The On-going Evolutions of Power Management in Xen

Kevin Tian
Open Source Technology Center (OTC)
Intel Cooperation
Agenda

• Brief history
• Evolutions of Idle power management
• Evolutions of run-time power management
• Tools
• Experimental data about Xen power efficiency
• Power impact from VM
**Brief History**

- Jul. 2007: Host S3
- Sep. 2007: Xen 3.1 released
- Jan. 2008: Xen 3.2 released
- May. 2008: Preliminary cpufreq and cpuidle support in Xen
- Jun. 2008: Mature deep C-states support
- Aug. 2008: Xen 3.3 released

Enhanced green computing
Improved stability
Better usability

Xen 3.1

Sep. 2007
Dom0 controlled freq/vol scaling

May. 2008
Preliminary cpufreq and cpuidle support in Xen

Aug. 2008
Xen 3.3 released

...
Evolutions of Idle power management
(Xen cpuidle)
Enhanced C-states support

‘noPM’ has both cpuidle and cpufreq disabled, and vice versa for other two cases. Compared to ‘C3’, noC3 has maximum C-state limited to C2

For idle watt, lower value means greener. For SPECpower score, higher value indicates more power efficient
TSC freeze

Software compensation according to elapsed counter of platform timer

Unsynchronized TSC
- Time went backwards warning
- Lots of lost ticks
- Faster ToD

Fluctuating TSC scale factor

Xen system time skew

Platform counter
- Always running TSC
- Hardware enhancement to have TSC never stopped (e.g., by Intel Core-i7)

TSC save/restore

CPU0
- Ideal TSC
- Actual TSC

CPU1
- TSC freeze

Timer

Percpu platform to TSC scale

Global platform to TSC scale

TSC

Platform counter

Hardware enhancement to have TSC never stopped (e.g., by Intel Core-i7)
APIC timer freeze

**Dom0**

- ACPI Parser
- External Control

**Registration**

- Hypercall
- Ladder
- CpuIdle driver
- Mwait/IO
- C1
- C2
- Deep
- C1E
- PIT/HPET broadcast

**Solution**

- Reprogram local APIC count-down timer
- Scan/execute expired timers
- Timer softirq handler
- Local APIC ISR

**Timer heap**

- Nearest deadline
- Interrupt when count down to zero (Delayed)

**Use platform timer (PIT/HPET) to carry nearest deadline, when APIC timer is halted in deep C-states:**

- Broadcast
- MSI based HPET interrupt
- Always running APIC timer

**Solution**

- Broadcast
- MSI based HPET interrupt
- Always running APIC timer

**Use platform timer (PIT/HPET) to carry nearest deadline, when APIC timer is halted in deep C-states:**

- Required since number of platform timer source is less than number of CPUs
- Will come soon to have more platform timer sources with reduced broadcast traffic
- Will be supported in new CPU soon
Menu governor

Expected minimal residency at Cn
Expected minimal residency at Cn+1

Promotion if continuous N Cn residencies > expectation
Demotion if current Cn+1 residency < expectation

less idle watt consumed (-5.2%)
Higher SPECpower score (+1.6%)
(HVM WinWPsp1)

To be further tuned!
Frequent C-state Entry/exit may Instead consume More power

For each timer, it accepts a range for expiration now:

\[[\text{expiration}, \text{expiration} + \text{timer\_slop}]\]

\((\text{default } 50\text{us for timer\_slop})\)

Overlapping ranges can be merged to reduce timer interrupt count
Range timer effect

One UP HVM RHEL5u1

<table>
<thead>
<tr>
<th>Timer interval</th>
<th>Idle (watt)</th>
<th>SPECpower (score)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50us</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>1ms</td>
<td>92.50%</td>
<td>101.20%</td>
</tr>
</tbody>
</table>

Multiple idle UP HVM RHEL5u1

[Graph showing timer interrupt rates for 50us and 1ms intervals]

Collected on a two-cores mobile platform
Current picture

Dom0

ACPI Parser

External Control

Xenpm

Registration

Hypercall
Statistics
Menu

CpuIdle driver

Deep C-states

C1
C2
C1E

PIT/HPET broadcast
TSC save/restore
Always running TSC

Power Aware

Schedulers

Enter Idle

Halt (C1)

Dynamic Tick
Timekeeping

Range timer
Timer

Xentrace

Xen

Software and Solutions Group
Evolutions of run-time power management
(Xen cpufreq)
Tools
Retrieve run-time statistics about Xen cpuidle and cpufreq

Apply user policy on exposed control knobs of Xen cpufreq (governor, set freq, etc)

More capabilities to be added later, e.g. profile...

