# How To Find The Cause Of A Broken PREEMPT_RT System

What tools and approaches are useful for investigating and understanding real-time Linux performance? This talk presents tools for investigating and debugging, the type of information the various tools provide, and where to find the tools and their documentation. The tools provide the information that you will need to investigate and solve your own problems, or that the community will request from you if you ask for assistance. This is not a detailed tutorial of command options, but instead provides an investigation and analysis framework.

Frank Rowand, Sony Network Entertainment

June 2, 2011

110601_2242

# How Do You Know There Is A Problem?

| | |
|---|---|
| - After Development | (reactive) |
| - During Development | (reactive)<br>(proactive) |
| - During Design and Specification | (proactive) |

# Detection Tools:

## After Development

- Obvious RT application failure can be detected by the application user or an observer.

# RT Application Fails Totally

(CC BY-NC-SA 2.0) http://www.flickr.com/photos/mattbraga/4044016187/

# RT Application Fails Totally

Catastrophic failure due to external causes is not covered by this presentation

(CC BY-NC-SA 2.0) http://www.flickr.com/photos/mattbraga/4044016187/

# RT Application Fails To Bad State

Example: autonomous vehicle may drift out of lane

http://www.flickr.com/photos/chrismetcalf/49282052/ (CC BY-NC-SA 3.0)

# RT Application Fails To Bad State

Example: autonomous vehicle may drift out of lane

http://www.flickr.com/photos/kwc/1866546241 (CC BY-NC 2.0)

# In the interest of truth

I do not know the actual causes of the situations that are shown in the previous autonomous vehicle photos.  The actual causes are probably totally unrelated to real-time Linux.

# RT Application Misbehaves

Examples:

Dropped Video Frames

- Movie take must be repeated
- Glitch may be fixed by CGI or editing

# RT Application Misbehaves

Examples:

Audio Dropouts

- Recording session must be repeated
- Live performance is marred
- Bad device reputation, resulting in poor product sales

# RT Application Misbehaves

Examples:

Control response is inconsistent, resulting in poor controller feel

- Video game controller difficult to use

- Sloppy device control

- Operator becomes fatigued

---> Fatigued pilot crashes plane

# Detection Tools:

## After Development

- Obvious RT application failure that can be observed by the application user or bystander
- Detection by the same tools as used during development

# Detection Tools:

## During Development

- Obvious RT application failure that can be observed by the application user or bystander
- Measured metric is out of range

# What Metrics Can Be Measured To Define A Problem?

- Latency

  Delay from event until reacting to event.

- Jitter

  Variation in latency.

- Throughput

  Rate at which events can be handled.

# Instrument The RT Application

- The most accurate and complete data
- Measures latency
- Measures jitter
- Measures throughput

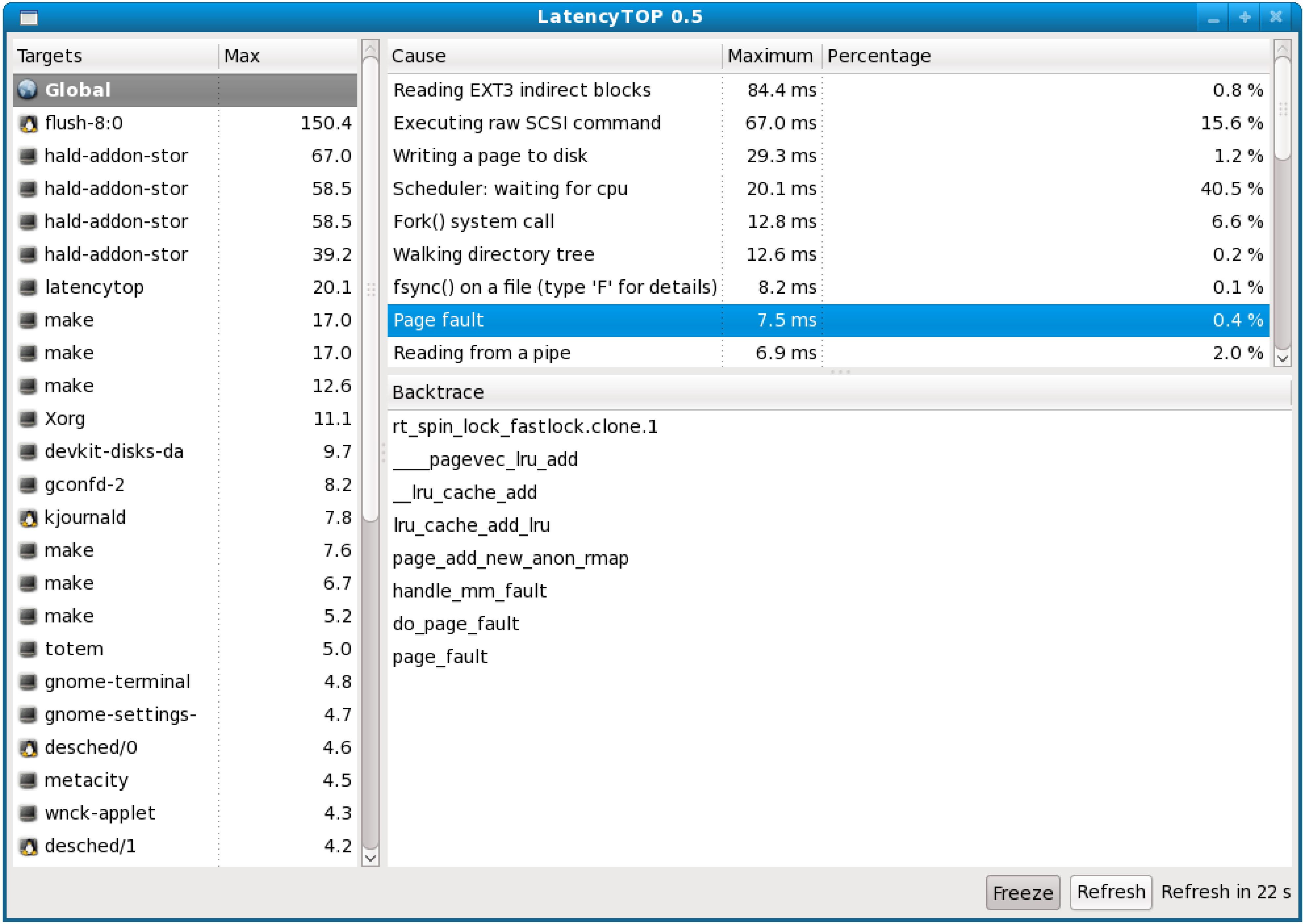
# Instrument The RT Application

## Difficult to achieve

- No toolkit, infrastructure, or API available
- Challenge of getting accurate event time stamp

# LatencyTOP

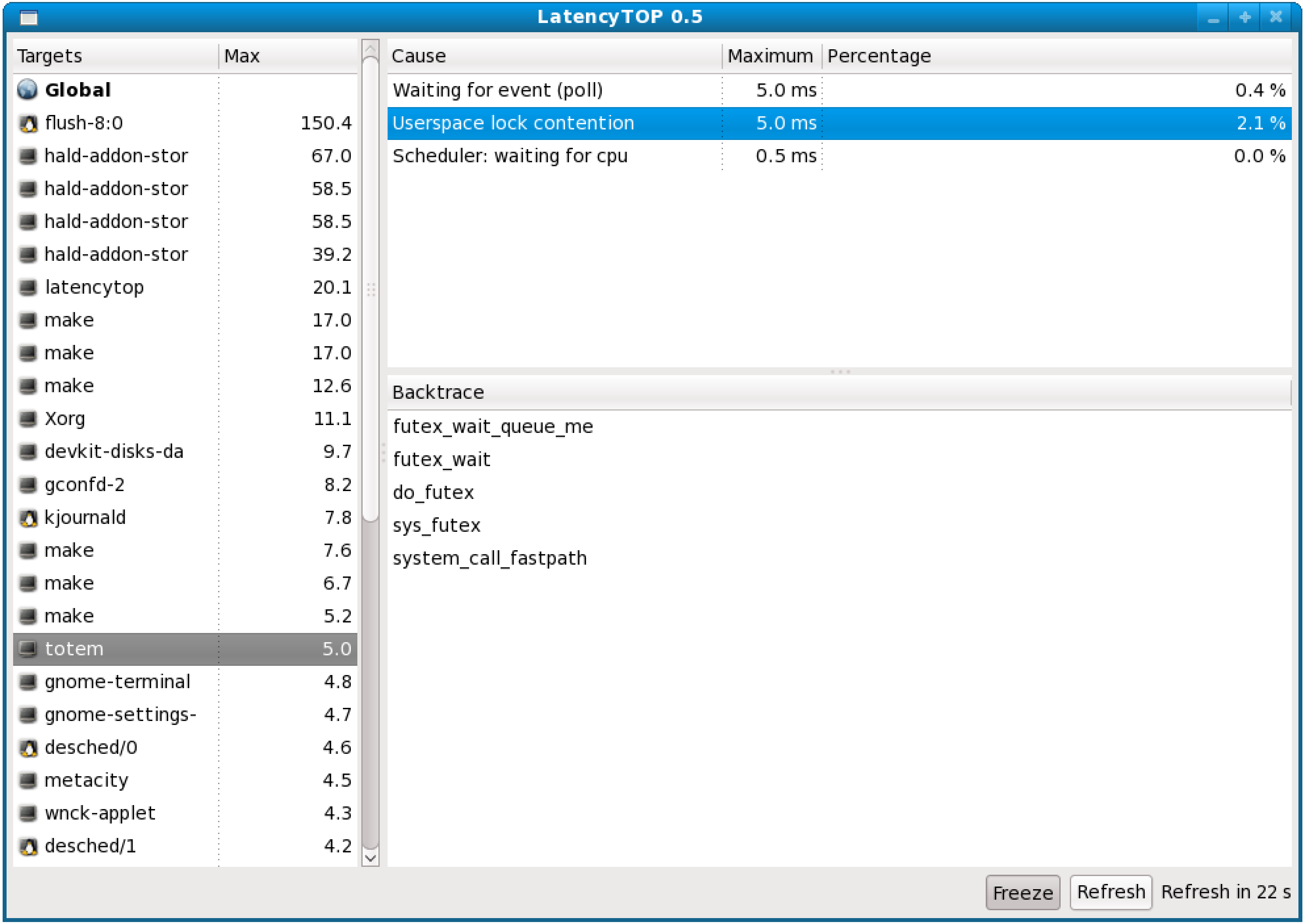
- Measures time waiting, not latency
- Does not measure jitter
- Does not measure throughput
- Stack trace may provide some hints of problem cause



