Effects of Clock Resolution on the Scheduling of Interactive and Soft Real-Time Processes

by Yoav Elston, Dan Tsafir, Dror G. Feitelson

Presented by James Volpe

Key Points

- Modern interactive applications not supported in commodity operating systems
- Instead of specialized APIs, tune commodity operating systems
- Clock interrupt rates unchanged last 30 years
- Increasing clock interrupt rate can help
- Additional overhead is acceptable

Real-time Problem

- Xine MPEG player
- Short clip of 500 frames

![Graph showing desired frames per second vs. time (seconds) for Xine MPEG player]

Process Scheduling

- Single CPU
- Execute for every interrupt
- Increase jiffies
- Bill process using CPU
- Adjust priorities
- Check task priorities
- etc.

Real-time Problem Up Close

- Commodity operating systems use priority-based scheduling
- Static and dynamic priorities
- High CPU usage, the lower the priority
- Conflict with modern multimedia applications
Related Work

- RT-Linux, one-shot timers, soft timers, firm timers and priority adjustments
- Require special APIs and/or non-trivial modifications to the system
- Why not increase the clock interrupt rate and see what happens?

Motivation

- 100 Hz clock interrupt rates
  - 10 MHz CPU 100,000 instructions/interrupt
  - 1 GHz CPU 10,000,000 instructions/interrupt
- Clock interrupt rate has become too coarse
  - Two orders of magnitude of additional instructions per interrupt

Benefits of High Tick Rate

- Take advantage of today’s faster processors
- Better timing for meeting deadlines
- Accurate billing of processes

Test Platform

- Pentium 90 to Pentium-IV 2.4 GHz
- Measurements done on 664 MHz Pentium III
- Linux kernel 2.4.8 (RedHat 7.0)
- Klogger used for logging scheduling related events: context switching, recalculation of priorities, forks, execs
- Workload from Emacs, Xine, Quake 3, CPU-bound processes
Billing Advantage

<table>
<thead>
<tr>
<th>Application</th>
<th>Billing ratio @10Hz</th>
<th>Billing ratio @100Hz</th>
<th>Billing ratio @1000Hz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>1.0342</td>
<td>1.0256</td>
<td>1.0392</td>
</tr>
<tr>
<td>Server</td>
<td>1.0310</td>
<td>1.0292</td>
<td>1.0337</td>
</tr>
<tr>
<td>CPU-bound</td>
<td>1.0271</td>
<td>1.0294</td>
<td>0.9739</td>
</tr>
<tr>
<td>CPU+Quake</td>
<td>1.0332</td>
<td>1.0296</td>
<td>0.8711</td>
</tr>
</tbody>
</table>

Table 3. Scheduler billing success rate

The Cost Is Overhead

- More clock interrupts increases overhead
- Time needed to process interrupts
- Higher number of context switches
- Operating systems become less efficient as clock rate increases

Clock Interrupt Overhead

- Do operating systems become as fast as hardware?

<table>
<thead>
<tr>
<th>Processor</th>
<th>Clock</th>
<th>Without Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>P-90</td>
<td>1.146</td>
<td>0.54</td>
</tr>
<tr>
<td>PP-200</td>
<td>3.056</td>
<td>2.12</td>
</tr>
<tr>
<td>PP-300</td>
<td>3.039</td>
<td>2.12</td>
</tr>
<tr>
<td>P6-450</td>
<td>3.088</td>
<td>2.12</td>
</tr>
<tr>
<td>P6-660</td>
<td>3.120</td>
<td>2.12</td>
</tr>
<tr>
<td>P6-800</td>
<td>3.135</td>
<td>2.12</td>
</tr>
<tr>
<td>P6-900</td>
<td>3.147</td>
<td>2.12</td>
</tr>
</tbody>
</table>

Table 1. Interrupt processing overheads on different processor generations

Clock Interrupt Overhead Cont.

- Overhead from sorting an array with 50ms quantum

<table>
<thead>
<tr>
<th>Processor</th>
<th>Context switch</th>
<th>Cache hit ratio</th>
<th>Trap</th>
</tr>
</thead>
<tbody>
<tr>
<td>P-90</td>
<td>131463</td>
<td>3134306</td>
<td>1.93</td>
</tr>
<tr>
<td>PP-200</td>
<td>256320</td>
<td>2536356</td>
<td>1.93</td>
</tr>
<tr>
<td>PP-300</td>
<td>256320</td>
<td>2536356</td>
<td>1.93</td>
</tr>
<tr>
<td>P6-450</td>
<td>2536356</td>
<td>2536356</td>
<td>1.93</td>
</tr>
<tr>
<td>P6-660</td>
<td>2536356</td>
<td>2536356</td>
<td>1.93</td>
</tr>
<tr>
<td>P6-800</td>
<td>2536356</td>
<td>2536356</td>
<td>1.93</td>
</tr>
<tr>
<td>P6-900</td>
<td>2536356</td>
<td>2536356</td>
<td>1.93</td>
</tr>
<tr>
<td>P6-1100</td>
<td>2536356</td>
<td>2536356</td>
<td>1.93</td>
</tr>
<tr>
<td>P6-1300</td>
<td>2536356</td>
<td>2536356</td>
<td>1.93</td>
</tr>
</tbody>
</table>

Table 2. Other overheads on different processor generations
Clock Interrupt Overhead Cont.

- Overhead from sorting an array with quantum equal to 5 ticks

Conclusion

- Increasing clock interrupt rate to 1000 Hz achieves timing and billing advantages
- Overhead is minimal
- Suggest making a settable parameter instead of compiled constant

Evaluation

- Results are promising
- Production level example would strengthen argument
- Changing clock interrupt rate in Linux not an end user job
- Testing is needed for implementation

Questions / Discussion