AM335x Power Management User guide

Navigation

- Main Page
- All pages
- All categories
- Popular pages
- Popular authors
- Popular categories
- Category stats
- Recent changes
- Random page
- Help
- Google Search

Print/export

- Create a book
- Download as PDF
- Printable version

Toolbox

- What links here
- Related changes
- Special pages
- Permanent link
- Browse properties

AM335x Power Management User's Guide

Linux PSP

Contents [hide]

1 Introduction
2 PM features in the kernel
3 cpufreq
3.1 Kernel configuration
3.2 Usage
3.2.1 Supported OPPs
3.2.2 Changing OPPs
4 cpuidle
4.1 Kernel configuration
4.2 C-states
4.2.1 CPU Idle Governor
4.2.2 CPU Idle Driver
4.3 Supported C-states
5 Suspend & Resume
5.1 Configuration
5.1.1 Enable/Disable suspend-resume support
5.1.2 Cortex-M3 binary blob for suspend-resume
5.2 Usage
5.2.1 Successful suspend-resume cycle
5.3 Wakeup sources

Introduction

AM335x processors support a rich set of power management techniques

1. Clock gating
2. Clock domain transitions
3. Power domain transitions
4. Dynamic Voltage and Frequency scaling
5. DeepSleep modes

These techniques are leveraged in the different power management frameworks in the Linux kernel.

PM features in the kernel

PM features in Linux kernel relevant to AM335x are

1. CPU Freq is available in kernel since release 04.06.00.04 onward.
2. CPU Idle is available in kernel since release 04.06.00.03 onward.
3. Suspend-to-RAM is available in kernel since release 04.06.00.07 onward.
4. Runtime PM is supported in most of the drivers. The usage will be made more aggressive in the upcoming releases.

cpufreq

Cpufreq framework provides support to change frequency of processor on the run. This helps to save processor power, when the load is less (processor power is proportional to frequency and square of the voltage). cpufreq framework works in conjunction with driver & governor.

- cpufreq driver provides link for framework to achieve frequency change at the hardware level.
- cpufreq governor decides frequency at which processor should run. Various governors like ondemand, userspace, performance (self descriptive names) exist in Linux Kernel.

Changing voltage is also achieved in cpufreq driver by dealing with voltage regulator. OPP infrastructure is made use to achieve it.

Kernel configuration

Start the *Linux Kernel Configuration* tool.

```
$ make menuconfig
```

Select *CPU Power Management* from the main menu.

```
    ...
    ...
    Boot options --->
    CPU Power Management --->
    Floating point emulation --->
    ...
```

Select *CPU Frequency Scaling* as shown here:

```
    ...
    ...
        CPU Frequency Scaling --->
    [*] CPU idle PM support
    ...
```

Select *CPU Frequency scaling* and required governers as shown here:

```

    [*] CPU Frequency scaling
    <*> CPU frequency translation statistics
    [ ] CPU frequency translation statistics details
        Default CPUFreq governor (userspace)  --->
    < >   'performance' governor
    < >   'powersave' governor
    -*-   'userspace' governor for userspace frequency scaling
    <*>   'ondemand' cpufreq policy governor
    < >   'conservative' cpufreq governor
    ...
```

NOTE
CPUFreq is enabled by default in the am335x_evm_defconfig.

Usage

ondemand and userspace governors are enabled in Kernel, default governor is userspace.

To view available governers,

```
target# cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_governors
ondemand userspace
target#
```

To view current governer,

```
target# cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
userspace
target#
```

To set a governer,

```
target# echo ondemand > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
target#
```

Current frequency can be obtained as,

```
target# cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_cur_freq
720000
target#
```

For current voltage,

```
target# cat /sys/class/regulator/regulator.3/microvolts
1262500
target#
```

Supported OPPs

OPP (Operating Performance Points) - voltage frequency pair that defines a specific power state that the SoC supports running at. Refer to SoC Technical Reference Manual for more details. cpufreq implementation controls OPP only with respect to the processor, i.e VDD_MPU (voltage fed to ARM Cortex-A8) and ARM Cortex-A8 frequency.