| Targets | Max |
|---|---|
| Global | |
| flush-8:0 | 150.4 |
| hald-addon-stor | 67.0 |
| hald-addon-stor | 58.5 |
| hald-addon-stor | 58.5 |
| hald-addon-stor | 39.2 |
| latencytop | 20.1 |
| make | 17.0 |
| make | 17.0 |
| make | 12.6 |
| Xorg | 11.1 |
| devkit-disks-da | 9.7 |
| gconfd-2 | 8.2 |
| kjournald | 7.8 |
| make | 7.6 |
| make | 6.7 |
| make | 5.2 |
| totem | 5.0 |
| gnome-terminal | 4.8 |
| gnome-settings- | 4.7 |
| desched/0 | 4.6 |
| metacity | 4.5 |
| wnck-applet | 4.3 |
| desched/1 | 4.2 |

| Cause | Maximum | Percentage |
|---|---|---|
| Reading EXT3 indirect blocks | 84.4 ms | 0.8 % |
| Executing raw SCSI command | 67.0 ms | 15.6 % |
| Writing a page to disk | 29.3 ms | 1.2 % |
| Scheduler: waiting for cpu | 20.1 ms | 40.5 % |
| Fork() system call | 12.8 ms | 6.6 % |
| Walking directory tree | 12.6 ms | 0.2 % |
| fsync() on a file (type 'F' for details) | 8.2 ms | 0.1 % |
| Page fault | 7.5 ms | 0.4 % |
| Reading from a pipe | 6.9 ms | 2.0 % |

```
Backtrace
rt_spin_lock_fastlock.clone.1
____pagevec_lru_add
__lru_cache_add
lru_cache_add_lru
page_add_new_anon_rmap
handle_mm_fault
do_page_fault
page_fault
```



| Targets | Max |
|---|---|
| **Global** | |
| flush-8:0 | 150.4 |
| hald-addon-stor | 67.0 |
| hald-addon-stor | 58.5 |
| hald-addon-stor | 58.5 |
| hald-addon-stor | 39.2 |
| latencytop | 20.1 |
| make | 17.0 |
| make | 17.0 |
| make | 12.6 |
| Xorg | 11.1 |
| devkit-disks-da | 9.7 |
| gconfd-2 | 8.2 |
| kjournald | 7.8 |
| make | 7.6 |
| make | 6.7 |
| make | 5.2 |
| totem | 5.0 |
| gnome-terminal | 4.8 |
| gnome-settings- | 4.7 |
| desched/0 | 4.6 |
| metacity | 4.5 |
| wnck-applet | 4.3 |
| desched/1 | 4.2 |

| Cause | Maximum | Percentage |
|---|---|---|
| Waiting for event (poll) | 5.0 ms | 0.4 % |
| Userspace lock contention | 5.0 ms | 2.1 % |
| Scheduler: waiting for cpu | 0.5 ms | 0.0 % |

| Backtrace |
|---|
| futex_wait_queue_me |
| futex_wait |
| do_futex |
| sys_futex |
| system_call_fastpath |

# LatencyTOP Limitations

- Interruptible sleep > 5 msec not reported
- A real time task should not be sleeping unless it is waiting for a stimulus
- Only measures SCHED_OTHER tasks
- Temporarily changing a SCHED_FIFO process to SCHED_OTHER may provide some insights into why the process is unexpectedly sleeping (but the magnitude of the sleep may not be representative of the SCHED_FIFO sleep duration)

# Cyclictest

- Measures latency
- Measures jitter
- Does not measure throughput

# How Cyclictest Measures

In one or more threads:

```
clock_gettime(&now)
while (not done)
   next = now + interval
   sleep(interval)
   clock_gettime(&now)
   latency = now - next
```

# What Latency Includes

IRQ overhead

Scheduler overhead

Latency causes:

- IRQs disabled
- Preemption disabled
- IRQ handlers running in IRQ context
- Priority inversion
- Lock contention
- SMI
- Cache issues
- Higher priority threads
- etc

# Cyclictest Example

## (latency)

```
$ cyclictest -q -n -t1 -p 48 -i 10000 -l 10000
T: 0 (11263) P:48 I:10000 C:  10000 Min:     18 Act:   21 Avg:   19 Max:      98
```

# Cyclictest Example

## (latency)

```
$ cyclictest -q -n -t1 -p 51 -i 10000 -l 10000
T: 0 (11263) P:48 I:10000 C:  10000 Min:     18 Act:   21 Avg:   19 Max:      98
```

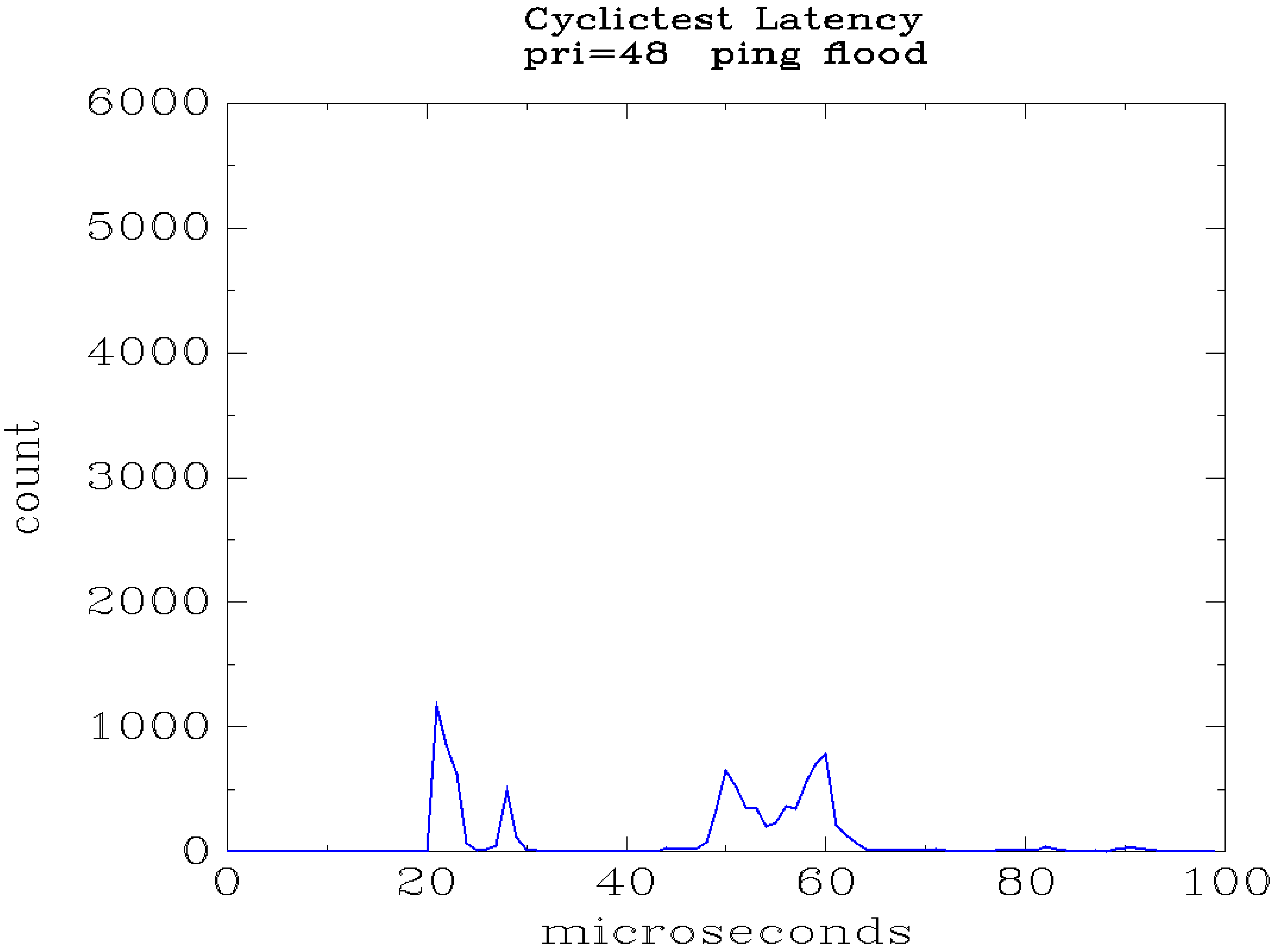
```
$ cyclictest -q -n -t1 -p 48 -i 10000 -l 10000

T: 0 (11263) P:48 I:10000 C:  10000

Min:     18 Act:   21 Avg:   19 Max:      98
```

