RedHawk NightStar Tools Tutorial

CONCURRENT COMPUTER CORPORATION™

0898009-000
September 2002

The information contained in this document is believed to be correct at the time of publication. It is subject to change without notice. Concurrent Computer Corporation makes no warranties, expressed or implied, concerning the information contained in this document.

To report an error or comment on a specific portion of the manual, photocopy the page in question and mark the correction or comment on the copy. Mail the copy (and any additional comments) to Concurrent Computer Corporation, 2881 Gateway Drive, Pompano Beach, FL 33069-4324. Mark the envelope **"Attention: Publications Department**." This publication may not be reproduced for any other reason in any form without written permission of the publisher.

Printed in U. S. A.

Revision History:	Level:	Effective With:
Original Release -- September 2002	000	RedHawk Linux 1.1

Preface

General Information

The RedHawk™ NightStar™ Tools allow users on an iHawk™ system running RedHawk Linux® to schedule, monitor, debug and analyze the run time behavior of their real-time applications as well as the RedHawk Linux operating system kernel.

The RedHawk NightStar Tools consist of the NightTrace™ event analyzer, the Night-Sim™ frequency-based scheduler, and the NightProbe™ data monitoring tool.

Scope of Manual

This manual is a tutorial for the RedHawk NightStar Tools.

Structure of Manual

This manual consists of one chapter which is the tutorial for the RedHawk NightStar Tools.

Syntax Notation

The following notation is used throughout this guide:

italic	Books, reference cards, and items that the user must specify appear in *italic* type. Special terms and comments in code may also appear in *italic.*
`list bold`	User input appears in **`list bold`** type and must be entered exactly as shown. Names of directories, files, commands, options and man page references also appear in **`list bold`** type.
`list`	Operating system and program output such as prompts and messages and listings of files and programs appears in `list` type. Keywords also appear in `list` type.
emphasis	Words or phrases that require extra emphasis use emphasis type.
window	Keyboard sequences and window features such as push buttons, radio buttons, menu items, labels, and titles appear in window type.
`[ ]`	Brackets enclose command options and arguments that are optional. You do not type the brackets if you choose to specify such option or arguments.

`{ }`	Braces enclose mutually exclusive choices separated by the pipe (`|`) character, where one choice must be selected. You do not type the braces or the pipe character with the choice.
`...`	An ellipsis follows an item that can be repeated.
`::=`	This symbol means *is defined as* in Backus-Naur Form (BNF).

Referenced Publications

The following publications are referenced in this document:

0890398	*NightTrace Manual*
0890458	*NightSim User's Guide*
0890465	*NightProbe User's Guide*

Contents

Chapter 1 Using the RedHawk NightStar Tools

Overview 1-1
Before you begin 1-1
Getting Started 1-3
Building the program 1-3
Using NightSim 1-5
Invoking NightSim 1-5
Configuring the Scheduler 1-5
Scheduling a process 1-7
Setting up the scheduler 1-9
Starting the simulation 1-9
Monitoring the simulation 1-10
Using NightProbe 1-11
Invoking NightProbe 1-11
Configuring NightProbe 1-12
Connecting to the target program 1-14
Starting sampling 1-15
Modifying program data 1-15
Using NightTrace 1-17
Invoking NightTrace 1-17
Configuring a kernel daemon 1-18
Configuring a user daemon 1-19
Creating the daemons 1-21
Resuming daemon execution 1-21
Flushing the buffers 1-21
Stopping the daemons 1-22
Displaying the kernel trace data 1-23
Creating a customized display page 1-24
Displaying the user trace data 1-25
Zooming in 1-26
Examining the kernel trace data 1-29
Exiting the tools 1-30
Exiting NightTrace 1-30
Exiting NightProbe 1-30
Exiting NightSim 1-30
Conclusion 1-31

Illustrations

Figure 1-1. NightSim Scheduler 1-5
Figure 1-2. NightSim Edit Process 1-7
Figure 1-3. Starting the simulation 1-9
Figure 1-4. NightSim Monitor 1-10
Figure 1-5. NightProbe Data Recording window 1-12
Figure 1-6. Configured NightProbe Data Recording window 1-13

Figure 1-7. NightProbe Spreadsheet Viewer window 1-14
Figure 1-8. User Authentication dialog 1-15
Figure 1-9. Modified values in NightProbe Spreadsheet Viewer 1-16
Figure 1-10. NightTrace main window 1-17
Figure 1-11. Daemon Definition dialog 1-19
Figure 1-12. Daemon Definition dialog 1-20
Figure 1-13. NightTrace main window with stopped daemons 1-23
Figure 1-14. NightTrace kernel display page 1-24
Figure 1-15. Customized NightTrace display page 1-25
Figure 1-16. User trace data in customized NightTrace display page 1-26
Figure 1-17. Zoomed in view of user trace data 1-27
Figure 1-18. Extreme zoomed in view of user trace data 1-28
Figure 1-19. Zoomed in view of kernel display page 1-29
Figure 1-20. Removing the scheduler 1-31
Figure 1-21. Removing the scheduler 1-31

1
Using the RedHawk NightStar Tools

The RedHawk NightStar Tools allow users on an iHawk system running RedHawk Linux to schedule, monitor, debug and analyze the run time behavior of their real-time applications as well as the RedHawk Linux operating system kernel.

The RedHawk NightStar Tools consist of the NightTrace event analyzer, the NightSim frequency-based scheduler, and the NightProbe data monitoring tool.

Overview

This is a demonstration of the RedHawk NightStar Tools. In this tutorial, we will use the following RedHawk NightStar Tools:

- NightSim
- NightProbe
- NightTrace

integrating them together into one cohesive example.

Please see “Before you begin” on page 1-1 for some important recommendations and considerations.

Before you begin

To automatically ensure that all files your user creates on the RedHawk Linux system are publicly readable and writeable, include the following command in your shell startup script:

```
umask 000
```

NOTE

This is important for the operation of the tutorial to succeed.

In addition, some of the activities in the RedHawk NightStar Tools Tutorial require either root access or user registration in the `fbscheduser` capabilities role. Either execute the

commands shown in the tutorial as the `root` user, or have your system administrator register you as an FBS user according to the following instructions:

1. Add the following line to the **`/etc/pam.d/rsh`** and **`/etc/pam.d/login`** files:

   ```
   session required /lib/security/pam_capability.so
   ```

2. Add the following line to the bottom of the **`/etc/security/capability.conf`** file:

   ```
   user user fbscheduser
   ```

   where *user* is the login name of the desired user.

After these activities are complete, you must log off and log back onto the RedHawk system.

Getting Started

We will start by creating a directory in which we will do all our work. On the RedHawk Linux system, create a directory and position yourself in it:

To create a working directory

- Use the **mkdir(1)** command to create a working directory.

  We will name our directory **tutorial** using the following command:

  ```
  mkdir tutorial
  ```

- Position yourself in the newly created directory using the **cd(1)** command:

  ```
  cd tutorial
  ```

Source files, as well as configuration files for the various tools, are included on the RedHawk NightStar Tools Installation CD. We will copy these tutorial-related files to our **tutorial** directory.

To copy the files from the RedHawk NightStar Tools Installation CD

- Insert the RedHawk NightStar Tools Installation CD in the CD-ROM drive.
- Mount the CD-ROM drive (assuming the standard mount entry for the CD-ROM device exists in **/etc/fstab**).

  ```
  mount /mnt/cdrom
  ```

- Copy all tutorial-related files to our local directory.

  ```
  cp /mnt/cdrom/tutorial/* ./tutorial
  ```

- Unmount the CD-ROM drive (otherwise, you will be unable to remove the RedHawk NightStar Tools Installation CD from the CD-ROM drive).

  ```
  umount /mnt/cdrom
  ```

Building the program

Our example uses a cyclic program which intends to do some work every time an external event triggers it.

We will use RedHawk Linux's Frequency Based Scheduler to control the execution of the program. The Frequency Based Scheduler allows us to field an external interrupt and control the execution of one or more programs.

A portion of one of the source files, **sim.c**, is shown below:

```
main()
{
   counters.SetWorkload(0);

   trace_setup ("sim-data") ;

   while (fbswait() == 0) {
      timer.start();
      counters.Increment(1);
      trace_event_arg (cycle_start, counters.Get());
      counters.Work();
      timer.stop();
      trace_event_arg (cycle_end, counters.Get() % 10);
      counters.cycle_time = (float) timer.elapsed();
   }
}
```

The program calls `fbswait()` which will cause it to block until the frequency-based scheduler determines that it is time for this program to execute.

