

# Appendix 1

## Latency on preemptible Real-Time Linux on **DCP-SH7780**

### Contents

---

1.1	Getting Started . . . . .	1
1.2	Rrtc Driver . . . . .	2
1.3	Test Program . . . . .	2
1.4	Hackbench Benchmark . . . . .	2
1.5	Evaluation of Jitter . . . . .	7
1.6	Conclusion . . . . .	9

---

### 1.1 Getting Started

The purpose of this document is to report latency and jitter of the preemptible Linux Kernel([linux-2.6.25.4](#)) running on our SH-board([DCP-SH7780](#)<sup>1</sup>) and to compare the latency of this standard Linux kernel with the latency of the kernel applying Ingo Molnar's Real-Time Preemption Patch<sup>2</sup>.

Those latencies are measured by Rrtc driver that uses unused timer channels included in the SH-CPU, and a test program that is waked up synchronously by the Rrtc driver which is interrupted periodically from the timer channels.

The test program that is waked up by an interrupt handler obtains the latency of the interrupt handler and current time from the Rrtc driver in order to calculate the latency of the program. The program writes the values to a file and finally prints the maximum value of the latencies.

---

<sup>1</sup>CompactPCI bus Board based on Renesas SH7780(SH-4A) processor at 400MHz and on-board memory 128M-byte DDR-SDRAM

<sup>2</sup>Refer to [http://source.mvista.com/linux\\_2.6\\_RT.html](http://source.mvista.com/linux_2.6_RT.html)

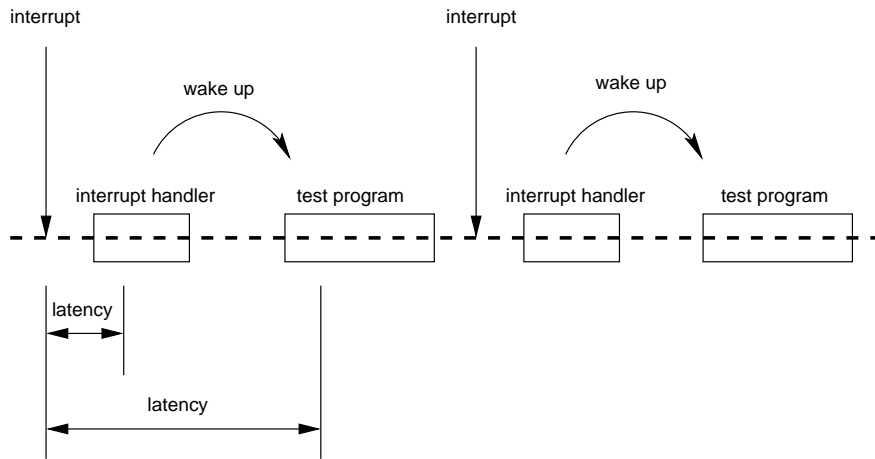


Figure 1: measurement of the latency

## 1.2 Rrtc Driver

The Rrtc driver uses two channels of the unused timer channels, one channel generates the interrupts periodically and the second channel is used to get current time in order to calculate the difference of the current time and an expected time with measuring both latencies(Refer to Fig.1).

## 1.3 Test Program

A test program is scheduled as real-time task by defining SCHED\_FIFO in order to measure the latencies. The program is disabled paging and locked in the memory by mlockall system call.

On the program starting with some options, the measurement of the latencies starts with periodic interrupts every 1 millisecond at a minimum time. If an option to write those latencies to a file is specified, the written data consists of three columns as follows.

```
0.000000 23.600000 2.400000
0.010000 13.679999 1.760000
0.020000 11.599999 1.520000
```

The first column shows time in seconds, the second column is a latency of the test program measuring it in microseconds, the third column is a latency of the interrupt handler of Rrtc driver in microseconds.

If any output file isn't specified, the test program records only the maximum latency instead of recording all latencies. If an output file is specified, these data should be written to a file on the memory disk in order to reduce the load on accessing the file.