Maximum latency of 98 looks like a problem

# Collect More Detailed Data

(latency, jitter)

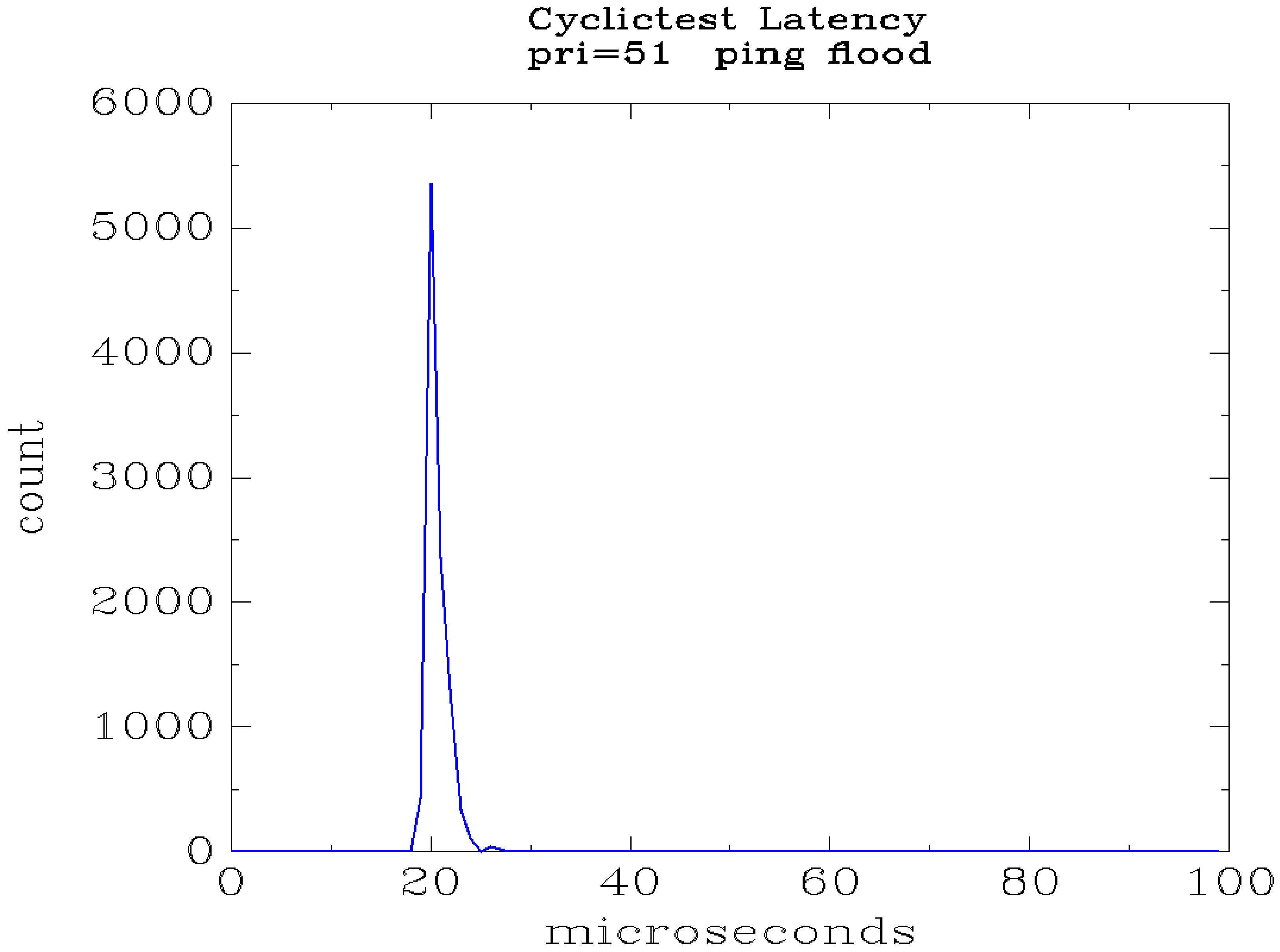
```
$ cyclictest -q -n -t1 -p 48 -i 10000 -l 10000 \
             --histogram=100
```

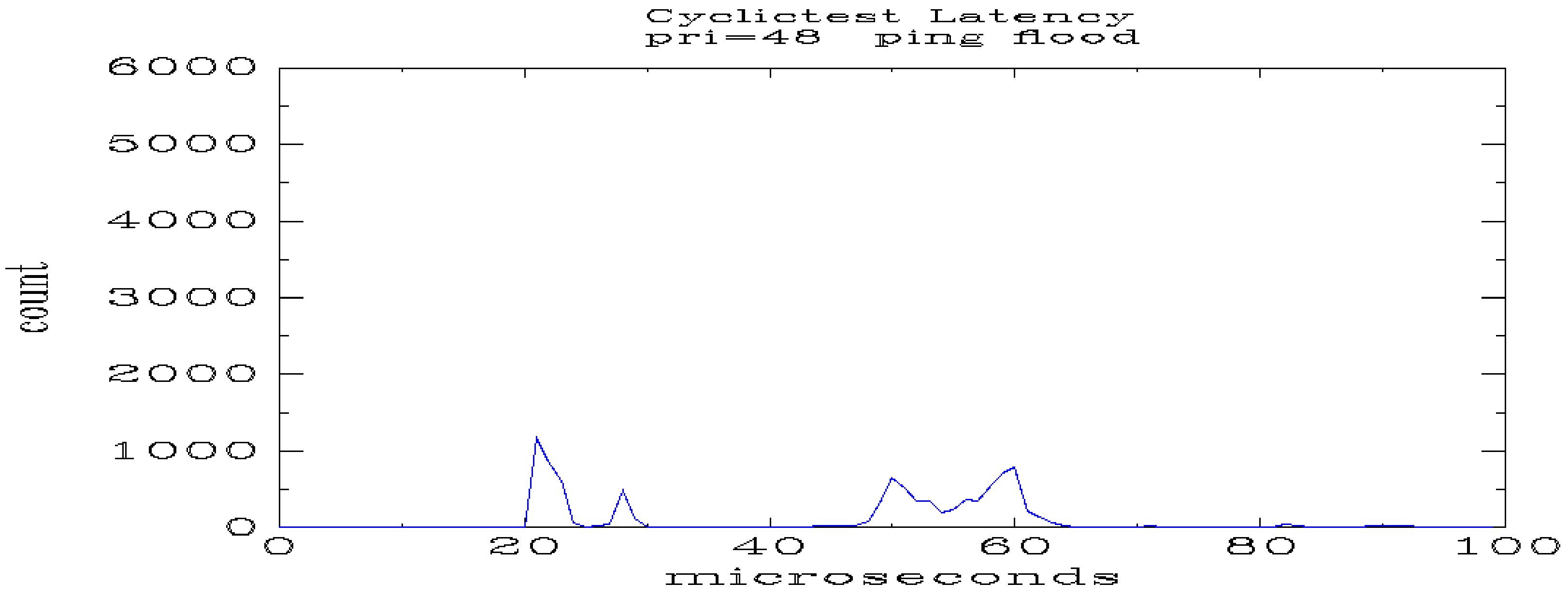

Cyclictest Latency
pri=48  ping flood
6000
5000
4000
3000
2000
1000
0
count
0
20
40
60
80
100
microseconds

