LabVIEW™ Execution Trace Toolkit User Guide

This document provides an overview of the LabVIEW Execution Trace Toolkit. The LabVIEW Execution Trace Toolkit is a real-time event and execution tracing tool that allows you to capture and display the timing and event data of applications you build with the LabVIEW Real-Time Module. Refer to the *LabVIEW Real-Time Module User Manual* for information about developing deterministic applications using LabVIEW.

Contents

April 2004
323738A-01

ni.com

NATIONAL INSTRUMENTS™

Installation Instructions

Complete the following steps to install the LabVIEW Execution Trace Toolkit.

1. **(Windows 2000/NT/XP)** Log in as an administrator or as a user with administrator privileges before you install the LabVIEW Execution Trace Toolkit.
2. Insert the LabVIEW Execution Trace Toolkit CD into the CD-ROM drive. The LabVIEW Execution Trace Toolkit installation program runs automatically.
3. Follow the instructions that appear on the screen.

Introduction to the LabVIEW Execution Trace Toolkit

The LabVIEW Execution Trace Toolkit includes the LabVIEW Execution Trace Tool and the Execution Trace Tool VIs. The Execution Trace Tool VIs allow you to capture the timing and execution data of VI and thread events for applications running on an RT target. The LabVIEW Execution Trace Tool displays the timing and event data, or trace session, on the host computer. Launch the LabVIEW Execution Trace Tool by selecting **Tools»LabVIEW Execution Trace Tool** in LabVIEW.

Capturing a Trace Session

The LabVIEW Execution Trace Tool displays trace sessions you capture using the Execution Trace Tool VIs. You must add Execution Trace Tool VIs to the block diagram of an application running on the RT target to start and stop the logging of timing and event data from VIs and operating system threads.

Starting a Trace Session

The TraceTool Start Trace VI starts logging event data on the RT target. Figure 1 shows the block diagram of a VI with the TraceTool Start Trace VI added.

The **Buffer Size** input of the TraceTool Start Trace VI sets the size of the memory buffer that collects all event data for the application on the RT target. Set the size of the memory buffer large enough to contain all event data. If you reach the memory buffer limit when logging event data, the TraceTool Start Trace VI overwrites the oldest data in the buffer.

Note You cannot change the size of the memory buffer on the RT target after you run the TraceTool Start Trace VI. You must reboot the RT target to resize the memory buffer.

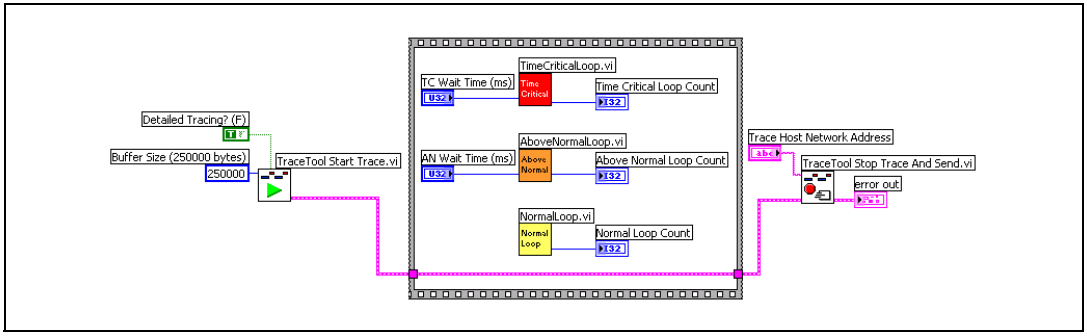

Figure 1. Starting and Stopping a Trace Session

The amount of time you can capture in a trace session depends on the size of the memory buffer and the type of events that you choose to capture. You can disable the logging of VI, thread, and detailed events. The **Thread Tracing?** and **VI Tracing?** inputs of the TraceTool Start Trace VI specify whether to log thread and VI events. The **Detailed Tracing?** input of the TraceTool Start Trace VI specifies whether to log detailed events, which the LabVIEW Execution Trace Tool uses to display analysis and custom events. Refer to the *Logging Analysis and Custom Events* section for information about analysis and custom events.