At that time, the program enters the loop where it increments some counters, logs a trace point with the NightTrace API `trace_event_arg()`, calls the procedure `counters.Work()`, logs another trace point to signal the end of the calculations done by `counters.Work()`, then returns to `fbswait()` to await the next cycle.

You only need to make a single FBS API call, `fbswait()`, to have a program which can be scheduled on the FBS.

Now that we have the source files, we need to build the program. We will use the **g++** compiler.

To build the executable

- From the local **tutorial** directory, enter the following commands:

```
g++ -c -g *.c
g++ -o sim *.o -lntrace -lccur_fbsched -lccur_rt
```

NOTE

NightProbe requires that the user application is built with DWARF debugging information in order to read symbol table information from user application program files. This enables NightProbe to determine which variables may be probed. For this reason, the **-g** compile option is specified. However, when compiling with releases prior to **gcc** 3.2, it is necessary to use the **-gdwarf-2** option in place of the **-g** option. Otherwise, symbols will not be visible in NightProbe.

Using NightSim

Because our sample program uses the frequency-based scheduler, we will use the NightSim Scheduler to schedule the process. NightSim is a tool for scheduling and monitoring real-time applications which require predictable, repetitive process execution. NightSim provides a graphical interface to the RedHawk Linux frequency-based scheduler and performance monitor. With NightSim, application builders can control and dynamically adjust the periodic execution of multiple coordinated processes, their priorities, and their CPU assignments. NightSim's performance monitor tracks the CPU utilization of individual processes and provides a customizable display of period times, minimums, maximums, and frame overruns. For more information on NightSim, refer to the *NightSim User's Guide* (0890480).

Invoking NightSim

To invoke the NightSim Scheduler

- From the local **`tutorial`** directory, enter the following command:

```
nsim &
```

Configuring the Scheduler

The NightSim Scheduler window is opened, ready for us to configure it for our particular simulation.

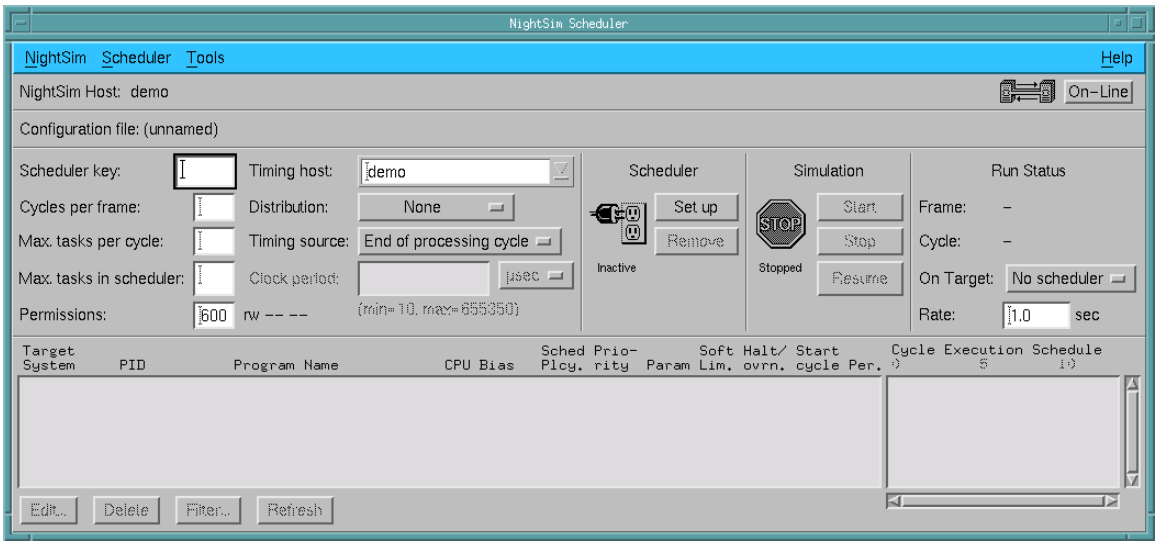

Figure 1-1. NightSim Scheduler

The FBS schedules processes in a cyclic manner based on some (usually cyclic) interrupt source.

We use the term *cycle*, or *minor cycle*, to represent the smallest amount of time between occurrences of the interrupt.

We use the term *frame*, or *major frame*, as simply a convenience to represent a set of one or more cycles. Often, the most simple schedulers have 1 cycle per frame. More complex applications may have different sets of activities that need to be accomplished before the entire application repeats; such applications would define multiple cycles per frame.

To configure a NightSim Scheduler

- Specify a Scheduler key. The key is a user-chosen numeric identifier with which the scheduler will be associated. For our example, we will use 1000.
- Specify the Cycles per frame. This field allows you to specify the number of cycles that compose a frame on the specified scheduler. We will use the value 5.
- Specify the Max. tasks per cycle. This field allows you to specify the maximum number of processes that can be scheduled to execute during one cycle. Enter 5 for our example.
- Specify the Max. tasks in scheduler. This field allows you to specify the maximum number of processes that can be scheduled on the specified scheduler at one time. For our example, we will specify the value 5.
- Enter the name of a RedHawk Linux system which will act as the Timing host for the simulation. You may use the drop down list associated with this field for the names of systems previously used as timing hosts. For our example, we will enter demo, an iHawk system.

NOTE

When NightSim is operating in On-Line mode, an attempt will be made to communicate with the system specified as the timing host. The user may experience a slight delay and the message Talking to Server... will appear in the Configuration File Name Area of the NightSim Scheduler as this occurs. See the *NightSim User's Guide* (0890480) for more information.

- Select a Timing source from the list provided. This list contains the set of devices available on the timing host. We will use Real-time clock 2c0.
- Specify Clock period.

  For our simulation, we would like the real-time clock to "fire" every .01 seconds (or 10000 microseconds).

  For our example, we will specify 10000 for the number of microseconds.

Scheduling a process

Once we have properly configured the Scheduler, we can add a process to the frequency-based scheduler.

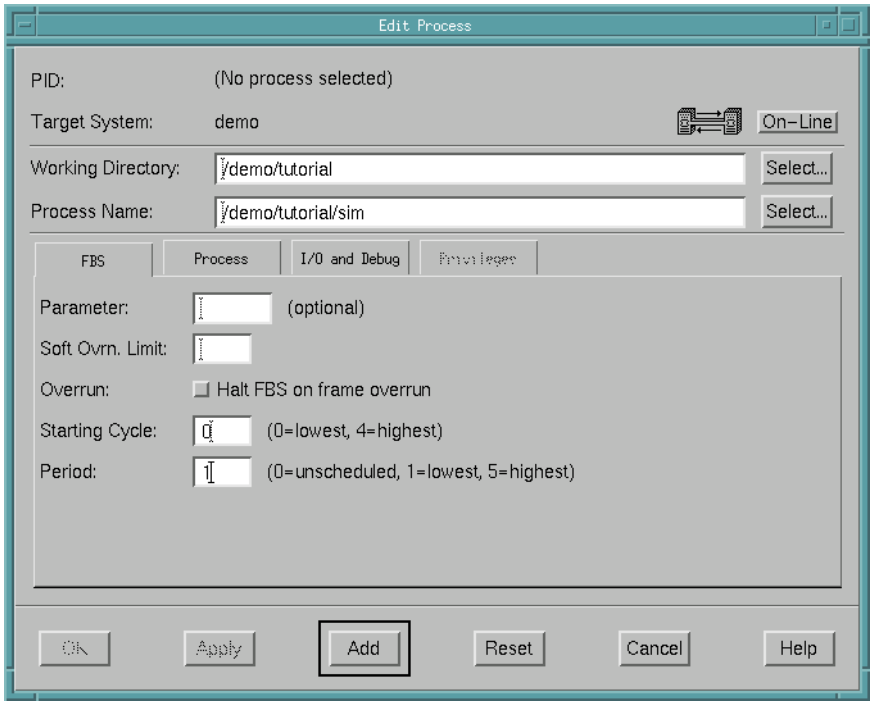

Figure 1-2. NightSim Edit Process

To add a process to the frequency-based scheduler

- Press the Edit… button on the NightSim Scheduler window. This will bring up the Edit Process window.
- Press the Select… button next to the Process Name field. This brings up the Select a Program dialog.
  - Choose the program we wish to schedule from the Files list. For our example, we will select **`sim`** from the list.
  - Press Select to select the program.
- Ensure that the Working Directory is the same directory that contains our program (the directory of the Process Name selected in the previous step).
- Click on the FBS tab:
  - Select Starting Cycle.

