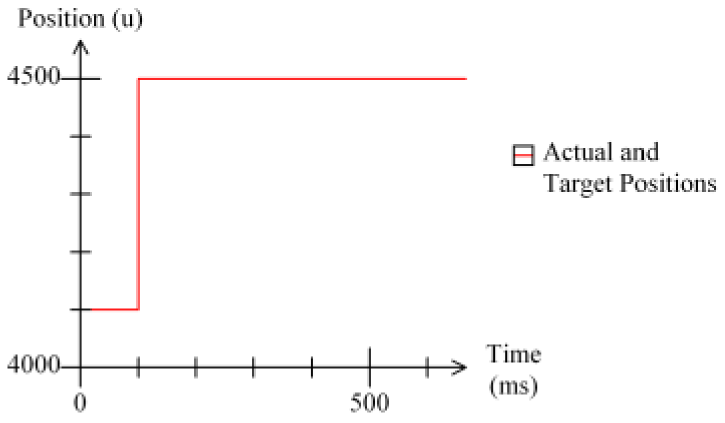
This feature should be left disabled in systems that need to react to quick changes in the reference input, as increasing this filter value will increase the phase delay in the system feedback.

To compute an initial value for your time constant value, recall that for a step jump in the feedback value, an IIR filter will reach 63% in 1 time constant, and 99% in 5 time constants.

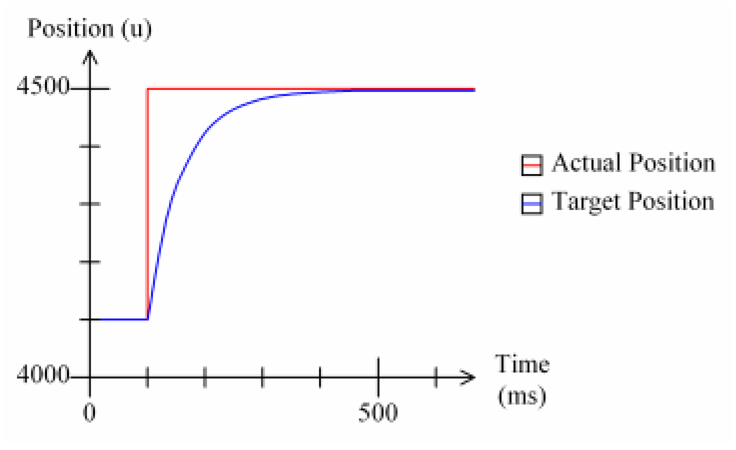
Note: When this parameter is set to zero, the IIR filter is disabled, as is the slew limiting controlled by the Velocity Limit, and Acceleration Limit parameters.

For reference axes, the Actual Position status field reflects the actual reading from the transducer, and the Target Position reflects the filtered position, after applying the Filter Time Constant, Deadband, Velocity Limit, and Acceleration Limit parameters.

For example, suppose the Actual Position makes a jump from 4100 to 4500 position units. With the position filter disabled (Filter Time Constant set the zero), the Target Position would also match the Actual Position:



By applying a non-zero Filter Time Constant, the Target Position does not make a step jump:



This and the other position filter parameters can also be changed through the Reference (W) command.

For details on reference axis filtering, see Reference Axis Filtering.

C.6.11 Reference Dead Band (Reference)

Default: 0
Range: 0 to 65,535 position units

Note: This parameter is available in RMC100 CPU firmware dated 20020222 or later.

This parameter is used to eliminate jitter in an analog reference input. It can be used in conjunction with the filter parameters (Filter Time Constant, Velocity Limit, and Acceleration Limit) to control the consistency of the analog reference.

The Reference Deadband works as follows. As each reading is made from the analog transducer, it is compared with the last value that was not ignored due to the deadband. If the value is within the Reference Deadband parameter, then it is ignored and the last value is re-used. If it is outside the deadband, then the new value is used.

Typically, this parameter is used to obtain final smoothing after using the filter parameters to reduce step jumps and more substantial noise.

For reference axes, the Actual Position status field reflects the actual reading from the transducer, and the Target Position reflects the filtered position, after applying the Filter Time Constant, Deadband, Velocity Limit, and Acceleration Limit parameters.

This and the other position filter parameters can also be changed through the Reference (W) command.

For details on reference axis filtering, see Reference Axis Filtering.

C.6.12 In Position

Default: 50

Range: 0 to 65535

This parameter specifies the size of a window around the Command Position. When the Actual Position gets within this window, the In Position bit is set in the Status word. Notice that the In Position bit is not latched and therefore could go off again if the axis moves back outside the In Position window.

Example:

If an axis Command Position is 10,000 and the In Position parameter is 30, the In Position bit will be set when the axis is stopped and its Actual Position is between 9,971 and 10,029. The bit will be cleared whenever the Actual Position is outside the range.

C.6.13 Following Error

Default: 250
Range: 0 to 65535

The Following Error determines how large the difference between the Target Position and Actual Position can get before the Following Error bit is set in the Status word.

C.6.14 Auto Stop

Default: 0x1FE0 (Soft Stops enabled, Hard Stops enabled for Transducer Errors)

Click here for the Auto Stop Bit Map

The Auto Stop parameter controls the action taken on the rising edge of any of the axis' error bits (the eight (8) most-significant bits in the Status word listed below). The user has control over which error bits cause which levels of stop, or whether an error will cause a stop at all. The default setting of all the AutoStops is Hard Stop, as described below.

During startup and tuning, you will typically need to set some AutoStops to Status Only to keep halts from interfering with the tuning. After completeing the startup procedure, make sure to set the AutoStops to setting that are safe for the machine operation.

Auto Stops are always active unless the axis is in Open Loop, in which case no Auto Stops will be triggered even though error bits may be set.

Note: AutoStops will not occur if the axis is in Open Loop!

The eight (8) most-significant bits in the Status word, as listed below, are the error bits, with the exception of the Home Input status bit for Quadrature axes with Analog (QUAD) or Stepper (STEP) output:

Fault	**QUAD**	**STEP**	**All Others**
7	Encoder Error/Fault	Encoder Error/Fault	No Transducer
6	Extend Limit	Extend Limit	Transducer Noise
5	Retract Limit	Retract Limit	Transducer Overflow
4	Overdrive	Overdrive	Overdrive
3	Parameter Error	Parameter Error	Parameter Error

2	Home Input	Home Input	Pos./Press. Overflow
1	Integrator Windup	Compensation Timeout	Integrator Windup
0	Following Error	Following Error	Following Error

The available actions for each fault are listed below:

- **Status Only**
  The fault will be reflected in its corresponding status bit in the Status word, but no further action will be taken. For some transducer types, this option may not be available for the following faults: No Transducer, Transducer Overflow, and Transducer Noise.
- **Soft Stop**
  If the axis is in closed loop, the fault will trigger ramping the axis to a stop. The speed will ramp down to zero using the current Deceleration value. If the axis is in Open Loop, the drive will not be affected. If there is an event sequence on the axis, it will stop executing. The Halt status bit will also be set in the Status word. Some faults will use the open loop ramp down although the axis was in closed loop: Position Overflow, No Transducer, Transducer Overflow, Transducer Noise, and Encoder Error/Fault Input. This is done because the position feedback is not dependable and closed loop control cannot be maintained.
- **Hard Stop**
  If the axis is in closed loop, the fault will trigger the drive output to go immediately to 0 mV on analog outputs and no steps for stepper outputs, and the axis will be placed in open loop mode. If the axis is in Open Loop, the drive will not be affected. If there is an event sequence on the axis, it will stop executing. The Halt and Open Loop status bit will also be set in the Status word.
- **Disable Drive**
  The fault will be handled as in a Hard Stop, but additionally the Amp Enable output on QUAD and STEP will be opened. Use the Amp Enable (a) command to close the output again. This option is only available for faults on QUAD and STEP axes.

To view or change the Auto Stop parameter value, open the Auto Stop popup editor in RMCWin using one of the methods described in Using Popup Editors. Select the desired action for each fault type and click OK. Changes to this parameter do not take effect until you issue a Set Parameters (P) command.

Manual Parameter Entry

This parameter can also be edited manually, but this is discouraged since it is much easier to use the popup editor. You can enter hexadecimal numbers by typing a leading 0x, as in 0x1FE0. Each of the eight faults listed above has two bits assigned to it, labeled S and H in the table below:

Bit	Description	Bit	Description
15	Fault 7 - Bit S	7	Fault 7 - Bit H
14	Fault 6 - Bit S	6	Fault 6 - Bit H
13	Fault 5 - Bit S	5	Fault 5 - Bit H
12	Fault 4 - Bit S	4	Fault 4 - Bit H
11	Fault 3 - Bit S	3	Fault 3 - Bit H
10	Fault 2 - Bit	2	Fault 2 - Bit H

	S		
9	Fault 1 - Bit S	1	Fault 1 - Bit H
8	Fault 0 - Bit S	0	Fault 0 - Bit H

For each fault, the two bits in the Auto Stop parameter define the action as follows:

Bit H	**Bit S**	**Description**
0	0	Status Only
0	1	Soft Stop
1	0	Hard Stop
1	1	Disable Drive

If you select Status Only for a fault that cannot use that action, then the axis will use a Soft Stop action for that fault. Similarly, if you select Disable Drive for a fault on an axis that does not have an Amp Enable output, the axis will use the Hard Stop action for the fault.

C.6.15 Auto Stop Bit Map

The table below provides an easy method to convert bit patterns to hexadecimal numbers.

```
                         F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                         E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
    Hexadecimal To       D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
    Binary Conversion    C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
    Table                B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                         A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                         9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                         8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                         7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                         6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                         5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                         4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                         3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                         2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                         1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                         0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0

                            _______   _______   _______   _______
                           | | | | | | | | | | | | | | | | | | | |
                           |1|1|1|1| |1|1| | | | | | | | | | | | |
                           |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
 Bit Definition            |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
----------------            / / / /   / / / /   / / / /   / / / /
S  No Xdcr/Encdr Flt - 15 -/ / / /   / / / /   / / / /   / / / /
O  Xdcr Nse/Ext Lmt -- 14 --/ / /   / / / /   / / / /   / / / /
F  Xdcr Ovr/Ret Lmt -- 13 ---/ /   / / / /   / / / /   / / / /
T  Overdrive  -------- 12 ----/   / / / /   / / / /   / / / /
                                 / / / /   / / / /   / / / /
S  Parameter Error --- 11 ------/ / / /   / / / /   / / / /
T  Pos Ovr/Home Inpt - 10 -------/ / /   / / / /   / / / /
O  Integrator Windup - 09 --------/ /   / / / /   / / / /
P  Following Error --- 08 ---------/   / / / /   / / / /
=========================             / / / /   / / / /
H  No Xdcr/Encdr Flt - 07 -----------/ / / /   / / / /
A  Xdcr Nse/Ext Lmt -- 06 ------------/ / /   / / / /
R  Xdcr Ovr/Ret Lmt -- 05 -------------/ /   / / / /
D  Overdrive --------- 04 --------------/   / / / /
                                           / / / /
S  Parameter Error --- 03 ----------------/ / / /
T  Pos Ovr/Home Inpt - 02 -----------------/ / /
O  Integrator Windup - 01 ------------------/ /
P  Following Error --- 00 -------------------/
```

If both Soft Stop and Hard Stop bits are set for a particular error condition, a Hard Stop will be executed and the Amp Enable output will be opened on QUAD and STEP axes.

Appendix D: Status Field Reference

D.1 Valid 16-Bit Positions

The positions used by the RMC are stored in a 16-bit number. This limits the range of positions to 65536 positions. However, by using the Scale and Offset parameters, the user can shift where this 65,536-position range is located.

If Scale is positive, then Offset holds the lowest position value. Any positions displayed below this value are not valid. They overflowed the absolute maximum supported position of 65,535.

If Scale is negative, then Offset holds the highest position value. Any positions displayed above this value are not valid. They overflowed the absolute minimum supported position of -65,535.

Example 1:

Scale**:** 30,300

Offset**:** 0

Position Range: 0 to 65,535

Discussion: Because Scale is positive, the positions range from Offset to Offset + 65,535.

Example 2:

Scale: 25,604

Offset: -1000

Position Range: -1000 to 64,535

Discussion: Because Scale is positive, the positions range from Offset to Offset + 65,535.

Example 3:

Scale: 32,050

Offset: 1000

Position Range: 1000 to 65,535

Discussion: Because Scale is positive, the positions range from Offset to Offset + 65,535. However, because the upper limit would be greater than 65,535 (66,535), it is truncated at 65,535. Notice that positions less than the Offset (1000) are not valid. They were beyond 65,535 and therefore wrapped back down to 0-999.

Example 4:

Scale: -30,300

Offset: 40,000

Position Range: -25,535 to 40,000

Discussion: Because Scale is negative, the positions range from Offset - 65,535 to Offset.

Example 5:

Scale: -31,541

Offset: -10,000

Position Range: -65,536 to -10,000

Discussion: Because Scale is negative, the positions range from Offset - 65,535 to Offset. However, because the lower limit would be less than -65,536 (-75,536), it is truncated at -65,536. Notice that positions greater than the Offset (-10,000) are not valid. They were beyond -65,535 and therefore wrap around to -9999 to 0.

D.2 Position Status Fields

D.2.1 Command Position

The Command Position is the requested position (specified by the Command Value) with bounds checking applied. If the requested position is outside the Retract or Extand Limit, the Command Position will be set to the value of the limit, and the axis will go only to the limit. The Command Position is updated when any move command, or Set Parameters (P) command is issued using the Command parameter.

Why Bother?

If the Command Position is not the same as the requested position then one of two things has happened:

1. A program error has asked the axis to go to an invalid position. In this case the program error should be corrected. The PARAMETER ERROR bit in the Status word will be ON when this occurs.

2. The Command Value field has just been changed and the Motion Controller has not had a chance to acknowledge the new request. Note that the In Position bit is no longer valid for the rest of the scan. The In Position bit is valid only when the Command Value is equal to the Command Position.

D.2.2 Target Position

When the axis is in Closed Loop Mode, the Target Position is the calculated instantaneous ideal position for the axis. The Target Position is calculated every loop by the target generator state machine of the on-board motion control program. The PID routine uses the difference between the Target Position and the Actual Position as its input to compute any required corrective drive.

During a move the path of the Target Position toward the Command Position will be the perfect profile for the Actual Position to follow.

Note: When an axis is stopped, the Target Position should be the same as the Command Position unless an error or HALT has occurred (see Status).

Note: When an axis is not in position Closed Loop Mode, Target Position is set to the Actual Position.

Why Bother?

Knowing the relationship between the Target Position and Actual Position is key to tuning the axis. The main goal in tuning the axis is to minimize the error between the Target and Actual Positions. The plot function is a very useful visual aid in tuning the axis.

D.2.3 Actual Position

The Actual Position is the measured position of the axis at any moment. This position is updated every control loop. The Actual Position is calculated from the Transducer Counts as follows:

For Analog, MDT, and SSI:

$$\text{Actual Position} = \text{Offset} + \frac{\text{Transducer Counts} \times \text{Scale}}{32768}$$

For Quadrature:

$$\Delta\text{Actual Position} = \frac{\Delta\text{Transducer Counts} \times \text{Scale}}{32768}$$

$$\text{Actual Position} = \text{Actual Position} + \Delta\text{Actual Position}$$

$$\text{Actual Position Displayed} = \text{Actual Position Internal} + \text{Offset}$$

For Resolvers:

$$\text{Actual Position} = \frac{(\text{Resolver Counts} - \text{Count Offset}) \times \text{Scale}}{32768}$$

D.2.4 COUNTS

This field holds a raw, non-scaled reading from the transducer. The exact meaning of this field depends on the type of transducer used. It is useful to know the counts in order to set up or verify your Scale and Offset settings. For pressure axes, this field is renamed COUNTS A. For force axes, there are two counts fields, COUNTS A and COUNTS B, corresponding to the first and second input channels. Refer to the appropriate section below for the type of feedback transducer(s) you are using:

Magnetostrictive Displacement Transducers (MDT)

The counts represent the number of 120MHz time periods that have passed either between the Start and Stop return pulses for a Start/Stop MDT or from the rising to falling edges of a PWM MDT.

