Seo Y, Kim J, Seo E. Effectiveness analysis of DVFS and DPM in mobile devices. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 27(4): 781–790 July 2012. DOI 10.1007/s11390-012-1264-6

Effectiveness Analysis of DVFS and DPM in Mobile Devices

Youngbin Seo[1], Jeongki Kim[2], and Euiseong Seo[3]

[1]*Computer Science and Engineering Major, University of Science and Technology, 217 Gajeong-ro, Yuseong-gu, Daejeon 305-350, Korea*

[2]*Electronics and Communications Research Institute, 138 Gajengno, Yuseong-gu, Daejeon, Korea*

[3]*College of Information and Communication Engineering, Sungkyunkwan University, 2066 Seubu-ro, Jangan-gu Suwon 440-746, Korea*

E-mail: youngbin@ust.ac.kr; jkk@etri.re.kr; euiseong@skku.edu

Received September 11, 2011; revised May 8, 2012.

Abstract The demand for high-performance embedded processors in multimedia mobile electronics is growing and their power consumption thus increasingly threatens battery lifetime. It is usually believed that the dynamic voltage and frequency scaling (DVFS) feature saves significant energy by changing the performance levels of processors to match the performance demands of applications on the fly. However, because the energy efficiency of embedded processors is rapidly improving, the effectiveness of DVFS is expected to change. In this paper, we analyze the benefit of DVFS in state-of-the-art mobile embedded platforms in comparison to those in servers or PCs. To obtain a clearer view of the relationship between power and performance, we develop a measurement methodology that can synchronize time series for power consumption with those for processor utilization. The results show that DVFS hardly improves the energy efficiency of mobile multimedia electronics, and can even significantly worsen energy efficiency and performance in some cases. According to this observation, we suggest that power management for mobile electronics should concentrate on adaptive and intelligent power management for peripheral devices. As a preliminary design, we implement an adaptive network interface card (NIC) speed control that reduces power consumption by 10% when NIC is not heavily used. Our results provide valuable insights into the design of power management schemes for future mobile embedded systems.

Keywords dynamic voltage scaling, power management, battery management, energy efficiency, smart phone

1 Introduction

The functions of mobile multimedia consumer electronics such as portable media players, smart phones, and digital cameras are rapidly diversifying as a result of digital convergence. Furthermore, the resolution of images and videos as well as the complexity of games handled by mobile electronics are continually growing. Both concurrent execution of multiple applications and processing of high-definition audio, video and images require strong processor performance. Therefore, these trends are driving manufacturers to use powerful processors in the embedded systems of mobile multimedia electronics[1].

However, as the power consumption of a processor increases in proportion to both the clock frequency and the square of the operating voltage, stronger processors generally require more power. The processor is one of the major power consumers in mobile multimedia embedded systems[2]. Therefore, the demand for high-performance processors in mobile multimedia electronics poses a serious threat to battery lifetime. In fact, the battery life of cutting-edge smart phones is merely six hours when they run multimedia applications.

Dynamic voltage and frequency scaling (DVFS) is a feature that adjusts the clock frequency and operating voltage of a processor on the fly. When DVFS technology is applied, a processor reduces additional power consumption caused by unnecessarily high clock frequency when the system load is low, and therefore the processor achieves significant energy savings. DVFS effectiveness in both servers and PCs has been proved for many commercial products over a long time. Therefore, most modern processors employ the DVFS feature in one way or another.

However, because of rapid improvement in processor design and manufacturing technology, the amount of energy required to execute an instruction has

Regular Paper

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology under Grant Nos. 2009-0089491, 2010-0003453.

been steadily decreasing[3]. In addition, the fine-grained fabrication technology is decreasing the dynamic power consumption of processors. Consequently, the proportion of leakage power in the total power consumption is increasing[4]. Considering that a significant portion of DVFS benefit comes from a lowered operating voltage, a decrease in the proportion of dynamic power leads to diminishing potential benefit of DVFS according to Amdahl's law[5].

The energy efficiency improvement from DVFS comes at a cost. To preserve the quality of services, operating systems must determine the performance demands of applications accurately. As every application has its own performance demand and tolerance level to execution delays, prediction of the performance demand of applications is difficult. Moreover, as the performance demand of multimedia applications grows more burdensome, the penalty of inaccurate prediction worsens. Therefore, the effectiveness of DVFS in modern mobile devices should be quantitatively assessed to justify this cost or penalty in performance.

DVFS algorithms collect diverse parameters about system conditions and predict the performance demand of applications in execution. According to the prediction results, the embedded operating system initiates performance transitions.

When the DVFS algorithms underestimate performance demand, the service quality of applications with deadlines, such as multimedia applications and games, will be critically harmed by unexpected execution delays. On the contrary, overestimation of performance demand by DVFS algorithms will induce the use of excessively high frequency, which in turn generates large power consumption and therefore decreases the energy efficiency of the system.

Although most contemporary operating systems use variants of interval-based DVFS algorithms, their inaccuracy in performance decisions has been continually highlighted[6-12]. Therefore, considering the shrinking benefits and expanding side-effects, a careful evaluation of the value of DVFS in mobile electronics is in order.

Although some research efforts on the use of DVFS in mobile devices have been published, only a few of them evaluate DVFS in real mobile embedded systems. Moreover, there is little known about the differences in DVFS characteristics between mobile embedded systems and other computing systems. In this paper, we analyze the effectiveness of DVFS in mobile multimedia electronics and suggest a strategy for power management for mobile devices based on the results.

