RT patch for Celleb

- patch status and performance measurements -

Tsutomu OWA
Corporate Software Engineering Center, TOSHIBA
Nov. 3. 2007

Contents

- **Background**
  - – realtime-preempt patch (RT patch)
- **RT patch status for Celleb/PowerPC64**
- **Measurements**
  - – Metrics
  - – Measurement Environment
  - – Results
- **Summary**

realtime-preempt patch (RT patch)

- **Patch created and maintained by Ingo Molnar, Thomas Gleixner, Steven Rostedt et.al.**
  - From at least 2004, steadily merged into mainline
  - Add preemption points in the kernel
    - spinlock → mutex w/ priority inheritance
    - hard/soft interrupts → kernel threads
    - etc.
- **URLs**
  - **“the Wiki Web for the CONFIG_PREEMPT community, and real-time Linux in general”.**
    - http://rt.wiki.kernel.org/
  - **“A realtime preemption overview”**
    - http://lwn.net/Articles/146861/
  - And many other news/materials on the net.

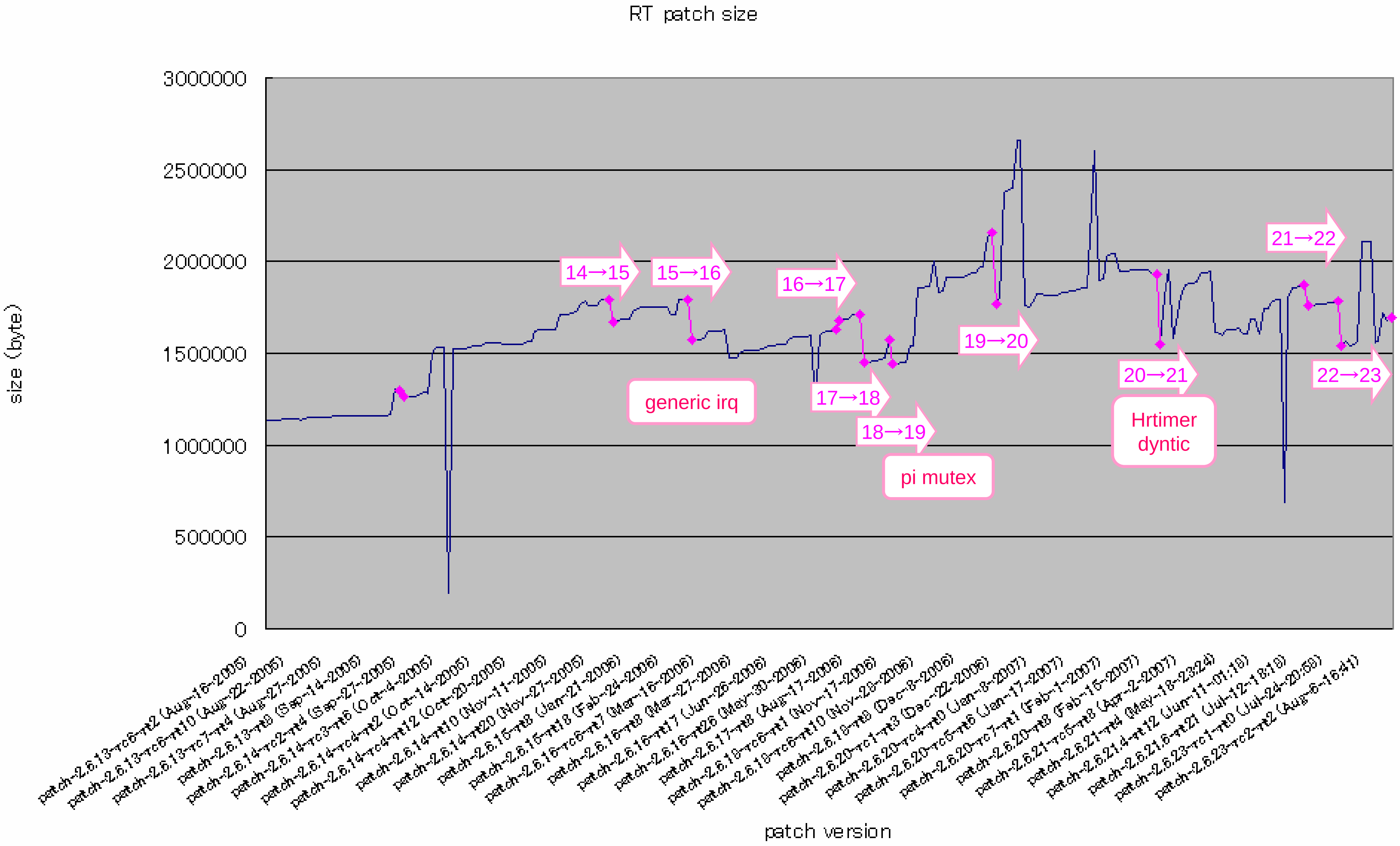
RT patch status - latest version

- **patch-2.6.23-rt1 was announced by Steven Rostedt on October 12, 2007**
  - Against linux-2.6.23
  - i386, x86_64, arm, mips, sh, powerpc, sparc,…

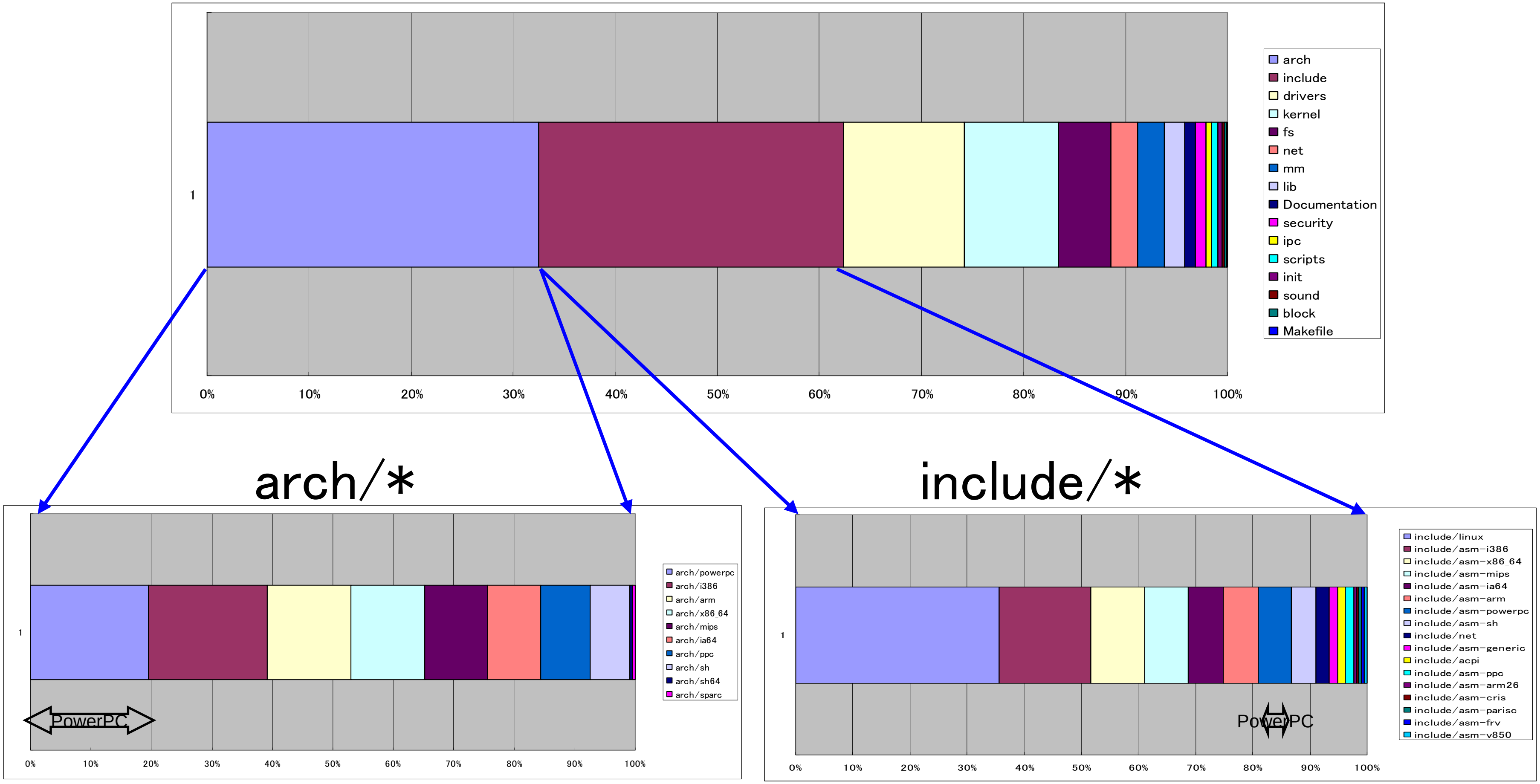
patch-2.6.23-rt1 is used in this presentation.

- **patch-2.6.23.1-rt5 was released on October 29, 2007**
  - patch-2.6.23-rt2: October 25, 2007
  - patch-2.6.23-rt3: October 25, 2007
  - patch-2.6.23.1-rt4: October 27, 2007

RT patch status - size



RT patch status - # of files


arch/*

include/*

TOSHIBA
Leading Innovation >>>

RT patch status - Celleb/PowerPC64

- **patch-2.6.23-rt1: Applies cleanly to linux-2.6.23**
  - Compile, boot and run tested on Celleb (Cell /B.E., PowerPC64 based)

- **Supported configs for powerpc64**
  - CONFIG_GENERIC_TIME
  - CONFIG_MCOUNT, CONFIG_FUNCTION_TRACE

```
pcscd-1772  0D..2 6867us : deactivate_task <pcscd-1772> (-2 1)
pcscd-1772  0D..2 6867us : dequeue_task (deactivate_task)
 <idle>-0     0D..2 6870us : __switch_to (__schedule)
```

  - CONFIG_CLOCKEVENT
  - And others?
  - CONFIG_LOCKDEP, CONFIG_STACKTRACE, CONFIG_TRACE_IRQFLAGS
    - Coming…

RT patch status - Celleb/PowerPC64 (Cnt.)

