# Unified Power Management Framework for Portable Media Devices

Ashwin Iyenggar, Ambudhar Tripathi, Ajit Basarur and Indranil Roy
{ashwin.iyenggar@gmail.com , ambudhar.tripathi@ittiam.com, ajit.basarur@ittiam.com and indranil.roy@ittiam.com }
Ittiam Systems Pvt. Ltd, 1 Richmond Road, Bangalore-560025, India.
{www.ittiam.com}

***Abstract* – Power management is one of the key design challenges in the design of Portable Multimedia Devices (PMD). Existing power management techniques are limited to hardware based DVFS (Dynamic schemes and do not take into account application states and many usage scenarios. In this correspondence, a new Unified Power Management Framework (UPMF) is proposed which describes a software architecture for a unified hardware and software controlled power management. The framework provides a set of rules for classifying the several devices on a PMD into device classes. Each of these device classes has a set of well-defined states based on the application state. The power management framework contains a power manager module that monitors the application and maintains the states of the devices and based on the power management strategy, controls the states of the devices. The proposed UPMF is very simple to implement and is well suited to handle power management strategies for several classes of applications. Performance of this proposed scheme for a Portable Media Player and Recorder device is also presented.**

## I. Introduction

With the advent of faster processors and higher capacity storage media the audio devices in the market are quickly transitioning into Portable Media Devices (PMD) capable of handling the functionalities of an audio player, video player, still camera, camcorder and gaming devices. This convergence in applications coupled with evolving consumer expectations and rapid development in processor capabilities pose tremendous challenges in the overall system design. One of the many design challenges to be addressed in this fast growing segment is the ever increasing demand for power hungry media processing functions to be offset with the demand for longer battery life.

Integration at the chip level often combining the many processing cores and peripheral devices interface into a single chip exacerbates the power dissipation problem. A similar integration in the software system capable of handling multiple functionalities degrades the efficiency of any hardware only power management techniques. Therefore, power management no longer remains a hardware only problem and needs to be addressed at the system level. Most existing techniques have limited themselves to hardware based Dynamic Voltage and Frequency Scaling (DVFS) schemes [3], [4], [5], [6]. However, these techniques have not been designed well enough to be scalable to handle the many application states that the devices support. Therefore there is a strong need to devise new system level power management solutions that utilizes the DVFS to the best possible extent as application moves from one state to other.

In this correspondence, we propose a Unified Power Management Framework (UPMF) that is capable of handling the many multimedia applications in a single device and provide best possible power optimization strategy suitable for each of the several applications supported in a single device.

## II. Multimedia Applications – Power Usage Characteristics

Today's portable multimedia devices use highly integrated SOCs. We will consider a typical multimedia SOC based system that has more than one DSP processing core and a RISC core with on-chip peripherals support for multimedia functionality. Commercially available devices like TI's TMS320DM320 [12], DaVinci™ [12] and ADI's BlackFin™ [13] series of processors are a good example of such integrated SOCs aimed at the portable consumer electronics market. Fig. 2 shows a high-level block diagram of a typical PMD system.

Portable Media Devices typically support one or more applications like audio player, image viewer, image viewer with audio player, audio-video player, audio recorder, still camera, movie recorder. These applications use one or more of the following on chip or on board devices: storage media (Compact Flash, Harddisk, SmartCards), Ethernet or Wireless LAN chip for accessing content, LCD or a NTSC encoder (TV out interface) for display, CCD/CMOS or a NTSC decoder (Video input interface) chip for capture, microphone or line in device for audio/voice capture, speaker or headphone for audio rendering, on board SDRAM for processing/storing data, audio and video DAC/ADC, hardware scaler, image processing modules. Consider an audio only player mode of operation with content from local storage media. With reference to Fig. 2, LCD, TV In, Video out, Ethernet modules will be inactive in this mode of operation. Also the many coprocessors available on chip will be idle in this mode and the System RAM can be clocked lower and the HDD can be turned off for longer time by intelligently caching data onto on-board SDRAM (or System RAM). Using existing hardware based power management techniques like DVFS [1], [2], [7], [11] and activity detection only it would be impossible to do power management of the on-chip peripherals and interfaces like TV out, AV DACs. Also we would need an application aware software power management technique to turn off these

1-4244-1039-8/07/$25.00 ©2007 IEEE

unused on chip and on board peripheral devices. Another usage scenario is one in which the user is browsing through the files. Since the usage pattern cannot be predicted in this case any activity based power saving using existing power management techniques would not help. A software controlled application aware power manager can identify this state and lower the clocks and turn of unused processing modules. Another significant challenge is that of hard disk power management in case of playing a movie. Partial buffering of movie files is done and the HDD is put to sleep mode before waking it up to buffer the next chunk. The challenge then is to predict accurately when to bring it back to the active state without affecting smooth playback of movie. In order to guarantee such high quality user experience, power saving schemes in multimedia applications need to be tightly coupled with the application itself. Hence, the challenge is to arrive at a sufficient level of abstraction for power management for the many application scenarios while trying to use the application specific information for power savings.