To analyze the power consumption patterns of short-term events, we suggest a novel synchronization methodology for processor power utilization. The suggested measurement technique can synchronize time series for power consumption and those for processor utilization within 10 ms precision. The amount of energy required to finish a certain application event can be measured using this scheme. To clarify the characteristics of DVFS in mobile devices, the effectiveness of DVFS in both embedded systems and diverse computing systems, including servers and laptops, is evaluated.

Finally, based on our observations we suggest a strategy for power management of mobile electronics that concentrates on the power consumption of peripheral devices. As a preliminary approach, we introduce an example of a power management scheme for a peripheral device.

The remainder of the paper is organized as follows. Section 2 describes the background and related work regarding the power consumption of processors, DVFS algorithms, and power management schemes for peripheral devices for mobile multimedia electronics. Section 3 introduces our processor utilization and power consumption measurement methodology. Then Section 4 analyzes the effectiveness and impacts of DVFS in mobile platforms in comparison to other computing systems, including servers and laptops. Based on the analysis, we suggest a strategy for power management for mobile multimedia electronics in Section 5. Finally, we conclude the paper in Section 6.

2 Background and Related Work

2.1 Processor Power Consumption

The power consumption, P, of a CMOS-based processor is related to its clock frequency f and operating voltage V as $P \propto V^2 \cdot f$. Also, the relation $V \propto f$ holds for these processors[13-14]. The dramatically increased power consumption caused by high clock frequency has threatened the battery lifetime in mobile multimedia electronics.

Although high performance is required in some cases, processors remain idle or under low load most of the time. Under low load or idleness, a high clock frequency results in unnecessarily additional power consumption. DVFS manages the tradeoff between performance and power consumption by adjusting the frequency and operating voltage on the fly.

A DVFS operation consists of two steps: a clock transition and a voltage transition. Previously, processors were halted during performance transition operations, which take up a few hundred μs. Recent processors perform transition operations following the procedure represented by Fig.1. When lowering performance, the frequency change occurs first. On the contrary,

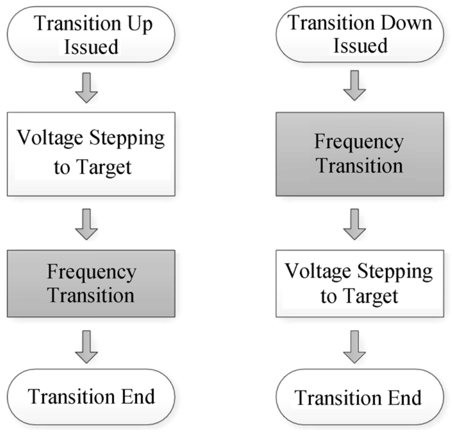


Fig.1. Internal operation sequences of processors during performance level transitions.

when raising performance, the voltage must be increased before increasing the clock speed. The processor needs to suspend its operation only for the frequency change, which takes less than 10 μs, in state-of-the-art processors[6].

The voltage transition requires approximately 100 μs. Therefore, even though there is no long suspension for processor operations, the entire transition process requires over 100 μs. However, in comparison to the length of time slices, which are the smallest time units of task scheduling, the transition time is negligibly short.

Currently, processor clock speeds have remained at the same level for a decade owing to the power wall, whereby the power consumption of a processor exceeds the limit allowed for thermal dissipation or power consumption of a system. Because of this situation, DVFS is one of the most effective means for reducing the power consumption and increasing the energy efficiency of a processor[13]. However, even server or desktop processors, for which both the total power consumption and the proportion of dynamic power to total power consumption are large, show greatly varying DVFS effectiveness, depending on the diverse workload and processor parameters[15]. Considering that both energy-per-instruction and the proportion of dynamic power to total power are small in mobile embedded processors in comparison to servers and PCs, DVFS effectiveness should be evaluated before it is utilized in mobile devices.

2.2 Existing DVFS Algorithms

When there is no deadline for an application, it is best to execute the application at the lowest clock frequency for the best energy efficiency. However, because multimedia and interactive applications have implicit or explicit deadlines, DVFS algorithms must provide sufficient performance for applications that are running.

In hard real-time systems, prediction accuracy is critical for system correctness[16-17]. Therefore, every task in a hard real-time system explicitly provides information on its performance demand and deadline. When this performance information is gathered for all applications in the system, prediction of the performance demand becomes a straightforward problem[18]. In spite of its accuracy, this approach is not used in commodity mobile electronics because existing applications must be modified to be served with proper performance.

Predicting future performance demands based on the past behavior of applications does not require any application modifications. Therefore, this is a viable option for soft real-time systems including most multimedia mobile electronics.

DVFS algorithms that make predictions based on history can be broadly categorized as interval-based algorithms or task-based algorithms[10]. Task-based algorithms[10-12,18-20] analyze the deadline and workload of each task, and then adjust the processor performance for each task based on the results. Interval-based algorithms[8,21-22] measure processor utilization in previous time intervals and use this information to set the current performance level for the next time interval.

In general, task-based algorithms save more energy and provide more suitable performance than interval-based algorithms. Task-based algorithms assume, however, that all tasks have explicit or implicit deadlines. As many applications do not have timeliness requirements, this assumption may cause meaningless deadline detection overheads in multimedia embedded systems. In addition, the runtime overhead and implementation complexity of task-based algorithms are generally higher than those of interval-based algorithms. Therefore, interval-based algorithms are more widely used in commercial products.