- **Patches not included in 2.6.23-rt1**
  - "RT: fix spin_trylock_irq"
    - From sebastien.dugue@bull.net
    - http://lkml.org/lkml/2007/10/11/120

merged into 2.6.23-rt2

  - "Hook compat_sys_nanosleep up to high res timer code"
    - By Anton Blanchard on Mon, 14 Oct 2007.
    - http://lkml.org/lkml/2007/10/14/190

merged into 2.6.23-rt?

  - "powerpc: 64 bits irqtrace / lockdep support"
    - From Benjamin Herrenschmidt on 15 Oct 2007
    - http://patchwork.ozlabs.org/linuxppc/patch?id=14172
    - Not RT specific but useful.

merged into 2.6.25?

RT patch status - Celleb/PowerPC64 (Cnt.)

- **Patches not included in 2.6.23-rt1**
  - “replace preempt_schedule w/ preempt_schedule_irq”
    - From Tsutomu OWA
    - http://lkml.org/lkml/2007/5/22/133

Still needed. Applied w/ 2.6.23-rt1

  - “Implement clockevents driver for powerpc” and its series
    - From Tony Breeds
    - http://patchwork.ozlabs.org/linuxppc/patch?id=13350
    - Still in discussion?

Not sure…

Measurement

- **Metrics**
- **Measurement Environment**
- **Results**

Measurements - Metrics

- **Previous works**
  - Many test results for x86
  - Less test results for other architectures / platforms
  - Adhock-metrics (often) specific to each platform
    - Different / too old kernel base version
    - Different period of time to measure
    - Different load, etc,etc.
    - http://elinux.org/Realtime_Testing_Best_Practices

- **What we'd like to have!**
  - Common test cases in order to compare results in a consistent manner
    - So that we'll be on the same ground
  - "IBM Test Cases" and "Cyclictest"
    - are widely used in the RT community
    - could be common testbeds

Measurements - Metrics (Cnt.)

- **IBM Test Cases**
  - "These test cases for testing a -rt kernel were contributed by IBM's Real-Time Linux development team. They include mostly functional tests, although some performance tests are slowly being added. If you would like to contribute, please use the discussion list above and contact User:dvhart. "
  - http://rt.wiki.kernel.org/index.php/IBM_Test_Cases
  - "Internals of the RT Patch" presented at OLS2007 by Steven Rostedt and Darren V. Hart.
    - http://www.linuxsymposium.org/2007/view_abstract.php?content_key=75
- **Cyclictest**
  - "Cyclictest is a high resolution test program, written by Thomas Gleixner "
  - http://rt.wiki.kernel.org/index.php/Cyclictest

IBM Test Cases used in this presentation

- **gtod_latency**
  - "to measure the time between several pairs of calls to gettimeofday()." (from gtod_latency.c)
  - On a SCHED_FIFO (99) priority thread.
- **async_handler**
  - "Measure the latency involved with asynchronous event handlers. Specifically it measures the latency of the pthread_cond_signal call until the signalled thread is scheduled." (from async_handler.c)
  - Two threads with priority set to 89.
- **sched_latency**
  - "A thread is created at a priority of 89. It periodically sleeps for a specified duration(PERIOD).
    - The delay is measured as

      delay = (now - start - i*PERIOD) converted to microseconds

      where, now = CLOCK_MONOTONIC gettime in ns, start = CLOCK_MONOTONIC gettime at the start of the test, i = iteration number, PERIOD = the period chosen" (from sched_latency.c)

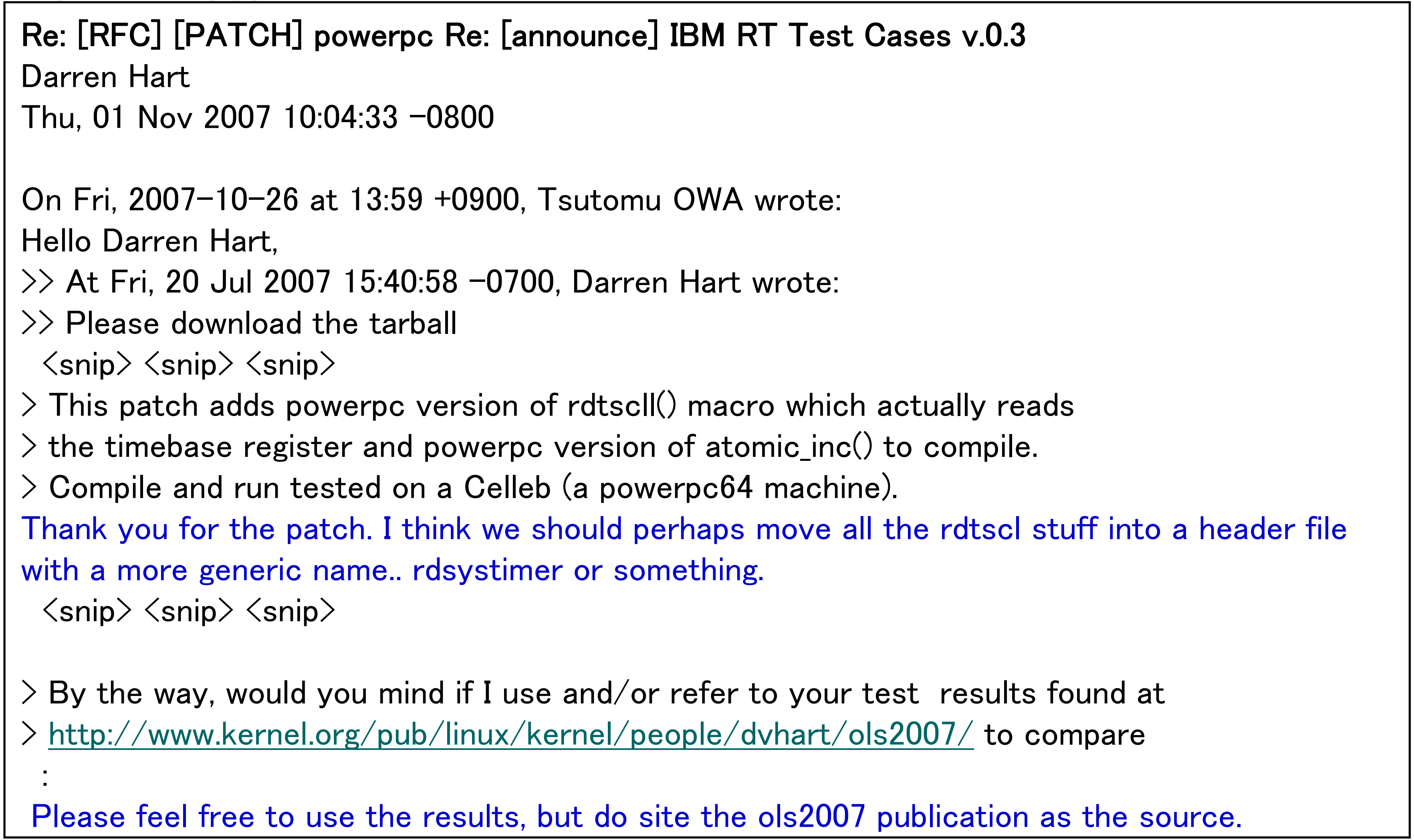
IBM Test Cases - arch dependencies

```
#if defined(_i386_)
#define rdtscll(val)    _asm_ _volatile_(″rdtsc″ : ″=A″ (val))
#elif defined(_x86_64_)
#define rdtscll(val)                                        ¥
            do {                                            ¥
                        uint32_t low, high;                 ¥
                        _asm_ _volatile_ (″rdtsc″ : ″=a″ (low), ″=d″ (high)); ¥
                        val = (uint64_t)high << 32 | low;   ¥
            } while(0)
#endif
```

```
static inline int atomic_add(int i, atomic_t *v)
{
            int _i;     _i = i;     asm volatile(           ¥
                                    ″lock; xaddl %0, %1;“   ¥
                                    :″=r″(i)                ¥
                                    :″m″(v->counter), ″0″(i));  ¥
            return i + _i;
}
```

IBM Test Cases - patch for powerpc

- http://www.mail-archive.com/linux-rt-users@vger.kernel.org/msg01830.html (local copy)



Re: [RFC] [PATCH] powerpc Re: [announce] IBM RT Test Cases v.0.3
Darren Hart
Thu, 01 Nov 2007 10:04:33 -0800

On Fri, 2007-10-26 at 13:59 +0900, Tsutomu OWA wrote:
Hello Darren Hart,
>> At Fri, 20 Jul 2007 15:40:58 -0700, Darren Hart wrote:
>> Please download the tarball
<snip> <snip> <snip>
> This patch adds powerpc version of rdtscll() macro which actually reads
> the timebase register and powerpc version of atomic_inc() to compile.
> Compile and run tested on a Celleb (a powerpc64 machine).
Thank you for the patch. I think we should perhaps move all the rdtscl stuff into a header file with a more generic name.. rdsystimer or something.
<snip> <snip> <snip>

> By the way, would you mind if I use and/or refer to your test results found at
> http://www.kernel.org/pub/linux/kernel/people/dvhart/ols2007/ to compare
:
Please feel free to use the results, but do site the ols2007 publication as the source.

Measurement

- Metrics

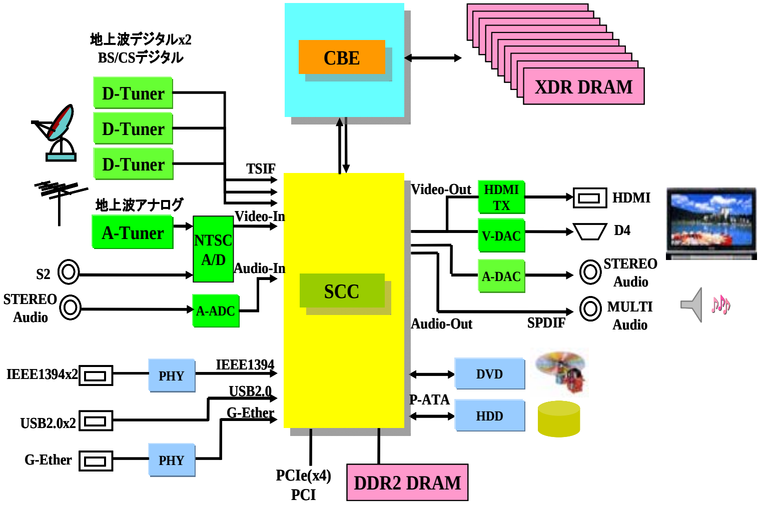
- **Measurement Environment**

- Results

Measurement env.