Counts will read 0 when the MDT does not return counts. The actual distance from the end of the rod to the magnet is found by multiplying the counts by the gradient value of the MDT. Therefore, to translate from counts to distance, use the following formulas:

Gradient in msec/in:

$$\text{Distance} = \frac{1}{\text{Gradient}} \times \frac{\text{Counts}}{120{,}000{,}000\ \text{Counts/sec}}$$

Example: An MDT has a gradient of 9.153 msec/in. The counts read 6000. The following equation finds the position:

$$\text{Distance} = \frac{1}{9.153\,\mu\text{sec/in}} \times \frac{1{,}000{,}000\ \mu\text{sec}}{\text{sec}} \times \frac{6000\ \times \text{sec}}{120{,}000{,}000}$$

$$\text{Distance} = 5.463\,\text{in}$$

Gradient in m/sec:

$$\text{Distance} = \text{Gradient} \times \frac{\text{Counts}}{120{,}000{,}000\ \text{Counts/sec}}$$

Example: An MDT has a gradient of 2845.06 m/sec. The counts read 6000. The following equation finds the position:

$$\text{Distance} = \frac{2845.06\,\text{m}}{\text{sec}} \times \frac{6000}{120{,}000{,}000\ \text{sec}}$$

$$\text{Distance} = 0.1423\,\text{m} = 142.3\,\text{mm}$$

Synchronous Serial Interface (SSI)

The counts are equal to the counts returned from the SSI interface. That is, SSI transducers return a digital count value. The resolution of the counts is defined for the particular SSI device. Therefore, to translate from counts to distance, use the following formula:

$$\text{Distance} = \text{Counts} \times \text{Resolution}$$

For linear devices, the resolution will be a unit of linear measure, such as 5um. For rotational devices, the resolution will be in degrees or fractions of a revolution.

Example 1: An SSI has a resolution of 5 um per count. The counts read 6000. The following equation finds the position:

$$\text{Distance} = 6000\ \text{counts} \times \frac{5\ \mu\text{m}}{\text{count}} = 30000\,\mu\text{m} = 30.00\,\text{mm}$$

Example 2: A single-turn rotary absolute encoder with SSI feedback has 8192 counts per revolution. The counts read 6000. The following equation converts the counts to degrees from top-dead-center:

$$\text{Distance} = 6000\ \text{counts} \times \frac{360^\circ}{8192\ \text{counts}} = 263.7^\circ$$

Quadrature

For quadrature devices, the counts increase by one each time an A or B quadrature input toggles such that the phase of A leads B. The counts decrease by one each time an A or B quadrature input toggles such that phase of B leads A. The counts are set to zero whenever a position is set by a Zero Position or Offset Positions command, but are otherwise relative.

Analog

The counts used for analog transducers are defined differently depending on the input range and quantity type of the channel. The input ranges include 0-10V, ±10V, 0-5V, ±5V, and 4-20mA. Quantity types include position, velocity, pressure, and force. Pressure, force, and velocity counts are signed 16-bit numbers, which range from 32,768 to 32,767, but position counts are unsigned 16-bit numbers, which range from 0 to 65,535. Therefore, the charts below treat position counts differently.

There are three error conditions that can be triggered from the counts. These are described below:

- **No Transducer**
  This condition occurs when the value read from the transducer is below a minimum value. The minimum value depends on the input range and quantity type, and is shown in the charts below. When this condition is detected, the No Transducer status bit is set, and the axis will be halted.

- **Transducer Overflow**
  This condition occurs when the analog reading is at its maximum value, which will be 32,767 for all analog axes except analog position inputs, in which case it will be 65,535. When this condition is detected, the Transducer Overflow status bit will be set. However, the axis may or may not be halted depending on the configuration of the Auto Stop parameter for this axis.

- **Transducer Noise**
  This condition occurs only for analog Position Control axes. For these axes, it is triggered when the transducer feedback jumps by more than 100 position units per millisecond, for at least six control loops in a row. The reason the value of 100 units/ms was chosen is that a Position Control can only be commanded in the RMC to move 65,535 units/s, and therefore 64 units/ms. So a feedback speed above 100 units/ms is deemed to be noise. When this condition is detected, the Transducer Noise status bit is set, and the axis is may or may not halt depending on the configuration of the Auto Stop parameter.

The tables below list the counts given for the minimum input readings, plus the over- and under-range readings for each input range.

0 to 10V

Pressure, Force, Velocity		Position	
<u>Input</u>	<u>Counts</u>	<u>Input</u>	<u>Counts</u>
>10.08V	32,767*	>10.07V	65,535*
10.00V	32,500	10.00V	65,100
0.00V	0	0.00V	100
<-0.50V	<-1,625**	<-0.02V	0**

-10 to 10V

Pressure, Force, Velocity		Position	
Input	Counts	Input	Counts
>10.08V	32,767*	>10.09V	65,535*
10.00V	32,500	10.00V	65,250
0.00V	0	0.00V	32,750
-10.00V	-32,500	-10.00V	250
<-10.08V	<32,768**	<-10.08V	0**

0 to 5V

Pressure, Force, Velocity		Position	
Input	Counts	Input	Counts
>+5.04V	32,767*	>5.03V	65,535*
5.00V	32,500	5.00V	65,100
0.00V	0	0.00V	100
<-0.25V	<-1,625**	<-0.01V	0**

-5 to 5V

Pressure, Force, Velocity		Position	
Input	Counts	Input	Counts
>5.04V	32,767*	>5.04V	65,535*
5.00V	32,500	5.00V	65,250
0.00V	0	0.00V	32,750
-5.00V	-32,500	-5.00V	250
<-5.04V	-32,768**	<-5.04V	0**

4 to 20mA

Pressure, Force, Velocity		Position	
Input	Counts	Input	Counts
>20.16mA	32,767*	>20.13mA	65,535*
20.00mA	32,500	20.00mA	65,100

4.00mA	6500	4.00mA	13,100
<3.60mA	<5850**	<3.60mA	11,800**

* Counts in this range will cause the Transducer Overflow bit to be set in the Status Word, as described above.

** Counts in this range will cause the No Transducer bit to be set in the Status Word, as described above.

D.2.5 Status (Position/Speed)

The Status register is a collection of 16 bits to provide a summary of the state of the axis. The bits are numbered 0 to 15, with bit 0 being the right-most, least-significant bit, and bit 15 being the left-most, most-significant bit.

The eight most-significant bits (MSBs) are generally used for error bits (there are some exceptions for quadrature axes). These eight bits can be configured to trigger an axis halt on their rising edges using the Auto Stop parameter. Each of the error bits latch on when the error is detected. Unless otherwise indicated in the individual bit descriptions below, the error bits are cleared when one of the following commands are issued to the axis:

- Set Parameters (P)
- Reset Position (q)
- Go (G and g)
- Relative Move (J and j)
- Move Relative to an Axis (0xC0-0xCF)
- Follow Spline Segment (f)
- Sine Move (~)
- Open Loop (O)
- Set Mode (M)
- Change Acceleration (A)
- Change Deceleration (D)
- Set Speed (V and v)

The Status register is displayed in RMCWin as a 16-bit hexadecimal number. However, determining which bits are on from this value can be difficult, especially for occasional users. Therefore, RMCWin provides the Status Bits Window, which labels each bit independently. See Using the Status Bits Window for details. The following topics also provide a chart summarizing the bits in this register:

- MDT, SSI and Analog Status Bitmap
- Quadrature (QUAD and STEP) Status Bitmap

Bit 15 (MDT, SSI, Analog, Resolver) - No Transducer

This error bit is set to indicate that the transducer is not responding.

For MDT and SSI transducers, this occurs when the transducer does not respond within six control loop times (a total of 6 or 12 ms).

For Analog transducers, this occurs when the analog counts are below a certain value depending on the input and axis types as described in the Transducer Counts topic.

For resolvers, this occurs when the module is not able to determine a valid position. Possible reasons include the RMC case not properly grounded, or actual accelerations or speeds are too high. See the Resolver Specifications topic for details.

This error bit is cleared when any of the commands listed above is issued, and the transducer is now present.

The RMC can be configured to automatically stop on the rising edge of this bit by using the Auto Stop parameter.

Bit 15 (QUAD or STEP) - Encoder Error or Fault Input

This status bit can represent either an encoder error or a Fault input error. The user configures which of these two conditions is reported by this status bit, and thereby also determines which can be used to trigger an Auto Stop. There are four options:

- **Encoder Error only**. This bit will go high if the encoder circuitry detects an error, which is defined as an invalid transition of the A and B lines. This usually occurs due to over-speed or noise conditions.
- **Fault Input only**. This bit will go high if the Fault input goes active. The Fault input is intended to be a means of the drive amplifier letting the RMC know that it no longer has control. If the Fault input is not used, it should be made inactive by wiring or by reversing the polarity in the Configuration Word parameter.
- **Encoder Error or Fault Input (default)**. This bit will go high if either of these conditions occurs. This is the default behavior for QUAD and STEP axes, but does leave it ambiguous as to whether the error was triggered due to an encoder error or the Fault input.
- **Unused**. This bit will always be low.

See Quadrature Configuration or Stepper Configuration for details on defining this bit.

This error bit is cleared when any of the commands listed above is issued, and the underlying error condition is no longer present.

The RMC can be configured to automatically stop on the rising edge of this bit by using the Auto Stop parameter.

Bit 14 (MDT, SSI, Analog, Resolver) - Transducer Noise

This error bit is set when the RMC detects persistent noise on the transducer. Noise is defined by the position read from the transducer jumping by more than 100 position units in a millisecond. Notice that this error bit does not occur on analog position reference, analog velocity control or reference, analog pressure or analog differential force axes. That is, it only occurs on MDT, SSI, and analog position control axes.

For the first five control loops that the transducer position is outside this range, the RMC will approximate the Actual Position using the current Actual Speed value and continue control normally. If the noise persists on the sixth control loop, then the Transducer Noise error bit is set.

This bit only applies to position axes. Therefore, analog velocity, pressure, and force axes never set this bit. Notice that for quadrature feedback axes, this error bit is redefined as described below.

This error bit is cleared when any of the commands listed above is issued, and the underlying error condition is no longer present.

The RMC can be configured to automatically stop on the rising edge of this bit by using the Auto Stop parameter.

Bit 14 (QUAD or STEP) - Extend/Clockwise Limit Switch

This bit is set when the axis is moving in the extend or clockwise direction and the Extend/Clockwise Limit Switch is activated.

Notice that this bit is not latched. That is, it is automatically cleared when the limit switch goes inactive, or the axis changes direction.

The Limit Switch bit in the Config parameter determines the active state of the limit switch. If the limit switch is not used, either hardwire it to the inactive state or use the Limit Switch bit to make the limit switch always inactive.

The RMC can be configured to automatically stop on the rising edge of this bit by using the Auto Stop parameter.

Bit 13 (MDT, SSI, Analog, Resolver) - Transducer Overflow

This error bit is set when the RMC detects that the position on the transducer is outside acceptable limits. The limits are defined as follows for each transducer:

Transducer	**Overflow Condition**
MDT (Start/Stop)	The MDT has not responded with a return pulse within eighteen bits of counts (262,144 counts), which occurs in just over two milliseconds. This is a distance of roughly 240 inches (6.1 m).
MDT (PWM)	The MDT has not responded with a falling edge within eighteen bits of counts (262,144 counts), which occurs in just over two milliseconds. This is a distance of roughly 240 inches (6.1 m) divided by the number of recirculations configured in the transducer.
SSI (Linear)	The SSI transducer indicated that it did not detect the magnet location. Balluff transducers indicate this by setting bit 21 of the count value, which limits the position to 2,097,152 counts. Other transducers (including Temposonic) indicate this by returning a count value of zero. Configure which method the RMC uses as described in SSI Configuration.
SSI (Rotary)	Not defined. This error bit will never be set.
Analog	The analog reading is above the maximum input allowed by the RMC. See the Counts topic for a description of what voltages or currents this range corresponds to. Notice that overflowing in the negative direction will result in a No Transducer error (bit 15) instead of a Transducer Overflow error (bit 13).

This error bit is cleared when any of the commands listed above is issued, and the underlying error condition is no longer present.

The RMC can be configured to automatically stop on the rising edge of this bit by using the Auto Stop parameter.

Bit 13 (QUAD or STEP) - Retract/Counterclockwise Limit Switch

This bit is set when the axis is moving in the retract or counterclockwise direction and the Retract/Counterclockwise Limit Switch is activated.

Notice that this bit is not latched. That is, it is automatically cleared when the limit switch goes inactive, or the axis changes direction.

The Limit Switch bit in the Config parameter determines the active state of the limit switch. If the limit switch is not used either hardwire it to the inactive state or use the Limit Switch bit to make the limit switch always inactive.

The RMC can be configured to automatically stop on the rising edge of this bit by using the Auto Stop parameter.

Bit 12 - Overdrive

This bit is set when the calculated drive output reaches or exceeds the maximum drive. For the RMC's analog outputs, the maximum drive defaults to ±10 Volts, but can be set lower using the Limit Drive (L) command. For stepper outputs, the limit is defined by the MAX STEPS/MSEC parameter, up to a maximum of 1024 steps/millisecond.

This error bit is cleared when any of the commands listed above is issued, and the underlying error condition is no longer present.

The RMC can be configured to automatically stop on the rising edge of this bit by using the Auto Stop parameter. Notice that it is highly recommended that the Auto Stop be enabled for this error.

This error usually indicates one of the following:

- The system does not have enough power to drive the axis at the requested Speed.
- There is a step jump in the Actual Position so that the error is large and the resulting corrective drive output exceeds ±10 volts. This is usually due to a noisy position/pressure transducer or encoder.
- The wiring from the RMC's drive output to the amplifier is backwards. Either the wiring must be swapped plus for minus or the Reverse Drive Config bit must be set.
- The axis is physically obstructed or there is a loss of quadrature counts so the Actual Position cannot reach the Target Position.
- The drive amplifier or hydraulic power is off. This bit will turn on when the Target and Actual Positions become separated and the closed loop drive increases until it reaches the limit. When power is returned to the drive or hydraulic power, the axis will jump unless the axis faulted due to an Auto Stop for this error.

Bit 11 - Parameter Error

This bit is set when an initialization parameter or control parameter is out of bounds. In some cases one parameter's limit will depend on the value of another parameter, so definite limits may not always be available. However, the motion controller does try to replace the erroneous value with another that is within range, so the offending parameter can be determined by comparing the parameter values before and after the error bit is set.

This bit is cleared when any command is issued to this axis. Notice that it may be immediately set again if the command was invalid or had invalid command parameters.

The RMC can be configured to automatically stop on the rising edge of this bit by using the Auto Stop parameter.

Because this error bit is used for many types of errors, RMCWin has the ability to display the last several parameter errors in more detail. This list of errors is compiled only while RMCWin is running, so it is recommended that it be left running while configuring the system.

To display this list, select Parameter Error List from the Window menu. For RMC CPU firmware versions dated 19971016 or later, this list should include brief descriptions of the last twenty parameter errors to occur on the motion controller. To receive help on a particular error, in the error list, click on the error, and then click Help on Error.

Bit 10 (MDT, SSI, Analog, Resolver) - Position Overflow

This bit is set when the actual position has exceeded the range that can be displayed with a 16-bit number. See Valid 16-bit Positions for details on this limitation. This bit is also set ifwhen recirculations are usedTransducer Counts is calculated to less than zero. Actual Position will be displayed as 65535 in the overflow case, and as zero (0) in the underflow case.

This error bit is cleared when any of the commands listed above is issued, and the underlying error condition is no longer present.

The RMC can be configured to automatically stop on the rising edge of this bit by using the Auto Stop parameter.

To recover from this error condition you must move the axis, either manually or with an Open Loop Command, until the Actual Position becomes valid again.

This error occurs for one of two reasons:

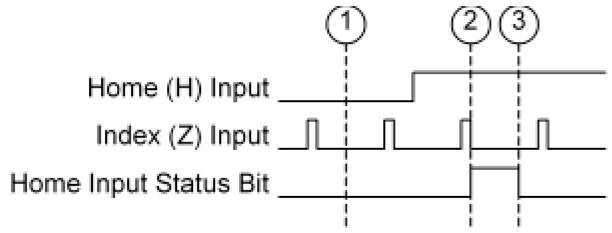
1. The overflow case (Actual Position displayed as 65535) happens because the transducer is longer than can be accommodated by the 16 bit Actual Position Field given the scaling of the transducer counts. To fix this, you can either reduce the Scale parameter; increase the Prescale Divisor in the Configuration Word; or, if you are using a PWM type transducer, increase the number of recirculations specified in the Configuration Word (without increasing the number of recirculations in the transducer).
2. The Underflow case (Actual Position displayed as 0) happens because the number of recirculations specified in the Configuration Word is too large. To fix this, decrease the number of recirculations specified in the Configuration Word (without changing the number of recirculations in the transducer).

Bit 10 (QUAD or STEP) - Home Input

The Home Input status bit operation is programmable by using the Level/Edge bit in the Config parameter:

- **Latched Edge Mode:** In this mode, the Home Input status bit is set when either the Index (Z) or Home input becomes inactive after both had been active. It remains latched until any of the commands that clear error bits are issued (listed above), or the Arm Home (@) command is issued. This mode is selected by clearing the Level/Edge configuration bit.

Example:



1. The Arm Home (@) command is issued. Without this command the Actual Position will not latch to the home position, nor will the Home Input status bit ever be set in this mode.

2. The Actual Position is homed because both the Home and Index (Z) inputs had been active, but one drops to inactive (the Z in this case). The Home Input status bit is set to indicate the axis has homed.