    This field allows you to specify the first minor cycle in which the specified program is to be wakened in each major frame.

We will choose the lowest value, 0, for our example.

- Select Period.

  This field allows you to establish the frequency with which the specified program is to be awakened in each major frame. Enter the number of minor cycles representing the frequency with which you wish the program to be awakened.

  For our example, we will specify a period of 1, indicating that the specified program is to be awakened every minor cycle.

- Click on the Process tab:
  - Click on the All CPUs checkbox to deselect all of the CPUs
  - Choose a single CPU for this process to run on.

    For our example, we will specify CPU 0 by clicking on the checkbox labeled 0.

  - Specify the Priority for this process.

    The range of priority values that you can enter is governed by the scheduling policy specified. NightSim displays the range of priority values that you can enter next to the Priority field. Higher numerical values correspond to more favorable scheduling priorities.

    For our example, we will give the process a priority of 50.

- Press Add to add the process to the frequency-based scheduler.

We would also like to measure the idle time on the same CPU. We can do this by scheduling the **/idle** process.

To schedule the /idle process

- In the Edit Process window, enter:

  /idle

  in the Process Name field.

- Press the Add button to add the **/idle** process.
- Press the Close button to dismiss the Edit Process window.

You will notice that two entries now appear in the Process Scheduling Area of the NightSim Scheduler window.

Setting up the scheduler

To set up the scheduler

- In the NightSim Scheduler window, press the Set up button.

  This action:

  - creates a scheduler that is configured according to the parameters we specified
  - schedules the processes that we have added to the NightSim Scheduler window and starts them running up to the first `fbswait()` call, and
  - attaches the timing source to the scheduler.

Starting the simulation

Now we need to start the simulation. When you click on the Start button, NightSim carries out the following actions:

- Attaches the timing source to the scheduler if not already attached or if the timing source has been changed
- If a real-time clock is being used as the timing source, sets the clock period in accordance with the value entered in the Clock period field in the Scheduler Configuration Area
- Starts the simulation with the values of the *minor cycle*, *major frame*, and *overrun* counts set to zero

To start a simulation in NightSim

- Press the Start button on the NightSim Scheduler window.

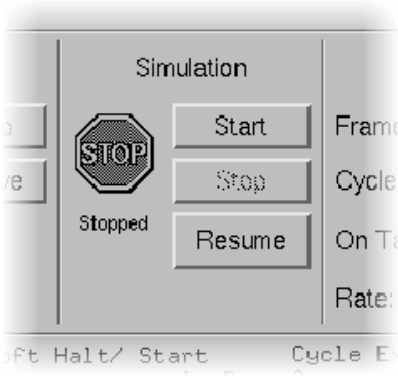

Figure 1-3. Starting the simulation

When the simulation begins, you should notice the values for Frame and Cycle in the Run Status Area begin to change.

Monitoring the simulation

The performance monitor is a mechanism that enables you to monitor FBS–scheduled processes' utilization of a CPU.

The performance monitor provides you with the ability to:

- Obtain performance monitor values by process or processor
- Start and stop performance monitoring by process
- Clear performance monitor values by processor

To create a performance monitor window

- Select Create Monitor Window from the NightSim menu on the NightSim Scheduler window.

Figure 1-4. NightSim Monitor

Notice the value under the Last Time column for the process **sim**. This value shows the amount of time (in microseconds) that the process has spent running between the last time that it was wakened by the scheduler and the next time it called `fbswait()`.

Using NightProbe

The NightProbe Data Monitoring Tool is a real-time graphical tool for monitoring, recording, and altering program data within one or more executing programs without intrusion. It can be used in a development environment as a tool for debugging, or in a production environment to create a "control panel" for program input and output.

NightProbe utilizes a non-intrusive technique of mapping the application's address space into its own. Subsequent direct reads and writes by NightProbe allow it to sample and modify user data without interrupting or otherwise affecting the user process.

There is no API for NightProbe. Applications need only ensure that their debug information is generated with the DWARF format by using the **`-g`** compilation option. Even without symbols, however, NightProbe can probe processes based on virtual addresses alone.

NOTE

When compiling with releases prior to **`gcc`** 3.2, it is necessary to use the **`-gdwarf-2`** option in place of the **`-g`** option. Otherwise, symbols will not be visible in NightProbe.

For more information on NightProbe, refer to the *NightProbe User's Guide* (0890465).

Invoking NightProbe

To invoke the NightProbe Data Monitoring Tool

- From the Tools menu of the NightSim Scheduler window, select NightProbe Data Recorder/Monitor.

The NightProbe Data Recording window is opened.

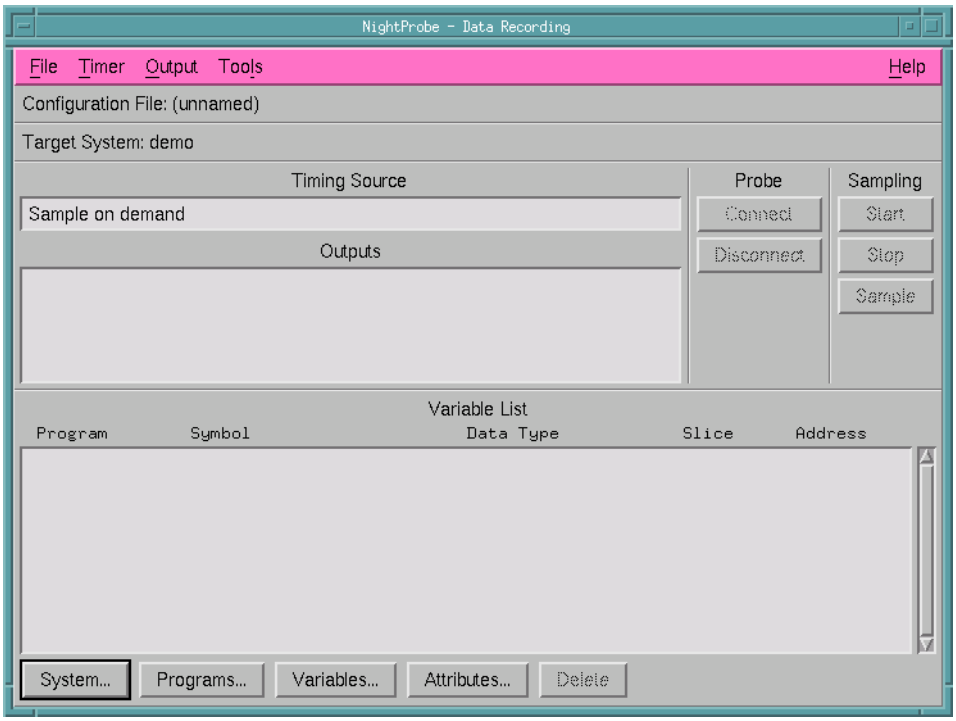

Figure 1-5. NightProbe Data Recording window

Configuring NightProbe

Our example will use a configuration file shipped with the RedHawk NightStar Tools Installation CD to configure NightProbe. This file, named **`nprobe.config`**, was copied to our local **`tutorial`** directory earlier in the step “Getting Started” on page 1-3.

To configure the NightProbe Data Monitoring Tool

- Select Open Config File... from the File menu of the NightProbe Data Recording window.

  You will be presented with a File Selection dialog.

- Maneuver to the local **`tutorial`** directory, if necessary.
- Select the file **`nprobe.config`** from the list of Files.
- Press OK to dismiss the dialog.

You should see the following members of the `counters` class listed in the Variable List area of the NightProbe Data Recording window:

- `counters.cycle_time`
- `counters.i_counter`
- `counters.workload`

We will be probing and modifying these variables.

