Balancing Power and Performance in the Linux Kernel

Kristen Accardi
Power Management basics

P-states on Intel platforms

Limitations of OS p-state selection

Hardware P-states (HWP)
Degrees of Idleness

- Runtime
- Idle
- Suspended
- Off
<table>
<thead>
<tr>
<th>Suspend</th>
<th>Runtime Idle</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Initiated</td>
<td>Opportunistic</td>
</tr>
<tr>
<td>User tasks</td>
<td>User tasks</td>
</tr>
<tr>
<td>frozen</td>
<td>scheduled</td>
</tr>
<tr>
<td>Devices forced</td>
<td>Opportunistic</td>
</tr>
<tr>
<td>idle</td>
<td>device idle</td>
</tr>
<tr>
<td>d-states</td>
<td>d-states</td>
</tr>
<tr>
<td>S-states or Idle</td>
<td></td>
</tr>
<tr>
<td>S3/S0iX/MWAIT</td>
<td>CPU C-states</td>
</tr>
<tr>
<td></td>
<td>(S0iX)</td>
</tr>
</tbody>
</table>
Active Power Management
Active Power Management in Linux – Dim the Lights

- CPU active power management (cpufreq)
  - Aka P-states or DVFS

- Device Active Power management
  - Some device support such as PCIe ASPM
  - Device specific how to activate

- GPU does it's own thing
P-States $\neq$ Frequency
P-States != Power
How do we pick the right p-state?

• Governors reflect user policy decision
  – intel_pstate supports only “powersave” and “performance” policies. Other drivers support more policies.
  – intel_pstate is actually a governor and hw driver all in one, whereas traditionally the governor is separate from the hw driver.

• intel_pstate “performance” policy always picks the highest p-state
  – Race to Halt – or just don't care about energy

• intel_pstate “powersave” policy attempts to balance performance with energy savings

• The driver looks at utilization and capacity to determine whether to increase or decrease the p-state. This is similar to many other governors.
P-state basics

P-states on Intel platforms

Limitations of OS p-state selection

Hardware P-states (HWP)
P1 - Pn is Guaranteed
P0 - P1 is turbo
Pn - LFM for Thermal
How Turbo works (Intel Speed Step®)
Ivy Bridge Platform

- Core VR variable voltage 0V-1.2V
- Graphics VR variable voltage 0V-1.2V
- PLL VR 1.8V
- System agent VR

Haswell Platform

- VccIn 0V-1.8V
- Haswell Processor
  - FIVR VRs:
    - Vccsa
    - Vccio
    - Vccioa
    - VccCore 0
    - VccCore 1
    - VccCore 2
    - VccCore 3
    - VccCache
    - Graphics0
    - Graphics1
    - VccEDRAM
    - VccOPIO

- DDRx
- DDR VR
Hardware Coordination of P-states

• Cores which share the same voltage domain vote for a p-state
Hardware Coordination of P-states

• Cores which share the same voltage domain vote for a p-state
• The highest p-state for each core wins
Hardware Coordination of P-states

• Cores which share the same voltage domain vote for a p-state
• The highest p-state for each core wins
• APERF/MPERF must be used to see what p-state was granted
Hardware Coordination of P-states

• Cores which share the same voltage domain vote for a p-state
• The highest p-state for each core wins
• APERF/MPERF must be used to see what p-state was granted
• acpi_cpufreq lies!!!!
P-state basics

P-states on Intel platforms

Limitations of OS p-state selection

Hardware P-states (HWP)
Limitations of OS P-state selection

- Capacity/Utilization is insufficient for determining whether to scale
Limitations of OS P-state selection

- Capacity/Utilization is insufficient for determining whether to scale
- Sample rate may cause incorrect utilization calculation
Limitations of OS P-state selection

• Capacity/Utilization is insufficient for determining whether to scale
• Sample rate may cause incorrect utilization calculation
• Scaling benefits unclear
P-state basics

P-states on Intel platforms

Limitations of OS p-state selection

Hardware P-states (HWP)
Intel® Speed Shift Technology (HWP)

- Most Efficient Frequency is Calculated at Runtime (Pe)
  Depends on system & workload

- EPP is Energy Performance Preference – will dictate how aggressive the algorithm (Pa)
  Depends on system, workload, OS

- Algorithm will operate between Pa and Pe

EDA LFM

Highest Frequency

EPP Frequency

Guaranteed Frequency

Most Efficient Frequency

Lowest Frequency

P0 – 1 core

T-States
<table>
<thead>
<tr>
<th>Address</th>
<th>Architectural</th>
<th>Register Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>770H</td>
<td>Y</td>
<td>IA32_PM_ENABLE</td>
<td>Enable/Disable HWP.</td>
</tr>
<tr>
<td>771H</td>
<td>Y</td>
<td>IA32_HWP_CAPABILITIES</td>
<td>Enumerates the HWP performance range (static and dynamic).</td>
</tr>
<tr>
<td>772H</td>
<td>Y</td>
<td>IA32_HWP_REQUEST_PKG</td>
<td>Conveys OSPM's control hints (Min, Max, Activity Window, Energy Performance Preference, Desired) for all logical processor in the physical package.</td>
</tr>
<tr>
<td>773H</td>
<td>Y</td>
<td>IA32_HWP_INTERRUPT</td>
<td>Controls HWP native interrupt generation (Guaranteed Performance changes, excursions).</td>
</tr>
<tr>
<td>774H</td>
<td>Y</td>
<td>IA32_HWP_REQUEST</td>
<td>Conveys OSPM's control hints (Min, Max, Activity Window, Energy Performance Preference, Desired) for a single logical processor.</td>
</tr>
<tr>
<td>777H</td>
<td>Y</td>
<td>IA32_HWP_STATUS</td>
<td>Status bits indicating changes to Guaranteed Performance and excursions to Minimum Performance.</td>
</tr>
<tr>
<td>19CH</td>
<td>Y</td>
<td>IA32_THERM_STATUS[bits 15:12]</td>
<td>Conveys reasons for performance excursions</td>
</tr>
<tr>
<td>64EH</td>
<td>N</td>
<td>MSR_PPERF</td>
<td>Productive Performance Count.</td>
</tr>
</tbody>
</table>
Figure 14-7. IA32_HWP_REQUEST Register
Linux Implementation

- intel_pstate driver checks cpuflags for support
- Enabled by default for whitelisted CPUs
- Autonomous mode only
- No EPP exposed today
- Min and Max pstate can be requested via min and max perf_pct sysfs files
References

2. http://events.linuxfoundation.org/sites/events/files/slides/LinuxCon_Japan_2015_idle_injection1_0.pdf