However, in spite of their popularity, interval-based algorithms frequently fail to predict performance demand, which can result in unexpected execution delays. Prediction failures for interval-based algorithms can be categorized into three typical cases: underestimations by averaging, late performance transitions, and inaccurate predictions[6].

Interval-based algorithms take the average processor utilization during the last whole interval as the prediction basis for the performance demand of the next interval. However, it ignores the timeliness requirements of applications. Suppose that a game requires 20 ms of execution at maximum performance to react to a command. Even though the required computation load is the same, providing only 2% of the maximum performance for 1 s may result in totally different user experiences because of an unexpected execution delay.

However, if the target processor can be set to a 2% performance level and the interval length is 1 s, an interval-based algorithm will provide such low performance during the next interval.

A second problem occurs because of the relatively long interval length. The length of an interval is set to be sufficiently long to obtain meaningful information about the system load and to prevent excessively frequent performance switching. In general, the interval duration lasts from tens of milliseconds to a few seconds. However, the activation intervals of multimedia or interactive applications are dramatically shorter. Therefore, interval-based algorithms occasionally provide the performance required a long time after it is first needed.

Finally, the performance demand prediction for the next interval, which can range in length from a few tens of milliseconds to a few seconds, shows poor hit rates. In particular, the prediction accuracy of interval-based algorithms is less than 50% for multimedia and interactive applications[7]. As stated previously, frequent erroneous predictions lead to a loss of quality of services or to energy waste, as shown in Fig.2.

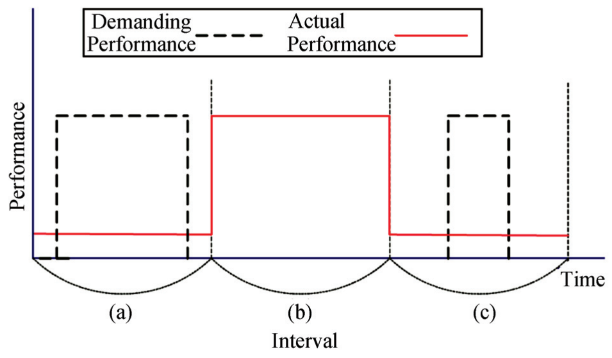


Fig.2. Performance demand prediction for interval (b) based on the history of interval (a) fails because the performance demand diminishes in interval (b). The prediction fails again in interval (c) since high performance demand spontaneously occurs in interval (b).

2.3 Power Management of Peripheral Devices

If processor power consumption is diminishing, then the power consumption of other components will proportionally increase in comparison to the power consumption of the entire system. Furthermore, as more diverse functionalities are integrated into mobile devices, various peripherals such as wireless network interface cards (NICs), cameras, GPS sensors, motion sensors, and ambient sensors, are included in mobile devices. Therefore, the contribution of peripheral devices to total power consumption will steadily increase[23].

Hardware vendors have been providing DPM (dynamic power management) features for the peripheral components of traditional computing systems, which make a component sleep while idle. However, as a peripheral device remains unavailable until it is fully awakened, the DPM feature may significantly harm the quality of services that require real-time responsiveness. Therefore, smart power management algorithms, for both processors and peripheral devices are necessary.

DPM algorithms that guarantee QoS in hard real-time systems have been suggested[24]. By identifying the first point at which a device is used after idling, DPM algorithms can determine whether performance transitions of the device violate task deadlines. They can also decide when to initiate the wake-up transition for the device.

Unlike hard real-time systems, most applications in a mobile device are sporadic tasks, for which both starting and execution times are difficult to predict. Therefore, it is challenging to apply the methods for hard real-time systems to consumer electronics. Consequently, a few studies have attempted to anticipate the use of a device based on its usage history[25-26].

In spite of the growing number of peripheral devices in mobile gadgets, there has been very little research on power management for peripherals in comparison to that for processors.

3 Measurement Methodology

3.1 Measurement Device

The power consumption of target systems was measured using a programmable DMM (digital multimeter). The DMM used in our experiments supports 3 000 readings/s. But because the basic scheduling length of embedded operating systems is approximately 10 ms, we programmed it to sample every 10 ms.

The DMM can measure both AC and DC current. Except the server system, which has an internal power supply, we measured the power input to the systems at the DC cables between the AC/DC adapters and the systems by cutting the cables and connecting the DMM to both ends of the cables in a series. When we measured the server system, we connected the DMM to the AC cable to the server.

3.2 CPU Utilization

Existing utilities that have been used to measure processor utilization samples at second granularity are not adequate for identifying performance demand changes for multimedia or interactive applications, for which activation intervals last for only a few tens of milliseconds. Therefore, we built our own tool to measure processor usage changes at millisecond granularity.

The */proc* file system, which contains information about task and system status, records the number of ticks used by each task and by the whole system. The CPU utilization measurement tool tracks changes in the numbers in every predefined time interval. As a result, the tool can monitor how much processor time was used for each application during the last time interval, as shown in Fig.3. The measurement interval here has nothing to do with the interval of interval-based algorithms.