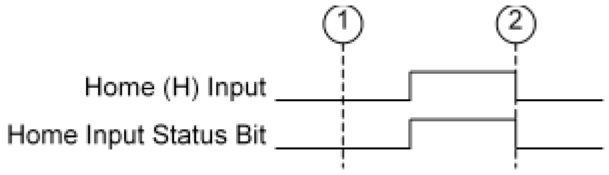
3. A Go (G) command is issued, which clears the Home Input status bit.

- **Level Mode:** In this mode, the Home Input status bit is set when the Home input is currently active, and is cleared when it is inactive. Notice that the level is not latched, and that neither the Index (Z) input nor the Arm Home (@) command affect this status bit.

Note: Revision 1 Quadrature modules require that the Arm Home (@) command be issued at least once before the Home Input status bit works correctly. This anomaly has been corrected in later revisions.

Notice that although the Home Input status bit ignores the Index (Z) input, the actual homing of the axis still occurs when either the Home (H) or Index (Z) input goes inactive after both had been active. However, this Home Input status bit mode is not intended to be used with the Index (Z) input, and therefore it is recommended that the Index (Z) input be left disconnected and configured as "active with no current" in the Configuration Word.

Example:



1. The Arm Home (@) command is issued. Without this command, the Actual Position will not be homed on the falling edge of the Home input. However, this has no effect on the Home Input status bit.

2. The Actual Position is homed because the Home input goes inactive. The Home Input status bit is cleared to indicate that the Home input is now inactive. As noted above, this example assumes that the Index (Z) input is always active.

See the Homing a Quadrature Axis topic for details on using this status bit to home an axis.

The RMC can be configured to automatically stop on the rising edge of this bit by using the Auto Stop parameter, but this is rarely used in applications.

Bit 9 (MDT, SSI, Analog, Resolver or QUAD) - Integrator Windup

This bit is set when the drive due to the integrator exceeds ±20% or 80% of full drive, depending on the setting of the Integrator Limit bit in the Configuration word.

This error bit is cleared when any of the commands listed above is issued, and the underlying error condition is no longer present.

The RMC can be configured to automatically stop on the rising edge of this bit by using the Auto Stop parameter.

The integrator will wind up under the following conditions:

- Axis is out of null and requires a large drive to stay in position while stopped. Notice that some null drift is common in hydraulic systems, but this should not be as high as the 20% or 80%

integrator limit.

- The axis' feed forward values are not tuned correctly. The feed forward values should be adjusted to reduce following errors.
- The axis is obstructed from moving.
- The drive or hydraulic power is not on and the axis is still in closed loop mode.

Bit 9 (STEP) - Compensation Timeout

This bit is set on stepper axes when the Compensation Timeout is enabled and the axis cannot get in position within the time specified by the Compensation Timeout parameter.

This error bit is cleared when any of the commands listed above is issued, and the underlying error condition is no longer present.

The RMC can be configured to automatically stop on the rising edge of this bit by using the Auto Stop parameter.

Bit 8 - Following Error

This bit is set when the difference between the Target Position and the Actual Position is greater than the Following Error parameter. If the Following Error occurs by itself during the move, the corrective action is to tune the axis or to reduce the acceleration, deceleration, or speed of the move.

This error bit is cleared when any of the commands listed above is issued, and the underlying error condition is no longer present.

The RMC can be configured to automatically stop on the rising edge of this bit by using the Auto Stop parameter.

Bit 7 - Acknowledge

This bit is intended to be used to synchronize communication of commands and status between the PLC and RMC. This bit toggles when any of the following events occur:

- A non-zero command is written to this axis's command register through one of the following means:
  - PROFIBUS-DP Message Mode
  - Modbus Plus Module
  - Serial (RS-232/422/485) Module
  - Ethernet (includes commands issued through messaging, EtherNet/IP I/O, the RMCLink ActiveX control, but not RMCWin)
- A spline has been successfully downloaded to this axis through the Spline Download Area. The Spline Download Area is available over Ethernet, PROFIBUS-DP Message Mode, Serial (RS-232/422/485 Module), and Modbus Plus.
- In PROFIBUS-DP Compact Mode without Sync, the Command register changes, or the Data Out register changes for a command that uses this register. This includes changes to the Status Area Request bits (bits 8-11 of the command).
- In PROFIBUS-DP Compact Mode with Sync, the Acknowledge bit functions as in Compact Mode without Sync, except that the Sync Register has to change before a new command will be

registered and the Acknowledge bit is toggled, as described in Using the PROFIBUS-DP Compact Mode.

Notice that commands received from RMCWin, the Step Table, the RMCLink ActiveX Control and .NET Assembly Component, or from discrete inputs do not affect the Acknowledge bit.

Why Bother?

This bit must be used when getting parameters, profiles, status bit and graph data so the program knows that the data being requested is valid. The In Position status bit is not valid after a move command is issued until the Acknowledge Bit toggles.

Bit 6 - Initialized

This bit is set after a Set Parameters (P) command is successfully executed. This bit is only cleared when the module is reset.

Until this bit is set, the axis cannot enter closed loop. Therefore, the following commands are not allowed until the Set Parameters (P) command has been issued to the axis:

- Go (G and g)
- Relative Move (J and j)
- Move Relative to an Axis (0xC0-0xCF)
- Follow Spline Segment (f)
- Set Position/Pressure (c)
- Sine Move (~)

Bit 4-5 - State Bits

Bits 4 and 5 show the state of the target generator. The exact definition varies depending on the current move type. Each move type is described below:

Trapezoidal or S-curve Moves

Trapezoidal and S-curve moves are invoked using the Go (G and g) commands, without the Quick Mode, Gear, or Rotational mode bits set, or through the Relative Move (J and j) and Move Relative to an Axis (0xC0-0xCF) commands. During these moves, the state bits reflect the current stage of the move:

Bit #		
5	**4**	**Description**
0	0	The target is stopped.
0	1	The target is accelerating.
1	0	The target is moving at constant velocity.
1	1	The target is decelerating.

Quick Moves

Quick moves are initiated using the Go (G and g) command with the Quick Move mode bit set. The state bits indicate the current stage of the move:

Bit #		
5	**4**	**Description**
0	0	The move is complete.
0	1	The drive is increasing to the requested drive.
1	0	The drive is holding at the requested drive.

1	1	The target is ramping down in closed loop.

Geared Moves

Geared moves are issued using the Go (G and g) command with the Gear mode bit set. The state bits are defined as follows to reflect the state of the gear ratio:

Bit #		
5	**4**	**Description**
0	0	The gear ratio is at the requested gear ratio, which is zero (gears disengaged).
0	1	The gear ratio is increasing away from zero (clutching in to gear).
1	0	The gear ratio is at the requested gear ratio, which is non-zero (gears engaged).
1	1	The gear ratio is decreasing toward zero (clutching out of gear).

Speed Control

Speed Control commands are issued through the Go (G and g) command with the Rotational mode bit set. The state bits reflect the state of the Target Speed:

Bit #		
5	**4**	**Description**
0	0	The Target Speed is at the requested speed, which is zero (0).
0	1	The Target Speed is increasing away from zero.
1	0	The Target Speed is at the requested speed, which is non-zero.
1	1	The Target Speed is decreasing toward zero.

Open Loop

Open Loop (O) commands cause the drive to ramp to a specific voltage. The state bits reflect the change in the output voltage:

Bit #		
5	**4**	**Description**
0	0	The drive is at the requested drive, which is zero (0).
0	1	The drive is increasing away from zero.
1	0	The drive is at the requested drive, which is non-zero.
1	1	The drive is decreasing toward zero.

Sine Moves

Sine moves are started using the Sine Move (~) command. In this state, the state bits indicate whether the move is accelerating, decelerating, or done:

Bit #		
5	**4**	**Description**

0	0	The sine move is complete.
0	1	The sine move is currently accelerating.
1	1	The sine move is currently decelerating.

Sine Move Continuous

Continuous sine moves are started using the Sine Move Continuous (0x30) command. In this state, the state bits indicate whether the move is between 0 and 180 degrees or between 180 and 360 degrees:

Bit #		
5	**4**	**Description**
0	0	The sine move continuous is complete.
0	1	The sine move continuous is between 0 and 180 degrees.
1	1	The sine move continuous is between 180 and 360 degrees.

Spline Moves

Spline moves are initiated using the Follow Spline (f) command. The state bits are not used for this move type, and therefore both remain set to zero.

Closed Loop Halt

When the axis halts in closed loop as a result of an Auto Stop or the Halt (H) command, the state bits will be defined as follows:

Bit #		
5	**4**	**Description**
0	0	The Target Speed has reached zero.
1	1	The Target Speed is decelerating to zero.

Stopped (Closed Loop)

At the end of all of the above moves, the state bits are both set to 0 to indicate that the target generator is complete. This will also be true after a Set Parameters (P) or Reset Position (q) command.

For example, to detect when an axis's Target Position has come to a halt after a Halt (H) command, monitor for the following status bit combination:

Halt bit	ON (1)	(Bit 2)
State bit A	OFF (0)	(Bit 4)

State bit B	OFF (0)	(Bit 5)

You may also want to monitor the Stopped bit (bit 1) to ensure that the Actual Position has stopped moving, or the In Position bit (bit 0) to ensure that the Actual Position is sufficiently close to the Command Position.

Bit 3 - Open Loop

This bit is set when the axis is in Open Loop Mode. Each axis starts in open loop mode, or may be switched into open loop through a Disable Drive (K) command, an Open Loop (O) command, a Hard Stop occurrence, or a Soft Stop occurrence due to a transducer error, which means that closed loop control cannot continue.

This bit automatically clears when the axis returns to closed loop control, which will only happen when a closed loop command is issued. Closed loop commands include the following:

- Set Parameters (P)
- Reset Position (q)
- Go (G and g)
- Relative Move (J and j)
- Move Relative to an Axis (0xC0-0xCF)
- Follow Spline Segment (f)
- Sine Move (~)

The Drive output will be set to Null Drive when entering open loop because of the Disable Drive (K) command or a Hard Stop, and will ramp to the Null Drive at a rate of 100 mV/ms if entering open loop because of a Soft Stop. If entering open loop because of an Open Loop (O) command, then the drive output ramps to the voltage specified by the Command Value of the command plus the Null Drive at a rate specified by the Acceleration and Deceleration command parameters.

Note: While an axis is in Open Loop mode it may drift due to a valve or motor 'out-of-null' condition.

Bit 2 - Halt

There are three conditions that will set this bit:

- A Halt (H) command is issued.
- An internal error causes a Soft Stop.
- An internal error causes a Hard Stop.

This bit is cleared when any of the commands listed above that clear error bits are issued.

One example of using this bit would be to detect when an internal error has caused a Hard Stop or an Open Loop Soft Stop, monitor for the following status bits:

Halt bit	ON (1)	(Bit 2)
Open Loop bit	ON (1)	(Bit 3)

Note: When an axis is halted (Halt bit ON, State A bit OFF and State B bit OFF) then the integral drive is automatically set to Null Drive and the Integrator stops updating. This is done so a Halt command can be given to the axes when the hydraulic system is turned off or the valves are disconnected so the module no longer can control the position. If the Integrator was allowed to update, the integral drive could become quite large and cause the axis position to jump when the hydraulic system was turned back on.

Bit 1 - Stopped

This bit is set when the actual speed for the axis—as computed and indicated by the Actual Speed status register—drops below 50 position units per second. This bit is cleared when the actual speed goes above 130 position units per second. The Stopped bit is not latched. Refer to the Actual Speed topic for details on how this value is computed. This bit can be used as an axis obstruction indication without having to monitor the Actual Speed register.

Bit 0 - In Position

This bit will be set when the difference between the Actual Position and Command Position is less than the value in the In Position field and the target generator is in a closed-loop stopped state. The axis must be in a closed-loop stopped state, which includes the following situations:

- At the end of a point-to-point move, such as those invoked by the Relative Move (J and j), Go (G and g, except for speed control and gearing), Move Relative to an Axis (0xC0-0xCF), and Sine Move (~) commands.
- When an axis follows a spline to the end following a Follow Spline (f) command.
- When the axis is stopped explicitly using a Set Parameters (P) or Reset Position (q) command.
- When the Target Speed reaches zero (0) after a speed control Go (G or g) command with a requested speed of zero.
- When the Target Position stops after a closed loop halt (caused by an Auto Stop or the Halt (H) command). Notice that the Command Position does not change in either of these cases, and therefore, unlike the above cases, the Target Position may not match the Command Position.

This bit is not latched. It will clear when the Actual Position moves outside the In Position window, or a command is issued that brings the axis out of the closed-loop stopped state.

Note: The control program can monitor this bit to determine when an axis has arrived at the commanded position or if the axis is holding position when a load is applied.

kadov_tag{ { }} Metadata type="DesignerControl" startspan Metadata type="DesignerControl" endspan

D.2.6 Status Word Bit Map (MDT, SSI, Analog, Resolver)

The axis Status word contains 16 bits of information about the status of the axis. The hexadecimal table below provides an easy way to convert hexadecimal numbers to bit patterns.

```
                        F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                        E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
   Hexadecimal To       D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
   Binary Conversion    C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
   Table                B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                        A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                        9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                        8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                        7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                        6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                        5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                        4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                        3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                        2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                        1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                        0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0
                           _______   _______   _______   _______
                          | | | | | | | | | | | | | | | | | | | |
                          |1|1|1|1| |1|1| | | | | | | | | | | | |
                          |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
   Bit Definition         |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
  ----------------         / / / /   / / / /   / / / /   / / / /
**No Tranducer ------ 15 -/ / / /   / / / /   / / / /   / / / /
**Transducer Noise -- 14 --/ / /   / / / /   / / / /   / / / /
**Transdcr Overflow - 13 ---/ /   / / / /   / / / /   / / / /
 *Overdrive --------- 12 ----/   / / / /   / / / /   / / / /
                                / / / /   / / / /   / / / /
 *Parameter Error --- 11 ------/ / / /   / / / /   / / / /
 *Position Overflow - 10 -------/ / /   / / / /   / / / /
 *Integrator Windup - 09 --------/ /   / / / /   / / / /
 *Following Error --- 08 ---------/   / / / /   / / / /
                                     / / / /   / / / /
  Acknowledge ------- 07 -----------/ / / /   / / / /
  Initialized ------- 06 ------------/ / /   / / / /
  State B ----------- 05 -------------/ /   / / / /
  State A ----------- 04 --------------/   / / / /
                                          / / / /
  Open Loop --------- 03 ----------------/ / / /
  Halt -------------- 02 -----------------/ / /
  Stopped ----------- 01 ------------------/ /
  In Position ------- 00 -------------------/
```

* Can cause a Soft or Hard Stop if the corresponding bits are set in the Auto Stop field.

** Will cause either a Soft or Hard Stop depending how Auto Stop bits 5, 6, and 7 are set.

D.2.7 Status Word Bit Map (Quadrature)

The axis Status word contains 16 bits of information about the status of the axis. The hexadecimal table below provides an easy way to convert hexadecimal numbers to bit patterns.

```
                          F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                          E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
  Hexadecimal To          D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
  Binary Conversion       C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
  Table                   B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                          A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                          9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                          8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                          7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                          6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                          5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                          4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                          3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                          2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                          1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                          0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0
                             _______   _______   _______   _______
                            | | | | | | | | | | | | | | | | | | | |
                            |1|1|1|1| |1|1| | | | | | | | | | | | |
                            |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
  Bit Definition            |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
 ----------------            / / / /   / / / /   / / / /   / / / /
**Encoder Fault ----- 15 -/ / / /   / / / /   / / / /   / / / /
**Extend/CW Limit --- 14 --/ / /   / / / /   / / / /   / / / /
**Retract/CCW Limit - 13 ---/ /   / / / /   / / / /   / / / /
 *Overdrive --------- 12 ----/   / / / /   / / / /   / / / /
                                / / / /   / / / /   / / / /
 *Parameter Error --- 11 ------/ / / /   / / / /   / / / /
 *Home Input -------- 10 -------/ / /   / / / /   / / / /
 *Integrator Windup - 09 --------/ /   / / / /   / / / /
 *Following Error --- 08 ---------/   / / / /   / / / /
                                     / / / /   / / / /
  Acknowledge ------- 07 -----------/ / / /   / / / /
  Initialized ------- 06 ------------/ / /   / / / /
  State B ----------- 05 -------------/ /   / / / /
  State A ----------- 04 --------------/   / / / /
                                          / / / /
  Open Loop --------- 03 ----------------/ / / /
  Halt -------------- 02 -----------------/ / /
  Stopped ----------- 01 ------------------/ /
  In Position ------- 00 -------------------/
```

* Can cause a Soft or Hard Stop if the corresponding bits are set in the Auto Stop field.