## III. UNIFIED POWER MANAGEMENT FRAMEWORK (UPMF)

UPMF defines a software architecture for handling power management of devices optimally for each of the application states. The architecture uses well-defined abstractions and strategies to achieve efficient power management. Fig. 1 indicates the architecture of a multimedia system with UPMF.

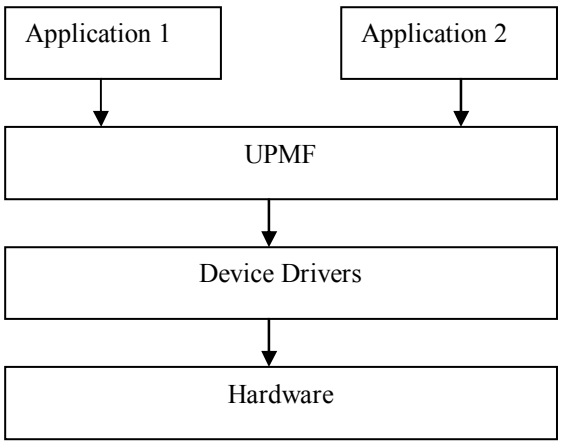


Figure 1 Architecture of Multimedia System with UPMF

### A. UPMF Classes

The UPMF Classes abstraction models the devices in the PMD into classes based on its usage pattern for any given application. The framework defines the following classes:
*Class A*: Multimedia specific components used only by multimedia applications with empirically determined usage patterns. Typical examples of such devices are image processing peripherals, capture and display peripherals like TV out and CCD/CMOS interface devices.
*Class B*: General components with empirically determined usage patterns - other non-multimedia applications will also use these components. Typical examples of such devices are memory devices like SDRAM, on chip RISC Core.
*Class C*: Components whose usage does not confirm to a particular predictable behavior pattern. Storage devices like hard disks typically fall into this category.
*Class D*: Components that need a user activity based power saving schedule. Devices like the LCD display fall into this category.

The criterion used for the above classification has been the nature of functioning of the devices themselves. To handle the differences in the functioning and usage of the devices, the power management framework handles each of these classes of devices independently. The framework expects applications to classify their components into one of the four classes described above.

### B. UPMF Classes Configurations

DVFS schemes are usually tied to OS tasks and are seldom determined by the applications making the arrangement sub optimal across application categories [8], [9] [10]. Under the UPMF framework the various classes are defined to have certain well-defined configurations that can be mapped onto a pre defined set of frequency and core voltage numbers. For Class A devices the framework requires that the application provide it with the optimal voltage and frequency for each of the components when the application is active. Since these devices are used exclusively by multimedia applications, the configuration settings can be used unchanged on the actual system. For Class B devices the framework expects two sets of configurations - one being the optimal configuration used by the multimedia application alone, and another being the minimal configuration guarantying desired level of performance under all applications. Classes C and D due to their nature of access will not have an optimal configuration and will be more dependent on access state.

### C. UPMF States

All devices classified under classes C and D would have well defined states as per the application requirements. Thus each of the applications would identify the defined states for each of the devices through a well-defined data structure as provided in the framework. A Class C device like HDD would have states of sleep, on, off. Similarly a Class D device like LCD would have states like on, off, 30% brightness, 50% brightness and so on.