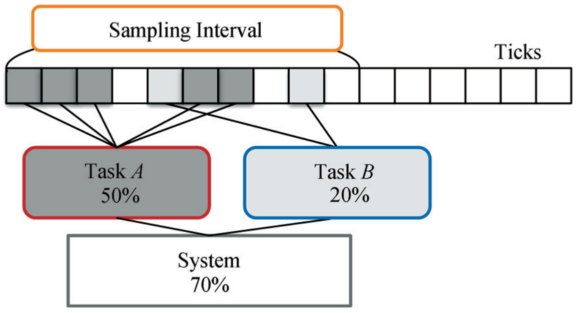


Fig.3. Processor utilization measurement tool collects tick-usage data for each task from the */proc* file system.

The accuracy and meaning of the measurement results depend on the measurement interval. Extremely short measurement intervals induce a significant processor overhead, and therefore may hinder the execution of target applications and increase power consumption. On the contrary, excessively long intervals scale down the sensitivity to changes in performance demand.

In this study, the measurement interval was set to 10 ms, which is the length of the minimum scheduling unit. When sampling every 10 ms, the tool adds less than 9% processor load. As the processor utilization data in this paper refer to processor usage by the overall system, the actual processor demand of a target application is less than the results presented here by approximately 9%.

When the overall processor utilization becomes saturated because of the additional performance demand of the measurement tool, the target application may not execute at its original speed. Therefore, the target workloads to be analyzed were carefully chosen to avoid saturation of processor utilization.

3.3 Power-CPU Synchronization

DVFS effectiveness seems to change with power consumption and therefore energy efficiency according to variations in performance demands. Consequently, either power consumption data or processor utilization data alone are insufficient for an understanding of the behavioral characteristics of a DVFS algorithm. Thus, it is necessary to display both types of data on a single plane as a function of time.

We first used the NTP (network time protocol) to synchronize the clocks of both the DMM host and target board to try to synchronize the data with timestamps from the clocks. However, even with a direct wired connection between the target board and the DMM host, significant differences as long as 0.1 s occurred between the clocks. Thus, we concluded that using NTP is not an adequate approach for our purpose.

We therefore developed a synchronization method that takes a power consumption signature as a milestone in time series data. The pseudocode illustrated in Fig.4 gives a sudden high load and decreases the load gradually. At the bottom point, it changes direction and increases the load again. Execution of the pseudocode generates a unique power consumption pattern that is "M"-shaped, as shown in Fig.5.

```
do_burst():
    conduct a lot of arithmetic operations
    return
do_sleep (timer):
    sleep for timer ms.
    return
Generate signature():
//increasing sleep interval
for (timer = 0; timer < 100; timer++) {
    do_burst();
    do_sleep (timer);
}
//decreasing sleep interval
for (timer = 100; timer > 0; timer++) {
    do_burst();
    do_sleep (timer);
    return
}
```

Fig.4. Power consumption signature generation pseudocode.

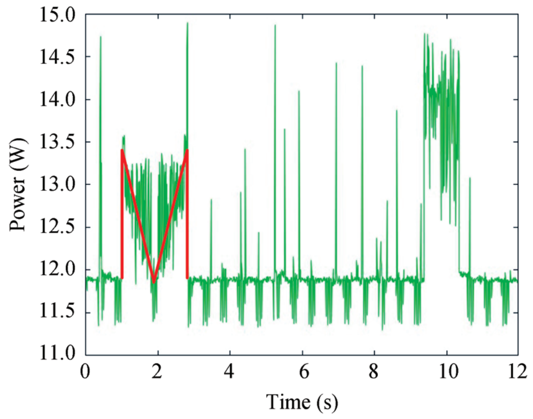


Fig.5. Power consumption signature, the "M" shape, appears at the 1 s point. It is used to synchronize the power measurement time series and processor usage time series.

Execution of the code induces an instant increase in power consumption and marks the "M" shape in the time series of the measurement result, so we know the exact starting time of the signature generation code and take this signature as the synchronization point between power measurement data and processor utilization data. In the analysis, we ran the signature generation code before every time we ran target applications.

After every experiment, we obtained two kinds of time series, which are for processor utilization and power consumption, respectively. Since it is easy to mark the time point of the signature on the processor utilization time series, we can correlate both time series together by unifying the signature starting points on them. The amount of energy consumption for accomplishing a specific event or activity can be calculated based on the combined time series.

We used this methodology in our research to measure the amount of energy used for specific activity.

4 Analysis of DVFS Effectiveness

4.1 Target Systems and Applications

We evaluated the DVFS characteristics in mobile platforms in comparison to other types of systems, including server and laptop platforms. Table 1 lists all the system specifications. All the systems were recently manufactured, off-the-shelf commercial products.

Table 1. Hardware Platform Specifications

Platform	Part	Description
EMB1	CPU	532 Mhz single core
	Memory	128 MB RAM
	Storage	Flash disk
	DVFS	533, 266 and 133 Mhz
EMB2	CPU	1.6 Ghz single core + Hyperthread
	Memory	128 MB RAM
	Storage	Flash disk
	DVFS	1600, 1333, 1067 and 800 Mhz
LAP	CPU	1.83 Ghz dual core
	Memory	1 GB RAM
	Storage	2.5 in. hard disk
	DVFS	1833, 1333 and 1000 Mhz
SVR	CPU	2.27 Ghz hyperthreaded quad core
	Memory	4 GB RAM
	Storage	3.5 in. hard disk
	DVFS	2268, 2000 and 1600 Mhz

As shown in Fig.6, both EMB1 and EMB2 were equipped with 3.5-inch and 7-inch touch screen LCDs, respectively. Each also had a sound chipset and a 1 Gbps wired NIC. LAP had a 15 inch LCD display with other peripherals, such as wireless and wired NICs and speakers. The power consumption of SVR excluded displays, keyboards and mice.