- **Cell Reference Set**
  - http://www.toshiba.co.jp/tech/review/2006/06
  - http://www.semicon.toshiba.co.jp/product/micro/cell/reference.html

  - HW
    - Cell/B.E. (PowerPC64 based PPU)



  - SW
    - Linux on a Hypervisor OS

Measurement env. (Cnt.)

- **Kernel Configuration**
  - Linux-2.6.23 vanilla
    - CONFIG_PREEMPT
  - Linux-2.6.23 + patch-2.6.23-rt1
    - CONFIG_PREEMPT_RT

- **Userland Configuration**
  - Fedora 7
    - gcc version 4.1.2, thread model: posix
    - glibc-2.6
    - run level: 3

- **Load**
  - Make linux-2.6.23 kernel (% make)
    - % uptime

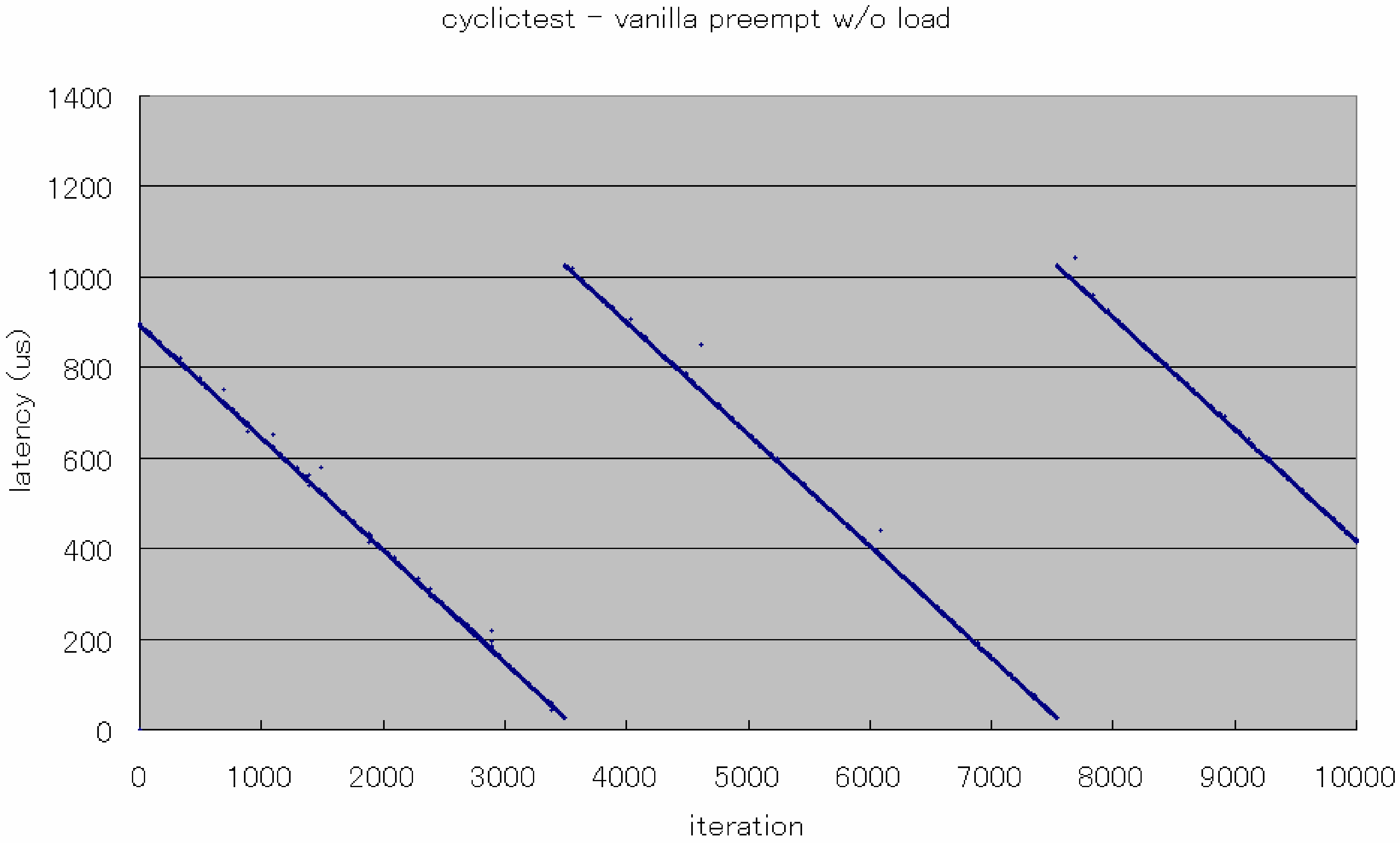
```
17:43:21 up  3:01,  2 users,  load average: 1.64, 0.67, 0.36
```

More than 15 deamons are running.