## 1.4 Hackbench Benchmark

Rusty Russell's Hackbench benchmark is used to load scheduler of Linux kernel with many processes in order to measure the latency and jitter performance. If Hackbench runs with large number, the load will increase.

In these testing, Hackbench runs with a number 1, 2, 4, 6, 10, 15, 20, 30 and 40. Fig.2 shows the Time's value asserted by Hackbench in seconds on the y-axis.

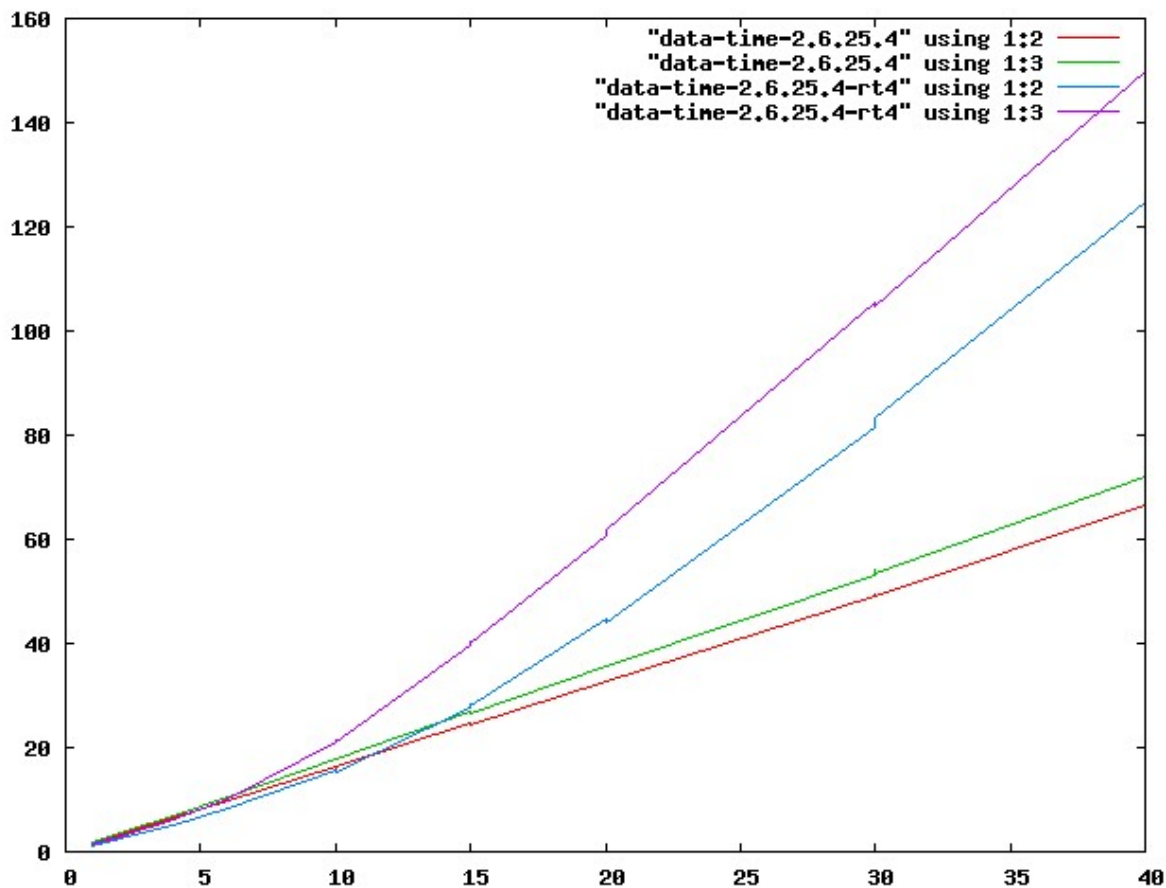


Figure 2: Times of Hackbench

Fig.2 plots following four kinds of Time's value.

