TI81XX PSP PM DVFS User Guide

TI81XX PSP PM DVFS User Guide

Linux PSP

IMPORTANT
TI81XX refers to TI816X, TI814X and TI813X.

About this manual

This document gives an overview and usage of Dynamic Voltage and Frequency scaling(DVFS) on TI81XX SOCs.

Acronyms & definitions

Acronyms

Acronym	Definition
DVFS	Dynamic Frequency and Voltage Scaling
CPUFreq	CPU Frequency scaling framework
OPP	Operating Performance Point

Introduction

Dynamic Voltage and Frequency scaling is a framework to change the frequency and/or operating voltage of a processor(s) based on system performance requirements at the given point of time.

Read this first: Please go through the following table to know current status of DVFS support on each device of TI81XX platform:

Supported platforms

Platform	Supported	Remarks
TI816X	NA	
TI814X	Yes	From Release 04.01.00.07 onwards
TI813X	TBD	

Legend: NA - No HW supported, YES - supported, NO - Not planned, TBD - planned but not implemented

Overview

DVFS framework operation involves two tasks listed below:

1. Frequency Scaling
2. Voltage Scaling.

These two are explained below in detail.

Frequency Scaling

Frequency scaling involves increasing or decreasing the cpu frequency based on a Governor's decision (user input, incase of UserSpace governor) and act upon the decision.

CPUFreq

CPUfreq is a linux kernel framework that monitors the performance requirements of a processor(s) and based on requirements takes decision either to increase to meet the performance requirements or decrease operating frequency in order to save power whenever possible.

CpuFreq Architecture

CPUFreq consists of two components

- *The Governor* - that makes decisions
- *The Driver* - acts based on the decisions made by the governor

Policy and Governor

Policy

Policy is set of rules the system is bound by such as min and max frequency for each cpu, availabilty of a frequency. Policy for a cpu is created during the CPUFreq framework initialization based on the frequency table.

Frequency Table

Frequency table consists of available frequencies for all cpus in the system. Frequency table is generated/populated based on the Operating Performance Point(OPP) list for each cpu.

OPP List

Operating performance Point(OPP) is a tuple consisting a frequency value and voltage required to run at the frequency. OPP table contains OPPs with a cpu/device name they are applicable to and an availabilty flag. OPP information of each device is added to OPP list.

Governor

Governor continuosly monitors the system perfomance requirements and when the requirement to change the frequency arises it checks the current cpu policy for frequency limits and requests the driver to change the frequency.

Driver

Multiple drivers can exist in the kernel but there will be only one scaling driver which performs actions based on governor's decision.When the govenor request the driver to change the frequency to a target value, driver checks the frequency availability in the OPP list. If its found it scales the device to new frequency.

Governors provided by CPUFreq

The following governors are available in CPUFreq frame work

- Performance
- Powersave
- Ondemand
- Userspace
- Conservative

*'**NOTE**: For more information each governor refer to Linux kernel documentation @ **<kernel>/Documentation/cpu-freq/governors.txt***

Enabling DVFS/CPUFreq in Linux PSP

To enable please do the following:

1. Enable CpuFrequency scaling and slect governors from CPU power management option as shown below:

```
CPU Power Management  --->
   [*] CPU Frequency scaling
               [ ]   Enable CPUfreq debugging (NEW)
               <*>   CPU frequency translation statistics (NEW)
               [ ]     CPU frequency translation statistics details (NEW)
                     Default CPUFreq governor (performance)  --->
               -*-   'performance' governor
               < >   'powersave' governor (NEW)
               <* >   'userspace' governor for userspace frequency scaling (NEW)
               < >   'ondemand' cpufreq policy governor (NEW)
               < >   'conservative' cpufreq governor (NEW)
               [ ] CPU idle PM support
```

2. Enable Architecture support and OPP Library

```
  Power management options  --->
       -*- Power Management support
               [*]   Power Management Debug Support
               [ ]     Extra PM attributes in sysfs for low-level debugging/testing
               [*]     Verbose Power Management debugging
               [*] Suspend to RAM and standby
               [ ]   Test suspend/resume and wakealarm during bootup
               < > Advanced Power Management Emulation
               -*- Run-time PM core functionality
               [*] Architecure Supports multiple OPPs
               [*]   Operating Performance Point (OPP) Layer library
```

Default Governor

Default governor is "performance". The default governor can be changed through 'menuconfig':

```
Chose the governor you want to use as default governor by selecting it from the list at:
 CPU Power management options --->
  [*] CPU frequency scaling
    Default CPUFreq governor (performance)  --->
      (x) performance
      () userspace
      () ondemand
      () conservative
```

Build Multiple governors

Multiple governors can be built and exist in the kernel by selecting them through "menuconfig".

```
Select the governors you want to build in to kernel.
 CPU Power management options --->
  [*] CPU frequency scaling
    Default CPUFreq governor (performance)  --->
  [*] performance
  [*] userspace
  [*] ondemand
  [] conservative
```

Governors can be switched at any time through the 'sysfs' interface.