Measurement

- Metrics
- Measurement Environment
- **Results**

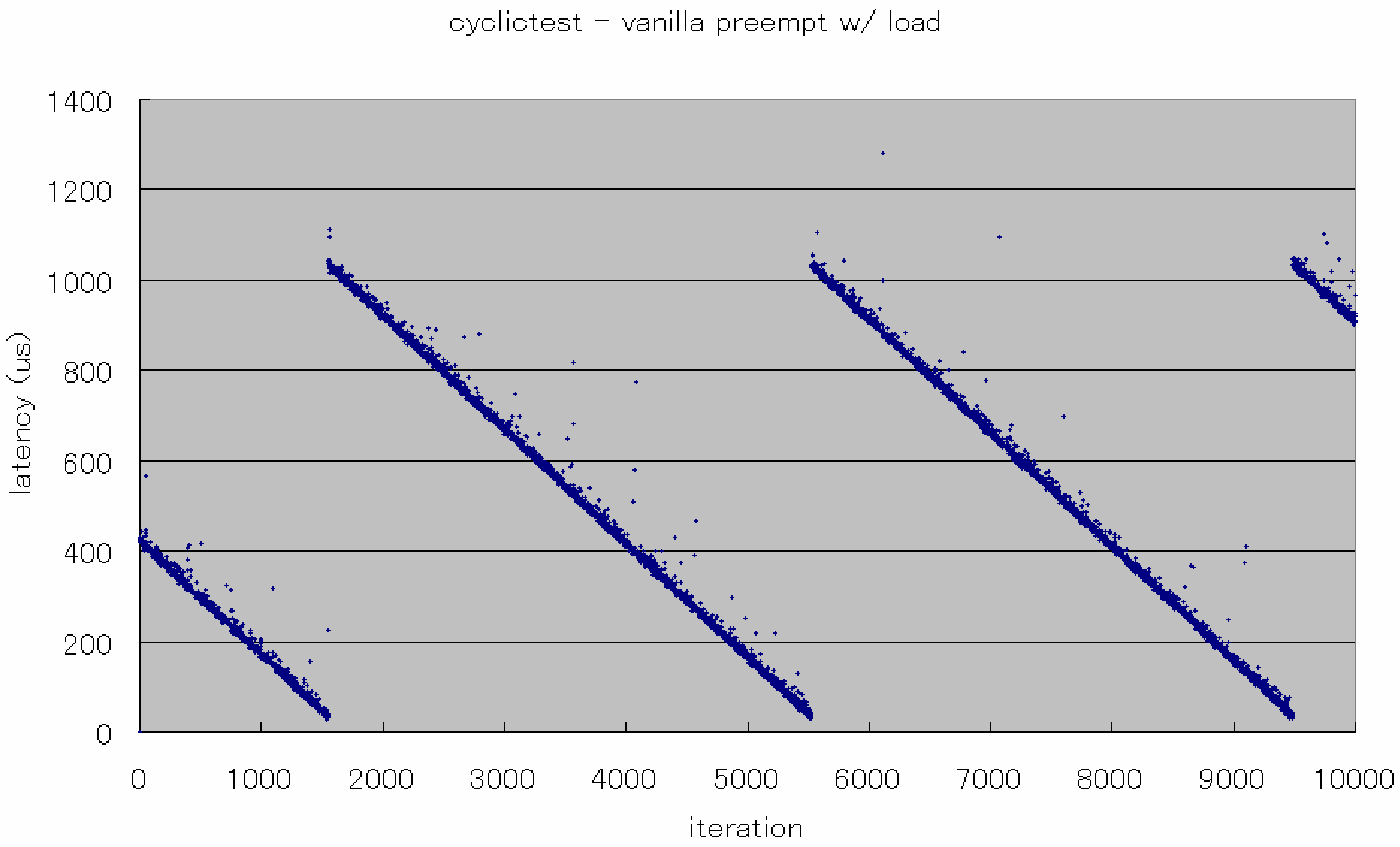
cyclictest – vanilla preempt w/o load



Warning: High resolution timers not available

TOSHIBA
Leading Innovation >>>

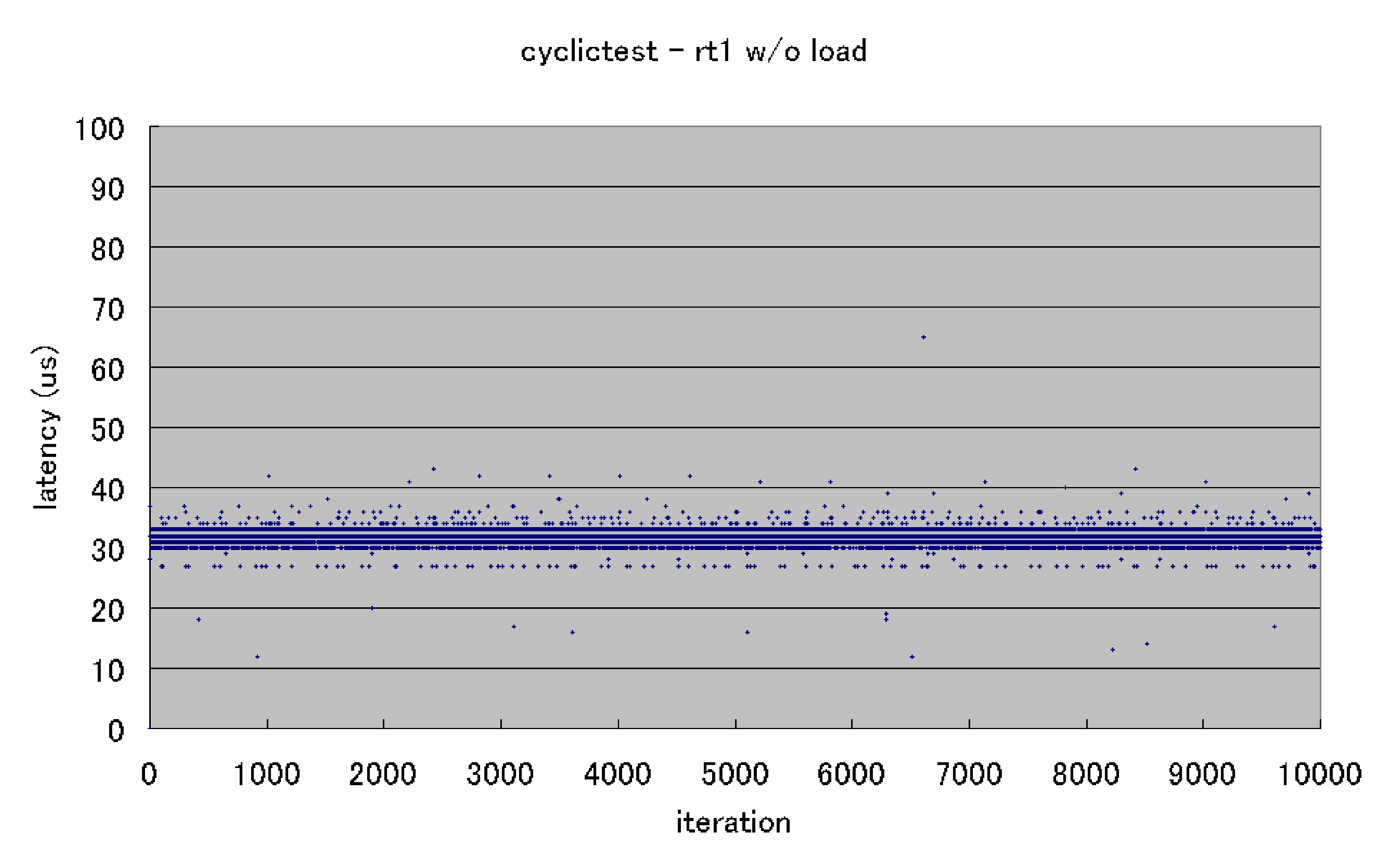
cyclictest – vanilla preempt w load



Warning: High resolution timers not available

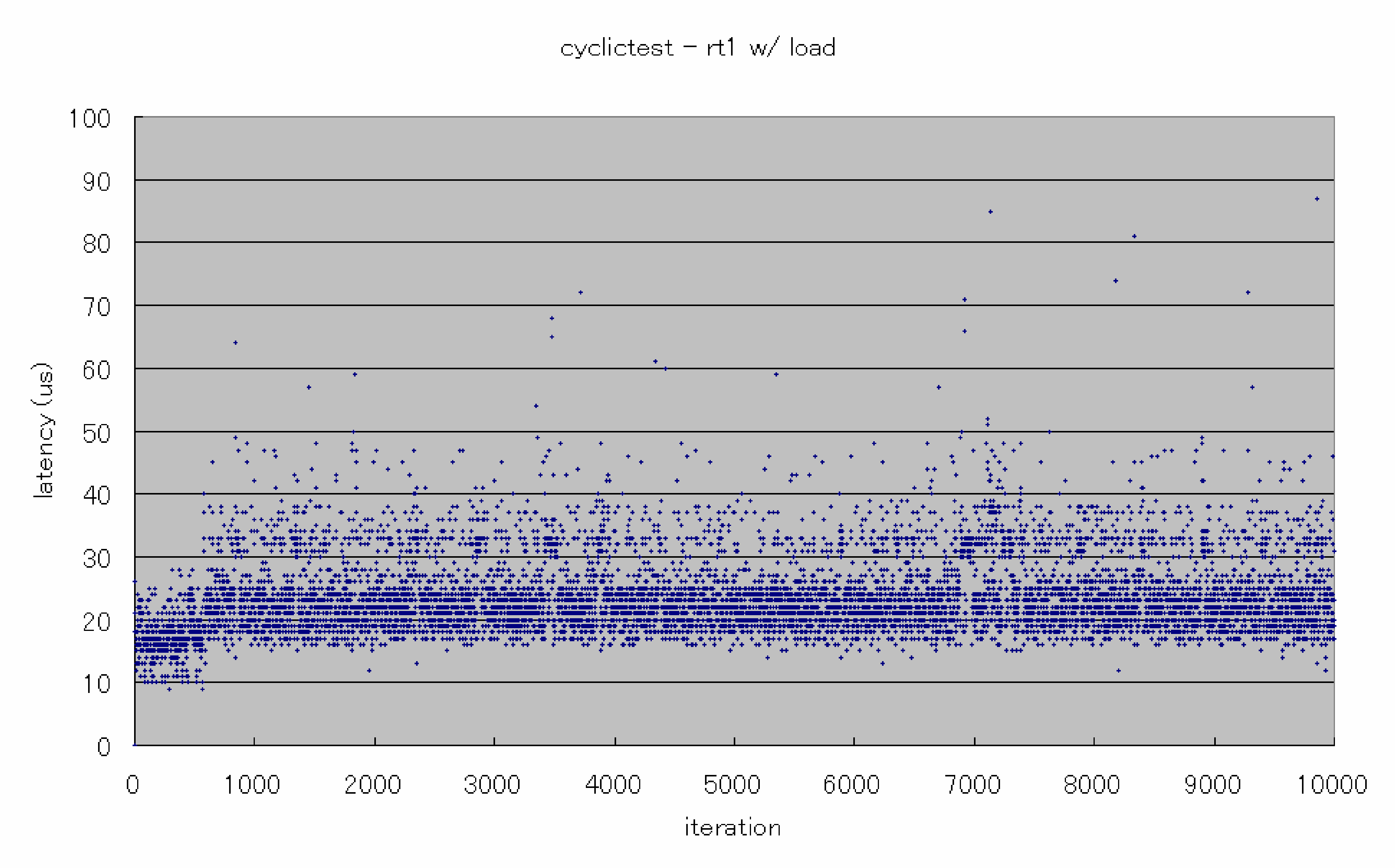
TOSHIBA
Leading Innovation >>>

cyclictest – rt1 w/o load



TOSHIBA
Leading Innovation >>>