**"data-time-2.6.25.4" using 1:2** without RT patch, without Rrttc driver

**"data-time-2.6.25.4" using 1:3** without RT patch, with Rrttc driver

**"data-time-2.6.25.4-rt4" using 1:2** with RT patch, without Rrttc driver

**"data-time-2.6.25.4-rt4" using 1:3** with RT patch, with Rrttc driver

These graphs show the influence of the load in each case of with or without RT patch and with or without Rrttc driver. The measurements of the latencies are taken every 1 millisecond and the latencies aren't written to the output file.

Fig.3 shows the range from 0 to 20 on the x-axis of the Fig.2. In the case of applying RT patch, it seems that the load increases if Hackbench runs with the value exceeded 10 and real-time scheduling of the test program becomes late because RT patch increases the cost for scheduling of the processes.

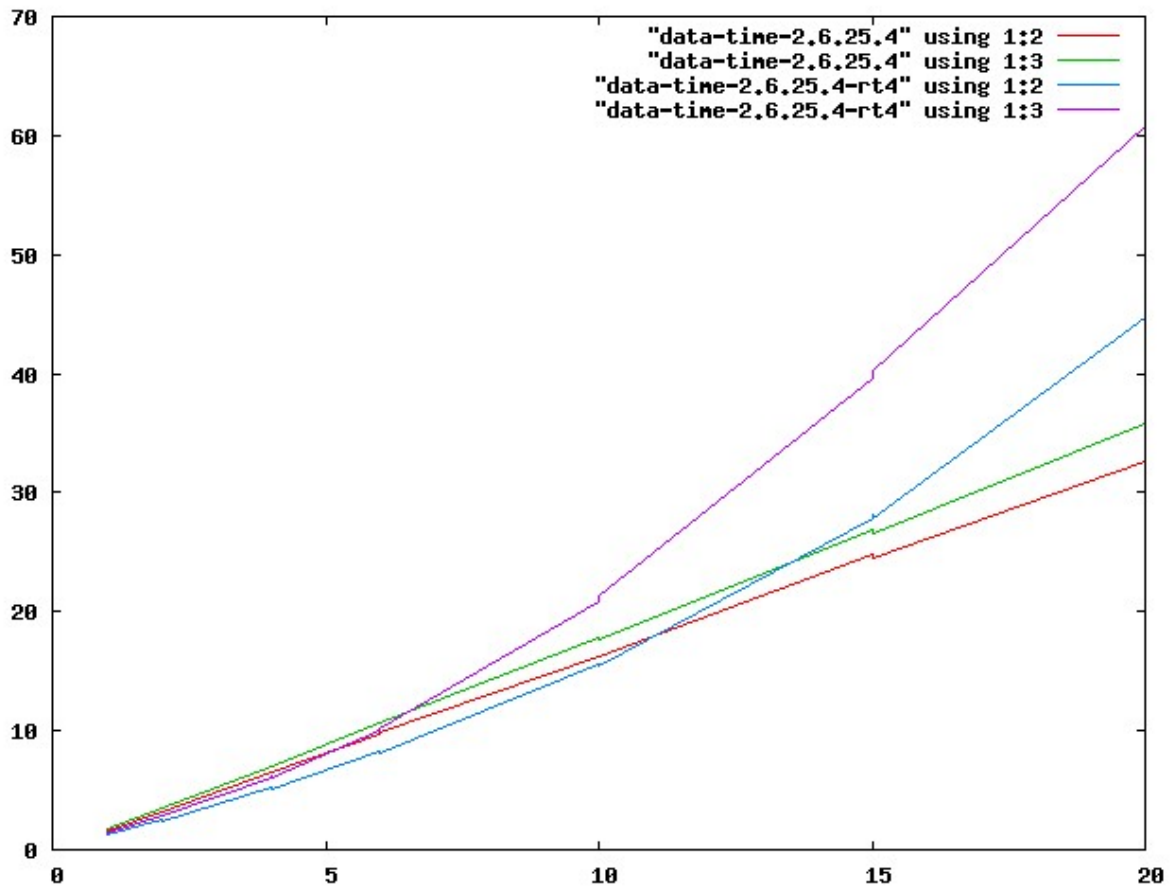


Figure 3: part of Times of Hackbench