** Will cause either a Soft or Hard Stop depending how Auto Stop bits 5, 6, and 7 are set.

D.2.8 DRIVE

This field displays the drive output in millivolts. This field has a minimum value of 10,000, representing full negative drive of -10,000 mV and a maximum value of +10,000, representing a full positive drive of +10,000 mV. The value in this field is translated to the physical drive output using a 12-bit (4000-step) digital-to-analog converter (DAC), which will generate a ±10,000 mV output in steps of 5 mV. The internal drive calculations are done to 14-bit resolution. This additional resolution is used to dither the least significant bit of the output, giving additional resolution.

In many applications, the full ±10 V range is not used, and drive output beyond that range may even be unsafe. For this reason, the Limit Drive (L) command can be issued to limit the drive output range to be smaller than ±10 V.

Note: This field is unused on 12-bit analog channels or 16-bit analog channels that do not include channels 0 and 2. That is, on 16-bit analog modules, only channels 0 and 2 have drive associated with them. On auxiliary pressure/force channels, the drive is only affected by the Open Loop command; when controlling pressure/force, the position axis's drive is used.

Note: Software adjusts the null. There are no hardware nulling adjustments on the motion control module. Null drive will be near 0 volts.

Why Bother?

If the DRIVE tries to go below -10,000 or above 10,000, an overdrive condition has occurred.

D.2.9 Actual Speed

The Actual Speed is the calculated speed at which the axis is currently moving, as computed from the change in the Transducer Counts status register. The Actual Speed is in position units per second.

This value is filtered to reduce the effect of quantizing errors. In firmware dating 19991216 or newer, the Actual Speed is filtered by applying a low pass Infinite Impulse Response (IIR) filter with a 7 ms time constant. In earlier firmware, the filter consists of summing the change in position over the last four time periods and dividing by four time periods.

What is Quantizing Error?

Before the position of an axis can be used by a digital controller such as the RMC, its real-world, infinite-resolution, analog value must be converted through a transducer into an integer number of transducer counts. The loss of position resolution that occurs when truncating to integers is referred to as quantizing error.

Quantizing error affects the accuracy of positions, but it also affects the accuracy of speeds even more significantly. The following example demonstrates the problem quantizing errors pose to computing the Actual Speed.

The simplest way to compute the Actual Speed would be to divide the change in position units since the last sample by the sample time period, as shown below:

$$\text{ActSpd} = \frac{x(n) - x(n-1)}{T}$$

where:	x(n) is the position sampled this control loop
	x(n-1) is the position sampled last control loop

T is the sample period (control loop time)

Suppose that the sample period (control loop time) is 1 millisecond, the transducer returns one count for every thousandth of an inch, and the axis is moving at 4.8 inches per second (4800 counts/s).

As the controller samples the position each control loop, it will see the position change by either 4 or 5 position units, but never the actual rate of 4.8, because the transducer returns a fixed number of counts. Therefore, when these values are divided by 0.001 seconds, the computed Actual Speed will jump back and forth between 4000 and 5000, but will never show any value between.

The situation improves with higher speeds and higher-resolution transducers, but worsens with lower speeds and lower-resolution transducers. Filtering can be applied to smooth out the Actual Speed, as the RMC does, but the filtering must be limited to avoid smoothing out real changes in the Actual Speed.

D.2.10 NULL DRIVE

This field displays the amount of drive in millivolts that it takes to hold the Actual Position at the Target Position while at rest. Ideally this number should be zero if the valve is perfectly nulled.

In RMC100 CPU firmware versions earlier than 19980414, this field is automatically updated toward the Integral Drive when the Actual Position is exactly equal to the Command Position and the speed is zero. However, in later versions this feature was removed in favor of allowing the user to manually update the NULL DRIVE using either the Set Null Drive, Set Null Drive to Integral Drive, or Restore Null Drive commands.

This field is only used in Open Loop Mode. NULL DRIVE is added to the drive output.

Why Bother?

The NULL DRIVE field can be used to null valves and drives. Usually there is a null or offset adjustment on the valve or amplifier that can be adjusted so the NULL DRIVE is 0.

D.2.11 STEP

This field shows the current step the axis is executing of the 256 steps in the Event Step Table. This is valid only when Event Control is being used. Use this field in conjunction with the Link Value status word to monitor the progress of an axis through the step table.

D.2.12 LINK VALUE

This field is used only when the axis is using the Event Step Table. It represents the current event step link value of the current step for the axis. The current step is displayed in the STEP status word.

For DelayMS (D) and DelayTicks (d) link types, this field displays the number of delay units left (in either counter ticks or milliseconds).

D.3 Pressure/Force Status Fields

D.3.1 Command Pressure/Force

This field holds the requested pressure or force. It is set to the Command Value of the Set Pressure command.

D.3.2 Target Pressure/Force

Note: In this topic, the word pressure is used. If you are using differential force, you can replace every occurrence of pressure with force.

The Target Pressure is the calculated instantaneous ideal pressure of the axis. The Target Pressure is calculated every control loop by the target generator state machine of the RMC. The PID routine uses the difference between the Target Pressure and the Actual Pressure as its input to compute any required corrective drive.

During a move the transition of the Target Pressure toward the Command Pressure will be the perfect profile for the Actual Pressure to follow.

Why Bother?
Knowing the relationship between the Target Pressure and Actual Pressure is key to tuning the axis. The main goal in tuning the axis is to minimize the error between the Target and Actual Pressures. The plot function is a very useful visual aid in tuning the axis.

D.3.3 Actual Pressure/Force

This field gives the instantaneous pressure or force of this axis. This value is updated every control loop (1 or 2 milliseconds, depending on the module).

For single-ended pressure axes, this field is calculated as follows:

$$\text{Actual Pressure} = \text{Offset A} + \frac{\text{Counts} \times \text{Scale A}}{32768 \times \text{Prescale Divisor}}$$

For differential force axes, this field is calculated as follows:

$$\text{Actual Force} = \text{Actual Force A} - \text{Actual Force B}$$

D.3.4 Status (Pressure/Force)

The pressure Status word contains 16 bits of information about the condition of the axis. You can use any of the first eight error bits to trigger a STOP on the axis using the Auto Stop parameter. To display the expanded Status bit window, see Using the Status Bits Window, or click here for the Axis Status Bit Map.

Error bits 8 through 14 are cleared whenever a Set Pressure (^) command is given. Error bit 15 is cleared after a Set Parameters (P) command if a transducer is detected at that time.

Note: Bit 15 is the most significant bit (MSB; left-most bit), Bit 0 is the least significant bit (LSB, right-most bit).

Tip: On the Window menu, click Status Bits (or press CTRL+B) to display the Bit Status window, which shows the individual bits of the Status words.

Bit 15 - No Transducer

This bit is set if the transducer falls into a range which indicates that no transducer is connected. In all voltage modes, this occurs if the COUNTS read -32768. In 4 to 20mA mode, this occurs if the COUNTS read below 5898, which represents 3.6mA. This bit causes a Hard Stop or a Soft Stop, depending on the setting of Auto Stop bit 7. This bit will stay on until a new command is given to the axis.

Bit 14 - Transducer Noise

This bit is currently unused for analog transducers.

Bit 13 - Transducer Overflow

This bit indicates that the analog input is being over-ranged. Ensure that the analog input is between the range selected in the Configuration Word. For example, if the range selected is between 0 and 10V, and the input voltage is 12V. See Counts for a list of the actual count ranges.

Bit 12 - Overdrive

This bit is only used with 16-bit analog modules that have drive outputs. This bit is set when this drive output exceeds the 12-bit range of the D/A converter. This drive output can only be set by using the Open Loop command, and therefore this bit is not affected by the axis regulating pressure. The module will truncate the drive to 12 bits (+10V or -10V). It causes no action, a Soft Stop, or a Hard Stop, depending on the setting of Auto Stop bits 4 and 12. This bit will stay on until a new command is given to the axis. Because the drive output on the analog module generally does not have anything to do with regulating the pressure or position, it is recommended that no Auto Stop bits be set for this error.

Bit 11 - Parameter Error

This bit is set when an initialization parameter or control parameter is out of bounds. In some cases one parameter's limit will depend on the value of another parameter, so definite limits may not always be available. However, the motion controller does try to replace the erroneous value with another that is within range, so the offending parameter can be determined by comparing the parameter values before and after the error bit is set. This error causes no action, a Soft Stop, or a Hard Stop, depending on the setting of Auto Stop bits 3 and 11. This bit will stay on until a new command is given to the axis.

Bit 10 - Pressure Overflow

This bit indicates that the pressure read from the transducer does not fit within a 16-bit number. For single-ended analog axes, an overflow is not possible because the counts never exceed 16-bits. For differential analog axes, because one 16-bit number is subtracted from another, the total range is 17 bits, and therefore an overflow can occur. It causes no action, a Soft Stop, or a Hard Stop, depending on the setting of Auto Stop bits 2 and 10. This bit will stay on until a new command is given to the axis.

Bit 9 - Integrator Windup

This bit is set when the integrator value is larger than 20% or 80%, depending on the setting of the Integrator bit in the Configuration word. It causes no action, a Soft Stop, or a Hard Stop, depending on the setting of Auto Stop bits 1 and 9. This bit will stay on until a new command is given to the axis.

Bit 8 - Following Error

This bit is set when the difference between the Target Pressure and the Actual Pressure is greater than the Following Error parameter. It causes no action, a Soft Stop, or a Hard Stop, depending on the setting of Auto Stop bits 0 and 8. This bit will stay on until a new command is given to the axis.

Bit 7 - Acknowledge

This bit will toggle after the motion controller receives a valid command or Status Area Request. This can be used to verify that the motion controller has received the command.

Note: This bit only toggles on commands received from the Programmable Controller and therefore will not toggle on commands from RMCWin.

Bit 6 - Initialized

This bit is set after a Set Parameter (P) command is successfully executed. Until this bit is set, the axis will not regulate pressure. This bit is cleared when the module is reset.

Bit 4-5 - State Bits

Bits 4 and 5 show the state of the target generator:

STATE	Bit 5 (State Bit B)	Bit 4 (State Bit A)
Not Regulating Pressure	0	0

Increasing Pressure	0	1
Constant Pressure	1	0
Decreasing Pressure	1	1

Bit 1 - Regulating Pressure Bit

This bit is set when a pressure axis is actively controlling pressure. You may use this bit to determine when the axis has crossed into pressure mode out of position mode. This bit is not latched, and will clear as soon as the axis leaves pressure mode.

Bit 0 - At Pressure

This bit will be set when the difference between the Actual Pressure and Command Pressure is less than the value in the At Pressure field; it is not latched.

Note: The control program can monitor this bit to determine when an axis has reached the commanded pressure.

D.3.5 Status (Pressure/Force) Bit Map

The axis Status word contains 16 bits of information about the status of the axis. This hexadecimal table provides an easy way to convert hexadecimal numbers to bit patterns.

```
                        F  1 1 1 1   1 1 1 1   1 1 1 1   1 1 1 1
                        E  1 1 1 0   1 1 1 0   1 1 1 0   1 1 1 0
   Hexadecimal To       D  1 1 0 1   1 1 0 1   1 1 0 1   1 1 0 1
   Binary Conversion    C  1 1 0 0   1 1 0 0   1 1 0 0   1 1 0 0
   Table                B  1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                        A  1 0 1 0   1 0 1 0   1 0 1 0   1 0 1 0
                        9  1 0 0 1   1 0 0 1   1 0 0 1   1 0 0 1
                        8  1 0 0 0   1 0 0 0   1 0 0 0   1 0 0 0
                        7  0 1 1 1   0 1 1 1   0 1 1 1   0 1 1 1
                        6  0 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
                        5  0 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
                        4  0 1 0 0   0 1 0 0   0 1 0 0   0 1 0 0
                        3  0 0 1 1   0 0 1 1   0 0 1 1   0 0 1 1
                        2  0 0 1 0   0 0 1 0   0 0 1 0   0 0 1 0
                        1  0 0 0 1   0 0 0 1   0 0 0 1   0 0 0 1
                        0  0 0 0 0   0 0 0 0   0 0 0 0   0 0 0 0
                           _______   _______   _______   _______
                          | | | | | | | | | | | | | | | | | | | |
                          |1|1|1|1| |1|1| | | | | | | | | | | | |
                          |5|4|3|2| |1|0|9|8| |7|6|5|4| |3|2|1|0|
   Bit Definition         |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|
  ----------------         / / / /   / / / /   / / / /   / / / /
**No Tranducer ------ 15 -/ / / /   / / / /   / / / /   / / / /
**Transducer Noise -- 14 --/ / /   / / / /   / / / /   / / / /
**Transdcr Overflow - 13 ---/ /   / / / /   / / / /   / / / /
 *Overdrive --------- 12 ----/   / / / /   / / / /   / / / /
                                / / / /   / / / /   / / / /
 *Parameter Error --- 11 ------/ / / /   / / / /   / / / /
 *Pressure Overflow - 10 -------/ / /   / / / /   / / / /
 *Integrator Windup - 09 --------/ /   / / / /   / / / /
 *Following Error --- 08 ---------/   / / / /   / / / /
                                     / / / /   / / / /
  Acknowledge ------- 07 -----------/ / / /   / / / /
  Initialized ------- 06 ------------/ / /   / / / /
  State B ----------- 05 -------------/ /   / / / /
  State A ----------- 04 --------------/   / / / /
                                          / / / /
  Reserved ---------- 03 ----------------/ / / /
  Reserved ---------- 02 -----------------/ / /
  Pressure Mode ----- 01 ------------------/ /
  At Pressure ------- 00 -------------------/
```

* Can cause a Soft or Hard Stop if the corresponding bits are set in the Auto Stop field.

** Will cause either a Soft or Hard Stop depending how Auto Stop bits 5, 6, and 7 are set.

D.3.6 DRIVE

This field displays the drive output in millivolts. This field has a minimum value of 10,000, representing full negative drive of -10,000 mV and a maximum value of +10,000, representing a full positive drive of +10,000 mV. The value in this field is translated to the physical drive output using a 12-bit (4000-step) digital-to-analog converter (DAC), which will generate a ±10,000 mV output in steps of 5 mV. The internal drive calculations are done to 14-bit resolution. This additional resolution is used to dither the least significant bit of the output, giving additional resolution.

In many applications, the full ±10 V range is not used, and drive output beyond that range may even be unsafe. For this reason, the Limit Drive (L) command can be issued to limit the drive output range to be smaller than ±10 V.

Note: This field is unused on 12-bit analog channels or 16-bit analog channels that do not include channels 0 and 2. That is, on 16-bit analog modules, only channels 0 and 2 have drive associated with them. On auxiliary pressure/force channels, the drive is only affected by the Open Loop command; when controlling pressure/force, the position axis's drive is used.

Note: Software adjusts the null. There are no hardware nulling adjustments on the motion control module. Null drive will be near 0 volts.

Why Bother?

If the DRIVE tries to go below -10,000 or above 10,000, an overdrive condition has occurred.

D.3.7 Actual Force A, Actual Force B

Only differential force axes use these two status fields. They give the analog transducer reading after conversion from Counts to force units.

These fields are calculated as follows:

$$\text{Actual Force A} = \text{Offset A} + \frac{\text{Counts A} \times \text{Scale A}}{32768 \times \text{Prescale Divisor}}$$

$$\text{Actual Force B} = \text{Offset B} + \frac{\text{Counts B} \times \text{Scale B}}{32768 \times \text{Prescale Divisor}}$$

The Actual Pressure/Force status field is then calculated from the difference between the two as follows:

$$\text{Actual Force} = \text{Actual Force A} - \text{Actual Force B}$$

Appendix E: Event Step Link Reference

E.1 Link Types and Link Values

Link Type and Link Value specify the condition that causes the motion controller to execute the next step in a sequence. These fields may be edited manually; however in most cases it is easier to use the Link Type and Link Value dialog box.

To use the Link Type and Link Value dialog box:

1. Select the Link Type or Link Value field on the axis you want to edit.
2. Press ENTER. You can also double-click on either of these fields.
3. Under Link Type Category, select the appropriate category. See the list below describing each category and the link types under each.
4. If you selected the Selected Axis (Enhanced) link type category, then select the axis you want to monitor.
5. Under Link Type, select the particular link type you want to use. See the list below and associated topics on each link type.
6. Under Link Condition, enter the settings for the particular link type selected.
7. The actual Link Type and Link Value that will be entered are displayed in the Link Type and Link Value dialog box to the left of the OK button. These values are updated as you make changes.
8. Click OK.

When viewed from a communication module (such as PROFIBUS-DP, Ethernet, etc.), the Link Type and Link Next fields are combined into a single 16-bit register, as detailed in the Link Type/Next topic. From PROFIBUS-DP Compact Mode and DI/O Command Mode, the Event Step Edit and Event Step Transfer commands are used to set this value. For all other communication types, they are accessed like any other register. Use the appropriate register map or use the Address Tool to find addresses of these values.