### D. UPMF Strategies

The power management framework is an active entity that once configured, manages the power requirements of each application, and appropriately handles the target components. For each of the applications all the power aware/configurable devices are categorized into the several UPMF classes and their corresponding configurations and states are defined and registered with the UPMF. A simplified architecture diagram of the UPMF is provided in Fig. 3 below. The UPMF pre

defines these interactions with the applications with a set of well-defined API. Similar sets of well-defined APIs are defined for interaction between the framework and the devices. As shown in Fig. 3, applications send their state information that can be directly translated into minimum required active state for different devices for the particular application by UPMF using the device class and the appropriate state information. This state information includes the minimum possible frequency of operation, minimum supply voltage and module on/off condition. This information is then fed to power manger that has information about states supported by the devices. Power manager queries the devices through the device drivers to determine the power aware states that are supported by the devices. Power manger resolves the conflict between different applications and puts devices in maximum active state requested by the union of all concurrently running applications. Power manager module monitors the application and maintains the states of the devices and based on the power management strategy dynamically controls the states of the devices, as shown in Fig. 4, thereby ensuring optimal power consumption by each of the devices for a given application. As shown in Fig. 4 after resolving the conflicts in the application states, indicated by the state manager, for each of the devices the Power Manager uses a look up mechanism to set the device state. The look up mechanism is device class specific and uses either a <Voltage, Frequency>-tuple or a device state logic based on threshold or dependency criteria.

## IV. RESULTS, CONCLUSIONS, FUTURE WORK

The proposed framework has been tested on a battery operated Portable Media Player Recorder (PMPR) device. The power savings using the UPMF are tabulated in Table 1.

In this correspondence it was shown that a new and efficient UPMF is capable of providing exceptional power optimization for each of the application classes. The unique features of this new framework are: it works on statistically determined device configuration settings, works independently of the application and does not require run-time intervention, it maintains multimedia applications agnostic of whether certain devices are shared by other applications, and gracefully handles devices that support power optimizations as well as those that do not. With continuous improvement in dynamic power and performance range of new processors through advances in semiconductor technology coupled with newer more power hungry applications, the proposed framework will need to be further fine tuned to keep up with the growing power saving demands of applications.

## REFERENCES


[1] L. Benini, A. Bogliolo, and G. D. Micheli, "A Survey of Design Techniques for System-Level Dynamic Power Management," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 10(2), pages. 299–316, June 2000.

[2] K. Nowka *et al.*, "A 32-bit PowerPC Syetem-on-a-Chip with Support for Dynamic Voltage Scaling and Dynamic Frequency Scaling," *IEEE Journal of Solid-State Circuits*, vol. 37(11), pp. 1441–1447, Nov. 2002.

[3] K. Govil, E. Chan, and H. Wasserman, "Comparing Algorithms for Dynamic Speed-Setting of a Low-Power CPU," in *Proc. ACM Int'l Conf. on Mobile Computing and Networking*, 1995, pp. 13–25.

[4] T. Pering, T. Burd, and R. Brodersen, "The Simulation and Evaluation of Dynamic Voltage Scaling Algorithms," in *International Symposium on Low Power Electronics and Design (ISLPED)*, August 1998, pp. 76–81.

[5] Dirk Grunwald and Philip Levis and Keith I. Farkas and Charles B. Morrey III and Michael Neufeld, "Policies for Dynamic Clock Scheduling," in *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, October 2000.

[6] K. Flautner and T. Mudge, "Vertigo: Automatic Performance Setting for Linux," in *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, Dec. 2002.

[7] H. Zeng, C. S. Ellis, A. R. Lebeck, and A. Vahdat, "ECOSystem: Managing Energy as a First Class Operating System Resource," in *Tenth International Conference on Architectural Support for Programming Languages and Operating Systems*, October 2002.

[8] C. Poellabauer and K. Schwan, "Power-Aware Video Decoding using Real-Time Event Handlers," in *5th International Workshop on Wireless Mobile Multimedia (WoWMoM)*, September 2002.

[9] M. Weiser, B. Welch, A. Demers, and S. Shenker, "Scheduling for Reduced CPU Energy," in *First Symposium on Operating Systems Design and Implementation*, November 1994.

[10] D. Son, C. Yu, and H.-N. Kim, "Dynamic Voltage Scaling on MPEG Decoding," in *International Conference on Parallel and Distributed Systems (ICPADS)*, June 2001, pp. 633–640.

[11] B. Brock and K. Rajamani, "Dynamic Power Management for Embedded Systems", in *Procedings of the IEEE International SOC Conference*, September 2003.