**"data-latency-2.6.25.4" using 1:2** without RT patch, latency of test program

**"data-latency-2.6.25.4" using 1:3** without RT patch, latency of interrupt handler

**"data-latency-2.6.25.4-rt4" using 1:2** with RT patch, latency of test program

**"data-latency-2.6.25.4-rt4" using 1:3** with RT patch, latency of interrupt handler

Fig.4 shows the values of following four kinds of maximum latency in microseconds on the y-axis. The measurements of the latencies are taken every 1 millisecond and the latencies aren't written to the output file.

measuring

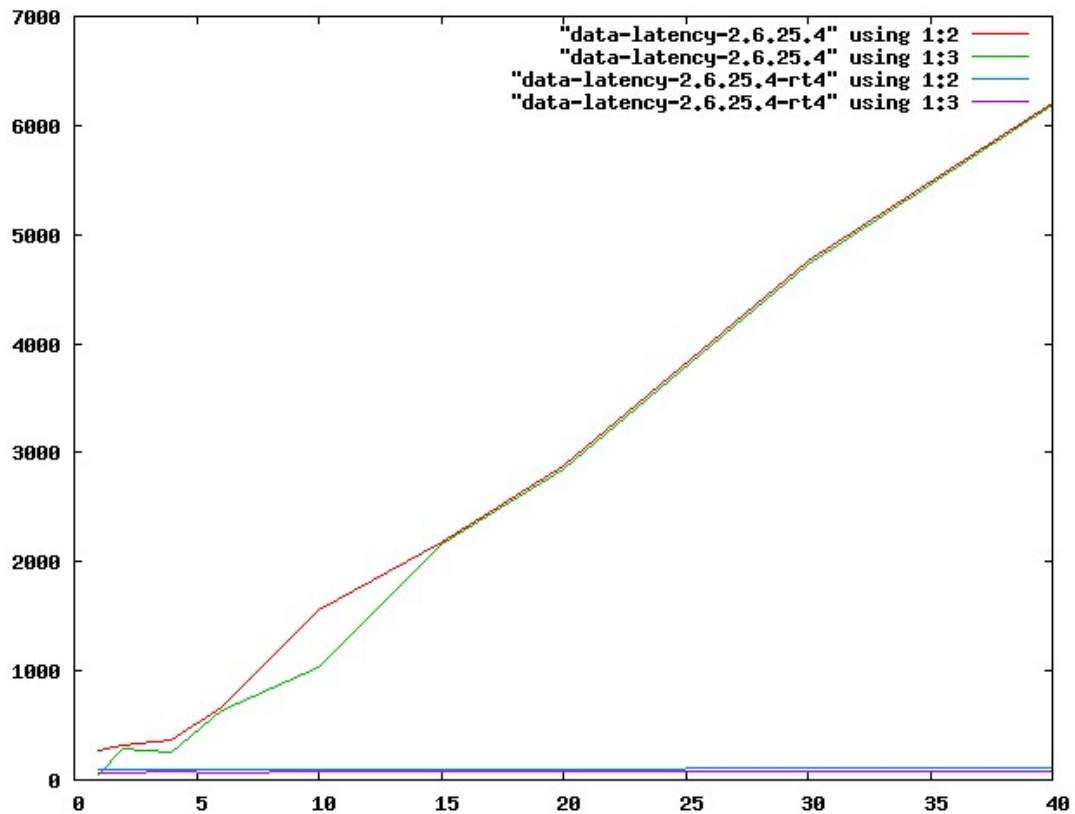


Figure 4: maximum latency

**"data-latency-2.6.25.4" using 1:2** without RT patch, latency of test program

**"data-latency-2.6.25.4" using 1:3** without RT patch, latency of interrupt handler

**"data-latency-2.6.25.4-rt4" using 1:2** with RT patch, latency of process

**"data-latency-2.6.25.4-rt4" using 1:3** with RT patch, latency of interrupt handler

In case without RT patch, it causes the interrupts to delay because all interrupts are disabled while many processes are loaded and scheduled.