Because EMB1 has a different instruction set architecture (ISA), the software packages available differ from those for the other platforms. Therefore, direct comparison of the energy efficiency for a certain task between EMB1 and the other platforms may be unfair. Therefore, we excluded EMB1 from the comparative analysis and it was only used in the experiment to measure the processor contribution to the power consumption of the entire system.

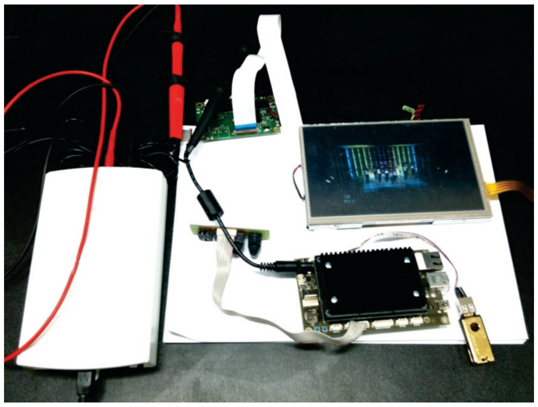


Fig.6. EMB1 board and digital multimeter.

We carefully selected target applications to reflect common workload characteristics of multimedia mobile electronics. The selected applications are listed in Table 2. To maintain consistency between experiments, we recorded and played the keyboard and mouse inputs for interactive applications, such as WEB and OFFICE, by using an input recording tool.

Table 2. Workload Description

Name	Description
HQMov	1280×720 10 Mbps MPEG4 movie file play
LQMov	352×288 500 Kbps MPEG4 movie file play
MUS	320 Kbps MP3 music file play
WEB	A popular web browser browsing a portal site
OFFICE	A word processor performing file open, editing, and word search

In order to analyze both the DVFS hardware issues and its effectiveness combined with a DVFS algorithm, the Ondemand algorithm[22] was used in our evaluation.

As shown in Fig.7, Ondemand is one of the popular interval-based algorithms, which is equipped and being widely used in the Linux kernel. It is a variant of the traditional interval-based algorithms that fits the fast performance transition speed of modern processors by using much shortened interval length, which can be set anywhere between a few hundred milliseconds and a few seconds depending on the power state transition latency of the processor. Ondemand also has a fast-up threshold that prevents excessively long response delay. If the CPU utilization of the last interval exceeds the fast-up threshold, the processor will immediately ramp up the clock speed to its maximum. To summarize, Ondemand conducts aggressive performance adjustments in comparison to conventional interval-based algorithms. In our research, the interval length was set to 80 ms as in the default configuration.

```
for every CPU in the system
  every $X$ milliseconds
    Get utilization since last check
    if (utilization > UP_THRESHOLD)
      Increase frequency to MAX
  every $Y$ milliseconds
    Get utilization since last check
    if (utilization < DOWN_THRESHOLD)
      Decrease frequency by 20%
```

Fig.7. Pseudocode of the Ondemand algorithm at a high level[22].

To measure the power consumption of EMB1, EMB2 and LAP, we obtained the current and voltage from the DC output of the corresponding power adaptors. The power consumptions of all systems were measured at the inputs to the power supply units. In addition, EMB1 was customized to expose the power input to the processor core, which is the main execution unit of a mobile embedded processor, so that a DMM directly measures the processor core power consumption.

4.2 Power Consumption Break-Down

To identify the processor power consumption as a proportion of that of the whole system, we measured both processor and system power consumption under diverse system conditions. Table 3 shows the results.

Table 3. Relative Power Consumption of the Processor Core in EMB1 in Comparison to Overall System Power Consumption

Clock (Mhz)	State	Processor Power (mW)	System Power (W)	Contribution (%)
532	Busy	244.3	4.59	5.3
266	Busy	109.3	4.32	2.5
133	Busy	59.8	4.27	1.4
532	Idle	15.4	4.17	0.4
266	Idle	9.7	4.16	0.2
133	Idle	9.3	4.15	0.2

The busy state in Table 3 means that the system was executing a dummy while-loop, so that the processor continually executed instructions. When the system was busy, the power consumed by the processor represented up to 5.3% of the total power. However, it was usually less than 4%. When the system was idle, the proportion decreased even further. We believe that this low idling power can be attributed to clock gating or similar power-saving techniques.

In addition, we can verify that the reduced power consumption of the processor due to the decreased clock speed barely affected system power consumption. The differences in contribution between the highest clock and lowest clock speeds were 3.9% and 0.2% under the busy state and idle state, respectively.

Considering the characteristics of modern mobile workloads, such as web page rendering and video decoding, a lower clock speed will undoubtedly directly affect the execution or response time of applications. As mentioned above, the overhead and possible performance risks induced by DVFS algorithms sometimes result in unexpected execution delays when they make incorrect predictions about future performance demands. When the clock speed was adjusted to 266 Mhz from the highest point, the power decreased by only 5.9%. Furthermore, when the clock speed was decreased to 133 Mhz from 266 Mhz, there was only a 1.2% power reduction. It is difficult to believe that this potential energy saving offsets the risks to QoS. As this difference becomes even more negligible when the system idles, it would be hard to justify the use of DVFS algorithms during idle time.

As a result, we conclude that, although the processor vendor provides three performance steps, the efficacy of DVFS is highly doubtful in EMB1 according to our data.