The Link Type and Link Value dialog box divides the link types into three categories:

- **System-wide (Basic, non-axis dependent)**

These link types allow the axis to wait on a global condition such as a time delay or discrete input. These link types are listed below; select a link type for further details:

Link Type	**Description**
End of Sequence	This link type stops the event sequence.
Comm Trigger	Wait for synchronization signal from the communications master.

Delay	Wait for either a number of milliseconds or a number of counts on the edge or quadrature counter (if available).
Error Check	Wait for any of one or more errors on one or more axes.
Inputs, Multiple (Level only)	Wait for multiple discrete inputs.
Inputs, Single (Level/Edge)	Wait for a single discrete input edge or level.
Jump Using Inputs	Jump immediately to one of four event steps.
Loop	Define a loop in the step table.
Math Compares/Errors	This family of link types evaluate the results of one of the math commands (Add, Subtract, and MulDiv).
Multiple Axes In Position	Wait for several axes to be simultaneously in position.
Skew	Wait until the difference in actual positions or (following error) between axes exceeds a limit.
Timer	Start or check a timer.
Check Wait Bits	Check the Wait Bits.

- **Current Axis (Basic)**

These link types allow the axis to wait for a condition based on its own state. These link types are listed below; select a link type for further details:

Link Type	**Description**
Absolute Limit Switch	Wait for the position to cross a specified value.
Pressure	Wait for the pressure (of either this axis, or the auxiliary pressure channel assigned to this axis) to cross a specified value.
Relative Limit Switch	Wait for the position to cross a position specified as a distance from either the start or end of the current move.
Speed	Wait for the speed to be above or below a specified value.
Status Bits	Wait for one or more status bits to be on or off.

- **Selected Axis (Enhanced)**

These link types allow the axis to wait for conditions on another axis to occur before proceeding. These link types are listed below; select a link type for further details:

Link Type	**Description**
Position/Pressure	Wait for the position on an axis to be above or below a specified value.
Speed	Wait for the speed on an axis to be above or below a specified value.
Status Bits	Wait for one or more status bits on an axis to be on or off.

E.2 Link Next

The Link Next field contains the number of the next step that will execute when the condition of the Link Type and Value are met.

The Link Next field can point to any step (from 0 to 255). When a chain of steps is finished the last step should have a Link Next of 0 to indicate so. If a never-ending loop is required, the last step will have a Link Next value of a step earlier in the chain.

When the Link Next field is read or written through PROFIBUS-DP Compact Mode and DI/O Command Mode, it is combined with the Link Type field. See the Link Type/Next field for details on how they are combined, and the Event Step Edit and Event Step Transfer command topics for details on changing these fields from the master controller.

Tip: When editing the steps, you can press CTRL+G to cause the cursor to jump to the step indicated by the Link Next field. This allows the user to view the steps in a step sequence.

E.3 Commanded Axes

As described in Event Control Overview, each event step contains a command and a link section. The command, by default is issued only to the axis running the event sequence. In this case, the Commanded Axes field will display "Default."

With RMC CPU firmware dated 19980414 or later, this field can be changed to select one or more specific axes to receive the command. There are three ways of changing this field:

1. Type directly in the Commanded Axes field. Enter the axes numbers separated by spaces or commas, and use a hyphen (-) to indicate a range of axes (e.g. the first four axes are indicated by 0-3; the second and fourth axes are indicated by 1,3; etc.).
2. Use the Commanded Axes popup editor. See Using Popup Editors for details on opening the editor. Once the editor is open, either click Use axis running event sequence, or click Use axes selected below and check the boxes next to the names of the axes you want to command.
3. With most communication types, you can change the event steps from the programmable controller. Refer to the section of the communication type you are using for details on changing event steps. The Commanded Axes field is stored in the most significant byte of the Command field. See Command/Commanded Axes for a bit map.

E.4 System-wide Link Types

E.4.1 Link Type - End of sequence

Link Type:	**0 (hex 0x00, dec 0)**
Link Value:	**Reserved - must be 0**

When this Link Type is encountered the step sequence stops: no additional steps are executed unless they are started as a new sequence. We recommend using Step 0 as the last Step of all chains so they all end in a known spot. If Step 0 is used this way, its link type must be zero.

E.4.2 Link Type - Comm Trigger

Link Type:	**CommTrig (C, hex 0x43, dec 67)**
Link Value:	**Sync Value**
Range:	**0 to 65,535**

Note: This link type is available only in RMC100 CPU firmware version 20010123 and later.

This link type is used to synchronize execution of the step table with changes made externally from the PLC or other controlling system (called master below). This link type pauses the step-sequence execution until the master indicates it is ready by changing the Link Value or Extended Link Value. This function is useful when the master updates information in the RMC each machine cycle.

The RMC maintains an internal Extended Link Value for each axis; each is initialized on RMC power-up to 0. When the CommTrig (C) link type is encountered, the Link Value is compared with the Extended Link Value for the axis running the event sequence:

- If they are equal: No action is taken; the event sequence will continue to wait for a difference.
- If they are not equal: The Link Value will be copied into the Extended Link Value for the axis running the event sequence, and the link will be taken to the step indicated by the Link Next field.

There are two ways to trigger this link type to proceed:

- Change the Extended Link Value: This can be done by issuing a Set Extended Link Value (l) command. When this value is changed, the link type will reset it back to the Link Value, so it is possible to trigger using the same value each time, as long as it is different from the Link Value.
- Change the Link Value: This can be done by writing directly to the event step's Link Value field

from the master. When this value is changed, the link type copies it into the Extended Link Value, so a new value must be written to the Link Value each time.

Note: The Extended Link Value is also used by the Skew Detection (<) link type. Care must be taken to ensure that these link types are not used on the same axis.

Note: Because the Extended Link Value powers up with 0, starting with a Comm Trigger (C) link value of 0 will cause the event sequence to wait the first time it encounters this link type.

Example:

	Step 10	**Step 11**	**Step 12**
Mode	0x0000	0x0001	0x0001
Accel	0	10	10
Decel	0	10	10
Speed	0	1000	1000
Command Value	0	4000	5000
Command		G	G
Commanded Axes	Default	Default	Default
Link Type	CommTrig	BitsON	BitsON
Link Value	0	1	1
Link Next	11	12	10

This example is an axis making two moves. Suppose the PLC wants to control these moves by downloading the step sequence to the RMC. Here is how this can be done:

- The RMC starts this sequence running on axis 0. That axis will first encounter the CommTrig (C) link type and wait until the Extended Link Value or Link Value change, since both are zero.
- The PLC waits until the axis is not running either step 11 or 12.
- The PLC downloads values to these two steps. These values can include, but are not limited to, accelerations, speeds, and positions.
- The PLC issues a Set Extended Link Value command with a command value of 1 to axis 0. Alternatively, the PLC could change the Link Value in the event step table itself, but it would need to write a different value each time.
- The CommTrig (C) link type sees that the Extended Link Value differs from the Link Value of 0, so it copies the Link Value of 0 into the Extended Link Value and jumps to step 11.

- Steps 11 and 12 are executed normally.
- When step 10 is reached, the event sequence pauses again, waiting for the PLC to re-trigger the sequence.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select System-wide (Basic, non-axis).
2. Under Link Type, select Comm Trigger.
3. Under Link Condition, enter the Sync Value you wish to start with.
4. Click OK.

Using without the Link Type and Link Value Dialog Box

1. Enter a 'C' into the Link Type field.
2. Enter the Link Value.

E.4.3 Link Type - Delay

Link Type:	**DelayMS (D, hex 0x44, dec 68)** **DelayTicks (d, hex 0x64, dec 100)**
Link Value:	**For DelayMS, number of milliseconds to delay** **For DelayTicks, number of counter ticks to delay**
Range:	**0 to 65,535**

Note: These link types cannot be used with the Poll (?) command. Both link types will continuously restart they countdowns. For measuring time while polling, use the Timer (T and t) link types.

These two link types are used to delay based on time or on counter ticks.

- **Delaying Based on Time**
  When delaying based on time, this link type will wait until the time specified in milliseconds in the Link Value field expires. It is common to use this link type with a Link Value of 0 ms to link immediately to the next step in the sequence.
  The Timer (T and t) link types offer similar functionality. However, while the Time Delay link type starts the timer when the event step is reached, the Timer (T and t) link types separate the starting and checking of the timer into two or more steps. Use the DelayMS (D) link type when you want to wait a certain fixed amount of time from the start of the current step before executing the next step, and use the Timer (T and t) link types when you want to wait a certain amount of time since an event earlier in the sequence, or when you want to polling several events including a timeout. See Timer (T and t) link types for more details.
- **Delaying Based on Counter Ticks**
  This link type uses a counter that may be enabled on either the Communication Digital I/O or Sensor Digital I/O module. Refer to those topics for details on selecting an edge or quadrature

counter.
If a counter is enabled, this link type will wait until the specified number of ticks have occurred on the counter and then proceed to the next event step in the sequence.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select System-wide (Basic, non-axis).
2. Under Link Type, select Delay.
3. Under Link Condition, select Delay based on time or Delay based on counter ticks.
4. Enter the number of milliseconds or counter ticks you want to wait for.
5. Click OK.

Using without the Link Type and Link Value Dialog Box

1. To delay based on time, enter a 'D' in the Link Type field. To delay based on counter ticks, enter a 'd' in the Link Type field.
2. Enter the number of milliseconds or counter ticks to wait in the Link Value field.

E.4.4 Link Type - Error Check on Multiple Axes

Link Type:	**ErrorCheck (E, hex 0x45, dec 69)**
Link Value:	**Indicates Which Errors to Monitor on Which Axes**
Range:	**All Values**

Note: This link type is available only in RMC100 CPU firmware version 20000905 and later.

This link type is used to monitor one or more axes for errors. Typically, this monitoring is done with a separate event sequence from the main event sequences. For example, axis 0 runs an event sequence that moves axes 0 and 1. Axis 1 runs an error monitoring sequence that starts with this link type. Therefore, the error monitoring sequence will only take action if an error is detected. The action taken depends on the commands that follow the ErrorCheck (E) link type.

This link type encodes which of the eight error Status bits to monitor and which axes to monitor. Multiple error bits and multiple axes can be specified. If any of the selected errors occur on any of the selected axes, then the link will be taken. See the discussion below on how to set up this link type from RMCWin.

Example:

	Step 10	Step 11	Step 12	Step 20	Step
Mode	0x0001	0x0001	0x0000	0x0000	0x0
Accel	10	10	0	0	

Decel	10	10	0	0	
Speed	1000	7500	0	0	
Command Value	4000	8000	0	0	
Command	G	G			
Commanded Axes	0-1	0-1	Default	Default	
Link Type	AxesInPos	AxesInPos	DelayMS	ErrorCheck	Delay
Link Value	0x0003	0x0003	250	0xFF03	
Link Next	11	12	10	21	

Normally, axes 0 and 1 will make two moves together, then delay for a quarter second, and then repeat the process. If an error occurs on either axis, the user wants both axes to halt. Here is how this is done using the ErrorCheck (E) link type:

- Axis 0 starts running the event sequence at step 10. This is the main sequence which will cause the axes to move during normal machine behavior.

  This sequence does the following:

  - Step 10 moves both axes to position 4000 at a speed of 1000 position units per second. It then waits for axes 0 and 1 to be in-position.
  - Step 11 moves both axes to position 8000 at a speed of 7500 position units per second. It then waits for axes 0 and 1 to be in-position.
  - Step 12 waits 250 ms, and then repeats the sequence starting with step 10.

- Axis 1 starts running the event sequence at step 20. This is the error monitoring sequence.

  This sequence does the following:

  - Step 20 waits for any error bit on axes 0 or 1. No command is issued by this step.
  - Step 21 issues a Halt (H) command to axes 0 and 1.

- The idea is that the error handling logic can be separated from the main sequencing. Therefore, as long as no errors occur, the sequence running on axis 1 (steps 20-21) will do nothing, so only the main sequencing running on axis 0 (steps 10-12) will be running. If an error occurs, axis 1 will jump to step 21, which halts the moves on both axes.

This particular example could also be done with the RMC's Synchronized Move feature. However, the ErrorCheck (E) link type is much more flexible. For example, it could also do the following:

- The error response handling could be expanded by adding steps after step 21 to do such things as turning a discrete output on or off.
- The error response handling need not halt the axes, but might instead move the axes back to a home position.

- The main sequence could be much more sophisticated, such that the axes do not always move together, whereas the Synchronized Move feature is very specific in the types of moves that can be made.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select System-wide (Basic, non-axis).
2. Under Link Type, select Error Check.
3. Under Link Condition, check the box next to each error you want to monitor.
4. Under Link Condition, check the box next to each axis you want to monitor.
5. Click OK.

Using without the Link Type and Link Value Dialog Box

1. Enter an E into the Link Type field.
2. Enter the Link Value in hexadecimal. You must understand binary and hexadecimal to enter this value manually. If you do not understand binary (actually, even if you do), you should use the Link Type and Link Value dialog box to edit these link types.
   - Bits 0-7 of the Link Value correspond to axes 0 through 7. Set the bit for each axis you want to monitor to 1.
   - Bits 8-15 of the Link Value correspond to Status word bits 8-15. Set the bit for each error you want to monitor to 1.

E.4.5 Link Type - Inputs, Multiple (Level only)

Link Type:	**InpAllON ([, hex 0x5B, dec 91) - All Inputs On** **InpAllOFF (], hex 0x5D, dec 93) - All Inputs Off** **InpAnyON ({ , hex 0x7B, dec 123) - Any Inputs On** **InpAnyOFF (}, hex 0x7D, dec 125) - Any Inputs Off** **InpAllMIX (\, hex 0x5C, dec 92) - All Inputs, Mixed On/Off** **InpAnyMIX (|, hex 0x7C, dec 124) - Any Input, Mixed On/Off**
Link Value:	**Mask of inputs to wait for**
Range:	**Depends on Configuration**

Note: This feature is available only in RMC100 CPU firmware version 19990625 and later.

These link types wait for a combination of level events on discrete inputs. These link types are limited to 16 inputs if all the events are of one polarity or to 8 inputs if both ON and OFF input levels are used. The inputs used depend on the RMC configuration according to the following rules:

- If a Sensor DI/O is present, Sensor DI/O inputs 0-15 (or 0-7) are used.
- If no Sensor DI/O is present, but a Communication DI/O is present, Communication DI/O inputs 0-15 (or 0-7) are used.
- If neither Sensor nor Communication DI/O is present, then only CPU inputs 0-1 are used.

This link type is similar to Inputs, Single (Level/Edge) with the following differences:

- **Inputs, Multiple:** allows up to 16 inputs to be evaluated at once.
  Inputs, Single: allows only one (1) input to be evaluated.
- **Inputs, Multiple:** allows waiting for level events only.
  Inputs, Single: allows waiting for either a level or edge event.
- **Inputs, Multiple:** for DI/O modules, usable inputs are limited to inputs 0-15 of one DI/O module.
- **Inputs, Single:** any discrete input can be used on any DI/O module, except the CPU inputs when the Communication DI/O is installed.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select System-wide (Basic, non-axis).
2. Under Link Type, select Inputs, Multiple (Level only).
3. Under Link Condition, select whether you want to wait for all of the input events you are going to enter, or any one of the events you will enter.
4. Under Link Condition, for each input, select whether you wish to have that input high, low, or ignored.

Example:

If you wish to wait for both inputs 0 and 1 to be high, select that you want all of your input events to be satisfied, for inputs 0 and 1, select the 1 option, and for all other inputs, select the X option.

Using without the Link Type and Link Value Dialog Box

1. Select the appropriate link type:
   - If you want to wait for multiple inputs to be ON at once, select '['.
   - If you want to wait for multiple inputs to be OFF at once, select ']'.
   - If you want to wait for any one of several inputs to be ON at once, select '{ '.
   - If you want to wait for any one of several inputs to be OFF at once, select '}'.
   - If you want to wait for multiple input conditions including both ONs and OFFs, select '\'.
   - If you want to wait for any one of several input conditions including both ONs and OFFs, select '|'.
2. To enter the link value, you must understand binary. If you do not understand binary (actually,

even if you do), you should use the Link Type and Link Value dialog box to edit these link types.

- If you selected a link type of '{ ', '}', '[', or ']', each of the sixteen bits (number 0 to 15, right to left) correspond to the sixteen inputs. Enter a 1 in each input's bit to that you care about (e.g., if you want inputs 2 and 4 to be off, the link value should be 0x000A).
- If you selected a link type of '\' or '|', then bits 0-7 correspond to the OFF level of inputs 0-7. Enter 1's in the bits of the inputs that must being OFF. Bits 8-15 correspond to the ON level of inputs 0-7. Enter 1's in the bits of the inputs that must be ON.

Example 1:

Link Type: InpAllMIX

Link Value: 0x0803

Link Next: 11

This link will check inputs 0, 1 and 3. Input bits 0 and 1 must be off and input bit 3 must be on before going to the next step.

Example 2:

Link Type: InpAnyMIX

Link Value: 0x0803

Link Next: 11

This link will check inputs 0, 1 and 3. If input bit 0 is off, bit 1 is off, or bit 3 is on then execute the next state.