Stopping a Trace Session

The TraceTool Stop Trace and Send VI stops logging event data on the RT target and then sends the trace session to the LabVIEW Execution Trace Tool running on the host computer. Specify the IP address of the host computer running the LabVIEW Execution Trace Tool using the **Trace Host Network Address** input of the TraceTool Stop Trace and Send VI. Figure 1 shows the block diagram of a VI with the TraceTool Stop Trace and Send VI added.

Note The LabVIEW Execution Trace Tool must be running on the host computer to receive the trace session.

The VI in Figure 1 uses a Flat Sequence structure to define the dataflow of the block diagram and force the TraceTool Start Trace VI to execute before VIs in the application. The TraceTool Stop Trace and Send VI executes after the application completes. You can start and stop the trace session in a subVI of an application to target a specific section of code and to conserve space in the memory buffer.

You also can use the TraceTool Stop Trace and Save VI to stop logging event data and save the trace session to a file on the RT target. You can transfer the trace session file to the host computer using the TraceTool Load Trace and Send VI. Refer to the *Working with Trace Sessions* section for information about working with trace sessions on the host computer. Refer to the *LabVIEW Help* by selecting **Help»VI, Function, & How-To Help** in LabVIEW for VI reference information about the Execution Trace Tool VIs.

Viewing Trace Sessions

The LabVIEW Execution Trace Tool displays a trace session graphically using the VI Events and Thread Events views, as shown in Figure 2.

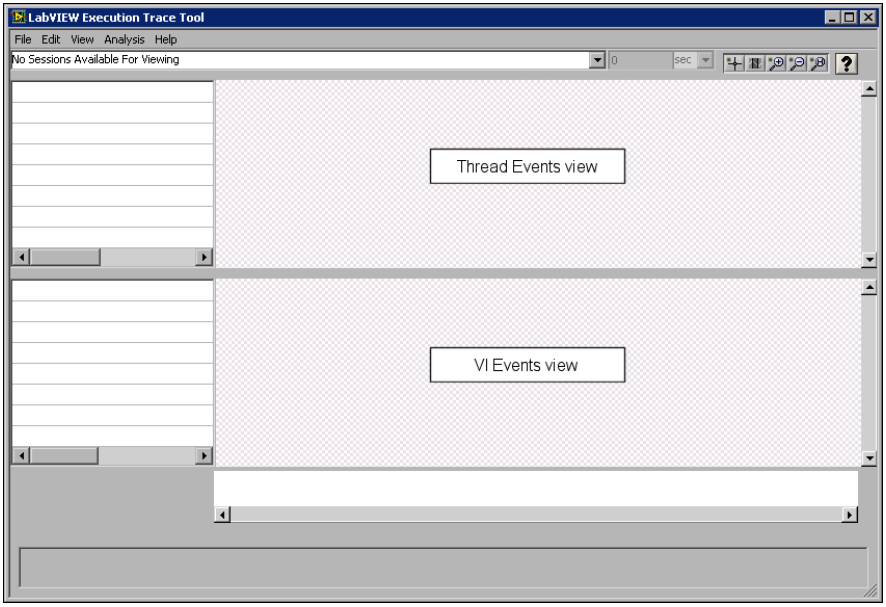

Figure 2. LabVIEW Execution Trace Tool Views

When the LabVIEW Execution Trace Tool loads a trace session, it displays the entire trace session in the VI Events and Thread Events views. To display only the Thread Events view or only the VI Events view in the LabVIEW Execution Trace Tool, deselect from the **View** menu the view that you want to hide.

You can zoom in and out of a region of the trace session or measure the timing of a specific range in the trace session using the following tools available in the LabVIEW Execution Trace Tool palette:

Use the Cursors tool to display two cursors in the VI Events or Thread Events view to measure the timing of the range contained between the two cursors. Click and drag each cursor on the VI Events or Thread Events view to set the range you want to measure. Use the pull-down menu to the left of the Cursors tool to select the units for the timing measurement. The text box to the left of the Cursors tool displays the measurement results.