4.3 Energy Consumption

We measured the energy consumption of playing media files during a fixed time frame for HQMov, LQMov and MUS. For WEB and OFFICE, we executed a set of activities of each application in a fixed time frame. If the activity set finished before the end of the time frame, the system idled for the remaining time.

Fig.8 shows the energy consumption of SVR. Because the processor performance of SVR is a few times faster than that of LAP and EMB1, it idled most of time when it ran the workloads used in our study. Furthermore, the processor of SVR automatically goes to low-power sleep mode when it is idle and thus the power consumption of the processor has nothing to do with

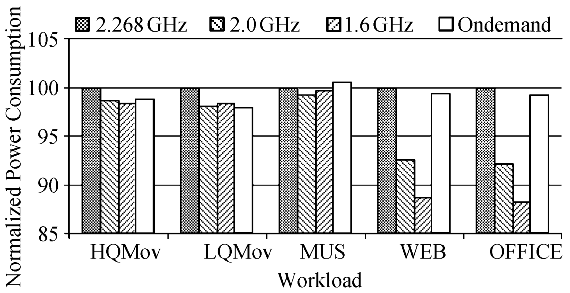


Fig.8. Energy consumption of SVR.

the clock speed when it is idling. Therefore, DVFS did not perform well under HQMov, LQMov and MUS, for which the processor was idling most of time.

However, because WEB and OFFICE require significant amounts of processing capacity in a short time, DVFS saved up to 12% of energy consumption for these workloads.

LAP consumed less power than SVR because a laptop usually consists of low-power components. Moreover, laptop processors are generally designed to achieve energy efficiency by fully utilizing DVFS. Consequently, LAP saved energy as much as approximately up to 15% of total energy consumption by using lower clock speeds or the DVFS algorithm as shown in Fig.9.

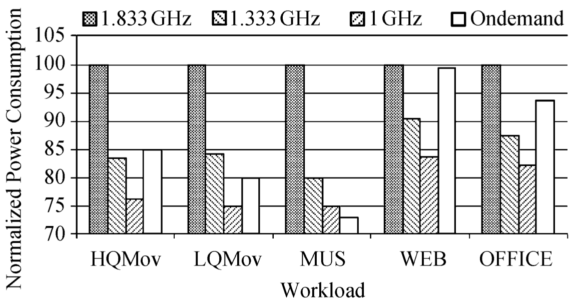


Fig.9. Energy consumption of LAP.

In contrast to SVR, the DVFS algorithm for LAP yielded less energy saving for both WEB and OFFICE than for the other workloads. This is because both WEB and OFFICE have high performance demands on LAP, and therefore the DVFS algorithm maintained the highest clock speed most of time.

In Subsection 4.2 we predicted that DVFS would be barely effective in EMB1 due to the low power consumption. However, because processors significantly contribute to the total power consumption in both SVR and LAP, despite variations, the efficacy of DVFS in both SVR and LAP was significant in our experiments.

Finally, Fig.10 shows the results of EMB2. As we verified in Subsection 4.2, embedded processors have extremely low-power consumption and therefore the effectiveness of DVFS is limited. We can find these phenomena also in Fig.10. The amount of energy saved using DVFS was less than 4% for every workload. The abnormally low energy consumption under HQMov at 0.8 Ghz is due to frame skipping for insufficient processor performance.

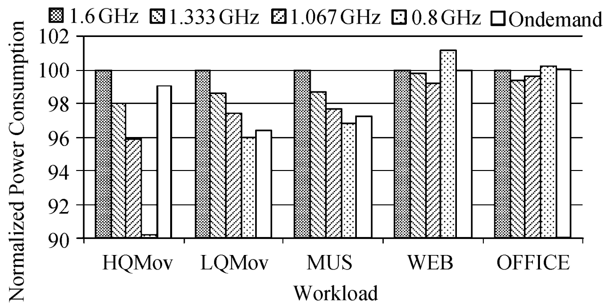


Fig.10. Energy consumption of EMB2.

Using 0.8 Ghz for both WEB and OFFICE worsened the energy efficiency. We believe that this is because the additional energy from prolonged use of the NIC or flash storage offsets the saved energy from lowered clock speed. In order to match the vast performance demand of both WEB and OFFICE workloads, DVFS used the highest clock speed whenever processing load was given to the system. Therefore, DVFS did not make any meaningful differences from the constantly high clock speed for both workloads.

5 Discussion and Suggestion

We found that DVFS saves only a small amount of energy, or sometimes even worsens the energy efficiency or execution time. In particular, the prolongation of execution time due to a decreased clock speed usually harms user experiences with mobile electronics more dramatically than other systems because the performance of mobile processors is generally marginal for most modern multimedia applications. Finally, unlike conventional computers, the power consumption of a processor accounts for less than 5% of the total power consumption. Therefore, it would be more beneficial to focus on the power management of peripheral devices.

Most embedded systems employ a system-wide power management policy, or some variant of the greedy approach[25] to manage the power states of peripherals. Under a system-wide power management policy, peripherals sleep simultaneously only when the system sleeps. Conversely, all peripherals remain awake while the system is on. The greedy approach puts a peripheral to sleep when the component idles for a larger amount of time than a predefined threshold.