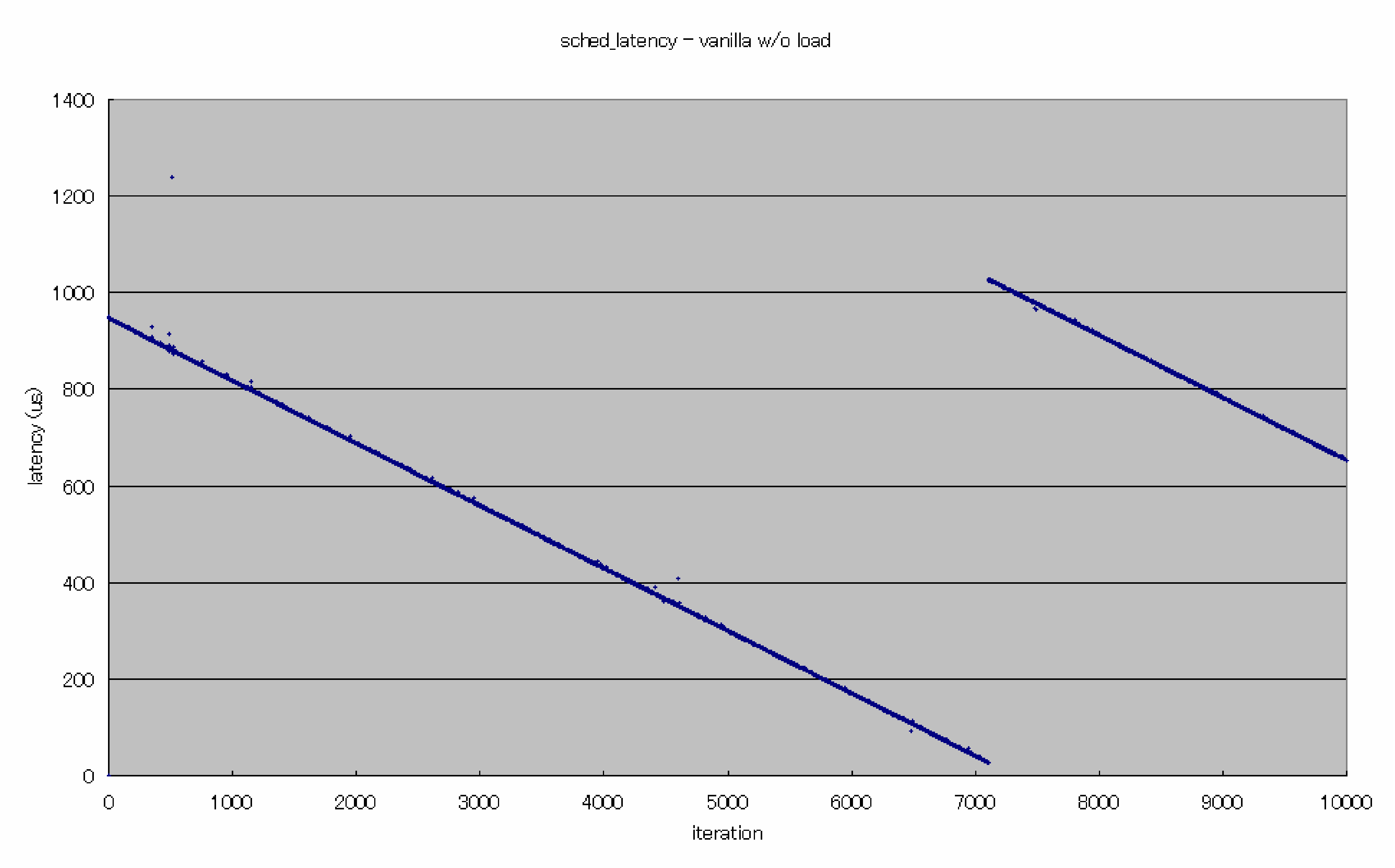
cyclictest – rt1 w load



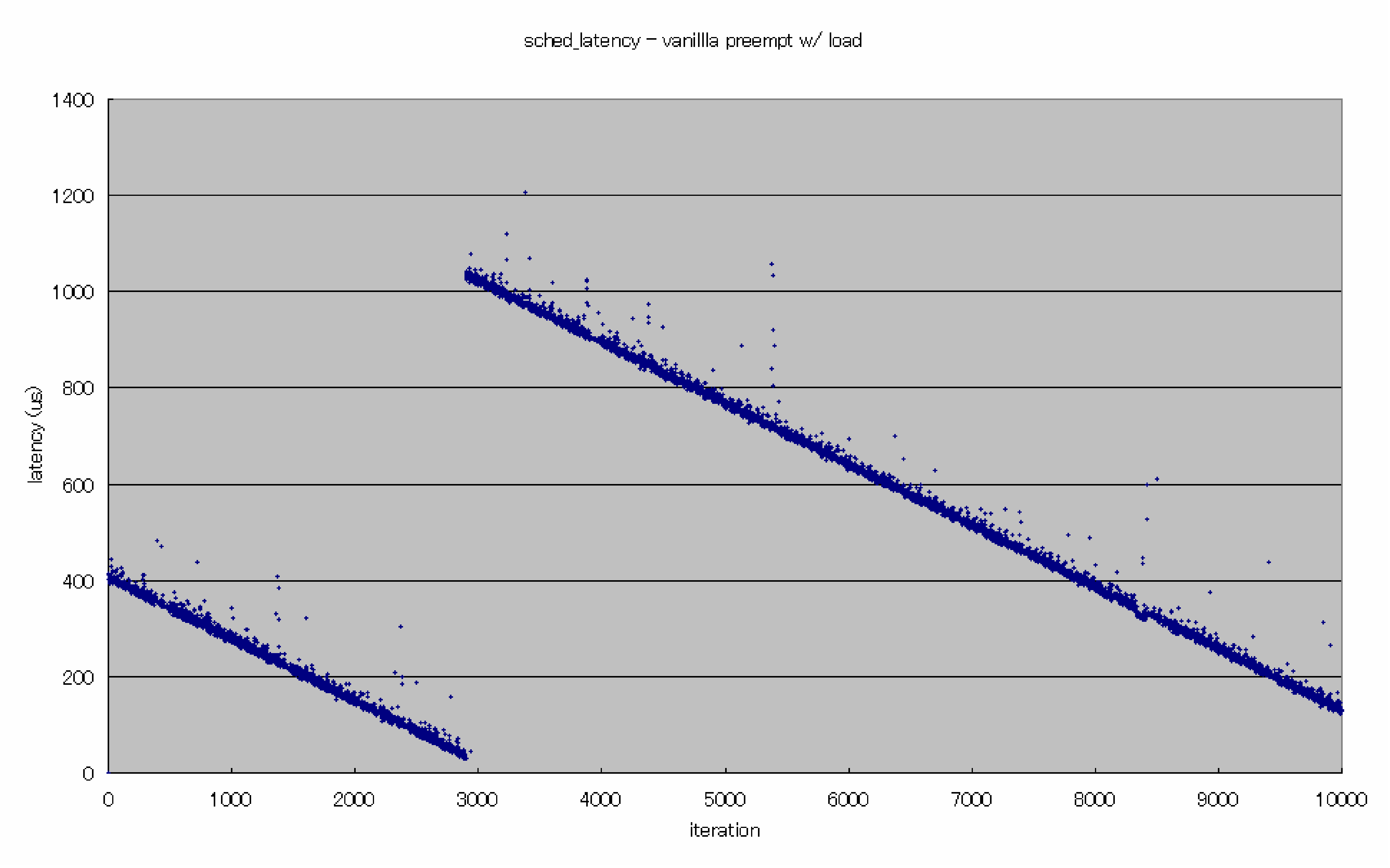
gtod_latency

latency (us)	vanilla preempt w/o load	vanilla preempt w/ load	rt1 w/o load	rt1 w load
0	842842	842820	168791	167132
1	156986	157017	830350	831984
2	13	7	9	12
3	75	82	3	6
4	71	59	0	0
5	1	3	14	1
6	4	3	255	15
7	0	2	565	46
8	1	0	12	460
9	1	1	0	331
10	0	0	1	11
11	1	0	0	0
12	2	1	0	0
13	1	2	0	0
14	1	1	0	0
15	0	0	0	0
16	0	0	0	1
17	1	1	0	0
18	0	0	0	0