Fig.5 shows only about applying RT patch on Fig.4.

In case with RT patch, the latencies are low even if many processes are loaded by Hackbench.

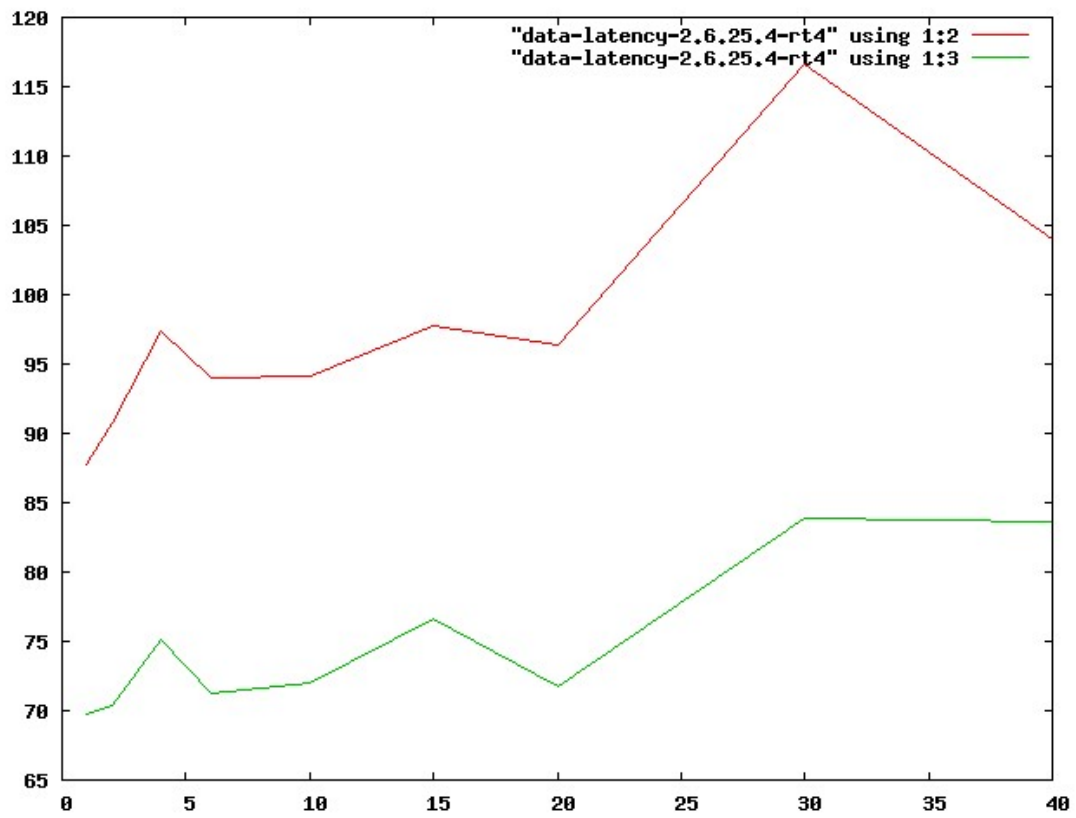


Figure 5: Maximum Latency With RT Patch

”data-latency-2.6.25.4-rt4” using 1:2 with RT patch, latency of test program

”data-latency-2.6.25.4-rt4” using 1:3 with RT patch, latency of interrupt handler

## 1.5 Evaluation of Jitter

The jitter is shown by plotting all latencies. Real-time applications ask low latency and low jitter.