The greedy approach can be applied to existing systems with only minor modifications of operating systems, and existing applications benefit in terms of energy efficiency without modification. However, because it does not anticipate the future use of peripheral devices, significant execution delays may occur as a result of power state transitions. In addition, as a peripheral must remain awake for a predefined threshold before going to sleep, it may lose energy that could be saved with smarter algorithms.

Therefore, like existing DVFS algorithms, DPM algorithms must be redesigned to reflect the characteristics of target applications, as well as other information, such as user behavior and ambient conditions, to proactively adjust the power states of peripherals.

As a preliminary study for the design of new DPM algorithms, we choose the wired NIC of EMB2 as the first target. The power consumption of an NIC increases with its operating speed[27]. It is not unusual for a portable media player or mobile Internet device to employ a wired NIC for the transfer of massive amounts of data. However, it is likely that its burst performance is rarely used.

We measured the power consumption of the overall system while changing the network speed. As illustrated in Table 4, there was a difference of up to 1.2 W, which is approximately 10% of the total power consumption, between the highest speed and the lowest speed. Based on this observation, we implemented an automatic performance setter that adjusts NIC according to the NIC utilization status. If the NIC is being heavily used, it is set to work at 1000 Mbps. Conversely, if the networking load is light, the NIC is set to its minimum speed. As expected, the prototype showed a significant enhancement of energy efficiency by up to 10%, which is approximately three times as much as the effectiveness of DVFS.

Table 4. Idle Power Consumption of the Overall Dystem for Different NIC Speeds

NIC Speed (Mbps)	Power (W)
1000	12.84
100	12.00
10	11.64

The LCD is another good example of the target peripherals for intelligent dynamic power management. The LCD equipped in EMB2 consumes about 5 W, which takes 40% of the total power consumption. Many commercial products now dim their LCDs to save energy under low light conditions or when the gadget is idle for a predefined threshold time.

6 Conclusions

The DVFS algorithm and the DVFS feature have been important contributors to improving energy efficiency in most computing systems. However, suspicion about the effectiveness of DVFS in mobile multimedia electronics, which can be equipped with many peripheral devices, has arisen since the emergence of low-power mobile processors. Although there have been many studies on efficient DVFS utilization, to the best of our knowledge this is the first study to quantitatively investigate the actual effectiveness of DVFS in mobile multimedia devices, including smart phones.

We proposed a novel measurement technique that can determine the energy required for handling a short event, such as opening a web page or searching for a word. Using the suggested measurement scheme, we analyzed DVFS effectiveness in diverse computing systems, including mobile embedded platforms. The results show that the power reduction due to DVFS is less than 5% in mobile platforms, and therefore, DVFS barely improves energy efficiency and can even sometimes significantly worsen both energy efficiency and performance.

From our analysis we can conclude that power management for mobile electronics should focus on the power consumption of peripheral devices. Based on the suggested strategy, we designed and implemented an automatic speed setter for a wired NIC, which consumes a significant amount of power because of its high frequency internal circuit. The prototype implementation saved up to 10% of the overall system power when the NIC was not being heavily used without any significant impact on performance.

We expect that the DVFS evaluation results and the measurement techniques suggested in this paper will be helpful for optimization of multimedia mobile electronics, for which energy efficiency is critical to product value.

References

[1] Park S, Jeong J, Jo H, Lee J, Seo E. Development of behavior-profilers for multimedia consumer electronics. *IEEE Transactions on Consumer Electronics*, 2009, 55(4): 1929-1935.

[2] Carroll A, Heiser G. An analysis of power consumption in a smartphone. In *Proc. the 2010 USENIX Conference on USENIX Annual Technical Conference*, Boston, Massachusetts, USA, June 23-25, 2010, p.21.

[3] Grochowski E, Annavaram M. Energy per instruction trends in Intel® microprocessors. *Technology@Intel Magazine*, http://support.intel.co.jp/pressroom/kits/core2duo/pdf/epi-trends-final2.pdf, Mar. 2006.

[4] Kim, N S, Austin T, Baauw D, Mudge T, Flautner K, Hu J S, Irwin M J, Kandemir M, Narayanan V. Leakage current: Moore's law meets static power. *IEEE Computer*, 2003, 36(12): 68-75.

[5] Amdahl G M. Validity of the single processor approach to achieving large scale computing capabilities. In *Proc. AFIPS Joint Computer Conferences*, April 18-20, 1967, pp.483-485.

[6] Seo E, Park S, Kim J, Lee J. TSB: A DVS algorithm with quick response for general purpose operating systems. *Systems Architecture*, 2008, 54(1-2): 1-14.

[7] Mun K, Kim D, Kim D, Park C. dDVS: An efficient dynamic voltage scaling algorithm based on the differential of CPU utilization. In *Lecture Notes in Computer Science* 3189, Yew P C, Xue J (eds.), Springer-Verlag, 2004, pp.160-169.

[8] Pering T, Burd T, Brodersen R W. The simulation and evaluation of dynamic voltage scaling algorithms. In *Proc. 1998 International Symposium on Low Power Electronics and Design*, Monterey, California, USA, Aug. 10-12, 1998, pp.76-81.

[9] Grunwald D, Morrey C B III, Levis P, Neufeld M, Farkas K I. Policies for dynamic clock scheduling. In *Proc. the 4th Symposium on Operating System Design and Implementation*, San Diego, California, USA, Oct. 23-25, 2000, pp.6-20.

[10] Lorch J R, Smith A J. Operating system modifications for task-based speed and voltage scheduling. In *Proc. the 1st*

International Conference on Mobile Systems, Applications, and Services, San Francisco, California, USA, May 5-8, 2003, pp.215-229.

