1 Introduction

In a previous article, we discussed the hardware aspects associated with power management and power budgeting for the OSD335x. However, to optimize the power performance of your design you must also fine tune the power usage of the AM335x inside the OSD335x through software. Using firmware to control the power management of the OSD335x leads to significant power savings in some applications. This document introduces software power management techniques for the OSD335x in Linux and provides examples to demonstrate the advantages of these techniques.

Table of Contents

2 Revision History

Revision Number	Revision Date	Changes	Author
1	7/11/2017	Initial Release	Neeraj Dantu

3 Power Saving Features of AM335x

The AM335x processor inside the OSD335x allows you to control different aspects of its functionality to optimize power consumption for your application. The following list outlines the main features used to minimize power consumption (Source: Linux Power Management User Guide).

1. Clock gating: Involves dynamic enabling/disabling clocks of peripherals to reduce dynamic power consumption. See section 8.1.3.2 in the technical reference manual for more information.
2. Clock domain transitions: Involves enabling/disabling a clock domain (which consists of a group of peripherals fed by clock signals controlled by the same clock manager) to reduce dynamic power consumption. See section 8.1.3.3 in the technical reference manual for more information.
3. Power domain transitions: Involves managing 4 functional power domains of the processor (WAKEUP, MPU, PER and RTC). See section 8.1.4.1 for more information on power domains.
4. Dynamic voltage and frequency scaling: Involves managing the Operational Performance Points (OPPs) of the processor. This is discussed in detail below.
5. Low power modes: Involves using one of the 4 low power modes of the processor based on application to optimize power consumption. See section 8.1.4.3 for more information on power modes.

You can use bare-metal (No OS) programming to utilize all these features and have complete control of power management of the device. However, under Linux, some of these features are leveraged automatically while others are accessible through simple abstractions. Linux provides constructs that enable you to easily minimize power consumption. The rest of this article will explore controlling the two most common features, Operational Performance Points, and Low Power Modes, through Linux. If you are interested in how to control the other features, please see the Technical Reference Manual of AM335x.

4 Operational Performance Points (OPPs)

The TPS65217C Power Management IC (PMIC), inside the OSD335x, supplies the different voltages to each of the many voltage domains inside the AM335x, such as VDD_MPU and VDD_CORE. These voltages are changed by pragmatically communicating with the PMIC over the I2C interface.

The AM335x also contains multiple clock domains, such as MPU PLL and Core PLL, that operate independently and are fully managed within the SoC.

Operational Performance Points (OPPs) consist of a set of predefined voltage ranges and maximum frequencies. These OPPs set the operating voltages for the voltage domains and the frequencies for the clock domains of the AM335x System-on-Chip (SoC). For example, to run the Cortex A8 processor within the AM335x at 1GHz, the VDD_MPU voltage domain must be set between 1.272V and 1.378V. This operating condition is delineated as OPP “Nitro” for VDD_MPU. OPPs affect current draw and device lifetime (power-on-hours). They are discussed in detail in AM335x datasheet, but, specifics for the DDR version and AM335x revision inside the are discussed below.

OPPs are set for two of the AM335x voltage domains, VDD_CORE and VDD_MPU. Selecting and using the appropriate OPP for your application is critical to optimizing the power consumption of the OSD335x. The different OPPs for the two domains, their voltage ranges, and frequencies of operation are given in the tables below. (This information can be found in section 5.4 (Operating Performance Points) of the AM335x datasheet and is provided here for convenience).

Table 1: VDD_MPU OPPs

VDD_MPU OPP	VDD_MPU			ARM Clock Speed
	MIN	NOM	MAX	
Nitro	1.272 V	1.325 V	1.378 V	1 GHz
Turbo	1.210 V	1.260 V	1.326 V	800 MHz
OPP120	1.152 V	1.200 V	1.248 V	720 MHz
OPP100	1.056 V	1.100 V	1.144 V	600 MHz
OPP50	0.912 V	0.950 V	0.988 V	300 MHz

Table 2: VDD_CORE OPPs

VDD_CORE OPP	VDD_CORE			DDR3L frequency	L3 and L4 frequency
	MIN	TYP	MAX		
OPP100	1.056 V	1.100 V	1.144 V	400 MHz	200 and 100 MHz
OPP50	0.912 V	0.950 V	0.988 V	–	100 and 50 MHz

Perk:
Using VDD_CORE OPP50 will disable the DDR3L memory present inside the OSD335x

There are also a valid set of combinations of VDD_CORE and VDD_MPU OPPs. Those are listed in the following table.

Table 3: Valid combinations of VDD_CORE and VDD_MPU OPPs

VDD_CORE	VDD_MPU
OPP50	OPP50
OPP50	OPP100
OPP100	OPP50
OPP100	OPP100
OPP100	OPP120
OPP100	Turbo
OPP100	Nitro

4.1 Controlling OPPs in Linux

Linux has a built-in framework for the user to make use of the OPPs. How to use this functionality is demonstrated below. The Linux image: bone-debian-8.4-lxqt-4gb-armhf-2016-05-13-4gb.img (Beagleboard.org Latest Images) was used on the OSD3358 SBC Reference Design board for the demonstration.