Use the Zoom Region tool to zoom into a region you select. Click and drag in the VI Events or Thread Events view to select a zoom range.

Use the Zoom In tool to zoom into the VI Events and Thread Events view. Click on the VI or Thread Events view with the Zoom In tool to zoom into the currently displayed range.

Use the Zoom Out tool to zoom out of the VI Events and Thread Events views. Click on the VI Events or Thread Events view with the Zoom Out tool to zoom out of the currently displayed range.

Use the View Entire Session tool to display the entire trace session in the VI Events and Thread Events views.

Viewing and Analyzing VI Events

The LabVIEW Execution Trace Tool displays VI event data in the VI Events view. The VI Events view shows all VIs in memory on the RT target when you captured the trace session and when the VIs executed on the RT target with respect to time from left to right. The LabVIEW Execution Trace Tool displays VI events in different colors to distinguish the execution priority of each VI. Table 1 lists the colors associated with each LabVIEW priority.

Table 1. Priority–Color Representation

Priority	Color
Time Critical	Red
High	Dark Pink
Above Normal	Light Pink
Normal	White
Background	Blue
Subroutine	Black

The LabVIEW Execution Trace Tool displays the time range for the current view below the VI Events view. Figure 3 shows the VI Events view of the trace session for the example application from Figure 1.

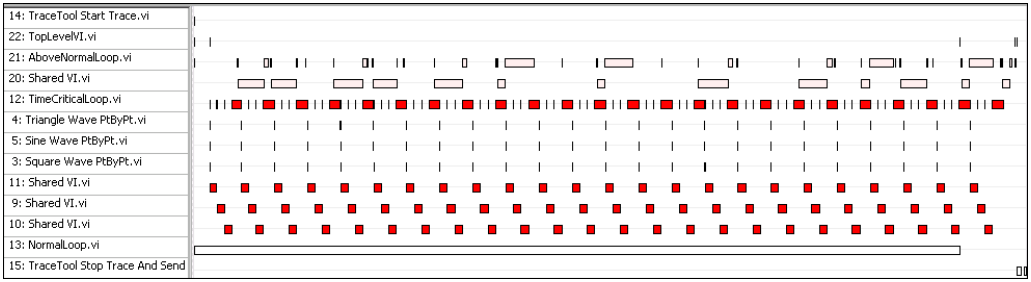

Figure 3. Example Application VI Events View

The left pane of the VI Events view lists the names of all VIs in memory on the RT target when you captured the trace session. The number that precedes the VI name is the dataspace ID assigned to the VI by the real-time operating system. The dataspace ID identifies the memory dataspace assigned to a VI. The Execution Trace Tool lists reentrant subVIs more than once with a different dataspace ID for each copy of the VI in memory, as shown with the Shared VI in Figure 3.

The LabVIEW Execution Trace Tool draws the events for each VI directly to the right of the VI name with respect to time. You can click on a VI name and drag it up or down in the list to rearrange the order of events in the VI Events view. The LabVIEW Execution Trace Tool redraws the VI Events view to show the new order.

You can use the tools in the LabVIEW Execution Trace Tool palette to display a specific region of the trace session. Figure 4 shows a zoomed region of the trace session shown in Figure 3.

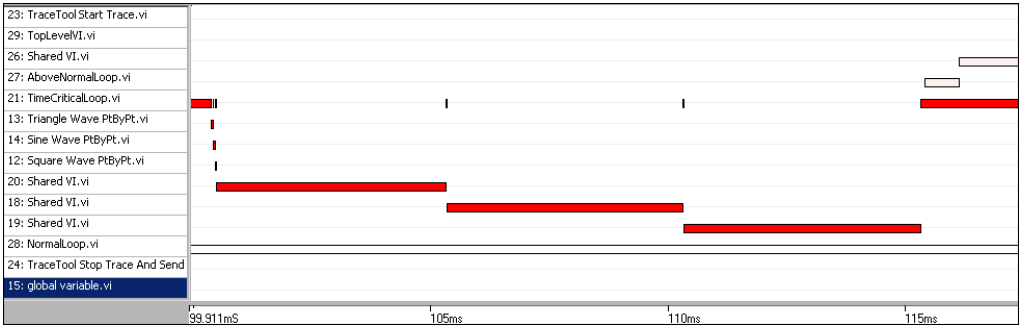

Figure 4. Zoomed View of the VI Events View

The zoomed region starts with the execution of the TimeCriticalLoop VI from Figure 5. The VI Events view shows that the TimeCriticalLoop VI launches three VIs that generate waveforms. The TimeCriticalLoop VI then launches three copies of the reentrant Shared VI, one after another.

