CPU frequency and voltage scaling code in the Linux(TM) kernel

Linux CPUFreq

CPUFreq Core

Dominik Brodowski <linux@brodo.de>
David Kimdon <dwhedon@debian.org>

Clock scaling allows you to change the clock speed of the CPUs on the fly. This is a nice method to save battery power, because the lower the clock speed, the less power the CPU consumes.

Contents:

1. General Information

The CPUFreq core code is located in drivers/cpufreq/cpufreq.c. This cpufreq code offers a standardized interface for the CPUFreq architecture drivers (those pieces of code that do actual frequency transitions), as well as to "notifiers". These are device drivers or other part of the kernel that need to be informed of policy changes (ex. thermal modules like ACPI) or of all frequency changes (ex. timing code) or even need to force certain speed limits (like LCD drivers on ARM architecture). Additionally, the kernel "constant" loops_per_jiffy is updated on frequency changes here.

Reference counting is done by cpufreq_get_cpu and cpufreq_put_cpu, which make sure that the cpufreq processor driver is correctly registered with the core, and will not be unloaded until cpufreq_put_cpu is called.

2. CPUFreq notifiers

CPUFreq notifiers conform to the standard kernel notifier interface. See linux/include/linux/notifier.h for details on notifiers.

There are two different CPUFreq notifiers - policy notifiers and transition notifiers.

2.1 CPUFreq policy notifiers

These are notified when a new policy is intended to be set. Each

CPUFreq policy notifier is called three times for a policy transition:

1.) During CPUFREQ_ADJUST all CPUFreq notifiers may change the limit if
    they see a need for this - may it be thermal considerations or
    hardware limitations.

2.) During CPUFREQ_INCOMPATIBLE only changes may be done in order to avoid
    hardware failure.

3.) And during CPUFREQ_NOTIFY all notifiers are informed of the new policy
   - if two hardware drivers failed to agree on a new policy before this
   stage, the incompatible hardware shall be shut down, and the user
   informed of this.

The phase is specified in the second argument to the notifier.

The third argument, a void *pointer, points to a struct cpufreq_policy consisting of five values: cpu, min, max, policy and max_cpu_freq. min and max are the lower and upper frequencies (in kHz) of the new policy, policy the new policy, cpu the number of the affected CPU; and max_cpu_freq the maximum supported CPU frequency. This value is given for informational purposes only.

2.2 CPUFreq transition notifiers

These are notified twice when the CPUfreq driver switches the CPU core frequency and this change has any external implications.

The second argument specifies the phase - CPUFREQ_PRECHANGE or CPUFREQ_POSTCHANGE.

The third argument is a struct cpufreq_freqs with the following values:

```
cpu     - number of the affected CPU
old     - old frequency
new     - new frequency
```

If the cpufreq core detects the frequency has changed while the system was suspended, these notifiers are called with CPUFREQ_RESUMECHANGE as second argument.