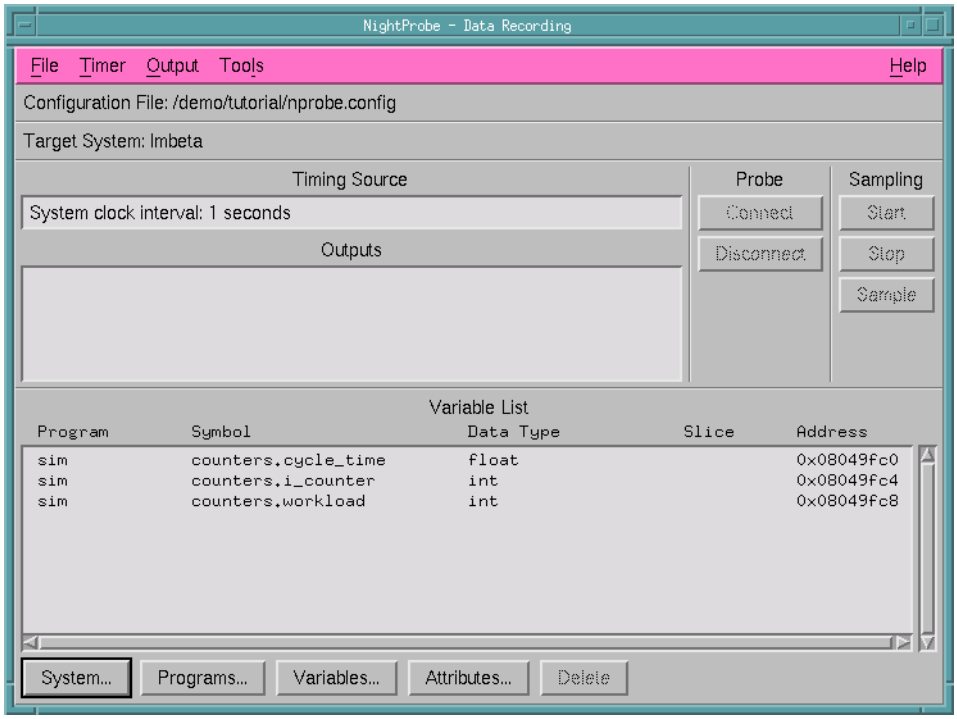

Figure 1-6. Configured NightProbe Data Recording window

Since the **`nprobe.config`** file specifies that NightProbe is to direct its output to a spreadsheet window, the Spreadsheet Viewer window is automatically opened as well.

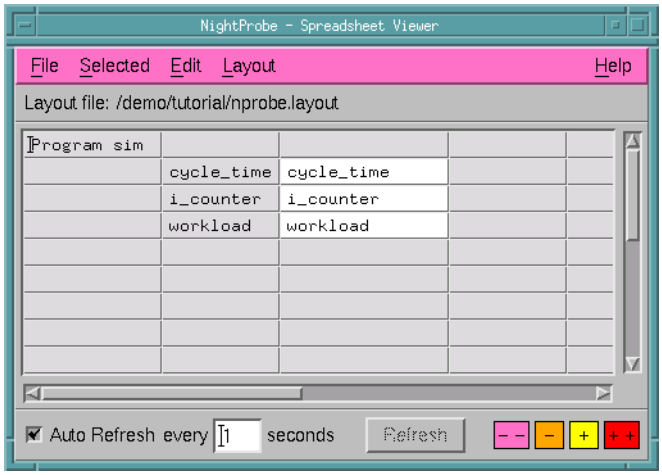

Figure 1-7. NightProbe Spreadsheet Viewer window

Connecting to the target program

When you are ready to perform data recording or monitoring, you must first connect NightProbe to a real-time NightProbe server on the target system.

The real-time NightProbe server performs initialization during the connection phase - opening output devices, verifying target processes, and mapping target process variable addresses.

The probed applications are not affected by this operation.

The real-time NightProbe server is the actual process that will read and write values from and to the user application's address space.

To connect to the target program

- Press the Connect button on the NightProbe Data Recording window.
- When presented with the User Authentication dialog, enter the login name of the user on the target system in the User field along with the corresponding password in the Password field.
- Press the OK button to continue.

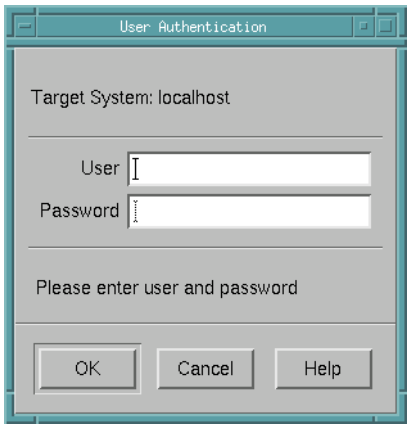

Figure 1-8. User Authentication dialog

Starting sampling

Once connected, we are ready to begin data recording.

Once started, the NightProbe server process will sample data based on the timing selection and will send the output to all specified output methods.

When we configured NightProbe (see "Configuring NightProbe" on page 1-12), we defined the timing selection to be the system clock (which fires once every second) and selected the Spreadsheet Viewer window as our output method.

To start sampling

- Press the Start button on the NightProbe Data Recording window.

  Note that the values in the Spreadsheet Viewer window will begin to change once a second.

  The user application that we are probing independently measures the time it takes for each cycle and saves that value in `counters.cycle_time`.

  Note that the value of `counters.cycle_time` (in units of seconds) is approximately the same as the Last Time statistic (in units of microseconds) in the NightSim Monitor window. (It will be slightly less than the value shown in the NightSim Monitor window because the application's calculations do not include all of its per-cycle activities.)

Modifying program data

NightProbe allows you to monitor and modify target locations while the program is running. We will modify the **`sim`** variable `counters.workload` to increase the amount of work the program does.

To modify the value of a variable

- In the Spreadsheet Viewer window, click on the value next to the label `workload`.
- Enter the value 10000.

  Notice that the value for `cycle_time` has increased significantly. In our example, it is now approximately 0.00035 seconds (this value is dependent on your machine speed). (You can also see this reflected in the Last Time statistic in the NightSim Monitor window.)

  In addition, the color of the cell containing the value of `cycle_time` has changed to yellow. NightProbe allows you to define caution and danger values for variables displayed in spreadsheets. Since the attributes for this cell (which were included in the configuration file **`nprobe.config`** - see “Configuring NightProbe” on page 1-12) specify that when the value exceeds 0.0002, the color of the cell will change to yellow signifying a state of high caution.

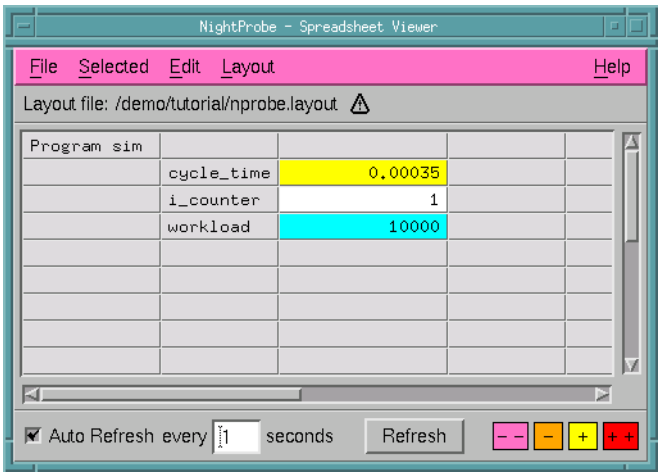

Figure 1-9. Modified values in NightProbe Spreadsheet Viewer

Experiment with different `workload` values until the color of the `cycle_time` value changes from the default background color to yellow and then to red.

Using NightTrace

NightTrace is a graphical tool for analyzing the dynamic behavior of single and multiprocessor applications. NightTrace can log application data events from simultaneous processes executing on multiple CPUs or even multiple systems. NightTrace combines application events with RedHawk Linux events and presents a synchronized view of the entire system. NightTrace allows users to zoom, search, filter, summarize, and analyze events in a wide variety of ways. RedHawk Linux events include individual system calls, context switches, machine exceptions, page faults and interrupts. Application events are defined by the user allowing logging of the data items associated with each event.

NightTrace allows users to manage user and kernel NightTrace daemons, providing the user with the ability to start, stop, pause, and resume execution of any of the daemons under its management.

Invoking NightTrace

To invoke NightTrace

- From the Tools menu of the NightProbe Data Recording window, select the NightTrace System Tracing and Analysis menu item.

  The NightTrace main window is opened.