Fig.6 shows a plot of the following two kind of data, Hackbench runs with 30 and the measurements of the latencies are taken every 10 milliseconds and the latencies are written to a output file on memory-disk, in seconds on the x-axis.

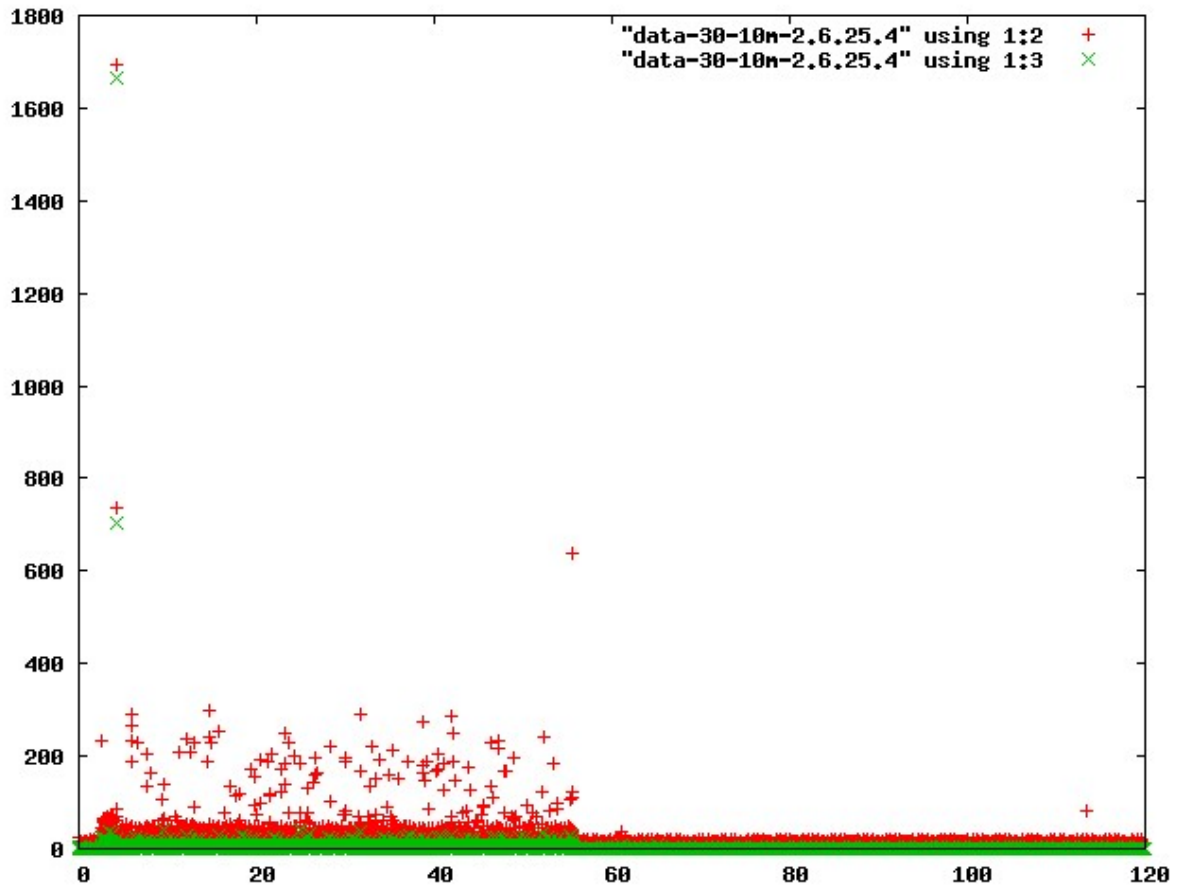


Figure 6: Latencies of the Standard Linux

"data-30-10m-2.6.25.4" using 1:2 without RT patch, latency of test program

"data-30-10m-2.6.25.4" using 1:3 without RT patch, latency of interrupt handler

Fig.7 shows round 300 microsecond of the y-axis on Fig.6.