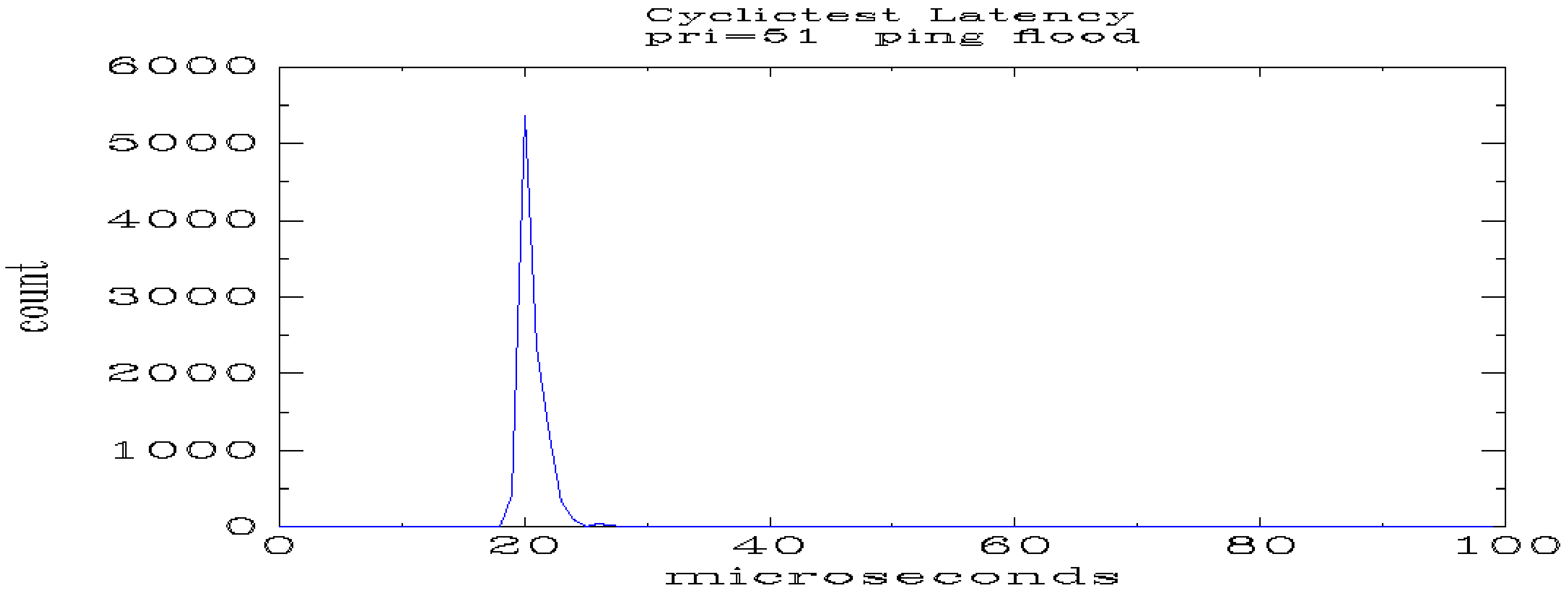
# Test if problem is related to priority

```
  PID  PPID S RTPRIO CLS CMD
    6     2 S     49 FF  [sirq-net-tx/0]
    7     2 S     49 FF  [sirq-net-rx/0]
   12     2 S     49 FF  [sirq-hrtimer/0]
   13     2 S     49 FF  [sirq-rcu/0]
 1775     2 S     50 FF  [irq/20-eth0]
```

```
$ cyclictest -q -n -t1 -p 48 -i 10000 -l 10000 \
             --histogram=100

$ cyclictest -q -n -t1 -p 51 -i 10000 -l 10000 \
             --histogram=100
```


Cyclictest Latency
pri=51 ping flood
6000
5000
4000
3000
2000
1000
0
count
0
20
40
60
80
100
microseconds

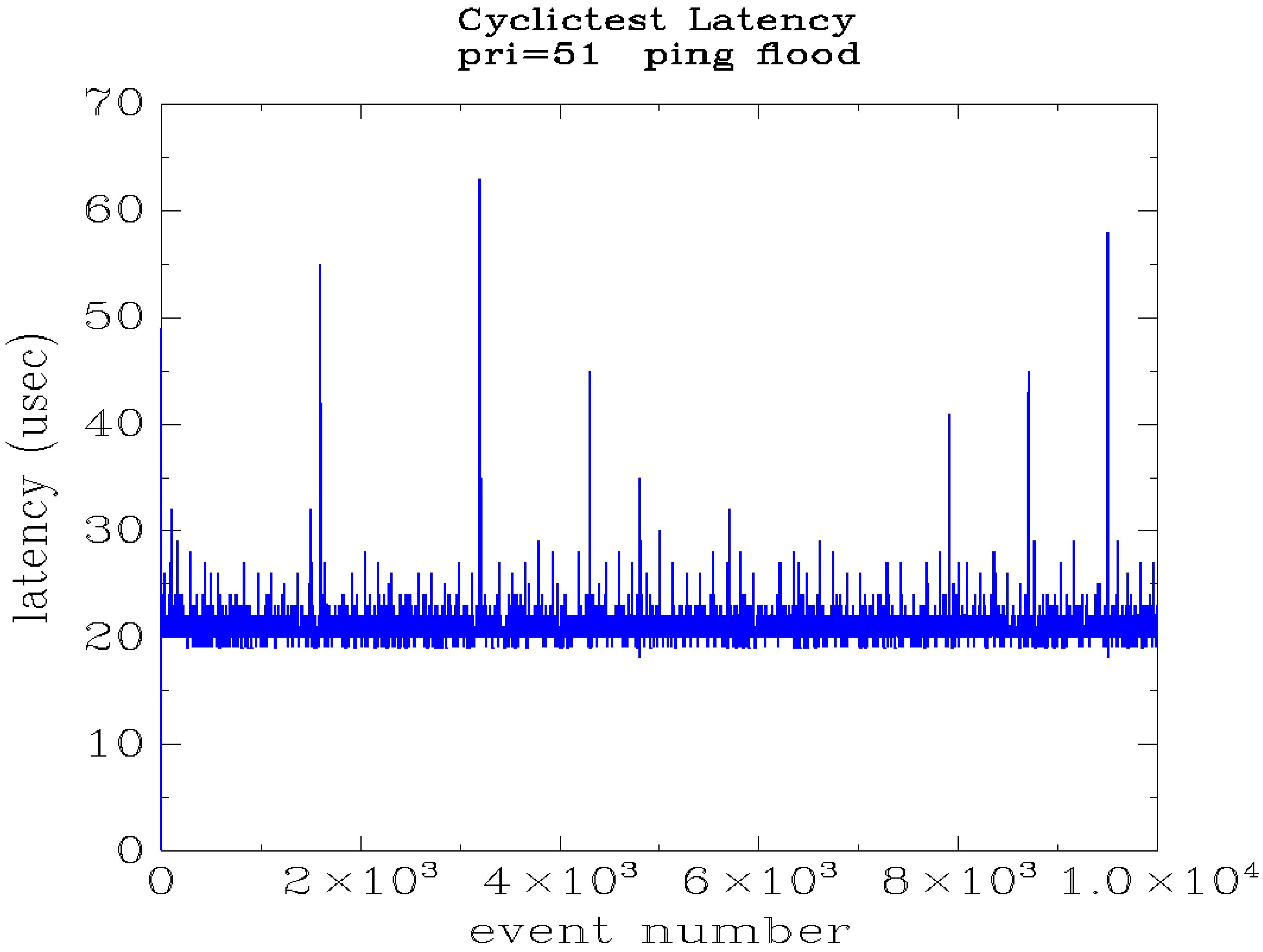

Cyclictest Latency
pri=48 ping flood
count
0
1000
2000
3000
4000
5000
6000
0
20
40
60
80
100
microseconds


Cyclictest Latency
pri=51 ping flood
count
0
1000
2000
3000
4000
5000
6000
0
20
40
60
80
100
microseconds

# Collect Even More Detailed Data

(jitter)

```
$ cyclictest -q -n -t1 -p 48 -i 10000 -l 10000 \
             -v
```


Cyclictest Latency
pri=51 ping flood
latency (usec)
70
60
50
40
30
20
10
0
0
2×10³
4×10³
6×10³
8×10³
1.0×10⁴
event number

# Cyclictest: Cause Analysis

cyclictest has options to control tracers

- IRQs off
- Preempt off
- context switch
- task wakeup
- events
- ftrace

# Cyclictest Limitations

- Event occurrence on a timed cadence
- Only event measured: timer

  Example of more complex RT application, components to handle an event :

  * network device IRQ context
  * network device IRQ thread
  * network softirq thread
  * RT application
- RT application execution time not measured

# Other Timer Based Latency Tools

realfeel

- uses /dev/rtc

  http://brain.mcmaster.ca/~hahn/realfeel.c

rf-etri aka. realfeel-etri (greatly enhanced)

- uses /dev/rtc
- promising, but I have not found the project page
- Described by

  http://elinux.org/images/8/8e/
  Real-time_Measurement-ELC2010-final.pdf

# Hardware Latency Measurement

LRTB

- Serial port signaling between two machines
- LRTBF 0.3 released 2005 / 07 / 08

# Hardware Latency Measurement

lpptest

- Parallel port signaling between two machines

- In RT kernel source tree:
  drivers/char/lpptest.c
  scripts/testlpp.c
  (from latency-measurement-drivers.patch)

- 2.6.26.8-rt16   existed

- 2.6.29-rc6-rt2  no longer existed

# Hardware Latency Measurement

Woerner test

- Serial port signaling between two machines
- Web site no longer exists

  Does anyone have an archived version to share with me?