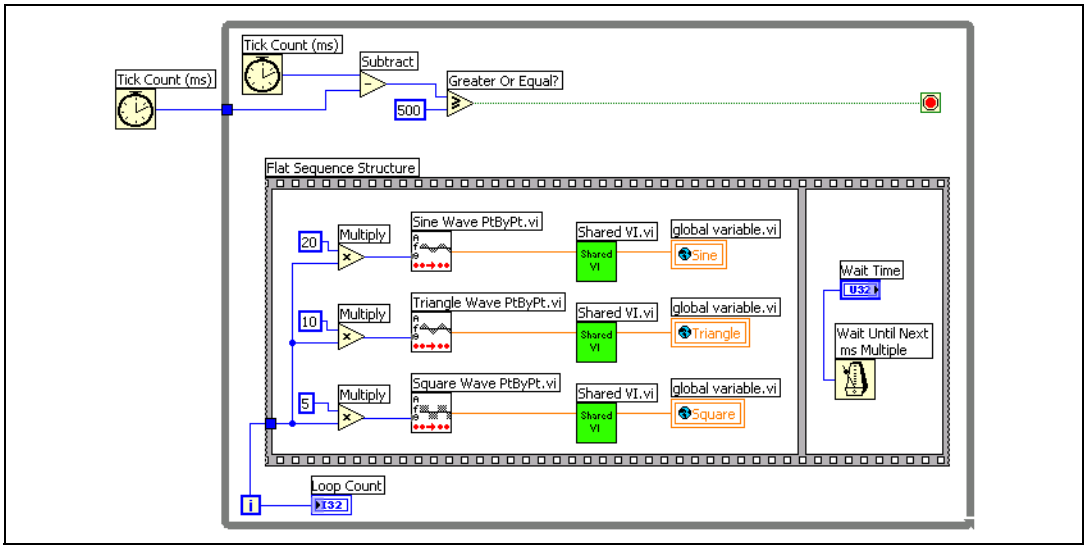

Figure 5. TimeCriticalLoop VI Block Diagram

The TimeCriticalLoop VI sends the three waveforms to the NormalLoop VI, shown in Figure 6, using global variables. The TimeCriticalLoop VI then sleeps, allowing other VIs to execute.

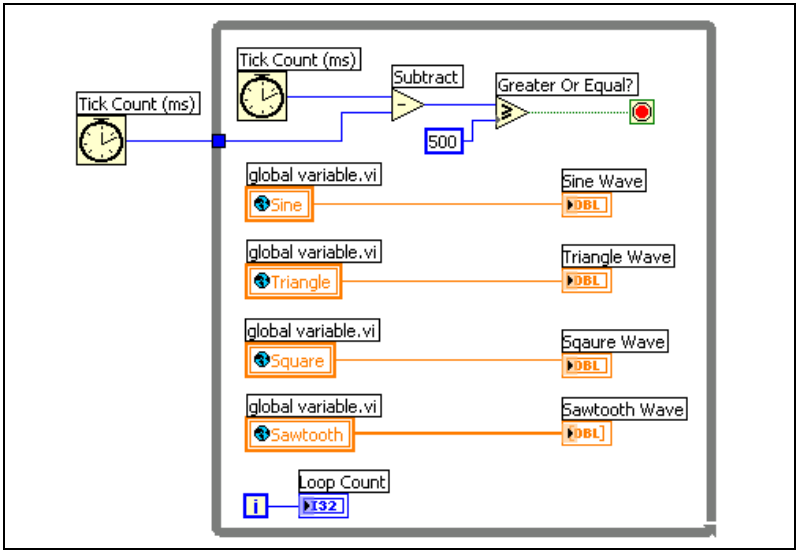

Figure 6. NormalLoop VI Block Diagram

Note The LabVIEW Execution Trace Tool lists the VI names of global variables in the left pane of the VI Events view but does not draw events for the variables.