To get supported OPP's (frequency),

```
target# cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies
275000 500000 600000 720000
target#
```

Changing OPPs

This can be done only for userspace governor. If ondemand governor is used, OPP change happens automatically based on the system load.

```
target# echo 275000 > /sys/devices/system/cpu/cpu0/cpufreq/scaling_setspeed
target#
```

cpuidle

The *cpuidle* framework consists of two key components:

- A governor that decides the target C-state of the system.
- A driver that implements the functions to transition to target C-state.

The idle loop is executed when the Linux scheduler has no thread to run. When the idle loop is executed, current 'governor' is called to decide the target C-state. Governor decides whether to continue in current state/ transition to a different state. Current 'driver' is called to transition to the selected state.

Kernel configuration

Start the *Linux Kernel Configuration* tool.

```
$ make menuconfig
```

Select *CPU Power Management* from the main menu.

```
    ...
    ...
    Boot options --->
    CPU Power Management --->
    Floating point emulation --->
    ...
```

Select *CPU idle PM support* to enable the cpuidle driver.

```
    ...
    ...
        CPU Frequency Scaling --->
    [*] CPU idle PM support
    ...
```

NOTE
cpuidle is enabled by default in the am335x_evm_defconfig.

C-states

A C-state is used to identify the power state supported through the cpu idle loop. Each C-state is characterized by its:

- Power consumption
- Wakeup latency
- Preservation of processor state while in 'the' state.

Currently these C-states have been defined:

C-states in AM335x

State	Description
C1	MPU WFI
C2	MPU WFI + DDR in sel refresh

CPU Idle Governor

The current implementation supports 'menu' and 'ladder' governors to decide the target C-state of the system.

CPU Idle Driver

The cpuidle driver registers itself with the framework during boot-up and populates the C-sates with exit latency, target residency (minimum period for which the state should be maintained for it to be useful).

Once the governor has decided the target C-state, the control reaches the function `am33xx_enter_idle()`. Here, the C-state is adjusted based on the value of *valid* flag corresponding to the chosen state.

Supported C-states

```
[root@arago /]# ls -l /sys/devices/system/cpu/cpu0/cpuidle/state0/
-r--r--r--    1 root     root         4096 Jan  1 00:02 desc
-r--r--r--    1 root     root         4096 Jan  1 00:02 latency
-r--r--r--    1 root     root         4096 Jan  1 00:02 name
-r--r--r--    1 root     root         4096 Jan  1 00:02 power
-r--r--r--    1 root     root         4096 Jan  1 00:02 time
-r--r--r--    1 root     root         4096 Jan  1 00:02 usage
```

```
[root@arago /]# ls -l /sys/devices/system/cpu/cpu0/cpuidle/state1/
-r--r--r--    1 root     root         4096 Jan  1 00:05 desc
-r--r--r--    1 root     root         4096 Jan  1 00:05 latency
-r--r--r--    1 root     root         4096 Jan  1 00:03 name
-r--r--r--    1 root     root         4096 Jan  1 00:05 power
-r--r--r--    1 root     root         4096 Jan  1 00:05 time
-r--r--r--    1 root     root         4096 Jan  1 00:02 usage
```

Suspend & Resume

The suspend operation results in the system transitioning to the lowest power state being supported. On AM335x, this maps to DeepSleep0 state. For more info on this refer to the AM335x TRM available @ www.ti.com/am335x

The drivers implement the `suspend()` function defined in the LDM. When the suspend for the system is asserted, the `suspend()` function is called for all drivers. The drivers quiesce the peripherals and release the clocks to reach the desired low power state. The actual transition to suspend is implemented in the function `am33xx_pm_suspend()`.

Configuration

Enable/Disable suspend-resume support

To enable/ disable suspend-resume support start the *Linux Kernel Configuration* tool.

```
$ make menuconfig
```

Select *Power management options* from the main menu.

```
...
...
Kernel Features  --->
Boot options  --->
CPU Power Management  --->
Floating point emulation  --->
Userspace binary formats  --->
Power management options  --->
[*] Networking support  --->
Device Drivers  --->
...
...
```

Select *Suspend to RAM and standby* to toggle the power management support.

```
[*] Suspend to RAM and standby
-*- Run-time PM core functionality
...
< > Advanced Power Management Emulation
```

Cortex-M3 binary blob for suspend-resume

On AM335x, suspend-resume involves loading a binary to the Cortex-M3 core during the boot process. For this the binary named am335x-pm-firmware.bin needs to be kept in the firmware/ folder on the kernel sources before the build process is started.