  http://geek.vtnet.ca/web/latencytests-3.2.1.tar.bz2

# Measurement and Analysis Tools

Some examples

# perf sched latency

(truncated on right)

```
---------------------------------------------------------------------------------------
 Task                  |   Runtime ms  | Switches | Average delay ms | Maximum delay ms
---------------------------------------------------------------------------------------
 perf:14632            |    235.239 ms |      306 | avg:    0.330 ms | max:    1.071 ms
 Xorg:1814             |   1162.011 ms |     2508 | avg:    0.078 ms | max:    1.051 ms
 sirq-rcu/0:13         |      0.000 ms |      299 | avg:    0.053 ms | max:    0.149 ms
 cyclictest:14630      |      3.004 ms |      161 | avg:    0.051 ms | max:    0.226 ms
 sirq-timer/0:5        |      0.000 ms |     1590 | avg:    0.035 ms | max:    0.153 ms
 sirq-hrtimer/0:12     |      0.000 ms |        2 | avg:    0.034 ms | max:    0.049 ms
 sirq-sched/0:11       |      0.000 ms |       28 | avg:    0.031 ms | max:    0.089 ms
 gnome-panel:2662      |      0.413 ms |        1 | avg:    0.030 ms | max:    0.030 ms
 kondemand/0:1224      |      0.403 ms |       14 | avg:    0.028 ms | max:    0.117 ms
 sirq-rcu/1:28         |      0.000 ms |      268 | avg:    0.025 ms | max:    0.041 ms
 gpm:1678              |      0.148 ms |        2 | avg:    0.024 ms | max:    0.032 ms
 events/1:32           |      0.000 ms |        2 | avg:    0.022 ms | max:    0.027 ms
 sirq-sched/1:26       |      0.000 ms |       38 | avg:    0.018 ms | max:    0.026 ms
 sirq-net-rx/0:7       |      0.000 ms |     7261 | avg:    0.016 ms | max:    0.065 ms
 sirq-timer/1:20       |      0.000 ms |     1348 | avg:    0.015 ms | max:    0.038 ms
 events/0:31           |      0.000 ms |        1 | avg:    0.014 ms | max:    0.014 ms
 :14631:14631          |      0.000 ms |      135 | avg:    0.012 ms | max:    0.016 ms
 sirq-hrtimer/1:27     |      0.000 ms |        1 | avg:    0.010 ms | max:    0.010 ms
 irq/20-eth0:1775      |      0.000 ms |     9768 | avg:    0.009 ms | max:    0.050 ms
---------------------------------------------------------------------------------------
 TOTAL:                |   1401.218 ms |    23733 |
---------------------------------------------------
```

# perf sched latency

## (full output)

```
-----------------------------------------------------------------------------------------------------------------
 Task                  |   Runtime ms  | Switches | Average delay ms | Maximum delay ms | Maximum delay at     |
-----------------------------------------------------------------------------------------------------------------
 perf:14632            |    235.239 ms |      306 | avg:    0.330 ms | max:    1.071 ms | max at: 189246.857372 s
 Xorg:1814             |   1162.011 ms |     2508 | avg:    0.078 ms | max:    1.051 ms | max at: 189246.782344 s
 sirq-rcu/0:13         |      0.000 ms |      299 | avg:    0.053 ms | max:    0.149 ms | max at: 189246.821176 s
 cyclictest:14630      |      3.004 ms |      161 | avg:    0.051 ms | max:    0.226 ms | max at: 189247.686091 s
 sirq-timer/0:5        |      0.000 ms |     1590 | avg:    0.035 ms | max:    0.153 ms | max at: 189247.987173 s
 sirq-hrtimer/0:12     |      0.000 ms |        2 | avg:    0.034 ms | max:    0.049 ms | max at: 189247.352181 s
 sirq-sched/0:11       |      0.000 ms |       28 | avg:    0.031 ms | max:    0.089 ms | max at: 189247.528107 s
 gnome-panel:2662      |      0.413 ms |        1 | avg:    0.030 ms | max:    0.030 ms | max at: 189247.322329 s
 kondemand/0:1224      |      0.403 ms |       14 | avg:    0.028 ms | max:    0.117 ms | max at: 189247.938179 s
 sirq-rcu/1:28         |      0.000 ms |      268 | avg:    0.025 ms | max:    0.041 ms | max at: 189246.998337 s
 gpm:1678              |      0.148 ms |        2 | avg:    0.024 ms | max:    0.032 ms | max at: 189246.947328 s
 events/1:32           |      0.000 ms |        2 | avg:    0.022 ms | max:    0.027 ms | max at: 189246.857339 s
 sirq-sched/1:26       |      0.000 ms |       38 | avg:    0.018 ms | max:    0.026 ms | max at: 189246.926305 s
 sirq-net-rx/0:7       |      0.000 ms |     7261 | avg:    0.016 ms | max:    0.065 ms | max at: 189246.920129 s
 sirq-timer/1:20       |      0.000 ms |     1348 | avg:    0.015 ms | max:    0.038 ms | max at: 189247.129316 s
 events/0:31           |      0.000 ms |        1 | avg:    0.014 ms | max:    0.014 ms | max at: 189247.704380 s
 :14631:14631          |      0.000 ms |      135 | avg:    0.012 ms | max:    0.016 ms | max at: 189247.129366 s
 sirq-hrtimer/1:27     |      0.000 ms |        1 | avg:    0.010 ms | max:    0.010 ms | max at: 189247.349146 s
 irq/20-eth0:1775      |      0.000 ms |     9768 | avg:    0.009 ms | max:    0.050 ms | max at: 189247.209038 s
-----------------------------------------------------------------------------------------
 TOTAL:                |   1401.218 ms |    23733 |
---------------------------------------------------
```

# perf PMU and event statistics

Example command:

```
perf stat -i -a \
   -e cycles -e instructions \
   -e cache-misses -e L1-dcache-load-misses \
   -e lock:lock_acquire -e lock:lock_contended \
   -e sched:sched_wakeup -e sched:sched_switch \
   -e sched:sched_migrate_task \
   cyclictest -q -n -t1 -p 51 -i 10000 -l 10000
```

# perf PMU and event statistics

```
Performance counter stats for
'cyclictest -q -n -t1 -p 51 -i 10000 -l 10000':

 353191943897  cycles
 216645413360  instructions            #  0.613 IPC
   1248311849  cache-misses
   1900719803  L1-dcache-load-misses
     57795570  lock:lock_acquire
         5494  lock:lock_contended
      1095924  sched:sched_wakeup
      1866984  sched:sched_switch
          747  sched:sched_migrate_task

101.508863128  seconds time elapsed
```

# perf trace

```
  irq/20-eth0-1775  [000] 189611.397897: sched_wakeup: comm=sirq-net-rx/0 pid=7 prio=50 su
  irq/20-eth0-1775  [000] 189611.397911: sched_switch: prev_comm=irq/20-eth0 prev_pid=1775
         Xorg-1814  [001] 189611.397919: sched_stat_runtime: comm=Xorg pid=1814 runtime=59
         Xorg-1814  [001] 189611.397924: sched_stat_sleep: comm=cyclictest pid=14720 delay
         Xorg-1814  [001] 189611.397928: sched_wakeup: comm=cyclictest pid=14720 prio=120
         Xorg-1814  [001] 189611.397936: sched_stat_runtime: comm=Xorg pid=1814 runtime=16
         Xorg-1814  [001] 189611.397940: sched_stat_wait: comm=cyclictest pid=14720 delay=
         Xorg-1814  [001] 189611.397944: sched_switch: prev_comm=Xorg prev_pid=1814 prev_p
   cyclictest-14720 [001] 189611.397956: sched_stat_runtime: comm=cyclictest pid=14720 run
   cyclictest-14720 [001] 189611.397960: sched_stat_wait: comm=Xorg pid=1814 delay=20222
   cyclictest-14720 [001] 189611.397964: sched_switch: prev_comm=cyclictest prev_pid=14720
sirq-net-rx/0-7     [000] 189611.397972: sched_stat_wait: comm=perf pid=14722 delay=89264
sirq-net-rx/0-7     [000] 189611.397975: sched_switch: prev_comm=sirq-net-rx/0 prev_pid=7
         perf-14722 [000] 189611.397990: sched_wakeup: comm=irq/20-eth0 pid=1775 prio=49 s
         perf-14722 [000] 189611.397996: sched_stat_runtime: comm=perf pid=14722 runtime=2
         perf-14722 [000] 189611.398001: sched_switch: prev_comm=perf prev_pid=14722 prev_
  irq/20-eth0-1775  [000] 189611.398024: sched_wakeup: comm=sirq-timer/0 pid=5 prio=50 suc
  irq/20-eth0-1775  [000] 189611.398030: sched_wakeup: comm=sirq-rcu/0 pid=13 prio=50 succ
  irq/20-eth0-1775  [000] 189611.398053: sched_switch: prev_comm=irq/20-eth0 prev_pid=1775
 sirq-timer/0-5     [000] 189611.398063: sched_switch: prev_comm=sirq-timer/0 prev_pid=5 p
```