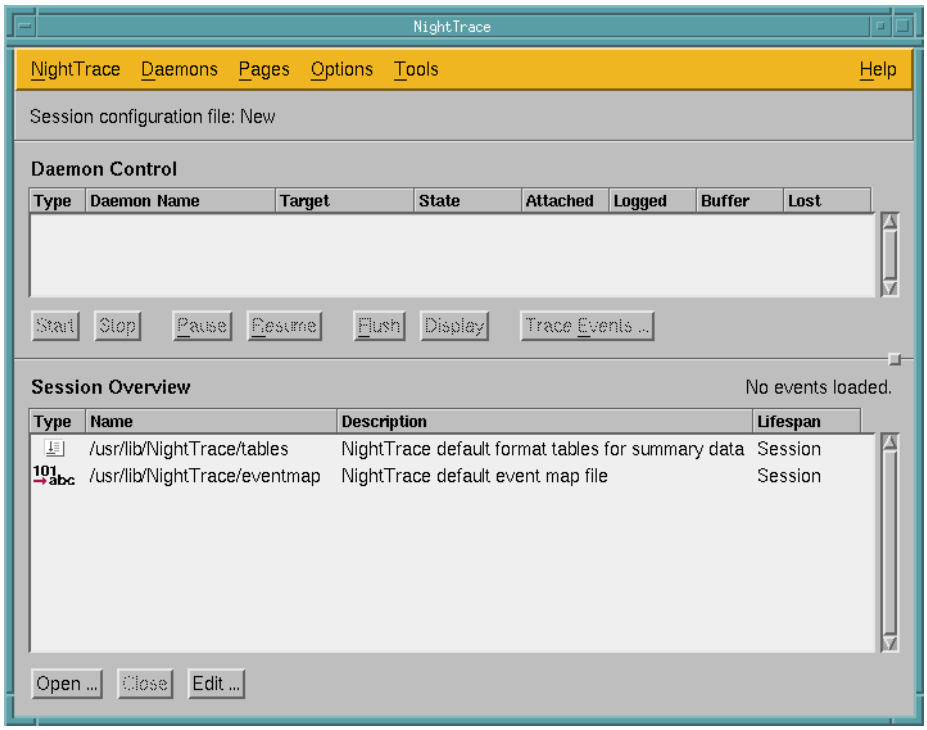

Figure 1-10. NightTrace main window

NightTrace allows users to manage user and kernel NightTrace daemons. It provides users with the ability to define a session consisting of one or more daemon definitions which can be saved for future use. These definitions include daemon collection modes and settings, daemon priorities and CPU bindings, and data output formats, as well as the trace event types that are logged by that particular daemon.

Using NightTrace, users can manage multiple daemons simultaneously on multiple target systems from a central location.

NightTrace offers the user the ability to start, stop, pause, and resume execution of any of the daemons under its management. The user may also view statistics as trace data is being gathered as well as dynamically enable and disable events while a particular daemon is executing.

Configuring a kernel daemon

NightTrace allows the user to configure a kernel daemon to collect data about the execution time of interrupts, exceptions, system calls, context switches, and I/O to various devices.

To configure a kernel daemon

- From the Daemons menu on the NightTrace main window, select the New... menu item.

  The Daemon Definition dialog is displayed.

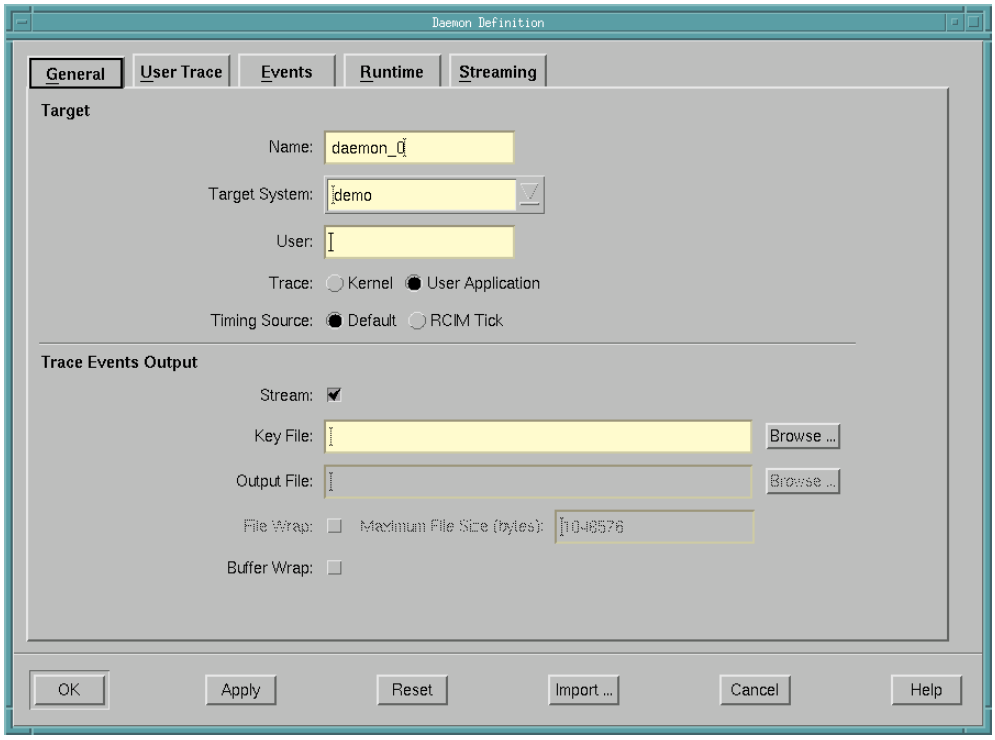

Figure 1-11. Daemon Definition dialog

- Select the Kernel radiobutton located in the Target section on the General page to indicate that we want this daemon to collect kernel events.
- Uncheck the Stream checkbox located in the Trace Events Output section on the General page so that our output can be sent to an output file.
- Enter the name `kernel-data` into the Output File text field.
- Press OK to complete the configuration of this daemon.

Configuring a user daemon

NightTrace allows the user to configure a user daemon to collect user trace events.

User trace events are generated by:

- user applications that use the NightTrace API
- the NightProbe tool (see the description of the To NightTrace menu item in the chapter titled “Using the Data Recording Window” in the *NightProbe User’s Guide* (0890480).

We will configure a user daemon to collect the events that our **`sim`** program logs.

To configure a user daemon

- From the Daemons menu on the NightTrace main window, select the New... menu item.

  The Daemon Definition dialog is displayed.

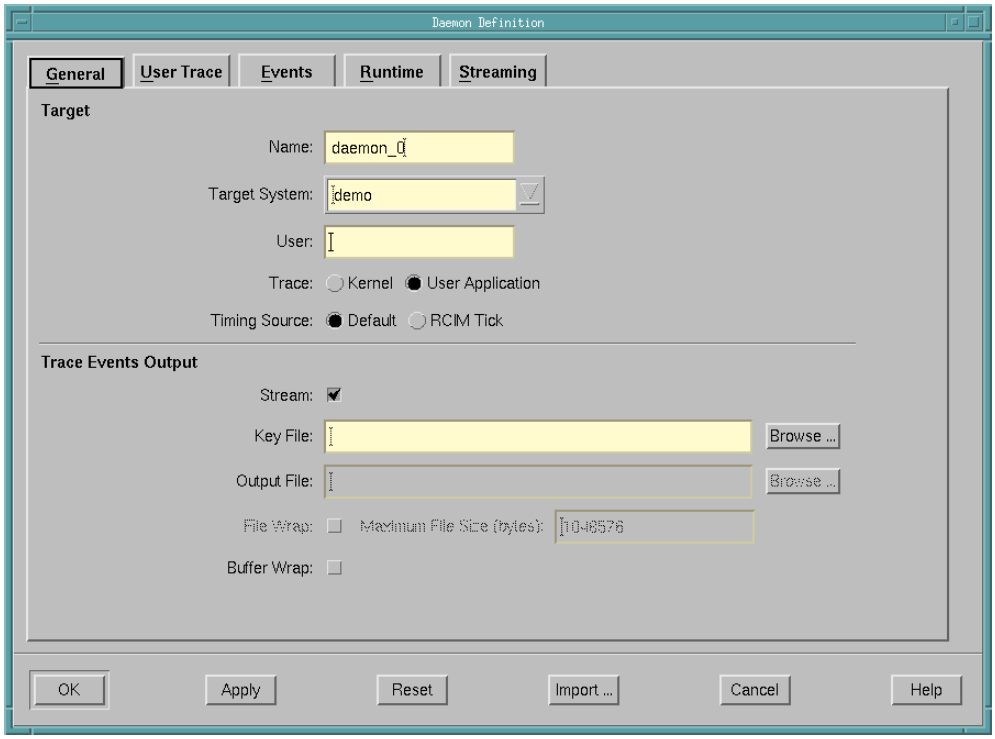

Figure 1-12. Daemon Definition dialog

- Enter the login name of the user on the target system in the User field.
- Select the User Application radiobutton located in the Target section on the General page to indicate that we want this daemon to collect user trace events.
- Uncheck the Stream checkbox located in the Trace Events Output section on the General page so that our output can be sent to an output file.
- Specify a Key File.