E.4.6 Link Type - Inputs, Single (Level/Edge)

Link Type:	**InputRise (I, hex 0x49, dec 73) - Rising Edge** **InputFall (i, hex 0x69, dec 105) - Falling Edge** **InputHigh (O, hex 0x4F, dec 79) - Level High** **InputLow (o, hex 0x6F, dec 111) - Level Low**
Link Value:	**Discrete Input number to wait for**
Range:	**Depends on Configuration**

Note: The input level portion of these link types is available only in RMC100 CPU firmware version 19990625 and later.

Note: The Rising (I) and Falling Edge (i) link types should not be used with the Poll (?) command. This is because the edges can be missed during the polling cycle since these link types only check for an edge in the last control loop. You should instead use the Level High (O) and Level Low (o) link types when polling.

These link types wait for an event on a single discrete input. This event can be an edge (rising or falling) or level (high or low). This link type may use all inputs unless the a Communication DI/O is installed, in which case the CPU inputs may not be used.

This link type is similar to Inputs, Multiple (Level only) with the following differences:

- **Inputs, Multiple:** allows up to 16 inputs to be evaluated at once.
  Inputs, Single: allows only one (1) input to be evaluated.
- **Inputs, Multiple:** allows waiting for level events only.
  Inputs, Single: allows waiting for either a level or edge event.
- **Inputs, Multiple:** for DI/O modules, usable inputs are limited to inputs 0-15 of one DI/O module.
  Inputs, Single: any discrete input can be used on any DI/O module, except the CPU inputs when the Communication DI/O is installed.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select System-wide (Basic, non-axis).
2. Under Link Type, select Inputs, Single (Level/Edge).
3. Under Link Condition, select the type of the event you want to wait for (rising or falling edge, or on or off level).
4. Under Link Condition, select the device whose discrete input you wish to use (either CPU, Communication DI/O, or Sensor DI/O).
5. Under Link Condition, select the input number on that device.

Using without the Link Type and Link Value Dialog Box

1. Select a Link Type:
   Use 'I' to wait for a rising edge, 'I' to wait for a falling edge, 'O' to wait for an input to be on, or 'o' to wait for an input to be off.
2. Calculate a Link Value:
   This will select the input to use. Begin with the input number from the device (0 17 for DI/O modules or 0-1 for the CPU). If the input is on a Sensor DI/O, add 32 to the input number. Use this value for the Link Value (e.g., CPU input 1 will use a Link Value of 1, Sensor DI/O input 4 will use a Link Value of 32 + 4 = 36).

Example 1:

Link Type: InputHigh (O)

Link Value: 16

Link Next: 11

This link waits until input bit 16 of the Communication DI/O is ON before going to step 11.

Example 2:

Link Type: InputLow (o)

Link Value: 0

Link Next: 15

This link waits until input bit 0 of the Communication DI/O (or CPU if no Communication DI/O is present) is OFF before going to step 15.

E.4.7 Link Type - Jump Using Inputs

Link Type:	**Jump (J, hex 0x4A, dec 74)**
Link Value:	**Link Value: Reserved – must be 0**
Range:	**Reserved – must be 0**

Note: This feature is available only in RMC100 CPU firmware version 19990625 and later.

Note: This link type cannot be used with the Poll (?) command.

This link type is unique in that both the Link Value and Link Next fields are ignored. When this link type is encountered two discrete inputs are used to determine which event step is jumped to. This link is taken immediately; no condition is waited for.

The inputs used depend on the RMC hardware available. If there is a Sensor DI/O, its inputs 16 and 17 are used. If there is no Sensor DI/O, but there is a Communication DI/O, its inputs 16 and 17 are used. If there is no DI/O module, then CPU inputs 0 and 1 are used.

This chart shows how the input levels determine the next event step:

Input		
17 or 1	**16 or 0**	**Next Event Step**
0	0	Current Event Step + 1
0	1	Current Event Step + 2
1	0	Current Event Step + 3

1	1	Current Event Step + 4

Note: If the next Event Step would be greater than 255, then the event step will wrap around to step 0. For example, if the current event step is 254 and inputs 17 and 16 are 1 and 0 respectively, then "Current Event Step + 3" would be 257, but this instead wraps around to event step 1. This situation is confusing and should be avoided when designing your event step sequences.

Example:

	Step 20	**Step 21**	**Step 22**	**Step 23**	**Step 24**
Mode	0x0081	0x0081	0x0081	0x0081	0x0081
Accel	100	100	100	100	100
Decel	100	100	100	100	100
Speed	1000	1000	1000	1000	1000
Command Value	0	0	0	0	0
Command					
Commanded Axes	Default	Default	Default	Default	Default
Link Type	Jump	DelayMS	DelayMS	DelayMS	DelayMS
Link Value	0	0	0	0	0
Link Next	0	30	40	50	60

When the RMC reaches step 20, it looks at the Jump (J) link type and checks inputs 16 and 17 (or CPU inputs 0 and 1 if no DI/O modules are present). If input 17 is off and input 16 is on, it will jump to step 22. If input 16 and 17 are both off, then it will jump to step 21.

Why Bother?

This is a way to select and execute different programs or recipes within the RMC.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select System-wide (Basic, non-axis).
2. Under Link Type, select Jump Using Inputs.

Using without the Link Type and Link Value Dialog Box

1. Enter a 'J' for the Link Type.
2. Enter a 0 in the Link Value.

E.4.8 Link Type - Loop

Link Type:	**LoopStart (#, hex 0x23, dec 35)** **LoopEnd (-, hex 0x2D, dec 45)**
Link Value:	**Number of loop iterations**
Range:	**0 to 65,535**

Note: This feature is available only in RMC100 CPU firmware version 19980811 and later.

Note: This link type cannot be used with the Poll (?) command.

The LoopStart (#) and LoopEnd (-) link types are used together to create loops in the Event Step table. The Link Value of the LoopStart link type indicates how many iterations you want through the loop. The LoopStart link type jumps immediately to the step indicated by the Link Next. This should typically indicate the step of the LinkEnd link type.

The LoopEnd link type tests the loop counter to see if it zero. If it is zero then the next step—not the step specified in the Link Next field—will be executed. If the loop counter is not zero then it is decremented by one and the step specified in the Link Next field is jumped to. The Link Value on the LoopEnd link type is ignored and should be set to zero (0).

Example:

The following example executes steps 2 though 5 three times. After the third loop step 7 is executed.

	Step 1	**Step 2**	**Step 3**	**Step 4**	**Step 5**	**St**
Mode	0x0081	0x0081	0x0081	0x0081	0x0081	0x(
Accel	100	100	100	100	100	
Decel	100	100	100	100	100	
Speed	10000	10000	10000	10000	10000	1(
Command Value	4000	10000	10000	4000	4000	4
Command		G		G		
Commanded Axes	Default	Default	Default	Default	Default	De
Link Type	LoopStart	BitsON	DelayMS	BitsON	DelayMS	Loop

Link Value	3	1	1000	1	1000
Link Next	6	3	4	5	6

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select System-wide (Basic, non-axis).
2. Under Link Type, select Loop.
3. Under Link Condition, select Start a Loop or End a Loop.
4. If you selected Start a Loop, enter the number of iterations you want to use.

Using without the Link Type and Link Value Dialog Box

1. To start a loop, enter '#' for the Link Type. To end a loop, enter '-' for the Link Type.
2. If you started a loop, enter the number of iterations in the Link Value. Otherwise enter 0 in the Link Value.

E.4.9 Link Type - Math Compares/Errors

Link Type:	**EQ (hex 0x10, dec 16) - Equals** **NEQ (hex 0x11, dec 17) - Not Equals** **MathOK (hex 0x12, dec 18) - No Math Error** **MathERR (hex 0x13, dec 19) - Math Error** **LES (hex 0x14, dec 20) - Less Than or Equal, Signed** **LTS (hex 0x15, dec 21) - Less Than, Signed** **GTS (hex 0x16, dec 22) - Greater Than, Signed** **GES (hex 0x17, dec 23) - Greater Than or Equal, Signed** **LEU (hex 0x18, dec 24) - Less Than or Equal, Unsigned** **LTU (hex 0x19, dec 25) - Less Than, Unsigned** **GTU (hex 0x1A, dec 26) - Greater Than, Unsigned** **GEU (hex 0x1B, dec 27) - Greater Than or Equal, Unsigned** **LEP (hex 0x1C, dec 28) - Less Than or Equal, Position** **LTP (hex 0x1D, dec 29) - Less Than, Position** **GTP (hex 0x1E, dec 30) - Greater Than, Position** **GEP (hex 0x1F, dec 31) - Greater Than or Equal, Position**
Link Value:	**Value to compare against (unused by MathOK and MathERR)**
Range:	**-32,768 to 32,767 for EQ, NEQ, LES, LTS, GTS, GES** **0 to 65,535 for LEU, LTU, GTU, GEU** **Valid 16-bit Position for LEP, LTP, GTP, and GEP**

Note: These link types are available only in firmware version 20011113 and later.

This family of link types is used to analyze the results of the RMC's three math commands: Add

(+), Subtract (-), and MulDiv ('). These link types evaluate the results of the last math command that was issued on the axis running the event sequence. Therefore, these link types can be used any time after the math command is issued (including on the same step) up until another math command is issued on that axis.

All of these link types are non-blocking. That is, if the condition they check for is true, then they jump to the step referenced by the Link Next field. However, if the condition they check for is false, then they jump to the next step.

These link types can be divided into three groups:

- **Detecting Math Overflows**
  The MathERR (0x13) and MathOK (0x12) link types jump to the Link Next step if there was or was not respectively an overflow in the last math command. Refer to each of the math commands for details on when overflows will occur.

- **Detecting Equality**
  The EQ (0x10) and NEQ (0x11) link types are used to compare the results of the last math command with the Link Value. The Link Value will be displayed in RMCWin as a signed 16-bit value. However, because the sixteen bits either match between the result and the Link Value or they do not, these link types can be used to compare with unsigned or position values, although the Link Value may be displayed incorrectly in RMCWin.

- **Detecting Less Than or Greater Than**
  To determine if one value is greater or less than another, one of twelve link types should be used. There are four comparison types (>, <, ³, and £) for each of three ranges (signed, unsigned, and position). The range refers to how the 16-bit result from the last math command and the 16-bit Link Value are interpreted:

  - For the signed compares, each is treated as a signed 16-bit number with a range of -32,768 to 32,767. Use LTS, LES, GES, and GTS for signed compares.

  - For unsigned compares, each is treated as an unsigned 16-bit number with a range of 0 to 65,535. Use LTU, LEU, GEU, and GTU for unsigned compares.

  - For position compares, each is treated as having the same range as an axis's position unit range. The axis whose position range will be used is selected with bits 4-6 of the MODE word on the link type's step. These are the same bits used to select the axis for the math commands themselves. Use LTP, LEP, GEP, and GTP for position compares.

  If there is any chance of an overflow in the math operation, then you must first use a MathERR or MathOK link type to handle the case where there was an overflow, and then do any compares you want to do with the value. Comparing an overflowed value will likely give incorrect or unexpected results.

Note: The Link Value is always treated as a constant when used in a comparison. However, you are not limited to comparing with a constant. You can use math commands to compute a value or copy a register into the Link Value field of the step that has the comparison link type.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select System-wide (Basic, non-axis).

2. Under Link Type, select Math Compares/Errors.

3. Under Link Condition, select whether you want to jump on a successful math operation, an overflow, or on a comparison.

4. If you selected to jump on a comparison, then select the comparison type (=, <, >, etc.) and enter the value to compare with.

5. If you selected a relational comparison (any comparison other than equal or not equal), then select the type of the result and compare values (signed, unsigned, or position units).

6. If you selected a relational comparison for position values, then select the axis whose position unit range you want to use.

Note: This axis is encoded in bits 4-6 of the step's Mode word. Therefore, changing this value will change the Mode for the event step. Ensure that this is acceptable for the command on that event step. If it is not, then you will need to move the link type to another step.

Using without the Link Type and Link Value Dialog Box

1. In the Link Type field, enter one of the link types listed at the top of this topic. You must enter it in hexadecimal.

2. For the MathERR and MathOK link types, enter 0 for the Link Value. Otherwise, enter the value you want to compare with.

E.4.10 Link Type - Multiple Axes In Position

Link Type:	**AxesInPos (A, hex 0x41, dec 65)**
Link Value:	**Bits 0-7 select axes to be monitored.**
Range:	**Any combination of bits 0-7. Bits 8-15 are reserved.**

Note: This feature is available only in RMC100 CPU firmware version 19990625 and later.

This link type waits for all selected axes to be either in position (for position axes) or at pressure (for pressure axes) before going to the next step. This link type will work with both types at the same time.

Example:

Link Type: AxesInPos (A)

Link Value: 0x000C

Link Next: 10

On a four-axis system this link would wait for the third and fourth axis to be in position before going to step 10. On a two axis systems with two pressure channels this link would wait for the two pressure channels to be at pressure before going to step 10. On a system with just two axes this would be a mistake and the RMC would never link to step 10.

Why Bother?

This is a GREAT way to get axes synchronized at startup and other places within a machine cycle.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select System-wide (Basic, non-axis).
2. Under Link Type, select Multiple Axes In Position.
3. Under Link Condition, check the boxes next to each axis that you want to be in-position before proceeding to the next event step.

Using without the Link Type and Link Value Dialog Box

1. Enter an 'A' into the Link Type field.
2. Enter the Link Value
   You must understand binary to enter the Link Value manually. Each bit of the Link Value corresponds to an axis. Bit 0 corresponds to axis 0, bit 1 corresponds to axis 1, etc. Bits 8-15 are always unused. The RMC will wait for all axes corresponding to the enabled bits to be in position (or at pressure) before continuing.

E.4.11 Link Type - Skew Detection

Link Type:	**Skew on Actual Position (<, hex 0x3C, dec 44)** **Skew on Following Error (hex 0x7F, dec 127)**
Link Value:	**Bits 0-7 select axes to be monitored.**
Range:	**Any combination of bits 0-7. Bits 8-15 are reserved.**

Note: The Skew on Actual Position link type is available only in RMC100 CPU firmware version 20000504 and later. The Skew on Following Error link type is available only in RMC100 CPU firmware version 20030916 and later.

What is Skew?

Skew is calculated over a group of axes. The minimum and maximum actual positions (or following errors) in that group are found, and the difference is calculated. The difference between the highest and lowest actual position (or following error) is the skew. The maximum detectable skew is 32,767 position units. This link type can also detect differences between pressure or force in selected axis, in which case the maximum detectable difference is 32,767 pressure or force units.

The RMC100 has two skew link types: Skew on Actual Position and Skew on Following Error.

Skew on Actual Position

This link type compares the Actual Positions of each axis to detect skew. For example, if four axes have positions 1000, 1010, 998, 1005, then the minimum and maximum positions would be 998 and 1010, so the skew would be 12 position units. For the skew on actual position to be meaningful, the axes must be configured to have the same position units for the same positions. For example, each of the four corners may be configured to have 0 be fully closed, 1000 be 1.000 inch above fully closed, etc.

Skew on Following Error

This link type compares the Following Errors of each axis to detect skew. In other words, each axis must be just as far ahead or behind its respective Target Position. This allows the target positions to be different on each axis.

This type of skew is useful if the axes cannot be configured to have the same position units for the same positions, or if they are offset.

Detecting Excessive Skew

This link type is used to detect excessive skew in a group of axes. This link type must be used in conjunction with the Set Extended Link Value (l) command. The skew limit is set with the Set Extended Link Value command, and then this link type is used with its link value defining the axes to be included in the skew calculation. When the skew among these axes exceeds the skew limit set in the Extended Link Value, this step sequence will jump to the next step in the sequence. Otherwise, the step sequence will continue to wait for the condition to be true.

Because this link type waits until the skew is excessive, it is usually used to detect an error condition, and therefore probably is running in a step sequence on an axis other than your primary control step sequence is running. For example, axis 0 may be running the standard step sequence, while axis 1 is waiting to see if the skew gets too large. When it detects this condition it will stop axis 0's step sequence and put the machine in a safe state.

Example:

	Step 10	**Step 11**
Mode	0x0000	0x0001
Accel	0	10
Decel	0	10
Speed	0	1000
Command Value	100	0
Command	l (lower case L)	K
Commanded Axes	Default	0-1
Link Type	Skew	DelayMS
Link Value	0x0003	0
Link Next	11	0

This example assume that axes 0 and 1 are moving together with axis 0 running the primary control step sequence and axis 1 running this skew-detection sequence. Axis 1 executes step 10 which sets the Extended Link Value to 100 and then waits until the difference between actual positions on axis 0 and 1 is greater to or equal the Extended Link Value. Should this limit be reached or exceed then step 11 is executed by the slave which issues a Kill Axis (K) command to stop both axes.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select System-wide (Basic, non-axis).
2. Under Link Type, select Skew Detection.
3. Under Link Condition, check the boxes next to each axis that you want to be included in the skew detection.
4. Under Link Condition, check the type of comparison to determine the skew.
5. Click OK.
6. Ensure that the Extended Link Value has been set up using the Set Extended Link Value command.