[11] Flautner K, Mudge T. Vertigo: Automatic performance-setting for Linux. In *Proc. the 5th Symposium on Operating System Design and Implementation*, Boston, Massachusetts, USA, Dec. 9-11, 2002, pp.105-116.

[12] Choi K, Soma R, Pedram M. Fine-grained dynamic voltage and frequency scaling for precise energy and performance trade-off based on the ratio of off-chip access to on-chip computation times. In *Proc. Design, Automation and Test in Europe Conference and Exhibition*, Los Angeles, California, USA, Feb. 16-20, 2004, Vol.1, pp.4-9.

[13] Venkatachalam V, Franz M. Power reduction techniques for microprocessor systems. *ACM Computing Surveys*, 2005, 37(3): 195-237.

[14] Seo E, Jeong J, Park S, Lee J. Energy efficient scheduling of real-time tasks on multicore processors. *IEEE Transactions on Parallel and Distributed Systems*, 2008, 19(11): 1540-1552.

[15] Miyoshi A, Lefurgy C, Hensbergen E V, Rajamony R, Rajkumar R. Critical power slope: Understanding the runtime effects of frequency scaling. In *Proc. the 16th International Conference on Supercomputing*, New York, NY, USA, Jun. 22-26, 2002, pp.35-44.

[16] Liu C L, Layland J W. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of ACM*, 1973, 20(1): 46-61.

[17] Lehoczky J, Sha L, Ding Y. The rate monotonic scheduling algorithm: Exact characterization and average case behavior. In *Proc. the Real Time Systems Symposium*, Santa Monica, California, USA, Dec. 5-7, 1989, pp.166-171.

[18] Pillai P, Shin K G. Real-time dynamic voltage scaling for low-power embedded operating systems. In *Proc. the 18th ACM Symposium on Operating Systems Principles*, Banff, Alberta, Canada, Oct. 21-24, 2001, pp.89-102.

[19] Yang A, Song M. Aggressive dynamic voltage scaling for energy-aware video playback based on decoding time estimation. In *Proc. the 9th ACM International Conference on Embedded Software*, Grenoble, France, Oct. 11-16, 2009, pp.1-10.

[20] Bang S, Bang K, Yoon S, Chung E. Run-time adaptive workload estimation for dynamic voltage scaling. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2009, 28(9): 1334-1347.

[21] Weiser M, Welch B, Demers A, Shenker S. Scheduling for reduced CPU energy. In *Proc. the 1st USENIX Conference on Operating Systems Design and Implementation*, Monterey, California, USA, Nov. 14-17, 1994, Article No.2.

[22] Pallipadi V, Starikovskiy A. The ondemand governor: Past, present and future. In *Proc. Linux Symposium*, Ottawa, Ontario, Canada, July 19-22, 2006, Vol.2, pp.223-238.

[23] Celebican O, Rosing T S, Mooney V J III. Energy estimation of peripheral devices in embedded systems. In *Proc. the 14th ACM Great Lakes Symposium on VLSI*, Boston, Massachusetts, USA, Apr. 26-28, 2004, pp.430-435.

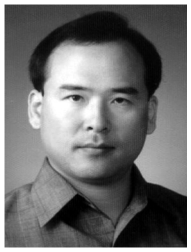
[24] Swaminathan V, Chakrabarty K. Pruning-based, energy-optimal, deterministic I/O device scheduling for hard real-time systems. *ACM Transactions on Embedded Computing Systems*, 2005, 4(1): 141-167.

[25] Qiu Q, Wu Q, Pedram M. OS-directed power management for mobile electronic systems. In *Proc. the 39th Power Source Conference*, Cherry Hill, New Jersey, USA, June 12-15, 2000, pp.506-509.

[26] Qiu Q, Wu Q, Pedram M. Dynamic power management in a mobile multimedia system with guaranteed quality-of-service. In *Proc. the 38th Annual Design Automation Conference*, Las Vegas, Nevada, USA, June 18-22, 2001, pp.834-839.

[27] Gunaratne C, Christensen K, Nordman B. Managing energy consumption costs in desktop PCs and LAN switches with proxying, split TCP connections, and scaling of link speed. *International Journal of Network Management*, 2005, 15(5): 297-310.

Youngbin Seo received his B.S. and M.S. degrees in electrical engineering from Ajou University. He is an engineer at LG U+ and conducting research projects on embedded systems software currently. He is also a Ph.D. candidate at University of Science and Technology, Korea. His research interest covers embedded operating systems, embedded system architectures and sensor networks.



Jeongki Kim received his B.S., M.S., and Ph.D. degrees in computer science from Chonbuk National University. Currently, he is a senior researcher at Electronics and Telecommunications Research Institute, Korea. His research area includes embedded system software, parallel processing, and database engineering.

Euiseong Seo received his B.S., M.S., and Ph.D. degrees in computer science from Korea Advanced Institute of Science and Technology (KAIST) in 2000, 2002, and 2007, respectively. He is currently an assistant professor in College of Information and Communication Engineering at Sungkyunkwan University, Korea. Before joining Sunkyunkwan University in 2012, he had been an assistant professor at Ulsan National Institute of Science and Technology (UNIST), Korea from 2009 to 2012, and a research associate at the Pennsylvania State University from 2007 to 2009. His research interests are in power-aware computing, real-time systems, embedded systems, and virtualization.