sched_latency – vanilla preempt w/o load

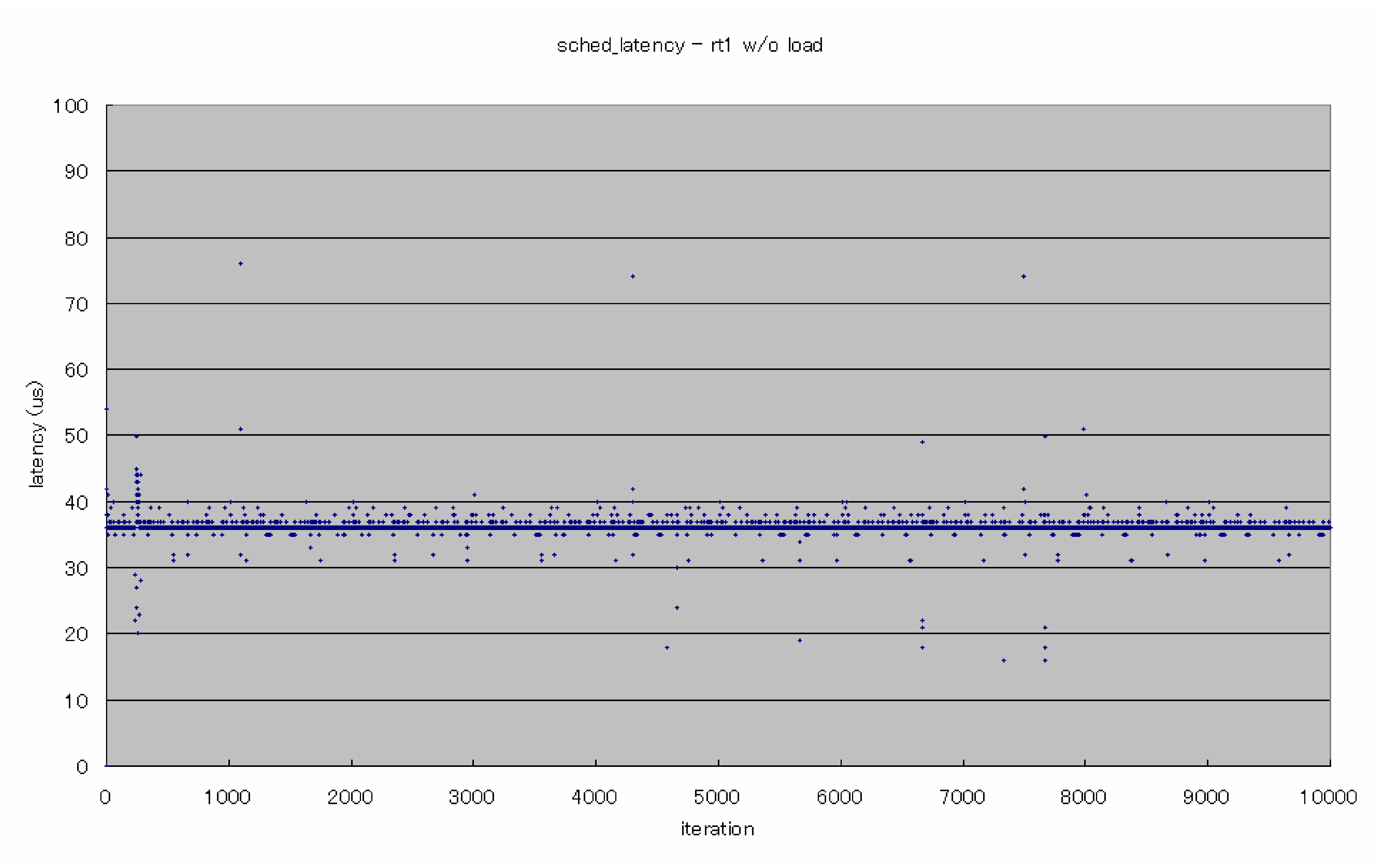


sched_latency – vanilla preempt w load

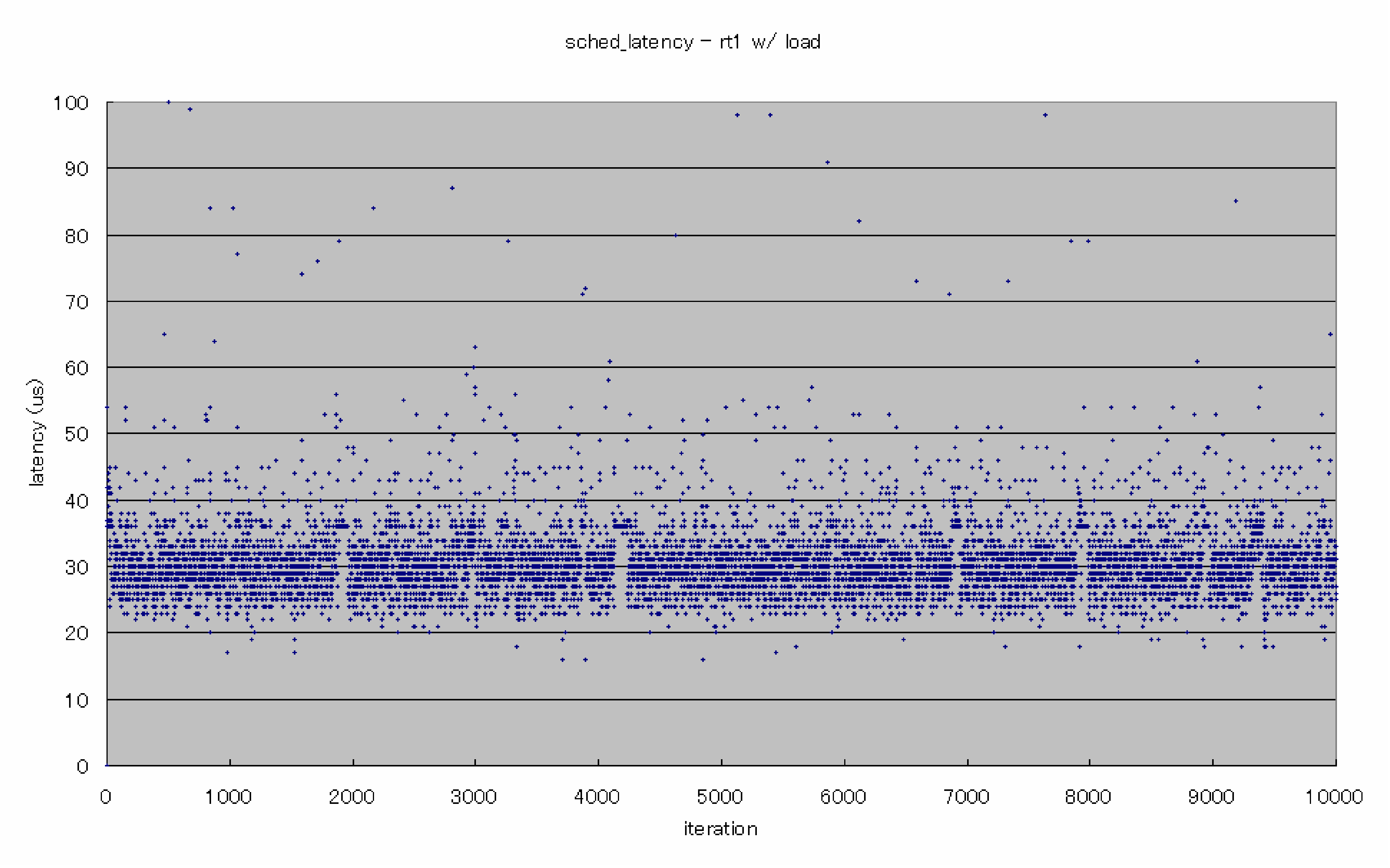


TOSHIBA
Leading Innovation >>>