  The key file is critical as it links the user daemon to specific user applications. Our **`sim`** application passed the value `sim-data` to the NightTrace API `trace_begin()` call as the name of the *key_file*. (See the source file **`sim.c`** for details.)

  Enter `sim-data` in the Key File text field.

- Press OK to complete the configuration of this daemon.

Creating the daemons

Once the daemons are configured, they must be created before they can begin collecting events.

To create the daemons

- Select both daemons in the Daemon Details Area of the NightTrace main window.
- Press Start.

  The daemons are now created and ready to capture data. Note that both daemons are in a Paused state.

NOTE

Starting a daemon does not imply that the daemon begins to collect events.

Resuming daemon execution

Now that the daemons are configured and created, waiting in a Paused state, we may resume their execution so they may begin collecting events.

To resume daemon execution

- Select both daemons in the Daemon Details Area of the NightTrace main window.
- Press Resume.

  The state of both daemons changes from Paused to Logging as they begin to collect trace data.

Flushing the buffers

The Buffer column in the Daemon Details Area of the NightTrace main window specifies the number of trace events currently held in the buffer.

These events will be flushed from the buffer either when the flush threshold is reached or when the user flushes them manually by pressing the Flush button.

To flush the buffers manually

- Wait a few seconds until you have a thousand events or so in the Buffer column for each daemon.

- Select both daemons in the Daemon Details Area of the NightTrace main window.
- Press Flush.

  You now should see numbers in the Logged column which indicates that the events held in the buffer have been transferred to the output files selected during the configuration of the daemons. (See “Configuring a kernel daemon” on page 1-18 and “Configuring a user daemon” on page 1-19 for details.)

Stopping the daemons

NOTE

Starting and stopping the daemons are time consuming and involve possibly connecting to a target system, as well as user authentication, etc. Once the daemon is started, it is more efficient to utilize the Pause and Resume operations which require less time and resources. However, since we are finished with the daemons for this example, we can stop both of them.

To stop the daemons

- Select both daemons in the Daemon Details Area of the NightTrace main window.
- Press Stop.

  Note that the state of both daemons changes to Stopped.

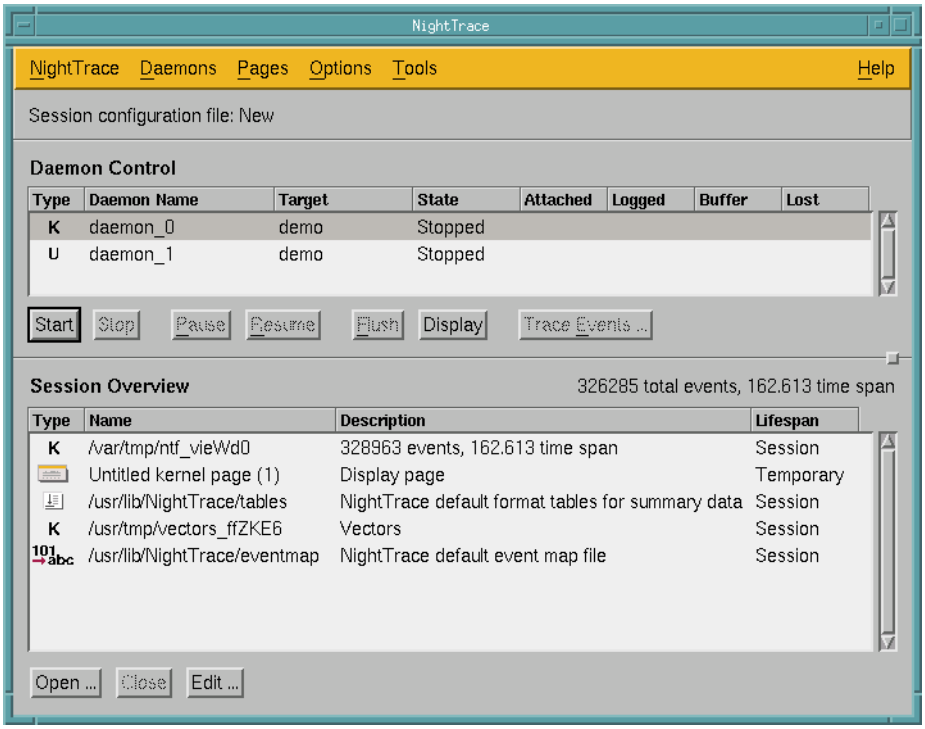

Figure 1-13. NightTrace main window with stopped daemons

Displaying the kernel trace data

Now that we have collected trace data from the RedHawk Linux kernel, we can display that data in a NightTrace kernel display page.

To display the kernel trace data

- Select *only* the kernel daemon in the Daemon Details Area of the NightTrace main window (as indicated by the K in the Type column).
- Press Display.

  A NightTrace kernel display page appears.

- Press the Zoom Out button on the kernel display page until data fills the grid area.

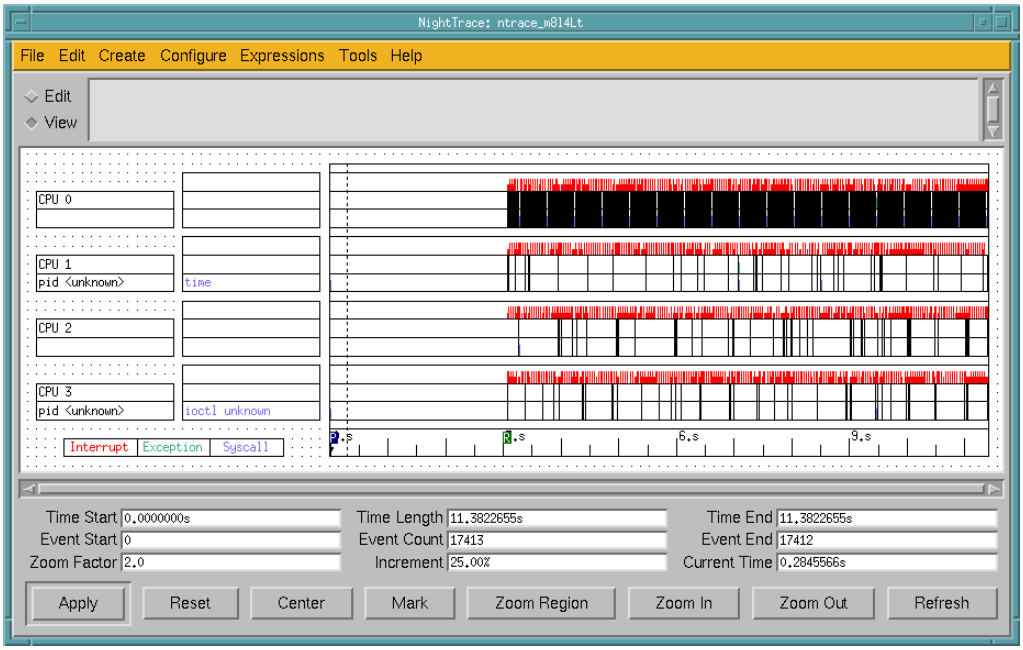

Figure 1-14. NightTrace kernel display page

Creating a customized display page

Now that we have collected trace data from our user application, **`sim`**, we can display that data in a NightTrace display page.

For this example, we would like to use a customized display page so we will use the configuration file shipped with the RedHawk NightStar Tools Installation CD. This file, named **`ntrace.config`**, was copied to our local **`tutorial`** directory earlier in the step "Getting Started" on page 1-3.

To create a customized display page

- Press the Open… button at the bottom of the NightTrace main window.

  You will be presented with an Open dialog.

- Select the file **`ntrace.config`** from the list.

- Press Open to create the display page as specified by the configuration file.