# perf trace

```
  irq/20-eth0-1775  [000] 189611.397897: sched_wakeup: comm=sirq-net-rx/0 pid=7 prio=50 success=1 target_cpu=000
  irq/20-eth0-1775  [000] 189611.397911: sched_switch: prev_comm=irq/20-eth0 prev_pid=1775 prev_prio=49 prev_state=S ==> next_comm=sirq-net-rx/0 next_pid=7 next_prio=50
         Xorg-1814  [001] 189611.397919: sched_stat_runtime: comm=Xorg pid=1814 runtime=590249 [ns] vruntime=244455952143042 [ns]
         Xorg-1814  [001] 189611.397924: sched_stat_sleep: comm=cyclictest pid=14720 delay=10056591 [ns]
         Xorg-1814  [001] 189611.397928: sched_wakeup: comm=cyclictest pid=14720 prio=120 success=1 target_cpu=001
         Xorg-1814  [001] 189611.397936: sched_stat_runtime: comm=Xorg pid=1814 runtime=16993 [ns] vruntime=244455952160035 [ns]
         Xorg-1814  [001] 189611.397940: sched_stat_wait: comm=cyclictest pid=14720 delay=16993 [ns]
         Xorg-1814  [001] 189611.397944: sched_switch: prev_comm=Xorg prev_pid=1814 prev_prio=120 prev_state=R ==> next_comm=cyclictest next_pid=14720 next_prio=120
   cyclictest-14720 [001] 189611.397956: sched_stat_runtime: comm=cyclictest pid=14720 runtime=20222 [ns] vruntime=244455947163264 [ns]
   cyclictest-14720 [001] 189611.397960: sched_stat_wait: comm=Xorg pid=1814 delay=20222 [ns]
   cyclictest-14720 [001] 189611.397964: sched_switch: prev_comm=cyclictest prev_pid=14720 prev_prio=120 prev_state=S ==> next_comm=Xorg next_pid=1814 next_prio=120
sirq-net-rx/0-7     [000] 189611.397972: sched_stat_wait: comm=perf pid=14722 delay=89264 [ns]
sirq-net-rx/0-7     [000] 189611.397975: sched_switch: prev_comm=sirq-net-rx/0 prev_pid=7 prev_prio=50 prev_state=S ==> next_comm=perf next_pid=14722 next_prio=120
         perf-14722 [000] 189611.397990: sched_wakeup: comm=irq/20-eth0 pid=1775 prio=49 success=1 target_cpu=000
         perf-14722 [000] 189611.397996: sched_stat_runtime: comm=perf pid=14722 runtime=29154 [ns] vruntime=722540038994762 [ns]
         perf-14722 [000] 189611.398001: sched_switch: prev_comm=perf prev_pid=14722 prev_prio=120 prev_state=R ==> next_comm=irq/20-eth0 next_pid=1775 next_prio=49
  irq/20-eth0-1775  [000] 189611.398024: sched_wakeup: comm=sirq-timer/0 pid=5 prio=50 success=1 target_cpu=000
  irq/20-eth0-1775  [000] 189611.398030: sched_wakeup: comm=sirq-rcu/0 pid=13 prio=50 success=1 target_cpu=000
  irq/20-eth0-1775  [000] 189611.398053: sched_switch: prev_comm=irq/20-eth0 prev_pid=1775 prev_prio=49 prev_state=S ==> next_comm=sirq-timer/0 next_pid=5 next_prio=50
 sirq-timer/0-5     [000] 189611.398063: sched_switch: prev_comm=sirq-timer/0 prev_pid=5 prev_prio=50 prev_state=S ==> next_comm=sirq-rcu/0 next_pid=13 next_prio=50
```

# ftrace

- irqsoff / preempt off
- max latency trace

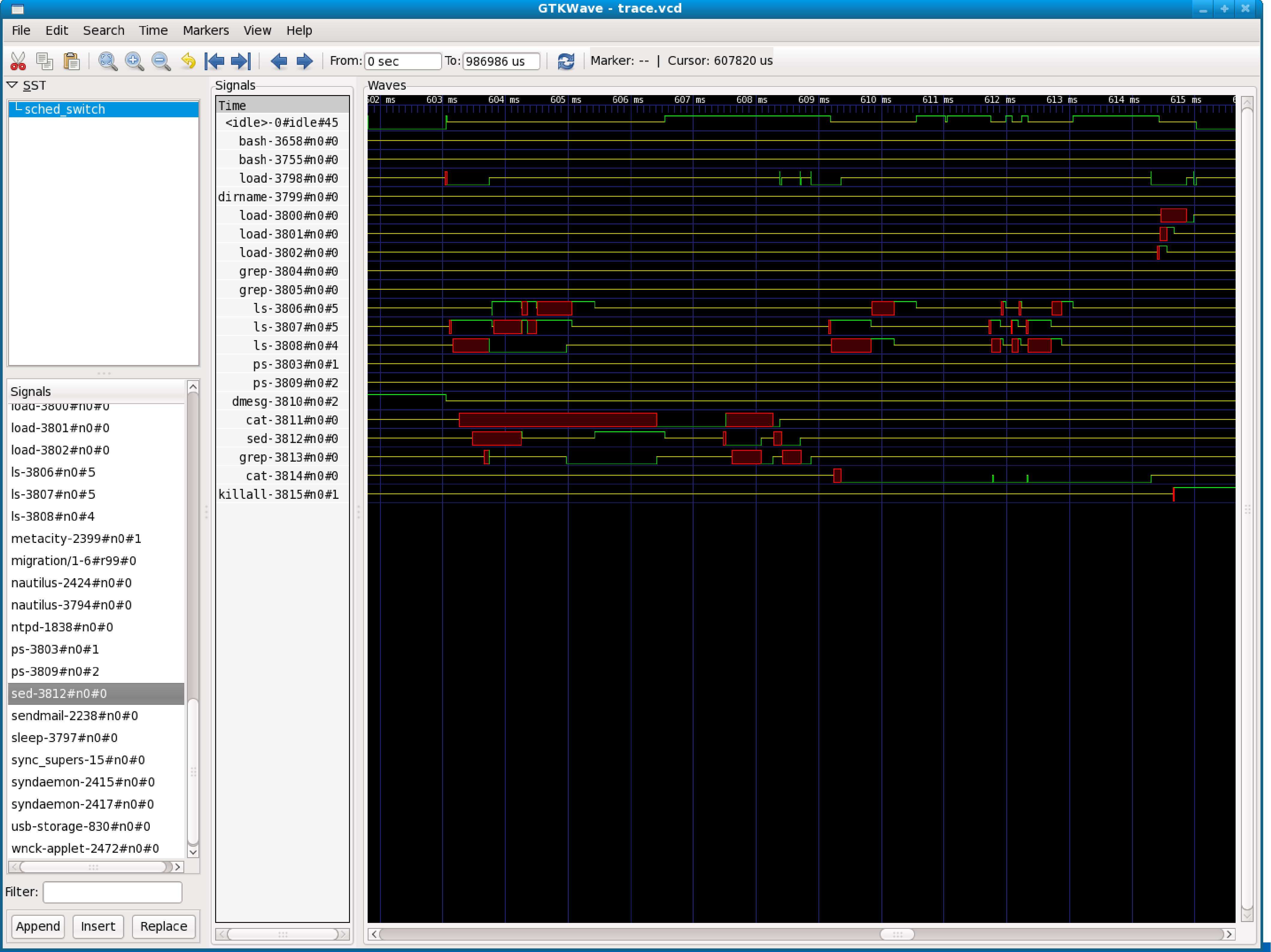
The next slides are an example report of the ftrace irqsoff tracer, with the latency-format option enabled, from Documentation/trace/ftrace.txt

```
 latency: 50 us, #101/101, CPU#0 | (M:preempt VP:0, KP:0, SP:0 HP:0 #P:2)
    -----------------
    | task: ls-4339 (uid:0 nice:0 policy:0 rt_prio:0)
    -----------------
 => started at: __alloc_pages_internal
 => ended at:   __alloc_pages_internal

#                _------=> CPU#
#               / _-----=> irqs-off
#              | / _----=> need-resched
#              || / _---=> hardirq/softirq
#              ||| / _--=> preempt-depth
#              |||| /
#              |||||     delay
#  cmd     pid ||||| time  |   caller
#     \   /    |||||   \   |   /
      ls-4339  0...1    0us+: get_page_from_freelist (__alloc_pages_internal)
      ls-4339  0d..1    3us : rmqueue_bulk (get_page_from_freelist)
      ls-4339  0d..1    3us : _spin_lock (rmqueue_bulk)
      ls-4339  0d..1    4us : add_preempt_count (_spin_lock)
      ls-4339  0d..2    4us : __rmqueue (rmqueue_bulk)
      ls-4339  0d..2    5us : __rmqueue_smallest (__rmqueue)
      ls-4339  0d..2    5us : __mod_zone_page_state (__rmqueue_smallest)
      ls-4339  0d..2    6us : __rmqueue (rmqueue_bulk)
      ls-4339  0d..2    6us : __rmqueue_smallest (__rmqueue)
      ls-4339  0d..2    7us : __mod_zone_page_state (__rmqueue_smallest)
      ls-4339  0d..2    7us : __rmqueue (rmqueue_bulk)
      ls-4339  0d..2    8us : __rmqueue_smallest (__rmqueue)
[...]
      ls-4339  0d..2   46us : __rmqueue_smallest (__rmqueue)
      ls-4339  0d..2   47us : __mod_zone_page_state (__rmqueue_smallest)
 latency: 50 us, #101/101, CPU#0 | (M:preempt VP:0, KP:0, SP:0 HP:0 #P:2)

#              |||||         delay
#  cmd     pid ||||| time  |   caller
#     \   /    |||||   \   |   /
      ls-4339  0...1    0us+: get_page_from_freelist (__alloc_pages_internal)
      ls-4339  0d..1    3us : rmqueue_bulk (get_page_from_freelist)
      ls-4339  0d..1    3us : _spin_lock (rmqueue_bulk)
      ls-4339  0d..1    4us : add_preempt_count (_spin_lock)
      ls-4339  0d..2    4us : __rmqueue (rmqueue_bulk)
      ls-4339  0d..2    5us : __rmqueue_smallest (__rmqueue)
      ls-4339  0d..2    5us : __mod_zone_page_state (__rmqueue_smallest)
      ls-4339  0d..2    6us : __rmqueue (rmqueue_bulk)
      ls-4339  0d..2    6us : __rmqueue_smallest (__rmqueue)
      ls-4339  0d..2    7us : __mod_zone_page_state (__rmqueue_smallest)
      ls-4339  0d..2    7us : __rmqueue (rmqueue_bulk)
      ls-4339  0d..2    8us : __rmqueue_smallest (__rmqueue)
[...]
      ls-4339  0d..2   46us : __rmqueue_smallest (__rmqueue)
      ls-4339  0d..2   47us : __mod_zone_page_state (__rmqueue_smallest)
      ls-4339  0d..2   47us : __rmqueue (rmqueue_bulk)
      ls-4339  0d..2   48us : __rmqueue_smallest (__rmqueue)
      ls-4339  0d..2   48us : __mod_zone_page_state (__rmqueue_smallest)
      ls-4339  0d..2   49us : _spin_unlock (rmqueue_bulk)
      ls-4339  0d..2   49us : sub_preempt_count (_spin_unlock)
      ls-4339  0d..1   50us : get_page_from_freelist (__alloc_pages_internal)
      ls-4339  0d..2   51us : trace_hardirqs_on (__alloc_pages_internal)
```

