FRIDAY, JANUARY 26, 2007

Debian HOW-TO : CPU power management

Last revision : February 5th, 2007

CPU frequency management is one of the keys to power preservation.

The Linux kernel now provides all the necessary tools to properly manage CPU frequency : no need to use a daemon (like cpufreqd or powernowd) to take care of your CPU.

Of course the benefits of such power management are obvious for a laptop, but most desktop users should also consider this.

In this tutorial, I use sudo to get root privileges.

• Prerequisites

Debian Etch (and Sid) should automatically configure CPU frequency management on most processors that supports it, so it might very well be already enabled. You can verify if that is the case using this command :

```
cpufreq-info
```

and analyze the output regarding the current policy.

If CPU frequency management is off (or the command is not found), then you can go on with this tutorial.

In order to make this work, you need to install the required packages:

```
sudo apt-get install cpufrequtils sysfsutils
```

Next, verify your exact CPU model :

```
cat /proc/cpuinfo | grep "model name"
```

Which should output something like that :

```
model name      : Intel(R) Pentium(R) M processor 1.73GHz
```

Once you know your exact CPU type, the next step is to load the proper modules : the **CPU frequency driver** and the **CPU frequency policy governor**.

- ## CPU frequency driver

As you may guess the CPU frequency driver will differ depending on your type of CPU. For example, my laptop is equipped with a Pentium M, so I type :

```
sudo modprobe speedstep_centrino
```

to load the proper driver.

Some of the other common drivers (or modules) are :

AMD K6 processors : powernow_k6
AMD K7 processors (Athlon, Duron, Sempron 32 bits) : powernow_k7
AMD K8 processors (Athlon 64, Turion 64, Sempron 64, Opteron 64) : powernow_k8

Pentium 4, Celeron D, Pentium D, Celeron M : p4_clockmod
Pentium M, Core Duo, Core 2 Duo : speedstep_centrino

There are of course other CPU frequency drivers. In doubt, you can use the **generic driver** : acpi_cpufreq

- ## CPU policy governor

Once the proper driver is loaded, you need to choose the desired CPU policy governor. This policy governor will manage the actual behavior of your CPU. Here is some policy governors and their module names :

performance, which sets the CPU statically to the highest possible frequency : cpufreq_performance
powersave, which is the opposite, clocks the CPU statically to the lowest frequency : cpufreq_powersave
ondemand, which sets the CPU speed dynamically depending on the work load (ideal for desktops) : cpufreq_ondemand
conservative, which also sets the CPU dynamically, but less aggressively then the ondemand governor (ideal for laptops) : cpufreq_conservative

For example, my machine has a Pentium M processor, so I type :

```
sudo modprobe speedstep_centrino
sudo modprobe cpufreq_ondemand
```

to load both the CPU frequency driver and the CPU policy governor.

- **CPU configuration**

Once the modules are loaded, you need to configure the policy governor. For example, I use the ondemand governor, so :

```
echo ondemand | sudo tee /sys/devices/system/cpu/cpu0/cpufreq
```

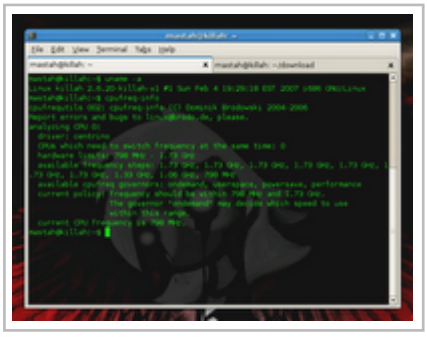
will enable it.

You can verify that everything went well with this command :

```
cpufreq-info
```

It should output your actual frequency, as well as the governor presently in use.

- **System configuration**



If everything is good, then you can make this configuration permanent. First make sure the proper modules are loaded at startup (in /etc/modules).

So in my case :

```
echo speedstep_centrino | sudo tee -a /etc/modules
echo cpufreq_ondemand | sudo tee -a /etc/modules
```

Finally, ensure that the CPU uses your policy governor of choice by default. Simply edit the file /etc/sysfs.conf with a line like this one :

```
devices/system/cpu/cpu0/cpufreq/scaling_governor = ondemand
```

That's it !