  The customized NightTrace display page is presented.

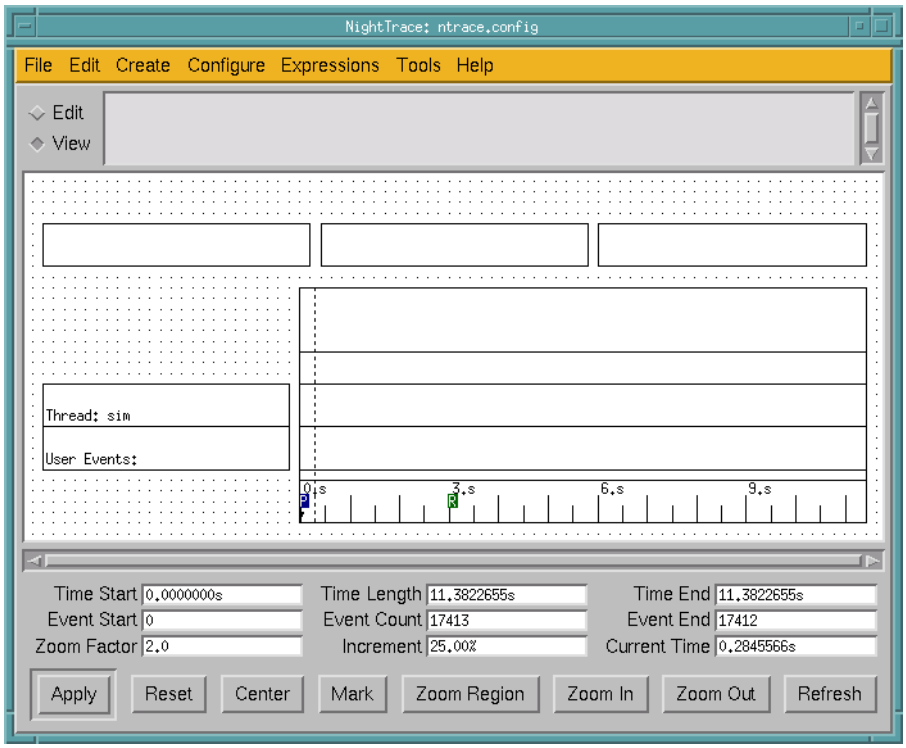

Figure 1-15. Customized NightTrace display page

Displaying the user trace data

Now that we have our customized display page, we can display the user trace data.

To display the user trace data

- Select the user trace daemon in the Daemon Details Area of the NightTrace main window (as indicated by the U in the Type column).
- Press Display.
- Press the Zoom Out button on the user display page until data fills the grid area.

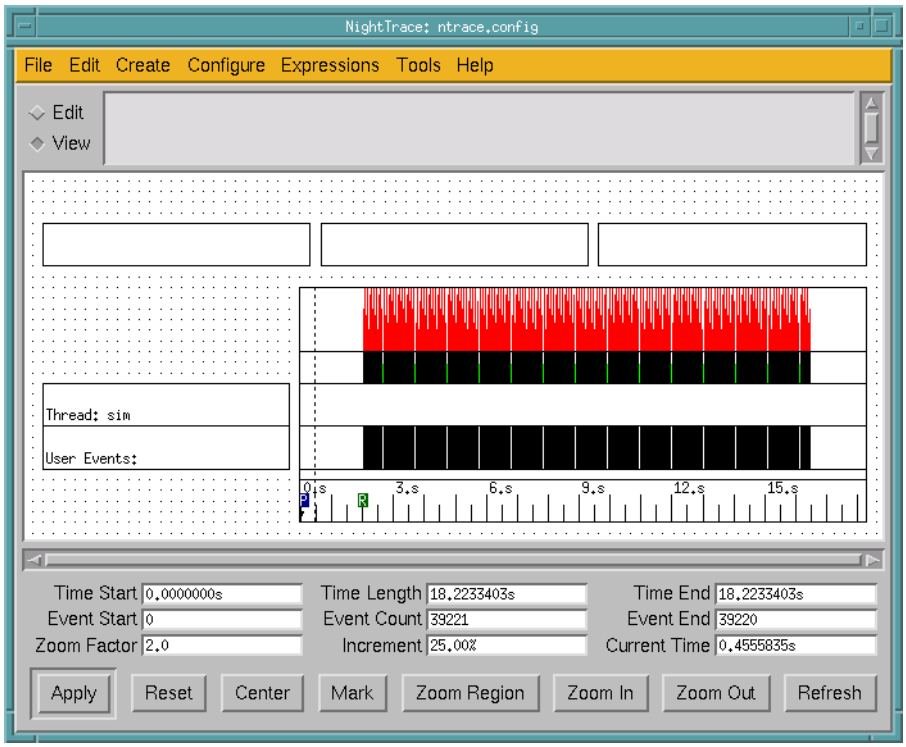

Figure 1-16. User trace data in customized NightTrace display page

NOTE

This display page is configured to only display events from the user application.

Zooming in

We can see a finer level of detail by zooming in on the user trace display page.

To zoom in

- Click the left mouse button somewhere in the black portion displayed in the grid on the user trace display page.
- Press the Zoom In button repeatedly until two black vertical lines with a green bar between them appears.

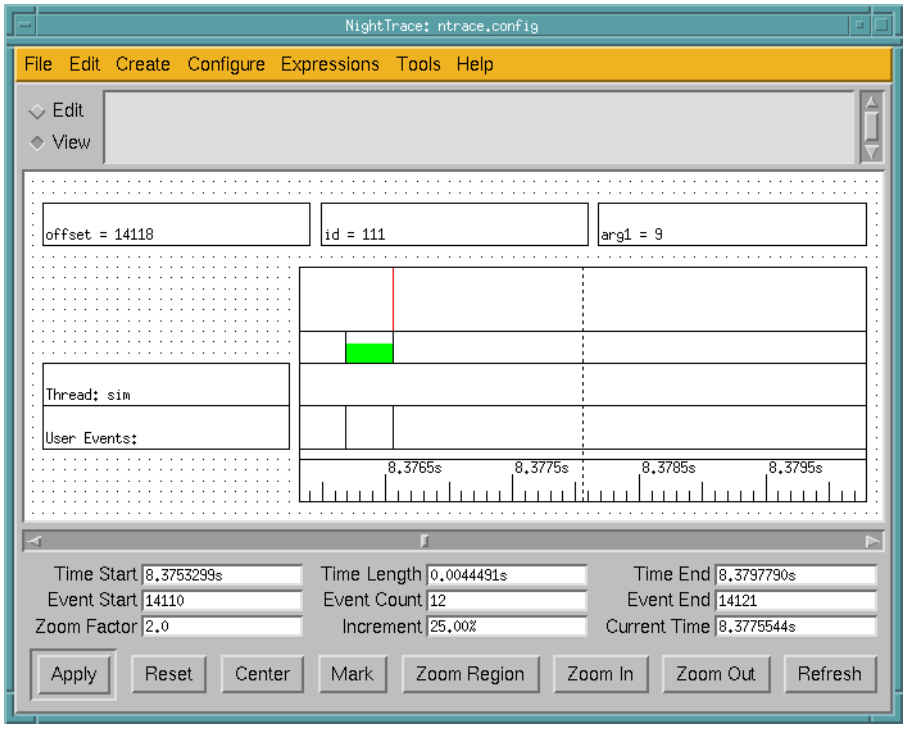

Figure 1-17. Zoomed in view of user trace data

- Click the left mouse button in the middle of the green bar.
- Press the Zoom In button repeatedly until no other lines or bars are visible on the grid except for those associated with the start and stop of the state represented by the green bar.