Log every state change for Xen cpuidle and cpufreq:

CPU0 391365842416 (+ 21204) cpu_idle_entry [ idle to state 2 ]
CPU0 391375951050 (+10108634) cpu_idle_exit [ return from state 2 ]

Raw data could be further processed by other scripts
Experimental data about Xen power efficiency
• All data shown in this section:
  – For reference only and not guaranteed
  – On a two-core mobile platform, with one HVM guest created
    • Server consolidation effect with multiple VMs/workloads are in progress
• Improvement when Xen cpuidle and cpufreq are enabled
  – SPECpower score is normalized (100% noPM score is 1000 ssj_ops / watt)
  – Similarly, consumed watt is also normalized (idle noPM watt is 10w)
• Below is a more attracting comparison
  – Native WinXPsp1’s idle watt and SPECpower score are both normalized to 100% as the base

<table>
<thead>
<tr>
<th>Normalized percentage (%)</th>
<th>idle (Watt)</th>
<th>SPECpower (ssj_ops/watt)</th>
</tr>
</thead>
<tbody>
<tr>
<td>native xpsp1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>native rhel5u1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>xen xpsp1 hvm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>kvm xpsp1 hvm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>xen rhel5u1 hvm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>kvm rhel5u1 hvm</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

HZ=1000 in RHEL5u1 incurs high timer interrupt

Similar Idle power consumption for Xen and KVM

Xen is slightly more power efficient than KVM
Power impact from VM
Bad App eats power like in native!

Kernel implementation matters!

Green features in VM helps!

VMM shouldn’t Be only Blamed!
• VMM shouldn’t be blamed as only reason for high power consumption!
• A ‘bad’ VM could eat power
  – Just like what ‘bad’ application could do in a native OS
  – Cause high break events (e.g. timer interrupts) with short C-state residency
• Which parts in VM could draw high power?
  – ‘Bad’ applications hurt just as what they could on native
  – How guest OS is implemented also matters
    • Periodic tick frequency – HZ
    • Timer usage in drivers
    • Time sub-system implementation
    • …
• Green guest OS wins!
  – Smaller HZ, idle tickles or fully dynamic tick, range timer, etc.
Idle power consumption

![Graph showing normalized watt and average residency (ms) across different conditions.](image-url)

- **Legend**:
  - **Idle power consumption**
  - **Average C-state residency**

- **Conditions**:
  - bare Dom0
  - PV
  - HVM
  - Winxp
  - HZ=1000
  - HZ=250
  - HZ=100
  - tickless

- **Observations**:
  - Idle power consumption and average residency vary across conditions.
  - Certain conditions are marked as 'Bad guest' and others as 'Green'.

- **Notes**:
  - The graph visually compares the energy consumption and residency times for different system configurations.
Power efficiency

![Normalized score (ssj_ops / watt) vs SPECpower score for different configurations: bare Dom0, PV, HVM Winxp, HZ=1000, HZ=100, tickless. The graph shows the power efficiency comparison across these configurations.](image-url)
Legal Information

- INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY RELATING TO SALE AND/OR USE OF INTEL PRODUCTS, INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT.

- Intel may make changes to specifications, product descriptions, and plans at any time, without notice.

- All dates provided are subject to change without notice.

- Intel is a trademark of Intel Corporation in the U.S. and other countries.

- *Other names and brands may be claimed as the property of others.

- Copyright © 2007, Intel Corporation. All rights are protected.