[12] http://focus.ti.com/docs/solution/folders/print/267.html

[13] http://www.analog.com/en/app/0,3174,996%255F1163,00.html

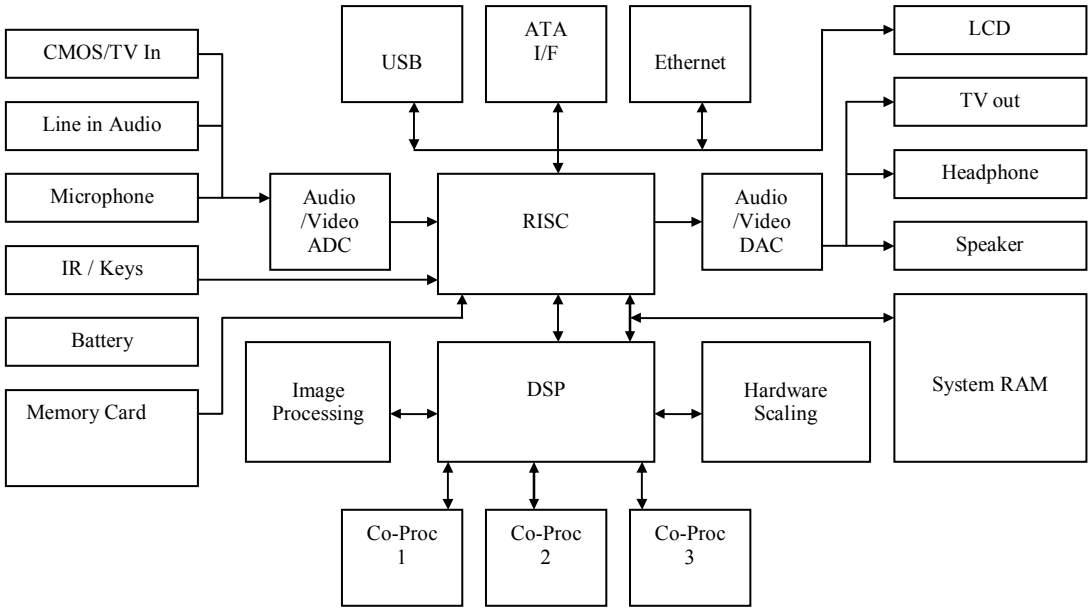


Figure 2: System Components of a Multimedia Device

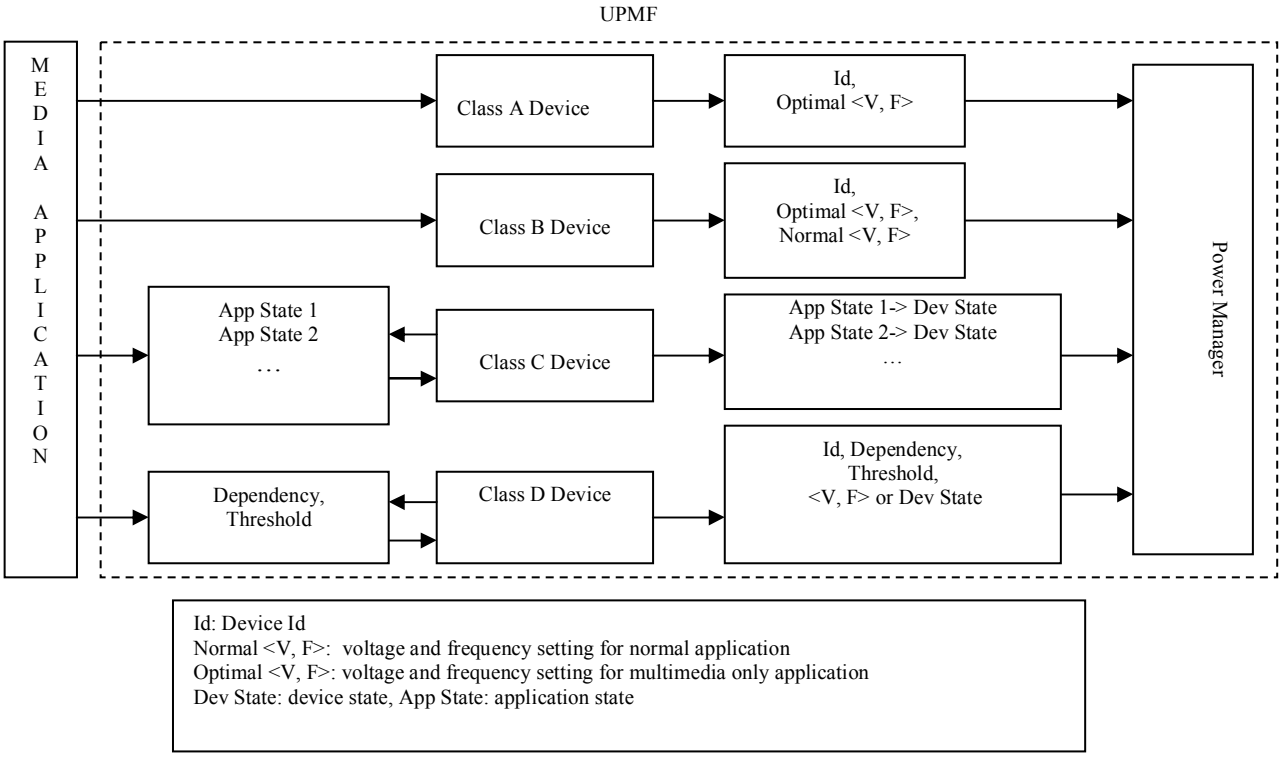


Figure 3 UPMF Architecture

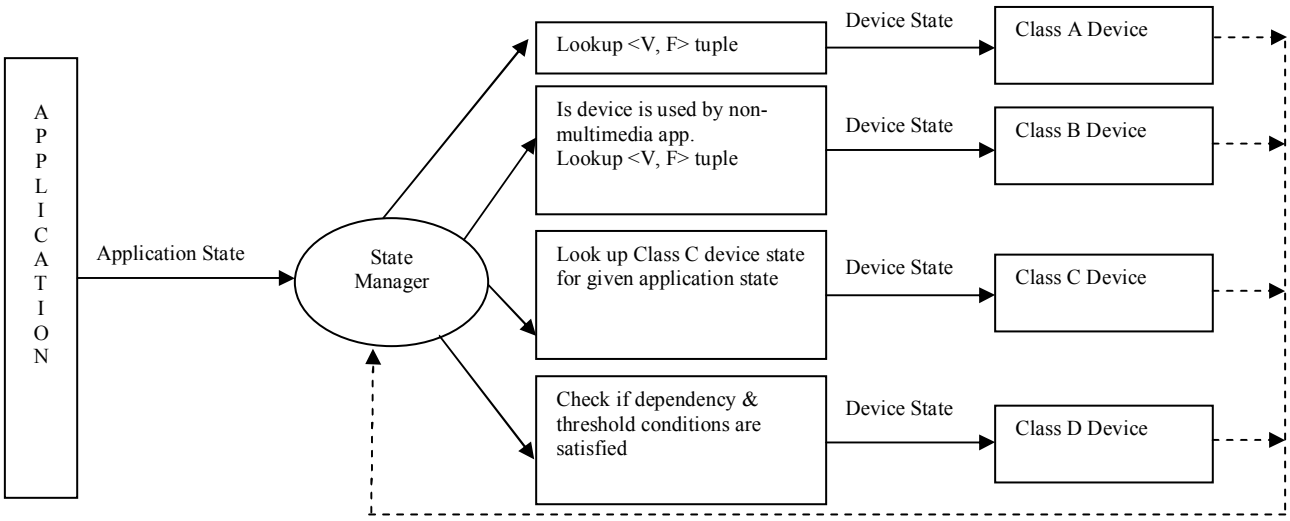


Figure 4 UPMF Control Flow

Table 1: Power savings (reduction in current) with UPMF for a Portable Multimedia Device

| **Application** | **Multimedia state** | **Without any power saving strategies (mA)** | **With UPMF (mA)** |
|---|---|---|---|
| GUI | LCD ON | 830 | 700 |
| | LCD OFF | NA | 610 |
| AV PLAYER | MOVIE SELECT-LCD ON | 1090 | 700 |
| | MOVIE SELECT-LCD OFF | NA | 630 |
| | MOVIE PLAY –HDD Wake UP | NA | 1300 |
| | MOVIE PLAY –HDD SLEEP | 1240 | 820 |
| | MOVIE PAUSE – LCD ON | 870 | 700 |
| | MOVIE PAUSE – LCD OFF | NA | 620 |
| AUDIO | SONG SELECT- LCD ON | 1080 | 700 |
| | SONG SELECT- LCD OFF | NA | 620 |
| | SONG PLAY – LCD ON | 1040 | 700 |
| | SONG PLAY – LCD OFF | NA | 620 |
| | SONG PAUSE – LCD ON | 1040 | 670 |
| | SONG PAUSE – LCD OFF | NA | 590 |
| IMAGE | IMAGE VIEW | 800 | 760 |
| IMAGE AND AUDIO | SONG SELECT – LCD ON | 1140 | 700 |
| | SONG SELECT – LCD OFF | NA | 620 |
| | SONG WITH PICTURE | 1020 - 1060 | 1010 |

NA – Not applicable state