# Sometimes a GUI is nice...

# ftrace

- sched_switch



From: 0 sec To: 986986 us | Marker: -- | Cursor: 607820 us

SST: sched_switch

Signals:

```
load-3800#n0#0
load-3801#n0#0
load-3802#n0#0
ls-3806#n0#5
ls-3807#n0#5
ls-3808#n0#4
metacity-2399#n0#1
migration/1-6#r99#0
nautilus-2424#n0#0
nautilus-3794#n0#0
ntpd-1838#n0#0
ps-3803#n0#1
ps-3809#n0#2
sed-3812#n0#0
sendmail-2238#n0#0
sleep-3797#n0#0
sync_supers-15#n0#0
syndaemon-2415#n0#0
syndaemon-2417#n0#0
usb-storage-830#n0#0
wnck-applet-2472#n0#0
```

Filter:

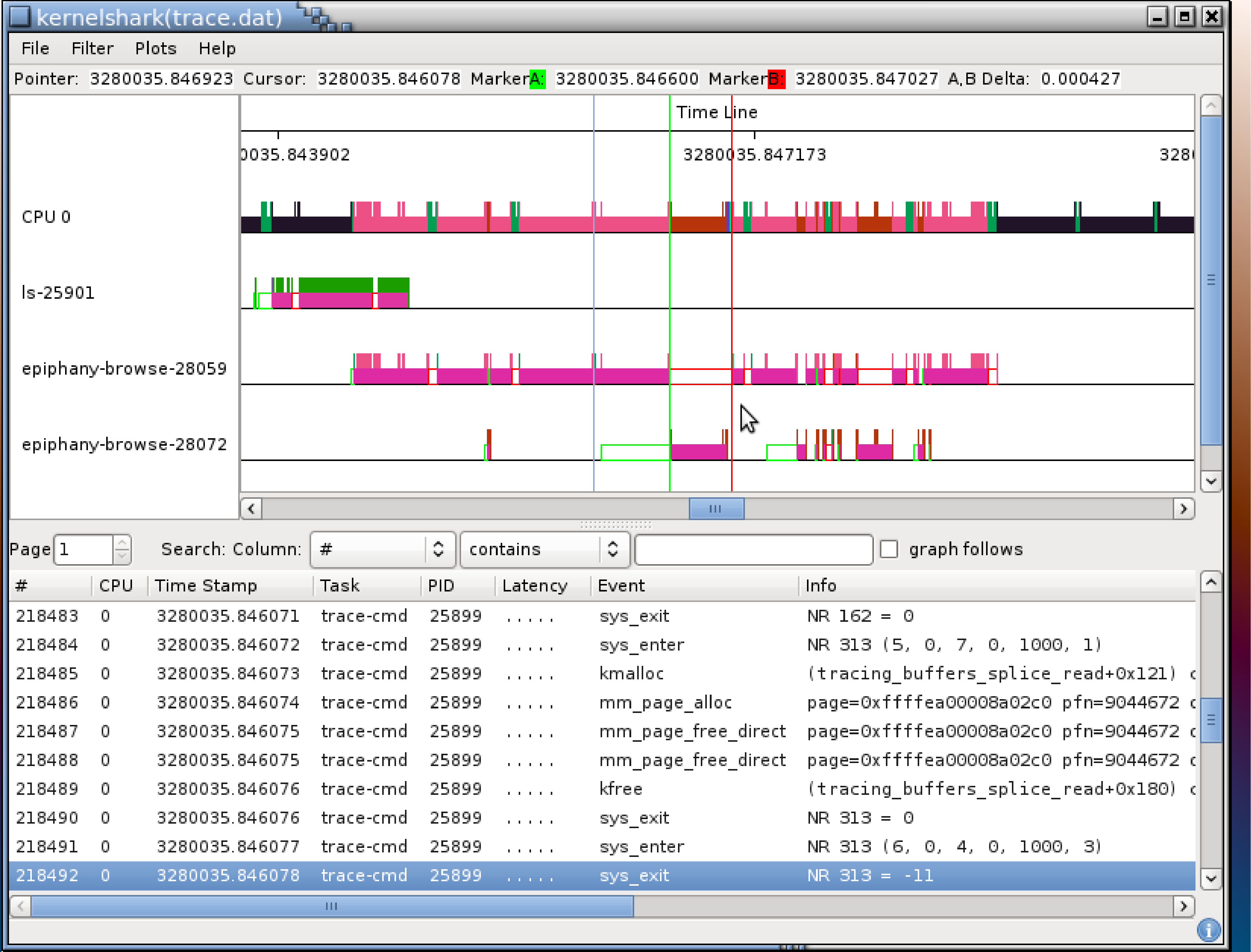
Signals:

```
Time
<idle>-0#idle#45
bash-3658#n0#0
bash-3755#n0#0
load-3798#n0#0
dirname-3799#n0#0
load-3800#n0#0
load-3801#n0#0
load-3802#n0#0
grep-3804#n0#0
grep-3805#n0#0
ls-3806#n0#5
ls-3807#n0#5
ls-3808#n0#4
ps-3803#n0#1
ps-3809#n0#2
dmesg-3810#n0#2
cat-3811#n0#0
sed-3812#n0#0
grep-3813#n0#0
cat-3814#n0#0
killall-3815#n0#1
```

# ftrace

## KernelShark GUI

Next slide is slide #21 from Steve Rostedt's
“KernelShark (quick tutorial)”, ELC 2011
http://elinux.org/images/6/64/Elc2011_rostedt.pdf



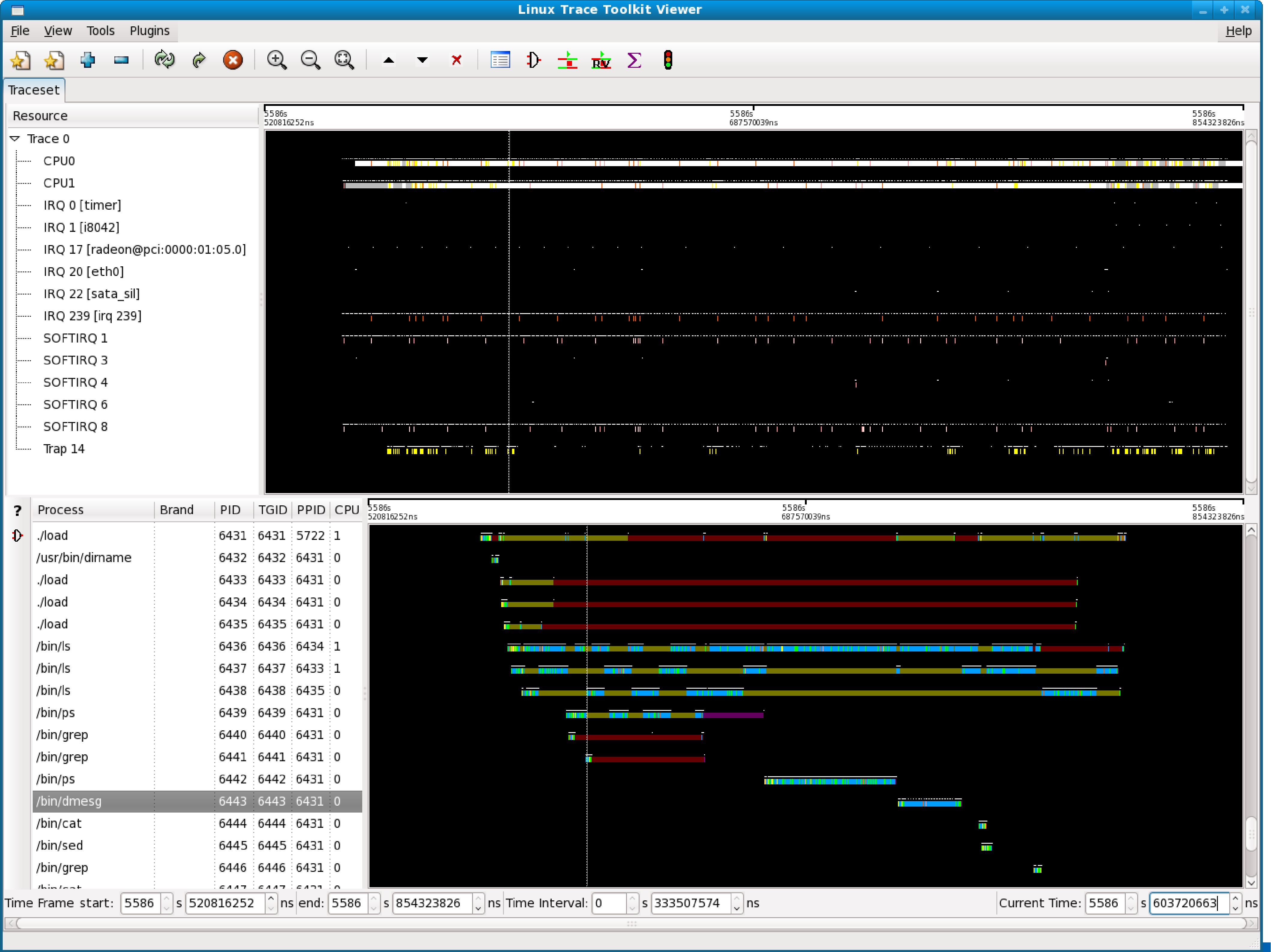
Pointer: 3280035.846923 Cursor: 3280035.846078 MarkerA: 3280035.846600 MarkerB: 3280035.847027 A,B Delta: 0.000427