Viewing and Analyzing Thread Events

The LabVIEW Execution Trace Tool displays thread event data in the Thread Events view. The Thread Events view shows the execution of threads in the real-time operating system of the RT target. The left pane of the Thread Events view lists the thread names preceded by a dataspace ID. The LabVIEW Execution Trace Tool shows the name of the execution system where a thread runs in brackets for all LabVIEW threads.

The LabVIEW Execution Trace Tool displays thread events in different colors to distinguish the execution priority of each thread. Table 1 lists the colors associated with each LabVIEW priority. The priority of thread events matches the priority of VI events of the same color. Figure 7 shows a section of the example trace session.

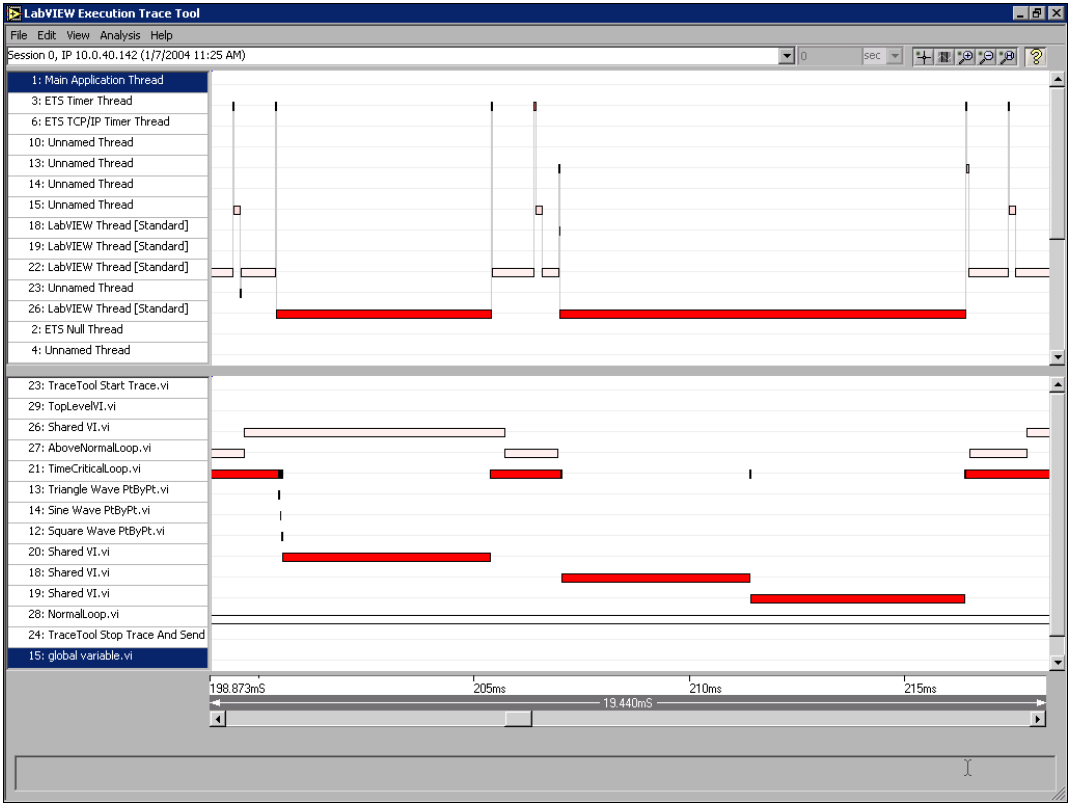

Figure 7. Thread Events View

The TimeCriticalLoop VI and the AboveNormalLoop VI appear to execute at the same time at about 206 ms. However, the Thread Events view shows a thread executed during this time running at a priority that matches the priority of the AboveNormalLoop VI.