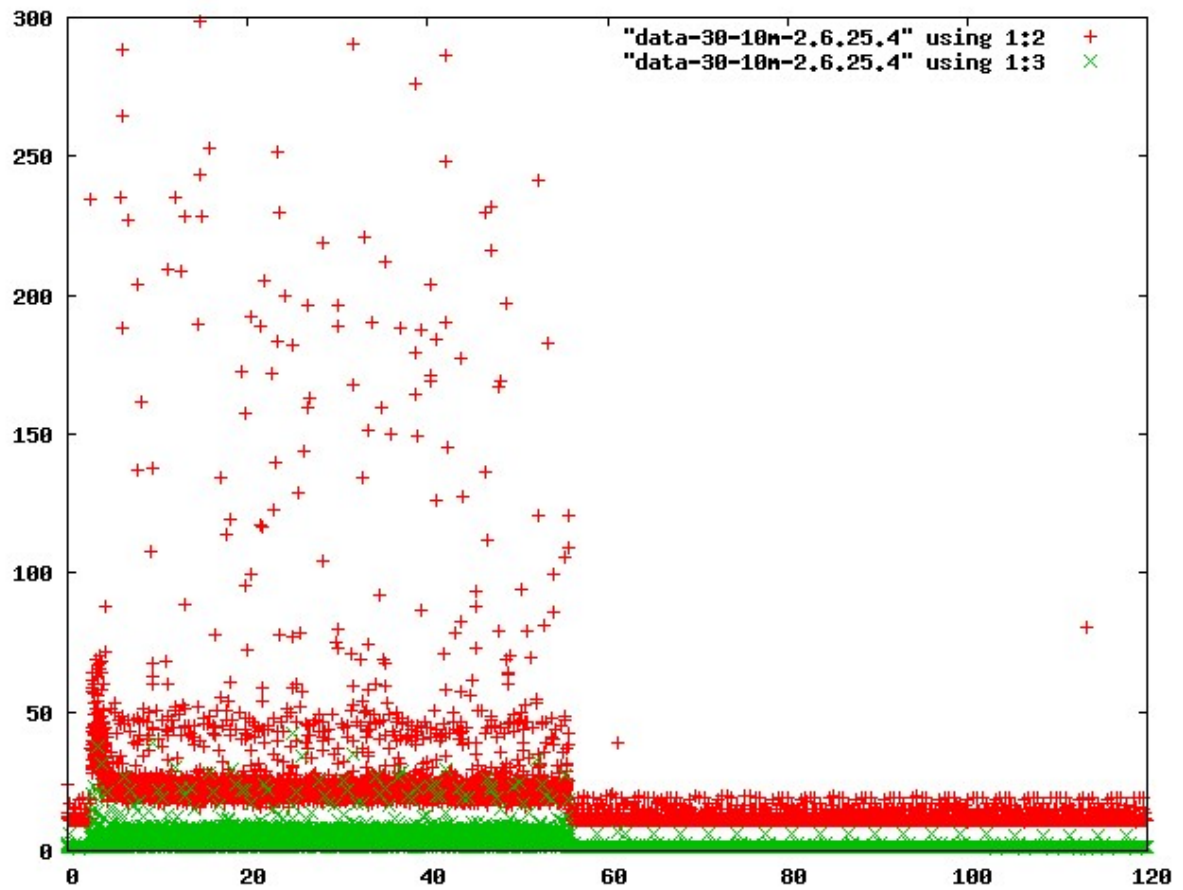


Figure 7: Part of Latencies of the Standard Linux

"data-30-10m-2.6.25.4" using 1:2 without RT patch, latency of test program

"data-30-10m-2.6.25.4" using 1:3 without RT patch, latency of interrupt handler

Fig8 shows a plot of the following two kind of data, Hackbench runs with 30 and the measurements of the latencies are taken every 10 milliseconds and the latencies are written to a output file on memory-disk, in seconds on the x-axis.