Page 1 Search: Column: # contains □ graph follows

| # | CPU | Time Stamp | Task | PID | Latency | Event | Info |
|---|---|---|---|---|---|---|---|
| 218483 | 0 | 3280035.846071 | trace-cmd | 25899 | ..... | sys_exit | NR 162 = 0 |
| 218484 | 0 | 3280035.846072 | trace-cmd | 25899 | ..... | sys_enter | NR 313 (5, 0, 7, 0, 1000, 1) |
| 218485 | 0 | 3280035.846073 | trace-cmd | 25899 | ..... | kmalloc | (tracing_buffers_splice_read+0x121) |
| 218486 | 0 | 3280035.846074 | trace-cmd | 25899 | ..... | mm_page_alloc | page=0xffffea00008a02c0 pfn=9044672 |
| 218487 | 0 | 3280035.846075 | trace-cmd | 25899 | ..... | mm_page_free_direct | page=0xffffea00008a02c0 pfn=9044672 |
| 218488 | 0 | 3280035.846075 | trace-cmd | 25899 | ..... | mm_page_free_direct | page=0xffffea00008a02c0 pfn=9044672 |
| 218489 | 0 | 3280035.846076 | trace-cmd | 25899 | ..... | kfree | (tracing_buffers_splice_read+0x180) |
| 218490 | 0 | 3280035.846076 | trace-cmd | 25899 | ..... | sys_exit | NR 313 = 0 |
| 218491 | 0 | 3280035.846077 | trace-cmd | 25899 | ..... | sys_enter | NR 313 (6, 0, 4, 0, 1000, 3) |
| 218492 | 0 | 3280035.846078 | trace-cmd | 25899 | ..... | sys_exit | NR 313 = -11 |

# Linux Trace Toolkit

ltt viewer



Resource

- Trace 0
  - CPU0
  - CPU1
  - IRQ 0 [timer]
  - IRQ 1 [i8042]
  - IRQ 17 [radeon@pci:0000:01:05.0]
  - IRQ 20 [eth0]
  - IRQ 22 [sata_sil]
  - IRQ 239 [irq 239]
  - SOFTIRQ 1
  - SOFTIRQ 3
  - SOFTIRQ 4
  - SOFTIRQ 6
  - SOFTIRQ 8
  - Trap 14

| Process | Brand | PID | TGID | PPID | CPU |
|---|---|---|---|---|---|
| ./load | | 6431 | 6431 | 5722 | 1 |
| /usr/bin/dirname | | 6432 | 6432 | 6431 | 0 |
| ./load | | 6433 | 6433 | 6431 | 0 |
| ./load | | 6434 | 6434 | 6431 | 0 |
| ./load | | 6435 | 6435 | 6431 | 0 |
| /bin/ls | | 6436 | 6436 | 6434 | 1 |
| /bin/ls | | 6437 | 6437 | 6433 | 1 |
| /bin/ls | | 6438 | 6438 | 6435 | 0 |
| /bin/ps | | 6439 | 6439 | 6431 | 0 |
| /bin/grep | | 6440 | 6440 | 6431 | 0 |
| /bin/grep | | 6441 | 6441 | 6431 | 0 |
| /bin/ps | | 6442 | 6442 | 6431 | 0 |
| /bin/dmesg | | 6443 | 6443 | 6431 | 0 |
| /bin/cat | | 6444 | 6444 | 6431 | 0 |
| /bin/sed | | 6445 | 6445 | 6431 | 0 |
| /bin/grep | | 6446 | 6446 | 6431 | 0 |
| /bin/cat | | 6447 | 6447 | 6431 | 0 |

Time Frame start: 5586 s 520816252 ns end: 5586 s 854323826 ns Time Interval: 0 s 333507574 ns Current Time: 5586 s 603720663 ns

# Other Tools

- Write your own tool
- Extend an existing tool

# Other Tools

Any performance tool is potentially useful for understanding a real-time performance issue.

I mentioned only a few that are frequently used to begin an investigation.

Be creative in choosing what tool to use for each unique problem.

The first page of resources (at the end of this talk) list some presentations that describe how tools were used to investigate several different problems.

# Suggestions from Google

## linux performance monitoring tools

Conky
GKrellM
Ksysguard
bonnie
cacti
free
gnome system monitor
htop
iostat
iozone
iptraf
isag

lsof
mpstat
mtr
nagios
netperf
netstat
nfsstat
nmap
ntop
oprofile
ping
pmap

ps
sa
sa2
sadc
sal
sar
smartmontools
smem
spray
ss
strace
sysstat

tcdump
tcpdump
time
top
traceroute
uptime
vmstat
w
wireshark
xload
xosview
/proc

# Useful Tools By Problem Area

# Scheduler overhead and latency

- ftrace sched_switch
- ftrace wakeup
- ltt
- perf sched

# Scheduling issues

- ftrace
- ltt
- perf sched

# Priority inversion

- ftrace sched_switch
- ftrace wakeup
- ltt
- perf sched

# Sleep

- ftrace
- latencyTOP
- ltt
- perf sched

# IRQs disabled

- ftrace irqsoff
- ftrace preemptirqsoff

# Preemption disabled

- ftrace preemptoff
- ftrace preemptirqsoff

# IRQ handlers in IRQ context

- ltt
- /proc/interrupts

# IRQ handlers in thread context

- ftrace
- ltt
- perf sched

# Lock contention

- ltt
- perf stat
- /proc/lock_stat

# SMI

- hwlat_detector

# Cache issues

- perf stat

# Review

Some ways of finding RT performance issues

- User interaction
- Instrumenting RT application
- Instrumented proxy RT applications (cyclictest)
- Measurement tools (ftrace, ltt, perf, etc)

# Review

Some ways of finding RT performance issues

- Different types of data available
    + describe existence or magnitude of problem
    + describe what is going on during problem

The tools have many more capabilities than were shown in this presentation.

# Questions?

(Resources will be listed in the following slides.)

# How to get a copy of the slides

1) leave a business card with me

2) frank.rowand@am.sony.com

# Resources

## Examples of using tools:

Survey of Linux Measurement and Diagnostic Tools
http://elinux.org/images/c/cf/Survey_of_linux_measurement_and_diagnostic_tools.pdf

Adventures In Real Timer Performance Tuning Part 1
http://elinux.org/images/b/b0/Adventures_in_real_time_performance_tuning_part_1-no_hidden.pdf

Adventures In Real Timer Performance Tuning Part 2
http://elinux.org/images/d/d2/Adventures_in_real_time_performance_tuning_part_2-no_hidden.pdf

Musings on Analysis of Measurements of a Real-Time Workload
http://elinux.org/images/4/41/Musings_on_analysis_of_measurements_of_a_real-time_workload.pdf

Identifying Embedded Real-Time Latency Issues: I-cache and Locks
http://elinux.org/images/d/dd/Elc2011_rowand.pdf

# Resources

cyclictest
  https://rt.wiki.kernel.org/index.php/Cyclictest

ftrace (also see kernel shark, sched_switch)
  kernel source: Documentation/trace/ftrace.txt
  http://people.redhat.com/srostedt/ftrace-tutorial-linux-con-2009.odp
  http://people.redhat.com/srostedt/ftrace-tutorial.odp
  http://people.redhat.com/srostedt/ftrace-embedded.odp
  http://people.redhat.com/srostedt/ftrace-latency-osadl-2009.odp
  http://people.redhat.com/srostedt/ftrace-world.odp

hwlatdect
  kernel source: Documentation/hwlat_detector.txt
  rt-tests: src/hwlatdetect/hwlat.txt

Kernel Shark Tutorial
  http://elinux.org/images/6/64/Elc2011_rostedt.pdf

# Resources

latencytop
  http://www.latencytop.org/

lrtb
  http://www.opersys.com/lrtbf/

ltt ng
  http://lttng.org/

perf
  perf help
  kernel source: tools/perf/Documentation/

realfeel
  http://brain.mcmaster.ca/~hahn/realfeel.c

# Resources

rt-tests (includes cyclictest, hwlatdetect, other)
  linux/kernel/git/clrkwllms/rt-tests.git
  http://git.kernel.org/?p=linux/kernel/git/clrkwllms/rt-tests.git;a=summary

sched_switch
  linux-rt-users (http://vger.kernel.org)
  Analyze sched_switch ftrace data with vcd viewer
  Herman ten Brugge <hermantenbrugge@xxxxxxx>
  Thu, 04 Jun 2009 20:58:57 +0200

timechart
  http://blog.fenrus.org/?p=5

Real Time wiki
  https://rt.wiki.kernel.org/index.php/Main_Page