user-guide.txt
CPU frequency and voltage scaling code in the Linux(TM) kernel

L i n u x C P U F r e q

U S E R G U I D E

Dominik Brodowski <linux@brodo.de>

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. Supported Architectures and Processors

1.1 ARM

The following ARM processors are supported by cpufreq:

ARM Integrator
ARM-SA1100
ARM-SA1110
Intel PXA

1.2 x86

The following processors for the x86 architecture are supported by cpufreq:

AMD Elan - SC400, SC410

AMD mobile K6-2+
AMD mobile K6-3+
AMD mobile Duron
AMD mobile Athlon
AMD Opteron
AMD Athlon 64
Cyrix Media GXm
Intel mobile PIII and Intel mobile PIII-M on certain chipsets
Intel Pentium 4, Intel Xeon
Intel Pentium M (Centrino)
National Semiconductors Geode GX
Transmeta Crusoe
Transmeta Efficeon
VIA Cyrix 3 / C3
various processors on some ACPI 2.0-compatible systems [*]

[*] Only if "ACPI Processor Performance States" are available
to the ACPI<->BIOS interface.

1.3 sparc64

The following processors for the sparc64 architecture are supported by cpufreq:

UltraSPARC-III

1.4 ppc

Several "PowerBook" and "iBook2" notebooks are supported.

1.5 SuperH

All SuperH processors supporting rate rounding through the clock framework are supported by cpufreq.

1.6 Blackfin

The following Blackfin processors are supported by cpufreq:

BF522, BF523, BF524, BF525, BF526, BF527, Rev 0.1 or higher
BF531, BF532, BF533, Rev 0.3 or higher
BF534, BF536, BF537, Rev 0.2 or higher
BF561, Rev 0.3 or higher
BF542, BF544, BF547, BF548, BF549, Rev 0.1 or higher

2. "Policy" / "Governor" ?

Some CPU frequency scaling-capable processor switch between various frequencies and operating voltages "on the fly" without any kernel or user involvement. This guarantees very fast switching to a frequency which is high enough to serve the user's needs, but low enough to save power.

2.1 Policy

On these systems, all you can do is select the lower and upper frequency limit as well as whether you want more aggressive power-saving or more instantly available processing power.

2.2 Governor

On all other cpufreq implementations, these boundaries still need to be set. Then, a "governor" must be selected. Such a "governor" decides what speed the processor shall run within the boundaries. One such "governor" is the "userspace" governor. This one allows the user - or a yet-to-implement userspace program - to decide what specific speed the processor shall run at.

3. How to change the CPU cpufreq policy and/or speed

3.1 Preferred Interface: sysfs

The preferred interface is located in the sysfs filesystem. If you mounted it at /sys, the cpufreq interface is located in a subdirectory "cpufreq" within the cpu-device directory (e.g. /sys/devices/system/cpu/cpu0/cpufreq/ for the first CPU).

cpuinfo_min_freq :	this file shows the minimum operating frequency the processor can run at(in kHz)
cpuinfo_max_freq :	this file shows the maximum operating frequency the processor can run at(in kHz)
cpuinfo_transition_latency	The time it takes on this CPU to switch between two frequencies in nano seconds. If unknown or known to be that high that the driver does not work with the ondemand governor, -1 (CPUFREQ_ETERNAL) will be returned. Using this information can be useful to choose an appropriate polling frequency for a kernel governor or userspace daemon. Make sure to not switch the frequency too often resulting in performance loss.
scaling_driver :	this file shows what cpufreq driver is used to set the frequency on this CPU

scaling_available_governors : this file shows the CPUfreq governors available in this kernel. You can see the currently activated governor in

scaling_governor, and by "echoing" the name of another governor you can change it. Please note that some governors won't load - they only work on some specific architectures or processors.

cpuinfo_cur_freq : Current frequency of the CPU as obtained from the hardware, in KHz. This is the frequency the CPU actually runs at.

scaling_available_frequencies : List of available frequencies, in KHz.

scaling_min_freq and scaling_max_freq show the current "policy limits" (in kHz). By echoing new values into these files, you can change these limits. NOTE: when setting a policy you need to first set scaling_max_freq, then scaling_min_freq.

affected_cpus : List of CPUs that require software coordination of frequency.

related_cpus : List of CPUs that need some sort of frequency coordination, whether software or hardware.

scaling_driver : Hardware driver for cpufreq.

scaling_cur_freq : Current frequency of the CPU as determined by the governor and cpufreq core, in KHz. This is the frequency the kernel thinks the CPU runs at.

bios_limit : If the BIOS tells the OS to limit a CPU to lower frequencies, the user can read out the maximum available frequency from this file. This typically can happen through (often not intended) BIOS settings, restrictions triggered through a service processor or other BIOS/HW based implementations. This does not cover thermal ACPI limitations which can be detected through the generic thermal driver.

If you have selected the "userspace" governor which allows you to set the CPU operating frequency to a specific value, you can read out the current frequency in

scaling_setspeed. By "echoing" a new frequency into this you can change the speed of the CPU, but only within the limits of scaling_min_freq and scaling_max_freq.