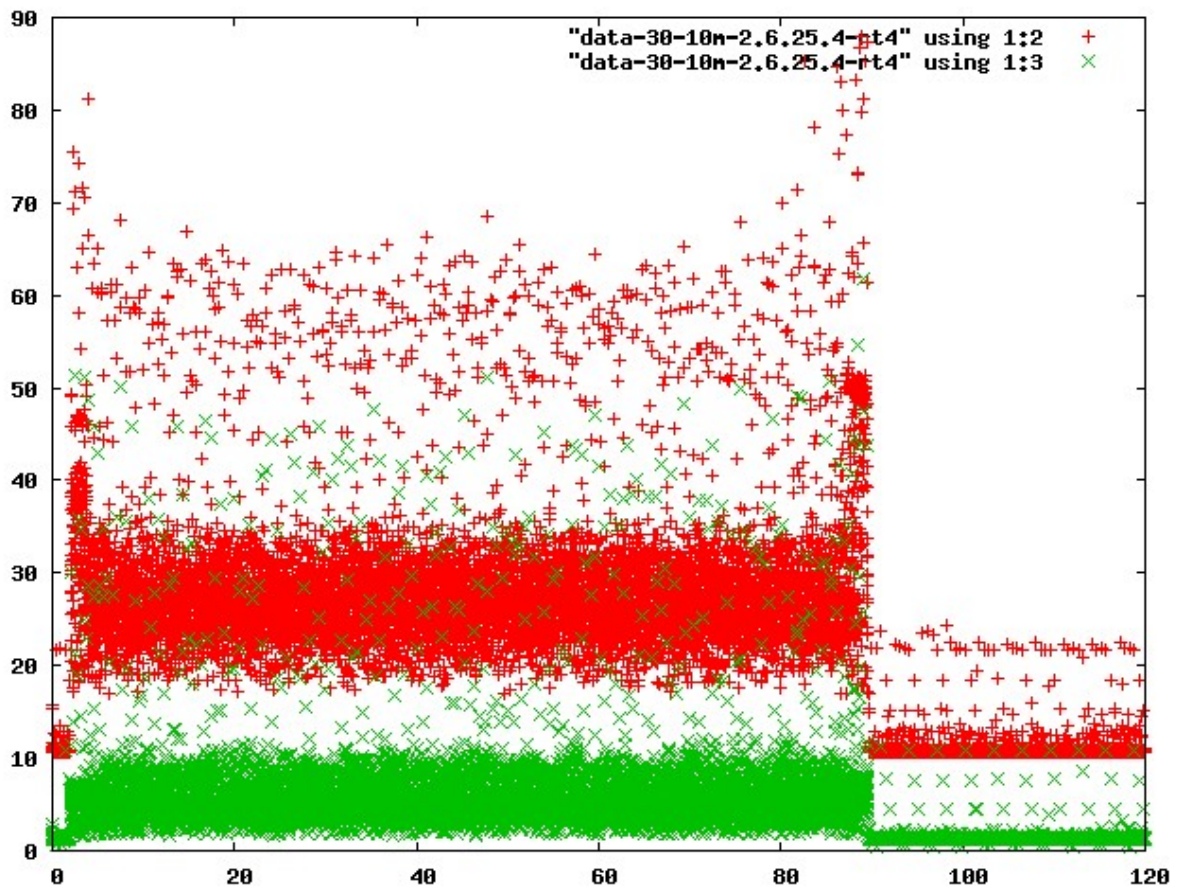


Figure 8: Latencies with RT Patch

"data-30-10m-2.6.25.4-rt4" using 1:2 with RT patch, latency of test program

"data-30-10m-2.6.25.4-rt4" using 1:3 with RT patch, latency of interrupt handler

## 1.6 Conclusion

We can get low latency and low jitter performance by applying RT patch, however, the speed of the benchmark test becomes slow.

If we select applying RT patch to standard Linux kernel, we need to measure the performance of latency, jitter and speed using a driver like as Rrttc driver.

Jul-02-2008