Using without the Link Type and Link Value Dialog Box

1. Enter '<' into the Link Type field for Skew on Actual Position, or '0x7f' for Skew on Following Error.
2. Enter the Link Value.
   You must understand binary to enter the Link Value manually. Each bit of the Link Value corresponds to an axis. Bit 0 corresponds to axis 0, bit 1 corresponds to axis 1, etc. Bits 8-15 are always unused. Each axis's bit that is on means that you wish to include that axis in the skew detection.
3. Ensure that the Extended Link Value has been set up using the Set Extended Link Value command.

E.4.12 Link Type - Timer

Link Type:	**TimerSt/Exp (T, hex 0x54, dec 84) - Start Timer/Timer Expired** **TimerNExp (t, hex 0x74, dec 116) - Timer Not Expired**
Link Value:	**0 to start timer, otherwise timer preset in milliseconds**
Range:	**0 to start timer, or 1 to 65,535 milliseconds**

Note: This feature is supported in RMC100 CPU firmware dated 20010420 or later.

These link types are used to start a timer and to wait for the timer to reach a preset. The DelayMS (D) link type offers similar functionality. However, while the DelayMS link type starts the timer when the event step is reached, the Timer (T and t) link types separate the starting and checking of the timer into two or more steps. This allows the following capabilities that are not available using the Delay (D) link type:

- The timer can be started at the beginning of a sequence of steps and checked after the sequence has completed. In effect, this timer can be used to check how much time has passed since the beginning of a section of the process rather than simply pausing for a fixed amount of time as the DelayMS (D) link type does. See Example 1 below.
- The Timer (T and t) link types can be used in a polling loop. See the Poll (?) command for details. One common use for polling link types is to keep track of a timeout. The DelayMS (D) link type does not work in these situations because it restarts its timer each time that step is reached. See

Examples 2 and 3 below.

- The Timer (T and t) link types allow multiple times to be checked since the beginning of a process. That is, the timer can be started at the beginning of a cycle. Then, later in the process, two or more Timer (T) link types can be used with different values to trigger events at different times since the beginning of the cycle. See Example 4 below.

  To support these various features, the Timer (T and t) link types come in three forms. Details for entering these forms into an Event Step table are given at the end of this topic.

- To start the timer, use a TimerSt/Exp (T) link type with a 0 link value. This will start the timer on the axis currently running this event sequence. This type should not be used with the Poll (?) command. See all four examples below.
- To follow the link if the timer has reached its preset, use a TimerSt/Exp (T) link type with the preset value in milliseconds in the link value. This type can be used either with or without the Poll (?) command.
  When used with the Poll (?) command, the link will jump to the Link Next step if the timer has reached its preset, and will otherwise jump to the next step in the sequence. See Example 2 below.
  When used without the Poll (?) command, the link will wait until the timer reaches the preset value and then jump to the Link Next step. See Examples 1 and 4 below.
- To follow the link if the timer has not reached its preset, use a TimerNExp (t) link type with the preset value in milliseconds in the link value. This type should only be used with the Poll (?) command, because using it without the Poll (?) command causes the link to wait until the timer is not expired, which is almost never the desired behavior. See Example 3 below.

  There is one timer per axis. The Timer (T and t) link types use the timer of the axis running the current event sequence. Therefore, it is possible to have as many timers running as axes are available, but no more. However, the timer used by this link type is separate from the timer used by the DelayMS (D) link type. Therefore, it is possible to have one or more DelayMS (D) link types in a process that is also being timed by a Timer (T or t) link type.

Note: This timer is only accurate to within one control loop (1 or 2 ms). For example, if a Timer (T) link type is used with a preset of one millisecond, then the link may be taken either on the first control loop or the second control loop of an RMC with a one-millisecond control loop. This will not affect most applications since much larger timeouts will be used with lower resolution requirements.

Example 1:

In this example, the user wants to make two moves, wait for each to complete, and then wait until ten seconds has passed since the beginning of the cycle before repeating.

	Step 10	**Step 11**	**Step 12**	**Step 13**
Mode	0x0000	0x0081	0x0081	0x0000
Accel	0	100	100	0
Decel	0	100	100	0
Speed	0	10000	10000	0
Command Value	0	4000	8000	0

Command		G	G	
Commanded Axes	Default	Default	Default	Default
Link Type	TimerSt/Exp	BitsON	BitsON	TimerSt/Exp
Link Value	0	0x0001	0x0001	10000
Link Next	11	12	13	10

Step 10 starts the timer and links immediately to step 11.

Step 11 starts a move to the first position and waits for the axis to get in position using the BitsON (B) link type, at which time control moves to step 12.

Step 12 starts a move to the second position and waits for the axis to get in position using the BitsON (B) link type, at which time control moves to step 13.

Step 13 waits for the timer to reach ten seconds, at which time control jumps back to step 10.

Example 2:

In this example, the user wants to move to a position, but if the axis does not get in position within five seconds, he/she wants to turn on discrete output 0. The following sequence accomplishes this:

	Step 10	**Step 11**	**Step 12**	**Step 13**	**Step 14**
Mode	0x0081	0x0081	0x0081	0x0081	0x0000
Accel	100	100	100	100	0
Decel	100	100	100	100	0
Speed	10000	10000	10000	10000	0
Command Value	4000	0	0	0	0x0001
Command	G	?	?		[
Commanded Axes	Default	Default	Default	Default	Default
Link Type	TimerSt/Exp	BitsON	TimerSt/Exp	DelayMS	DelayMS
Link Value	0	0x0001	5000	0	0
Link Next	11	15	14	11	15

Step 10 initiates the move to 4000, starts the timer, and links immediately to step 11.

Step 11 checks the In Position bit of the Status Word using the BitsON (b) link type. Therefore, if the axis is in position, then the link is taken to step 15. Otherwise, control passes to step 12.

Step 12 checks if 5000 ms (five seconds) have expired since the timer was started in step 10. If the timer is expired, then control passes to step 14. Otherwise, control passes to step 13.

Step 13 immediately links back to step 11 to restart the polling loop. Notice that this extra step does take one control loop to process and therefore increases the time taken to process the entire polling loop by one control loop (1 or 2 ms).

Step 14 handles the timeout condition by turning on discrete output 0.

Now, compare this method of implementing a timeout with the one used in the next example, which is a little more difficult to follow but removes the need for the extra do-nothing step in the polling loop (step 13 above).

Example 3:

This example offers an alternative approach to the same problem as Example 2: the user wants to make a move and wait for the move to complete or a timeout to occur. It is a little more difficult to follow, but removes one step from the polling loop, therefore reducing the amount of time taken to catch any condition. Keep in mind that for many applications, this slight gain in performance will not justify the reduction in clarity.

	Step 10	**Step 11**	**Step 12**	**Step 13**	**&**
Mode	0x0081	0x0081	0x0081	0x0000	
Accel	100	100	100	0	
Decel	100	100	100	0	
Speed	10000	10000	10000	0	
Command Value	4000	0	0	0x0001	
Command	G	?	?	[	
Commanded Axes	Default	Default	Default	Default	
Link Type	TimerSt/Exp	BitsON	TimerNExp	DelayMS	
Link Value	0	0x0001	5000	0	
Link Next	11	14	11	14	

Step 10 initiates the move to 4000, starts the timer, and links immediately to step 11.

Step 11 checks the In Position bit of the Status Word using the BitsON (b) link type. Therefore, if the axis is in position, then the link is taken to step 14. Otherwise, control passes to step 12.

Step 12 checks if 5000 ms (five seconds) have expired since the timer was started in step 10. Notice that, this time, the Timer Not Expired (t) link type is used. Therefore, if the timeout has not occurred, then control jumps back to step 11. If the timeout has occurred, then control passes to step 13.

Step 13 handles the timeout condition by turning on discrete output 0.

Here is a summary of the advantages and disadvantages of doing a polled loop using this method versus the method shown in Example 2:

- The advantage of this method is that the timeout and the In Position bit will be checked every two control loops since there are only two steps in the polling loop (steps 11 and 12).

- The disadvantage of this method is that many users will find it confusing to have to reverse the sense of the last condition in the polling loop. That is, the user has to use the Timer Not Expired (t) link type instead of the more intuitive Timer Expired (T) link type. This situation is not unique to using the Timer link types, but will occur with any series of polling link types for the last condition in the polling cycle.

Example 4:

In this example, the user wants to complete a move, and then turn on and off discrete output 0 at five and ten seconds respectively since the start of the move.

	Step 10	**Step 11**	**Step 12**	**Step 13**	**Step 14**
Mode	0x0000	0x0081	0x0000	0x0000	0x0000
Accel	0	100	0	0	0
Decel	0	100	0	0	0
Speed	0	10000	0	0	0
Command Value	0	4000	0	0x0001	0x0001
Command		G		[	]
Commanded Axes	Default	Default	Default	Default	Default
Link Type	TimerSt/Exp	BitsON	TimerSt/Exp	TimerSt/Exp	DelayMS
Link Value	0	0x0001	5000	10000	0
Link Next	11	12	13	14	0

Step 10 starts the timer and links immediately to step 11.

Step 11 starts the move and waits for the move to complete using the BitsON (B) link type before linking to step 12.

Step 12 waits until five seconds have expired since the timer was started in step 10 before linking to step 13.

Step 13 turns on discrete output 0 and waits until ten seconds have expired since the timer was started in step 10 before linking to step 14.

Step 14 turns discrete output 0 back off before linking back to step 0.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select System-wide (Basic, non-axis).
2. Under Link Type, select Timer.
3. Under Link Condition, select one of the following:
   - If you want to start the timer, select Start the Timer and Link to the Link Next.
   - If you want to take the link when the timer reaches or passes its preset, select Link if the Timer is Expired.

- If you want to take the link when the timer has not yet reached its preset, select Link if the Timer is Not Expired. This is only useful when used with the Poll (?) command.

4. If you selected either the second or third option above, then type the preset (timeout) value in the Timer Value text box.
5. Click OK.

Using without the Link Type and Link Value Dialog Box

1. Enter a Link Type:
    - If you want to start the timer, enter 'T'.
    - If you want to take the link when the timer is expired, enter 'T'.
    - If you want to take the link if the timer is not expired, enter 't'. This is only useful when used with the Poll (?) command.
2. Enter a Link Value
    - If you want to start the timer, enter 0.
    - Otherwise, enter the preset value in milliseconds.

See also:

DelayMS (D) link type

Poll (?) command

E.4.13 Link Type - Check Wait Bits

Link Type:	**hex 0x57, dec 87**
Link Value:	**Bits that must be cleared : bits that must be set**

The Check Wait Bits link type can be used to check the state of the internal wait bits that are set or reset by either the Set and Reset Wait Bits Command or the Profibus DP spline processor. The link value field is split into two 8 bit fields. The high 8 bits specify the wait bits that must be cleared (0) and the lower 8 bits specify the wait bits that must be set (1). It is possible to check for some bits to be set and others cleared while still ignoring other bits all in the same link. When the wait bits match the conditions specified by the link value the step table will go to the next step specified by the Link Next field.

See the Set and Reset Wait Bits Command topic for more details.

E.5 Current Axis Link Types

E.5.1 Link Type - Current Axis Absolute Limit Switch

Link Type:	**TarPos (L, hex 0x4C, dec 76) - Target Position** **ActPos (l, hex 0x6C, dec 108) - Actual Position**
Link Value:	**Limit Position**
Range:	**Any valid position in position units**

These link types are used to detect when the current axis crosses a position. This link type can be used to change speeds on-the-fly or trigger events on other axes. The Link Value holds the Limit position in position units. Either the Target Position or Actual Position can be used to compare with the Limit position.

Note: The link is made once the position crosses the limit, rather than being made whenever the position is on one side or the other of the limit. If you wish to wait for the axis to be on one side or the other, use the Any Axis Position link type.

Example:

Suppose a move is being made from 4000 to 12000 position units. If the link type is TarPos and the Link Value is 8000, the next step would be executed when the Target Position reaches 8000.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select Current Axis (Basic).
2. Under Link Type, select Absolute Limit Switch.
3. Under Link Condition, select whether you wish to use Target or Actual Position for the comparison.
4. Under Link Condition, enter the Limit position in the Threshold box.

Using without the Link Type and Link Value Dialog Box

1. Select the Link Type:
   Use 'L' to use the Target Position in the comparison, 'l' to use the Actual Position.
2. Enter the Limit position in the Link Value.

E.5.2 Link Type - Current Axis Relative Limit Switch

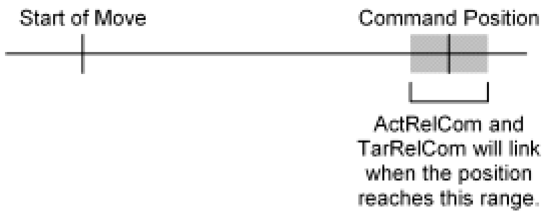
Link Type:	**TarRelStart (R, hex 0x52, dec 82)** **ActRelStart (r, hex 0x72, dec 114)** **TarRelCom (N, hex 0x4E, dec 78)** **ActRelCom (n, hex 0x6E, dec 110)**
Link	**Limit Position Window from Start or End of Move**

Value:

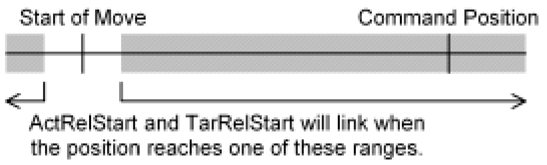
Range: **0 to 65,535 position units**

This family of link types waits for the position of the current axis to reach a distance from the start or end of a move. There are four separate link types in this family to cover comparing against the start or end of the move using either the Target or Actual Position.

The TarRelCom and ActRelCom link types will link to the next step when the Target or Actual Position (respectively) is within the Limit Position Window specified by the Link Value from the Command Position status field, as shown below:


Start of Move
Command Position
ActRelCom and TarRelCom will link when the position reaches this range.

The TarRelStart and ActRelStart link types will link to the next step when the Target or Actual Position is outside the Limit Position Window specified by the Link Value from the start of the move, as shown below:


Start of Move
Command Position
ActRelStart and TarRelStart will link when the position reaches one of these ranges.

The start of the move is defined by the position when a Go (G) command is issued when the axis is stopped, when a Sine Move (~) command is issued, or when an Open Loop (O) command is issued.

Example:

To move to the next event step when the Target Position has moved 1000 position units from the start of the move, the TarRelStart link type with a Link Value (Link Position Window) of 1000 would be used. Therefore, if the move is from 4000 to 12000, the link will trigger when the Target Position reaches 5000. If the move is from 4000 to 1000, the link will trigger at 3000.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select Current Axis (Basic).
2. Under Link Type, select Relative Limit Switch.
3. Under Link Condition, select whether you wish to use Target or Actual Position for the comparison.
4. Under Link Condition, enter the Limit Position Window in the text box.

Using without the Link Type and Link Value Dialog Box

1. Select the Link Type.

Use the following table to choose a Link Type:

Link Type	Position Used	Relative To
R	Target Position	Start of Move
r	Actual Position	Start of Move
N	Target Position	End of Move
n	Actual Position	End of Move

2. Enter the Limit Position Window in the Link Value.

E.5.3 Link Type - Current Axis Pressure

Link Type:	**ActPrs (P, hex 0x50, dec 80) - Actual Pressure** **TarPrs (p, hex 0x70, dec 112) - Target Pressure**
Link Value:	**Limit Pressure**
Range:	**-32,768 to 32,767 pressure units**

These link types are used to detect when the pressure assigned to an axis crosses a user-specified value. Notice that if this link type is used on an auxiliary pressure or force channel, its own pressure is used in the comparison, but if it is used on any other type of axis, the pressure of the channel assigned to the current axis is used in the comparison. Either the Target Pressure or Actual Pressure can be used to compare with the Limit Pressure.

Note: The link is made once the pressure crosses the limit, rather than being made whenever the pressure is on one side or the other of the limit. If you wish to wait for the axis to be on one side or the other, use the Any Axis Position/Pressure link type.

Example:
Suppose the first axis (Axis0) has a ActPrs link type with a link value of 2500, and this axis has the first auxiliary axis assigned to it (Aux0). This means that when the Actual Pressure on Aux0 crosses 2500, the next step in the event sequence will be taken. Notice, that the exact same behavior would occur if the link type was executed on Aux0 instead of Axis0.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select Current Axis (Basic).
2. Under Link Type, select Pressure.
3. Under Link Condition, select whether you wish to use Target or Actual Pressure for the comparison.
4. Under Link Condition, enter the Limit Pressure in the Threshold box.