4.1.1 Linux Governors

The Linux construct for controlling OPPs, specifically the VDD_MPU OPP, is CPUfreq governors. They can be used to change the frequency and voltage of the processor on the fly. They manage the OPPs of the processor while providing a user friendly construct. For a full explanation of governors see: kernel.org Governors documentation. The following command can be used to view all the available governors.

```
root@beaglebone:~#cat
/sys/devices/system/cpu/cpu0/cpufreq/scaling_available_governors
conservative ondemand userspace powersave performance
```

Each governor uses a different set of criteria to dynamically set the processor to optimize performance. For example, when the *powersave* governor is used, the processor is set to the lowest defined frequency. When the *ondemand* governor is used the frequency is automatically adjusted to match the load on the processor. The governor for the CPU is set by writing its value to the to the scaling_governor file.

The following command sets the active governor to *ondemand*.

```
root@beaglebone:~#echo ondemand >
/sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

4.1.2 Manually setting OSD335x frequencies

Manually setting the frequency of the processor requires setting the governor to *userspace*. This allows the user to switch between pre-defined OPPs. These OPPs are defined in the device tree which is loaded during boot. To see the frequencies (in kHz) that are supported by the OSD335x look at the scaling_available_frequencies file. The following command can be used to see them:

```
root@beaglebone:~#cat
/sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies
300000 600000 720000 800000 1000000
```

Now that the available frequencies are known, you can set them using the following command.

```
root@beaglebone:~# echo 720000 >
/sys/devices/system/cpu/cpu0/cpufreq/scaling_setspeed
```

This command sets the frequency of the CPU to 720 MHz. Simply change the 720000 to your desired frequency to make the CPU operate at that frequency.

In addition to setting the frequency, the voltage rails of the processor can also be monitored using a similar approach. The following command returns the output voltage of regulator 2 (VDDS_DDR).

```
root@beaglebone:~# cat /sys/class/regulator/regulator.2/microvolts
1500000
```

Several CPU frequency statistics can be found by running 'cpufreq-info' in command line.

```
root@beaglebone:~# cpufreq-info
cpufrequtils 008: cpufreq-info (C) Dominik Brodowski 2004-2009
Report errors and bugs to cpufreq@vger.kernel.org, please.
analyzing CPU 0:
      driver: cpufreq-dt
      CPUs which run at the same hardware frequency: 0
      CPUs which need to have their frequency coordinated by software: 0
      maximum transition latency: 300 us.
      hardware limits: 300 MHz - 1000 MHz
      available frequency steps: 300 MHz, 600 MHz, 720 MHz, 800 MHz, 1000
      MHz
      available cpufreq governors: conservative, ondemand, userspace,
      powersave, performance
      current policy: frequency should be within 300 MHz and 1000 MHz.
            The governor "userspace" may decide which speed to use within
            this range.
      current CPU frequency is 300 MHz (asserted by call to hardware).
      cpufreq stats: 300 MHz:93.27%, 600 MHz:1.49%, 720 MHz:0.47%, 800
      MHz:0.30%, 1000 MHz:4.47% (194)
```

4.1.3 OPPs effect on Power Consumption

Operating performance points can be leveraged to reduce overall current consumption of the board. Table 4 shows a comparison between the current consumption of the board in different OPPs running the same example application. The example application involves the processor gathering sensor data to make decisions.

Table 4: Application power consumption for various OPPs

OPP MPU Frequency	300MHz	600MHz	800MHz	1000MHz
Average Current Consumption	195 mA	230 mA	266 mA	295 mA

As you can see, the average current consumption of the example application drops as the OPP frequency changes from 1GHz to 300MHz. One thing to note about the example application is that it is I/O limited and not CPU limited. This means that the CPU frequency can be reduced without reducing the overall performance of the application. In the case of a CPU limited application, care needs to be taken that the change in performance from modifying the OPP does not cause real-time deadlines to be missed.

5 Low Power Modes

Often, the system can be put in a low power mode when there is nothing to do to save power. This is especially useful in battery powered applications. While the AM335x has 4 low powered states, 3 are supported by the OSD335x. The unsupported RTC-Only mode is due to the version of the PMIC inside the SiP (ie the OSD335x uses the TPS65217C). The low power states and their corresponding power consumption are listed in the following table taken from AM335x datasheet.

POWER MODES	APPLICATION STATE	POWER DOMAINS, CLOCKS, AND VOLTAGE SUPPLY STATES	NOM	MAX	UNIT
Standby	DDR memory is in self-refresh and contents are preserved. Wake up from any GPIO. Cortex-A8 context/register contents are lost and must be saved before entering standby. On exit, context must be restored from DDR. For wake-up, boot ROM executes and branches to system resume.	Power supplies: • All power supplies are ON. • VDD_MPU = 0.95 V (nom) • VDD_CORE = 0.95 V (nom) Clocks: • Main Oscillator (OSC0) = ON • All DPLLs are in bypass. Power domains: • PD_PER = ON • PD_MPU = OFF • PD_GFX = OFF • PD_WKUP = ON DDR is in self-refresh.	16.5	22.0	mW
Deepsleep1	On-chip peripheral registers are preserved. Cortex-A8 context/registers are lost, so the application needs to save them to the L3 OCMC RAM or DDR before entering DeepSleep. DDR is in self-refresh. For wake-up, boot ROM executes and branches to system resume.	Power supplies: • All power supplies are ON. • VDD_MPU = 0.95 V (nom) • VDD_CORE = 0.95 V (nom) Clocks: • Main Oscillator (OSC0) = OFF • All DPLLs are in bypass. Power domains: • PD_PER = ON • PD_MPU = OFF • PD_GFX = OFF • PD_WKUP = ON DDR is in self-refresh.	6.0	10.0	mW
Deepsleep0	PD_PER peripheral and Cortex-A8/MPU register information will be lost. On- chip peripheral register (context) information of PD-PER domain needs to be saved by application to SDRAM before entering this mode. DDR is in self-refresh. For wake-up, boot ROM executes and branches to peripheral context restore followed by system resume.	Power supplies: • All power supplies are ON. • VDD_MPU = 0.95 V (nom) • VDD_CORE = 0.95 V (nom) Clocks: • Main Oscillator (OSC0) = OFF • All DPLLs are in bypass. Power domains: • PD_PER = OFF • PD_MPU = OFF • PD_GFX = OFF • PD_WKUP = ON DDR is in self-refresh.	3.0	4.3	mW

Figure 1: AM335x Low Power Modes

A list of resources for power management including implementation of various power modes are given below:

1. AM335x Power Management User Guide
2. Linux Core Power Management User's Guide
3. AM335x Linux Power Management User Guide
4. AM335x Power Management Standby User's Guide

Note that some of the features in the above resources are specific to TI's Linux distributions and are not available in all Linux images. Please make sure your image supports the functions you wish to use.

5.1 Setting Power Modes

Setting processor power modes is straight forward. To see all the available low power states, look at the state file using the following command