- To list all available governors:

```
cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_governors
```

- To see current active governor

```
cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

- To switch to a different governor

```
echo –n "<governor_name>" > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

e.g. to switch to 'userspace' governor

```
echo –n "userspace" > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

- To see current

CPUFreq Frequency transition notification

CPUFreq driver supports notifications of frequency changes

CPUFreq User Interface

CPUFreq exports a lot of information to user through sysfs interface.
To see information pertaining to a cpu(let's say cpu0):

- current frequency of a cpu:

```
cat /sys/devices/system/cpu/cpu0/cpufreq/cpuinfo_cur_freq
```

- Available frequencies:

```
cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies
```

- Available governors

```
cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_governors
```

- Current governor/ governor in use

```
cat /sys/devices/ system/cpu/cpu0/cpufreq/scaling_governor
```

Voltage Scaling

Voltage scaling is achieved using voltage layer and regulator framework(driver). When the CPUFreq driver scales the device frequency, voltage corresponding to the frequency(target_voltage) is looked-up in the opp list. The device scale function requests the voltage layer to scale the device voltage to the target_voltage.

Voltage Layer

The voltage layer consists of the information of all voltage domains in the system and configures all VDDs during voltage layer initialization. When a VDD is configured a regulator supply handle is acquired and stored in the corresponding vdd structure.The regulators scale/set voltage function is plugged in to the VDD's voltage scale function pointer. Thus when a voltage change is requested forwarded to a VDD. The voltage layer requests the regulator framework to change the device voltage to the target voltage. Regulator driver verifies if the target voltage is in within the limits of the voltage domain and regulator supply constraints. If all the checks go through then the regulator changes voltage of the requested device to the target voltage.

Transitioning to a different Power state(OPPs)

Userspace governor

Using "userspace" governor one can change the current OPP:

- Select userspace governor

```
echo –n "userspace" > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

- When using "userspace" governor to change to a low/high power state/OPP, first check the available frequencies:

```
$cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies
```

Select a frequency and to change current cpu frequency, execute the following:

```
$echo –n "<new_frequency> > /sys/devices/system/cpu/cpu0/cpufreq/scaling_setspeed
```

when the frequency is changed, system voltage is also changed to meet the new requirements as part of scaling: This is done in two ways:

```
* when new frequency is higher (moving to high power state/opp)
 * Voltage is increased first then the frequency,
* when new frequency is lower (moving to low power state/opp)
 * Frequency is reduced first then the voltage.
```

All other governors

For all other governors user does not need to give input in order to move to a low power/high power state.
The governor takes the decision to transit based on system requirements and does the scaling of frequency and voltage.
Each governors provides some configuration options which can be set/modified through *sysfs*. For detailed configuration options available for each governor refer to kernel documentation on governors **@<kernel>/Documentation/cpu-freq/governors.txt**

References

For more information on CPUFreq framework refer to the documentation available in Linux Kernel source @ **<Kernel>/Documentation/cpu-freq**.

Article Sources and Contributors

TI81XX PSP PM DVFS User Guide *Source*: http://processors.wiki.ti.com/index.php?oldid=124081 *Contributors*: RK

Image Sources, Licenses and Contributors

Image:TIBanner.png *Source*: http://processors.wiki.ti.com/index.php?title=File:TIBanner.png *License*: unknown *Contributors*: Nsnehaprabha

License

8. Miscellaneous

a. Each time You Distribute or Publicly Perform the Work or a Collection, the Licensor offers to the recipient a license to the Work on the same terms and conditions as the license granted to You under this License.
b. Each time You Distribute or Publicly Perform an Adaptation, Licensor offers to the recipient a license to the original Work on the same terms and conditions as the license granted to You under this License.
c. If any provision of this License is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this License, and without further action by the parties to this agreement, such provision shall be reformed to the minimum extent necessary to make such provision valid and enforceable.
d. No term or provision of this License shall be deemed waived and no breach consented to unless such waiver or consent shall be in writing and signed by the party to be charged with such waiver or consent.
e. This License constitutes the entire agreement between the parties with respect to the Work licensed here. There are no understandings, agreements or representations with respect to the Work not specified here. Licensor shall not be bound by any additional provisions that may appear in any communication from You. This License may not be modified without the mutual written agreement of the Licensor and You.
f. The rights granted under, and the subject matter referenced, in this License were drafted utilizing the terminology of the Berne Convention for the Protection of Literary and Artistic Works (as amended on September 28, 1979), the Rome Convention of 1961, the WIPO Copyright Treaty of 1996, the WIPO Performances and Phonograms Treaty of 1996 and the Universal Copyright Convention (as revised on July 24, 1971). These rights and subject matter take effect in the relevant jurisdiction in which the License terms are sought to be enforced according to the corresponding provisions of the implementation of those treaty provisions in the applicable national law. If the standard suite of rights granted under applicable copyright law includes additional rights not granted under this License, such additional rights are deemed to be included in the License; this License is not intended to restrict the license of any rights under applicable law.