Using without the Link Type and Link Value Dialog Box

1. Select the Link Type:
   Use 'P' to compare with the Actual Pressure, 'p' to compare with the Target Pressure.

2. Enter the Limit Pressure in the Link Value.

E.5.4 Link Type - Current Axis Speed

Link Type:	**TarSpd (S, hex 0x53, dec 83) - Target Speed** **ActSpd (s, hex 0x73, dec 115) - Actual Speed**
Link Value:	**Limit Speed**
Range:	**0 to 65,535 position units per second**

These link types are used to detect when the speed of the current axis has reached a user-specified value. This link type can be used to change speeds on-the-fly or trigger events on another axis. The Link Value holds the Limit speed in position units per second. Either the Target Speed or Actual Speed may be used to compare with the Limit speed.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select Current Axis (Basic).

2. Under Link Type, select Speed.

3. Under Link Condition, select whether you wish to use Target or Actual Speed for the comparison.

4. Under Link Condition, enter the Limit speed in the Threshold box.

Using without the Link Type and Link Value Dialog Box

1. Select the Link Type:
   Use 'S' to compare with the Target Speed, 's' to compare with the Actual Speed.

2. Enter the Limit speed in the Link Value.

E.5.5 Link Type - Current Axis Status Bits

Link Type:	**BitsON (B, hex 0x42, dec 66)** **BitsOFF (b, hex 0x62, dec 98)**
Link Value:	**Bit pattern that must be set or cleared**
Range:	**Any bits**

These link types wait for at least one of a user-selectable group of Status Bits to be set or cleared on the current axis.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select Current Axis (Basic).
2. Under Link Type, select Status Bits.
3. Under Link Condition, click the appropriate option for whether you wish to wait for one or more bits to be ON or OFF.
4. Under Link Condition, check the boxes next to the Status bits for which you wish to wait.

Using without the Link Type and Link Value Dialog Box

1. Select the Link Type:
   Use 'B' if you wish to wait for Status Bit(s) to be on, 'b' if you wish to wait for Status Bit(s) to be off.
2. Enter the Link Value. Use the STATUS Word Bit Map to calculate a 16-bit hexadecimal word that matches the bits you wish to wait for.

E.6 Selected Axis Link Types

E.6.1 Link Type - Any Axis Position/Pressure

Link Type:	**TarPos# < (see below for values)** **TarPos# > (see below for values)** **ActPos# < (see below for values)** **ActPos# > (see below for values)**
Link Value:	**Limit Position**
Range:	**Any valid position in Position Units**

Note: In the following discussion position means pressure or force for auxiliary analog inputs.

These link types are used to detect when the position of the axis selected by the link type is above or below a limit. This link type can be used to change speeds on-the-fly or trigger events on other axes. The Link Value holds the Limit position in position units. Either the Target Position or Actual Position may be used to compare with the Limit position.

Using with the Link Type and Link Value Dialog Box

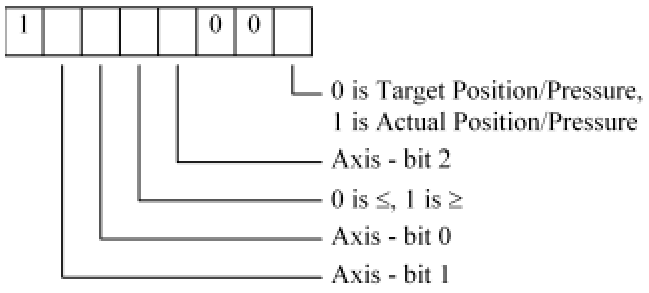
1. Under Link Type Category, select Selected Axis (Enhanced).
2. Under Axis, select the axis you desire to monitor.
3. Under Link Type, select Position/Pressure.
4. Under Link Condition, select whether you wish to use Target or Actual Position for the comparison.
5. Under Link Condition, select whether you wish to wait until the position is above or equal, or below or equal the Limit position.

6. Under Link Condition, enter the Limit position in the Threshold box.

Using without the Link Type and Link Value Dialog Box

1. Calculate the Link Type:

This involves converting the following bit fields into a hexadecimal byte. It is highly recommended that the Link Type and Link Value dialog box be used instead of doing this manually. Use the following diagram to calculate the byte in binary and then convert to hexadecimal or decimal and enter in the Link Type field.



Axis - bit 2, **Axis - bit 1** and **Axis - bit 0** specify the number (in binary) of the axis to monitor (Axis 0 to 7).

2. Enter the Limit (threshold) position in the Link Value field.

E.6.2 Link Type - Any Axis Speed

Link Type:	**TarSpd# < (see below for values)** **TarSpd# > (see below for values)** **ActSpd# < (see below for values)** **ActSpd# > (see below for values)**
Link Value:	**Limit Speed**
Range:	**0 to 65,535 position units per second**

Note: In the following discussion speed means change in pressure or force for auxiliary analog inputs.

These link types are used to detect when the speed of the axis selected by the link type is above or below a user-specified value. This link type can be used to change speeds on-the-fly or trigger events on another axis. The Link Value holds the Limit speed in position units per second. Either the Target Speed or Actual Speed may be used to compare with the Limit speed.

Using with the Link Type and Link Value Dialog Box

1. Under Link Type Category, select Selected Axis (Enhanced).

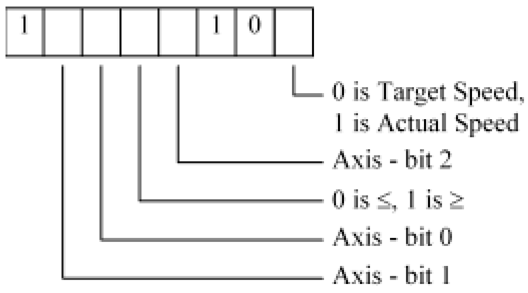
2. Under Axis, select the axis you desire to monitor.

3. Under Link Type, select Speed.
4. Under Link Condition, select whether you wish to use Target or Actual Speed for the comparison.
5. Under Link Condition, select whether you wish to wait until the speed is above or equal, or below or equal the Limit speed.
6. Under Link Condition, enter the Limit speed in the Threshold box.

Using without the Link Type and Link Value Dialog Box

1. Calculate the Link Type:

   This involves converting the following bit fields into a hexadecimal byte. It is highly recommended that the Link Type and Link Value dialog box be used instead of doing this manually. Use the following diagram to calculate the byte in binary and then convert to hexadecimal or decimal and enter in the Link Type field.



   Axis - bit 2, **Axis - bit 1** and **Axis - bit 0** specify the number (in binary) of the axis to monitor (Axis 0 to 7).

2. Enter the Limit (threshold) speed in the Link Value field.

E.6.3 Link Type - Any Axis Status Bits

Link Type:	**Bits# ON (see below for values)** **Bits# OFF (see below for values)**
Link Value:	**Bit pattern that must be set or cleared**
Range:	**Any bits**

These link types wait for at least one of a user-selectable group of Status Bits to be set or cleared on the axis selected by the link type.

Using with the Link Type and Link Value Dialog Box

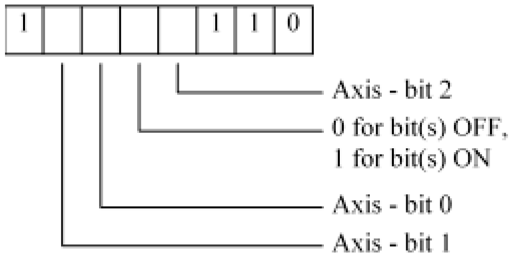
1. Under Link Type Category, select Selected Axis (Enhanced).
2. Under Axis, select the axis you desire to monitor.
3. Under Link Type, select Status Bits.

4. Under Link Condition, click the appropriate option for whether you wish to wait for one or more bits to be ON or OFF.
5. Under Link Condition, check the boxes next to the Status bits that you wish to monitor with this link type.

Using without the Link Type and Link Value Dialog Box

1. Calculate the Link Type:

This involves converting the following bit fields into a hexadecimal byte. It is highly recommended that the Link Type and Link Value dialog box be used instead of doing this manually. Use the following diagram to calculate the byte in binary and then convert to hexadecimal or decimal and enter in the Link Type field.



Axis - bit 2, **Axis - bit 1** and **Axis - bit 0** specify the number (in binary) of the axis to monitor (Axis 0 to 7).

2. Enter the Link Value. Use the STATUS Word Bit Map to calculate a 16-bit hexadecimal word that matches the bits you wish to wait for.

Appendix F: RMC100 Specifications

F.1 RMC100 Specifications

Motion Control

Control loop time	1 or 2 ms depending on module configuration
Maximum speed	65,535 user-defined position units per second

RS232 Port

Interface with Delta's RMCWin software and the RMCLink ActiveX Control and .NET Assembly Component	Requires a PC running Windows 98/NT/2000/Me/XP/Vista/7.
Connector	DB9 Male DTE
Cable	Null modem to a PC

RJ-11 LCD Terminal Jack

Interface with Delta's optional four-line, 20-character LCD display with keypad	Allows viewing status information, changing parameters, and issuing commands.

Power

Voltage	+24 VDC ±20% For UL compliance, a class 2 power supply must be used.
Current - 2 axes (3 slots)	Typical 290 mA @ 24 VDC, max 375 mA
4 axes (4 slots)	Typical 385 mA @ 24 VDC, max 500 mA
6 axes (5 slots)	Typical 485 mA @ 24 VDC, max 625

	mA
8 axes (6 slots)	Typical 585 mA @ 24 VDC, max 750 mA
DC-DC converter isolation	500 VAC, 700 VDC, input to controller

Mechanical

Dimensions - 2 axes (3 slots)	4.12 x 5.95 x 4.75 in (10.5 x 15.0 x 12.1 cm) (WxHxD)
8 axes (6 slots)	7.12 x 5.95 x 4.75 in (18.1 x 15.0 x 12.1 cm) (WxHxD)
Weight - 2 axes (3 slots)	2.0 lb (0.9 kg) max
8 axes (6 slots)	3.0 lb (1.4 kg) max

Mounting

Options	Symmetrical DIN 3 or panel-mount
Orientation	The RMC should be mounted upright on a vertical surface, such that the air holes are on top and bottom.
Clearance, above and below	The amount of clearance required depends on the maximum ambient temperature: Above 122°F (50°C) 3 in (7.6 cm) 86 to 122°F (30 to 50°C) 2 in (5.1 cm) Less than 86°F (30°C) 1 in (2.5 cm)

Environment

Operating temperature	+32 to +140°F (0 to +60°C)
Storage temperature	-40 to +185°F (-40 to +85°C)
Humidity	93%, non-condensing
Compliance	CE, UL, Canadian UL (see details below)

Agency Compliance

CE Tests Performed	See also General Wiring Information.
Radiated Emissions	EN55022 Class A
Conducted Emissions	EN55022 Class A
Electrostatic Discharge	EN61000-4-2 ±4 kV contact EN50082-1 ±8 kV air
Radiated Immunity	EN61000-4-3 3 V/m EN50082-1 ENV50204
Electrical Fast Transient Burst	EN61000-4-4 1 kV (A/C) EN50082-1 0.5 kV (I/O)
Conducted Immunity	EN61000-4-6 3 V rms EN50082-1
UL, C-UL Specifics	See also General Wiring Information.
File Number	E141684 Input power 24VDC, 750 mA max. All inputs and outputs must be connected to Class 2 circuits only.

Failsafe Timers

No internal bus activity	After 15µs, the drive outputs are disabled.
Software watchdog	After 26ms, the module resets and drive outputs are disabled if the microprocessor does not retrigger the onboard watchdog timer.

Flash Memory Storage

Retention Time	10 years at 150°C, 20 years at 125°C
Write/erase Cycles	100,000 minimum

Transducer Inputs and Drive Outputs

See the appropriate section(s):

- MDT Specifications
- Analog Transducer Specifications
- Quadrature Specifications

- Stepper Specifications
- SSI Specifications

Digital I/O

See Digital I/O Specifications

Communication Modules

See the appropriate section(s):

- PROFIBUS-DP Module
- SERIAL Module
- Ethernet Module
- Modbus Plus Module

F.2 General Wiring Information

For CE compliance and to minimize electrical interference:

- Use twisted pairs for all wiring where possible.
- Use shielded cables for all wiring.
- Keep RMC wiring separate from AC mains or conductors carrying high currents, especially high frequency switching power such as conductors between servo drives and motors or amplifiers and proportional valves.

For UL and C-UL compliance:

- Power supply must be Class 2.
- All RMC inputs and outputs must be connected to Class 2 circuits only.

Appendix G: Glossary

G.1 Glossary

Clockwise

Rotating in the direction of increasing encoder or transducer counts.

Closed Loop Mode

Sometimes called Servo mode. In this mode the difference between the Target and Actual position/pressure is the error that the PID routine uses to compute a corrective drive that minimizes the error.

Counter-Clockwise

Rotating in the direction of decreasing encoder or transducer counts.

Extending

Going in the direction of increasing encoder or transducer counts.

GSD file

This is a device description file required by every PROFIBUS device. The RMC's GSD file is installed with RMCWin's program files. Refer to PROFIBUS Configuration for details on using this file.

Hard Stop

An emergency stop condition where the drive is immediately set to the null drive value.

LDT

Linear Displacement Transducer. Technically, this is any sensor that can sense a linear position. Many use this term interchangeably with MDT.

MDT

Magnetostrictive Displacement Transducer. A device that senses position by sending an electron pulse down a wave guide. A twist is imparted on the wave guide as the pulse reaches the magnetic field of a magnet. The twist takes time to be sensed at the transducer head. It is this time is proportional to the distance between the transducer head and the magnet. The RMC measures this time to determine the distance.

Note: Balluff and Temposonics rods are both MDTs whereas any linear measurement device can be a LDT.

Open Loop Mode

The drive output is set to a value without any corrective action taken by the PID routine.

PID

Proportional Integral Derivative. A simple algorithm used by the motion controller to reduce the error between the target and actual position, pressure, etc. The three words of this acronym represent the three types of gains used for controlling a system. See PID Loop below.

PID Loop

The process of repeatedly executing the PID routine

Error: = Target - Actual

E0: Current Error, Error for current Target and Actual

E1: Previous Error, Error for previous Target and Actual

U0: Current output

U1: Previous output

Kp: Proportional Gain

Ki: Integral Gain

Kd: Differential Gain

Proportional Term: U0p = Kp * E0

Integral Term: U0i = U1i + Ki * E0

Differential Term: U0d = (E0 - E1) * Kd

The Sum is: U0 = U0p + U0i + U0d

The PID LOOP is:

```
DO FOREVER
    WAIT FOR NEXT TIME PERIOD
    READ ACTUAL FROM POSITION OR PRESSURE SENSOR
    E0 = TARGET - ACTUAL

    U0p = Kp * E0
    U0i = U1i + Ki * E0
    U0d = ( E0 - E1 ) * Kd
    U0 = U0p + U0i + U0d
    E1 = E0
    CALCULATE NEXT TARGET POSITION
END
```

Retracting

Going in the direction of decreasing encoder or transducer counts.

Soft Stop

An emergency stop condition where the drive ramps down to the null drive value.

SSI

Synchronous Serial Interface. A widely accepted controller interface to sensors and absolute encoders. Position data is encoded in a 13- to 25-bit binary or Gray Code format and transmitted over a high-speed synchronous interface.

Target Generator

The part of the motion control software that determines where the axis should be at any moment using the requested position, speed, acceleration and deceleration provided by the user. A different target generator is used for position control, speed control, gearing, synchronized moves and pressure control.

Appendix H: ASCII Table

H.1 ASCII Table

Dec	Hex	ASCII	Dec	Hex	ASCII	Dec	He
32	20	Space	64	40	@	96	60
33	21	!	65	41	A	97	61
34	22	"	66	42	B	98	62
35	23	#	67	43	C	99	63
36	24	$	68	44	D	100	64
37	25	%	69	45	E	101	65
38	26	&	70	46	F	102	66
39	27	'	71	47	G	103	67
40	28	(	72	48	H	104	68
41	29	)	73	49	I	105	69
42	2A	*	74	4A	J	106	6A
43	2B	+	75	4B	K	107	6B
44	2C	,	76	4C	L	108	6C
45	2D	-	77	4D	M	109	6D
46	2E	.	78	4E	N	110	6E
47	2F	/	79	4F	O	111	6F
48	30	0	80	50	P	112	70
49	31	1	81	51	Q	113	71
50	32	2	82	52	R	114	72
51	33	3	83	53	S	115	73
52	34	4	84	54	T	116	74
53	35	5	85	55	U	117	75
54	36	6	86	56	V	118	76
55	37	7	87	57	W	119	77
56	38	8	88	58	X	120	78
57	39	9	89	59	Y	121	79
58	3A	:	90	5A	Z	122	7A
59	3B	;	91	5B	[	123	7B
60	3C	<	92	5C	\	124	7C
61	3D	=	93	5D	]	125	7D
62	3E	>	94	5E	^	126	7E
63	3F	?	95	5F	_		

Index