For more information on how to obtain the above mentioned binary, refer to the accompanying Release Notes for the PSP package.

The pre-built kernel image in the PSP package has the binary blob compiled into the kernel image.

To manually compile a kernel image with the PM firmware start the *Linux Kernel Configuration* tool.

```
$ make menuconfig
```

Select *Device Drivers* from the main menu.

```
...
...
Kernel Features  --->
Boot options  --->
CPU Power Management  --->
Floating point emulation  --->
Userspace binary formats  --->
Power management options  --->
[*] Networking support  --->
Device Drivers  --->
...
...
```

Select Generic Driver Options

```
Generic Driver Options
CBUS support
...
...
```

Configure the name of the PM firmware and the location as shown below

```
...
-*- Userspace firmware loading support
[*] Include in-kernel firmware blobs in the kernel binary
(am335x-pm-firmware.bin) External firmware blobs to build into the kernel binary
(firmware) Firmware blobs root directory
```

Note: The above configuration assumes that the PM firmware named **am335x-pm-firmware.bin** has been placed under **firmware** folder of the kernel sources.

NOTE
The above changes are enabled by default in the am335x_evm_defconfig.

Usage

Once a kernel image with the PM firmware has been compiled and booted up, the following messages are displayed during the bootup

```
Uncompressing Linux... done, booting the kernel.
[    0.000000] Linux version 3.2.0-...
[    0.000000] CPU: ARMv7 Processor [413fc082] revision 2 (ARMv7), cr=10c53c7d
...
...
[    2.118896] Power Management for AM33XX family
[    2.127075] Trying to load am335x-pm-firmware.bin (60 secs timeout)
[    2.137084] Copied the M3 firmware to UMEM
...
...
```

The suspend for device can now be asserted as follows:

```
$ echo mem > /sys/power/state
```

Successful suspend-resume cycle

```
[root@arago /]# echo mem > /sys/power/state
[    8.774536] PM: Syncing filesystems ... done.
[    8.804199] Freezing user space processes ... (elapsed 0.01 seconds) done.
[    8.827575] Freezing remaining freezable tasks ... (elapsed 0.01 seconds) done.
[    8.847717] Suspending console(s) (use no_console_suspend to debug)
[    8.868530] PM: suspend of devices complete after 12.542 msecs <-- Wake event using console (UART0)
[    8.870147] PM: late suspend of devices complete after 1.556 msecs
[   10.840209] GFX domain entered low power state
[   11.044891] PM: early resume of devices complete after 204.184 msecs
[   11.284027] PM: resume of devices complete after 238.709 msecs
[   11.317382] Successfully transitioned all domains to low power state
[   11.325012] Restarting tasks ... done
[root@arago /]#
```

Note: In order to avoid data loss, it is recommended that the users issue a "sync" command before suspending the system

```
[root@arago /]# sync; echo mem > /sys/power/state
```

Wakeup sources

In the default configuration wakeup from suspend using the following interfaces are supported:

- UART0 (console)
- GPIO0
- Touchscreen:

To disable wakeup from touchscreen interface the following command can be used:

```
[root@arago /]# echo disabled > /sys/devices/platform/omap/tsc/power/wakeup
```

Note: By using this feature, ADC used for touchscreen is stopped, hence the power consumption is further reduced. This feature is available from 04.06.00.07 release.

To re-enable wake from touchscreen, use:

```
[root@arago /]# echo enabled > /sys/devices/platform/omap/tsc/power/wakeup
```

For technical support please post your questions at http://e2e.ti.com. Please post only comments about the article ***AM335x Power Management User guide*** *here.*

Links

ARM Microcontroller MCU | ARM Processor | Digital Media Processor | Digital Signal Processing | Microcontroller MCU | Multi Core Processor

Ultra Low Power DSP | 8 bit Microcontroller MCU | 16 bit Microcontroller MCU | 32 bit Microcontroller MCU

Categories: AM335x | AM35x | AM37x | AM1x | AM18x | Linux | PSP

Leave a Comment

This page was last modified on 25 July 2012, at 01:15.

This page has been accessed 3,854 times.

Privacy policy   About Texas Instruments Embedded Processors Wiki   Disclaimers