RMCTools Troubleshooting Tips

Plots and the Event Log are your most important troubleshooting tools. Here are tips and advanced techniques.

Saving the Event Log

You can save the Event Log. This may be requested by Delta Technical Support.

To save the Event Log, in the Event Log Monitor, click the **Save Event Log** button. It will be saved as an individual file. It will not be saved in the RMCTools project.

Correlate Plots and the Event Log

It is often useful to be able to correlate the Event Log entries with the plot.

1. Run a user program, upload the plot, and open the Event Log. If new entries are appearing, you may need to click the **Pause** button.

2. On the **Plots** menu (at the top of RMCTools), choose **View** and make sure **Show Absolute Time** is checked.

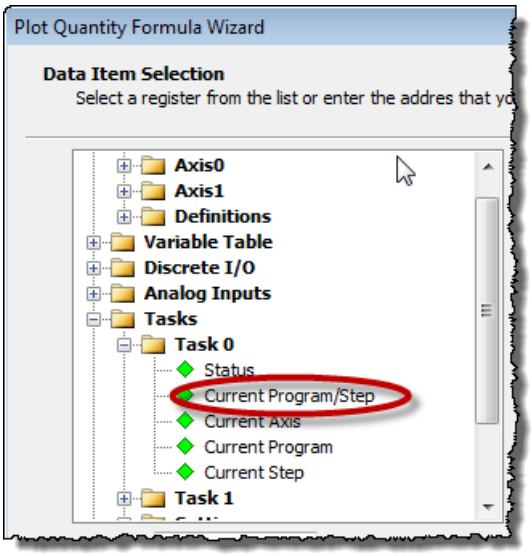
3. The **Time (Abs)** shown in the **Plot Details** is the same as that listed in the **Time** column of the Event Log. Find an event in the Event Log, and the corresponding time in the plot.

4. If you paused the Event Log, makes sure to click the **Resume** button.

Include User Programs Steps in Plot

It is often important to be able to track which user program steps are running at various points in a plot.

1. Add the **Task 0 Current Program/Step** register to a plot template. You can browse to it as shown here:



2. Run **MyProgram** and upload the plot.

3. In the plot detail window, see how the task and step are displayed as you move the cursor in the plot.

Event Log Filtering

You can choose to filter out certain events in the Event Log. This is useful in large RMCTools projects where the Event Log may fill up with events that you are not interested in, making it difficult to find the ones you *are* interested in.

In addition, you can include certain items that are not normally included in the Event Log, such as communication transactions.

To modify the Event Log filter:

1. In the Project tree, right-click **Event Log** and choose **Properties**.

2. Browse through the folder tree to see the items that can filtered out.

3. When you have finished your selections, click **OK**.

Trigger Plot and Stop Event Log to Capture Error

Intermittent errors are often difficult to troubleshoot. Here is a method to capture a plot and stop the Event Log when an error occurs. This will allow you to let the system run for some time, and come back later after the error occurred. A plot will have been captured that includes data from before and after the error occurred, and the Event Log will have stopped, so you will have plenty of information to help determine the cause of the problem.

This method involves the following:

- **Set up a plot for manual trigger**
  This will allow the plot to capture data from before and after the moment at which the plot is triggered, providing important data about the motion leading up the error.
- **Create a user program to wait for the error**
  Make a user program that waits for the error to occur, then triggers the plot and stops the Event Log. As long as the user program is running, you know that the error has not yet occurred.

Here are step-by-step instructions:

1. Edit a Plot Template to contain the data items you need to capture, such as Actual Positions, Status and Error bits, Control Output, Task Status, etc.

2. Set the **Plot Duration** to a suitably long time.

3. Set the Plot Template's **Trigger Settings** (at the bottom of the Plot Template Editor) as follows:
    a. Uncheck **Enable Automatic Trigger**.
    b. Choose **Manual Rearm**.
    c. Set the **Trigger Location** to 50%.