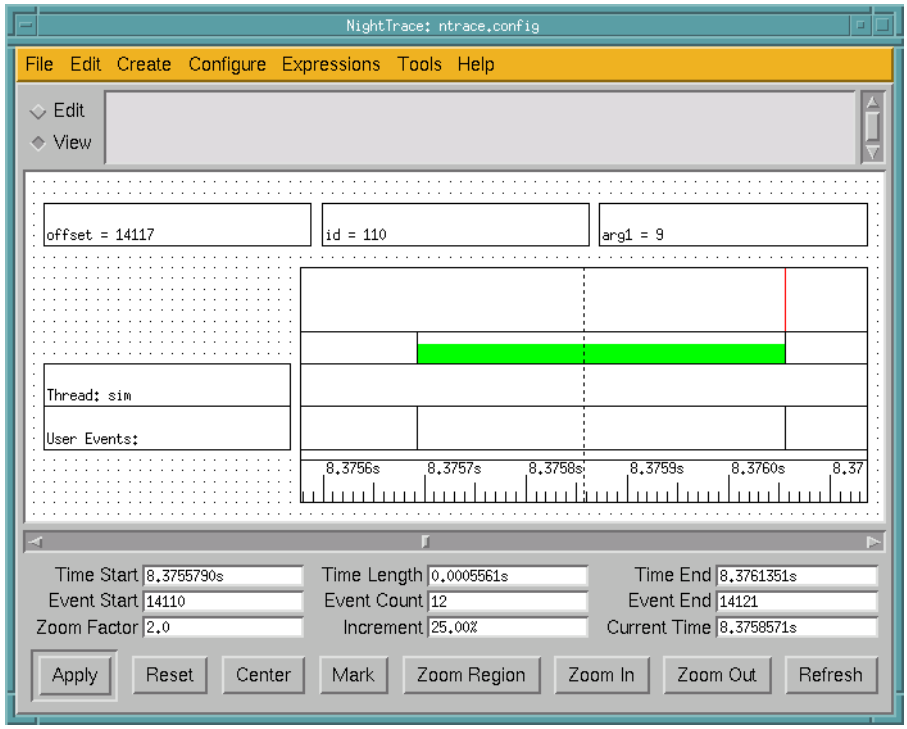

Figure 1-18. Extreme zoomed in view of user trace data

Remember that our program logs a trace event immediately when we start our cycle (exiting `fbswait()`), then it performs some calculations using `counters.Work()`, and finally it logs another trace event when it is finished before returning to the `fbswait()` call at the top of the loop. (See "Building the program" on page 1-3 to see that code fragment.)

The black lines represent the individual events logged in the application by the `trace_event_arg()` API calls. The green bar is a state graph; the start of the state is defined to be the `cycle_start` event logged when we begin our cycle (event #110) and the end of the state is defined by `cycle_end` (event #111) which is logged when we complete our cycle.

The red line that appears at the end of the state graph is an entry in a datagraph whose value is that of the argument logged with the `cycle_end` event in the second `trace_event_arg()` call. This value which ranges from 0 to 9.

NOTE

If no red line appears at the end of the state graph, the value logged in this instance was most likely zero which will not display. Zoom out, choose another nearby occurrence, and zoom back in.

Examining the kernel trace data

Now let's take a look at the kernel trace data to see how it coincides with the user trace data.

NOTE

NightTrace automatically synchronizes all display pages so that every display page shows the same time frame. Thus, our kernel display page reflects the system activity corresponding to the time period displayed in our user trace display page.

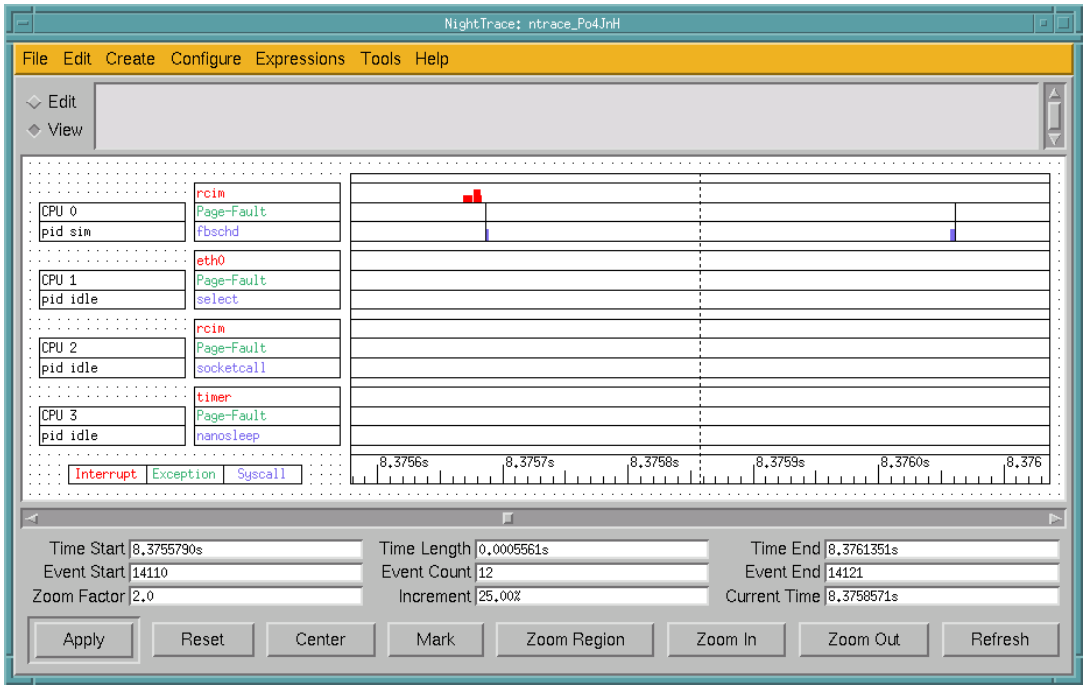

Figure 1-19. Zoomed in view of kernel display page

The first red bar displayed on the grid for CPU 0 indicates the interrupt from the RCIM device (there are actually nested interrupts as shown by the different levels of red bars).

A context-switch then occurs as indicated by the first black vertical line. The blue bar following that first black line is the `fbsched` system call. In our source code, this is when we exit the `fbswait()` call.

The application then performs its calculations (as indicated by the expanse of white space) before it comes back to the `fbswait()` call (the second blue bar).

The lack of any activity in the white space indicates that the user application did not make any intervening system calls, received no machine exceptions, and was not disturbed by some other interrupt.

NOTE

Since this is live data, it is possible that you may see additional activity, most likely red interrupt activity, between the exit and reentry to `fbswait()`.

In a real-life scenario, we would tune and shield the system for optimal real-time performance.

Exiting the tools

In conclusion of our tutorial, we will exit each of the tools.

Exiting NightTrace

To exit NightTrace

- From the NightTrace main window, select Exit from the NightTrace menu.
- When NightTrace presents the warning dialog asking if you would like to save changes to the new session, press No.

Exiting NightProbe

To exit NightProbe

- From the NightProbe Data Recording window, press the Stop button to stop sampling data.
- Press the Disconnect button to disconnect from the application.
- From the File menu, select Exit.

Exiting NightSim

To exit NightSim

- In the NightSim Scheduler window, press the Stop button.
- Press the Remove button.

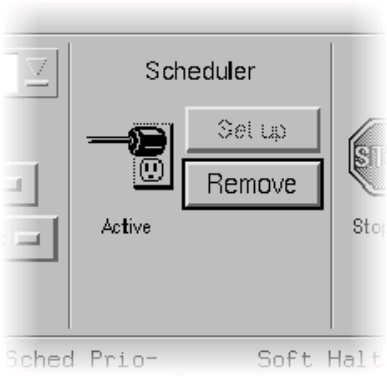

Figure 1-20. Removing the scheduler

You will be presented with the following dialog:

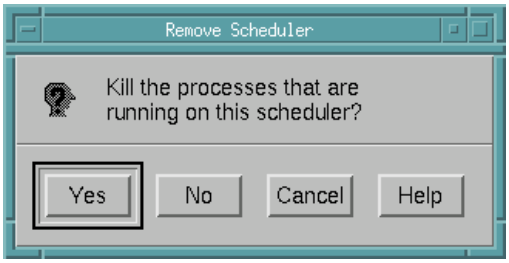

Figure 1-21. Removing the scheduler

- Press Yes to kill the processes that are currently scheduled on the scheduler.
- From the NightSim menu, select Exit.
- When NightSim presents the warning dialog asking if you would like to save the current configuration, press No.

Conclusion

This concludes our tutorial for the RedHawk NightStar Tools.

Spine for 1/2” Binder

Product Name: 0.5" from top of spine, Helvetica, 36 pt, Bold

Volume Number (if any): Helvetica, 24 pt, Bold

Volume Name (if any): Helvetica, 18 pt, Bold

Manual Title(s): Helvetica, 10 pt, Bold, centered vertically within space above bar, double space between each title

Bar: 1" x 1/8" beginning 1/4" in from either side

Part Number: Helvetica, 6 pt, centered, 1/8" up