The TimeCriticalLoop VI attempted to send a waveform to the NormalLoop VI using a global variable. The above normal priority thread running the AboveNormalLoop VI blocked the time-critical priority thread that was executing the TimeCriticalLoop VI from accessing the global variable. This example illustrates how resource contention causes unexpected timing behavior. The real-time operating system allowed a lower priority thread to block the time-critical thread in order to release the resource. Refer to the *LabVIEW Real-Time Module User Manual* for information about shared resources and resource contention.

Logging Analysis and Custom Events

The LabVIEW Execution Trace Tool can show when events such as sleep spans, memory manager calls, and resource mutexes occur in an application. You also can create custom events that you can log in a trace session to show the execution of specific sections of an application.

Note You must enable detailed tracing and thread tracing using the TraceTool Start Trace VI when you start the trace session to display analysis and custom events. Refer to the *Capturing a Trace Session* section for information about enabling detailed tracing when starting a trace session.

Displaying Analysis Events

The LabVIEW Execution Trace Tool can show the occurrence of specific analysis events in the trace session of an application. Select the analysis events you want to view from the **Analysis** menu of the LabVIEW Execution Trace Tool. The LabVIEW Execution Trace Tool displays a flag followed by a dashed line in the Thread Events view to indicate the time range for the occurrence of the analysis event. You can show the following analysis events:

- Sleep Span—Occurs when a thread sleeps to allow lower priority threads to execute. The LabVIEW Execution Trace Tool displays Sleep Span events using a blue flag.
- WaitFor(*)Object Span—Occurs when a thread must wait for access to a low-level resource before executing. The LabVIEW Execution Trace Tool displays WaitFor(*)Object Span events using a red flag.
- WaitForMem Span—Occurs when a thread conducts any memory management operation. The LabVIEW Execution Trace Tool displays WaitForMem Span events using a green flag.

Figure 8 shows a Sleep Span event that begins in an unnamed thread running at normal priority. The unnamed thread sleeps and other threads execute during the Sleep Span event. The unnamed thread continues to execute when the Sleep Span event ends.

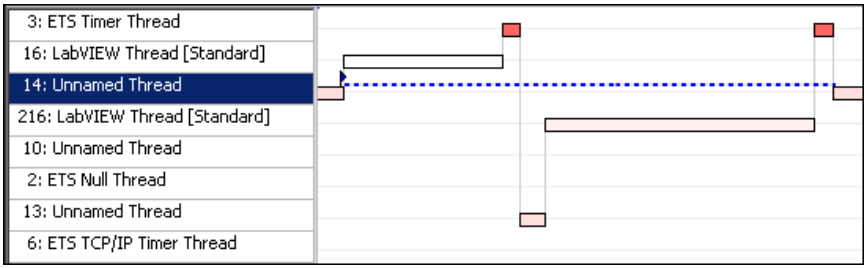

Figure 8. Analysis Event Flag

Logging and Displaying Custom Events

The LabVIEW Execution Trace Tool also can display custom user events. Select **Analysis»Custom Event Flags** to open the **Customize Event Flags** dialog box. Use the **Customize Events Flags** dialog box to create a custom event flag. You must assign a color and event code for the custom event flag. You can use any number from 0 to 255 to differentiate different custom events by event codes.

Use the TraceTool Log User Event VI to log a custom event in the trace session. The **Event ID** input of the TraceTool Log User Event VI must contain the event code of a custom event flag that you create in the **Customize Events Flags** dialog box.

Figure 9 shows the block diagram of the AboveNormalLoop VI from the example application with the TraceTool Log User Event VI added. The **Event ID** input receives the event code of 17. The trace session logs any occurrence of this custom event. You then can create a custom event flag in the LabVIEW Execution Trace Tool with an event code of 17 to display the occurrences of the custom event in the Thread Events view.