4. Make a user program that does the following:
    a. **Step 0 Command:** Send the **Rearm Plot (103)** command for the Plot Template you just edited, and–if you are monitoring for error bits–send the **Clear Faults (4)** command to clear out any error bits on the pertinent axes.
    b. **Step 0 Link Type:** Use the **Wait For** link type and enter the condition you need to monitor for.

    > **Tip:** If you are doing this as an exercise, check for the **Output Saturated** error bit on an axis. You can easily cause this bit to turn on by commanding the axis to move faster than it is able to.

    c. **Step 1 Command:** Send the **Trigger Plot (102)** command, and the **Pause/Resume Log (95)** command.

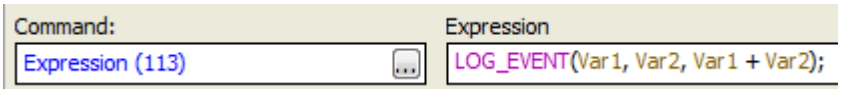
5. Start the user program on a task that will not have anything else running on it. The user program will re-arm the plot, then wait at step 0 for the error to occur. Once the error occurs, the user program will trigger the plot, stop the Event Log, and then end. You can came back to the system some time later, view and save the Event Log and plot. Remember to restart the Event Log after saving it.

> **Tip:** If you followed the previous tip, send a command to the axis to move it faster than it is able to. The **Output Saturated** error bit will turn on, and you can look at the captured plot and Event Log.

Log Event() Function

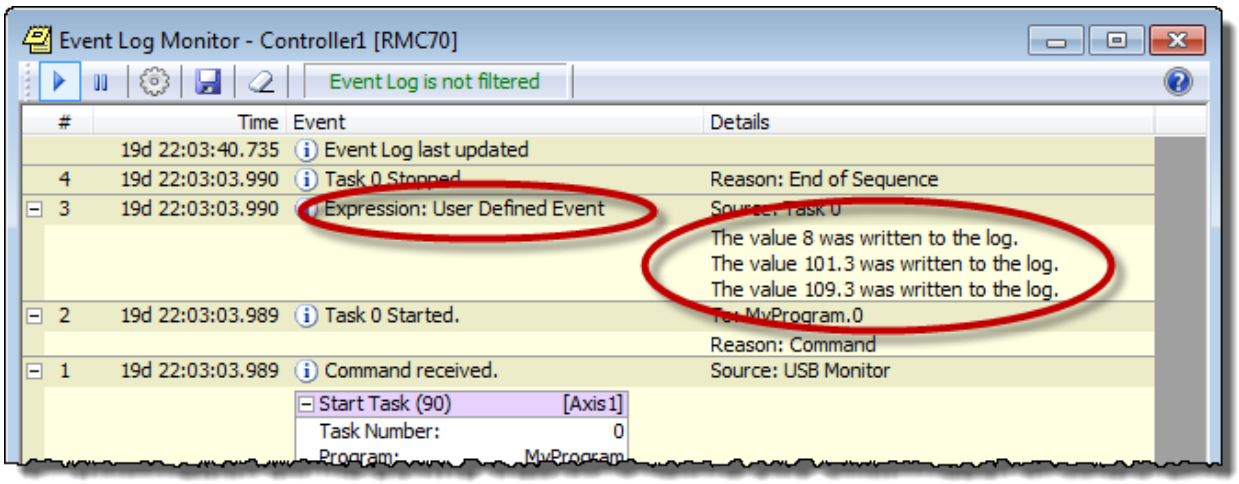
In the user program expressions, you can use the **Log_Event** function to log values (such as variable values) in the Event Log. This is helpful when debugging complex calculations.

1. Create two variables called **Var1** and **Var2**.
2. In a user program, add an Expression (113) command.
3. In the **Expression** box, type
   LOG_EVENT(Var1, Var2, Var1 + Var2);



4. Set the current values of the variables to some values.

5. Run the user program and look in the Event Log. You will see some **User Defined Events**, which are the values given by the Log_Event function, and will see that the variable values and their sum were reported.



Want more?

This is a taste of the concepts offered in RMCTools trainings. Sign up for a live-online or classroom training class to learn more about using the RMC75E and RMC150E motion controllers. For details, go to www.deltamotion.com.