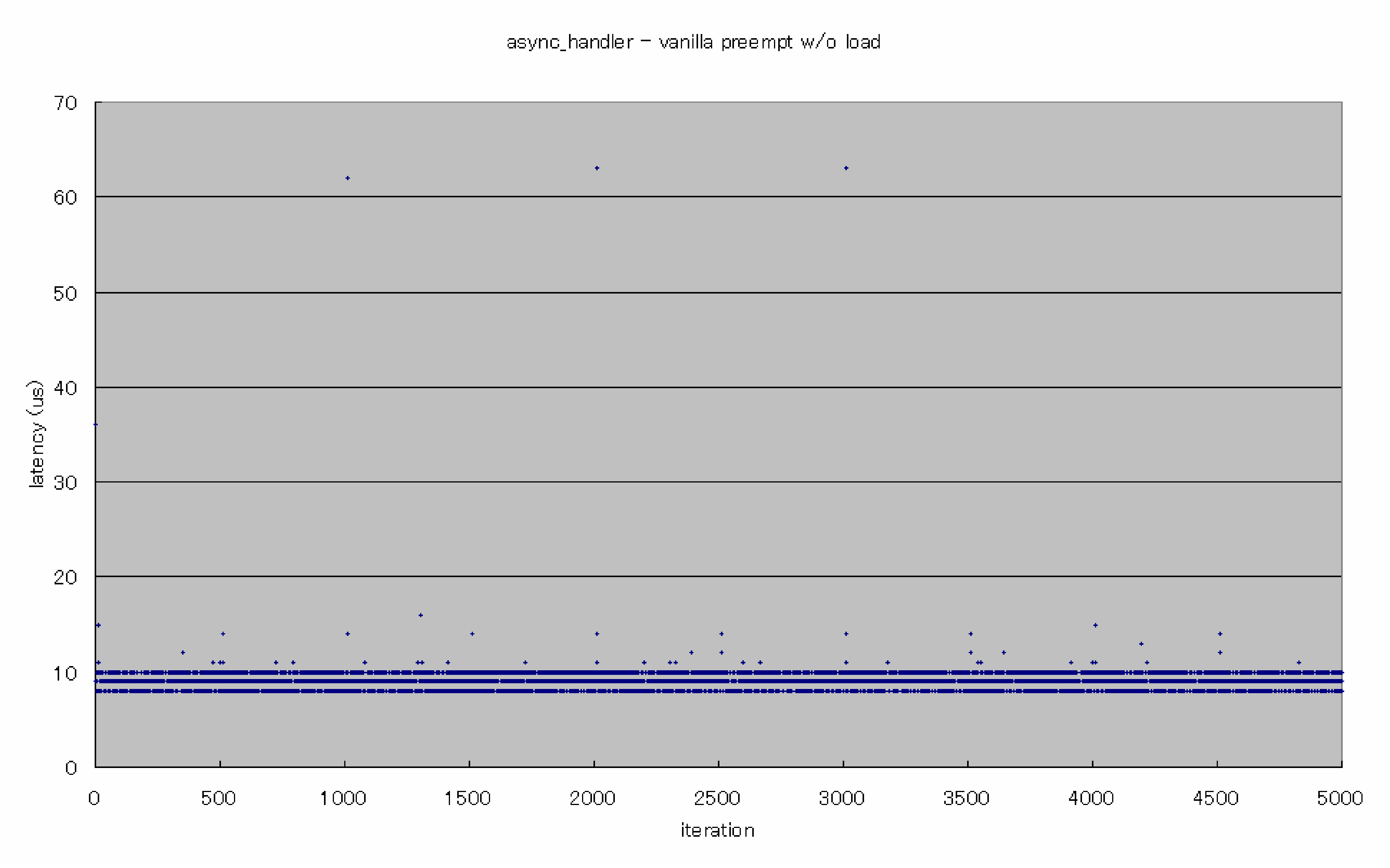
sched_latency – rt1 w/o load



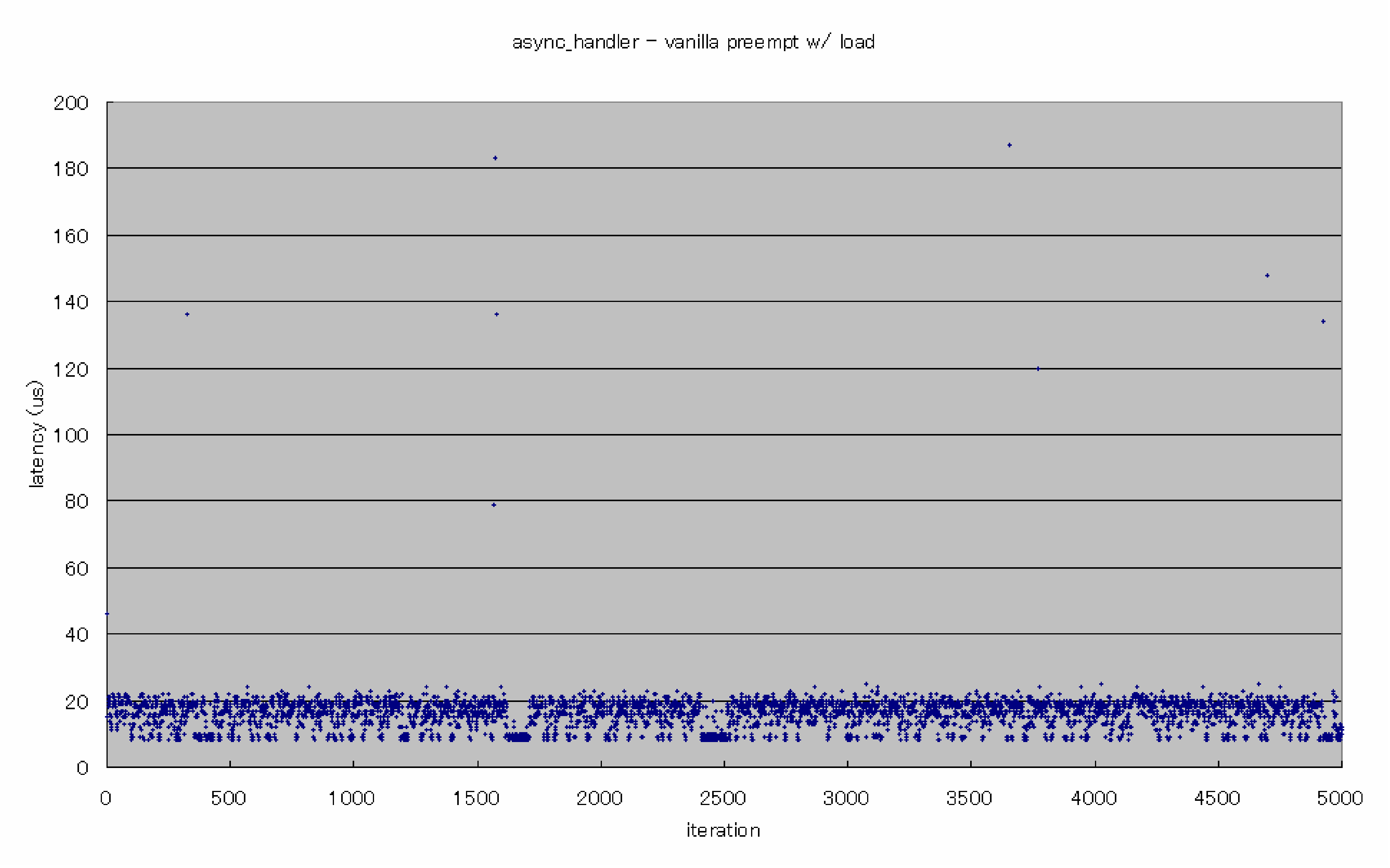
sched_latency – rt1 w load



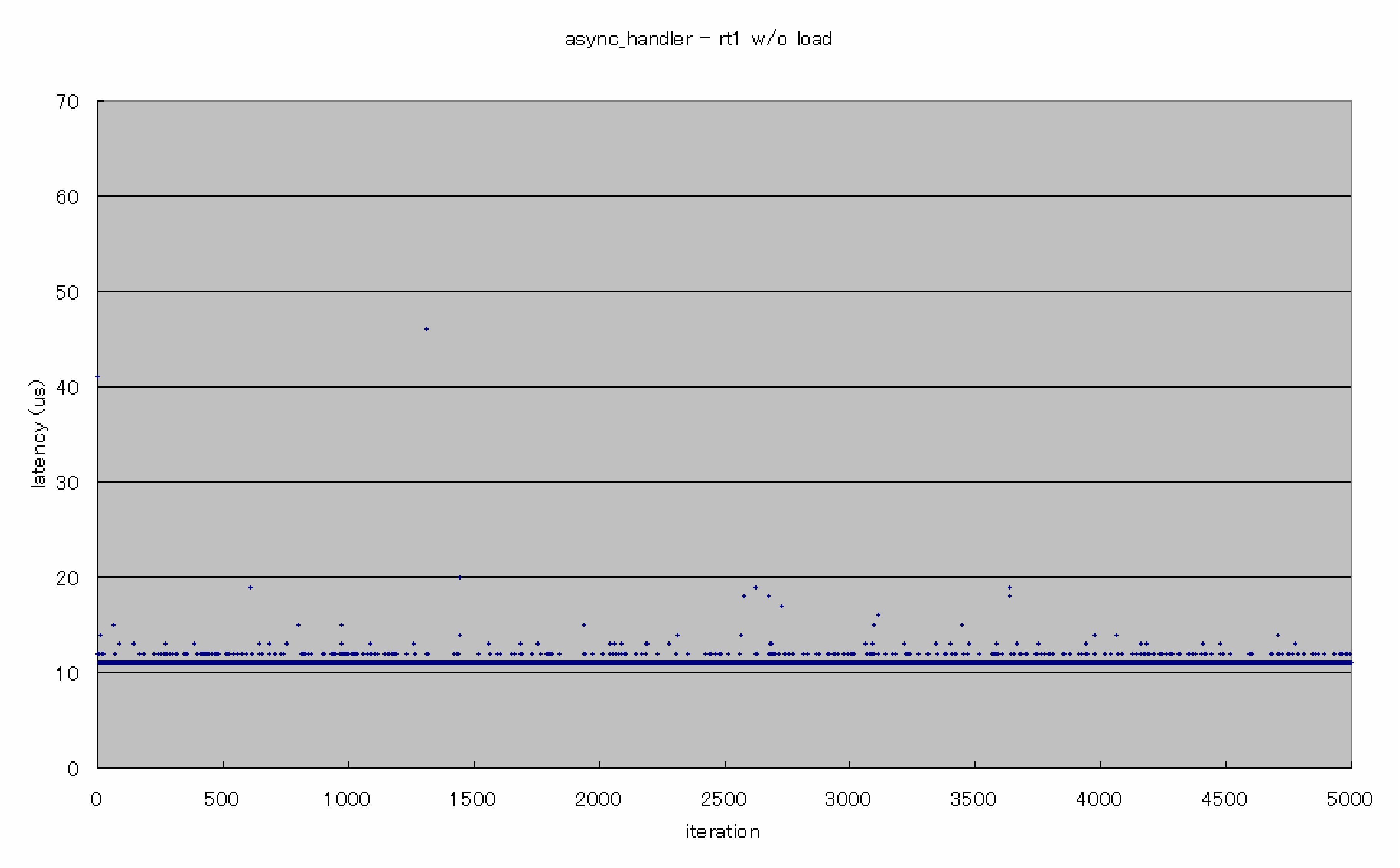
async_handler - vanilla preempt w/o load