```
root@beaglebone:~# cat sys/power/state freeze standby mem disk
```

While the power state names are not the same as modes listed in Table 5, some of them do map to the processors low power modes. The *mem* power state corresponds to *deepsleep0* state of the processor described in Table 5. Similarly, the *standby* power state corresponds to the *standby* state of the processor. More information on these low power modes can be found in the references.

Depending on the version of the Linux image used, some of the low power modes might not be supported.

To put the processor in the desired sate, just write the power mode you wish to operate in to the state file. This command will put the processor in the *standby* power state:

```
root@beaglebone:~# echo standby > /sys/power/state
```

Waking the processor up from a low power mode requires an interrupt sent through one of the wakeup sources. Availability of this feature by default depends on the Linux image version on the device. Please check the documentation of the image to see all the wakeup sources and if they are enabled by default.

5.2 Maximizing Power on Time Using Low Power Modes

This section illustrates how power consumption can be optimized in an example application scenario. We are using the OSD3358 SBC Reference Design running bone-debian-8.6-lxqt-4gb-armhf-2016-11-06-4gb (Beagleboard.org Latest Images) as a data collector in this example.

Let us assume the OSD335x SBC reference design is being used to collect, process and write data to the eMMC on board. The data is being collected over an I2C bus from sensors that have their own power source. After receiving the data, the processor will perform some basic arithmetic operations on it to determine a course of action such as turning one of the GPIO pins on/off. It will then take the action required and write the data to the on board eMMC.

To save power when the board is not collecting any data, it can be put into standby mode as shown earlier. It is setup to wake up using the UART0_RX pin. So, the board has two states of operation. The current consumption of each state and the percentage of total time spent in these states are shown below. Some field experimentation may be required to determine the percentage of time spent in a state.

Table 5: SBC power consumption in different states

State	Current consumed for 5V input	% time spent in this state
Application running	291.1 mA	20
Stand-by	48mA	80

If the board is being powered by a 3000 mAh battery, the board should be able to run for 31 hours before the battery needs to be changed. This is in comparison to 10.3 hours with no standby mode. This scenario illustrates the important nature of low power modes especially in event driven systems.

6 Conclusion

Along with hardware power budgeting, power management through software can play a major role in reducing power consumption especially in application driven scenarios. These power savings are achieved through leveraging various features of the PMIC and the AM335x processor inside the OSD335x.

7 Reference Documents

1. TPS65217C datasheet
2. TL5209 LDO datasheet
3. AM335x datasheet
4. Powering the AM335x with TPS65217C
5. Processor SDK Linux kernel performance guide
6. AM335x power consumption summary
7. AM335x Power Management User Guide
8. Linux Core Power Management User's guide
9. AM335x Power Management User Guide
10. AM335x Power Management Standby User's Guide

Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View OSD3358-512M-ISM on WIN SOURCE
- Octavo Systems Information

Optimize Your Supply Chain with WIN SOURCE Solutions

- Global Sourcing Solution
- Obsolete Management
- Cost Control Management
- Shortage Management
- Alternative Solution
- Excess Inventory Management