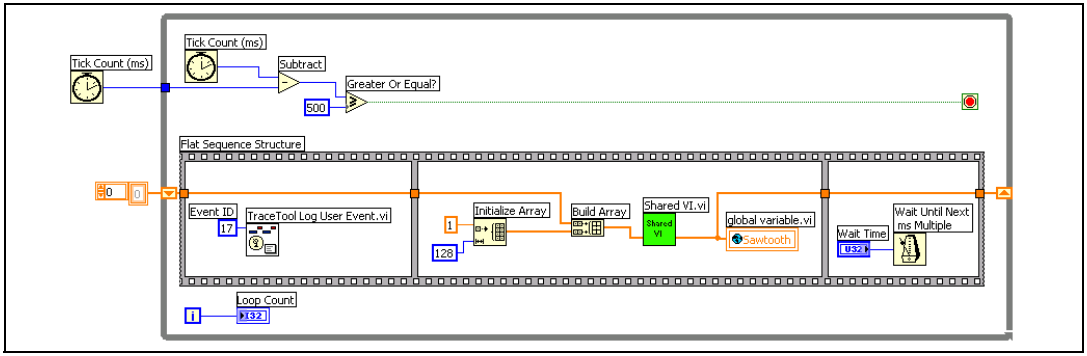

Figure 9. Logging Custom User Events in the AboveNormalLoop VI

Figure 10 shows a custom event logged in the AboveNormalLoop VI. The AboveNormalLoop VI executed in a LabVIEW thread running at normal priority and assigned to the Standard execution system. The custom event flag shows the exact time when the TraceTool Log User Event VI executes in the application.

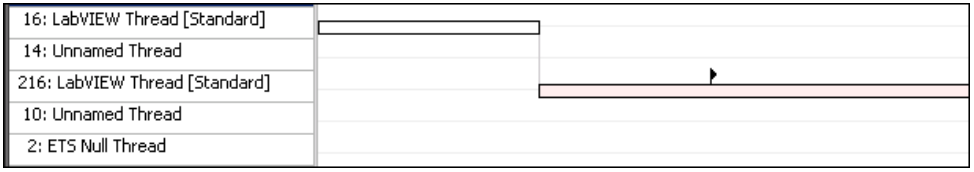

Figure 10. Custom Event Flag

Refer to the *LabVIEW Execution Trace Toolkit Help* for more information about how to create a custom event using the LabVIEW Execution Trace Tool.

Working with Trace Sessions

You can print the current trace session from the LabVIEW Execution Trace Tool. You also can save a trace session and then load the session in the LabVIEW Execution Trace Tool at a later time.

Printing Trace Sessions

You can print the current view of a trace session by selecting **File»Print Window** from the LabVIEW Execution Trace Tool. You can print the entire trace session, or you can print a specific range of the trace session by using the zoom tools to display the range before printing. Refer to the *Viewing Trace Sessions* section for information about using the tools on the LabVIEW Execution Trace Tool palette to display specific ranges of a trace session.

Saving Trace Sessions

After the LabVIEW Execution Trace Tool receives a trace session from the RT target, you can save the trace session on the host computer. Select **File»Save Session** to save the trace session to file. You can load the trace session into the LabVIEW Execution Trace Tool at a later time.

Select **File»Close Session** to close a trace session. You must save the trace session before closing if you want to load the session into the LabVIEW Execution Trace Tool at a later time.

Loading Trace Sessions

You can load saved trace sessions into the LabVIEW Execution Trace Tool by selecting **File»Open Session**. If you have more than one trace session open in the LabVIEW Execution Trace Tool, use the pull-down list at the top of the window to choose the trace session you want to view.

Troubleshooting

Refer to the *Troubleshooting* topic of the *LabVIEW Execution Trace Toolkit Help*, available by clicking the Troubleshooting button on the LabVIEW Execution Trace Tool palette, for answers to common questions you might have when using the LabVIEW Execution Trace Tool.

Where to Go from Here

The LabVIEW Execution Trace Toolkit provides example VIs that you can explore to learn more about using the LabVIEW Execution Trace Tool and the Execution Trace Tool VIs to analyze LabVIEW applications.

You can access the LabVIEW Execution Trace Toolkit examples by selecting **Help»Find Examples** in LabVIEW.

323738A-01 Apr04