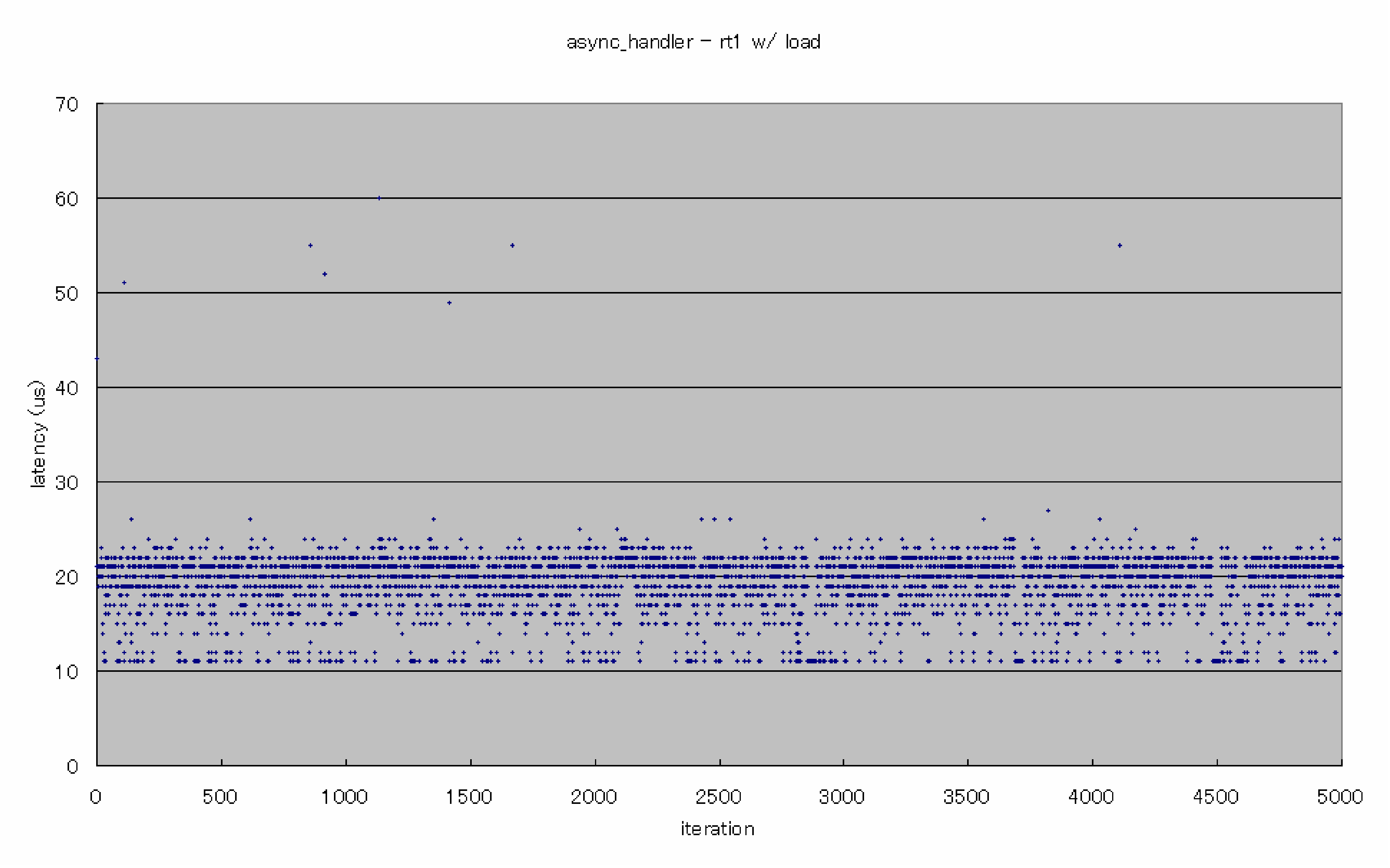
async_handler - vanilla preempt w load



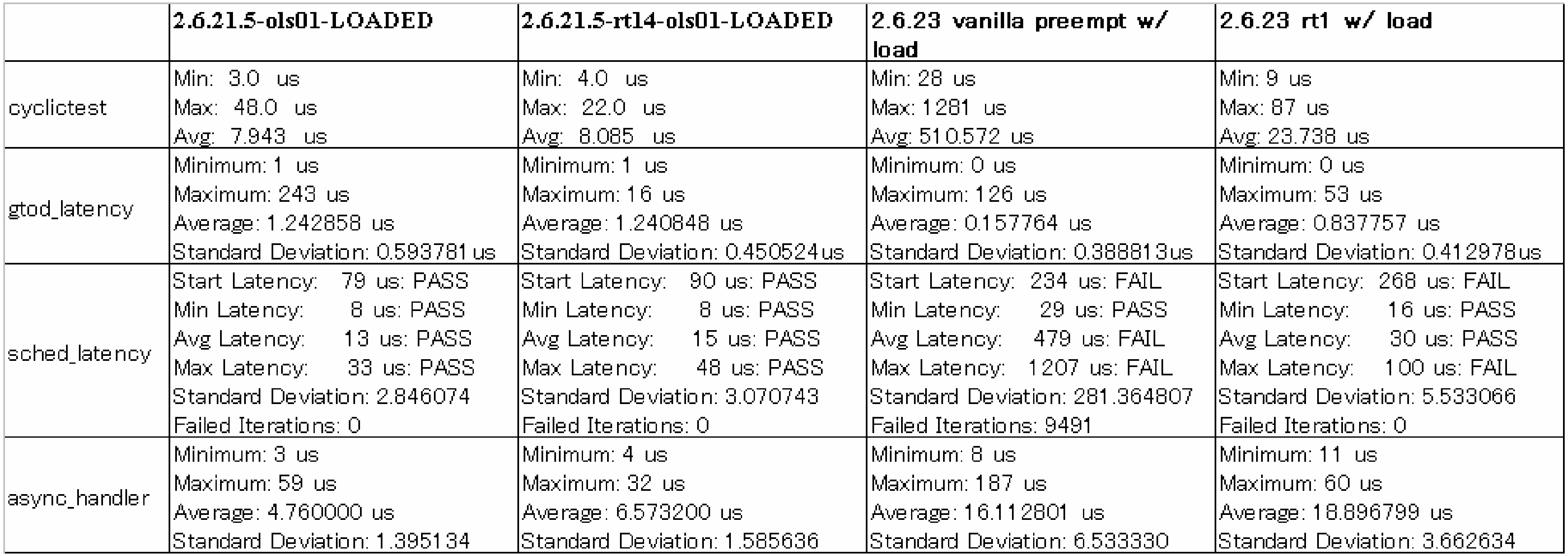
async_handler – rt1 w/o load



async_handler – rt1 w load



Comparison with x86



	2.6.21.5-ols01-LOADED	2.6.21.5-rt14-ols01-LOADED	2.6.23 vanilla preempt w/ load	2.6.23 rt1 w/ load
cyclictest	Min: 3.0 us Max: 48.0 us Avg: 7.943 us	Min: 4.0 us Max: 22.0 us Avg: 8.085 us	Min: 28 us Max: 1281 us Avg: 510.572 us	Min: 9 us Max: 87 us Avg: 23.738 us
gtod_latency	Minimum: 1 us Maximum: 243 us Average: 1.242858 us Standard Deviation: 0.593781us	Minimum: 1 us Maximum: 16 us Average: 1.240848 us Standard Deviation: 0.450524us	Minimum: 0 us Maximum: 126 us Average: 0.157764 us Standard Deviation: 0.388813us	Minimum: 0 us Maximum: 53 us Average: 0.837757 us Standard Deviation: 0.412978us
sched_latency	Start Latency: 79 us: PASS Min Latency: 8 us: PASS Avg Latency: 13 us: PASS Max Latency: 33 us: PASS Standard Deviation: 2.846074 Failed Iterations: 0	Start Latency: 90 us: PASS Min Latency: 8 us: PASS Avg Latency: 15 us: PASS Max Latency: 48 us: PASS Standard Deviation: 3.070743 Failed Iterations: 0	Start Latency: 234 us: FAIL Min Latency: 29 us: PASS Avg Latency: 479 us: FAIL Max Latency: 1207 us: FAIL Standard Deviation: 281.364807 Failed Iterations: 9491	Start Latency: 268 us: FAIL Min Latency: 16 us: PASS Avg Latency: 30 us: PASS Max Latency: 100 us: FAIL Standard Deviation: 5.533066 Failed Iterations: 0
async_handler	Minimum: 3 us Maximum: 59 us Average: 4.760000 us Standard Deviation: 1.395134	Minimum: 4 us Maximum: 32 us Average: 6.573200 us Standard Deviation: 1.585636	Minimum: 8 us Maximum: 187 us Average: 16.112801 us Standard Deviation: 6.533330	Minimum: 11 us Maximum: 60 us Average: 18.896799 us Standard Deviation: 3.662634

2.6.21*-LOADED: quoted from
“Internals of the RT Patch” presented at OLS2007 by Steven Rostedt and Darren V. Hart.
http://www.linuxsymposium.org/2007/view_abstract.php?content_key=75

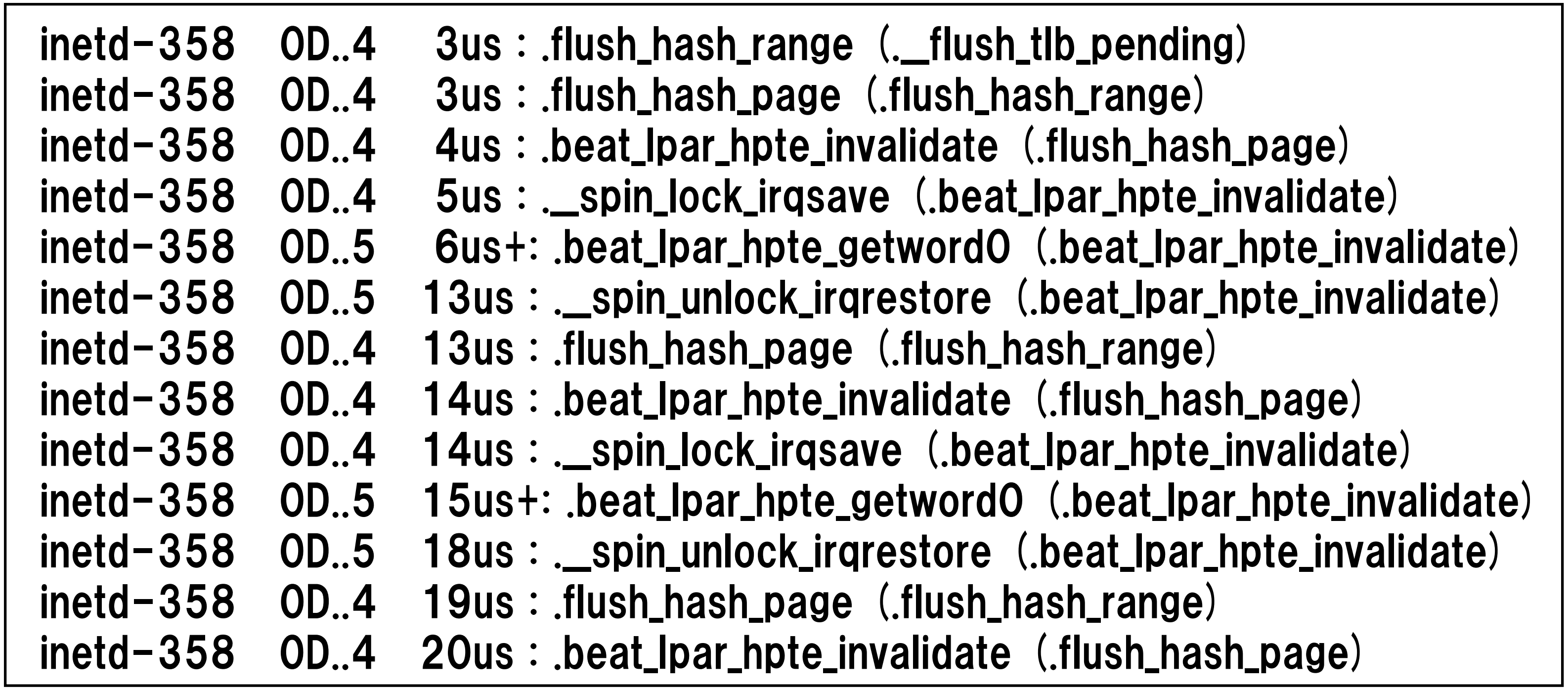
Celleb specific optimization

- **arch/popwerpc/mm/tlb_64.c: hpte_need_flush()**

```
#ifdef CONFIG_PREEMPT_RT
        ／＊
         ＊ Since flushing tlb needs expensive hypervisor call（s） on celleb，
         ＊ always flush it on RT to reduce scheduling latency.
         ＊／
        if （machine_is（celleb））｛
                __flush_tlb_pending（batch）；
                return;
        ｝
#endif ／＊ CONFIG_PREEMPT_RT ＊／
```

Celleb specific optimization (Cnt.)

- **Flushing a tlb needs expensive hypervisor calls.**



```
inetd-358   0D..4    3us : .flush_hash_range  (.__flush_tlb_pending)
inetd-358   0D..4    3us : .flush_hash_page  (.flush_hash_range)
inetd-358   0D..4    4us : .beat_lpar_hpte_invalidate  (.flush_hash_page)
inetd-358   0D..4    5us : .__spin_lock_irqsave  (.beat_lpar_hpte_invalidate)
inetd-358   0D..5    6us+: .beat_lpar_hpte_getword0  (.beat_lpar_hpte_invalidate)
inetd-358   0D..5   13us : .__spin_unlock_irqrestore  (.beat_lpar_hpte_invalidate)
inetd-358   0D..4   13us : .flush_hash_page  (.flush_hash_range)
inetd-358   0D..4   14us : .beat_lpar_hpte_invalidate  (.flush_hash_page)
inetd-358   0D..4   14us : .__spin_lock_irqsave  (.beat_lpar_hpte_invalidate)
inetd-358   0D..5   15us+: .beat_lpar_hpte_getword0  (.beat_lpar_hpte_invalidate)
inetd-358   0D..5   18us : .__spin_unlock_irqrestore  (.beat_lpar_hpte_invalidate)
inetd-358   0D..4   19us : .flush_hash_page  (.flush_hash_range)
inetd-358   0D..4   20us : .beat_lpar_hpte_invalidate  (.flush_hash_page)
```

  - Was very effective on linux-2.6.21
  - But not effective on linux-2.6.22 and later.

Future Work

- **Performance improvement**
  - Try to
    - Figure out reasons for these spikes using latency tracing mechanism.
    - (Back)Port TRACE_IRQFLAGS and STACKTRACE patches to RT.
    - Figure out how to put "same" load between different platforms
- **Real-world applications?**

Summary

- **gettimeofday() appears to be slow down for RT *on PowerPC.***
  - generic version of gettimeofday() needs reference to xtime (and locks) while original powerpc version of gettimeofday() does not.
- **sched_latency on vanilla shows the similar result with that of x86 non RT kernel (except for 1ms vs 4ms due to HZ differences).**
- **Maximum latency of cyclictest and sched_latency for rt1 are less than 90us. Seems fairly good.**
- **These Metrics could be a good references.**
- **Some more debugging/investigation is necessary to improve latencies.**

Thank you!

Let’s try cyclictest and IBM Test Cases!

TOSHIBA

Leading Innovation >>>